TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

THAPATHALI CAMPUS

A Major Project Final Report

On

Nepali To English Speech Translation With Prosody Preservation

Submitted By:

Pragyan Bhattarai (THA077BEI030)

Prashant Raj Bista (THA077BEI032)

Shakshi Kejriwal (THA077BEI044)

Sudipti Upreti (THA077BEI045)

Submitted To:

Department of Electronics and Computer Engineering

Thapathali Campus

Kathmandu, Nepal

May 2025

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

THAPATHALI CAMPUS

A Major Project Final Report

On

Nepali To English Speech Translation With Prosody Preservation

Submitted By:

Pragyan Bhattarai (THA077BEI030)

Prashant Raj Bista (THA077BEI032)

Shakshi Kejriwal (THA077BEI044)

Sudipti Upreti (THA077BEI045)

Submitted To:

Department of Electronics and Computer Engineering

Thapathali Campus

Kathmandu, Nepal

In partial fulfillment for the award of the Bachelor's Degree in Electronics, Communication and Information Engineering.

Under the Supervision of

Er. Kshetraphal Bohara

May 2025

DECLARATION

We hereby declare that the report of the project entitled "**Nepali To English Speech Translation With Prosody Preservation**" which is being submitted to the **Department of Electronics and Computer Engineering, IOE, Thapathali Campus**, in the partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in **Electronics, Communication and Information Engineering**, is a bonafide report of the work carried out by us. The materials contained in this report have not been submitted to any University or Institution for the award of any degree and we are the only author of this complete work and no sources other than the listed here have been used in this work.

Pragyan Bhattarai (THA077BEI030) _____

Prashant Raj Bista (THA077BEI032) _____

Shakshi Kejriwal (THA077BEI044) _____

Sudipti Upreti (THA077BEI045) _____

Date: May 2025

CERTIFICATE OF APPROVAL

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering, IOE, Thapathali Campus, a minor project work entitled **"Nepali To English Speech Translation With Prosody Preservation"** submitted by **Pragyan Bhattarai, Prashant Raj Bista, Shakshi Kejriwal**, and **Sudipti Upreti** in partial fulfillment for the award of Bachelor's Degree in Electronics, Communication and Information Engineering. The Project was carried out under special supervision and within the time frame prescribed by the syllabus.

We found the students to be hardworking, skilled, and ready to undertake any related work to their field of study and hence we recommend the award of partial fulfillment of a Bachelor's degree in Electronics, Communication, and Information Engineering.

_____

Project Supervisor

Er. Kshetraphal Bohara

Telecommunications Engineer, Nepal Telecom, Babarmahal, Kathmandu

_____

External Examiner

Er. Abhishek Koirala

AI Researcher, Fusemachines Nepal Pvt. Ltd.

_____

Project Coordinator

Er. Sudip Rana

Department of Electronics and Computer Engineering, Thapathali Campus

_____

Head of the Department,

Er. Umesh Kanta Ghimire

Department of Electronics and Computer Engineering, Thapathali Campus

May 2025

COPYRIGHT

ACKNOWLEDGEMENT

We would like to express our earnestness to the Institute of Engineering, Tribhuvan University, for including a Major Project as part of a Bachelor's degree in Electronics, Communication, and Information Engineering. This journey will be an incredible opportunity for us students to pursue and implement the knowledge we have garnered over the years related to different topics and concepts.

We, as a team, have strived to make the project better, successful, and feasible on all fronts. Under the supervision of **Er. Kshetraphal Bohara**, we have worked to address the project's limitations while understanding its concept and implementation. We are hopeful for the success of this project, titled "Nepali To English Speech Translation With Prosody Preservation."

We would like to extend our heartfelt acknowledgment to **Dr. Rishiram Sharma**, a renowned Nepali writer and teacher at Tribhuvan University, for granting us permission to utilize materials from his book to create our dataset. His contributions have been invaluable to the development of this project.

Likewise, we would also like to express our gratefulness to **Professor Dr. Dhruba Karki, HOD, Central Department of English, TU, Kirtipur,** for reviewing and approving our dataset for use. His efforts have been essential in driving this project's success.

Finally, we extend our sincere thanks to the Department of Electronics and Computer Engineering at Thapathali Campus for providing us with the necessary resources and support to complete this project.

Pragyan Bhattarai (THA077BEI030)

Prashant Raj Bista (THA077BEI032)

Shakshi Kejriwal (THA077BEI044)

Sudipti Upreti (THA077BEI045)

ABSTRACT

"Nepali To English Speech Translation With Prosody Preservation" is a Translation system that leverages Automatic Speech Recognition (ASR), Neural Machine Translation (NMT), and Text-to-Speech (TTS) Generation. The project has utilized State-of-the-art techniques in these respective areas. 'wav2vec 2.0' has been the guiding model for the conversion of Nepali speech to Nepali text. The mBART model has been used to translate the Nepali text into English text. A combination of datasets from Open Speech and Language Resources i.e. SLR143, SLR43, and common voice by Mozilla has been pre-processed and used for fine-tuning wav2vec 2.0. The dataset for NMT was created manually and used for the fine-tuning of the mBART. The fine-tuned wav2vec 2.0 model has achieved a word error rate of 0.44 on the validation set and 0.41 in the test dataset on 16 GB Nvidia P100 GPU. The NMT model has achieved a BLEU Score of 0.4006, ROUGE 1 score of 0.70, ROUGE 2 score of 0.49, ROUGE-L score of 0.66, SacreBLEU score of 38.84 and TER Score of 43 on the validation dataset. The parallel Nepali and English audio dataset was also created and was used for the training of FastSpeech 2, Speech T5-TTS and F5-TTS. F5-TTS has proven to give modest result among these models with average SIM of 0.77.

Keywords: FastSpeech 2, mBART, prosody prediction, speech-to-speech translation, wav2vec 2.0

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ASR	Automatic Speech Recognition
BART	Bidirectional and Auto-Regressive Transformer
BLEU	Bilingual Evaluation Understudy
BrE	British English
CER	Character Error Rate
CNN	Convolutional Neural Network
CTC	Connectionist Temporal Classification
DCT	Discrete Cosine Transform
DiT	Diffusion Transformer
DFT	Discrete Fourier Transform
F5	Fairytaler that Fakes Fluent and Faithful Speech with Flow Matching
FFT	Fast Fourier Transformer
G2P	Grapheme-to-Phoneme
GELU	Gaussian Error Linear Unit
GMM	Gaussian Mixture Models
GPU	Graphics Processing Unit
HMM	Hidden Markup Model
LSTM	Long Short-Term Memory
mBART	Multilingual Bidirectional and Auto-Regressive Transformer
MCD	Mel Cepstral Distortion
MFCC	Mel-Frequency Cepstral Coefficients
MMS	Massive Multilingual Speech
MOS	Mean Opinion Score
MSE	Mean Squared Error
MT	Machine Translation
NLP	Natural Language Processing

NMT	Neural Machine Translation
OSLR	Open Speech Language Resource
OOV	Out-of-Vocabulary
ORM	Object-Relational Mapping
ReLU	Rectified Linear Unit
SER	Speech Error Rate
S2T	Speech-to-Text
S2UT	Speech-to-Unit Translation
SIM	Speaker Similarity
TTS	Text-to-Speech
U2S	Unit-to-Speech
WER	Word Error Rate
XLSR	Cross-Lingual Speech Recognition

1. INTRODUCTION

Speech-to-Speech translation is a system that facilitates spoken language conversion from one language to another, enabling seamless cross-linguistic communication. This thesis focuses on developing a Nepali-to-English Speech Translation System with Prosody Preservation, which aims to preserve the emotional and contextual accuracy of speech in the target language. The system integrates speech processing techniques, machine learning models, and linguistic insights to ensure accurate and culturally sensitive translation. Given the increasing need for effective multilingual communication, this project aspires to contribute to the development of a robust translation system that enhances cross-cultural understanding. The system will have prospects in the field of intercommunication, particularly beneficial in fields such as schools, universities, international business meetings, and customer service centers.

One of the main crucial aspects of this system is its ability to predict prosody that conveys speaker's emotions. This unique approach contains the potential to revolutionize cross-language communication through the combination of advanced speech-processing techniques and machine-learning models. It authentically aims to resonate with English-speaking audiences. Introducing a diverse perspective aims to develop a system that bridges the gap between diverse linguistic and cultural landscapes. Regardless of the transformative tool, it boosts up understanding and harmony between them. With continuous progress, research, and development effort, there will be refinement in the system's accuracy, capabilities, and prosodic elements.

1.1 Background

In the field of cross-linguistic communication, speech translation has served its purpose as a premise since the mid-twentieth century. Most of the methods currently available have good accuracy regarding the translation task, and somehow there is less concern for speech characteristics such as intonation, pitch, intensity, etc. The non-concern of speech characteristics while translation acts as a lead to verbal misinterpretation or misunderstanding between the groups that are pre-divided by language.

Like any other branch of technology, the field of translation and inter-lingual communication is also constantly evolving. There have been numerous successful

efforts to narrow down the language barrier and create a more inclusive world with less verbal restriction. A few of the advancements are Machine Translation, Natural Language Processing, etc. These evolving technologies provide a better system with improved accuracy and more fluent translation. Meanwhile, the NLPs are widely used in sentiment analysis and general human language analysis.

Prosody is a component of any human language which evinces the naturalness of the voice. It is apt towards the suprasegmental elements of a sound such as intonation, pitch, rhythm, and stress. This component is essential in terms of flowing natural communication because it adds meaning and emotion to a speech. While defining individually, the stress corresponds to how any certain word or syllable is relatively prominent while speaking. Rhythm means how certain syllables have a pattern in terms of stress. Variation in the pitch while speaking is known as the intonation. Lastly, the speed of speech is referred to as tempo.

Continuous advancements in the field of “verbal” translation have given rise to the S2ST approach, which is textless for humans and based on discrete speech units. These approaches are heavily applicable in translating an “unwritten” or an “unspoken” language while maintaining the critical element in terms of communication, i.e., emotion. Since the Nepali language has its script which is “Devanagari”, and the “Latin” script stands as the base for the modern-day English language it will be an appealing approach to process the S2ST. This project aims to develop and improve the fluency as well as the accuracy of the speech-to-speech translation model while being influenced by the speaker's native prosodic features.

This project proposes to develop a Nepali-to-English speech translation system with a focus on predicting the prosody in the target language as same as the source language. Nepali has distinct phonetic and prosodic qualities inherent in its spontaneous speech patterns. However, when these characteristics are lost in translation, the product can be technically correct but emotionally bland and contextually misleading. As a result, preserving these prosodic characteristics is critical for ensuring that the translated speech accurately conveys the original speaker's intent and expression.

1.2 Motivation

The classic methods used for language translation cannot convey the full range of human voice. It fails to predict the prosodic features of voice such as intonation, pitch, natural rhythm, and stress, which leads to less effective interactions. There is a language gap between speakers of different languages and they face different challenges in their respective fields. Hence, to bring an effective translation system that can preserve prosody during the translation of language we propose this system, “Nepali To English Speech Translation With Prosody Preservation”. This project aims to create user user-friendly tool to enhance communication in different settings.

1.3 Problem Definition

Translation systems have made significant progress in translating text from one language to another, achieving high levels of accuracy in many language pairs. However, traditional text-based translation often fails to convey the nuances of spoken language, such as the prosodic feature of the speech including the rhythm, stress, and intonation of speech. For languages like Nepali, which have distinct phonetic and prosodic characteristics, translating not only the text but also predicting appropriate prosody in the target language (English) is crucial for applications in speech translation. Maintaining the original emotion of the speech is a crucial aspect of effective communication and interaction. Traditional methods lack natural flow and expressive qualities. Sometimes, it may fail to convey the intended message of the speaker. To enhance the authenticity and effectiveness of cross-language communication within different contexts, there should be advancement in the speech processing method with the enhancement of machine learning models to better understand prosodic elements and better linguistic translation.

1.4 Objectives

- To develop a Nepali-to-English speech-to-speech translation system with prosody preservation on the target language.

1.5 Project Scope and Application

This project enhances speech translation by not just converting language but also preserving the natural flow and emotions. These efforts has helped in making translations more accurate and expressive for real-world applications.

1.5.1 Project Scope

This project focuses on developing a Nepali-to-English speech translation system with prosody preservation. The primary goal is to preserve the emotional aspects, pitch variations, and intensity of the speaker's voice in the translated speech, ensuring a more natural and expressive output. To achieve this, the system will integrate state-of-the-art machine learning models for both translation and prosody preservation. These models will analyze and replicate key prosodic features such as intonation, stress, rhythm, and speech intensity, enhancing the overall accuracy and emotional expressiveness of the translated speech. By maintaining these elements, the system aims to improve the quality of cross-linguistic communication and create a more immersive and context-aware translation experience.

1.5.2 Project Limitations

The project is subjected to the following limitations:

- It only facilitates one-way translation i.e. from Nepali to English.
- It may require significant computational resources and affect performance on less powerful devices.

1.5.3 Project Applications

The project has been visualized to aid to verbal purposes that includes translation form Nepali language to its corresponding English, all while minimizing the gaps brought in by prosodic miscommunications. Thus, the project is applicable in following domains:

Educational Institutions and Universities

The project aims to significantly enhance educational and research collaboration among institutions globally. By leveraging advanced language learning features, including pronunciation and intonation guides, the application provides students with a

comprehensive tool to improve their English language skills. This not only supports academic achievement but also prepares students for international communication and collaboration, essential in today's interconnected world.

International Business and Corporate Settings

In corporate environments, the developed application serves as a pivotal tool in various training programs aimed at Nepali-speaking professionals. By offering accessible and tailored language learning modules, the platform contributes to enhancing workforce skills and professional growth opportunities. This not only improves internal communication but also enables seamless interaction with international clients and partners, fostering stronger business relationships and expanding market reach.

Tourism

For tourism the application plays a crucial role in improving communication with non-native speakers, particularly Nepali-speaking individuals. By providing translation and communication support, the platform enhances customer experience and satisfaction. It ensures smoother interactions, clearer explanations, and better service delivery, thereby boosting tourism engagement.

Government and Public Services

In the realm of government and public services, the application facilitates effective communication with Nepali-speaking citizens. It provides clear instructions, information, and guidance in their native language, ensuring inclusivity and accessibility. This capability is particularly critical during emergencies, public health crises, and community outreach programs where accurate information dissemination is paramount. By bridging language barriers, the platform promotes civic engagement, enhances public trust, and improves overall service delivery efficiency.

1.6 Report Organization

The whole report is divided into nine chapters. Each chapter discusses different sections of the whole project.

- The first chapter is the **Introduction** chapter. It gives a basic insight on what this project, "Nepali To English Speech Translation With Prosody Preservation" is about and a brief background on it alongside its scope and applications.
- The second chapter, **Literature Review** contains information about existing, proposed and previous works concerning Speech Translation Models and implementation on prosodies to match naturalness of the sound with references from various journals, articles and research papers.
- The chapter **Requirement Analysis** includes list of hardware and software requirements in order to carry out the whole project. Alongside, is the feasibility analysis of the project that dwells on how practical and legally acceptable the project is.
- In **Theoretical Background,** we have discussed about theories and concepts relevant to the project to lay a strong theoretical foundation.
- Following that is the section of **System Architecture and Methodology,** where the block diagram and working methodology of the project has been described.
- In the next chapter, **Implementation Details,** the methods for data collection, data preprocessing and model fine tuning has been described.
- The chapter **Result and Analysis** describes about the findings of the training process of various models and datasets.
- The next chapter is, **Future Enhancements**, talks about the probable enhancements
- The last chapter, **Conclusion**, gives a brief conclusion about the whole project.
- The remaining chapters contain additional information such as project budget, project timeline, analysis code snippets, phoneme tables etc. The references which were taken throughout the course of this project development are included at the end of the report.

2. LITERATURE REVIEW

Tacotron [1] is an innovative end-to-end generative TTS model that synthesizes speech directly from character input, eliminating the need for complex, multi-stage pipelines and hand-engineered features. Unlike traditional systems, Tacotron generates speech in chunks (frames) all at once, making it faster than sample-level autoregressive models. By leveraging a Seq2Seq framework with attention, a novel CBHG module, and frame-level prediction, Tacotron achieves a 3.82 MOS on US English, outperforming traditional parametric systems in naturalness. The Seq2Seq model, consisting of an encoder and a decoder, processes input characters into spectrogram frames, which are then converted to waveforms. The encoder uses a CBHG module to generate robust text representations, reducing overfitting and mispronunciations compared to standard multi-layer RNN encoders. This integrated approach allows for rich conditioning on attributes like speaker, language, or sentiment from the start, improving resilience and adaptability to new data. However, challenges remain, such as artifacts from the Griffin-Lim waveform synthesis, reliance on text normalization, and difficulties with long or complex sentences. Expanding Tacotron's capabilities for multi-speaker and multi-language synthesis, as well as developing more efficient neural vocoders, are key areas of future improvement. Despite these limitations, Tacotron has significantly advanced TTS technology by making speech synthesis more natural and accessible. Its ability to learn from large, diverse datasets while avoiding the errors of multi-stage models makes it a powerful tool in the field of artificial speech generation.

Shen et al. [2], describe how Tacotron 2 enhances Tacotron by integrating it with a modified WaveNet vocoder, achieving high-quality, natural-sounding speech. This end-to-end architecture processes raw text-to-speech waveforms without the need for extensive manual feature engineering. A recurrent sequence-to-sequence network with attention to converting text into Mel-spectrograms. A modified WaveNet vocoder to synthesize waveforms from these spectrograms. Tacotron 2 achieves nearly human-level speech quality with a MOS of 4.53, nearly matching human-recorded speech (MOS 4.58) and outperforming older. TTS systems and the original Tacotron. By using Mel-spectrograms to reduce WaveNet's complexity while maintaining audio quality, Tacotron 2 efficiently produces high-quality speech. This fully neural end-to-end model simplifies training and generates natural-sounding speech without extensive manual

feature engineering, marking a significant advancement in text-to-speech technology. However, certain challenges remain, such as occasional mispronunciations, unnatural prosody, and limitations in handling out-of-domain text. Future research will focus on optimizing meal frequency bin selection and enhancing robustness to diverse text inputs.

Ren et al. [5] introduced FastSpeech 2, an improved non-autoregressive TTS model that enhances the efficiency and quality of speech synthesis compared to its predecessor, FastSpeech. Unlike FastSpeech, which relies on a teacher-student distillation pipeline, FastSpeech 2 trains directly with ground truth targets, eliminating information loss and achieving 3× faster training speed. The model architecture consists of an encoder, variance adapter, and mel-spectrogram decoder. The encoder, composed of four FFT blocks, converts the phoneme embedding sequence into a hidden representation. The variance adapter incorporates duration, pitch, and energy predictors, improving expressiveness by addressing the one-to-many mapping problem in TTS. The mel-spectrogram decoder then transforms the adapted hidden sequence into mel spectrograms, which are subsequently converted into waveforms. FastSpeech 2 improves voice quality by conditioning speech synthesis on more accurate variance information, resulting in more natural and expressive speech. Additionally, the model enables fully end-to-end text-to-waveform synthesis, significantly reducing inference time. However, challenges such as occasional mispronunciations, unnatural prosody, and reliance on external tools for alignment and pitch extraction remain. Future research could focus on achieving fully end-to-end training without external dependencies and improving robustness across diverse datasets.

The paper by Zhou et al. [6] proposes an emotional style transfer framework based on the Variational Auto-encoding Wasserstein Generative Adversarial Network (VAW-GAN), making use of a pre-trained speech recognition model (SER). It addresses the challenge of emotional voice conversion and aims to transfer emotional prosody while preserving linguistic content and speaker identity. It introduces the emotional speech dataset (ESD) for multilingual and multi-speaker applications in speech synthesis and voice conversion. The dataset consists of 350 parallel utterances each with an average duration of 2.9 seconds and delivered by 10 native English and 10 native Mandarin speakers. And, for each group of language, there are 5 male and 5 female speakers with

5 different emotions: happy, sad, neutral, angry, and surprised. It is noted that it achieves better performance for neutral-to-happy conversion due to the poor performance of SER on happy (29.95%) compared with 84.32% on sad, and 70.47% on angry. For unseen emotion (anger), DeepEST still achieves comparable results with the baseline in terms of emotion similarity, which is very encouraging.

Jeuris and Niehus have created a German-English Speech-to-Speech Translation Corpus, namely LibriS2S [7]. Using this corpus, a Text-to-Speech (TTS) model based on FastSpeech 2 has been proposed in the paper that integrates source language information. The paper asserts that audio pairs of the same sentences in two languages are needed to build the proposed TTS system. So, the data collected by the authors include audiobooks from Librivox consisting of German speech, transcriptions, and their English translations of audiobooks. Yet there is no guarantee that the sentences in both languages are expressed with the same prosody when using audiobooks.

Khadka et al. [8] have used Tacotron2 for Mel spectrogram generation, and HiFiGAN and WaveGlow to synthesize speech from the generated Mel spectrogram. The paper provides insight into various data collection techniques as well. They collected data from three different sources: the OpenSLR dataset, self-recorded data, and YouTube audiobook clippings. According to their work, the OpenSLR dataset contains many repetitive phrases, which might not be ideal for speech conversion tasks. Since the OpenSLR dataset didn't meet their requirements, they recorded about 1.2 hours of audio clips with specific guidelines. For this purpose, they used publicly available resources such as the Nepali Essay collection, the Shwet Bhairabi e-book to enhance Nepali vocabulary, and Nepali newspaper sentence snippets for OOV words. To ensure consistency and quality, they fixed the distance between the recording device and the speaker, added consistent durations of silence before and after each recording, and controlled the volume levels at lower levels. However, using audiobook recordings from YouTube proved ineffective due to prominent background noise. The self-recorded dataset was not sufficient to fully meet the prosody requirements, although efforts were made to address this issue. The paper also presents a method for generating high-quality synthesized Nepali speech using the Tacotron2 model for mel-spectrogram generation, combined with HiFiGAN and WaveGlow vocoders for speech synthesis. The system achieved a high MOS of 4.03 for naturalness, marking the highest score for

Nepali TTS systems to date. Furthermore, the study uses FastSpeech 2 for mel-spectrogram prediction and integrates prosodic features, significantly improving speech naturalness. The study also highlights the importance of dataset collection and cleaning, which was carried out using Python, and the combination of these techniques resulted in a highly natural-sounding speech output.

The paper by Akarsh et al. [9] makes a significant contribution to SSMT by introducing a stress-annotated dataset for Indian English and developing a system that effectively transfers stress into the target Hindi speech. By training a stress detection model on manually annotated data and integrating it into a modified FastPitch-based TTS system with a Pitch-Duration-Energy Modifier block, the study enhances the naturalness and expressiveness of translated speech. However, several challenges were encountered, including the lack of prosodic annotations, data imbalance in stressed regions, difficulties in word alignment between English and Hindi, and instability in modifying pitch and energy variations in TTS. These were addressed through Fleiss Kappa inter-annotator agreement, SMOTE-based data balancing, multilingual BERT-based word alignment, and careful variance modification in TTS. The proposed system achieved promising results, with improved stress detection accuracy and a st-MOS of 4.09, indicating effective stress integration. Future enhancements include exploring alternative TTS architectures, preserving speaker identity in translated speech, and developing automated metrics for evaluating prosody transfer accuracy. Mathematically, the study leverages concepts like Fleiss Kappa for annotation agreement, SMOTE for handling data imbalance, MSE loss for optimizing mel-spectrogram predictions, and acoustic feature extraction (MFCC, SDC, F0, Energy) for stress detection. This research establishes a strong foundation for expressive and natural SSMT, bridging the gap between linguistic meaning and prosodic accuracy in speech translation.

Dong et al. proposed PolyVoice [10], a novel S2ST framework utilizing language models for direct translation and speech synthesis. Unlike conventional encoder-decoder architectures, PolyVoice employs two decoder-only language models: one for S2UT and another for unit-to-speech synthesis. The system leverages discretized speech units extracted through self-supervised learning (e.g., HuBERT), enabling translation for both written and unwritten languages. A key advantage of this approach

is its ability to preserve the speaker's voice and speaking style across translations, using VALL-E X-based synthesis techniques. However, challenges include the quality of semantic unit extraction, which affects translation accuracy, data scarcity, especially for low-resource languages, and the difficulty in retaining speaker characteristics during speech synthesis. The experimental results demonstrate improvements in naturalness and speaker similarity over previous models, though translation accuracy (ASR-BLEU) still lags slightly due to the limitations of unit-based modeling. Future enhancements involve improving semantic unit extraction, scaling the model with larger datasets, and refining zero-shot learning capabilities for low-resource languages. Mathematically, the framework incorporates discrete unit clustering (k-means for HuBERT units), transformer-based sequence modeling, and speaker similarity scoring (ASV models) for evaluation. PolyVoice represents a significant step toward realistic and expressive S2ST, bridging the gap between linguistic meaning and natural prosody in translated speech.

The study by Tathe et al. [11] addresses the Hindi-to-English speech conversion framework, integrating XLSR wav2Vec 2.0 for ASR, mBART for NMT, and Bark for TTS synthesis. The system enhances Hindi speech recognition accuracy by fine-tuning the XLSR wav2Vec 2.0 model on the Common Voice Hindi dataset, overcoming challenges posed by low-resource ASR for Hindi. The mBART model ensures high-quality Hindi-to-English translation, leveraging its multilingual capabilities. Finally, Bark, a transformer-based TTS system, generates natural-sounding English speech, offering high clarity and expressiveness. The key challenges in this research include limited Hindi speech data, requiring model fine-tuning, maintaining translation fluency across different sentence structures, and handling computational complexity when integrating these large models. The proposed system achieves improved speech recognition accuracy (WER 0.428) and high-quality synthesized English speech, making it useful for applications such as real-time translation, audiobook creation, and assistive technologies. Mathematically, the study employs self-supervised learning for ASR, transformer-based sequence modeling for NMT, and attention-based deep learning for speech synthesis. Additionally, hyperparameter tuning was conducted, optimizing learning rates, weight decay, and training epochs for better model performance. This research provides a scalable and effective pipeline for cross-lingual speech conversion, advancing the field of multimodal AI-driven communication. The

study did not provide detailed results or evaluation metrics, underscoring the need for comprehensive assessments in future research to better understand the models' effectiveness.

The paper by Ghimire et al. [12] provides a comprehensive review of Nepali Automatic Speech Recognition systems, highlighting advancements, challenges, and future directions. It discusses various datasets used in Nepali ASR research, emphasizing the scarcity of high-quality, diverse, and publicly available speech corpora. The study reviews traditional ASR models, such as HMM and GMM, alongside deep learning-based E2E models, including CNN, RNN, LSTM, GRU, ResNet, and Transformer architectures. The model combining CTC and the Attention Model got the best CER of 10.3% on the OSLR54 dataset. Despite notable progress, several challenges persist, including the lack of high-quality datasets, limited computational resources, inefficient language models, and difficulties in handling multilingual code-switching. Future research should focus on improving dataset quality through data augmentation and collection of spontaneous speech, developing advanced transformer-based language models, and optimizing ASR architectures like Conformer and wav2vec. Additionally, real-world applications of Nepali ASR, such as voice assistants and customer service automation, require further exploration. Mathematically, the study employs techniques like HMM for probabilistic modeling, MFCC for feature extraction, and CTC for aligning speech inputs with textual outputs. By addressing these challenges and leveraging modern neural network architectures, Nepali ASR systems can achieve improved performance and broader applicability in language technology.

The tiny paper by Hira et al. [14], presents CrossVoice as a cascade-based S2ST system employing Faster-Whisper for ASR, Google's NMT model for MT, and the Massive Multilingual Speech (MMS) model based on VITS-TTS (Variational Inference with adversarial learning for end-to-end Text-to-Speech) for TTS. The system uses transfer learning on a voice cloning module for prosody preservation. The pre-trained speaker encoder generates X-vector embeddings which is coupled with FreeVC's voice conversion module to transfer speaker prosody. The models are trained on several open source datasets such as IndicTTS-en, VoxPopuli-French, VoxPopuli-Spanish, VoxPopuli-German, and others. Although CrossVoice has obtained a MOS of 3.75 out

of 4, the system encounters challenges in accurately transferring prosodic features like intonation and stress patterns across languages.

The paper by Chen et al. [19], introduces F5-TTS, a fully non-autoregressive TTS system leveraging Flow Matching and DiT for highly natural and expressive speech synthesis. Unlike traditional TTS models that rely on phoneme alignment, duration prediction, and text encoders, F5-TTS simplifies the process by padding text input with filler tokens and directly denoising speech, enabling zero-shot generation and seamless code-switching between languages. The model incorporates ConvNeXt for refined text representation and proposes a novel Sway Sampling strategy, which improves inference efficiency and speech naturalness without requiring retraining. The challenges faced include slow convergence and alignment failures observed in previous non-autoregressive models, which hinder robustness. Additionally, balancing speech naturalness and speaker similarity while maintaining fast inference posed a significant obstacle. The outcomes demonstrate state-of-the-art performance, achieving an inference RTF of 0.15, which is significantly faster than existing diffusion-based TTS models while maintaining high WER accuracy and speaker similarity. The future enhancements aim to further improve model efficiency, optimize speech alignment, and integrate automated evaluation metrics for robust real-world applications. Mathematically, Flow Matching loss is used to guide speech generation, optimizing a probability path between a Gaussian prior and real speech data. The model also employs Conditional Flow Matching with Optimal Transport formulation to improve alignment. Additionally, CFG is utilized to balance fidelity and diversity in synthesized speech. This research advances non-autoregressive TTS systems by offering a faster, more robust, and expressive speech synthesis approach, setting a new benchmark for zero-shot multilingual text-to-speech generation.

3. REQUIREMENT ANALYSIS

This section includes project requirements with feasibility analysis of the project.

3.1 Project Requirements

This section mainly focuses on hardware and software requirements for the project.

3.1.1 Hardware Requirement

Efficient deep learning requires powerful hardware to handle complex computations and speed up processing.

3.1.1.1 GPU

Deep learning models are computationally intensive and require substantial processing power for efficient training and optimization. High-performance GPUs play a crucial role in accelerating these tasks due to their parallel processing capabilities, which significantly outperform traditional CPU for deep learning workloads. GPUs are specifically designed to handle large-scale matrix operations and tensor computations, which are fundamental to deep learning algorithms. Their architecture enables massive parallelism, allowing thousands of cores to execute computations simultaneously. This parallelism is particularly beneficial for training deep neural networks, where large datasets and complex models necessitate extensive numerical computations. In this project, the use of high-performance GPUs is essential to ensure the efficient training and optimization of deep learning models. Leveraging GPU acceleration allows for the development of more sophisticated models while maintaining practical computational feasibility.

3.1.2 Software Requirement

Likewise, various software tools have been integrated to ensure smooth development and bring the project to life.

3.1.2.1 VS Code

It is a commonly used code editor for a range of programming languages and tools through its extension. It is suitable for this project as it involves multiple programming

languages and frameworks. It provides facilities such as extensions for Python, Django, debugging, Git integration, and many more.

3.1.2.2 Python

It is the major programming language for this project. It provides readability and broad library support in data processing. It includes different libraries such as NumPy, Pandas, and more, important for diverse task handling in ASR, NMT, and TTS components. Due to its simple syntax and strong community support facilitates rapid development and problem solving.

3.1.2.3 Librosa

This is one of the dedicated Python libraries for analyzing and processing audio signals and is extensively used in the ASR module. It enables the preprocessing of speech data and extracts necessary features for speech recognition by providing efficient audio loading and feature extraction. The speech data is accurately transformed to a suitable format through this functionality.

3.1.2.4 Pytorch

It is an open-source machine-learning library that offers flexible and strong tools for deep learning. It enables the training of complex neural networks necessary for the ASR (wav2vec 2.0), NMT (mBART), and TTS (FastSpeech 2) models. Its wide library support makes it the preferred framework for the project & machine-learning needs.

3.1.2.5 Matplotlib

It is a Python library that allows to creation of static, animated, and interactive representation that offers a variety of plot types and customization possibilities as well as seamless integration with other Python libraries. Here it will be used to visualize data, provide clear insights, track training progress, and facilitate continual improvement.

3.1.2.6 Hugging Face

It is a library with a collection of pre-trained models and tools mainly designed for natural language processing tasks. It supports a wide range of transformer-based models

and provides an API that works well with PyTorch. Here, the transformer library is used for the NMT component, mainly for the mBart model for the Nepali text-to-English translation using modern NLP approaches.

3.1.2.7 Streamlit

Streamlit is a Python library built to create intuitive and interactive web applications with minimal effort. It enables the seamless development of the translation system's frontend by offering an easy-to-use interface, customizable widgets, and live data updates. With its focus on simplicity and rapid prototyping, Streamlit makes it effortless to visualize data and build scalable applications for engaging user experiences.

3.1.2.8 Google Colab

It offers powerful GPUs like the Nvidia Tesla T4 with 16 GB of memory, RAM 12 GB is ideal for activities that require quick model training and optimization, such as FastSpeech 2 TTS. It provides free access to GPU and integrates with Google Drive.

3.1.2.9 Kaggle

On Kaggle, Nvidia Tesla T4 x2 GPUs, and Nvidia Tesla P100 GPUs are limited to 30 hours per week or 9 hours per session, and TPU VM-2 instances are restricted to a certain 20 hours per week or 3 hours per session with a RAM of 29 GB per session and maximum session duration of 12 hours. Kaggle provides free GPU and TPU resources, with access to diverse datasets directly integrated with notebooks, and a vibrant and supportive community, making it an ideal platform for a collaborative development environment.

3.2 Feasibility Analysis

In a feasibility study, a proposed plan or project is evaluated for its practicality. As part of a feasibility study, a project is evaluated for its viability in order to determine whether it will be successful. This evaluation helps identify potential risks, resources required, and the overall likelihood of achieving the project's goals. Additionally, it ensures that the project aligns with available technical, financial, and operational resources. We have checked the feasibility on the following basis:

3.2.1 Technical Analysis

The project “Nepali to English Speech Translation with Prosody Preservation” focuses on the integration of technologies like ASR, NMT, and TTS. Google Colab, Kaggle and Lightning AI offers high-end GPUs. In case of a greater amount of data, cloud-based services or more powerful local GPUs has been used. The available local GPUs are NVIDIA GeForce RTX 3060 (6GB of dedicated GPU memory) and NVIDIA GeForce GTX 1650 (4GB of dedicated GPU memory). This has helped strike a good compromise between performance and cost for the proposed project. The NVIDIA GeForce MX230's small power (2GB of dedicated GPU memory) makes it just enough for fundamental testing and inference. The resources have been made freely available with some technical support. Hence, the project can be regarded as technically feasible.

3.2.2 Economic Analysis

The project does not include any hardware components so the project is economical however some costs has been incurred during the process of data collection, printing reports, and premium subscription for GPU usage.

3.2.3 Operational Analysis

The developed system does not require a complex environment for operation. It can be operated on a laptop with an internet connection to communicate with the pre-trained models in the cloud.

3.2.4 Legal Analysis

The collected data remained confidential and was not disclosed publicly. User privacy was strictly maintained and respected throughout the project. Furthermore, the project fully complies with the National ICT Policy of Nepal (2015) and adheres to all relevant legal and ethical guidelines, ensuring its legal feasibility.

4. THEORETICAL BACKGROUND

This section provides an overview of the theoretical background relevant to the project. It covers fundamental concepts, principles, and methodologies. Additionally, it explores key theories, frameworks, and prior studies that contribute to understanding the project's objectives and implementation.

4.1 Audio and Audio Sampling

Audio is in the form of sound waves whose vibrations travel through a medium like air, water or solid materials. These vibrations generate continuous analog waveforms, which must be converted into digital format for processing, storage, and transmission in computational systems. The digital representation of audio involves encoding these waveforms into discrete numerical values, making it possible to manipulate and analyze sound efficiently using digital signal processing techniques. Various digital audio formats, such as MP3, WAV, and AAC, are widely used for storage and playback, each offering different levels of compression and quality.

One of the fundamental processes in digital audio representation is audio sampling, which involves converting continuous analog signals into discrete digital signals. Since computers and most digital systems can only process discrete numerical data, analog signals must be sampled at regular intervals to create a digital representation. The sampling rate, measured in Hertz (Hz), refers to the number of samples taken per second. According to the Nyquist-Shannon sampling theorem, the minimum required sampling rate to accurately reconstruct an analog signal is at least twice the highest frequency present in the signal.

$$f_s \geq 2f_m \tag{4-1}$$

Human hearing typically ranges from 20 Hz to 20 kHz; however, most of the essential information in human speech is concentrated below 8 kHz. A sampling rate of 16 kHz is commonly used in speech processing applications, as it effectively captures frequencies up to 8 kHz, ensuring intelligibility while optimizing data storage and computational efficiency. High-fidelity audio applications, such as music production, often use sampling rates of 44.1 kHz or 48 kHz to preserve greater sound detail.

4.2 Fundamental Parameters of Sound Waves

The three fundamental parameters of sound waves time period, frequency, and amplitude play a critical role in the development of speech translation systems, particularly in prosody prediction and natural speech synthesis. These parameters influence the rhythm, pitch, and intensity of speech, all of which are essential for preserving the natural flow and expressiveness of spoken language across different speakers and languages.

The time period of a sound wave refers to the duration required for one complete cycle of the wave to pass a given point. It is measured in seconds and is inversely proportional to frequency, which represents the number of cycles per second and is measured in Hertz. A longer time period corresponds to a lower frequency, producing a deeper voice, whereas a shorter time period results in a higher frequency and a higher-pitched voice. This relationship is particularly significant in speech translation systems, as variations in time period and frequency must be accurately modeled to maintain the natural pitch and rhythm of speech. For example, male voices typically have longer time periods and lower frequencies (ranging between 85 Hz and 180 Hz), whereas female and children's voices have shorter time periods and higher frequencies (typically between 165 Hz and 255 Hz for females). Accurately capturing these variations ensures that translated speech retains its natural characteristics, making it more intelligible and expressive.

Amplitude determines the loudness or intensity of a sound wave and is measured in decibels (dB). It represents the maximum displacement of a sound wave from its equilibrium position, where a higher amplitude corresponds to a louder sound and a lower amplitude results in a softer sound. In speech, amplitude variations reflect vocal intensity, contributing to the emotional and prosodic features of speech. For instance, louder amplitudes indicate emphasis, excitement, or urgency (as in shouting or singing), while softer amplitudes represent calmness, hesitation, or subtlety (as in whispering or soft-spoken speech). In speech translation systems, preserving amplitude variations is crucial for accurately conveying emotions, stress, and emphasis in different languages. Ensuring that amplitude changes are effectively modeled enhances the naturalness and

expressiveness of synthesized speech, making it more engaging and contextually appropriate for users.

Frequency acts on a sound wave to determine, among other things, one of the most defining features of voice pitch. High-frequency sound waves, caused by faster vibrations of the vocal cords, produce higher-pitched voices, such as those of children or soprano singers. Low-frequency waves result in deeper voices from their slower vocal cord vibrations and therefore may be found in adult males or bass singers. Frequency variation plays an important role in prosody, which is the rhythm, stress, and intonation patterns of speech. Correct pitch modeling is necessary for the speech translation system to produce natural-sounding speech outputs. For example, in many languages, a question involves a rising pitch at the end of a sentence, while a statement may have a steady or falling pitch. By carefully modeling these changes in frequency, speech translation systems can convey such differences in translated speech.

$$Frequency\ (f) = \frac{1}{T} \tag{4-2}$$

The time period, amplitude, and frequency all interact, where understanding is key to maintaining prosody for speech naturalness and intelligibility. Prosody will not only affect the pitch but also the rhythm and emotional tone of speech, all necessary to make sure translation and synthesis are accurately conveyed. For instance, advanced models, such as FastSpeech 2, while being popular for speech synthesis, predict mel-spectrograms, incorporating prosodic features that ensure the output speech aligns with variations in natural pitch, duration, and intensity.

4.3 Fundamental and Dominant Frequency in Speech Processing

The fundamental frequency is one of the important concepts of speech processing. It is often denoted by F_0 or f_0. It is defined as the average number of oscillations per second our vocal cords vibrate while generating voiced sounds. It is expressed in Hertz (Hz). The individual speaker's fundamental frequency depends on the length of vocal folds. Typically, its value ranges from 85-155Hz in males, 165-225Hz in females, and around 300Hz in children. Due to physiological differences, men tend to have lower frequency than women.

F_0 is influenced by different factors, usually melody, mood, distress, etc. Sometimes it is closely referred to as pitch, however, pitch is a perceptual attribute of sound denoting highness or lowness of sound whereas fundamental frequency is an objective attribute of sound wave and can be quantified. Variations in F_0 provides information about the speaker's gender, age, and emotional state making it a key feature in speech synthesis, speaker identification, and emotion recognition.

The dominant frequency refers to the frequency component with the highest amplitude within a speech signal. It often corresponds to the most prominent formant, which helps characterize vowel sounds. Fundamental frequency has additional frequencies called harmonics, which are integer multiples of it ($2F_0$, $3F_0$, $4F_0$, etc.). The dominant frequency is one of these harmonics and plays a significant role in speech perception, helping distinguish different phonetic sounds. Furthermore, the analysis of dominant frequency components aids in enhancing speech intelligibility, reducing background noise, and improving the overall quality of speech processing systems.

4.4 Phonemes

Phonemes are the smallest units of sound in a language, essential for reading, writing, and word formation. These phonemes are represented by graphemes, which may consist of single letters or letter combinations. The number of phonemes in a language varies based on accents, dialects, and phonetic structures. In English, there are 26 alphabets (graphemes) but 44 phonemes, categorized into 24 consonants and 20 vowels, which further include 12 monophthongs (pure vowels) and 8 diphthongs (gliding vowels). Linguists use the International Phonetic Alphabet (IPA) to standardize pronunciation across different speakers. In contrast, Nepali phonetics consists of 36 consonants (व्यंजन) and 12 vowels (स्वर), following a more syllabic structure where pronunciation closely aligns with spelling. Unlike English, Nepali features aspirated consonants (e.g., "ठ" /ʈʰ/ and "ढ" /ɖʱ/), which have no direct equivalents in English, highlighting the phonetic diversity between the two languages.

The way phonemes are pronounced varies based on accent and dialect, influencing intonation, stress patterns, and word articulation. In English, British and American

accents exhibit differences in phoneme usage, such as the pronunciation of "car" (/kɑːr/ in British English vs. /kɑr/ in American English).

Now, in case of Nepali language is a descendent of Indo-Aryan language family, and is the official language of Nepal. The foundation of Nepali language was laid by early settlers of Sanskrit language. However, the actual Nepali language emerged from “Khasa Prakrit”, a type of Sanskrit-Prakrit language.

In oral form, Nepali language has several dialects and regional variations. As regards to dialects, there are two types of it: Eastern and Western dialects. These dialects have a different pronunciation and vocabulary. Similarly, as regards to regional variations, there exists an influence of elements of local language and of those who use Nepali Language as second language.

In written form, Nepali language is written in Devanagiri Script, which is an alpha syllabary script. Thus, each character represents a consonant with an inherent vowel sound that can be modified using diacritics. Devanagiri script is written from left to right, and there exist no uppercase or lowercase letters, rather the words are grouped together with a “शिरोरेखा″ running over them. Nepali language has 11 vowel phonemes, categorized into short and long vowels where as it has 33 consonants based on their place and articulation manner. The script is logical and phonetic, has following statutes to follow:

- सुन्दराको अक्षराल्य : This is a decorative placement of certain letters in the middle of a word, usually add visual beauty and elegance to script.
- ठुल्को अक्षराल्य : This is a typographic feature of the script to make words look more appealing with the addition of small curved lines above certain letters.
- वर्ण वृद्धि : This represents a particular syllable sound, done by addition of tilde ~ symbol to indicate nasalization of sound.
- मात्रा : These are the small symbols used alongside consonant letters of Nepali Phonemes to represent sounds and syllables missing from the actual language.
- रेखा : This is the use of vertical and horizontal lines alongside curves to form letters.
- अनुस्वारा : This is a diacritic that indicates nasal sound at the end of a syllable.

- अवग्रह : This is a diacritic that indicates long vowels and their sounds.
- बिन्दी : This is a diacritic that indicates short vowels and their sounds.

4.5 Spectrogram and Mel-Spectrogram

The spectrogram is the visual representation of frequencies of a given signal with time. It is a three-dimensional plot where the x-axis represents time, the y-axis represents frequency, and the color intensity represents the amplitude or magnitude of a given frequency at a particular time. Darker regions indicate higher amplitude (louder sounds), while lighter regions correspond to lower amplitude (softer sounds). Spectrograms provide a comprehensive way to analyze sound patterns, making them widely used in speech processing, music analysis, and acoustic signal processing. The term "spectrogram" originates from "spectrum," referring to the breakdown of a wave into its constituent frequency components at any given time.

A Mel-spectrogram is a spectrogram that uses a perceptually scaled frequency axis based on the Mel scale, which mimics the human ear’s sensitivity to sound frequencies. Unlike a standard spectrogram, which represents frequencies linearly, the Mel scale applies a non-linear transformation to group lower frequencies more densely while spreading out higher frequencies. This reflects how humans perceive pitch, as we are more sensitive to changes at lower frequencies than at higher ones. Mel-spectrograms are widely used in speech recognition, music classification, and deep learning applications in audio processing due to their ability to capture speech-relevant frequency features more effectively than traditional spectrograms.

4.6 Cepstrum

In Fourier analysis, the cepstrum is obtained by applying an Inverse Fourier Transform to the logarithm of the magnitude spectrum. This transformation helps in analyzing the frequency characteristics of a signal by separating rapid and slow variations in the spectrum, making it useful for distinguishing different spectral components. The term "cepstrum" is derived by rearranging the letters of the word "spectrum," highlighting its role in spectral analysis.

The cepstral analysis is particularly useful in speech processing, as it allows for the separation of the vocal source from the vocal tract characteristics. This separation aids in applications such as speaker identification, pitch detection, and speech enhancement by improving the clarity and intelligibility of audio signals. Additionally, the concept of quefrency—an independent variable in cepstral analysis—represents time delays within the signal and helps in identifying periodic components such as pitch harmonics.

4.7 Mel-Frequency Cepstral Coefficients (MFCCs)

MFCCs are a set of coefficients that can capture the shape of the power spectrum of a sound signal. They are widely used features for speech recognition. In the speech recognition problem, there will be an audio signal as input to the model and it has to predict the text from it. But it contains a lot of noise in the audio signal. For much better performance of the model, it is observed that extracting features from the audio signal and then using it as input will be more beneficial. Thus, MFCC is a technique used as feature extraction for the speech signal. They represent the spectral characteristics of sound that are suited for machine learning tasks. By isolating relevant features, MFCCs help reduce the complexity and enhance the model's ability to understand speech accurately. This method allows the model to focus on the most important aspects of the signal, such as pitch and tone, improving recognition accuracy. As a result, the speech recognition process becomes more efficient and reliable, even in noisy environments. Here is the breakdown of the process involved in it:

- **A/D Conversion**

It converts the analog signal to a digital one with a sampling rate of 16kHz.

- **Pre-emphasis**

Pre-emphasis helps to boost the high frequencies in the signal countering the spectral tilt(related to the glottal source) and making the system less susceptible to noise.

$$y(t) = x(t) - \alpha . x(t-1) \tag{4-3}$$

Where α is typically set to 0.97.

- **Windowing**

Windowing divides the signal into overlapping frames analyzing the short segments of the signal.

$$x(n) = y\,(n).w(n) \tag{4-4}$$

Where x(n) is the sliced frame, y(n) is the original audio clip, and w(n) is the window applied function to the given signal.

- **Discrete Fourier Transform (DFT)**

Discrete Fourier Transform is applied to each frame where the time domain is transformed to the frequency domain.

$$X[k] = \sum_{n=0}^{N-1} x[n] exp\left(-j\frac{2\pi}{N}kn\right) \tag{4-5}$$

Where X[k] is the k^{th} frequency component of the Fourier Transform, x[n] is the discrete signal in the time domain, n is the time index, and N is the total number of samples in the signal x[n].

- **Mel Filterbank**

The output of the DFT is squared and called as power spectrum, which is applied with a set of triangular pass-band filters. These are spaced according to the Mel scale and output called as Mel-scale power spectrum.

$$Y_t[m] = \sum_{k=1}^{N} W_m[k]|X_t[k]|^2 \tag{4-6}$$

Where $Y_t[m]$ represents the Mel-scale power spectrum.

$$M_f = 2595 \log_{10}\left(\frac{f}{700} + 1\right) \tag{4-7}$$

Where f is the frequency in Hz and m_f is the Mel scale.

- **Logarithm**

The Mel power spectrum is converted to the logarithm scale to remove the F0 information and make the extracted features independent of others.

- **Discrete Cosine Transform (DCT)**

The log Mel power spectrum is converted back to the cepstral domain. The output of the DCT provides the cepstral coefficients known as MFCC (Mel Frequency Cepstrum Coefficient).

$$y_t[n] = \sum_{m=0}^{M-1} \log(Y_t[m]) \cos\left(n(m + 0.5)\frac{\pi}{M}\right) \quad (4\text{-}8)$$

Where $y_t[n]$ represents the MFCC and m is the number of the coefficients.

4.8 Delta MFCC and Delta-Delta MFCC

Delta MFCC is the first-order derivative of MFCCs. Delta MFCC is a mathematical feature used in speech and audio processing to report how MFCC changes over time. It is used to record how these unknown features change from one component of audio to another. Delta MFCC is an uncomplicated and effective way to capture the short-term changes in the sound, helping machines to understand changes in speech and other audio signals more effectively. They provide information about the speed and direction of changes in the audio signal's spectrum. They quantify the difference between the MFCCs of consecutive frames. By capturing this temporal variation, Delta MFCCs allow the model to recognize dynamic speech features that are crucial for accurate recognition. This difference tells how much each MFCC coefficient has changed over time.

Double delta MFCC is the second-order derivative of MFCCs. They provide additional information about how the rate of change of MFCCs progresses over time. They disclose whether the changes in MFCCs are increasing or decreasing over time. They help in identifying different patterns of speech and sound changes, supporting tasks such as speech recognition, speaker identification, and emotion detection. Double delta MFCCs extend the short-term information captured by Delta MFCCs, providing a deeper understanding of how the spectral characteristics of audio signals evolve over time. This added layer of information enhances the model's ability to discern subtle variations in speech and improves its performance in complex audio tasks. Ultimately, they allow for more accurate interpretation and recognition of speech in dynamic and varied environments.

4.9 Connectionist Temporal Classification (CTC)

CTC is an algorithm used for training deep neural networks, where there is no explicit information about alignment between input and output sequences. CTC helps computers to learn from past data patterns and make better predictions about the future. To better understand, let's consider the audio clip as an input and its transcribed words as output. The problem is that the audio input and transcribed word alignment are unknown, and vary from person to person. CTC addresses this challenge. It gives an output distribution over all possible output sequences. It helps to assess the probability of a given output. CTC is alignment-free, i.e. it doesn't need alignment between input sequence and output sequence. This flexibility allows the model to work with unaligned sequences, making it highly effective for tasks like speech recognition, where the timing of the spoken words is unpredictable. By predicting the most likely sequence of words, CTC enables accurate transcription even with imperfect or variable inputs.

4.9.1 CTC Basic Steps

Its algorithm can be understood in three steps:

- **Alignment**

Dealing with the alignment introduces two problems:

a. Output text taking more than one time step,
b. And repeating characters

To solve this CTC introduces a new token for the repetitive character known as a blank symbol (ε).

- **Loss Function**

For the loss calculation, the input sequence is modified including blank symbols between each pair of characters. It enhances flexibility in aligning the input sequence with the output sequence. The CTC loss function calculates the probability of the correct output sequence from the input sequences.

Labeling sequence with blanks,

$$\pi = (\pi_1, \pi_2, \pi_3, \dots, \pi_T) \quad (4\text{-}9)$$

Where $\pi_T \epsilon L$

Probability of an alignment,

$$P(\pi\,|x) = \prod_{t=1}^{T} y_{\pi_T(t)} \quad (4\text{-}10)$$

Where $y_{\pi_T(t)}$ is the network output probability of label π_T at time *t*.

Then,

$$CTC\ loss\ (L_{CTC}) = -\log \sum_{\pi\,\epsilon\,B\,(y)} P\,(\pi\,|x) \quad (4\text{-}11)$$

Where *B(y)* is a set of all valid alignments collapse to the output sequence y.

- **Inference**

CTC inference aims to find the most likely output sequence y given the input sequence x and the network's output probabilities.

It is given by,

$$\hat{y} = \arg max_y \sum_{\pi \epsilon B(y)} \prod_{t=1}^{T} y_{\pi_T(t)} \quad (4\text{-}12)$$

Where $\hat{y}$ most likely output sequence.

4.9.2 CTC Algorithm

Required: Input sequence x, Output sequence y, Neural network with output probabilities $y_T(t)$, blank symbol ϵ.

a. **Initialization:** Introduce a new token for the repetitive character called a blank symbol ϵ
b. **Preprocessing:** Modify the input sequence including blank symbols between each pair of characters, at the start and end.
c. **Computing Output Probabilities:** Calculate network output probabilities $y_T(t)$ for each time step t and each possible label T.
d. **Alignment:** Compute the probability of an alignment,

$$P(\pi|x) = \prod_{t=1}^{T} y_{\pi_T}(t) \quad (4\text{-}13)$$

Where$y_{\pi_T}(t)$ is the network output probability of label π_T at time *t*.

e. **CTC Loss Calculation**

$$CTC\ loss\ (L_{CTC}) = -\log \sum_{\pi\ \epsilon\ B\ (y)} P\ (\pi\ |x) \quad (4\text{-}14)$$

Where B(y) is a set of all valid alignments collapse to the output sequence y.

f. **Inference (Decoding the Output):** Find the most likely output sequence y.

$$\hat{y} = \arg max_y \sum_{\pi \epsilon B(y)} \prod_{t=1}^{T} y_{\pi_T(t)} \quad (4\text{-}15)$$

5. SYSTEM ARCHITECTURE AND METHODOLOGY

5.1 System Block Diagram

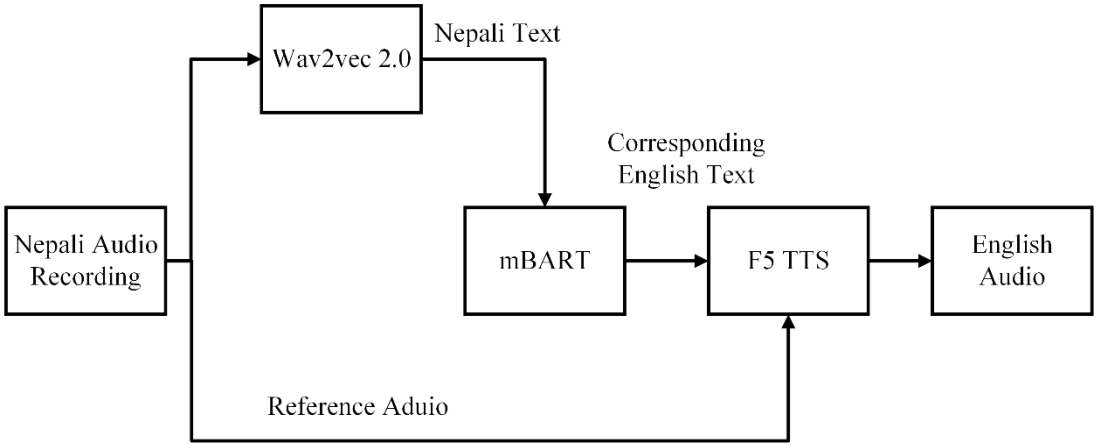


Figure 5-1: System Block Diagram

The system begins with Nepali audio recording, which is processed by the wav2vec 2.0 model to generate the corresponding Nepali text. This text is then passed to the mBART model, which translates the Nepali text into English. The translated English text is then fed into the F5 TTS model, which also takes in the speaker embedding from a speaker encoder for voice adaptation. The F5 TTS model synthesizes the final English audio output while maintaining the desired prosody and speaker characteristics.

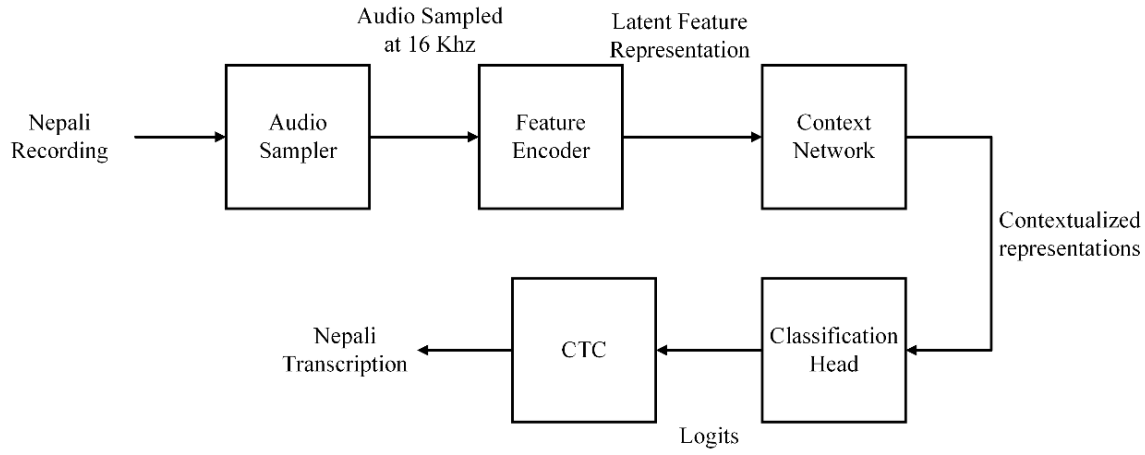


Figure 5-2: Flow of Data in wav2vec 2.0

The ASR model utilized in this system is wav2vec 2.0, a self-supervised learning model pre-trained on large-scale speech data and further fine-tuned on Nepali speech-text pairs

to accurately transcribe Nepali speech into text. The process begins with Nepali audio recording, which is sampled at 16 kHz by the Audio Sampler to standardize the input signal. The sampled audio is then fed into the Feature Encoder, which extracts low-level acoustic features and transforms the raw waveform into a latent feature representation. This latent representation is subsequently processed by the Context Network, which captures long-range dependencies and contextual relationships within the speech signal to generate a contextualized feature representation. The Classification Head, which serves as the final layer of the network, converts the contextualized features into logits, representing the probability distribution over possible character sequences. These logits are then passed through a CTC decoder, which aligns and transforms them into the final Nepali transcription, effectively mapping the spoken input to its textual counterpart.

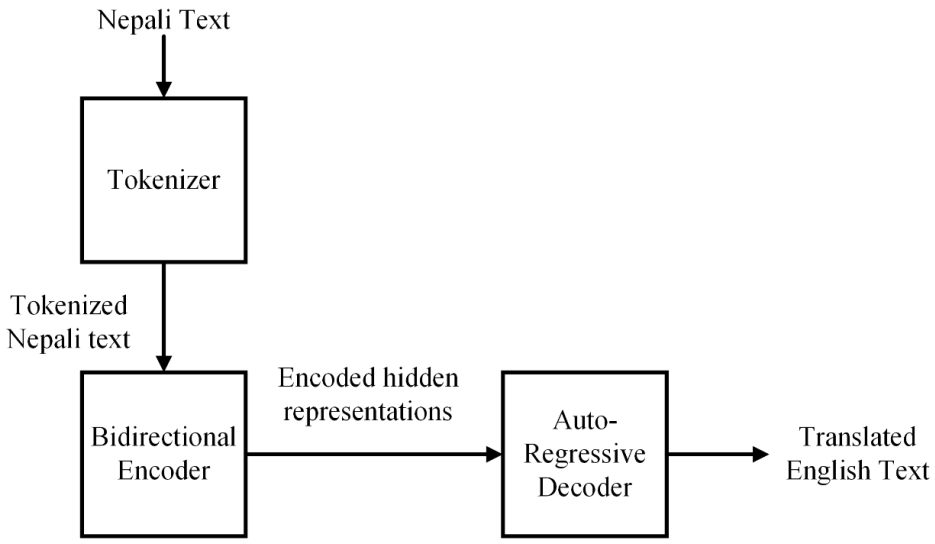


Figure 5-3: Flow of Data in mBART

The generated Nepali transcription is subsequently processed by the machine translation system, which, in this case, is the multilingual BART (mBART) model. mBART is a pre-trained sequence-to-sequence model that has been trained on a diverse set of languages, including Nepali. To ensure high-quality translation from Nepali to English, the model undergoes fine-tuning using a parallel corpus of Nepali-English text pairs. This fine-tuning process enables mBART to effectively learn language mappings and enhance its translation capabilities.

The translation pipeline begins by passing the Nepali transcription to a tokenizer, which converts the input text into a sequence of subword tokens. These tokenized representations are then fed into the bidirectional encoder of the mBART model. The encoder processes the input sequence and generates a contextualized hidden representation, capturing both syntactic and semantic information. This encoded representation is subsequently forwarded to the autoregressive decoder, which iteratively generates the translated English text in a left-to-right manner. The decoder employs self-attention mechanisms and cross-attention with the encoded representations to produce a coherent and contextually accurate translation. Finally, the generated English text undergoes further processing and is passed to a TTS model, which synthesizes the textual output into spoken language. This pipeline ensures an end-to-end system capable of converting spoken Nepali into fluent English speech while preserving the integrity of the original message.

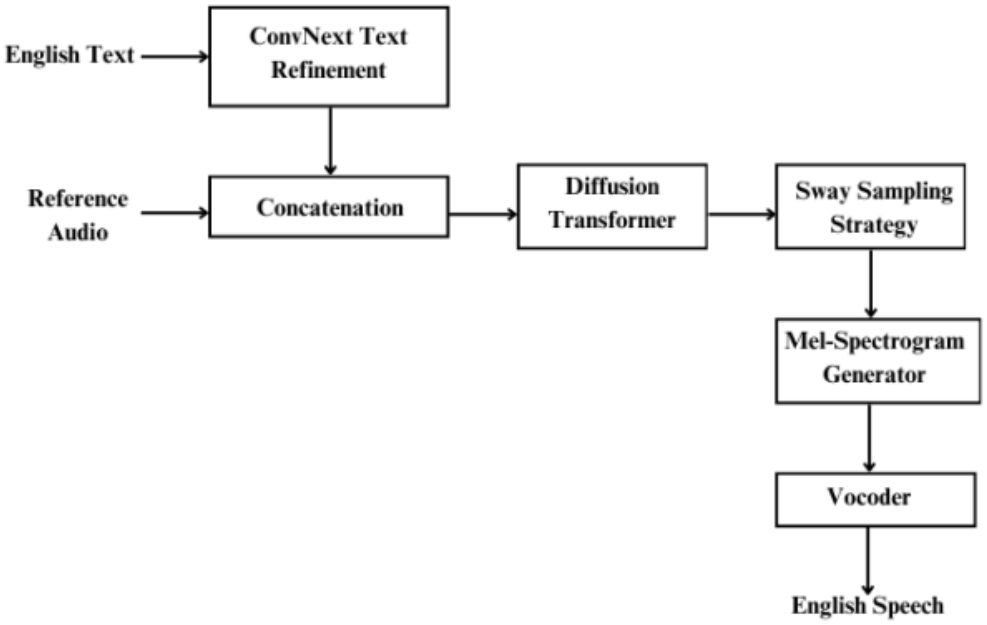


Figure 5-4: Flow of Data in F5-TTS

In this system, the TTS module is implemented using the F5-TTS model, which converts English text into natural-sounding speech. The process begins with the input English text, which serves as the foundation for speech synthesis. This input undergoes a ConvNeXt-based Text Refinement stage, where it is processed to enhance its structure and align with the characteristics required for natural speech generation. Concurrently, an Audio Reference is incorporated, providing crucial information regarding the

speaker's style, prosody, and voice quality. Following text refinement, the processed text is concatenated with a noisy audio input to form a combined representation. This composite data is then fed into the Diffusion Transformer, where the speech signal undergoes iterative refinement, progressively shaping the output to achieve a natural and coherent flow. During inference, the Sway Sampling Strategy is employed to optimize the selection of key steps, ensuring both efficiency and clarity in the generated speech. Subsequently, the refined representation is transformed into a Mel-Spectrogram, an intermediate acoustic representation that captures the spectral characteristics of speech. Finally, this Mel-Spectrogram is processed by the Vocoder, which synthesizes it into natural and intelligible speech, thus completing the TTS pipeline. This approach ensures high-quality speech synthesis, preserving both speaker characteristics and linguistic nuances.

5.2 Flowchart

The process of converting Nepali speech to English speech involves a series of advanced procedures designed to ensure accurate translation while preserving the original prosody. Initially, the Nepali speech signal is recorded and sampled at a rate of 16 kHz, optimizing audio quality and ensuring compatibility with subsequent processing stages.

The sampled audio is then processed by wav2vec 2.0, an ASR model, which transcribes the speech into its corresponding Nepali text representation. The transcribed text is subsequently passed to mBART, a sophisticated multilingual neural machine translation system. The mBART model, fine-tuned for Nepali-English translation, accurately converts the Nepali text into English while maintaining contextual integrity and linguistic nuances.

Simultaneously, a speaker encoder extracts speaker embeddings directly from the raw audio input. These embeddings encapsulate crucial speaker-specific characteristics, including voice timbre, speech rhythm, and prosody. The extracted speaker embeddings, along with the translated English text, are then fed into F5 TTS, a high-performance neural speech synthesis model. F5 TTS generates English speech while preserving the prosodic features and stylistic attributes of the original Nepali speaker.

The resulting English speech maintains naturalness, ensuring that the translated output closely resembles the original speaker's expressive patterns. This pipeline effectively integrates high-quality translation with prosody preservation, resulting in a seamless and intelligible speech-to-speech translation system. By maintaining speaker identity and natural expressiveness, this approach enhances the overall quality and usability of cross-lingual speech conversion.

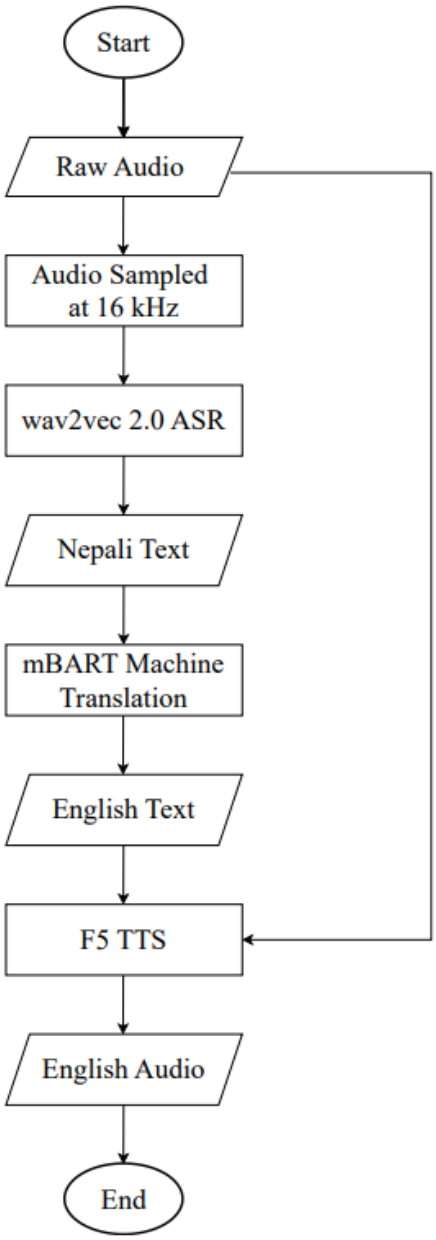


Figure 5-5: Flowchart of the System

5.3 wav2vec 2.0

wav2vec 2.0 is a pre-trained model developed by Facebook research. This model addresses the challenge of less data by achieving high accuracy even with significantly less labeled data making a significant advancement in speech recognition. wav2vec 2.0 is a self-supervised learning framework for learning latent speech representation from raw audio data.

5.3.1 Architecture of wav2vec 2.0

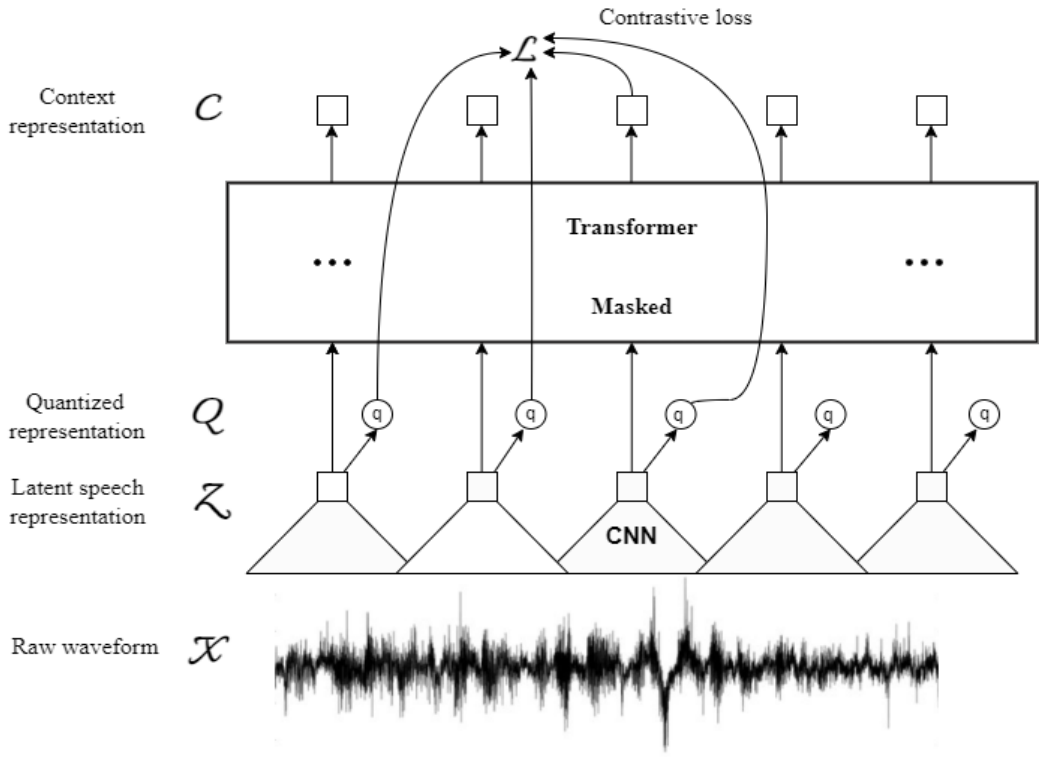


Figure 5-6: wav2vec 2.0 Architecture

5.3.1.1 Feature Encoder

The model consists of a multi-layer convolutional feature encoder that maps the input raw waveform to the latent representation. Here, it takes input raw audio and converts it into a sequence of feature vectors or latent speech representation for different time steps. This step is also known as a pre-processing step. The audio waveform is normalized by subtracting the mean and dividing the variance (i.e. zero mean and unit variance). This ensures the model finds the raw audio in a consistent format enhancing better feature extraction and improved speech recognition performance. This 1D

convolution layer is the core building of the feature encoder. It contains multiple hidden layers. The CNN extracts different elements from the audio i.e. essential for speech recognition. The normalized waveform is passed to the 1D convolution layer as input. This layer has a 7-layer 1D convolution network with 512 channels at each layer. The convolution layer processes the normalized input and extracts different features from the input data using different convolution kernels and strides and down samples the signal. The layer normalization step is performed after each convolutional layer and strengthens the training process by adjusting the internal covariate shift. After the normalization, the activation function is applied. Instead of the ReLU activation function GELU activation function is used. The result of the feature encoder is a Latent feature vector. The latent feature vector is a sequence of feature vectors that capture high-level information about the audio signal, ready to be processed by the context network.

5.3.1.2 Contextualized Representation with Transformers

The core part of the wav2vec 2.0 is its transformer encoder. The output from the feature encoder is passed to the feature projection layer that projects the 512-channel to 1024-channel to match the input of the transformer block. This projected output is passed to the transformer-based context network to generate contextualized representation and process it through 24 transformer blocks. Instead of absolute positional information, as in the original transformer, relative positional information is added to the input. Relative position information is achieved through the grouped convolutional layer. Grouped convolution divides the input channels into smaller groups, applies separate convolutions to each group, and then concatenates the results. Transformer-based architecture can capture long-range dependencies in sequence taking masked latent speech representations as input and processing key, query, and value matrices to account for the relationships between the audio segment and preserving the context of audio. The context representation is the output from the transformer network.

5.3.1.3 Quantization Module

The Quantization module is the module designed to convert latent continuous vectors to discrete speech representation. This quantization module is used for self-supervised learning. Speech is continuous which leads to changes in its value over time and in

neural networks typically performed on discrete data, quantization is performed. Similar latent vectors are grouped to form a limited set of codewords. To make the training process more stable, the Gumbel SoftMax trick is used. The Gumbel-Softmax trick is a technique that enables the differentiation of sampling from a categorical distribution, allowing for the training of neural networks with discrete variables using gradient-based methods. For each latent representation Zi in Z, logits are calculated for each codeword. Logits are raw outputs indicating the likelihood of each class. These are used by the SoftMax function to produce a probability distribution.

$$y_i = \frac{(exp^{\frac{log(\alpha_i)+g_i}{T}})}{\sum_{j=1}^{k} exp\left(\frac{log(\alpha_j)+g_j}{T}\right)} \quad (5\text{-}1)$$

Where, $log(\alpha_i)$ is logits for each class, g_i is noise for each class and T is a temperature parameter.

5.3.1.4 Linear Layer

The linear layer receives the output from the encoder layer of the Transformer block of dimension 1024 and maps it to the number of characters in the vocabulary.

5.3.1.5 Pre-Training

While pre-training the model, the time steps were masked in the latent feature encoder. The training objective required the model to correctly identify the quantized latent audio representation from the distractors for each masked time step. During the pre-training two losses were taken into consideration contrastive loss and diversity loss. For masking, about 49% of the total time steps were masked with a mean span length of 299 ms. The model was pre-trained on the Librespeech dataset for 2.3 days and LibriVox for 5.2 days on 128 V100 GPUs. The weight for the diversity loss was used as 0.1.

5.3.1.6 Contrastive Loss

Contrastive loss plays a crucial role in training wav2vec 2.0. It is used during the pre-training phase to capture similar data points. This loss helps the model learn representations by distinguishing between true future representation (positive samples) and incorrect predictions i.e. distractors (negative samples). The contrastive loss aims

to maximize the similarity between context representation (c_t) and the corresponding quantized latent representation (positive sample) while minimizing the similarity between context representation and the negative samples. The model is responsible for identifying correct quantized latent speech representation q_t among K+1 quantized candidate representation q Ɛ Q_t which consists of target q_t and K distractors sampled uniformly from other masked time steps.

Mathematically, the loss function is defined as,

$$L_m = -\log \frac{exp(sim(c_t, q_t)/K)}{\sum_{q \sim Q_t} exp(sim(c_t, q)/K)} \quad (5\text{-}2)$$

Where K is the temperature constant during training and sim is cosine similarity sim (a,b).

5.3.1.7 Diversity Loss

Diversity is designed to encourage the use of a quantized codebook for discrete latent representations. Instead of using only a few code words, it encourages the equal use of entire codebooks by maximizing the entropy.

Diversity loss is defined by:

$$L_d = \sum_{i=1}^{v} \pi \log \pi \quad (5\text{-}3)$$

5.3.1.8 Training Model

The training model is the total of two loss functions i.e. contrastive loss and diversity loss.

Mathematically,

$$L = L_m + \alpha L_d \quad (5\text{-}4)$$

Where α is the tuned hyper parameter that controls the weight of the diversity loss relative to the contrastive loss.

5.4 mBART

BART stands for Bidirectional and Auto-Regressive Transformers and is a denoising autoencoder for pre-training sequence-to-sequence models. BART is designed to handle a variety of NLP tasks by reconstructing text that has been corrupted through a specific noising process. The model leverages the robustness of Transformer architecture, combining the strengths of both bidirectional and autoregressive transformers. The encoder is bidirectional and the decoder is autoregressive in nature. Such encoder-decoder architecture allows BART to excel in tasks that require both understanding context and generating coherent, contextually appropriate text such as translation.

mBART (Multilingual BART) is an extension of BART that is designed specifically for multilingual sequence-to-sequence tasks. It builds on BART's architecture to handle the complexity of translating between different languages with a single, unified model. mBART aims to enhance the performance of machine translation and other text generation tasks across multiple languages by leveraging its pre-training on large-scale monolingual corpora from a variety of languages. By using the BART objective of denoising texts, it is able to clean and improve the quality of the input data, making it more effective at understanding and generating multilingual text. This allows mBART to provide more accurate translations, even for low-resource languages, and it can be fine-tuned for specific tasks, making it versatile for a wide range of natural language processing applications. Additionally, mBART can handle a variety of language pairs without requiring separate models for each, which drastically reduces computational overhead and simplifies implementation. Its ability to transfer knowledge between languages enables it to produce high-quality outputs even in scenarios with limited training data. This makes mBART an excellent choice for real-world multilingual applications where scalability and efficiency are key.

5.4.1 Architecture of mBART

mBART uses standard sequence-to-sequence Transformer architecture followed by the architecture of GPT. mBART uses 12 layers of the encoder and 12 layers of the decoder with a model dimension of 1024 on 16 heads. An additional normalization layer was added to the architecture to stabilize the training. The encoder is bidirectional, capturing

context from both directions, while the decoder is autoregressive, generating text sequentially. The explicit architecture of the BART is not provided in the base paper.

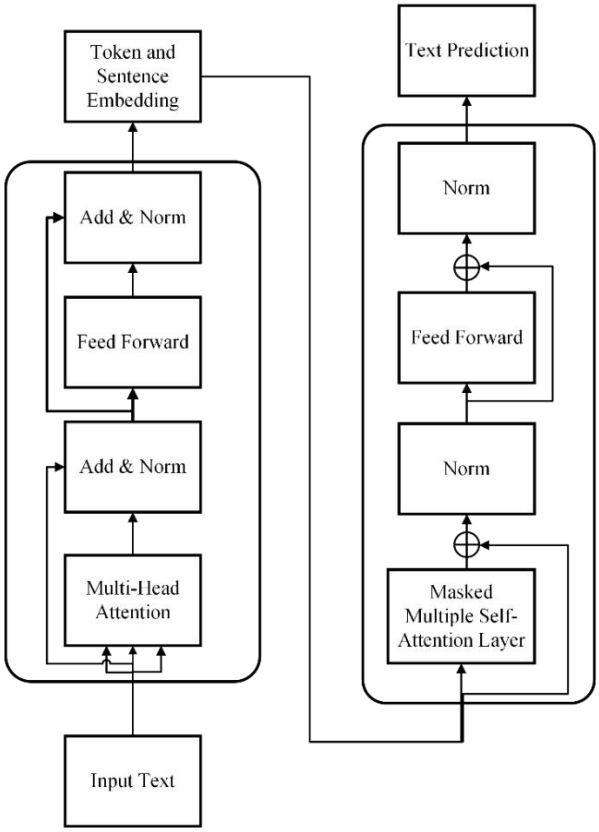


Figure 5-7: mBART Architecture

5.4.1.1 Encoder

The figure 5-7 depicts the architecture of mBART. The first block is the encoder that is bidirectional in nature and second block is of decoder that is autoregressive. The input is passed to the multi-head attention block. Multi-head Attention is a module for attention mechanisms that run through an attention mechanism several times in parallel. The multi-head attention block replaces the random token present in the sentence with the mask token. Here each takes the input token and added masks using from each head.

$$MultiHead\ (Q, K, V) = [head_1, head_2, \ldots.. head_n]W_0 \quad (5\text{-}5)$$

Where W_0 is the learnable parameter and $head_i = Attention(QW_i^Q, KW_i^k, VW_i^v)$

The different blocks of the head contain different values within different ranges. So, to bring uniformity among all the values of the attention head all the parameters from the

multi-head attention blocks are added and normalized to a constant value using a monotonic function. The connection from the input is used to combine both complete clean text as well as randomly masked tokens.

$$AttentionOutput = LayerNorm(X' + MultiHead) \quad (5\text{-}6)$$

Where, X' is the embedding of the input text.

The output of the normalized layer is passed to the feedforward network that introduces non-linearity in the network that enables the model to learn deep features in the network.

$$FFNOutput = FFN(AttentionOutput) \quad (5\text{-}7)$$

Another layer of normalization is added to the output of the feed-forward network and a residual connection from the input of the feed-forward network to its output.

$$LayerOutput = LayerNorm(AttentionOutput + FFNOutput) \quad (5\text{-}8)$$

5.4.1.2 Decoder

The output of the normalization layer is passed to the decoder. The first component of the decoder is masked multiple self-attention which has a similar architecture to that of the multiple attention head but works sequentially instead of parallel. The masked multiple attention block tries to decode the masked embedding to semantically coherent tokens.

The output is added and normalized with the original embedding. The output of the normalized layer is passed to the standard feed-forward layer block where information is learned to decode the sentence from the processed embedding. At the final layer, the predicted tokens are tried to match with the clean output which is backtracked over the entire encoder-decoder network.

5.4.1.3 Pre-Training

The data were first pre-processed by tokenizing i.e. converting the given input sentences into word sub-unit. No other pre-processing techniques were applied. The main goal of the model training was to introduce a noise function that corrupts the given

input text and the model predicts the original text when the noise function is given. During the training of mBART, two types of noise were introduced. The first type of noise was replacing the text with a mask token. About 35% of words in each training instance were replaced with mask tokens following random sampling. The second noise was to permute the sequence of sentences within each sentence.

5.5 F5-TTS

F5-TTS, a Fairytaler that Fakes Fluent and Faithful Speech with Flow matching is a fully non-autoregressive text-to-speech model basically based on flow matching with Diffusion Transformer(DiT). Unlike other traditional models, it does not require complex designs such as text encoder, duration model, and phoneme alignment rather the input text is simply padded with filler tokens to the same length and then denoising for speech generation. F5-TTS illustrates the strong robustness of the generation of faithful speech for given text alongside managing comparable speaker similarity. F5-TTS is a zero-shot speech synthesizer that generates fluent, faithful, and prosodically natural speech by jointly aligning the linguistic content of text input to the prosody and style of a reference voice.

5.5.1 Architecture of F5-TTS

The input to the model is text and audio reference which generates synthesized speech that is prosody-aware. This, in turn, involves text preprocessing, prosody alignment, temporal modeling using a Diffusion Transformer, and finally, waveform generation with a vocoder. First of all, linguistic feature extraction is applied to the text input, which in most contexts would imply a sentence where the text is then tokenized into smaller units such as phonemes or characters to represent the sounds or structures in the text. After that, the tokens are subjected to a trainable embedding layer to return dense vector representations of the text data known as embeddings. These embeddings are referred to as Z_{text} and represent phonetic or linguistic representations of the input text. The goal here is to project the linguistic information of the input in a form that can be satisfactorily used by other components of the model toward speech generation.

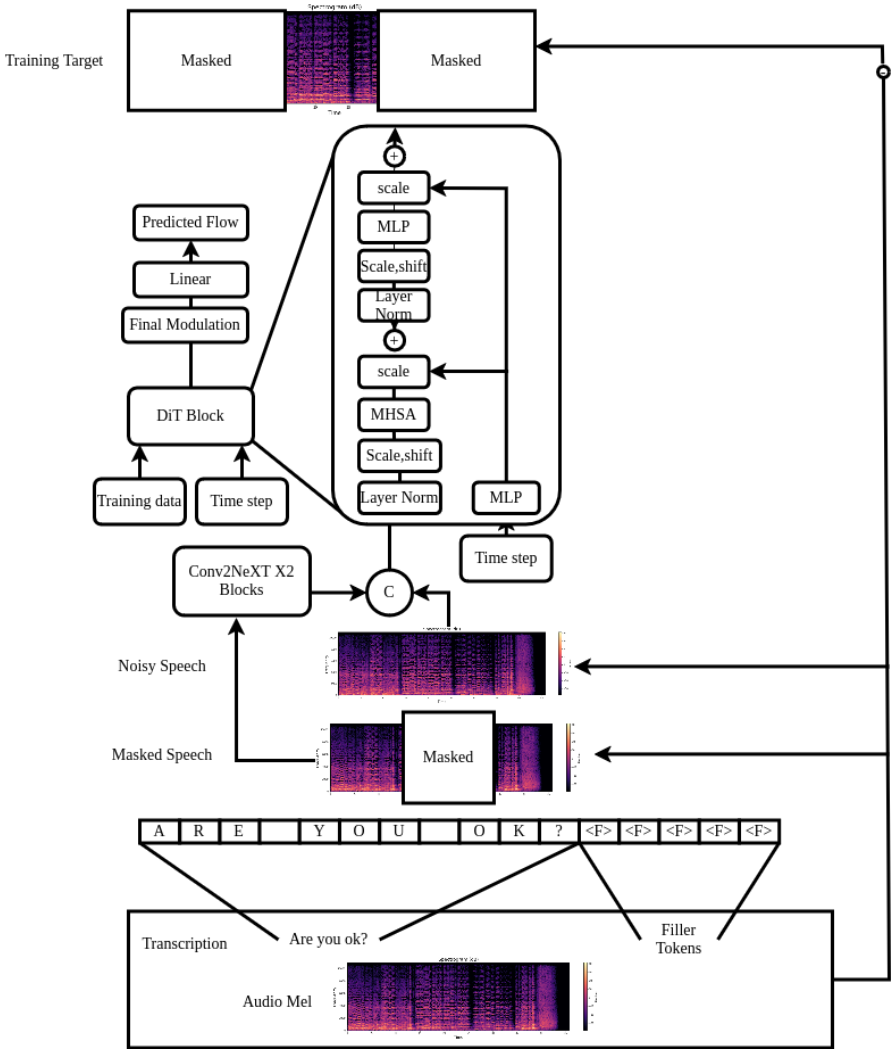


Figure 5-8: F5-TTS Architecture

5.5.1.1 Audio Reference Encoding

The system processes a reference audio clip as input to enable zero-shot voice cloning to capture the particularities of the speaker's voice and speaking style. This begins by converting the reference audio into a mel-spectrogram, M_{Ref}, which is the mapping to capture frequency and intensity variations along the time. After that, this mel-spectrogram will be fed to the CNN-based style encoder to catch such important features. These extracted features are encoded into dense vector representations known as style embeddings, denoted as Z_{style}. This is done to capture the prosody and other special characteristics of the reference speaker's voice, which will guide the synthesis process to mimic the speaker's style effectively.

5.5.1.2 ConvNeXt Refinement

. The ConvNeXt Refinement Block is an essential module that aligns and combines the linguistic information from the text embeddings with the prosodic features extracted from the reference audio. This way, it ensures that the synthesized speech catches both the linguistic structure of the input text and the prosodic characteristics of the reference voice. Text embeddings, Z_{text}, go through a stack of ConvNeXt V2 blocks. These blocks fine-tune the linguistic features to identify meaningful patterns that are essential to speech synthesis. The reference audio, x_{ref}, is also passed through ConvNeXt V2 blocks so that it can extract prosodic features like pitch, rhythm, and intonation. These features express the speaking style of the reference audio. Then the output of the ConvNeXt Refinement Block h_{concat}, is a unified representation that encodes both the linguistic structure of the input text and the prosodic features of the reference audio.

The text embeddings Z_{text} are processed through the ConvNeXt V2 blocks:

$$h_{text} = ConvNeXt(Z_{text}) \tag{5-9}$$

Where h_{text} represents the refined linguistic features.

Likewise, the mel-spectrogram of the reference audio, x_{ref} is processed through another ConvNeXt V2 block:

$$h_{audio} = ConvNeXt(\, x_{ref}) \tag{5-10}$$

This h_{audio} encodes the prosodic characteristics of the reference audio.

The refined linguistic features and prosodic features are concatenated along the feature dimension:

$$h_{concat} = [h_{text}; h_{audio}] \tag{5-11}$$

Where h_{concat}encodes both the linguistic structure of the input text and the prosodic features of the reference audio.

5.5.1.3 Diffusion Transformer

F5-TTS relies on the main archetype called the Diffusion Transformer (DiT), which generates high-quality mel-spectrograms by effectively merging the diffusion-based denoising with a Transformer network. Diffusion models learn to reverse a noising process. In this method, noise is gradually introduced in the ground truth mel-spectrogram throughout training across fixed numbers of timesteps *T*. The model learns to predict and denoise the spectrogram at each step. The Transformer captures long-range dependency along time frames of the mel-spectrogram, which is crucial for speech generation with coherent temporal structures. This conditions the process of denoising both by linguistic embedding (text) and prosodic embedding (style).

5.5.1.4 Training Phase

The training phase has two further process: Noising Process and Denoising Process.

Noising Process

In the training phase, the ground truth mel-spectrogram x_1 is perturbed by adding Gaussian noise at each timestep t to create a noisy version of the spectrogram x_t .The process makes the training robust to noise enabling the model to denoise effectively during inference. The noised spectrogram x_t is calculated as:

$$x_t = \alpha_t x_1 + \sqrt{1 - \alpha_t^2 \varepsilon} \tag{5-12}$$

Where x_1 is the ground truth mel-spectrogram, α_t is the noise schedule coefficient that controls the level of noise at timestep t and ε ~N(0, I) is Gaussian noise sampled from a normal distribution.

Denoising Process

The Diffusion Transformer is trained to predict the denoised spectrogram $\widehat{x_1}$ from the noisy spectrogram x_t at each timestep t. The prediction formula is:

$$\widehat{x_1} = DiT(x_b\ Z_{text}\ Z_{style}\ t) \tag{5-13}$$

Where DiT represents the Diffusion Transformer.

5.5.1.5 Inference Phase

During inference, the model starts from pure Gaussian noise $x_T$and iteratively refines it to recover a clean mel-spectrogram $\widehat{x_1}$. The process involves reversing the noising steps by predicting $x_T$from xt at each timestep t.

$$\widehat{x_1} = DiT(x_b\ Z_{text}\ Z_{style}\ t) \tag{5-14}$$

Where DiT represents the Diffusion Transformer.

5.5.1.6 Sway Sampling Strategy

The Sway Sampling Strategy optimizes the process of denoising in F5-TTS by dynamically adjusting the timestep distribution $\pi(t)$ according to the quality of denoising at each time step. It ensures that more effort will be placed on regions that are challenging and guarantees smooth transitions and efficient sampling. Instead of a uniform distribution, the timestep probability is weighted proportionally to the difficulty of denoising at t allowing the model to focus on refining critical areas.

$$\pi(t) \alpha\ quality\ of\ denoising\ at\ t \tag{5-15}$$

Here, $\pi(t)$ represents the probability of sampling timestep t during the denoising process.

5.6 Activation Function

In neural networks, the activation function plays an important role. Each neuron in the network receives some information, processes it, and then decides whether to pass it on to the next neuron. The decision-making part of the neuron is where the activation function comes in. Activation functions apply a non-linear transformation and decide whether a neuron should be activated or not. This helps the model to learn, make improvements, and model complex patterns in data.

5.6.1 GeLU

GeLU stands for Gaussian Error Linear Units. Here, this activation function is used in the wav2vec 2.0 and the mBART model. GeLU introduces non-linearity into the network using a probabilistic approach with the combined characteristics of the ReLU and sigmoid functions. This leads to better performance in the speech-processing areas. It has a smooth and continuous curve that prevents gradient flow.

Its function is given by:

$$GELU(x) = x.\varphi(x) = \frac{x}{2}\left(1 + erf\left(\frac{x}{\sqrt{2}}\right)\right) \tag{5-16}$$

Where φ(x) is the cumulative distribution function of the standard normal distribution, and the range of the GeLU activation function is [-∞,∞].

Approximately,

$$GELU(x) \approx 0.5x\left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}\,(x + 0.044715x^3)\right)\right) \tag{5-17}$$

Where 0.5x is a linear component of the function, tanh is a hyperbolic tangent function to the input helps in maintaining a smooth non-linearity transition, and $\sqrt{\frac{2}{\pi}}$ indicates a normalization factor based on Gaussian distribution.

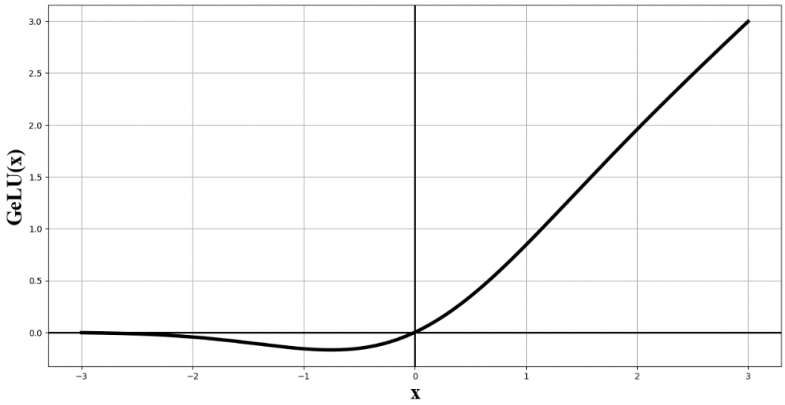


Figure 5-9: GeLU Activation

5.6.2 ReLU

ReLU stands for Rectified Linear unit. It is the most commonly used activation function due to its simplicity and effectiveness. Here this activation function is used in the FastSpeech2 model. ReLU introduces non-linearity to the network that helps to mitigate the vanishing gradient problem and enables them to learn more complex relationships in data. ReLU function outputs the input value if it is positive, and if the input value is negative, it outputs zero. It ranges from [0, ∞).

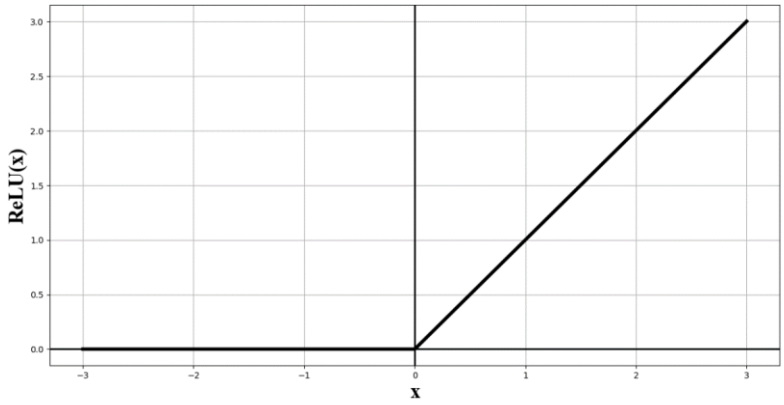


Figure 5-10: ReLU Activation

Its function is defined as:

$$RELU(x) = \max(0, x) \tag{5-18}$$

5.7 Loss Function

A loss function is a mathematical function used to measure the difference between the predicted output and the actual target value. This helps to evaluate the performance of the model ensuring smaller the loss, the better the model's predictions. Different loss functions used in the system are given below.

5.7.1 Cross Entropy Loss

mBART, designed for multilingual sequence-to-sequence tasks, relies on cross-entropy loss. The cross-entropy loss is used to measure the difference between the predicted token probabilities and the actual token distributions. For each token in the target

sequence, the loss function evaluates how well the model's predicted probability distribution aligns with the true token.

$$Cross\ entropy\ loss\ (L) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \quad (5\text{-}19)$$

Where C is the number of classes, y_i is the true label for class i (usually a one-hot encoded vector where the true class is 1 and the others are 0) and $\hat{y}_i$ is the predicted probability for class i.

5.7.2 Mean Squared Error (MSE)

Mean Squared Error measures the amount of error in statistical methods. It is a commonly used loss function that describes how close a regression line is to a set of pre-set points of information. It measures the average square difference between the predicted and actual target values. And lower the MSE, the better the model's prediction. It is used to optimize predictions for durations, fundamental frequency(f0), and energy in the FastSpeech 2 for more natural synthesis.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (5\text{-}20)$$

Where y_i is the actual value (duration, F0, or energy) for the i-th data point, $\hat{y}_i$ is the predicted value for the i-th data point and N is the number of data points.

5.7.3 Mean Absolute Error (MAE)

Mean Absolute Error (MAE) is also known as L1 loss. It measures the absolute difference between the predicted value and the actual target values. FastSpeech 2 uses L1 loss to measure the difference between the predicted and actual ground truth mel-spectrograms. It helps to maintain stability during the training process with the noisy data. Minimizing the loss helps to generate the Mel spectrograms improving the speech intelligibility and its naturalness.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{Y}_i - Y_i\right| \quad (5\text{-}21)$$

Where $\hat{Y}_i$ is the predicted value for the i-th sample, *Yi* is the actual value for the i-th sample and N is the total number of samples.

5.8 Evaluation Metrics

Evaluation metrics are the measures used to check the performance of the learning model. They provide meaningful insights related to the work performance of the model and also help in comparing different algorithms of the model. In evaluating the activity of the model, it is important to assess its predictive ability, generalization capability, and quality of the overall model. The selection of the evaluation metric is based on the specific problem domain, the type of data used, and the outcome desired.

5.8.1 WER

WER stands for Word Error Rate. WER is used to evaluate the ASR model. Word Error Rate (WER) is a metric widely used for the purpose of comparing and evaluating the performance of the systems of Automatic Speech Recognition (ASR). WER estimates the difference by measuring how far the ASR system output is from the reference (correct) transcription. WER is calculated as the number of errors divided by the total words.

$$WER = \frac{S+I+D}{N} \tag{5-22}$$

Where S is the number of substituted words, I is the number of intersected words, D is the number of deleted words and N is the total number of words.

WER ranges from 0% to 100%, where 0% means perfect transcription and 100% means each and every word transcribed is incorrect. This metric is important for ASR systems because it provides a simple way to measure how good the transcription is. Lower the WER better the recognition, lesser errors. WER below 10% is considered good for practical purposes, and WER below 5% is considered excellent. There are other metrics like CER (Character Error Rate) and SER (Sentence Error Rate), but they are not quite perfect. CER doesn't penalize much for word level errors while SER is too harsh, marking entire sentence as wrong even when only one word in the sentence was wrong. WER directly targets what matters more, i.e., word-level errors.

5.8.2 BLEU Score

For the evaluation of the machine translation model, the BLEU score will be used. BLEU stands for Bilingual Evaluation Understudy. BLEU score measures the similarity between the predicted translation and reference translations by computing the overlap between output and reference text. BLEU calculates how many n-grams from the predicted translation exactly match n-grams in any of the reference translations. The BLEU score was computed over tokenized text over both system output and the references.

$$\text{Brevity Penalty } p = \begin{cases} 1 \; if \; c > r \\ e^{1-\frac{r}{c}} \; if \; c \leq r \end{cases} \tag{5-23}$$

$$BLEU = p \times e^{\sum_{n=1}^{N}(\frac{1}{N} * \log P_n)} \tag{5-24}$$

Where P is the brevity penalty, P_n is the precision for n-grams (sequences of words) and N is the maximum length of n-grams considered (usually 4).

BLEU scores lie in the range 0 to 1, where for the 100% match with the reference text, the score is equal to 1. This metric is crucial in the transition models since it enables the organization to compare the quality of the translation consistently. According to the BLEU scores the systems that got higher scores are more accurate in their translation and fluency. In particular, it should be noted that the BLEU coefficient values which are greater than 0 indicate good quality of the text. 0.5 is considered adequate, and the total scores of 0.7 mean that the corresponding values are good, but the acceptable tolerance in this case might be narrower, depending on the concrete work and language combination. Other measures like METEOR, TER, and ROUGE are also applied in MT. Unlike the METEOR that takes account of synonymy and stemming though it is useful, BLEU is widely used due to its simplicity in practice. TER measures the quantity that is the number of edits required to bring all segments into correspondence with the reference, which may not be as easily comprehensible. As a result, it can be noted that ROUGE is more appropriate for activities in the realm of summarization. Due to these reasons, simplicity and efficiency, BLEU has been found to be the most chosen one.

5.8.3 ROUGE Score

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. ROUGE scores are mainly based on Recall, and it was actually designed keeping in mind text-summarization, where the model-generated text is usually shorter than the reference text. It serves to evaluate the quality of machine-generated summaries by comparing them with human-written reference summaries. The ROUGE score usually lies in the range from 0 to 1, where values near 1 indicate a higher degree of similarity between candidate and reference. While ROUGE itself is a robust and interpretable metric, its proper implementation and interpretation are very much crucial for reliability in estimating model performance.

5.8.3.1 ROUGE-N

ROUGE-N measures n-grams, word pairs, and word sequences between the reference and candidate summaries. ROUGE represents a set of metrics for automatic text summarization, which means constructing a short summary from a longer text. N indicates the number of N grams which can be 1 and 2. ROUGE-1 refers to the overlap of unigrams (each word) between the reference summaries and machine-generated. ROUGE-2 refers to the overlap of bigrams between the reference and machine-generated. To calculate ROUGE scores, it is important to understand the concepts of Recall and Precision in text analysis. Recall is the percentage of overlapping units (like words or n-grams) in the reference text that are also present in the candidate text while precision measures the proportion of overlapping units in the candidate text that appear in the reference text.

$$Recall = \frac{Overlapping\ number\ of\ n-grams}{Number\ of\ n-grams\ in\ the\ reference} \tag{5-25}$$

$$Precision = \frac{Overlapping\ number\ of\ n-grams}{Number\ of\ n-grams\ in\ the\ candidate} \tag{5-26}$$

To finalize the calculation, the F1 score i.e. the harmonic mean of Recall and Precision is computed.

$$F1 = \frac{2\ X\ Precison\ X\ Recall}{Precision + Recall} \tag{5-27}$$

These metrics collectively form the basis for ROUGE scores, enabling effective comparison of textual similarity between a candidate and reference text.

5.8.3.2 ROUGE-L

ROUGE-L is based on the LCS, which represents the longest sequence of words that appear in both the candidate and reference summaries while preserving the word order. Unlike consecutive word matches, LCS allows gaps between words as long as the sequence remains in order. This makes ROUGE-L particularly useful for capturing meaningful overlaps between summaries, even when the matches are not exact. The ROUGE-L score combines Precision, Recall, and a harmonic mean, influenced by the β parameter.

$$ROUGE - L = \frac{(1+ \beta^2)\ X\ Precision\ X\ Recall}{(\beta^2\ X\ Precision) + Recall} \quad (5\text{-}28)$$

5.8.3.3 TER

One of the most widely used metrics in evaluating machine translation systems is the Translation Edit Rate. It basically means the number of edits to get a machine-generated translation to the reference translation i.e. the ratio relative to the total number of words in the reference text. Edits involve insertions, deletions, substitutions, and shifts-word or phrase. TER is used in the evaluation of MT systems, providing an idea about the effort of post-editing a machine-generated output into a good enough reference.

$$TER = \frac{Number\ of\ edits}{Total\ number\ of\ words\ in\ the\ reference} \quad (5\text{-}29)$$

TER directly corresponds to the number of edits required for translation improvement, making it highly interpretable. A lower TER score indicates a higher-quality translation, as fewer edits are needed. Generally, a TER score below 0.20 (20%) is considered indicative of high-quality machine translation. TER complements other metrics like BLEU and ROUGE by providing an edit-focused perspective on translation accuracy.

5.8.3.4 SacreBLEU

The SacreBLEU score is a metric used to evaluate how good a machine-generated translation is by comparing it to reference translations created by humans. SacreBLEU

is mainly used in natural language processing, especially for tasks like machine translation, where it helps measure how closely the machine's output matches the reference translations. One of the reasons SacreBLEU is preferred over the traditional BLEU score is that it standardizes many of the elements involved in the evaluation process, such as how the text is tokenized, how punctuation is handled, and how reference translations are formatted. This ensures that the results are consistent and comparable across different systems and studies. BLEU scores can differ depending on how these elements are treated in each implementation, which can make comparisons difficult. SacreBLEU solves this problem by fixing these variables, so the scores are more reliable. It uses the same formula as the original BLEU score that is based on n-gram precision with a brevity penalty to account for short translations.

5.8.4 MOS

MOS stands for Mean Opinion score that measure used in the domain of quality. For the calculation of this metric, a set of rating scales is defined typically in the range 1 to 5. An audio sample in this case is given to the 'N' number of people to rate the quality. The recorded rating from all the individuals is summed and the average is calculated. The formula is given as follows.

$$MOS = \frac{\sum_{n=1}^{N} R_n}{N} \tag{5-30}$$

Where R_n is the rating by each person and N is the total number of people participated

In the evaluation of quality of synthesized speech as in FastSpeech 2, Mean Opinion Score (MOS) is a measure that is used. MOS uses listeners' rating of speech quality on a scale of 1 to 5, in which 5 is superior, 4 good, 3 fair, 2 poor and 1 is bad. This metric targets to derive the perception of people on how different aspects of speech quality look live; natural, clear and intelligible. MOS is very important in TTS as it indicates how actual users rate the quality of the synthesized speech. Higher MOS values indicate better performance. An MOS of 4.5 or higher is typically considered excellent, while an MOS of 4.0 is very good. MOS is valuable because it provides direct feedback from human listeners, which is essential for developing and fine-tuning speech synthesis models to produce more natural and intelligible speech.

5.8.5 SIM

The SIM (Speaker Similarity) metric evaluates how closely the generated audio matches with the actual speaker's voice. The metric seems to be crucial for voice cloning tasks where preserving the vocal traits of a speaker is important. A pre-trained speaker verification model is basically used to extract high-dimensional embeddings that help to capture the speaker-specific characters from both the generated audio (e_{gen}) and reference audio (e_{ref}). SIM is mathematically represented using cosine similarity.

$$SIM = \cos(\theta) = \frac{e_{gen}\, e_{ref}}{||e_{gen}||||e_{ref}||} \tag{5-31}$$

Here, $e_{gen}.\ e_{ref}$ is the dot product of the two embedding vectors, and $||e_{gen}||\ ||e_{ref}||$ their respective magnitudes (Euclidean norms). The result of this calculation lies in the range of [−1, 1], where a value closer to **1** indicates high similarity, **0** indicates no similarity, and values closer to **-1** indicate complete dissimilarity.

6. IMPLEMENTATION DETAILS

The implementation starts with a careful phase of preparing the data, ensuring it's properly structured for smooth integration and optimal performance across the ASR, NMT, and TTS systems.

6.1 Data Processing for ASR

A robust ASR system relies on efficient and systematic data processing. In this project, a structured pipeline was developed to preprocess audio datasets for training, ensuring data quality and consistency. Raw audio data, sourced from repositories such as OpenSLR and Common Voice, underwent a comprehensive cleaning and refinement process. This included removing empty or corrupted entries, trimming silence, reducing background noise, and standardizing transcription formats to enhance model performance. The following diagram provides a visual representation of the complete workflow, illustrating the meticulous transformation of raw datasets into high-quality, structured inputs optimized for ASR training.

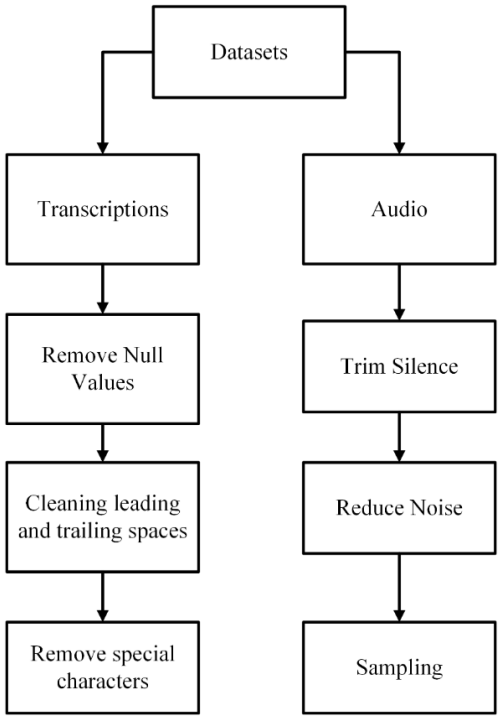


Figure 6-1: Data Pre-processing Pipeline

6.1.1 Audio Data Collection

Audio data were collected from various sources such as OpenSLR and Common Voice. From the OpenSLR, SLR43 [4] and SLR143 [8] were used and from the Common Voice [15], only validated speech was taken.

6.1.2 Audio Pre-processing

To ensure high-quality input for the Automatic Speech Recognition (ASR) system, the collected audio data from OpenSLR and Common Voice underwent a series of preprocessing steps. These steps included resampling, silence trimming, and noise reduction, all aimed at enhancing the clarity and consistency of the speech signals. The audio was also normalized to keep the volume levels consistent across the dataset, and irrelevant sections were cut out to focus only on the spoken content. By minimizing background noise and enhancing the overall quality, these steps help make the speech signals clearer for the ASR system, ultimately improving its accuracy.

6.1.3 Silence Trimming

The audio signal was segmented into smaller chunks, and the RMS energy was computed for each segment. This RMS value was then compared against a predefined silence threshold. The beginning of a non-silent segment was identified as the point where the RMS value exceeded the threshold. Simultaneously, the total duration of silence preceding this point was recorded. To ensure accuracy, the same procedure was applied in reverse, processing the audio from the end to the beginning. Finally, the detected silent regions were removed from the original audio, resulting in a cleaner and more concise speech signal.

6.1.4 Reducing the Noise

To minimize background noise, the audio signal was divided into short, overlapping frames, which were transformed into the frequency domain using the Fourier Transform. A noise spectrum estimate was obtained from silent or low-energy portions of the audio. This estimated noise spectrum was then subtracted from the overall audio spectrum, effectively reducing noise interference. The cleaned frequency-domain

representation was subsequently converted back to the time domain using the Inverse Fourier Transform, yielding a clearer speech signal.

6.1.5 Resampling the Audio

To standardize the sampling rate across different datasets, the audio signals were down sampled to 16 kHz. The resampling process involved passing the signal through a low-pass filter with a cutoff frequency of 8 kHz to prevent aliasing. The decimation ratio was then determined based on the original sampling rate. According to this ratio, a subset of samples was selected while others were discarded. For instance, with a decimation ratio of 3, every third sample was retained while the rest were removed. This ensured that all audio data was consistently processed at the target 16 kHz sampling rate, optimizing it for ASR model training.

6.2 Data Processing for NMT

High-quality data is essential for training an effective Neural Machine Translation (NMT) model. For this project, the Nepali-English parallel sentences dataset was collected. The details of this process are outlined in the following sections.

6.2.1 Dataset Collection

The dataset for the Nepali-to-English NMT model was curated to ensure high-quality bilingual sentence pairs. The final dataset comprised 24,240 sentence pairs, with each pair consisting of a sentence in Nepali and its corresponding English translation. The dataset was primarily constructed by collecting and generating sentences related to travel scenarios in Nepal, focusing on how a traveler might seek information about different locations. Key attributes of various places were identified, and monologue-style dialogues were manually composed in Nepali. These sentences were then carefully translated into English to maintain linguistic and contextual accuracy. Additionally, to enhance dataset diversity, some sentences were synthetically generated using a Large Language Model. However, all machine-generated sentences underwent manual verification and corrections to ensure fluency, accuracy, and naturalness in both source and target languages. This combination of manually created, translated, and verified synthetic data resulted in a well-balanced dataset optimized for training the NMT model.

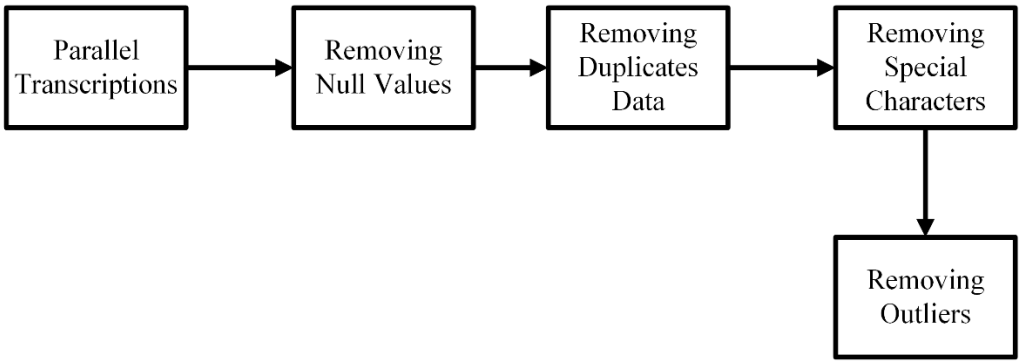


Figure 6-2: Data Processing For NMT

6.2.2 Dataset Cleaning and Preprocessing

The raw dataset, collected from multiple sources, was initially consolidated into a single directory for streamlined processing. To ensure data quality and consistency, the dataset underwent a rigorous preprocessing pipeline. This involved filtering out incomplete, empty, or duplicate sentence pairs, removing excessively short (fewer than three words) or long (more than 100 words) sentences, and standardizing text formatting. For Nepali text, special attention was given to preserving the integrity of the Devanagari script while eliminating special characters, extraneous symbols, and inconsistent formatting. These preprocessing steps ensured that the dataset remained clean, meaningful, and uniform, ultimately improving model performance.

6.2.3 Removing Null Values

Entries with null values in the transcription field were removed, as missing data can cause errors during preprocessing and model training. Since neural models require complete inputs, filtering out null entries ensured that only valid and meaningful transcriptions were retained, reducing inconsistencies and preventing pipeline disruptions.

6.2.4 Removing Duplicates

Duplicate sentence pairs can introduce bias by over-representing certain linguistic patterns, potentially leading to overfitting and poor generalization on unseen data. By eliminating redundant entries, the dataset maintained diversity and ensured each sample contributed equally to the learning process, resulting in a more balanced and unbiased dataset.

6.2.5 Removing Special Characters

Special characters, such as symbols and emoji, are generally irrelevant for text-based transcription tasks and can introduce unnecessary noise into the dataset. Removing these characters helped clean the text, retaining only essential linguistic components. This step improved the model's ability to focus on meaningful patterns, reducing complexity and enhancing learning efficiency.

6.2.6 Removing Outliers

Outliers, such as excessively long, unusually short, or nonsensical transcriptions can distort model learning by introducing atypical patterns. To maintain dataset consistency, such outliers were identified and removed. This step ensured that the training data reflected standard and meaningful linguistic structures, leading to more effective and stable model training.

6.3 Data Processing for TTS

The performance of a language model such as mBART is heavily dependent on the quality of the dataset used for training. When working with multilingual data, it is crucial to ensure that the data is clean, consistent, and well-structured. This section outlines the steps taken to prepare datasets for mBART, emphasizing the importance of thorough preprocessing in achieving optimal results in natural language tasks.

6.3.1 Dataset Collection

To develop the TTS model, a parallel Nepali-English audio dataset was manually recorded. The recordings were conducted by a diverse group of speakers, consisting of two males and three females. The total duration of the collected dataset was approximately 4.5 hours, comprising 2,132 audio files, with an average length of 6.16 seconds per file.

In order to generate transcriptions for the recorded audio, parallel NMT dataset was utilized. Additionally, textual data was sourced from the Nepali book *"आधुनिक नेपाली निबन्धमाला"* by Dr. Rishi Ram Sharma (2077 B.S.). The selected text was manually translated into English, and both the Nepali and English versions were recorded by the

speakers. This process ensured the creation of high-quality bilingual speech data, which is essential for training an effective TTS model.

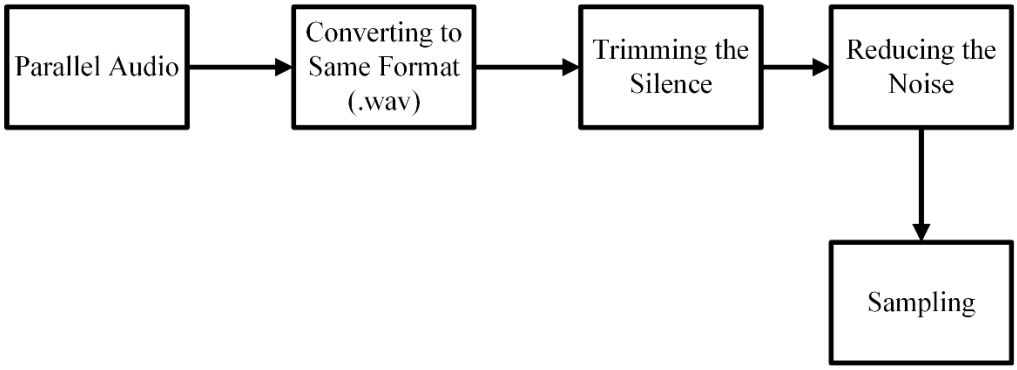


Figure 6-3: Data Processing For TTS

6.3.2 Dataset Processing

The recorded audio dataset underwent a systematic preprocessing pipeline to enhance consistency and quality. The key steps involved in preprocessing were format standardization, silence trimming, noise reduction, and resampling.

6.3.3 Format Standardization

The raw audio recordings were captured in various formats, including .m4a and .mp3, depending on the recording devices used by the speakers. These format inconsistencies posed challenges in processing and training. To ensure uniformity and compatibility across the dataset, all audio files were converted to the .wav format. This standardization facilitated a structured input for the model, improving processing efficiency.

6.3.4 Silence Trimming

Many audio recordings contained unnecessary leading and trailing silence, which increased the dataset size and introduced redundant information. To optimize dataset efficiency and improve training performance, these silent segments were removed. Silence trimming ensured that only relevant speech data was retained, reducing the overall dataset size and improving model performance.

6.3.5 Noise Reduction

Background noise in the recordings could potentially degrade the model's ability to learn accurate speech representations. To improve the clarity and quality of the audio data, noise reduction techniques were applied. These included spectral subtraction and filtering, which effectively reduced unwanted background noise. This step significantly enhanced the overall dataset quality, leading to better speech synthesis results during model inference.

6.3.6 Sampling

The audio recordings were captured at varying sampling rates, which introduced potential inconsistencies during preprocessing and model training. To ensure uniformity, all recordings were resampled to a standard sampling rate of 16 kHz. The resampling process included the application of a low-pass filter with a cutoff frequency of 8 kHz to prevent aliasing, followed by down sampling based on a calculated decimation ratio. This step ensured a consistent input representation, facilitating smoother and more effective model training.

6.4 Fine-tuning ASR

Fine-tuning an ASR system involves transforming raw audio data into meaningful text representations. A crucial aspect of this process is constructing a comprehensive vocabulary and a tokenizer that can accurately interpret and process the audio input. This section outlines the key steps required to prepare audio transcriptions for training advanced ASR models.

6.4.1 Vocabulary Generation

The vocabulary generation process is essential for converting audio transcriptions into a format that the model can comprehend. This involves creating a list of unique characters from the transcriptions and assigning each character a specific numerical value. The unique characters are systematically sorted, and special tokens such as unknown tokens and padding tokens are incorporated with designated indices. To enable the model to process textual data, it must be converted into numerical representations. Each unique character in the vocabulary is assigned a distinct index,

facilitating the conversion of text into a numerical sequence. The appendix provides a table displaying the extracted unique characters and their corresponding indices.

6.4.2 Tokenizer Initialization

The tokenizer plays a critical role in converting input text into tokens that the model can process. Utilizing the generated vocabulary, the tokenizer maps characters to their respective indices, ensuring efficient text-to-token conversion for ASR model training.

6.4.3 Data Collator

The data collator is responsible for dynamically padding both input values and labels during the training process. This ensures that all input sequences within a batch maintain uniform length. Specifically, the data collator pads each audio array to match the length of the longest audio segment within the batch. Similarly, textual transcriptions are padded to align with the longest text sequence in the dataset.

6.4.4 Hyperparameters

Hyperparameters play a crucial role in regulating the training process and enhancing model performance. Key parameters such as learning rate, batch size, and the number of training steps significantly impact the efficiency and accuracy of the ASR model. Proper tuning of these parameters ensures optimal performance while maintaining computational efficiency.

To achieve the best configuration for the ASR model, a structured hyperparameter tuning process was implemented. The primary objective was to identify the optimal set of hyperparameters that maximize model accuracy while ensuring efficient resource utilization. For this purpose, Bayesian Optimization was employed as the tuning strategy.

6.4.4.1 Bayesian Optimization for Hyperparameter Tuning

Bayesian Optimization is a highly efficient technique for hyperparameter tuning that leverages a probabilistic model to predict the performance of different hyperparameter configurations. Unlike exhaustive search methods such as grid search or random search, Bayesian Optimization utilizes previous evaluations to intelligently determine the next

set of hyperparameters to test. This approach significantly reduces the number of trials required for convergence, leading to faster and more effective tuning. By systematically applying Bayesian Optimization, the hyperparameter tuning process was optimized to improve the ASR model's overall performance, ensuring accurate speech recognition while maintaining computational efficiency.

A detailed table in the appendix outlines the hyperparameters utilized in the training process.

Table 6-1: Hyperparameters for ASR

Hyperparameters	**Values**
Train Epochs	100
Batch Size	2
Gradient Accumulation	2
Evaluation Steps	250
Learning Rate	1e-5
Learning Rate Scheudler	linear
Weight Decay	0.01
Metrics	WER

6.5 Fine-tuning mBART

mBART models consist of two primary variants: "facebook/mbart-large-cc25" and "facebook/mbart-large-50". The "facebook/mbart-large-cc25" model is the original version, trained on 25 languages using the CommonCrawl corpus. The "facebook/mbart-large-50" model extends the original model's embedding layers and was further trained on 50 languages to enhance its multilingual capabilities. In this project, the "facebook/mbart-large-cc25" model has been utilized.

6.5.1 Tokenization

To enable the mBART model to process raw text data, it must first be converted into smaller units known as tokens. These tokens may represent words, subwords, or individual characters. For this project, the tokenizer provided with the mBART model was employed. This specialized tokenizer supports multiple languages by incorporating language-specific tokens, allowing it to handle diverse text inputs for tasks such as translation and multilingual text generation.

The tokenizer must correctly interpret the language context for both input and output sequences. To achieve this, mBART employs special language code tokens, such as <en_XX> for English and <ne_NP> for Nepali, which indicate the source and target languages. The tokenizer also truncates sequences to a specified maximum length to ensure uniform input sizes, which is essential for batch processing during training and inference. The tokenization process consists of two main steps encoding which converts text into token IDs that the model can process and decoding which converts token IDs back into human-readable text to interpret the model's output. Additionally, the tokenizer handles preprocessing tasks such as lowercasing, removing special characters, and ensuring consistent formatting, thereby improving the efficiency of the model training process.

6.5.2 Padding the Input

In the training process of the mBART model for translation, a critical step involves padding the input sequences. This step ensures uniformity in the lengths of input sequences, which is essential for efficient batch processing during training. Specifically, shorter sequences within a batch are padded with a special token, typically denoted as `<pad>`, until they match the length of the longest sequence in the batch. This padding allows the model to handle variable-length inputs consistently, enabling parallel computation across the batch on GPU or TPU hardware. However, to prevent the model from treating padding tokens as meaningful input, a masking mechanism is applied during attention computations, ensuring that the padded positions do not contribute to the loss calculation or influence the model's predictions. This padding step is fundamental to maintaining the scalability and performance of mBART across diverse multilingual datasets.

6.5.3 Hyperparameter

Bayesian optimization is a powerful and efficient strategy for hyperparameter tuning in machine learning models, when dealing with computationally expensive evaluations, such as training large neural networks or complex models. In the context of optimizing the performance of our model, Bayesian optimization leverages a probabilistic model typically a Gaussian Process to model the objective function that maps hyperparameters to model performance. By iteratively balancing exploration and exploitation, it selects hyperparameter configurations that are likely to yield improvements, reducing the number of trials needed compared to grid or random search methods. The process begins with an initial set of random hyperparameter combinations, after which it uses the acquired data to update its surrogate model and acquire function, guiding the search toward promising regions of the hyperparameter space. The following table depicts the hyperparameters used in our study, including learning rate, batch size, dropout rate, and number of layers, each of which was systematically tuned using Bayesian optimization to maximize model performance while minimizing computational cost. The following table shows the key hyperparameters tuned in our study, such as learning rate, batch size, dropout rate, and number of layers, each of which was optimized using Bayesian techniques to achieve the best balance of performance and efficiency.

Table 6-2: Hyperparameters for mBART

Hyperparameters	Values
Train Epochs	10
Train Batch Size	4
Gradient Accumulation	4
Evaluation Strategy	Epoch
Learning Rate	3e-5
Weight Decay	0.01
Metrics	sacreBLEU

6.6 Finetuning the TTS Model

Fine-tuning the F5 TTS model involves adapting its pre-trained architecture to a specific dataset to improve speech synthesis performance for a particular language, or domain. The model utilizes a ConvNext-based text refinement module, a Diffusion Transformer, and a Sway Sampling Strategy to generate high-quality speech. The fine-tuning process requires a structured approach involving dataset preparation, tokenization, reference audio processing, diffusion model training, and vocoder optimization. Each of these components must be carefully adjusted to ensure seamless integration and optimal speech synthesis results.

6.6.1 Tokenization and Embedding Initialization

To enable the model to process text inputs effectively, the text is converted into smaller units known as tokens. These tokens may represent words, subwords, or individual characters, depending on the tokenizer used. The F5 TTS model utilizes a multilingual tokenizer that incorporates language-specific tokens, allowing it to handle various languages efficiently. Once tokenized, the text is mapped to numerical representations through an embedding layer. This embedding layer is initialized with pre-trained weights, which help retain previously learned knowledge and improve training stability. Using a well-trained embedding layer ensures that the model captures meaningful relationships between linguistic elements, leading to more natural speech synthesis.

6.6.2 Reference Audio Processing and Concatenation

One of the key aspects of the F5 TTS model is its ability to leverage reference audio to guide speech synthesis. The reference audio undergoes feature extraction to capture important acoustic attributes such as pitch, speaker identity, and energy levels. These extracted features are then concatenated with the processed text embedding, creating a composite input representation that helps the model maintain speaker consistency and prosody. This process allows the model to generate speech that closely resembles the characteristics of the reference audio, making it particularly effective for voice cloning and speaker adaptation tasks. This method not only enhances the naturalness of the synthesized speech but also enables more personalized and context-aware speech generation, making it highly effective for applications where speaker identity and emotion need to be retained.

6.6.3 Finetuning the Diffusion Transformer

The Diffusion Transformer plays a critical role in learning the relationship between text embeddings and mel-spectrogram features, which serve as an intermediate representation of speech. Unlike traditional autoregressive models, the diffusion-based approach introduces controlled noise into the training process and then learns to reverse this noise step by step to reconstruct high-quality speech. This process enhances the model's ability to capture fine-grained speech details and maintain consistency across different linguistic contexts. During fine-tuning, the diffusion model is optimized using loss functions such as MSE, which measures the difference between predicted and ground-truth spectrograms. Training stability is maintained through techniques such as gradient clipping and adaptive learning rates to prevent issues like mode collapse and overfitting.

6.6.4 Sway Sampling Strategy Implementation

To ensure high-quality speech synthesis, the Sway Sampling Strategy is fine-tuned to optimize the sampling efficiency. This strategy helps balance the trade-off between speech diversity and naturalness, ensuring that the generated speech does not contain artifacts or distortions. By adjusting the sampling rate and noise reduction parameters, the model can generate speech outputs that sound more realistic and expressive. The fine-tuning process for this component involves experimenting with different noise schedules and evaluating their impact on the clarity and fluency of synthesized speech.

6.6.5 Mel-Spectrogram Generation and Vocoder Finetuning

After the diffusion model generates a mel-spectrogram, the next step is to convert it into an audio waveform using a vocoder. The vocoder is responsible for transforming the frequency-domain representation into a time-domain signal, making it intelligible and natural. Pre-trained vocoders such as HiFi-GAN, WaveGlow, or Parallel WaveGAN are often used, but fine-tuning them on the target dataset ensures that the generated speech aligns with the desired voice characteristics. During this stage, loss functions such as Mel Spectrogram Loss and GAN-based losses are used to optimize both fidelity and perceptual quality. By fine-tuning the vocoder, the model produces speech that is clear, expressive, and free from unnatural robotic artifacts.

6.6.6 Hyperparameter Tuning

To achieve optimal performance, various hyperparameters such as batch size, learning rate, gradient accumulation steps, and noise schedules need to be fine-tuned. The process of hyperparameter tuning is conducted using techniques such as methods like Bayesian Optimization, Grid Search, or Random Search. Among these, Bayesian Optimization stands out as a particularly effective approach, as it intelligently selects hyperparameter values based on the outcomes of previous trials, thereby reducing the number of iterations required to identify the best configuration. Regularization techniques such as dropout and weight decay are also applied to prevent overfitting. By carefully optimizing these hyperparameters, the model can achieve higher accuracy, improved speech naturalness, and faster convergence during training, making it more efficient. This meticulous process of tuning plays a vital role in the overall success of the model, ensuring it delivers high-quality results while balancing computational efficiency. As the model undergoes continuous evaluation, any adjustments made to the hyperparameters lead to refinements that enhance its robustness, allowing for better generalization to unseen data and real-world scenarios.

Table 6-3: Hyperparameters for TTS

Hyperparameters	Values
Train Epochs	100
Train Batch Size	8
Gradient Accumulation	4
Evaluation Strategy	Epoch
Learning Rate	1e-5
Weight Decay	0.01
Metrics	SIM

6.7 Use Case Diagram

A Use Case Diagram helps to provide a better understanding of the system's working. The above diagram illustrates the project system. Users can record the speech and upload audio files containing Nepali speech, which further transcribes it into text with speech-to-text conversion.

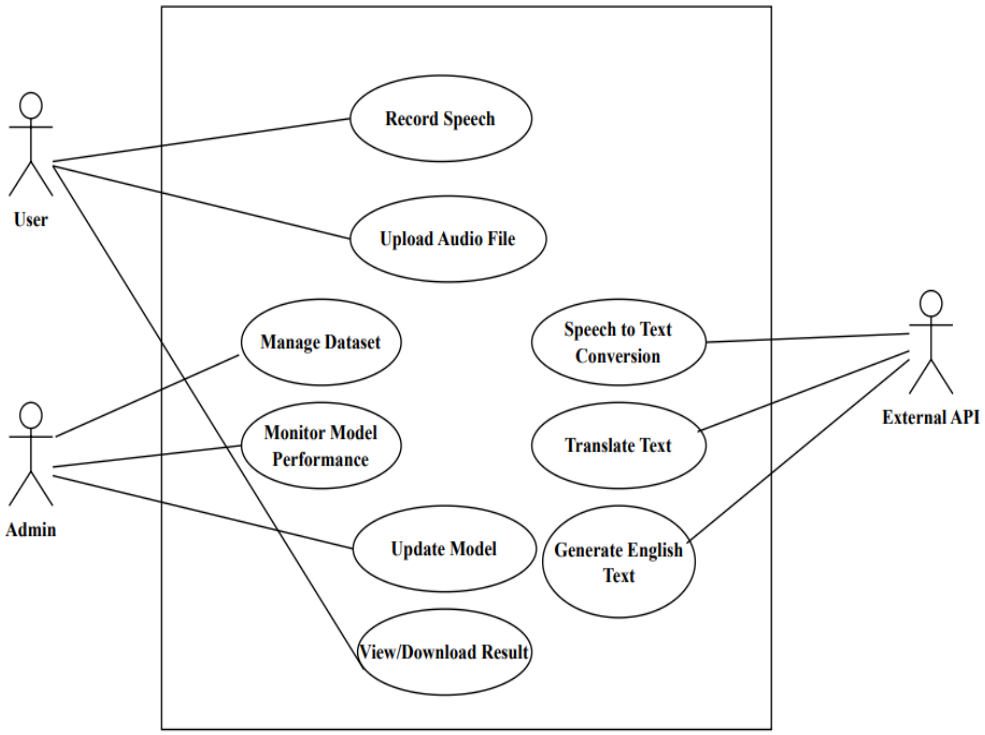


Figure 6-4: Use Case Diagram

The translated text into English is converted into speech. Users can view or download the result in audio format. The system allows the management of training datasets and enables the updating of data for the enhancement of model performance to the administrators. Also, they can monitor the key performance metrics like accuracy and loss by updating the machine learning models to improve the system's effectiveness based on these metrics.

7. RESULT AND ANALYSIS

A comprehensive overview of the models' performance in this project is presented below. Detailed graphs and metrics have been mentioned to illustrate the effectiveness of the model.

7.1 Dataset and Result Analysis for ASR

The various data characteristics for ASR model has been presented below alongside its analysis.

7.1.1 Available Dataset Exploration for ASR

The dataset for the ASR was gathered from three distinct sources: OpenSLR43, OpenSLR143, and Mozilla Common Voice. This resulted in a total of 3511 audio files accompanied by their corresponding transcriptions. The distribution of audio files among these sources was as follows: OpenSLR43 contributed the largest share with 2064 audio files, OpenSLR143 provided 675 files, and Mozilla Common Voice added 772 files. This diverse collection ensured a broad representation of speech variations and nuances, which is crucial for developing a robust ASR system. The collected audio files varied in length, with an average duration of 4.3 seconds. About 1110 audio files were longer than or equal to 5 seconds, while the remaining 2401 files were shorter than 5 seconds.

Table 7-1: Dataset Available for ASR

Source	**Number of Audio**
OpenSLR43	2064
OpenSLR143	675
Common Voice	772
Total	3511

There was variation in the dataset as it was collected from different sources, and several preprocessing steps were undertaken to achieve uniformity across the audio files. The

primary challenges addressed during preprocessing included differences in sampling rates, variations in loudness, and the presence of leading and trailing silences. The dataset was not symmetrical in terms of the length distribution of the audio files. While the majority of the audio files were less than 5 seconds long, a significant portion exceeded this duration.

This imbalance can affect the training process, potentially biasing the model towards shorter or longer utterances. To counter this, the audio data were padded during the training process which made the audio of equal length. The padding of the audio data can mitigate the effect of the variable audio length.

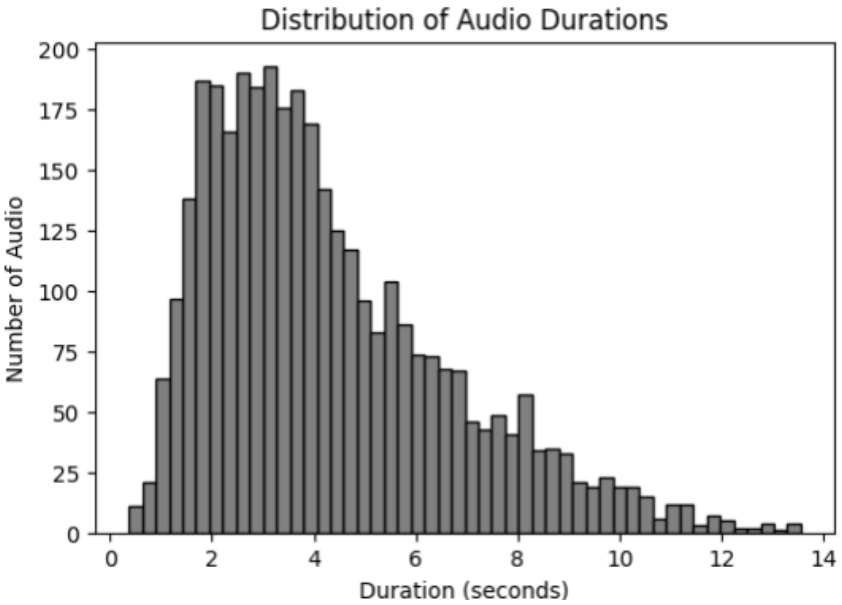


Figure 7-1: Distribution of Audio Durations

The following two graphs show the distribution of the dataset before and after processing. Before the pre-processing of the data, the loudness ranges from 0 to around 0.175 RMS. The distribution is wider and more spread out, which indicates there is a higher variance in loudness levels among the audio samples. After preprocessing, the loudness range is much more compressed, ranging from 0 to around 0.14 RMS. The distribution is narrower and more concentrated around the lower RMS values, indicating a more uniform loudness level after preprocessing.

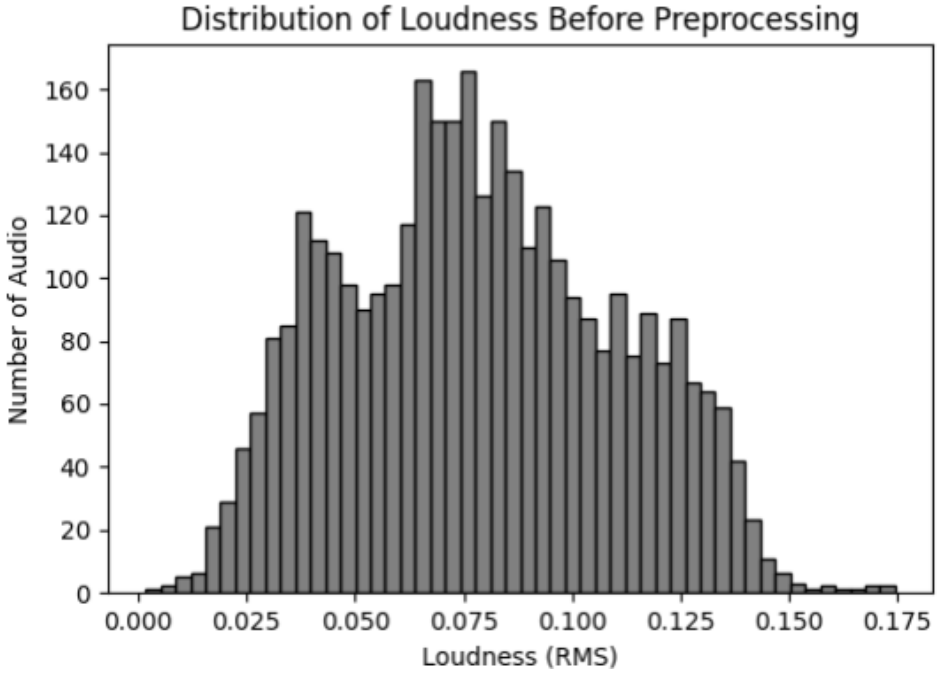


Figure 7-2: Distribution of Loudness Before Preprocessing

The loudness levels of the audio files were also inconsistent due to the different recording conditions and equipment used. Normalizing the loudness across all audio files helped mitigate the variations, thereby enabling the model to focus on the speech content rather than the amplitude differences.

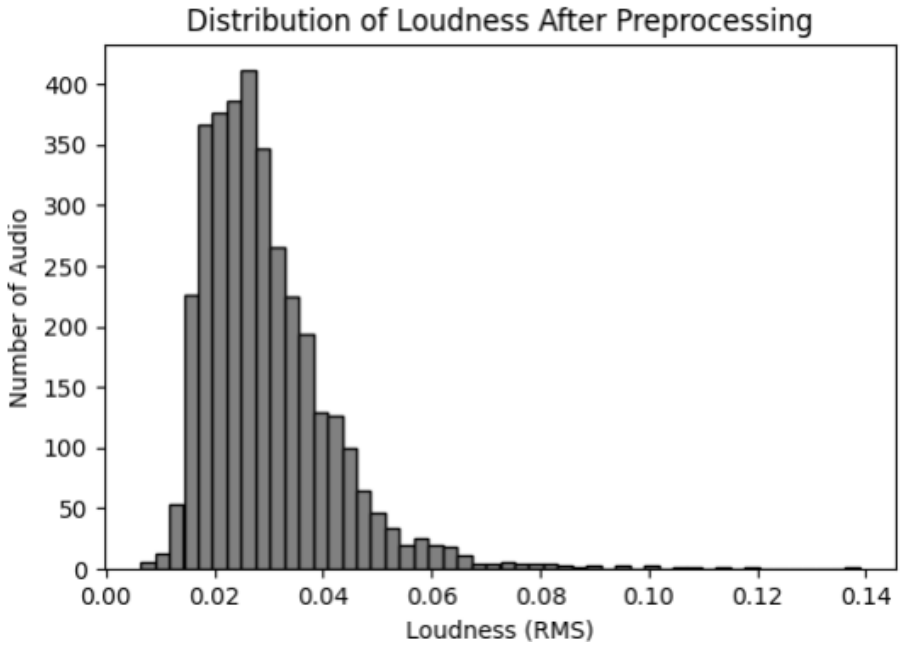


Figure 7-3: Distribution of Loudness After Preprocessing

The audio files sourced from different databases had varying sampling rates. A consistent sampling rate is vital for effective feature extraction and model training. Therefore, all audio files were resampled to a uniform rate of 16KHz to ensure compatibility and consistency in the dataset.

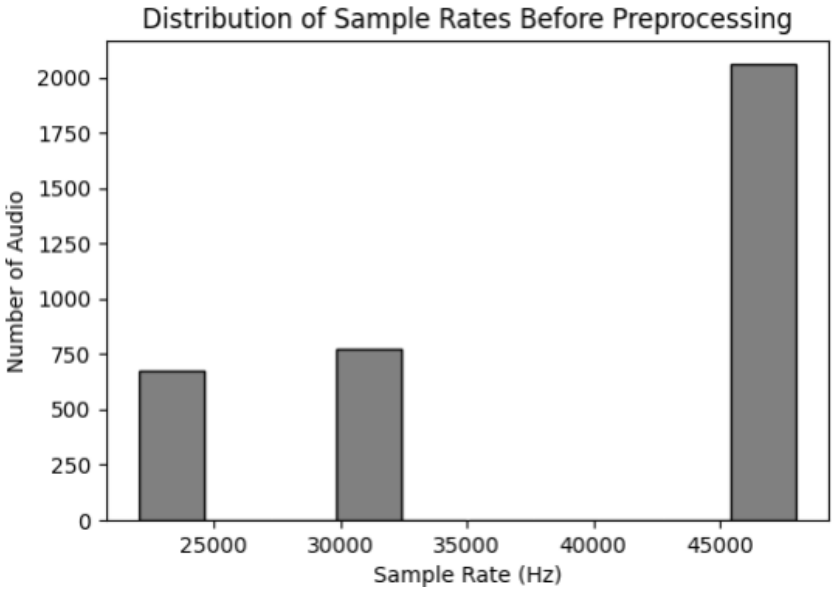


Figure 7-4: Distribution of Sample Rates Before Preprocessing

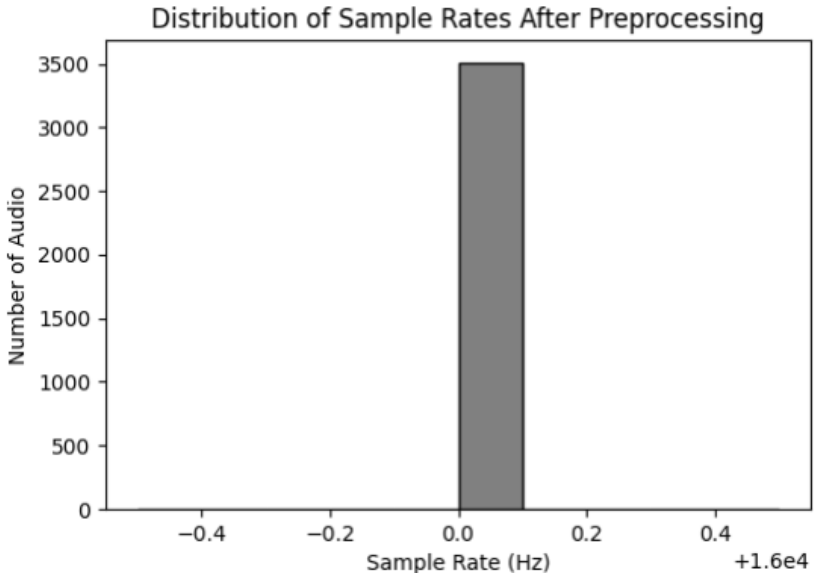


Figure 7-5: Distribution of Sample Rates After Preprocessing

Among all the datasets prepared, there are about 2753 audio that contain the voice of the female speaker remaining and 758 audio that contain the male speaker.

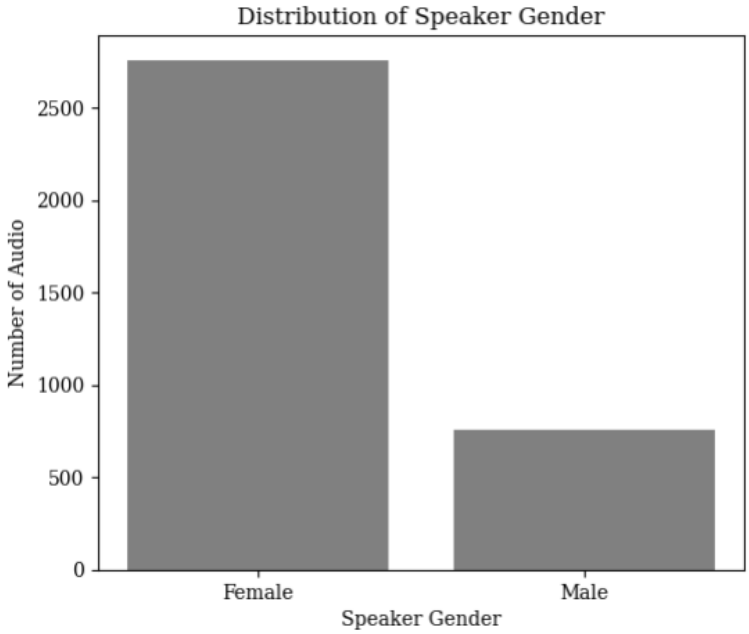


Figure 7-6: Distribution of Speaker Gender

Leading and trailing silences in the audio files can introduce noise and affect the model's performance. By trimming these silences, the dataset was refined to contain more relevant speech segments.

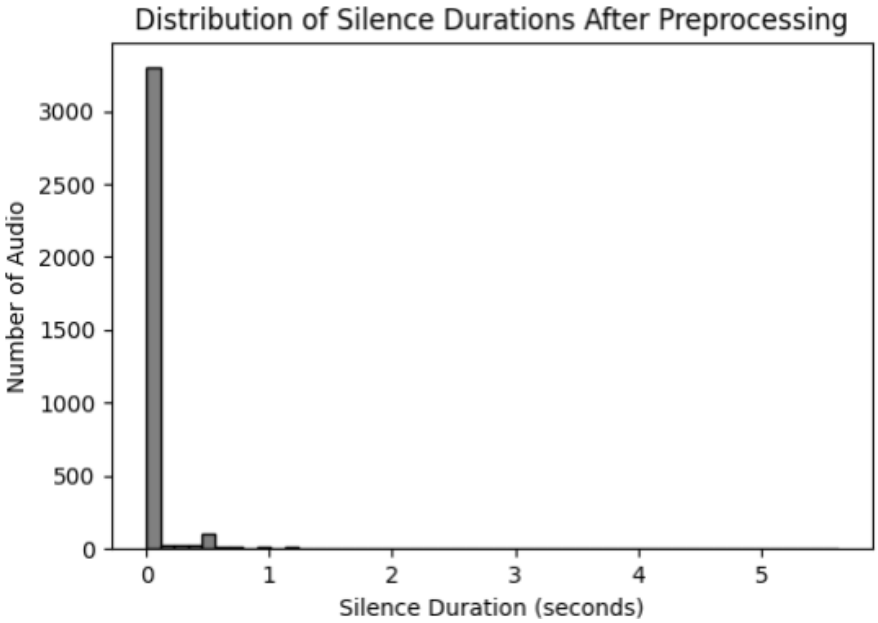


Figure 7-7: Distribution of Silence Duration After Preprocessing

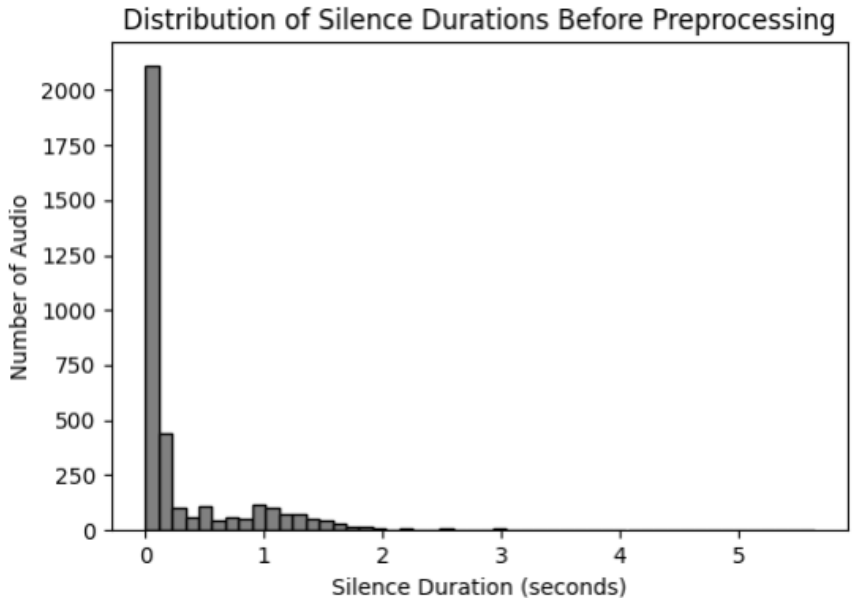


Figure 7-8: Distribution of Silence Duration Before Preprocessing

7.1.2 Transcription of Data Exploration

The table below shows the longest and shortest transcriptions, along with the frequency of the most common words, providing a quantitative and qualitative understanding of the dataset. The longest transcription contains 179 characters. This suggests that the dataset includes long and complex sentences, which are essential for training the ASR model to handle longer and more complex sentences. The shortest transcription is a single word with 4 characters. The presence of very short transcriptions indicates that the dataset also includes brief utterances, which are crucial for the training of model to accurately transcribe short responses or commands.

Table 7-2: Longest and Shortest Sentence in ASR

	Sentence	**Length**
Longest	म्याडम क्युरी फलोरेन्स नाइटिङ्गेल जुनको ताबेइ आदि यस्ता व्यक्तित्व हुन् जसले आफ्नो प्रतिभाको माध्यमबाट आफूलाई विश्वभर चिनाउन समर्थ भए र यिनी सबै नारी नै हुन् जसले शिक्षा पाएका थिए	179
Shortest	सेवा	4

The table below presents the ten most frequently occurring words in the dataset, offering valuable insights into common usage patterns and the linguistic structure of the transcriptions. These words primarily consist of functional and frequently used terms in the Nepali language, indicating a diverse representation of everyday speech patterns. Notably, words such as "छ", "हो", and "पनि" appear with high frequency, with "छ" occurring 495 times and "हो" appearing 450 times. Additionally, words like "यो", "एक", and "छन्" are observed with moderate frequency, with occurrences ranging between 132 and 178. The presence of both high-frequency and moderately occurring words contributes to the richness of the dataset, ensuring that the dataset consists of a wide range of commonly used linguistic elements. The dataset encompasses a balanced mix of frequently, moderately, and less frequently occurring words, reflecting a well-distributed linguistic diversity essential for robust ASR model performance. The diversity in transcription length and word frequency is beneficial for training a SR model. The presence of both long narratives and short utterances ensures that the model can handle varying speech lengths. Additionally, the frequent occurrence of common functional words alongside specific terms indicates that the model will be well-equipped to understand and transcribe everyday conversations.

Table 7-3: Most Frequent Words in ASR

Word	**Frequency**	**Word**	**Frequency**
छ	495	यो	178
हो	450	एक	162
पनि	263	छन्	142
रहेको	220	सय	132
भएको	207	जिल्लामा	132

The analysis of rare words in the transcription dataset reveals the presence of words with very low frequency, indicating their limited occurrence within the dataset. These rare words play a significant role in assessing the potential challenges they may pose for the Automatic Speech Recognition (ASR) model. The table presents a list of words that appear only once in the dataset. The inclusion of such rare words suggests that the

dataset encompasses a broad vocabulary, including less commonly used terms. This diversity is essential for training an ASR model capable of handling a wide range of linguistic variations. However, the infrequent occurrence of these words may pose challenges for the model, as it might struggle to learn and accurately transcribe them due to the limited exposure during training.

Table 7-4: Rare Words in ASR

Word	**Frequency**	**Word**
आइरहन्छ	1	डुङ्गा
झट्का		जाऔ
भूकम्पको		लगाको
बगाउन		ओहो
भान्छा		कुर्सी

7.1.3 Outputs from ASR

The evaluation of the ASR model on the validation set provides useful insights into its accuracy and highlights areas that need improvement. The analysis is based on 281 validation examples, where we compare the true and predicted words and their frequencies. The table offers a breakdown of the most common errors, showing words that the model misclassified frequently. For example, the word "र" was predicted incorrectly 150 times, while "छ" was misclassified 139 times. These errors point out where the model has trouble recognizing common words, which suggests areas for improvement in transcription accuracy. By examining these errors further, we can see specific challenges the model faces, such as distinguishing between similar-sounding words or handling different speech variations, like accents. We also measured the word error rate (WER) and sentence error rate (SER), which helped us understand where the model struggles the most, particularly with less common words or more complex sentences. Tackling these recurring misclassifications will be crucial in improving the ASR system's overall performance, making it more reliable in everyday use.

Table 7-5: True and Predicted Words in Validation Set

True	Predicted	Frequency
र	हो	150
छ	र	139
र	छ	138
हो	र	131
छ	हो	123
हो	छ	113
पनि	र	79
पनि	हो	73
र	पनि	71
पनि	छ	70

Table 7-6: True and Predicted Words in Test Set

True	Predicted	Frequency
जोशीको	जोसीको	2
सूची	सुची	
कुन्नि	नि	
भयो	भय	
यी	ी	
नजिकै	नजीकै	
बढी	बढि	
ठूलो	ठुलो	
सम्पुर्ण	सम्पूर्ण	

The above table presents examples of true words and their corresponding predicted words, along with their respective frequencies. The model tends to replace "सम्पुर्थ" with

"सम्पूर्थ" on two occasions, suggesting a potential challenge in differentiating between words that sound similar or distinguishing shorter forms from their longer counterparts. The occurrences of deletions and insertions, each occurring five times, are relatively infrequent. However, despite their rarity, these errors still impact the overall WER and can significantly affect the accuracy of the transcriptions.

The Word Error Rate (WER) for the test set is 0.4188, representing the proportion of words that were predicted incorrectly by the model. With 703 test examples, this WER value indicates that there is potential for further improvement in the model's accuracy. The dataset contains both high-frequency common words and rare words, highlighting the importance of adopting a balanced approach to model training. Although the ASR model performs reasonably well with common words, the relatively high WER suggests that additional efforts are needed to enhance its overall performance.

This metric reflects the proportion of words that were incorrectly substituted, deleted, or inserted by the model. There were 2041 substitutions, indicating that the model frequently replaced one word with another inappropriately. Additionally, there were 5 deletions, demonstrating instances where the model failed to transcribe words that were present in the audio. Similarly, there were 5 insertions, where the model added extra words that were not included in the audio.

7.1.4 Graphs

The graph illustrates the training and validation loss of an ASR model over a series of training steps, as depicted in the provided plot. Loss is a measure of how well the model's predictions align with the actual data, is shown on the y-axis, with lower values indicating better performance. The x-axis represents the number of training steps, ranging from 0 to 50,000.

Both the training loss (solid line) and validation loss (dashed line) start at high values, which is typical since the model begins with random weights and is far from optimal. As shown in the graph, both losses exhibit a sharp decline in the early stages (up to around 10,000 steps), indicating rapid learning as the model adjusts its weights to minimize the loss. After this initial drop, the training loss continues to decrease gradually but at a slower rate, reflecting the model's ongoing improvement on the

training data. Similarly, the validation loss decreases, suggesting that the model is generalizing effectively to unseen data.

As the training progresses, both the training and validation losses flatten out, particularly after approximately 30,000 steps, as seen in the plateau toward the right side of the graph. This flattening indicates that the model is approaching its minimum possible loss on both the training and validation datasets, and additional training steps are unlikely to yield significant improvements. The convergence of the training and validation losses, with a small and narrowing gap between the two curves, suggests that the model is well-fitted to the data and not overfitting.

The small gap between the training and validation loss curves throughout the graph provides insight into the model's strong generalization ability. The rapid decrease in loss during the initial stages, followed by a gradual leveling off, reflects an efficient training process where the model quickly learns the most critical features of the data and then fine-tunes its weights. Overall, the graph indicates that the ASR model is well-optimized for the given data, with no evident signs of significant overfitting or underfitting..

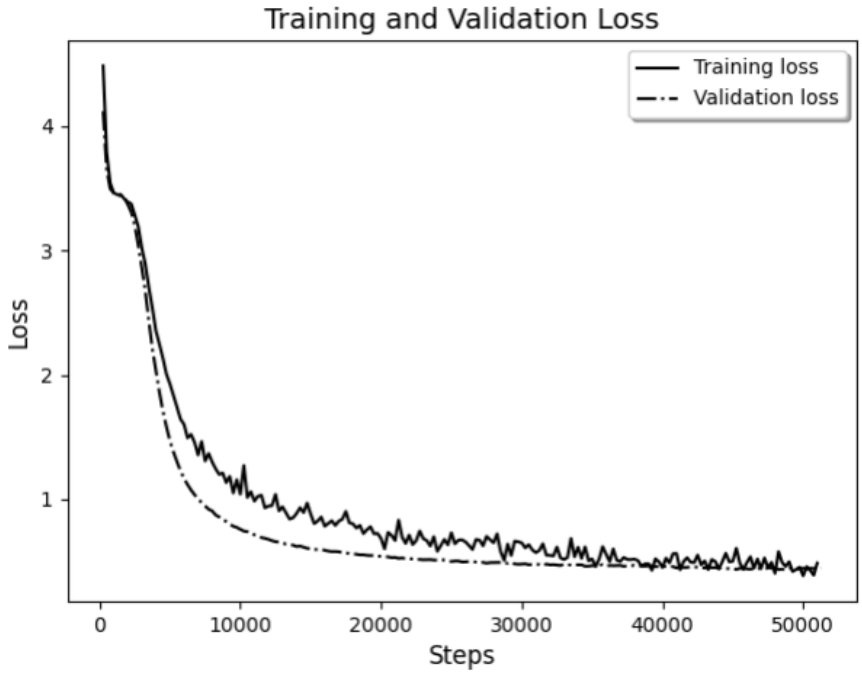


Figure 7-9: Training and Validation Loss in ASR

The graphs illustrate the WER on the evaluation set over a series of training steps for an ASR model. As depicted in the plot, the WER starts at an exceptionally high value close to 1, signaling extremely poor initial performance as the model, beginning with random weights or suboptimal parameters, likely makes a significant number of errors, struggling to accurately transcribe speech. There is a rapid decline in WER during the initial training steps, clearly demonstrating that the model swiftly identifies and learns the most critical and impactful features necessary for substantially reducing errors and improving transcription accuracy. However, as training progresses further, the rate of decline in WER noticeably slows down, indicating that while the model continues to make incremental improvements, the most straightforward and easiest-to-learn patterns have already been largely captured, making further reductions in WER increasingly difficult and requiring finer adjustments. The WER curve eventually flattens out, as evident in the later stages of the graph, strongly suggesting that the model has nearly reached convergence, with additional training steps yielding only minimal and marginal improvements in performance. Despite this pronounced plateau, the WER still exhibits a subtle yet persistent downward trend, indicating some ongoing, gradual, and continuous enhancement in the model's ability to accurately recognize and transcribe speech. The final WER achieved, as specified in the data, is 0.44, reflecting a significant overall improvement from the initial performance but also highlighting the challenges of achieving further reductions at this advanced stage of training.

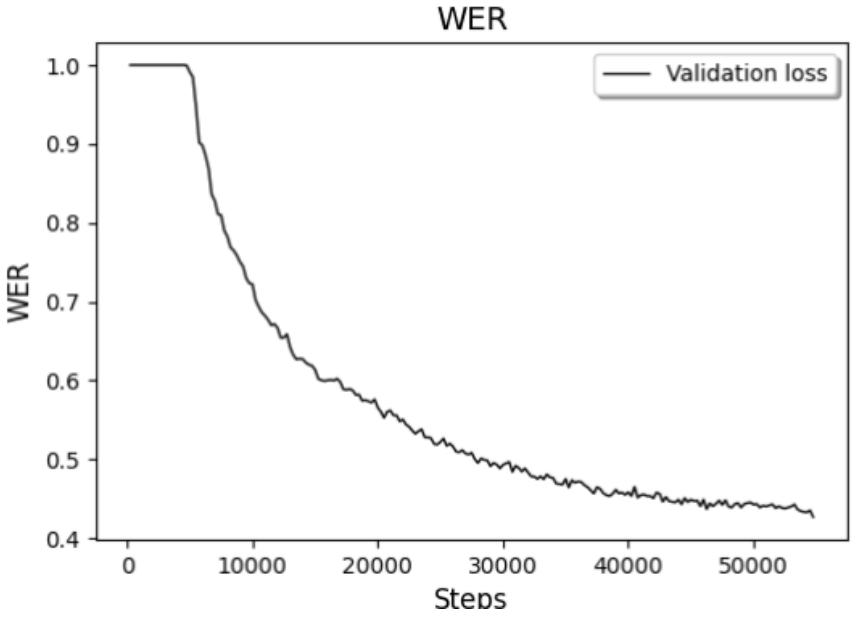


Figure 7-10: WER v/s Steps

The graph depicts the learning rate during the training process, following a linear learning rate scheduling strategy, as shown by the solid line labeled "Training Loss" on the y-axis (learning rate) and "Steps" on the x-axis (ranging from 0 to 50,000). The learning rate starts at a low value of 1×10^{-6}, helping stabilize the initial training phase and prevent large, destabilizing updates. It then increases linearly to a peak of approximately 1×10^{-5} early in the process, enabling the model to take larger steps, escape shallow local minima, and explore the parameter space effectively for faster convergence. After reaching this peak, the learning rate decreases linearly back to around 2×10^{-6} by the 50,000th step, allowing for fine-tuning with smaller, more precise updates. This warm-up phase (increasing learning rate) and subsequent decay ensure stability, prevent overshooting, and support optimal parameter adjustments throughout training.

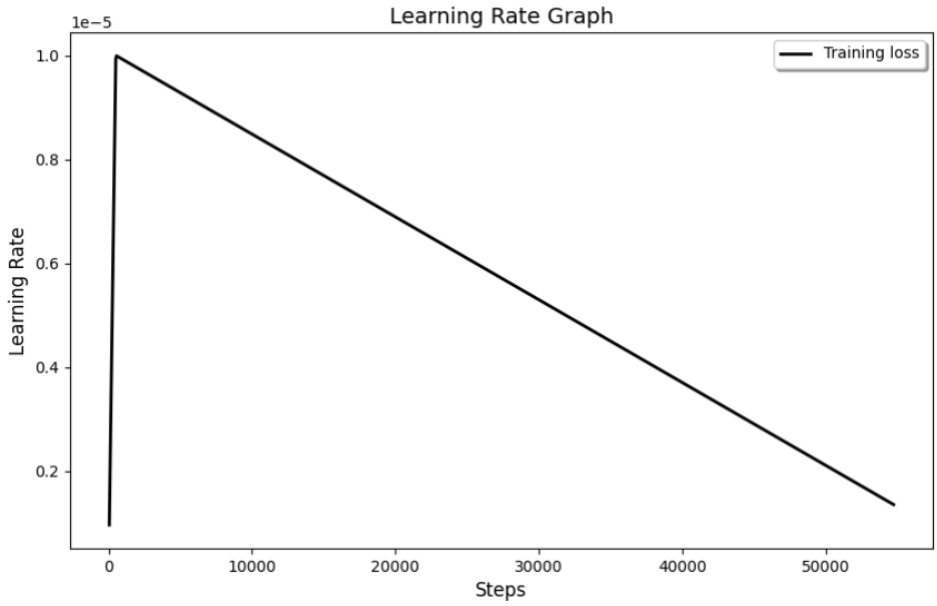


Figure 7-11: Learning Rate Graph

7.3 Dataset and Result Analysis for NMT Model

After cleaning, a total of 1901 bilingual sentence pairs is what was used for the Nepali-to-English transcription analysis, each pair consisting of a sentence in Nepali-in Devanagari script and its English translation. The linguistic diversity and variability within this dataset make it quite attractive for MT and ASR development. The average number of characters in Nepali transcription is 72 characters, while the average for an English transcription is 78, which shows slight verbosity in English.

Table 7-7: Most Frequently Used Words in NMT Model

Nepali Word	**Frequency**	**English Word**	**Frequency**
वा	16043	the	59045
र	9418	of	36000
गर्न	6520	to	27775
बमोजिम	6436	or	17825
कुनै	6372	and	17495
गर्ने	6101	in	15984
नेपाल	4953	be	11696
तथा	4866	a	11321
ऐन	4429	shall	10575
उपदफा	3760	by	7801

The word frequency analysis shows that in both the Nepali and English datasets, conjunctions, prepositions, and formal terms are most frequent, which suggests a structured and descriptive language style. . In Nepali, words like “र” (571 occurrences), “छ” (447 occurrences), and “तर” (270 occurrences) highlight the use of connectors and clauses, which could be part of descriptive or informational content about tourist

destinations. In English, common words such as “the” (2,144 occurrences), “shall” (10,575 occurrences), and “or” (17,825 occurrences) suggest a formal tone often used in guidelines, recommendations, or descriptions. Words like “यो” (290 occurrences), “यहाको” (222 occurrences), and “देख्दा” (188 occurrences) further indicate a localized focus on specific places or activities. The frequent appearance of these words suggests that the dataset likely contains structured content, such as travel guides, brochures, or regulatory documents related to tourism, with a focus on detailed and clear communication about travel destinations and services.

Table 7-8: Evaluation Metrics

Evaluation Metrics	**Validation Score**	**Test Score**
Bleu Score	0.4006	0.4083
Rouge 1 Score	0.70	0.6917
Rouge 2 Score	0.49	0.4868
RougeL Score	0.66	0.6655
SacreBleu Score	38.84	40.83
TER Score	43	41.75

These evaluation metrics give an insight into the generated text's performance, each measuring the quality from a different perspective. The BLEU score of 0.4006 shows a fair degree of n-gram overlap, hence the reasonable accuracy at the word level while matching the generated and reference texts. In contrast, the ROUGE-1 score is higher, 0.70, which implies the model performs well at capturing unigram overlaps (single words), which is a key measure of content recall. While ROUGE-2 with a score lesser than ROUGE-1, at 0.49, it shows that overall, a decent level of bigram overlap occurs, though there is room for improvement regarding capturing consecutive word pairs.

ROUGE-L, with a score of 0.66, underlines the model's capability to retain longer sequences and sentence-level structure, which is crucial for summarization and translation tasks. On the other hand, the SacreBLEU score of 38.84% agrees with the BLEU score, confirming that the model's n-gram overlap is reasonably consistent across evaluations. However, the TER score is 43%, which means that most of the generated text requires many edits to match the reference, meaning although it captures some key elements, there is substantial room for refinement in fluency and coherence. Overall, while the model works reasonably well for most metrics, the relatively high score for TER indicates further improvements in minimizing the need for edits, and enhancing text fluency could lead to more accurate and natural outputs.

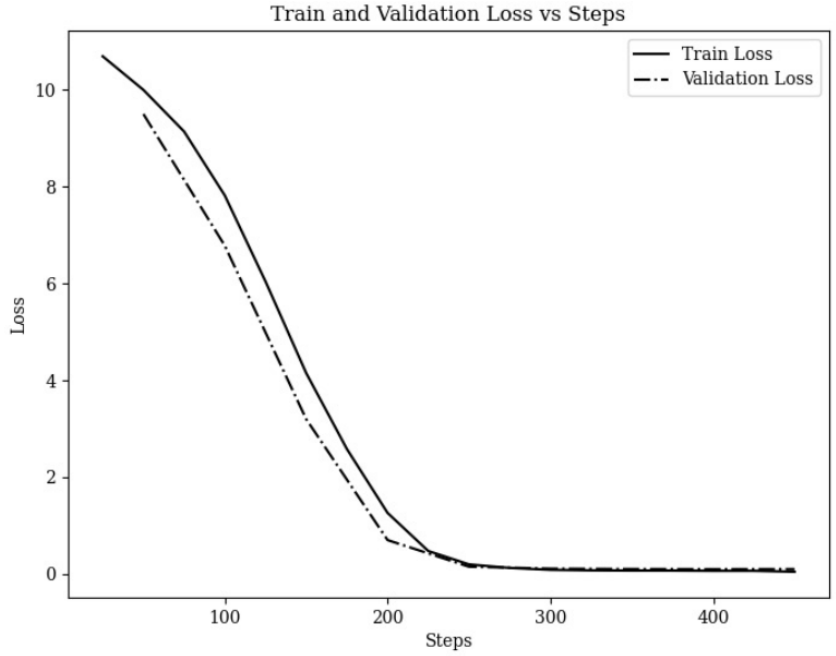


Figure 7-12: Training and Validation Loss vs Steps

The graph, titled "Train and Validation Loss vs Steps," illustrates the performance of a model during training, with the x-axis representing training steps (from 0 to 400) and the y-axis showing loss values. It features two curves: the training loss (solid line) and the validation loss (dashed line), both decreasing as training progresses. Initially, both losses start at a high value of around 10, indicating that the model begins untrained and makes significant errors. As the number of training steps increases, the losses drop sharply, particularly within the first 200 steps, demonstrating that the model is learning effectively and reducing errors for both the training and validation datasets. After 200

steps, both curves stabilize near zero, suggesting that the model has converged and generalizes well to unseen validation data. The close alignment of the training and validation losses throughout the graph indicates no overfitting, with the model performing consistently across both datasets. This smooth decline and convergence confirm a stable training process and reliable model performance

7.4 Dataset and Result Analysis for TTS Model

The dataset consists of a total of 932 audio files, the dataset encompasses recordings in both Nepali and English, contributed by five distinct speakers two male and three female. The aggregate duration of these audio files amounts to 15,883 seconds, equivalent to approximately 265 minutes of speech, representing the cumulative temporal extent of the dataset across all recordings. Given the relatively brief average duration of individual audio files, calculated at 6.16 seconds. The dataset is structured around multiple short phrases or sentences. This granularity enhances the dataset's suitability for such tasks, facilitating accurate transcription and processing by minimizing the complexity associated with longer, continuous speech inputs.

Table 7-9: Audio File Duration Analysis

Parameter	**Value**
Total Duration	15883seconds (approximately 265 minutes)
Average Length of Audio Files	6.16 seconds
Maximum Duration of Audio File	15.16 seconds
Minimum Duration of Audio File	1.34 seconds

The analysis of the duration range within the dataset reveals that the shortest audio file has a duration of 1.34 seconds, representing the briefest recorded segment. These short files may result from silent pauses, background noise, or very brief utterances. The longest audio file extends to 15.16 seconds, corresponding to more complex or detailed utterances, such as complete sentences or extended thoughts. Regarding speaker distribution, the dataset includes contributions from five speakers, comprising two male

voices and three female voices. This gender distribution enables further exploration of speech characteristics, such as pitch, energy levels, and speaking style. For example, male voices typically exhibit a lower pitch range compared to female voices, a difference that may manifest in audio analyses as variations in pitch, tempo, or spectral features. This dataset's analysis of duration and speaker distribution is highly valuable for identifying potential biases or imbalances, particularly when training speech recognition models. By ensuring exposure to a diverse range of voices, the dataset supports the development of robust and equitable models. Additionally, it facilitates the examination of gender-based differences in speaking patterns, which is critical for applications in speaker identification and classification. This comprehensive analysis enhances the preprocessing and preparation of the dataset, making it well-suited for a variety of speech-related tasks, including speech recognition, translation, and synthesis.

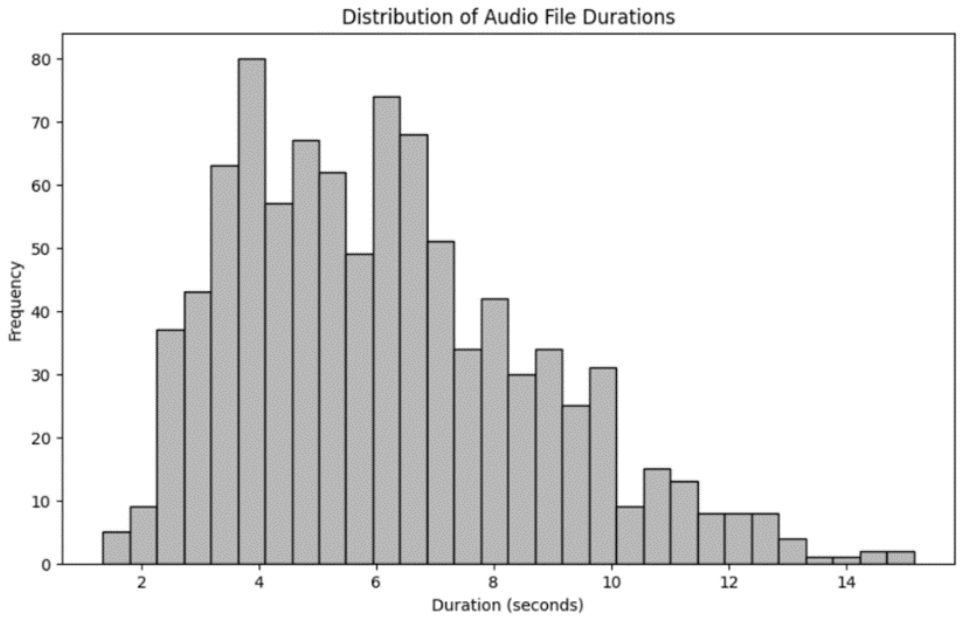


Figure 7-13: Distribution of Audio File Durations

The above graph illustrates the distribution of audio file lengths within the dataset. The x-axis represents the duration of the audio files in seconds, ranging from approximately 1 second to 15 seconds, while the y-axis indicates the frequency (number of files) corresponding to each duration range. The data reveals that most audio files are relatively short, with a prominent peak in frequency occurring between 4 and 6 seconds, reaching a maximum of approximately 80 files. As the duration extends beyond 6 seconds, the frequency decreases steadily, showing a gradual decline in the number of

longer audio files. By the time the duration exceeds 10 seconds, the frequency drops sharply, with very few files longer than 12 seconds. Overall, the distribution is right-skewed, characterized by a high concentration of short-duration files and a tapering off of longer-duration files.

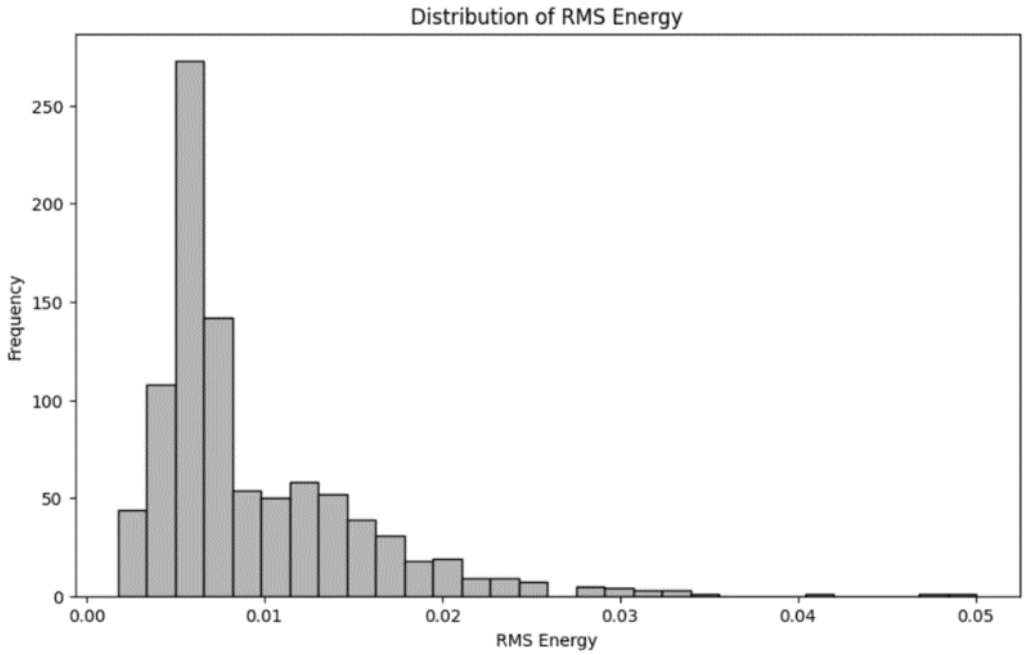


Figure 7-14: Distribution of RMS Energy

The above graph shows the distribution of RMS energy, a measure of signal intensity or loudness, within the dataset. The x-axis represents RMS energy values, ranging from approximately 0.00 to 0.60, while the y-axis indicates the frequency (number of occurrences) for each energy range. The data reveals that the majority of RMS energy values are concentrated at the lower end, specifically between 0.005 and 0.01, where the frequency peaks at over 250 occurrences. As the RMS energy increases beyond this range, the frequency decreases steadily, forming a long tail extending to the right. This right-skewed distribution indicates that most signals in the dataset have low energy, suggesting they are quiet or low in intensity, while only a small number of signals exhibit higher energy values. The presence of the long tail highlights the occasional occurrence of louder or more intense signals, though these are significantly less common compared to the quieter ones.

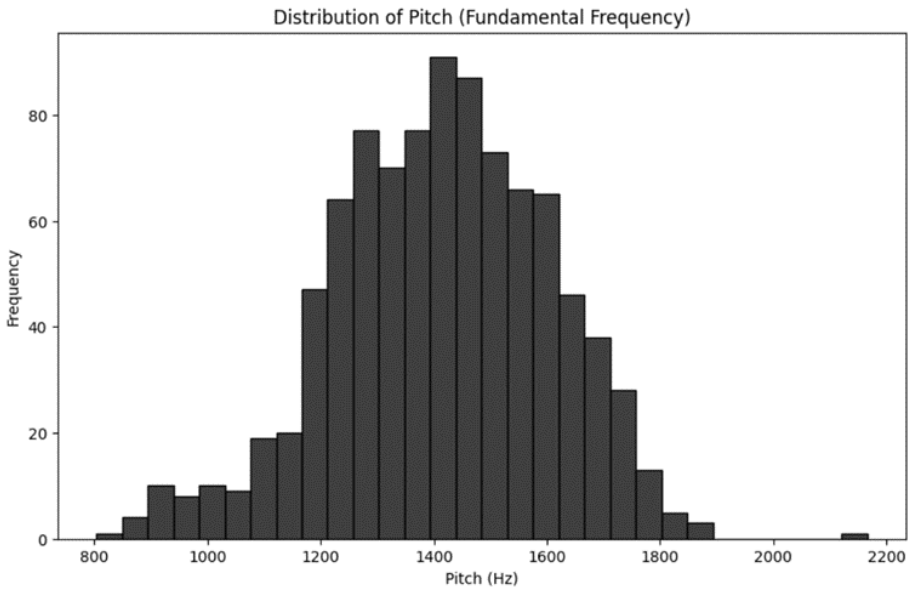


Figure 7-15: Distribution of Pitch

This above graph illustrates the distribution of pitch values within the dataset. The x-axis represents pitch values in Hertz, ranging from approximately 800 Hz to 2200 Hz, while the y-axis indicates the frequency (number of occurrences) for each pitch range. The data shows that pitch values are most concentrated between 1200 Hz and 1600 Hz, with the highest frequency occurring around 1400 Hz, where the graph peaks. The overall shape of the distribution is bell-shaped, resembling a hill, indicating that most voices in the dataset fall within this mid-range pitch. Very low pitches (below 1200 Hz) and very high pitches (above 1600 Hz) are less common, as evidenced by the lower frequencies at the extremes of the distribution. This pattern suggests that the voices in the dataset exhibit a natural and consistent pitch range, with fewer instances of extremely low or high pitch values. Such a distribution provides valuable insights into voice characteristics, which can be leveraged in speech technology, audio analysis, and the evaluation of consistency with natural speech behavior, as depicted in the histogram.

7.4.1 Transcription Data Exploration

The transcription dataset comprises 1,901 Nepali sentences and their English translations, offering a rich resource for multilingual analysis. Preprocessing involves removing non-alphanumeric characters, trimming spaces, and standardizing text to eliminate inconsistencies, ensuring reliability for further analysis. Key steps include cleaning, script normalization, and tokenization. Sentence lengths vary significantly,

with the longest Nepali sentence containing 222 characters and the longest English sentence 199 characters, reflecting the dataset's ability to capture both short phrases and complex narratives. This variability makes the dataset ideal for evaluating multilingual models like mBART, testing their performance across concise and elaborate language patterns. The dataset's diverse sentence structures provide valuable insights into Nepali-English usage, enabling accurate assessments in translation quality, sentiment analysis, and multilingual model evaluation.

Table 7-10: Longest and Shortest Sentence

	Sentence	**Length**
Longest Nepali	यो मन्दिरले त्यो ठीक ठाउँलाई चिन्ह लगाउँछ जहाँ महारानी मायादेवीले सिद्धार्थ गौतमलाई जन्म दिनुभएको थियो जसले पछि बुद्धत्व प्राप्त गर्नुभयो यहाँ उभिँदा यो ठाउँले मानव जातिमा पार्ने प्रभावको बारेमा सोच्दा मेरै रौं ठाडा भएको छ	222
Longest English	the temple marks the exact spot where queen maya devi gave birth to siddhartha gautama who later became the buddha just standing here gives me chills knowing the impact this place has had on humanity	199
Shortest Nepali	यस्तो विशेष स्वाद	17
Shortest English	it made me sad	14

The table below displays the frequency of common Nepali words in a parallel Nepali-English transcription dataset, highlighting their significance in shaping the text's structure and meaning. Words like "र" (and), "छ" (is), and "छन्" (are) are essential for

sentence construction, while “तर” (but) introduces contrast, and “यो” (this) specifies subjects or objects. Pronouns such as “मलाई” (to me) and expressions like “लाग्छ” (seems) convey personal perspectives or feelings. Descriptive terms like “देख्दा” (while seeing) and location-specific words like “यहाँको” (of here) add contextual and narrative depth to the dataset. The high frequency of these words underscores their role in understanding sentence flow, personal viewpoints, and contextual nuances, which are critical for enhancing ASR systems and speech synthesis in Nepali. The prevalence of these terms reflects the dataset’s conversational, everyday language, making it valuable for training models to handle real-world speech patterns, such as those in virtual assistants or translation systems. Understanding the frequency and usage of these words facilitates improvements in transcription accuracy, enables NLP models to grasp contextual variations, and ultimately enhances user experience.

Table 7-11: Most Common Nepali Words

Word	**Frequency**	**Word**	**Frequency**
र	571	छन्	211
छ	447	देख्दा	188
यो	290	मलाई	184
तर	270	हो	182
यहाँको	222	लाग्छ	172

The table below lists the ten most frequent English words in a given text, along with their frequencies. The words "the," occurring 2,144 times, "of," occurring 947 times, and "in," occurring 450 times, are prepositions or articles crucial for sentence construction. Similarly, "is," with 432 occurrences, and "are," with 282 occurrences, are forms of the verb "to be," essential for forming statements and expressing existence or actions. The conjunction "and," appearing 601 times, commonly connects ideas or elements, while "to," with 590 occurrences, functions as both a preposition and a component of infinitive verbs. The indefinite article "a," occurring 470 times, and the adverb "here," occurring 312 times, often denoting location, provide additional context. Lastly, "with," appearing 271 times, is a preposition indicating relationships between

entities or actions. Despite their simplicity, these words are vital for linking ideas and structuring sentences, ensuring coherence and fluency in the text. Known as stop words, they are frequently excluded in text analysis due to their lack of significant standalone meaning. However, within sentences, they play a key role in syntactic structure and comprehension.

Table 7-12: Most Common English Words

Word	**Frequency**	**Word**	**Frequency**
the	2144	in	450
of	947	is	432
and	601	here	312
to	590	are	282
a	470	with	271

The table below displays the ten least frequent Nepali words in the transcription dataset, each appearing only once. These uncommon words highlight the presence of specialized vocabulary within the text.

Table 7-13: Least Common Nepali Words

Word	**Frequency**	**Word**
अजेय	1	जयकार
रंगशालामा		तपाईका
फुटबल		प्रतिध्वनित
खेल		प्रतिस्पर्धाको
भीडको		उत्सवजस्तो

For instance, अजेय suggests strength or dominance, while जयकार implies celebration. Terms such as रंगशालामा and फुटबल point to sports-related contexts, though they are

infrequently used. The word तपाईका, a polite form of address, typically appears in formal interactions. Additional words like प्रतिध्वनित, खेल, and प्रतिस्पर्धाको relate to events and activities, while भीडको and उत्सवजस्तो occasionally depict scenes of public gatherings. Their rare usage is indicative of the richness of the language in the dataset.

The table below shows the 10 least frequent words in the given dataset of English, each with a frequency of 1, which itself shows how rare these words are in the text. ‘hero’, ‘match’, and ‘stadium’ are terms that are generally associated with sports or competitions. ‘crashing’ and ‘cheers’ denote action or emotion, possibly related to exciting moments or celebrations in the dataset. ‘unstoppable’ conveys strength or dominance, referring to something or someone who cannot be defeated. ‘ears’ can be used for sensory perception but may be used metaphorically in some instances. ‘football’ and ‘competition’ are clearly related to sports and competitive environments, but are infrequent in the dataset. The presence of such uncommon words indicates the highlight of specific themes and rich text for analysis, showing that it will also include some references to sports, celebrations, and intense actions. However, they are rather scarcely used within the whole corpora. Their infrequent use can indicate that they deal with special situations or issues, further enriching diversity in the language.

Table 7-14: Least Common English Words

Word	**Frequency**	**Word**
Hero	1	match
Satasidham		stadium
Crashing		cheers
Unstoppable		ears
Football		competition

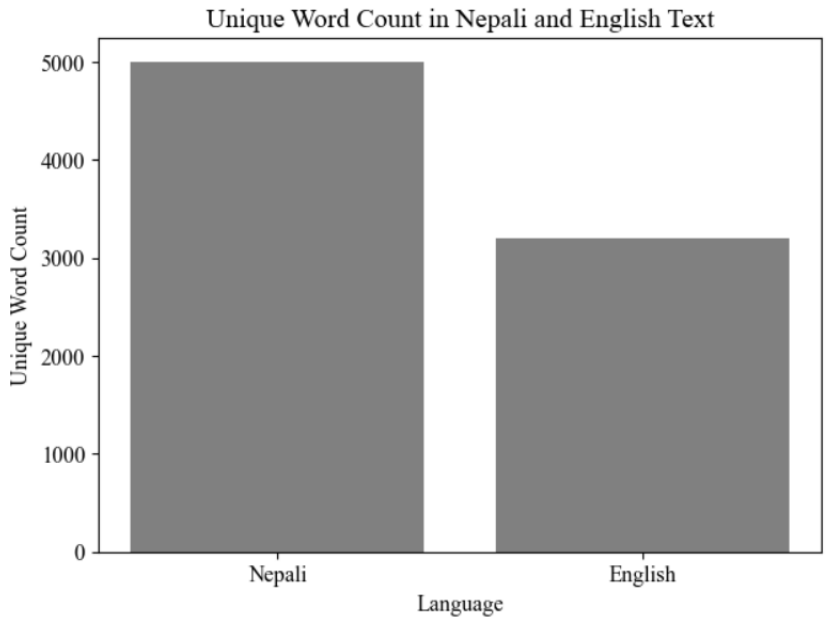


Figure 7-16: Unique Words Count in Nepali and English Text

The above bar chart shows the number of unique words in Nepali and English, providing valuable insights into their linguistic characteristics. Nepali exhibits a higher unique word count of 5,102 compared to 3,252 in English, indicating greater lexical diversity and a broader vocabulary. This disparity suggests that Nepali, as an agglutinative language, frequently employs synonyms, varied expressions, and compound words, potentially reflecting structural differences from English. The increased unique word count in Nepali may also point to a more complex or diverse corpus, with a higher prevalence of context-dependent or culturally specific terms, posing challenges for translation between the two languages. This linguistic richness implies that learners of Nepali may encounter a wider variety of vocabulary, underscoring the language's complexity and the difficulty of achieving fluency. Overall, the chart offers critical insights into language richness, text complexity, and translation challenges, making it highly relevant for corpus analysis, language processing, and multilingual translation applications.

The dataset, themed around travel blogs, highlights the richness and diversity of language used to describe Nepal's cultural and geographical landscapes. It captures vivid experiences and detailed descriptions of various regions, including local customs, traditions, and unique attractions across Nepal's provinces. The Nepali text reflects the depth and variety of local culture, while the English translation effectively conveys

these experiences to a global audience. The analysis emphasizes the importance of careful language processing and translation strategies in travel blog writing to preserve Nepal's regional nuances and diversity, ensuring relatability for both local and international readers.

7.4.2 Experimentation with TTS Model

The TTS model is an integral part of the speech-to-speech translation pipeline, responsible for converting written text into natural-sounding spoken words. These models leverage advanced deep learning techniques to synthesize speech that is both intelligible and expressive. In this project, various state-of-the-art TTS models were explored, including FastSpeech2, Tacotron 2, SpeechT5, and F5, to evaluate their suitability for the task of Nepali-to-English translation, particularly in handling domain-specific terms. All models were trained on the same dataset, which included approximately 1066 audio English audio samples. These samples were carefully curated to include diverse linguistic features and domain-specific vocabulary, such as the names of districts and other culturally significant terms, to test the models' ability to generalize and synthesize accurate pronunciation. The training process for each model involved fine-tuning different hyperparameters to optimize performance. Key parameters such as learning rate, batch size, and the number of training epochs were adjusted to match the specific requirements of each architecture.

7.4.2.1 Result for FastSpeech2

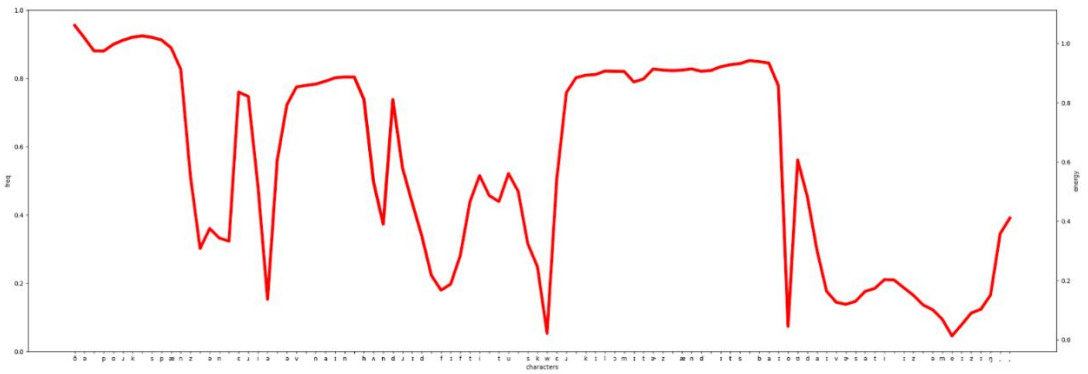


Figure 7-17: Average Energy Ground Truth in FastSpeech2

The above graph shows the ground energy, which describes the natural energy distribution according to speech across phonemes or characters. This is important, as it

shows a way of comparing differences in energy naturally existing in speech, where peaks correspond to stressed or emphasized sounds, while the valleys correspond to softer or less intensely stressed utterances. The importance of this is that it forms the baseline for evaluating the energy prediction quality of the text-to-speech system and also, capturing the natural prosody of speech as the basis for the nuances in speech dynamics, intending to generate more expressive and human-like speech.

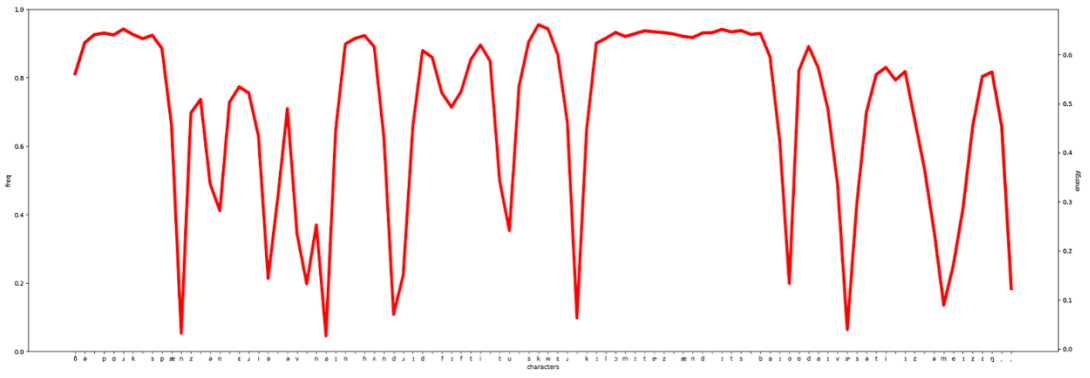


Figure 7-18: Average Energy Predicted in FastSpeech2

This graph illustrates the average predicted energy, representing the energy levels generated by the text-to-speech model based on its input, plotted over time for comparison with the ground truth. When compared to the ground truth, which depicts the natural energy distribution of speech with distinct, sharp peaks corresponding to stressed or emphasized sounds and deep valleys indicating softer, less intense utterances, the predicted energy shows some alignment but also significant discrepancies: the predicted graph exhibits smoother, less intense peaks and shallower valleys, suggesting the model underestimates the dynamic range of natural speech energy, while the timing of peaks often deviates, indicating challenges in accurately capturing the temporal prosody of speech. These differences highlight areas where the model struggles to replicate the finer nuances of natural speech dynamics, such as the precise intensity and timing of stressed syllables or subtle energy variations that contribute to human-like expressiveness, suggesting the need for further training or adjustments potentially through additional data, fine-tuning, or advanced prosodic features to better emulate the richness and variability of natural speech as shown in the ground truth..

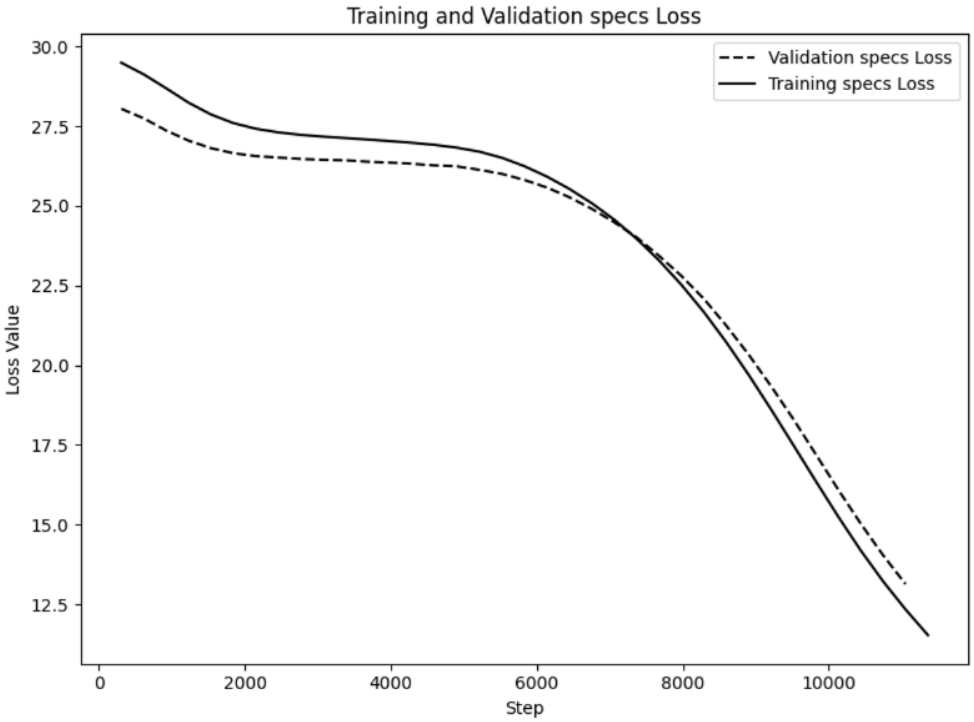


Figure 7-19: Training and Validation Specs Loss in FastSpeech2

This graph presents the training and validation loss curves for a machine learning model over 10,000 training steps, demonstrating the reduction in loss as training progresses, which signifies effective model optimization. The solid line depicts the training loss, while the dashed line illustrates the validation loss; both curves begin at an initial loss value of approximately 30 and exhibit a consistent decline throughout the training process. However, a analysis reveals that, contrary to an initial assumption of close convergence after 6,000 steps, the validation loss consistently remains slightly higher than the training loss, with a noticeable gap persisting even at 10,000 steps. Specifically, by the end of training, the training loss stabilizes around 14, while the validation loss hovers between 15 and 16. This divergence suggests potential minor overfitting or differences in the data distribution between the training and validation sets. While the overall downward trend in both losses indicates successful learning, the persistent gap underscores the need for further improvements in model generalization..

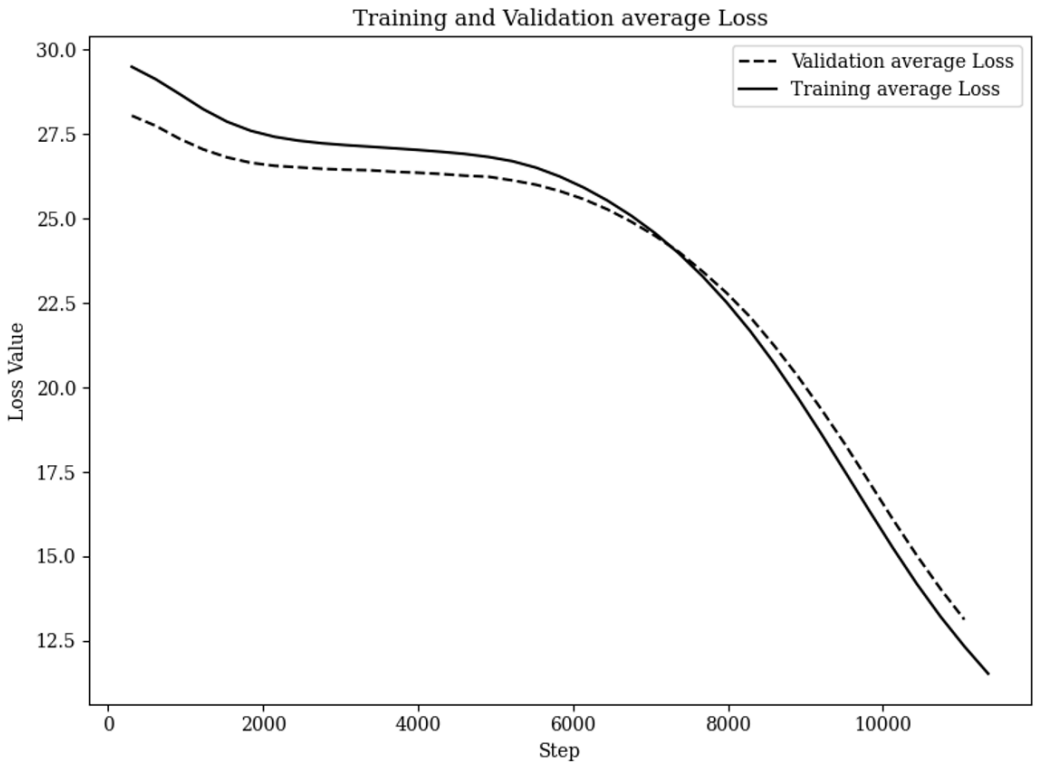


Figure 7-20: Training and Validation Average Loss in FastSpeech2

This graph presents the average loss curves for the training and validation phases of a text-to-speech model over 10,000 training steps, offering insight into the model's learning progress and performance optimization. The average loss, defined as the discrepancy between the model's predicted outputs such as mel-spectrogram predictions, duration, and energy and the actual target outputs, serves as a comprehensive metric encompassing errors across these key parameters. The initial loss values are approximately 30, with both the solid line representing the training average loss and the dashed line represents the validation average loss starting near 30 and declining steadily over the training steps. By the end of the 10,000 steps, both curves stabilize around 14 for training loss and 15 for validation loss, indicating a consistent reduction in error but with a persistent gap between training and validation losses. This downward trend reflects the model's successful adaptation to the training data, as the minimization of loss signifies improved accuracy in generating speech features, resulting in more natural and precise synthesized speech. However, the slight but noticeable divergence between training and validation losses where the validation loss remains higher suggests potential minor overfitting or differences in data distribution, warranting further investigation.

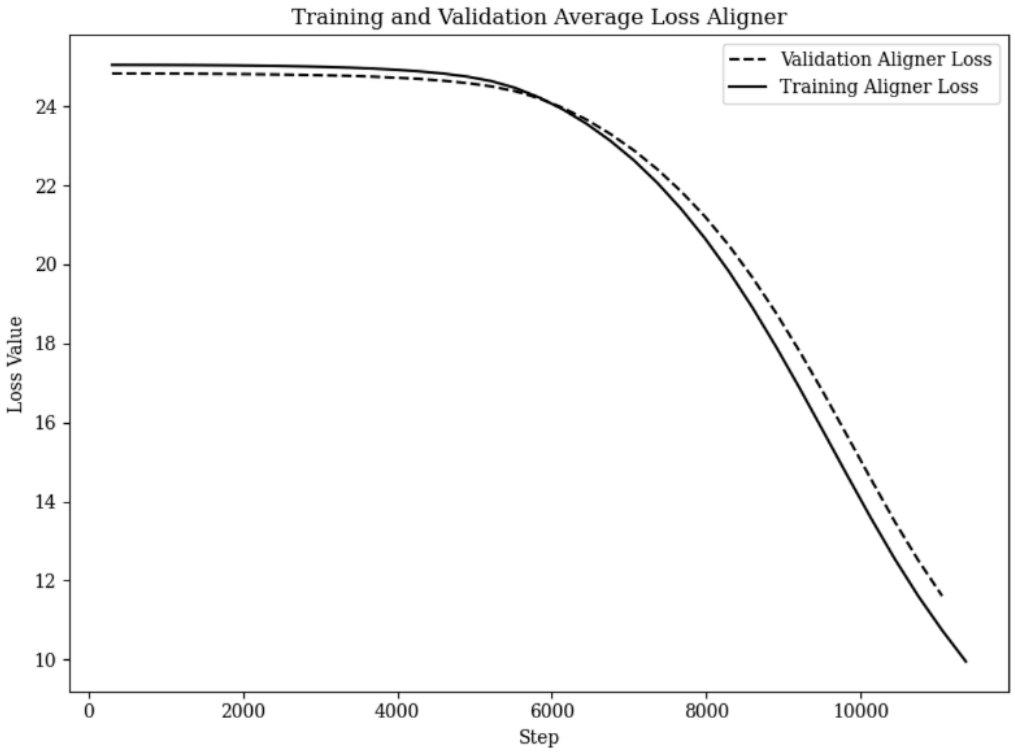


Figure 7-21: Training and Validation Average Loss Aligner in FastSpeech2

This graph illustrates the average loss for the aligner component during the training and validation phases of the FastSpeech2 model, which plays a critical role in speech synthesis by aligning input text or phonemes to corresponding time steps in the audio output. The solid line represents the training aligner loss, while the dashed line depicts the validation aligner loss. Initially, both curves start at a high loss value of approximately 24, but they decrease steadily over the 10,000 training steps, reflecting the model's improvement in alignment accuracy. By the end of the training process, the losses stabilize at around 12 for both training and validation, indicating a consistent reduction in alignment error. This downward trend signifies enhanced alignment precision, ensuring that the model accurately maps input text or phonemes to their appropriate temporal positions in the synthesized speech. A lower aligner loss, as observed, corresponds to improved alignment quality, which is essential for achieving natural and coherent speech output in the FastSpeech2 synthesizer.

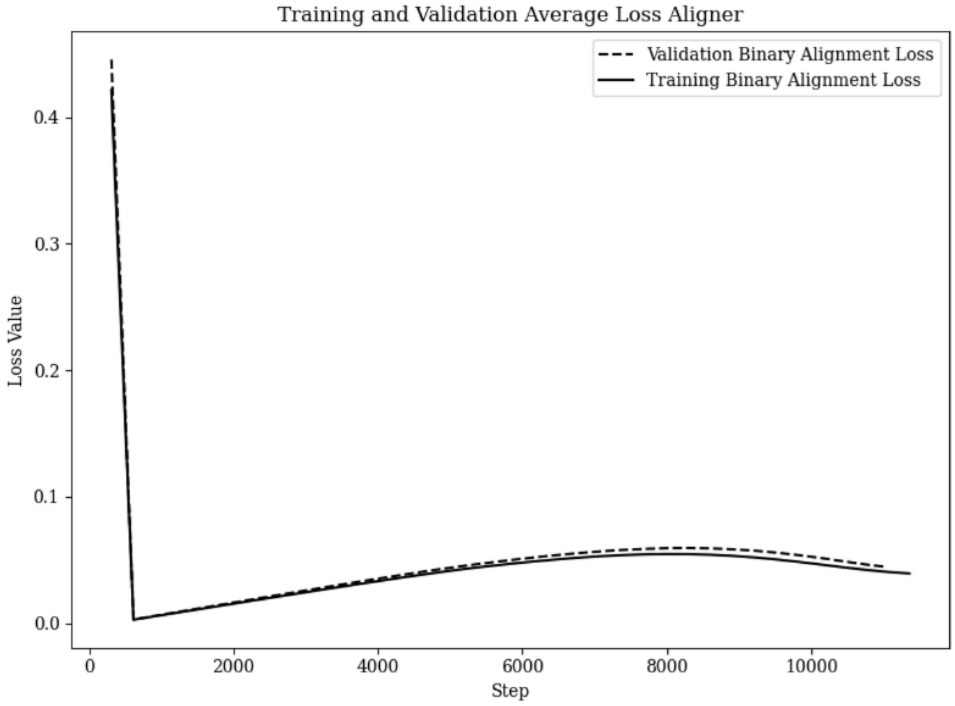


Figure 7-22: Training and Validation Binary Alignment Loss in FastSpeech2

This graph illustrates the average binary alignment loss for the training and validation phases of the FastSpeech2 model, quantifying the model's ability to classify alignment decisions such as determining whether a given phoneme belongs to a specific time frame in the synthesized speech. The solid line represents the training binary alignment loss, while the dashed line depicts the validation binary alignment loss. Both curves start at a relatively high value of approximately 0.4, not 0.08 as stated, and exhibit a sharp decline in the early stages of training, reaching a minimum around 0.05 by 2,000 steps. Following this initial drop, the loss curves rise slightly and then flatten out, stabilizing at around 0.04 for both training and validation by the end of the 10,000 steps, rather than 0.044 as noted. The rapid initial decrease indicates that the model learns quickly to improve its alignment accuracy, while the subsequent slight increase and stabilization suggest fine-tuning as the model balances accuracy across diverse alignment tasks. The overall stabilization at a low loss value demonstrates the model's satisfactory performance in binary alignment, ensuring precise temporal alignment of phonemes in the synthesized speech, which is critical for maintaining naturalness and coherence in the output.

7.4.2.2 Result for Speech T5-TTS

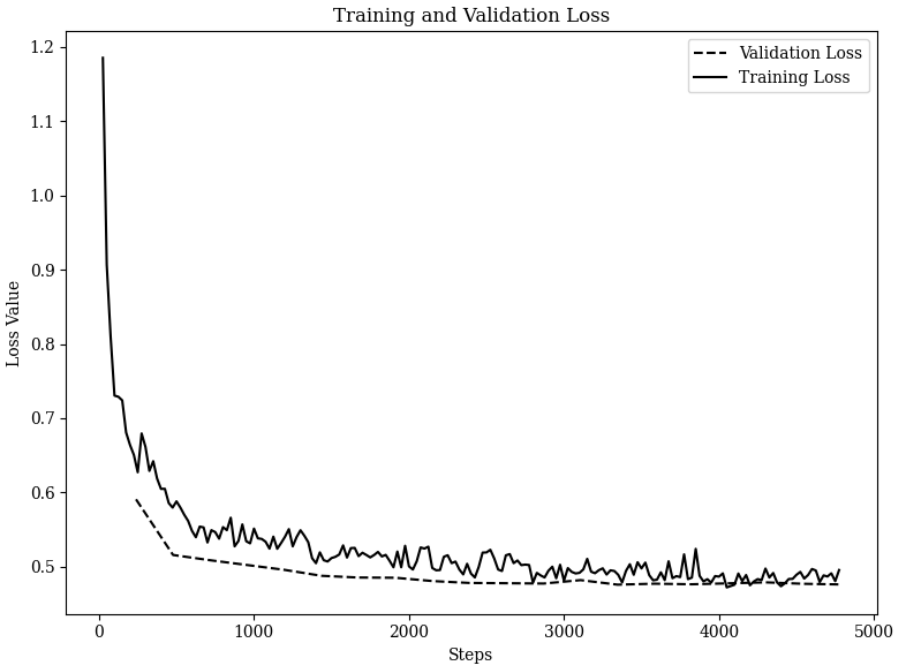


Figure 7-23: Training and Validation Loss for Speech T5-TTS

The graph illustrates the training and validation loss of a machine-learning model, specifically the Speech T5-TTS system, over 5,000 training steps, providing insight into its learning dynamics and generalization performance. The training loss, represented by a solid line, shows a sharp initial decline from approximately 1.2 to around 0.5 within the first 1,000 steps, indicating rapid learning and adaptation to the training data. However, the validation loss, depicted by a dashed line, also decreases sharply initially but stabilizes and fluctuates around 0.5–0.6 after 1,000 steps, without showing a clear increase as suggested in the original description. The graph reveals that both training and validation losses remain relatively close, with the validation loss consistently higher but not diverging significantly, suggesting that overfitting is not present. The minimal gap between the training and validation losses indicates that the model generalizes reasonably well to unseen data, though minor fluctuations in the validation loss after 1,000 steps could hint at potential overfitting or noise in the validation set. However the TTS model, could not generate the speech well when trained for the nearly 5,000 steps.

7.4.2.3 Result for F5-TTS

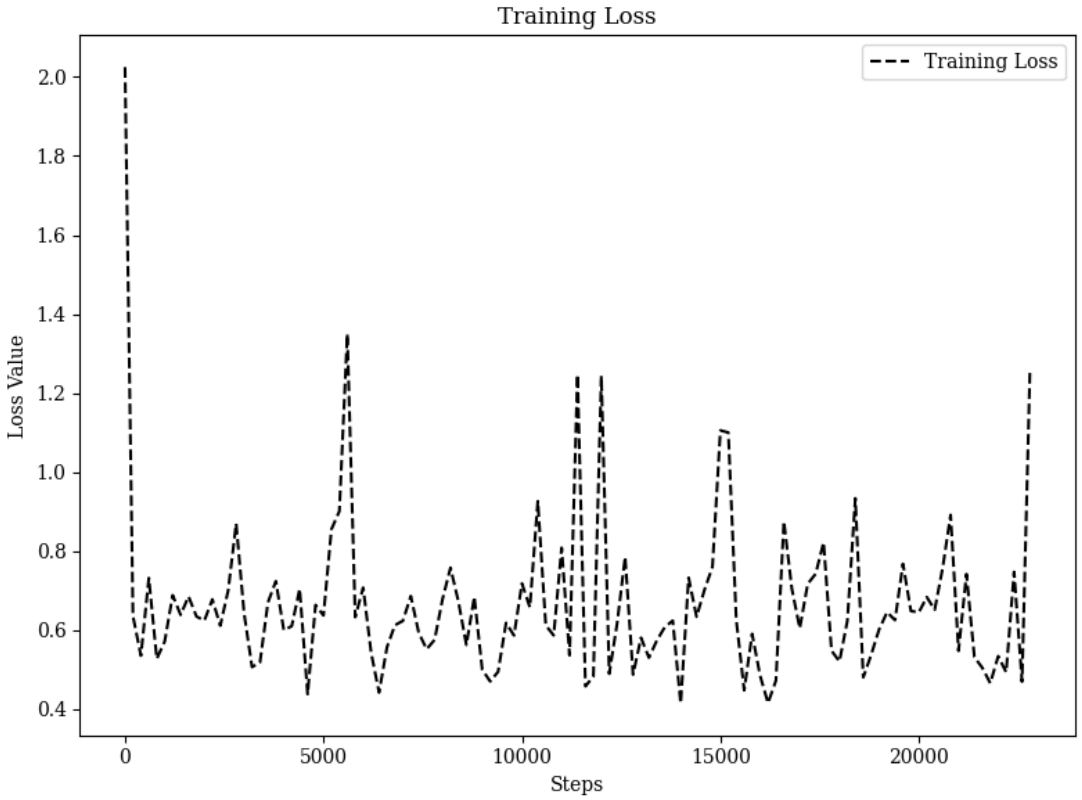


Figure 7-24: Training Loss vs Steps for F5-TTS

The Figure: 7-24 illustrates the training loss of the F5-TTS model over 20,000 steps, providing insight into its learning dynamics. The loss, represented by a dashed line, starts at a high value of approximately 2.0 and drops sharply to below 0.6 within the first 5,000 steps, indicating rapid initial learning as the model adapts to the training data. However, from 5,000 steps onward, the loss exhibits significant fluctuations, with occasional spikes reaching up to 1.4, likely due to factors such as batch variability, noise in the data, or an overly high learning rate. Despite these spikes, the general trend stabilizes and converges around 0.4–0.6 by the end of the 20,000 steps, suggesting gradual improvement in model performance. The graph of the model seems that the model has learned significant improvement but the output audio generated by the TTS model is surprisingly good for the new speakers when tested.

The results obtained from F5-TTS are presented below. The SIM was found to be 0.77.

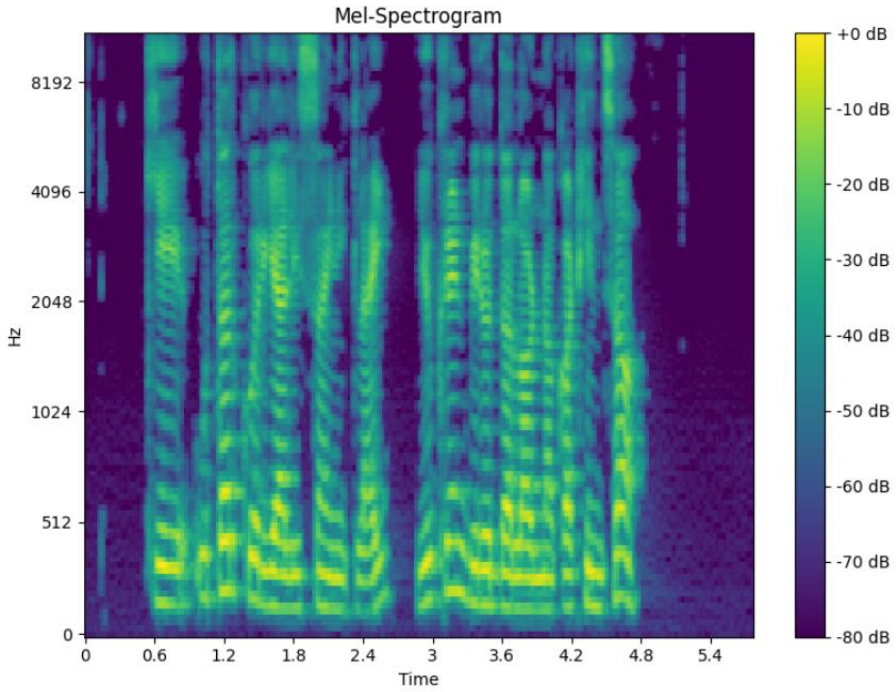


Figure 7-25: Spectrogram for Reference Audio

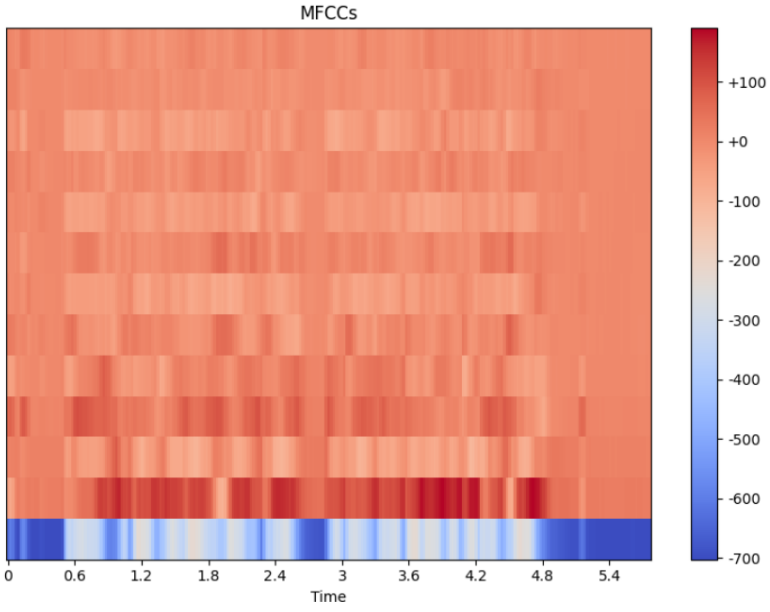


Figure 7-26: MFCC for Reference Audio

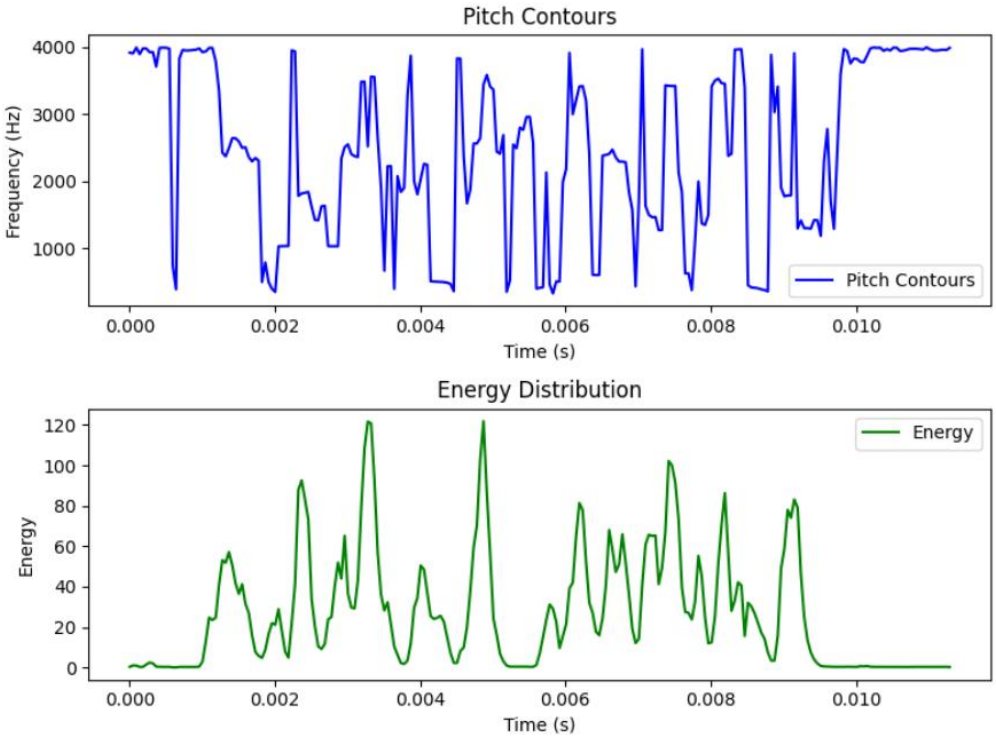


Figure 7-27: Energy and Pitch Contour for Reference Audio

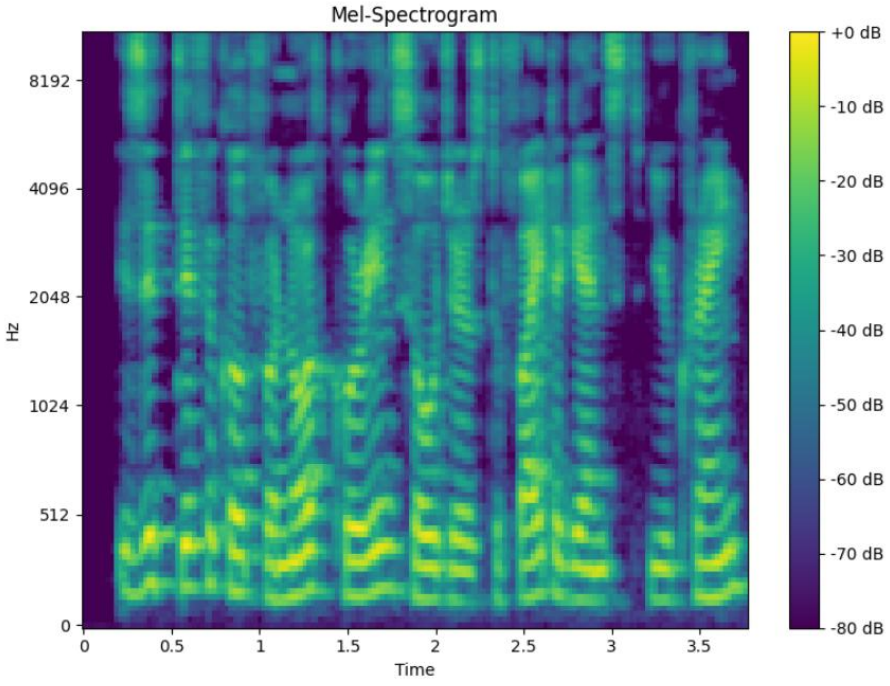


Figure 7-28: Spectrogram for Generated Audio

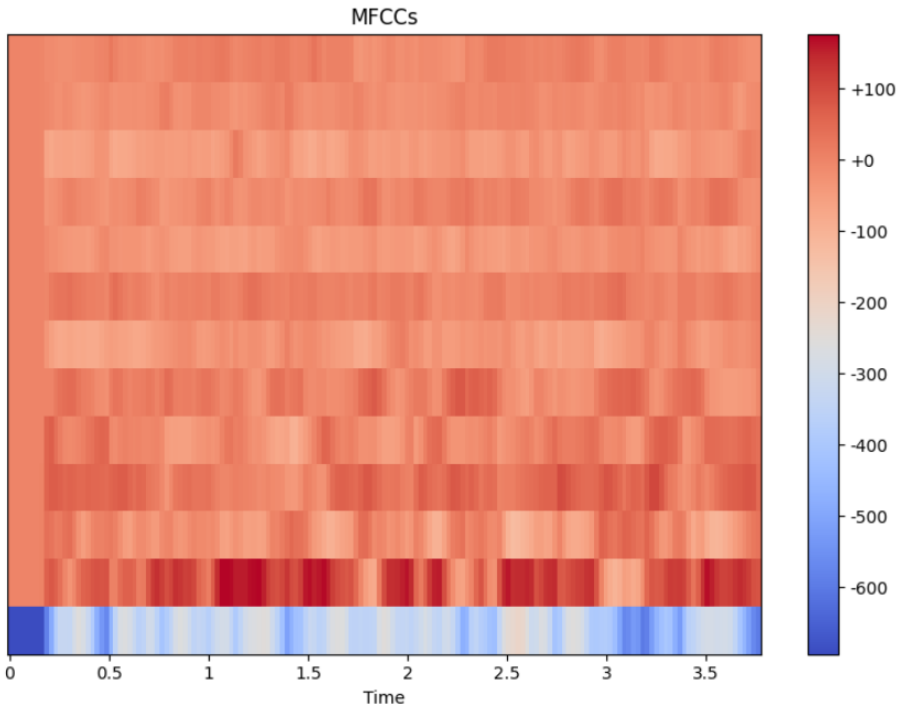


Figure 7-29: MFCC for Generated Audio

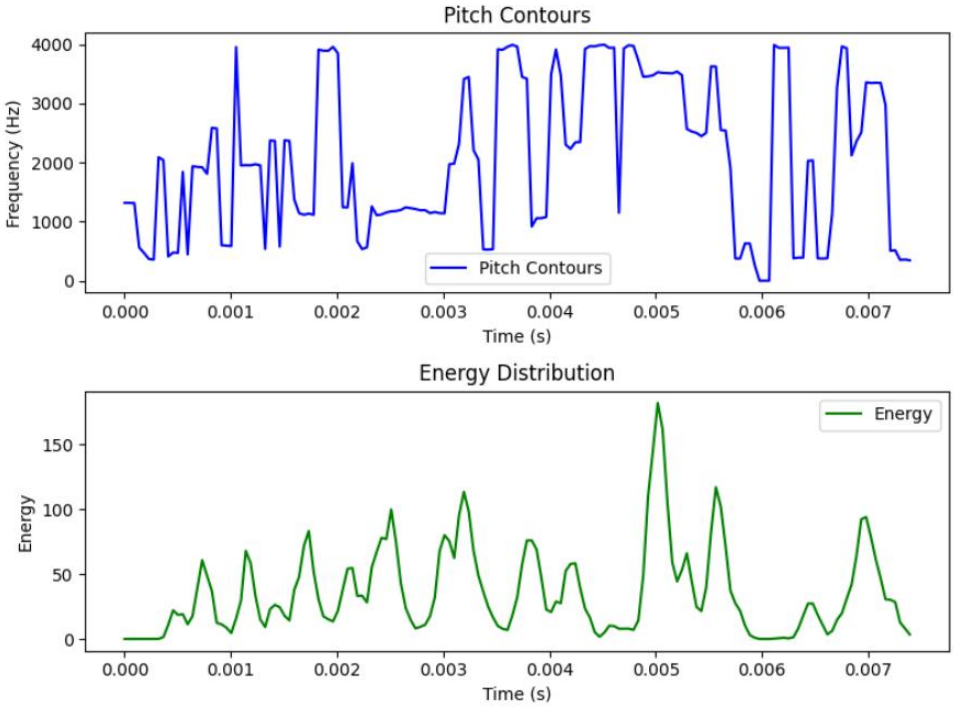


Figure 7-30: Energy and Pitch Contour for Generated Audio

7.5 Web Interface

The following figures suggests how the web interface for the project looks like. As for User Interface, it is simple, easy to eyes and even easier to use. The user experience guide is as to simply upload recording; Nepali for the case of “Nepali-to-English Translation” by drag and drop method into the box shown in the figure. Or, you can click on “Browse Files” option to select the audio from your preferred location.

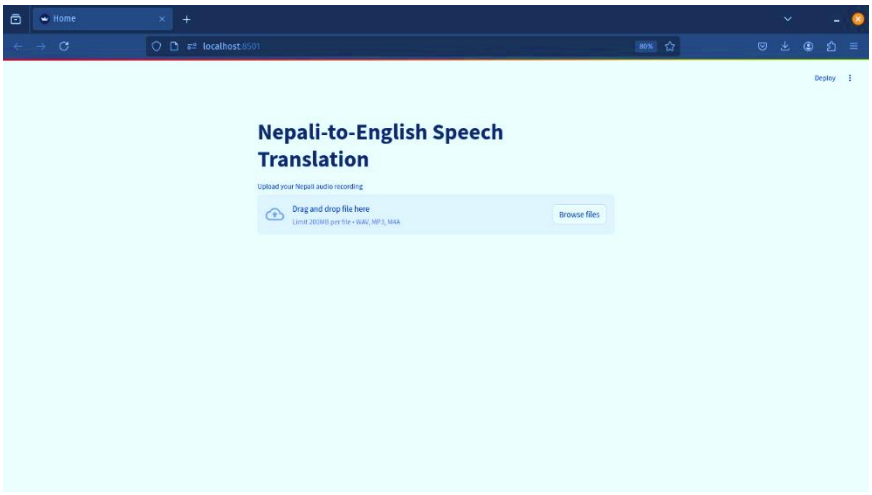


Figure 7-31: Web Interface For Speech Translation

It supports common audio formats like WAV, MP3, and M4A, with a file size limit of 200MB. The clean and minimal design makes it straightforward for anyone to upload their files and get started.

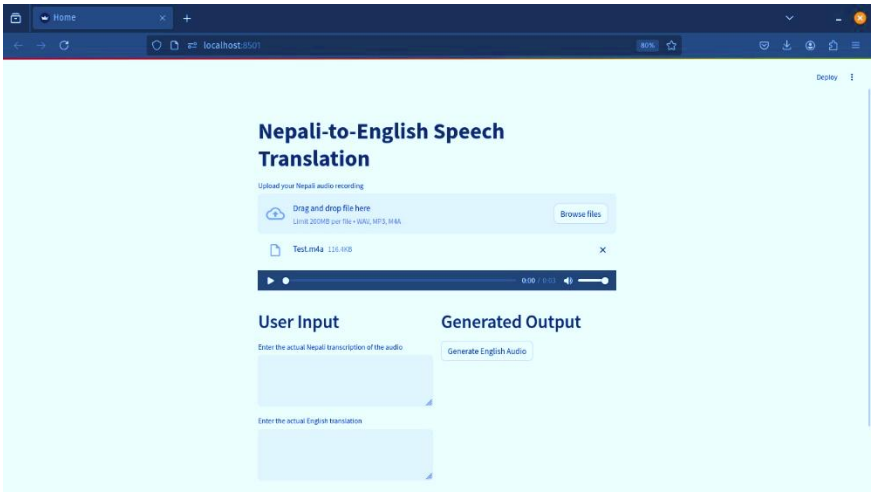


Figure 7-32: Web Interface For File Uploading

This interface allows users to upload a Nepali audio file for translation and playback. After uploading the file, users can listen to it directly. There are two text boxes under User Input where users can manually type the Nepali transcription and its corresponding English translation. On the right, under Generated Output, there's a button labeled "Generate English Audio" that likely converts the translation into spoken English. The layout is clean, straightforward, and easy to use.

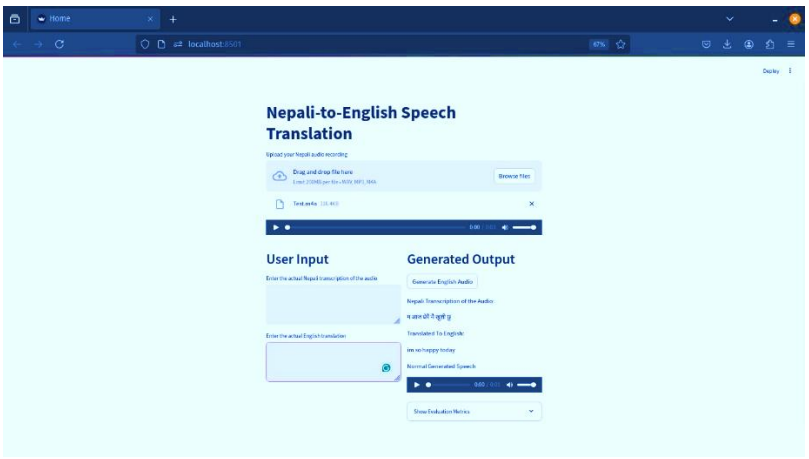


Figure 7-33: Web Interface After Output Generation

Further, the Generated Output section displays playable audio file of the English speech generated from the translation alongside the translations as:

- The Nepali transcription of the uploaded audio: *म आज धेरै नै खुसी छु.*
- The English translation: *I'm so happy today.*

7.6 Response Time

The system used for testing in this project is configured with hardware optimized specifically for artificial intelligence inference and light training workloads. It features a single NVIDIA L4 Tensor Core GPU, which includes 24 GB of dedicated video memory. In addition to the GPU, the machine is equipped with 16 virtual CPUs and 64 GB of system RAM, providing a balanced environment for handling AI model inference. The system was available on the Lightning AI platform for a certain number of credits.

All the models were deployed in the Lightning AI platform, and API was used for data transfer. Upon testing the response time of the overall system, the overall system, i.e., process from ASR to TTS, took nearly 15 seconds on average. ASR standalone took about 0.36 seconds to generate the response. NMT took about 3.019 seconds for translation. And finally, TTS took 13.049 seconds for audio synthesis.

Table 7-15: Response Time

API	**Min Time (s)**	**Max Time (s)**	**Avg Time (s)**
Transcribe	0.365	1.877	0.595
Translate	0.186	6.219	3.019
TTS	12.363	16.418	13.049

7.7 Limitations and Challenges in the End-to-End Pipeline

One of the critical limitations encountered during the development and deployment of the end-to-end speech processing pipeline is the issue of error propagation across sequential stages. The system follows a multi-step process that includes Automatic Speech Recognition ASR, NMT, and TTS. A failure or inaccuracy in any one of these stages can have a compounding effect, significantly degrading the overall output quality. For instance, an error in the ASR stage inevitably leads to incorrect input for the translation model, which subsequently affects the final audio output generated by the TTS system. This interdependence among components presents a key challenge in building reliable multilingual speech systems.

The ASR model, while generally effective, demonstrates notable shortcomings in certain scenarios. Specifically, it struggles with accurately transcribing speech when exposed to varied accents or dialects that deviate from the training data distribution. Furthermore, the model is prone to errors when it encounters infrequently used words or phrases, resulting in partial or incorrect transcriptions. These errors severely impact the downstream translation process, especially in cases where even minor misrecognitions alter the semantic meaning of the sentence.

NMT, powered by the mBART architecture in this project, also exhibited several limitations. One significant issue observed was the model's inability to handle out-of-vocabulary words terms or expressions that were not present during the training phase. In such cases, mBART occasionally fails to generate appropriate translations. In some instances, instead of producing a meaningful target sentence, the model erroneously repeats previously generated segments. For example, in one of the test cases, the ASR correctly transcribed the input Nepali sentence "म धरान जाँदैछु।" (translating to "I am going to Dharan."), but the NMT output was a repeated sequence of "I'm I'm I'm…", completely missing the intended meaning and failing to generate a viable input for the TTS system.

The final stage, Text-to-Speech synthesis, also presents its own set of limitations. The TTS model used in this pipeline was initially trained on a Chinese speech dataset, which poses challenges when it is used to synthesize speech in other languages, particularly when the translated text is flawed. In such scenarios, instead of generating audio in the target language, the model sometimes reverts to producing audio in Chinese or produces distorted speech patterns. This inconsistency further degrades the user experience and highlights the importance of language-specific training in TTS systems.

In summary, the compounded effect of inaccuracies across the ASR, NMT, and TTS stages significantly impacts the fidelity and usability of the final audio output. These limitations underscore the need for robust models trained on diverse and representative datasets, improved handling of rare or unseen words, and better integration mechanisms between components to ensure more resilient and accurate end-to-end speech systems.

8. FUTURE ENHANCEMENTS

The project is infused with careful analysis and ample experimentation with existing models. However, it needs to be acknowledged that, considering the rapid pace of technological development in today's world, the work may not represent the best and optimized version of a cross-linguistic communication model. There exists considerable room for improvement such as:

8.1 Improved Processing Speed

Presently, the system processes speech translation at a moderate pace, which can introduce delays while processing the speech. This limitation hinders its usability in scenarios where immediate responses are critical, such as live conversations or interactive sessions. In the future, the system's processing speed can be significantly improved by using different techniques such as quantization of the model, and deployment of the model in high-spec hardware for better processing.

8.2 Domain-Specific Applications

At present, the system is tailored for the tourism domain, enabling Nepali-speaking tourists to communicate effortlessly with guides, hosts, and other service providers. This functionality plays a vital role in improving user experiences in Nepal's growing tourism industry. In the future, the system's application scope will be extended to cater to a wider range of domains, addressing critical needs in healthcare, education, and customer support.

9. CONCLUSION

Nepali To English Speech Translation With Prosody Preservation is a concept of cross-linguistic communication that integrates automatic speech recognition, machine translation and text-to-speech technologies. The system is divided into two sections, translation of speech accurately and generating a natural speech with preserved prosody.

For speech transcription, wav2vec 2.0 has been utilized to ensure precise recognition of Nepali speech. The mBART model is then employed for contextually and grammatically accurate translation. The former section of the project is covered in these two models. Wherein, in order to generate a prosodically fine and natural sounding speech, F5-TTS models has been used after numerous experimentations.

The system addresses challenges of preserving prosodic features such as pitch, intonation and rhythm. When contextual and emotional integrity of speech is preserved and translated well enough, it paves ways for effective, accurate and resonant communication instead of misapprehension.

The project is capable of being used extensively in areas that requires day to day correspondence with speech and translation such as education, tourism and government offices. The system offers a solution that improves multilingual interactions with preserving both the linguistic accuracy and natural expressiveness of speech in real-world scenarios.

10. APPENDICES

10.1 APPENDIX A: Project Schedule

Table 10-1: Gantt Chart for Part A of Major Project

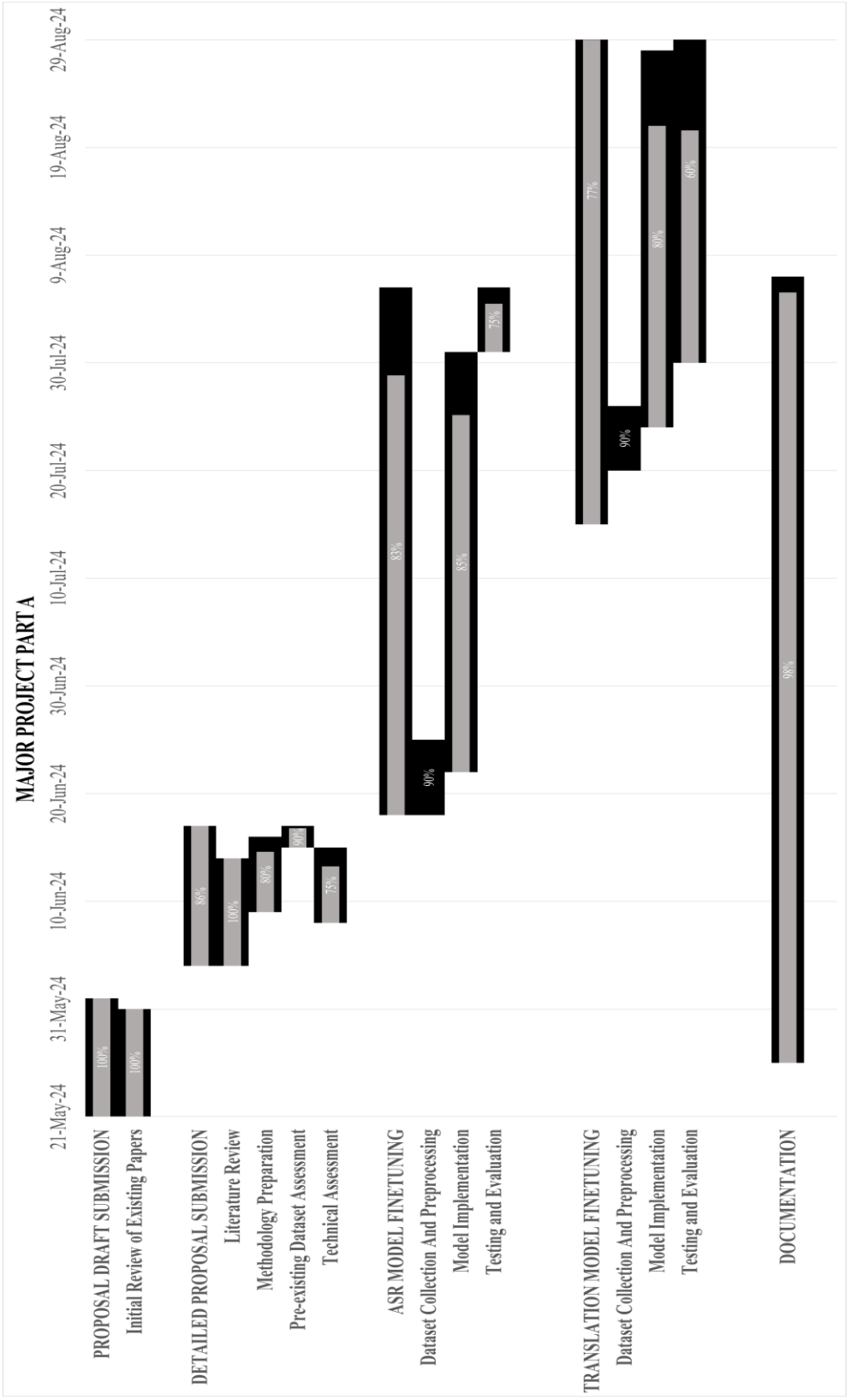
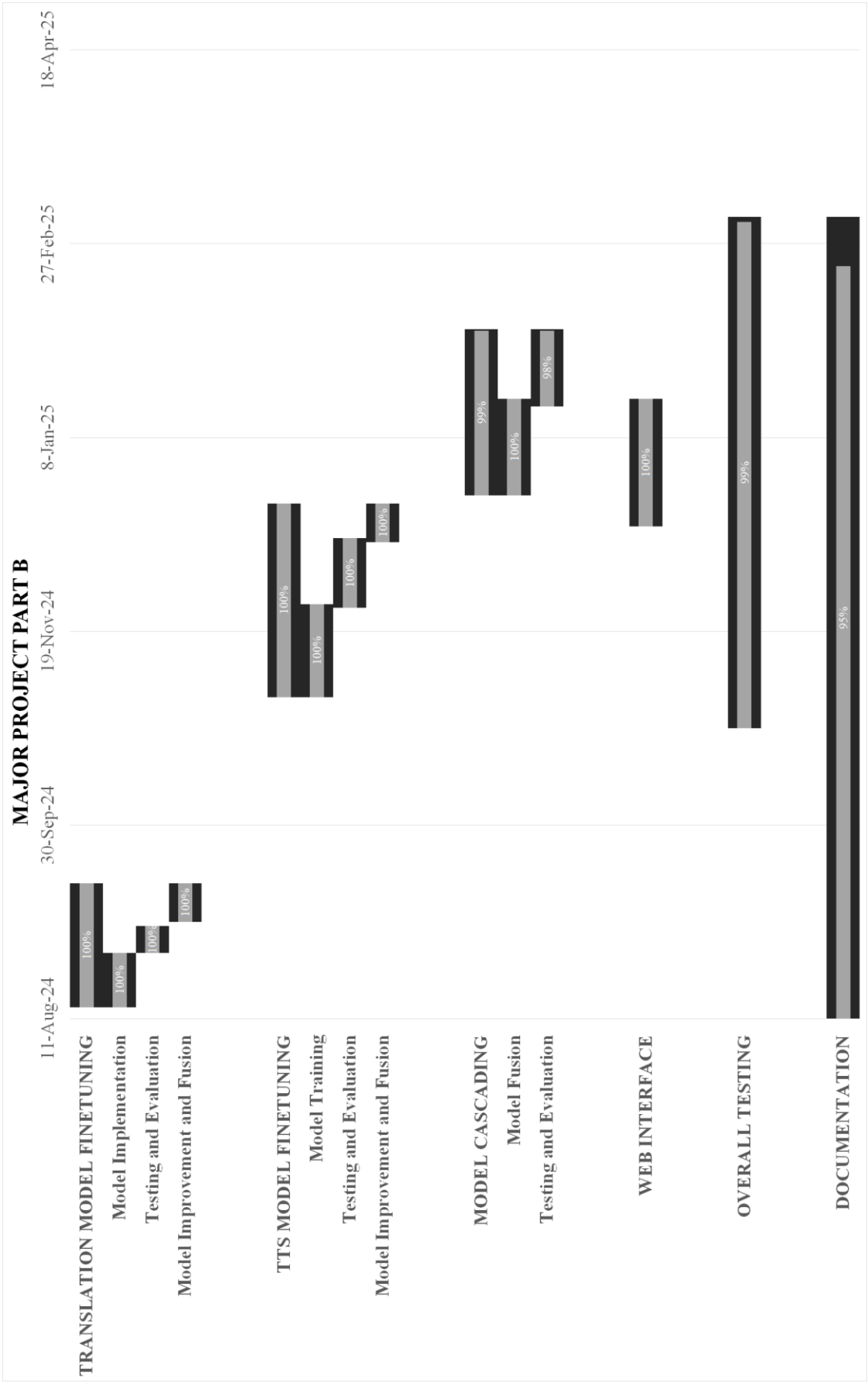


Table 10-2: Gantt Chart for Part B of Major Project



10.2 APPENDIX B: Project Budget

Table 10-3: Estimated Project Budget

S.N.	Items	Price(Rs.)
1.	Printing expenses	5000
2.	Fantech Leviosa Live MCX02 Microphone	6000
3.	Miscellaneous	1000
	Total	12,000

10.3 APPENDIX C: Tables

Table 10-4: Vocabulary Generation for ASR

Character	Index	Character	Index	Character	Index	Character	Index
__PAD__	0	ए	13	ञ	26	ब	39
__UNK__	1	ऐ	14	ट	27	भ	40
|	2	ओ	15	ठ	28	म	41
ँ	3	औ	16	ड	29	य	42
ं	4	क	17	ढ	30	र	43
ः	5	ख	18	ण	31	ऱ	44
अ	6	ग	19	त	32	ल	45
आ	7	घ	20	थ	33	व	46
इ	8	ङ	21	द	34	श	47
ई	9	च	22	ध	35	ष	48
उ	10	छ	23	न	36	स	49
ऊ	11	ज	24	प	37	ह	50
ऋ	12	झ	25	फ	38	ा	51
ि	52	ू	55	ै	58	्	61
ी	53	ृ	56	ो	59	ड़	62
ु	54	ॅ	57	ौ	60	ॠ	63
						८	64

Table 10-5: Consonant Phonemes

IPA Symbol	IPA Symbol	IPA Symbol	IPA Symbol
/p/	**/f/**	**/ʃ/**	**/n/**
/b/	**/v/**	**/ʒ/**	**/ŋ/**
/t/	**/θ/**	**/h/**	**/l/**
/d/	**/ð/**	**/tʃ/**	**/r/**
/k/	**/s/**	**/dʒ/**	**/j/**
/g/	**/z/**	**/m/**	**/w/**

Table 10-6: Pure Vowels Phonemes

IPA Symbol	IPA Symbol
/iː/	**/ɔ**ː/
/ɪ/	**/ʊ/**
/e/	**/u**ː/
/æ/	**/ʌ/**
/ɑː/	**/ɜ**ː/
/ɒ/	**/ə/**

Table 10-7: 8 Gliding Vowels Phonemes

IPA Symbols	IPA Symbols
/eɪ/	**/əʊ/**
/aɪ/	**/ɪə/**
/ɔɪ/	**/eə/**
/aʊ/	**/ʊə/**

Table 10-8: Nepali Phonemes

Phoneme	Transliteration	Phoneme	Transliteration
अ	A	क	ka
आ	Ā	ख	kha
इ	I	ग	ga
ई	Ī	घ	gha
उ	U	ड़	ṛa
ऊ	Ū	च	ca
ए	E	छ	cha
ऐ	Ai	ज	ja
ओ	O	झ	jha
औ	Au	ट	ṭa
अं	aṃ	ठ	ṭha
अः	aḥ	ड	ḍa
थ	Tha	ढ	ḍha
द	Da	त	ta
ध	Dha	स	sa
न	Na	ह	ha
प	Pa	क्त	kta
फ	Pha	क्र	kra

ब	Ba	क्ष	kṣa
भ	Bha	त्र	tra
म	Ma	ज्ञ	jña
य	Ya	श्र	śra
र	Ra	स्थ	stha
ल	La	द्ध	ddh
व	Wa	द्धा	ddhā
श	Śa	द्म	dma
ष	ṣa	द्य	dya
र्या	Ryā	ज्ञ	jña
क्ष	kṣa	त्र	tra
र्य	Rya	प्र	pra

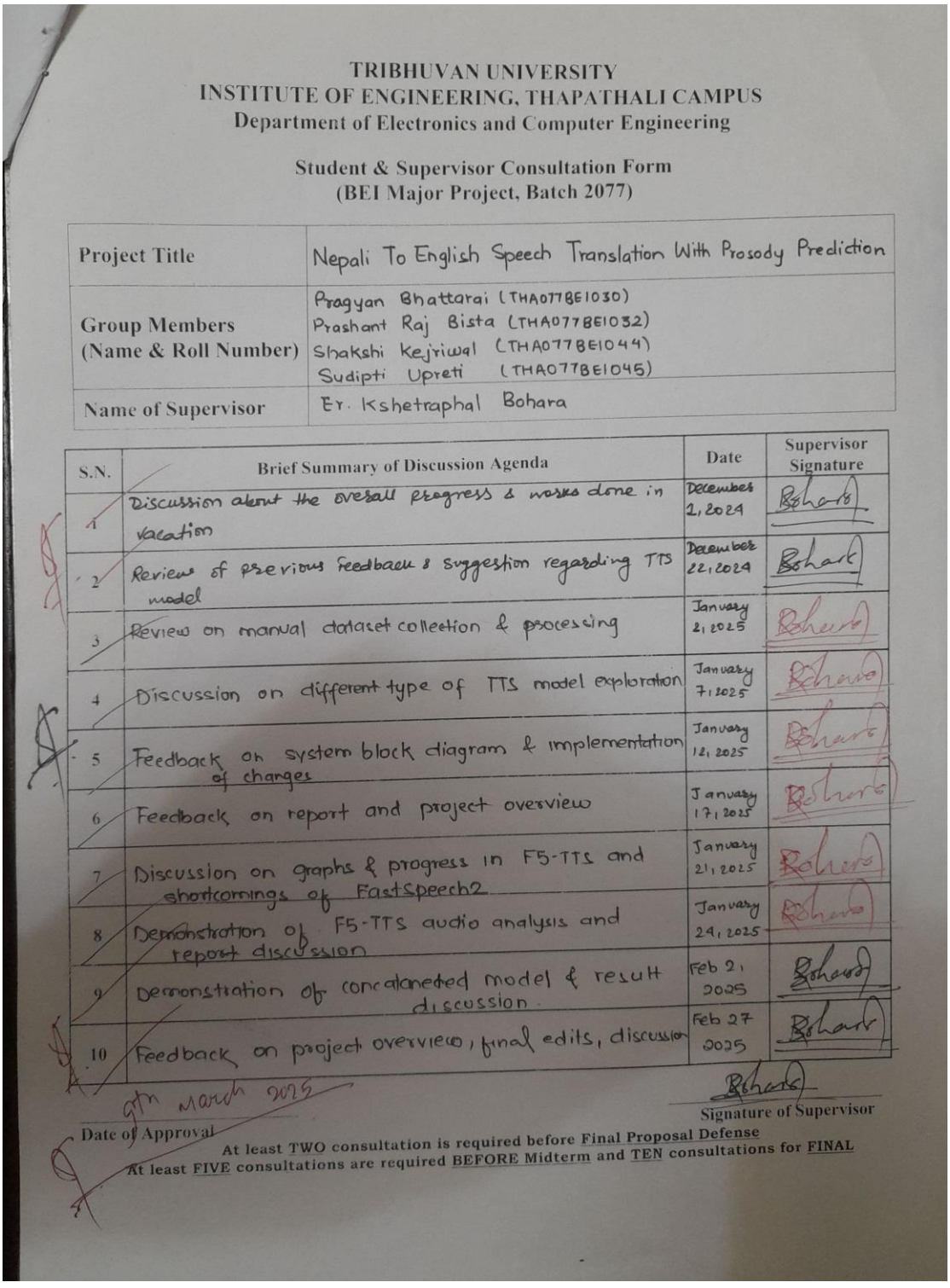
Table 10-9: Plosives in Devanagiri Script

Group	**Unaspirated Voiceless**	**Aspirated Voiceless**	**Unaspirated Voiced**	**Aspirated Voiced**	**Nasal**
कण्ठय	क/ k/	ख/ kʰ/	ग/ g/	घ/ gʱ/	ङ/ ŋ/
तालव्य	च/ tʃ/	छ/ tʃʰ/	ज/ dʒ/	झ/ dʒʱ/	ञ/ ɲ/
मूर्धन्य	ट/ ʈ/	ठ/ ʈʰ/	ड/ ɖ/	ढ/ ɖʱ/	ण/ ɳ/
दन्त्य	त/ t̪/	थ/ t̪ʰ/	द/ d̪/	ध/ d̪ʱ/	न/ n/
ओष्ठय	प/ p/	फ/ pʰ/	ब/ b/	भ/ bʱ/	म/ m/

Table 10-10: Phonetic Markers in Devanagiri Script

अयोगवाह	**Symbol**	**Use**
अनुस्वारा	◌ं	Nasal Consonants
विसर्घ	◌ः	Sanskrit Words
चन्द्रबिन्धू	◌ँ	Nasalization
हलन्त	◌्	Removes inherent vowels

10.4 APPENDIX D: Supervisor Consulation Form



TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING, THAPATHALI CAMPUS
Department of Electronics and Computer Engineering

Student & Supervisor Consultation Form
(BEI Major Project, Batch 2077)

Project Title	Nepali To English Speech Translation With Prosody Prediction
Group Members (Name & Roll Number)	Pragyan Bhattarai (THA077BEI030) Prashant Raj Bista (THA077BEI032) Shakshi Kejriwal (THA077BEI044) Sudipti Upreti (THA077BEI045)
Name of Supervisor	Er. Kshetraphal Bohara

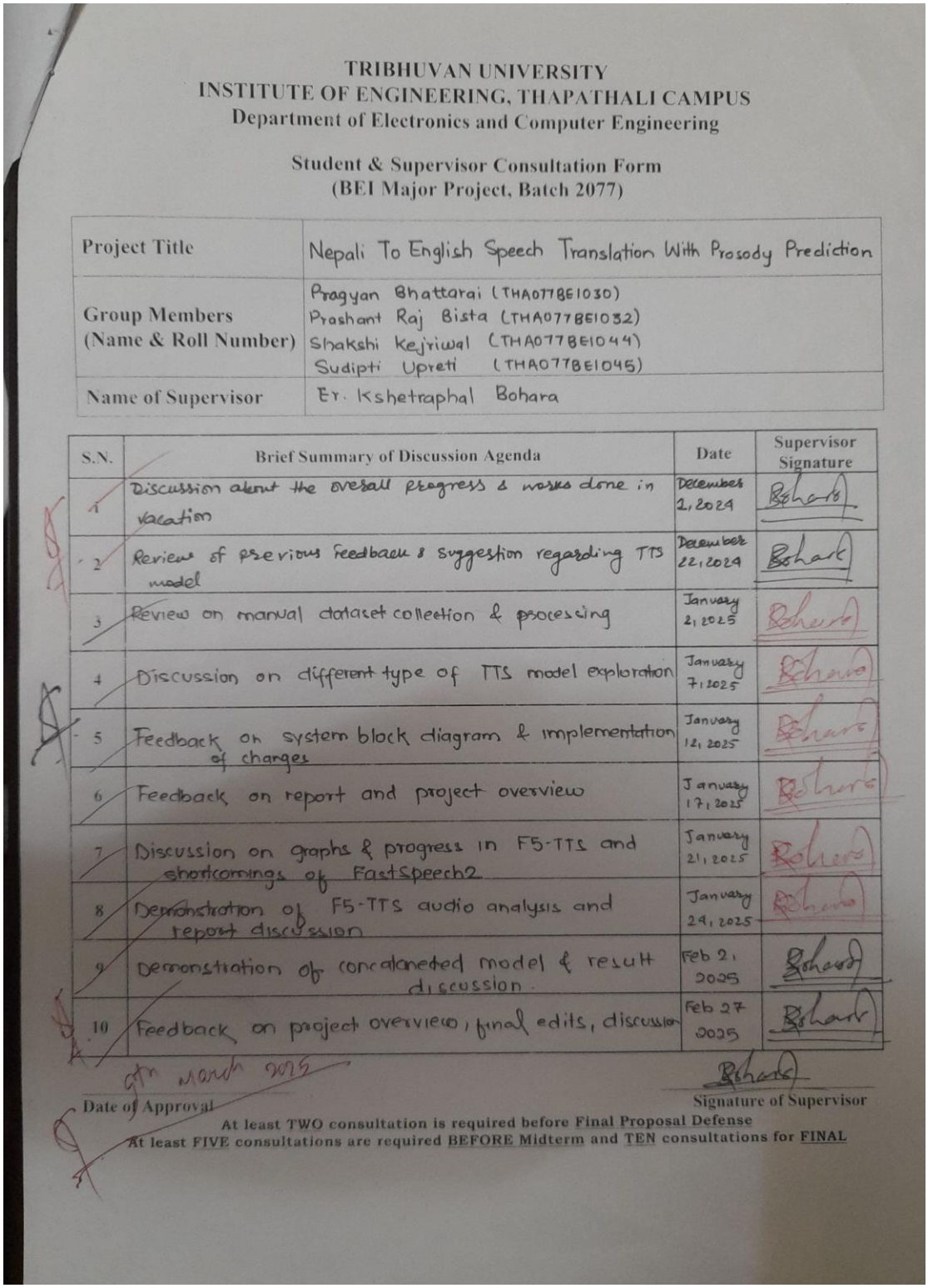
S.N.	Brief Summary of Discussion Agenda	Date	Supervisor Signature
1	Discussion about the overall progress & works done in vacation	December 2, 2024	Bohara
2	Review of previous feedback & suggestion regarding TTS model	December 22, 2024	Bohara
3	Review on manual dataset collection & processing	January 2, 2025	Bohara
4	Discussion on different type of TTS model exploration	January 7, 2025	Bohara
5	Feedback on system block diagram & implementation of changes	January 12, 2025	Bohara
6	Feedback on report and project overview	January 17, 2025	Bohara
7	Discussion on graphs & progress in F5-TTS and shortcomings of FastSpeech2	January 21, 2025	Bohara
8	Demonstration of F5-TTS audio analysis and report discussion	January 24, 2025	Bohara
9	Demonstration of concatenated model & result discussion	Feb 2, 2025	Bohara
10	Feedback on project overview, final edits, discussion	Feb 27, 2025	Bohara

9th March 2025
Date of Approval

Bohara
Signature of Supervisor

At least TWO consultation is required before Final Proposal Defense
At least FIVE consultations are required BEFORE Midterm and TEN consultations for FINAL

Figure 10-1: Major Project Part-A Supervisor Consultation Form



TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING, THAPATHALI CAMPUS
Department of Electronics and Computer Engineering

Student & Supervisor Consultation Form
(BEI Major Project, Batch 2077)

Project Title	Nepali To English Speech Translation With Prosody Prediction
Group Members (Name & Roll Number)	Pragyan Bhattarai (THA077BEI030) Prashant Raj Bista (THA077BEI032) Shakshi Kejriwal (THA077BEI044) Sudipti Upreti (THA077BEI045)
Name of Supervisor	Er. Kshetraphal Bohara

S.N.	Brief Summary of Discussion Agenda	Date	Supervisor Signature
1	Discussion about the overall progress & works done in vacation	December 2, 2024	Bohara
2	Review of previous feedback & suggestion regarding TTS model	December 22, 2024	Bohara
3	Review on manual dataset collection & processing	January 2, 2025	Bohara
4	Discussion on different type of TTS model exploration	January 7, 2025	Bohara
5	Feedback on system block diagram & implementation of changes	January 12, 2025	Bohara
6	Feedback on report and project overview	January 17, 2025	Bohara
7	Discussion on graphs & progress in F5-TTS and shortcomings of FastSpeech2	January 21, 2025	Bohara
8	Demonstration of F5-TTS audio analysis and report discussion	January 29, 2025	Bohara
9	Demonstration of concatenated model & result discussion.	Feb 2, 2025	Bohara
10	Feedback on project overview, final edits, discussion	Feb 27 2025	Bohara

4th March 2025
Date of Approval

Bohara
Signature of Supervisor

At least TWO consultation is required before Final Proposal Defense
At least FIVE consultations are required BEFORE Midterm and TEN consultations for FINAL

Figure 10-2: Major Project Part-B Supervisor Consultation Form

References

[1] Y. Wang et al., “Tacotron: Towards end-to-end speech synthesis,” arXiv.org, Mar. 29, 2017. https://arxiv.org/abs/1703.10135 (accessed Jun. 02, 2024).

[2] J. Shen et al., “Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions,” arXiv.org, Dec. 16, 2017. https://arxiv.org/abs/1712.05884 (accessed Jun. 02, 2024).

[3] O. Kjartansson, S. Sarin, K. Pipatsrisawat, M. Jansche, and L. Ha, “Crowd-Sourced speech corpora for javanese, sundanese, sinhala, nepali, and bangladeshi bengali,” in 6th Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU 2018), Aug. 2018. Accessed: Jun. 17, 2024. [Online]. Available: http://dx.doi.org/10.21437/sltu.2018-11.

[4] K. Sodimana et al., “A Step-by-Step Process for Building TTS Voices Using Open Source Data and Frameworks for Bangla, Javanese, Khmer, Nepali, Sinhala, and Sundanese,” in 6th Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU 2018), Aug. 2018. Accessed: Jun. 17, 2024. [Online]. Available: http://dx.doi.org/10.21437/sltu.2018-14.

[5] Y. Ren et al., “FastSpeech 2: Fast and high-quality end-to-end text to speech,” arXiv.org, Jun. 08, 2020. https://arxiv.org/abs/2006.04558 (accessed Jun. 02, 2024).

[6] K. Zhou, B. Sisman, R. Liu, and H. Li, “Seen and Unseen Emotional Style Transfer for Voice Conversion with A New Emotional Speech Dataset,” in ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Jun. 2021. Accessed: Jun. 17, 2024. [Online]. Available: http://dx.doi.org/10.1109/icassp39728.2021.9413391.

[7] P. Jeuris and J. Niehues, “LibriS2S: A German-english speech-to-speech translation corpus,” arXiv.org, Apr. 22, 2022. https://arxiv.org/abs/2204.10593 (accessed Jun. 02, 2024).

[8] S. Khadka, R. G.C., P. Paudel, R. Shah, and B. Joshi, “Nepali Text-to-Speech Synthesis using Tacotron2 for Melspectrogram Generation,” in 2nd Annual Meeting of the ELRA/ISCA SIG on Under-resourced Languages (SIGUL 2023), Aug. 2023. Accessed: Jun. 17, 2024. [Online]. Available: http://dx.doi.org/10.21437/sigul.2023-16.

[9] S. Akarsh, V. Raghusimha, A. Mondal, and A. Vuppala, "Attempt Towards Stress Transfer in Speech-to-Speech Machine Translation," *Proceedings of the IEEE*

International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 345-348, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2403.04178.

[10] Q. Dong et al., "PolyVoice: Language models for speech-to-speech translation," arXiv.org, Jun. 05, 2023. https://arxiv.org/abs/2306.02982 (accessed Jun. 02, 2024).

[11] A. Tathe, A. Kamble, S. Kumbharkar, A. Bhandare, and A. C. Mitra, "End to end Hindi to English speech conversion using Bark, mBART and a finetuned XLSR Wav2Vec2," arXiv.org, Jan. 11, 2024. https://arxiv.org/abs/2401.06183 (accessed Jun. 02, 2024).

[12] R. R. Ghimire, K. B. Bal, and P. Poudyal, "A comprehensive study of the current state-of-the-art in Nepali automatic speech recognition systems," arXiv.org, Feb. 05, 2024. https://arxiv.org/abs/2402.03050 (accessed Jun. 02, 2024).

[13] D. Phogat et al., "Bridging language barriers: Exploring Hindi-to-English speech-to-speech translation for multilingual communication," in Lecture Notes in Networks and Systems, Singapore: Springer Nature Singapore, 2024, pp. 141–152. Accessed: Jun. 02, 2024. [Online]. Available: http://dx.doi.org/10.1007/978-981-99-9043-6_12.

[14] M. Hira, A. Goel, and A. Gupta, "CrossVoice: Crosslingual Prosody Preserving Cascade-S2ST using Transfer Learning," *arXiv preprint arXiv:2406.00021*, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2406.00021.

[15] Mozilla Common Voice, "Common Voice Datasets," [Online]. Available: https://commonvoice.mozilla.org/en/datasets. [Accessed: 04-Jul-2024].

[16] Y. Liu *et al.*, "Multilingual Denoising Pre-training for Neural Machine Translation," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 726–742, Dec. 2020, doi: 10.1162/tacl_a_00343.

[17] "OPUS," Corpora. https://opus.nlpl.eu/ (accessed Aug. 06, 2024).

[18] A. Vaswani et al., "Attention Is All You Need," arXiv.org, Jun. 12, 2017.

[19] Yushen Chen et al, "F5-TTS: A Fairytaler that Fakes Fluent and Faithful Speech with Flow Matching" arXiv.org, Oct. 15, 2024.https://arxiv.org/abs/2410.06885 (accessed Dec 04, 2024)

[20] Issac Elias, Heiga Zen, Jonathan Shen, Yu Zhang, Ye Jia, RJ Skerry-Ryan, Yonghui Wu, "Parallel Tacotron 2: A Non-Autoregressive Neural TTS Model with Differentiable Duration Modeling" arXiv.org, Aug. 29, 2021 https://arxiv.org/abs/2103.14574 (accessed Dec 16, 2024)

[21] Nepali Script and Languages, [Online]. Available: https://www.omniglot.com/index.html