TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

A

PROJECT REPORT

ON

AAWAJ : AUGMENTATIVE COMMUNICATION SUPPORT FOR THE VOCALLY IMPAIRED USING NEPALI TEXT-TO-SPEECH

SUBMITTED BY:

MAUSAM BASNET (PUL075BCT049)

NISHAN POUDEL (PUL075BCT057)

SAMPANNA DAHAL (PUL075BCT072)

SUKRITI SUBEDI (PUL075BCT089)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

May, 2023

Page of Approval

TRIBHUVAN UNIVERSIY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certifies that they have read and recommended to the Institute of Engineering for acceptance of a project report entitled **"Aawaj : Augmentative Communication Support for the vocally impaired using Nepali Text-to-Speech"** submitted by **Mausam Basnet**, **Nishan Poudel**, **Sampanna Dahal**, **Sukriti Subedi** in partial fulfillment of the requirements for the Bachelor's degree in Electronics & Computer Engineering.

.....
Supervisor
Nishchal Acharya
Assistant Professor
Department of Electronics and Computer Engineering,
Pulchowk Campus, IOE, TU.

.....
External examiner
Mahendra Thapa
Fusemachines

Date of approval:

Copyright

Acknowledgments

We would like to express our sincere gratitude to everyone who has contributed to the successful completion of our major project.

Firstly, we would like to thank our project supervisor Assistant Professor **Nishchal Acharya**, for his invaluable guidance, support, and encouragement throughout the project. His expertise and insightful feedback have been crucial in shaping the direction and scope of this work.

Furthermore, we would like to extend our gratitude to **Dr.Basanta Joshi**, whose invaluable direction and guidance helped us bootstrap the project initially. We would also like to extend our heartfelt thanks to the faculty members of the **Department of Electronics and Computer Engineering** for their continuous support and motivation.

Also, a special note of thanks is due to all the individuals and organizations who have provided us with the necessary resources, and insights, namely **Mr. Kabin Maleku, BP Koirala Institute of Health Sciences (BPKIHS), and Handicapped International (HI)**, with a special ode to their guidance, contribution, and support necessary to carry out this project.

Finally, we would also like to express our sincere appreciation to our classmates, families, and friends for their encouragement, love, support, suggestions, and constructive criticism. Their unwavering support and motivation have kept us going throughout the project.

Thank you all for your invaluable support and contribution toward the successful completion of this project.

Abstract

As of 2016, more than 147,000 of the Nepali population suffer from some variant of speech or hearing impairments. Similarly, there has been a visible scarcity of reliable Text-to-Speech (TTS) engines in the context of the Nepali language. Aawaj is a dedicated mobile application that addresses these impediments to create augmentative communication support for the vocally impaired populace of Nepal using a Nepali TTS engine. BIt utilizes vocal features such as timbre, prosody, rhythm, etc., to create a natural-sounding TTS engine, based on the open-source Tacotron2 TTS architecture published by Google. Rare conditions such as cerebral palsy, spinal cord injury, muscular dystrophy, and amyotrophic lateral sclerosis (ALS) have also led to a physical impediment in speech generation for a large population. This report further proposes an Augmentative and Alternative Communication (AAC) platform using accessibility features such as text prompt generation that provides accessibility to the intended populace of this mobile application.

Keywords: *Speech Impairments, Text-to-Speech (TTS), Augmentative and Alternative Communication (AAC), Augmentative Communication Support*

Contents

List of Figures

List of Tables

List of Abbreviations

TTS	Text-to-Speech
ALS	Amyotrophic Lateral Sclerosis
CV	Computer Vision
NLP	Natural Language Processing
DSP	Digital Signal Processing
API	Application Programming Interface
ML	Machine Learning
CNN	Convolutional Neural Network
AI	Artificial Intelligence
LSTM	Long Short-Term Memory
AAC	Augmentative Alternative Communication
FFT	Fast Fourier Transform
STFT	Short Time Fourier Transform
HMM	Hidden Markov Model
CRF	Conditional Random Field
ASR	Automatic Speech Recognition

1. Introduction

1.1 Background

Language is the ability to communicate a person's thoughts and ideas through a set of signals, which may be graphical, gestural, auditory, or even musical. It is a distinguishing trait of human beings, the only creatures capable of employing such an organized system. Speech is one of the prominent components of conveying this linguistic feature among humans. It is archaic, yet also the most popular and efficient mode of communication between people.

People with speech impairment often use sign language or gestures as their primary mode of communication because these methods allow them to convey their thoughts and feelings more effectively than speech alone.

Sign language is a visual language that uses a combination of hand gestures, facial expressions, and body language to communicate. There are many different sign languages used around the world, each with its own unique set of signs and rules for grammar and syntax. Sign language allows people with speech impairments to express themselves fully and interact with others more easily.

Gestures are another common form of nonverbal communication used by people with speech impairment. These can include pointing, nodding, shaking the head, or using facial expressions to convey emotions. While gestures are not as complex as sign language, they can still be a powerful way for people with speech impairments to communicate their needs and desires.

For some people with speech impairments, using sign language or gestures may not be enough to fully express themselves. In these cases, assistive technology such as communication devices with voice output or text-to-speech software can be used to help bridge the communication gap.

1.2 Problem Statement

A 2016 report on disabilities in Nepal showed that more than 147,000 people in Nepal suffered from a form of speech or hearing-related disorder[3]. Similarly, in 2016, a study among pediatric patients conducted by the Disability Research Centre, Kathmandu University found that 12.6% of the pediatric population had some kind of speech or language disorder[4]. This number of speech-impaired patients is estimated to be around 300,000 by the 2021 census.

Consequently, this has led to a growing need for assistive devices such as speech synthesis and conversational aids. Speech impairment can manifest in various forms, including stuttering, voice disorders, and articulation difficulties, making communication with others a difficult task.

Unfortunately, there hasn't been much assistance in Nepal when it comes to speech synthesis and conversational aid devices. While the government and various organizations have recognized the need for support systems for individuals with speech impairment, the resources and funding available to develop and distribute such devices are limited [5].

As a result, individuals with speech impairment in Nepal often struggle to communicate with others, and the lack of assistance devices can hinder their ability to participate fully in society. Without access to proper tools and resources, they may feel isolated, frustrated, and unable to express themselves effectively.

While there have been recent forays in the field of Text-To-Speech(TTS) in the Nepali language, the application of these TTS engines to aid the physically impaired has been little to none in the context of Nepal. When leaving the context of Nepal, there have been multiple speech impairment-centered applications aiming to provide an augmentative speech-aiding application with the use of underlying TTS engines. However, there have been scarcely any applications that aim to target overall physical disability in terms of a speech synthesizer.

1.3 Objectives

The primary objectives this project proposes to accomplish are enumerated below:

- Develop a TTS Engine for Nepali speech synthesis.
- Develop a mobile application to provide augmentative communication support for the vocally impaired with the help of said Nepali TTS engine.

1.4 Scope

This proposed application serves the purpose of fulfilling an unaddressed problem of accessibility for the speech impaired. The scope of this project in terms of the global market, demands fulfilled, and applications created are listed below:

- Speech impairment focused TTS Application
- Cursor replacement accessibility feature for Android
- Improved Nepali TTS engine

2. Literature Review

2.1 Related Work

2.1.1 Google Text-to-Speech

Google's TTS API provides a unique interface to developers where it facilitates the synthesis of speech in 100+ voices in multiple languages. Google utilizes the state-of-the-art WaveNet architecture along with their custom neural networks to ensure the speech sounds as natural as possible. Based on DeepMind's research on WaveNet, it is one of the strongest TTS engines out there. Google also reportedly has a Nepali TTS engine for its internal use, which hasn't been released to the public yet.

This project can employ the WaveNet architecture to create efficient TTS engines in the Nepalese domain. However, it would need expertise in Nepalese phonemes, syntax, and morphology. The existing engine architecture can be tweaked to support this objective.

2.1.2 Festival TTS

Festival TTS, or Festival Text-to-Speech, is a software application for speech synthesis developed at the Centre for Speech Technology Research at the University of Edinburgh. It is a free, open-source system that allows users to create synthesized speech from text input. Festival TTS is widely used in research and development in the field of speech synthesis, as well as in various applications such as assistive technology for individuals with visual impairments. Festival TTS uses a modular architecture that allows users to choose from various language models, acoustic models, and other components to customize the system to their needs. The system can produce synthetic speech in a variety of languages, and can also produce different voices with varying levels of expressiveness and naturalness. Festival TTS has a command-line interface, which means that users can interact with the system by entering commands in a terminal window. This allows users to automate text-to-speech tasks and integrate Festival TTS into larger software systems. Festival TTS also includes a graphical user interface (GUI) called Festival Lite, which provides a more user-friendly interface for creating and editing speech synthesis scripts. One of the key features of Festival TTS is its support for a scripting language called Scheme. This language allows users to define complex rules for generating speech from text, such as mapping phonemes to graphemes or controlling prosody (the melody and rhythm of speech). Scheme scripts can also incorporate other

programming languages and external tools, making it a powerful tool for speech synthesis research and development. Festival TTS is used in a variety of applications, including assistive technology for individuals with visual impairments, interactive voice response systems, and language learning tools. The system's flexibility and customizability make it a popular choice for researchers and developers working in the field of speech synthesis, and it has contributed significantly to advances in the technology of synthetic speech.

2.1.3 NepaliSpeech

NepaliSpeech.com is a practical application of TTS in the Nepalese language domain created by developer Kaushal Subedi. Initially created to aid humanitarian aid workers in Nepali speech synthesis after the 2015 earthquake, it utilizes the Flite-TTS engine created based on different Indian languages and dialects. It manipulates the pitch and speed of these dialects to create a more Nepali-sounding speech synthesizer. Utilizing Google's romanized Nepali to Devnagari transliterator, it synthesizes Nepali speech based on a user's text input.
The idea behind this application can be derived to manipulate pre-existing Indian language TTS models to make it sound in a more Nepalese dialect. Similarly, the dominating abundance of Hindi audio datasets means this avenue of accent manipulation can also be explored in case of inadequate Nepalese speech datasets.

2.1.4 Speech Assistant AAC

Speech assistant AAC is a specialized Android application designed to provide assistance with communication for individuals who have difficulty speaking or writing. It is an augmentative and alternative communication (AAC) tool that uses speech synthesis technology to convert written or typed text into spoken words or pre-recorded messages. Speech assistant AAC can be downloaded from the Google Play Store and installed on any Android device, such as a smartphone or tablet. The application is user-friendly and customizable, allowing users to adjust the font size, background color, and other settings to suit their needs. Once installed, speech assistant AAC allows users to enter text using a variety of input methods, including typing, writing, or selecting pre-programmed messages from a menu. The application includes a range of pre-programmed messages that cover common communication needs, such as greetings, requests, and social interactions. Users can also create and save their own messages for later use, and can organize their messages into categories for easy access. The speech output can be adjusted for speed and volume, and users can choose from a range of different voices and accents to suit their preferences. Speech assistant AAC can be particularly helpful for individuals with conditions such as cerebral palsy, ALS, or autism spectrum disorder, who may have difficulty communicating effectively due to limited motor

function or cognitive impairment. The application can also be useful for individuals with temporary communication difficulties, such as those recovering from a stroke or traumatic brain injury. Overall, speech assistant AAC is a useful tool for individuals with communication difficulties, allowing them to express themselves and communicate more effectively with others.

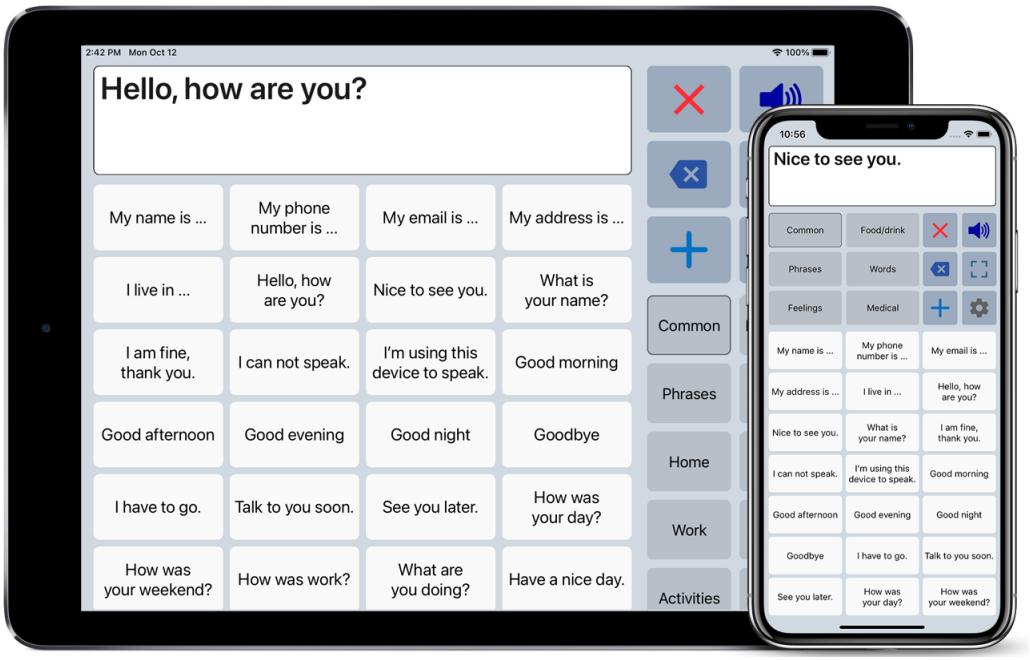

Figure 2.1: Speech Assistant AAC

2.1.5 AsTeRICS Grid

AsTeRICS is an open-source framework that was developed in Europe to create assistive technologies for people with disabilities. The name stands for "Assistive Technology Rapid Integration Construction Set", and it is designed to be easy to use and customizable. It is an innovative and powerful tool for creating assistive technology solutions that can improve the quality of life for people with disabilities. Its open-source nature also makes it accessible to a wide range of users and developers, which helps to promote collaboration and innovation in this important field.

The AsTeRICS Grid is an innovative tool that is built on top of the AsTeRICS framework. It is designed to provide a more user-friendly and accessible interface for creating assistive technology solutions using the AsTeRICS platform.

The AsTeRICS Grid uses a visual drag-and-drop interface to enable users to create assistive technology solutions without the need for programming skills. It consists of a grid of cells, each of which represents a software module or hardware component that can be

combined to create custom solutions.

The cells in the AsTeRICS Grid can be easily arranged and connected to create a flow of data and control between modules. This allows users to build custom workflows that can be tailored to meet the needs of individual users with disabilities.

Figure 2.2: AsTeRICS Grid GUI

One of the key benefits of the AsTeRICS Grid is its accessibility. It is designed to be easy to use for individuals with a range of disabilities, including those with physical impairments or cognitive disabilities. The visual nature of the interface and the ability to use alternative input methods, such as switches or eye-tracking devices, make it a highly flexible and adaptable tool.

2.2 Related Theory

2.2.1 Augmentative Alternative Communication

Augmentative and alternative communication (AAC) refers to the various methods used to enhance or replace speech and writing as a means of communication. AAC is used by individuals with communication impairments, including those with developmental disabilities, acquired brain injuries, or progressive neurological disorders, such as ALS or Parkinson's disease. AAC includes all forms of communication (other than oral speech) that are used to express thoughts, needs, wants and ideas. We all use AAC when we make facial expressions or gestures, use symbols or pictures, or write [6].

AAC can take many different forms, depending on the individual's abilities and needs. Some examples of AAC include:

1. Picture boards: These are boards with pictures or symbols that the individual can point to in order to communicate their needs or wants.

2. Communication books: These are books with pictures or symbols that the individual can use to construct sentences or phrases.

3. Speech-generating devices: These are electronic devices that generate speech from the user's input, such as typing or selecting pre-programmed phrases.

4. Eye-tracking devices: These are devices that track the user's eye movements and translate them into speech or text.

AAC can be used in various settings, including at home, school, and in the community. AAC systems can be customized to fit the individual's needs, preferences, and abilities. AAC can help individuals with communication impairments to express their thoughts, feelings, and needs, participate in social interactions, and engage in educational and vocational activities.

In addition, AAC can also help individuals to develop their language and communication skills, as they can practice using different symbols, phrases, and communication strategies. AAC can also promote independence and reduce frustration and isolation for individuals with communication impairments, as they can communicate more effectively and efficiently with others.

Overall, AAC is an important tool for individuals with communication impairments to enhance their communication abilities and participate more fully in their communities.

2.2.2 Text-to-Speech

Text-to-speech synthesis is defined as "the production of speech by machines, by way of the automatic phonetization of the sentences to utter"[7].
Speech, in itself, consists of several non-mutually exclusive levels of description - namely *acoustic, phonetic, phonological, morphological, syntactic, semantic, and pragmatic* levels[7]. Text-to-speech synthesis incorporates various forms of speech technology from speech coding, and automatic speech recognition, to speaker identification[8]. An ideal TTS engine attempts to replicate speech as naturally as the one produced by humans in a normal environment. A vocal sound produced by a human being is governed by dynamic cases of partial differential equations of fluid dynamics, dictated by our lung pressure, glottis tension, and vocal and nasal tract configuration[9].
As per Dutoit, a general TTS synthesizer comprises of two major components; a Natural

Figure 2.3: Augmentative and Alternative Communication Device
[6]

Language Processing (NLP) module and a Digital Signal Processing (DSP) module[9]. The NLP module is responsible for the generation of phonetic transcription of the text that is to be read, along with the desired level of intonation and rhythm. This "rhythm" is more precisely termed as *prosody*. The DSP module is then tasked with converting these representations of speech in the form of symbolic information into actual speech in the mechanical medium. The general NLP-DSP pipeline is illustrated in

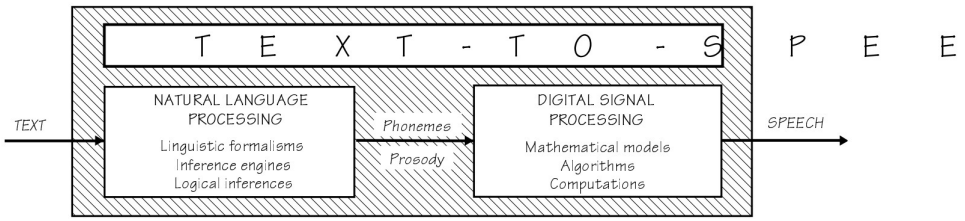

Figure 2.4: General Functional Diagram of a TTS System
[9]

Text to Speech (TTS) is one of the proposed applications of Artificial Intelligence (AI) in this project. Here, textual content which may be in any language is used to generate a corresponding speech waveform. There are many approaches to generating speech from text. The first one is the concatenative approach, where an input text is divided into phonemes from its graphemes and each phoneme has a particular waveform associated with it. All the corresponding phonemes present in the input text are concatenated to produce the fi-

nal speech. Such a method produces robot-like speech, which lacks timbre, prosody, and rhythm. Therefore, artificial neural networks are used to synthesize speech. First, an input sentence is converted into a list of phonemes. Then a neural network is trained that learns a mapping from a phoneme to a mel-spectrogram of the corresponding waveform. At the last step, there is another neural network that learns a mapping from a mel-spectrogram to a waveform. Such neural networks need to be trained on a huge amount of transcribed speech data. We can also fine-tune a speech model using another person's voice to act as a voice cloner.

2.2.3 Neural Networks

Neural Networks are one of the most popular deep learning algorithms used mainly for classification/regression tasks. Each neural network is composed of an arbitrary number of layers with the first one being the input layer and the last one the output layer. Neural Networks are essentially graph structures with nodes belonging to consecutive layers connected with each other by some weighted link. The weights of all the links present in a neural network reflect the information learned by the network during its training. Before that, the weights are randomly initialized using some probability distribution. Backpropagation is the algorithm used to update the weights of the links present in a neural network over the course of its training phase. Backpropagation is essentially calculating the error in predictions of the network using the labeled/ground-truth values and the values predicted by the randomly initialized network which are fed backward using derivatives. Essentially, neural networks try to learn a probability distribution pertaining to certain types of data and a function that can predict the class/values of previously unseen data.

2.2.4 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are specialized types of neural networks for grid-like data types such as images. It uses a concept of parameter sharing where a small set of parameters is used repeatedly between two layers. This is necessary as traditional neural networks become computationally infeasible to train with real-sized images. The generally used terminology is 'Kernel' which is the weights matrix for a certain channel. The number of channels in the input may vary. The convolution operation is used to act as a kind of filter so that each kernel is responsible for learning a certain kind of pattern observed in images. In the first few layers, kernels learn simple patterns such as vertical lines, slanted lines, etc and as we go deep into the neural network, kernels start learning more complex patterns

like faces, hands, etc. The convolutional layer is usually followed by a batch normalization layer or a max-pooling layer in order to decrease the spatial size of the input. In practice, convolutional neural layers are followed by fully connected layers before coming to the last output layer. The general architecture of a CNN is shown in **fig. 2.7**.

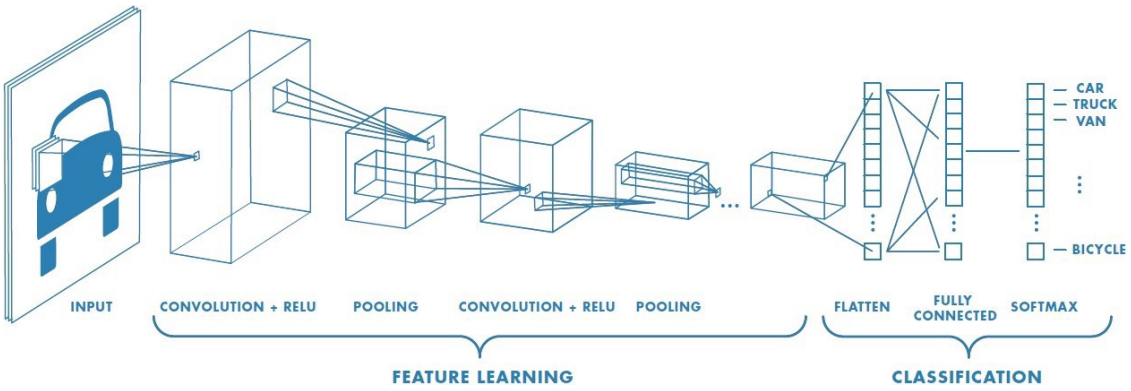

Figure 2.5: Convolutional Neural Networks
[10]

2.2.5 WaveNet

WaveNet is a deep learning model designed for generating raw audio waveforms, developed by researchers at DeepMind in 2016. It is based on the idea of a dilated convolutional neural network, which uses convolutional layers with progressively increasing dilation rates to increase the receptive field of the network while maintaining a relatively small number of parameters [11].

The WaveNet model operates on raw audio waveforms, rather than on spectral representations such as mel-spectrograms, which allows it to capture fine-grained details of the audio signal that may be lost in the transformation to the frequency domain. It is trained using a variant of the cross-entropy loss, known as the softmax cross-entropy loss, which allows it to model the distribution of the next audio sample given a sequence of previous samples.

WaveNet uses a generative model approach, meaning that it is trained to generate new audio samples by sampling from the learned distribution. During training, the model is presented with a sequence of audio samples and is trained to predict the next sample in the sequence. At generation time, the model can be seeded with an initial sequence of samples and then iteratively generate new samples by sampling from the predicted distribution at each step.

One of the key features of WaveNet is its ability to generate high-quality, natural-sounding audio, even for long sequences. This is due to the use of dilated convolutional layers, which allow the network to capture long-range dependencies without requiring a large number of

parameters. Additionally, the model incorporates skip connections that allow it to bypass some of the convolutional layers and directly connect input to output, which further improves the quality of the generated audio.

WaveNet has been used in a variety of applications, including text-to-speech synthesis, music generation, and audio signal processing. It has also been extended and adapted for use in other domains, such as image generation and natural language processing. However, due to its large computational requirements, it is typically used in applications that do not require real-time performance or that can be run on powerful hardware such as GPUs or TPUs.

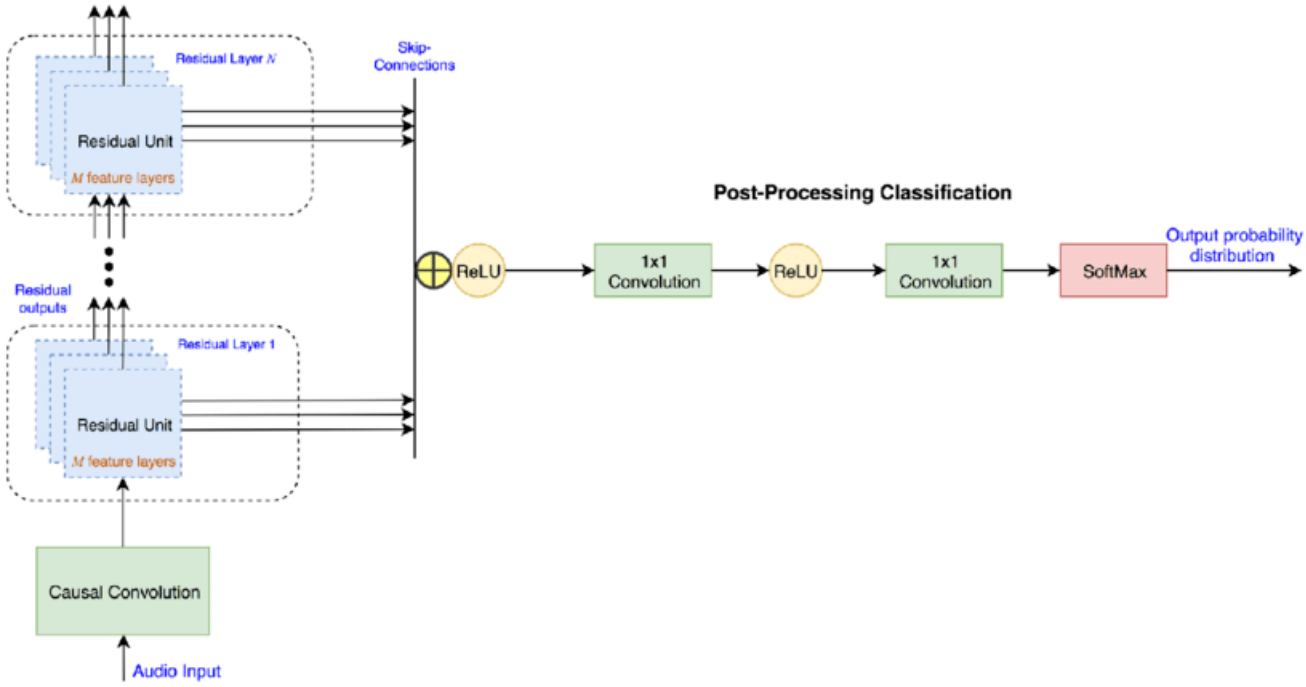

Figure 2.6: The architecture of WaveNet vocoder

2.2.6 Tacotron 2

Tacotron 2 is a deep learning-based text-to-speech (TTS) system developed by Google. It is a neural network model that can generate natural-sounding speech from text inputs. The system takes in a sequence of text characters as input and outputs a corresponding sequence of audio samples, resulting in synthesized speech.

Tacotron 2 is an improved version of the original Tacotron model, which had limitations in terms of voice quality and naturalness. Tacotron 2 uses a sequence-to-sequence model with ”attention” to generate mel spectrograms, which are then fed into a WaveNet vocoder to generate the final audio output. The model is trained using a combination of supervised

and unsupervised learning techniques, with the goal of minimizing the difference between the generated audio and the target audio.

One of the key features of Tacotron 2 is its ability to generate expressive speech, including variations in intonation, emphasis, and pacing. The model is also able to learn and reproduce different speaker styles, allowing for more natural and personalized synthetic speech. Additionally, Tacotron 2 can handle input text in multiple languages and can even switch between languages within a single sentence.

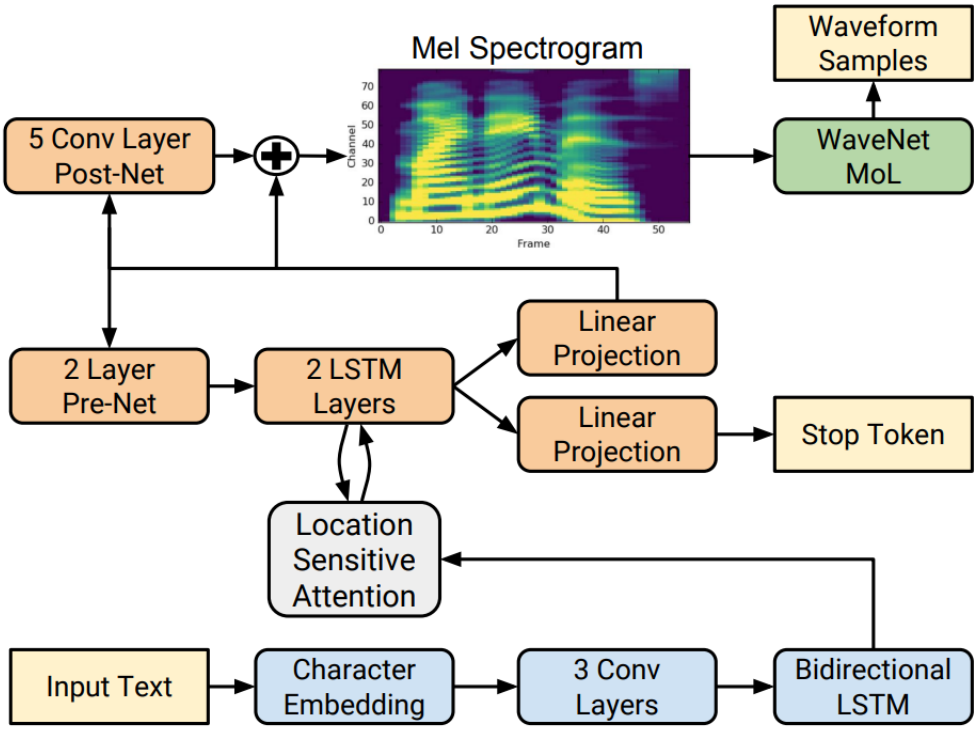

Figure 2.7: Block diagram of the Tacotron 2 system architecture [1]

As shown in **fig. 3.1**. The Tacotron 2 system consists of two major components:

1. a recurrent sequence-to-sequence feature prediction network with attention which predicts a sequence of mel spectrogram frames from an input character sequence, and
2. a modified version of WaveNet which generates time-domain waveform samples conditioned on the predicted mel spectrogram frames.[1]

Intermediate Feature Representation

In this study, Tacotron 2 bridges the two components using Mel-frequency spectrograms, a low-level acoustic representation. Tacotron 2 trains the two components independently using a representation that is easily computed from time-domain waveforms. Due to its invariance to phase inside each frame, this format is smoother than waveform samples and simpler to train with a squared error loss. A linear-frequency spectrogram, also known as the STFT magnitude, and a mel-frequency spectrogram are related. It is obtained by applying a non-linear transform to the frequency axis of the STFT and condensing the frequency information into fewer dimensions. This non-linear transform is inspired by measuring responses from the human auditory system itself, which ensures speech intelligibility. As a mel spectrogram is a simple, low-level acoustic representation of audio signals, it is then a straightforward process to condition a WaveNet model on mel spectrograms to generate audio.

Spectrogram Prediction Network

Mel spectrograms are generated in Tacotron 2 through short-time Fourier Transform (STFT) using a 50 ms frame size, 12.5 ms frame hop, and a Hann window function. The STFT magnitude in then converted to mel scale using an 80-channel mel filterbank spanning 125 Hz to 7.6 kHz, followed by log dynamic range compression. An encoder and a decoder with attention make up the network. A character sequence is transformed by the encoder into a hidden feature representation that the decoder uses to forecast a spectrogram. A learned 512-dimensional character embedding is used to represent the input characters, which are then passed through a stack of three convolutional layers, each of which contains 512 filters with the shape 5x1, or where each filter spans 5 characters, before batch normalization and ReLU activations. These convolutional layers incorporate longer-term context such as N-grams into the input character sequence. The output of these layers is then fed to a single bi-directional LSTM layer containing 512 units to generate the encoded features.

The encoder output is then consumed by an attention network that summarizes the fully encoded sequence as a fixed-length context vector. The decoder is an autoregressive recurrent neural network that predicts a mel spectrogram from the encoded input sequence one frame at a time.

In essence, the Tacotron 2 model uses simple building blocks, using vanilla LSTM and convolutional layers in the encoder and decoder.

WaveNet Vocoder within Tacotron 2

Tacotron 2 uses a modified version of the WaveNet architecture where it is trained so as to invert the mel spectrogram feature representation provided by the spectrogram prediction network. This inversion represents those features in the form of time-domain waveform samples. There are 30 dilated convolution layers, grouped into 3 dilation cycles, as shown in **fig. 2.8**.

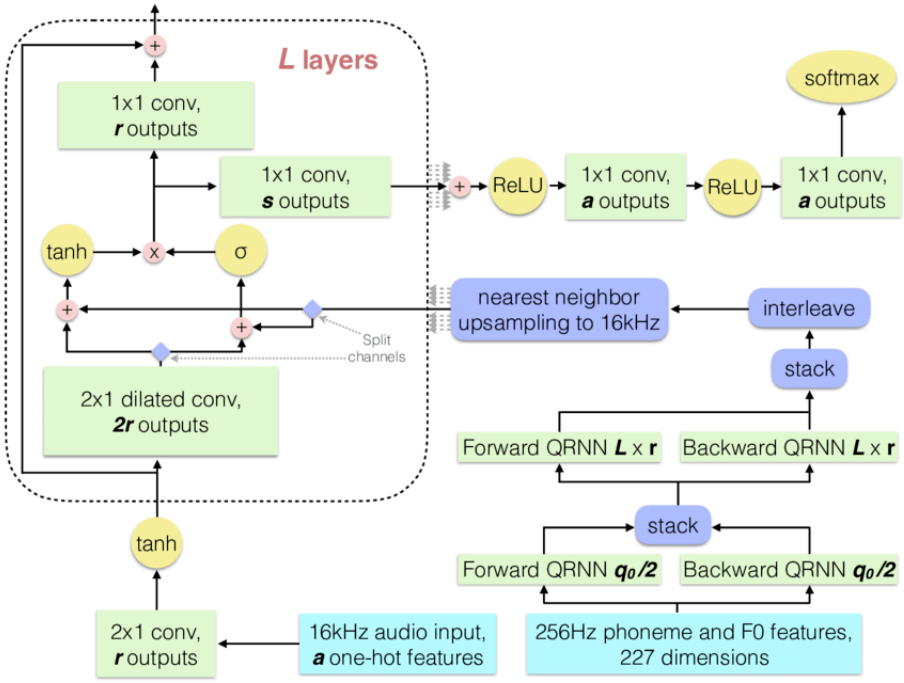

Figure 2.8: The modified architecture for WaveNet within Tacotron 2

The Mean Opinion Score (MOS) of Tacotron 2 in comparison to other notable TTS systems are shown in **Table 2.1**.

System	MOS
Parametric	3.492 ± 0.096
Tacotron (Griffin-Lim)	4.001 ± 0.087
Concatenative	4.166 ± 0.091
WaveNet (Linguistic)	4.341 ± 0.051
Ground truth	4.582 ± 0.053
Tacotron 2	4.526 ± 0.066

Table 2.1: Mean Opinion Score (MOS) evaluations for various systems [1].

2.2.7 WaveGlow

WaveGlow is a neural vocoder, which is a type of algorithm that can synthesize high-quality speech from text or other types of input signals. Developed by researchers at NVIDIA, it combines insights from Glow5 and WaveNet6 in order to provide fast, efficient, and high-quality audio synthesis, without the need for auto-regression [2].

WaveGlow is based on a type of neural network called a generative flow, which allows it to produce high-quality, natural-sounding, and highly intelligible speech. Unlike other types of neural vocoders, which require large amounts of training data to generate speech, WaveGlow is able to produce high-quality speech with relatively few training examples.

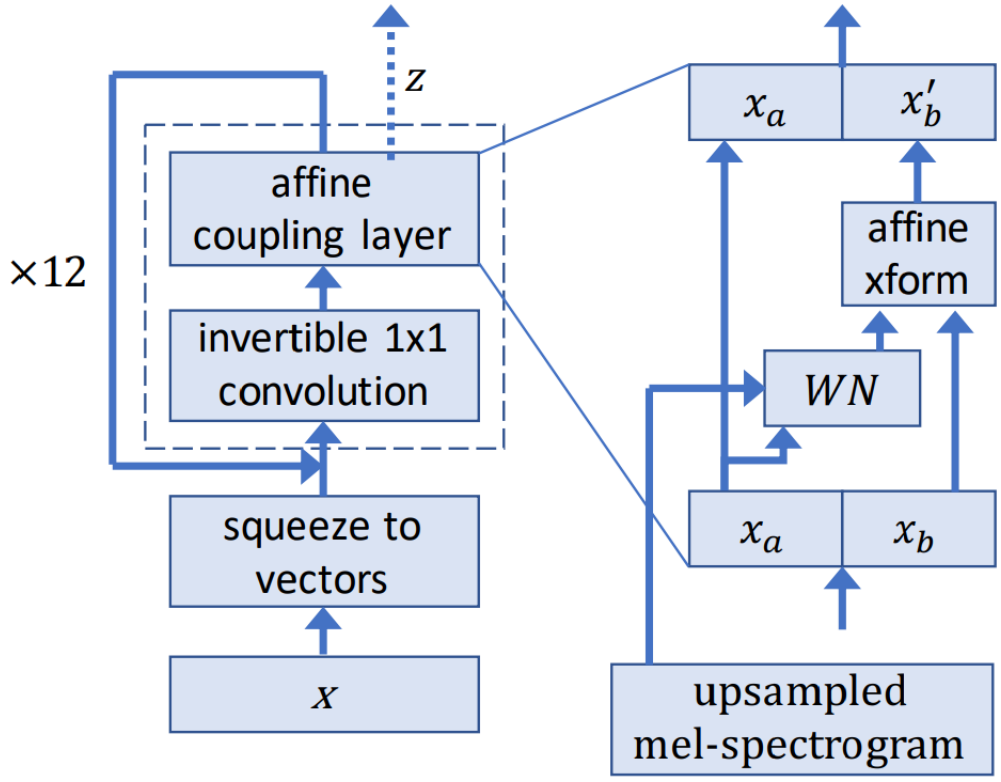

Figure 2.9: Block Diagram of WaveGlow Vocoder

Working operation of WaveGlow

It is a generative model that generates audio by sampling from a distribution. When using a neural network as a generative model, samples from a simple distribution—in this case, a zero mean spherical Gaussian with the same number of dimensions as the desired output—are processed through a series of layers to convert the simple distribution into one that has the desired distribution. In this instance, WaveGlow models the audio sample distribution conditioned on a mel-spectrogram.

The Mean Opinion Score (MOS) of WaveGlow in comparison to other notable vocoders are shown in **Table 2.2**.

Model	MOS
Griffin-Lim	3.823 ± 0.1349
WaveNet	3.885 ± 0.1238
Ground Truth	4.274 ± 0.1340
WaveGlow	3.961 ± 0.1343

Table 2.2: Mean Opinion Score (MOS) evaluations for various vocoder systems [2].

2.2.8 Natural Language Processing

Natural Language Processing (NLP) is a field of analysis of linguistic content, either in textual or speech form. It mainly focuses on the interaction between natural language and computers. NLP is a large field with a long history spanning before machine learning. In traditional NLP, a rule-based approach is taken where different linguistic concepts such as syntax trees are used to carefully analyze the textual content. In contrast, ML-based NLP is all about using statistics to learn patterns in textual data. The first step in most NLP involving textual data is preprocessing or cleaning the data so that everything is in the form of lowercase words. Then all the works are tokenized and latent representations of individual words are generated so that they can be fed to a neural network. Since it is impossible to predict the length of a sentence, neural networks have been modified to work with sequential data. These neural networks are known as Recurrent Neural Networks. Nowadays, an even more powerful concept called 'attention' is used in deep learning which is now customary in many NLP-related tasks. The general pipeline of steps within an NLP task is illustrated in **fig. 2.10**.

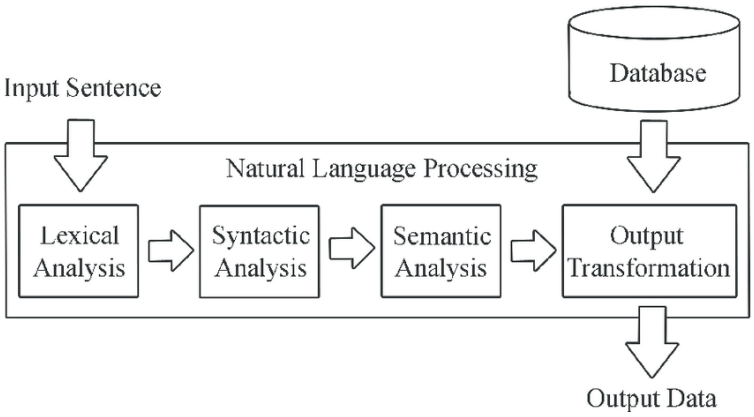

Figure 2.10: Natural Language Processing steps
[12]

2.2.9 Markov Chain

Markov chains are a mathematical concept that has been used in a wide range of applications including natural language processing, genetics, economics, and physics. In the context of natural language processing, Markov chains have been particularly useful for modeling the probabilistic relationships between words in a sentence or text.

In the context of natural language processing, Markov chains have been widely used for next-word prediction. A Markov chain is a probabilistic model that describes a sequence of events where the probability of each event depends only on the state of the previous event, not on any earlier events. In the case of next-word prediction, a Markov chain can be used to estimate the probability distribution of the next word given the previous words in a sentence. In the 1950s, Claude Shannon and his colleagues developed the idea of n-gram models, which are a type of Markov chain that models the probability of a word given its n-1 previous words. N-gram models have been widely used for various natural language processing tasks, including speech recognition, machine translation, and text classification.

The basic idea is to represent the sentence as a sequence of words, where each word is a state in the Markov chain. The probability of the next word is then estimated based on the conditional probabilities of each word given its previous words. This can be achieved by estimating the transition probabilities between the different states of the Markov chain, which can be done using statistical methods such as maximum likelihood estimation or Bayesian inference.

In addition to traditional Markov chains, there are several variations and extensions that have been proposed for natural language processing tasks. For example, hidden Markov models (HMMs) are a type of Markov chain that incorporates hidden states to model latent variables, and have been widely used for speech recognition and named entity recognition. Conditional random fields (CRFs) are another type of Markov chain that model the joint distribution of the output variables given the input variables and have been used for various sequence labeling tasks.

In the context of the Nepali language, previous studies have used Markov chains for various natural language processing tasks, including part-of-speech tagging, sentiment analysis, and machine translation. However, to the best of our knowledge, there is limited research on using Markov chains for next-word prediction in the Nepali language. Therefore, this project aims to implement Markov chains to aid next-word prediction as part of the text prompt system with lexical configuration.

2.2.10 Language Models and N-gram models

Language Models are computational statistical models that are used to predict the probability of the occurrence of a sequence of words or characters in the written form of a language. They are generally trained on a large corpus of text from various sources and used in various applications such as sentence completion, text generation from a prompt, text summarization, question-answering models, chatbots etc. The goal of a language model is to understand the underlying patterns of words and sentences within a language and to predict the occurrence of the most probable words when given a certain context.

At its very core, all that a language model does is predict the next most probable word in a sequence based on the previously predicted sequence of words. There actually is a limit we can set on how many of the previous words to be considered when trying to predict the word. Unigram language models do not consider any previous words and try to predict words based on their frequency of occurrence in the corpus. Bigram language models depend only on one word to predict the next word. Similarly, trigram language models use two previous words and four-gram language models use three previous words to predict the next most probable word.

3. Methodology

3.1 Problem Recognition

Creating augmentative communication support for the physically disabled involves a lot of steps. The first step towards creating this application would be the generation of Nepali speech from a textual prompt for Devnagari. To cater the application to a physically disabled audience suffering from different conditions, the next step would be augmentation and assistive function. For the second part of this project, the use of traditional keyboard-based feedback will be replaced by a "text prompt"-aided interface where even a physically disabled user can interact with the application with touch gestures. After achieving an accessibility-oriented UI design, the user has to select a certain combination of words that they need to communicate.

The first problem is a speech synthesis problem. For this, proper pre-processing of text input must be done on the end-user device and the prompt should be fed through a TTS model to generate Nepali speech. The second problem is the accessibility problem. The application needs to detect the person's intended input through accessible media. For this, certain calibrations might be required on the end-user device.

3.2 Data Collection

For the TTS engine using the Nepali language, the SLR43 dataset from OpenSLR containing 2064 spoken Nepali sentences has been utilized [13].

3.3 Frontend Development

The front end of this application has been developed using Material UI. It is a comprehensive library of components that features the implementation of Google's Material Design System. Flutter has been used as the Android application development framework for this project.

The major challenge in front-end design has been usability. The target users of this application are not the typical mobile phone users but vocally impaired users facing speech impediments due to a variety of reasons, be it vocal, physical, or something else. So, the

graphic design in the application should be tailored according to the specific impediment faced by the user.
For this purpose, the application proposes a 'Hands-On' feature, where physically impaired users unable to type comprehensive text can use touch gestures for text selection and choose the text they want to be uttered through a lexical configuration provided by the application.

3.4 Backend and Database

Flutter has been the primary framework for Android application development. It is a cross-platform mobile application development framework based on Dart language. The Foundation library written in Dart provides all the classes and functions required to build applications in Flutter.
Python has been used as the primary language for the backend of this application. The Tacotron2 model has been utilized using PyTorch and TensorFlow in a customized Anaconda environment, with the use of Flask micro-framework to function as a localhost server. This localhost server has then been tunneled to the Internet using ngrok, a simplified API-first ingress-as-a-service that adds connectivity to the back-end Python code. Similarly, Python has also been used to provide lexical next-word prediction, which facilitates word recommendation to the user.

3.5 Algorithm Implementation

3.5.1 Text-to-Speech

Tacotron 2

The primary approach to building a TTS model has been to train a Tacotron2 model. Tacotron2 is a transformer-based architecture proposed by Google comprising a neural network architecture for speech synthesis directly from the text. It contains a recurrent sequence-to-sequence feature prediction network with attention that predicts a sequence of mel spectrogram frames from an input character sequence. The training pipeline for the Tacotron 2 model includes transfer learning where a pre-trained English female voice model based on the LJ Speech Dataset is used. The pre-trained model used for this project is named 'tacotron2_statedict.pt'. It is fed with the OpenSLR speech dataset to tweak its usage for Nepali TTS.

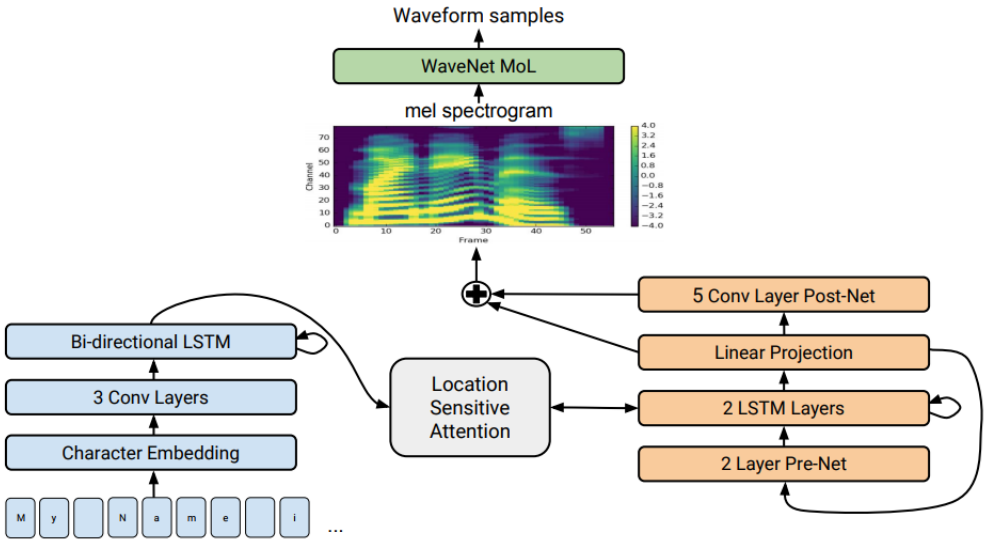

Figure 3.1: Block Diagram of Tacotron 2

[1]

WaveGlow Vocoder

Instead of a traditional Tacotron2 architecture, where the mel spectrogram is fed to a WaveNet vocoder to generate speech, we use WaveGlow. The pre-trained model used for this purpose is named 'waveglow_256channels_ljs_v3.pt'

3.6 Mobile Application

The increasing demand for mobile applications has paved the way for the development of a diverse range of applications in the mobile platform. Mobile applications have become an integral part of our lives, allowing us to perform a wide range of tasks with just a few clicks. With the advancement in mobile technology, developers are now able to create feature-rich applications that can run on a wide range of mobile devices.

As a result, Aawaj has also been primarily developed as a mobile application. In this section, we will discuss how Aawaj has been designed for the mobile platform.

3.6.1 Test Bench

- **Device** : Google Pixel 3A
- **OS** : Android 13
- **Chipset** : Qualcomm SDM670 Snapdragon 670
- **GPU** : Adreno 615

- **RAM** : 16 GB

3.6.2 Application Details

- Compatible Platforms: Web, iOS, Android (preferred)

3.7 Integration

After the TTS models are trained, the developed mel-spectrogram generating Tacotron 2 model is integrated with a pre-trained WaveGlow model. This helps synthesize voice from text prompts. After assembling a speech-synthesizing system, they need to be integrated with the Flutter application. The approach towards this integration has been the use of API. The TTS model has been run on an ad-hoc server at home using Flask. This Flask API has then been tunneled to the Internet using ngrok. The Flutter application accesses this ngrok endpoint every time it is run as an Android application.

3.8 Requirements

3.8.1 Tools and Technology Used

The different tools and technologies proposed to be used by this project are:

- **Frontend framework:** Flutter
- **Backend frameworks:** Flutter, Flask, ngrok
- **Programming Languages:** Dart, Python
- **IDE frameworks:** Android Studio, VS Code, XCode
- **Version Control:** GitHub
- **Prototyping:** Figma
- **Audio Processing Packages:** ffmpeg, PyAudio, SpeechRecognizer

4. System Design

4.1 System Class Diagram

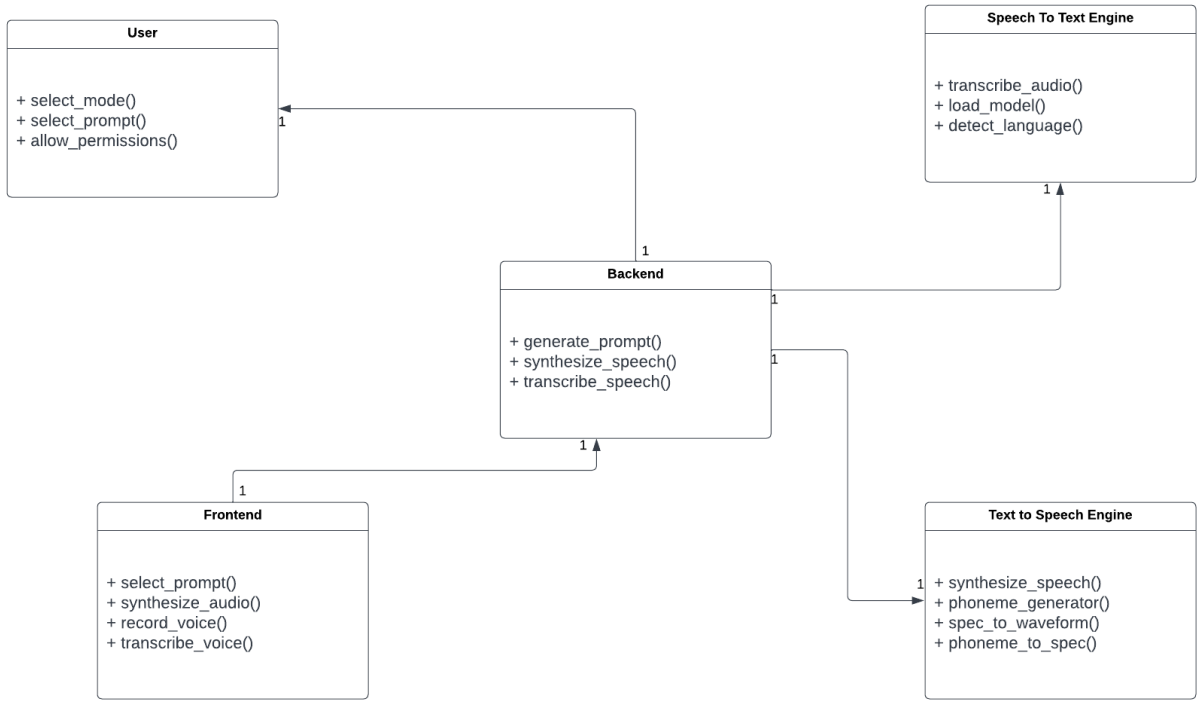

Figure 4.1: Class UML Diagram of Aawaj

4.2 Training Pipeline

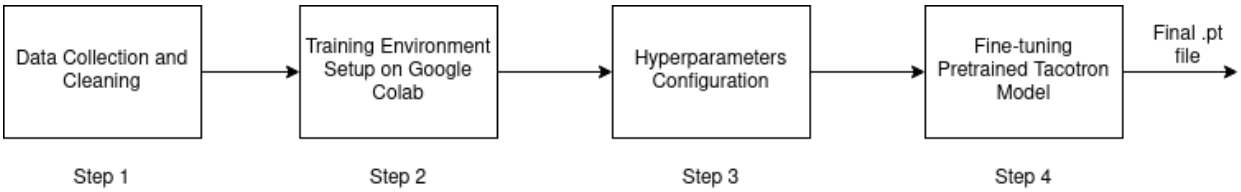

Figure 4.2: Training pipeline for TTS Engine

4.3 Sequence Diagram

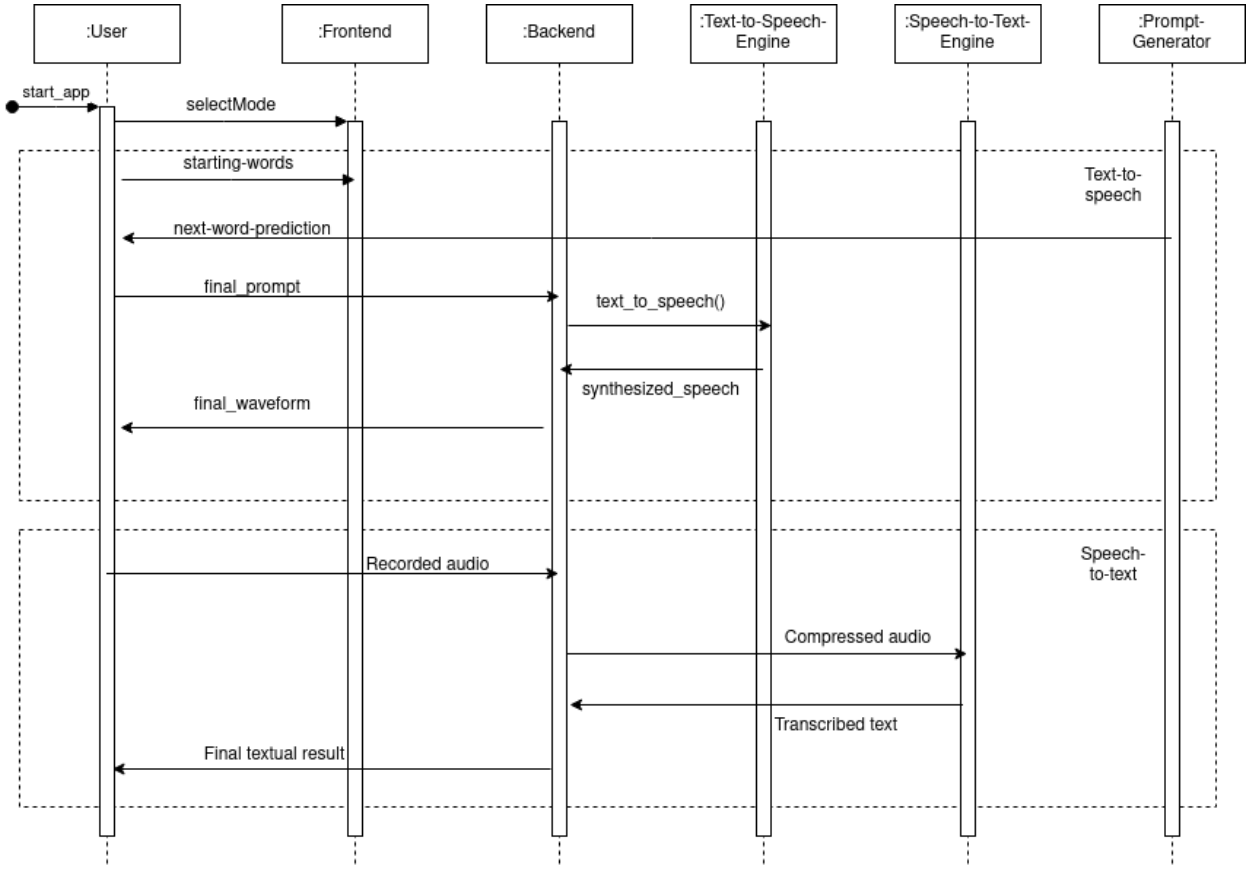

Figure 4.3: Sequential UML Diagram of Proposed System.

4.4 TTS Pipeline

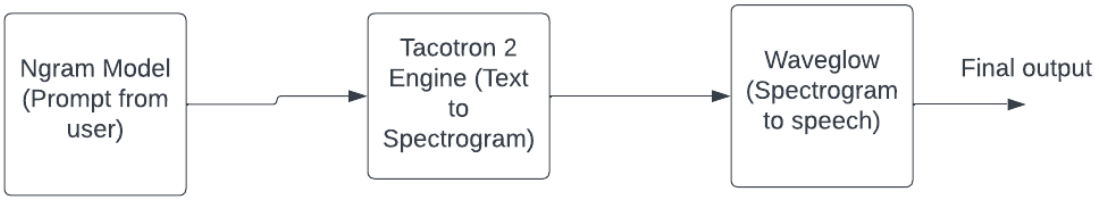

Figure 4.4: Inference pipeline for TTS Engine

4.5 ASR Pipeline

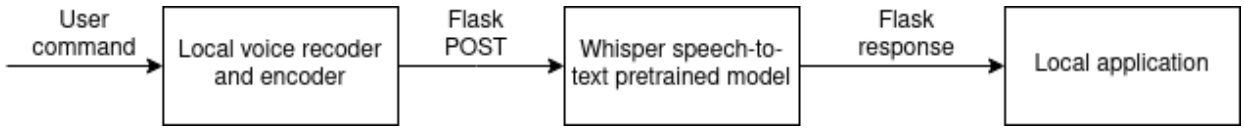

Figure 4.5: Inference pipeline for ASR Engine

5. Results and Discussion

Augmentative and assistive speech synthesis technology has become increasingly popular in recent years, particularly in mobile platforms. This technology can be used to help individuals with speech or language impairments to communicate more effectively, by converting text into synthesized speech. In the case of the Nepali language, this technology can be particularly useful, given the relatively low literacy rates and high linguistic diversity in Nepal.

The results of an augmentative and assistive speech synthesis technology for the Nepali language in the mobile platform would depend on a variety of factors, including the accuracy and quality of the speech synthesis, the ease of use of the technology, and the level of customization and personalization available to the user.

In terms of accuracy and quality, the success of Nepali language speech synthesis technology would depend on the availability of high-quality text-to-speech models, as well as the ability of the technology to accurately recognize and interpret different accents, dialects, and speech patterns.

Thus, the results of this project are discussed in terms of the training and quality of the TTS model, as well as the useability of the mobile application.

5.1 TTS Training

With a focus on providing an accessible user experience for the vocally impaired, the impetus for training a TTS model has been imperative where the synthesized speech needed to be well-structured, understandable, and natural sounding.

The model was trained on Google Colab with the following hyperparameters:

```
hparams.p_attention_dropout=0.1
hparams.p_decoder_dropout=0.1
hparams.decay_start = 15000
hparams.min_learning_rate = 1e-5
generate_mels = True
hparams.show_alignments = True
alignment_graph_height = 600
alignment_graph_width = 1000
hparams.batch_size = 32
```

```
hparams.load_mel_from_disk = True
hparams.epochs = 100
```

The training and validation loss metrics are shown in **fig. 5.1** and **fig. 5.2**, respectively.

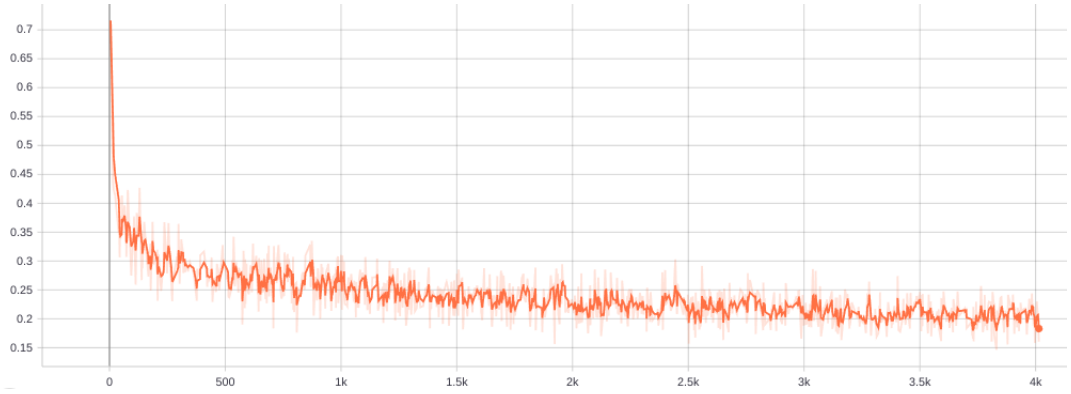

Figure 5.1: Training Loss for Tacotron 2 model

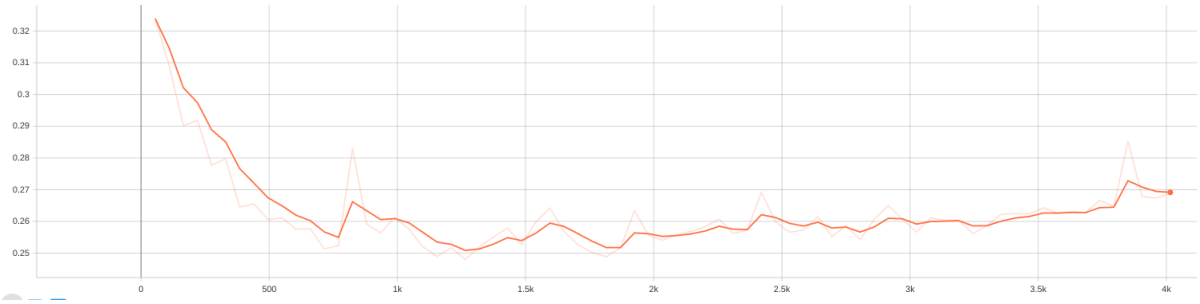

Figure 5.2: Validation Loss for Tacotron 2 model

The validation metrics show how the model began to overfit upon 2 hours of training the model on a Nepali dataset of 2000 utterances.

5.2 Evaluation

For the evaluation of the TTS model, 3 sentences were randomly selected as the evaluation set of the TTS model. Along with Aawaj's in-house TTS model, the highly-acclaimed Nepali language TTS provided by Microsoft Azure as well as the TTS utilized by NepaliSpeech.com (FliteTTS) were compared in order to provide context regarding the standing of Aawaj's TTS quality.

Among the three models, it should be noted that NepaliSpeech.com is an open-source alternative developed in 2016 as a stop-gap application to provide Nepali TTS features when there were no open datasets for the Nepali language. The OpenSLR dataset used by

Aawaj was published in 2018. As a result, it utilizes Hindi language utterances to provide a "Nepali-like" TTS with Hindi accents.

Furthermore, the Azure TTS provided by Microsoft is a commercial application behind a paywall rather than an open-source technology.

For the evaluation, the subjective Mean Opinion Score (MOS) of human listeners was recorded as part of the survey. On a scale of 1-5 with "1" standing for bad and "5" standing for excellent, the sets of audio generated by these models were then evaluated by 24 people as part of the evaluation survey. The results are shown in **Table 5.1**.

Model	MOS
Aawaj	3.6042
Azure Nepali TTS	4.2292
NepaliSpeech	1.6458

Table 5.1: Mean Opinion Score (MOS) evaluations for various available Nepali TTS.

A key interpretation of these scores is that Microsoft Azure's Nepali TTS is currently the best in the market. Since it has been developed by a large-scale conglomerate, it should be noted that Microsoft recorded its own proprietary Nepali speech dataset of 2 distinct male and female speakers. This superior quality of the dataset is one of the major reasons behind its excellence, rather than the underlying technology being so vastly superior. As a result, the MOS of Azure TTS is 4.2292.

Aawaj, on the other hand, has trained its model by employing transfer learning on a pre-trained Tacotron2 model with an English language female voice. Furthermore, it has been trained on a relatively low-quality dataset with 19 different speakers, which is not optimal when training a TTS model. As a result, the MOS score for Aawaj is 3.6042. The lower MOS score compared to Azure, as previously mentioned, is primarily attributed to the difference in the dataset quality.

The NepaliSpeech model, when compared to the other 3 models, comes off as the worst with an MOS of 1.6458 as it utilizes a Hindi language dataset.

The MOS scores for each individual sentence achieved by the three models are also shown in **Table 5.2**.

Model	MOS		
	मलाई भोक लाग्यो	तिमी मलाई रमाइलो लाग्छ	मेरो नाम कमला हो
Aawaj	3.5000	3.5000	3.8125
Azure Nepali TTS	3.9375	4.5625	4.1875
NepaliSpeech	1.6875	1.6250	1.6250

Table 5.2: Mean Opinion Score (MOS) evaluations for individual sentences.

5.3 Linguistic Diversity

The OpenSLR training dataset comprised a total of 19 female speakers along with 2000 utterances. It was noted early on that most of the speakers in the dataset exhibited Newari accents in their utterances. So the linguistic features of the dataset are shown in **Table 5.3**.

Voice Model	Gender	Language
Model-1	Female	Nepali

Table 5.3: Linguistic features of the TTS model.

The linguistic diversity of a custom-made Nepali language TTS model is an essential consideration to ensure that the model can produce high-quality synthesized speech for a wide range of users. If the training dataset primarily consists of voices with a Newari accent, it is crucial to evaluate how well the TTS model can handle other Nepali accents and dialects.

Nepali is the official language of Nepal and is spoken by a majority of the population. However, Nepal is home to many other languages and dialects, with Newari being one of the most prominent. Newari is spoken primarily in the Kathmandu Valley and is considered to be one of the oldest languages in Nepal. It has a unique tonal system, which differs significantly from other Nepali dialects.

While a custom-made Nepali language TTS model that displays a Newari accent can be useful for those who speak Newari or are accustomed to its tonal system, it may not be ideal for individuals from other regions or communities who speak Nepali with a different accent or dialect. Therefore, it is essential to evaluate the TTS model's performance across a range of Nepali accents and dialects to ensure that it can produce high-quality synthesized speech for a wide range of users.

Overall, a custom-made Nepali language TTS model that displays a Newari accent is a positive development, as it can provide a voice for Newari speakers and help to preserve the

language's unique tonal system. However, to ensure that the TTS model is accessible and effective for all Nepali speakers, it is necessary to evaluate its performance across a range of accents and dialects and make any necessary adjustments to improve its accuracy and naturalness.

It also shows the need to add a male voice, provided a proper dataset exists, along with the provision of a general dialect that represents the entire Nepali-speaking population for universally acceptable usage.

5.4 Accessibility and Augmentation

The current result of this project has been a mobile application that can be used by speech-impaired users for ease of their communication. Currently, this application has been best catered to people with hearing and speaking difficulties who are capable of providing traditional touch-based input from their mobile phones. The user can either use a keyboard to type romanized Nepali text as input or they can tap the given morphologically arranged set of words to create a sentence, which greatly decreases the time taken to enter a sentence, thus increasing the quality of life for the users.

Accessibility

When it comes to accessibility, the application provides text prompts to encourage user input, which is significantly more accessible to disabled users when compared to the problems that typing in a keyboard layout may pose to people with disabilities such as autism, cerebral palsy, etc. It provides the option to the user whether they want text prompts or a keyboard layout, as shown in **Figure 5.3**.

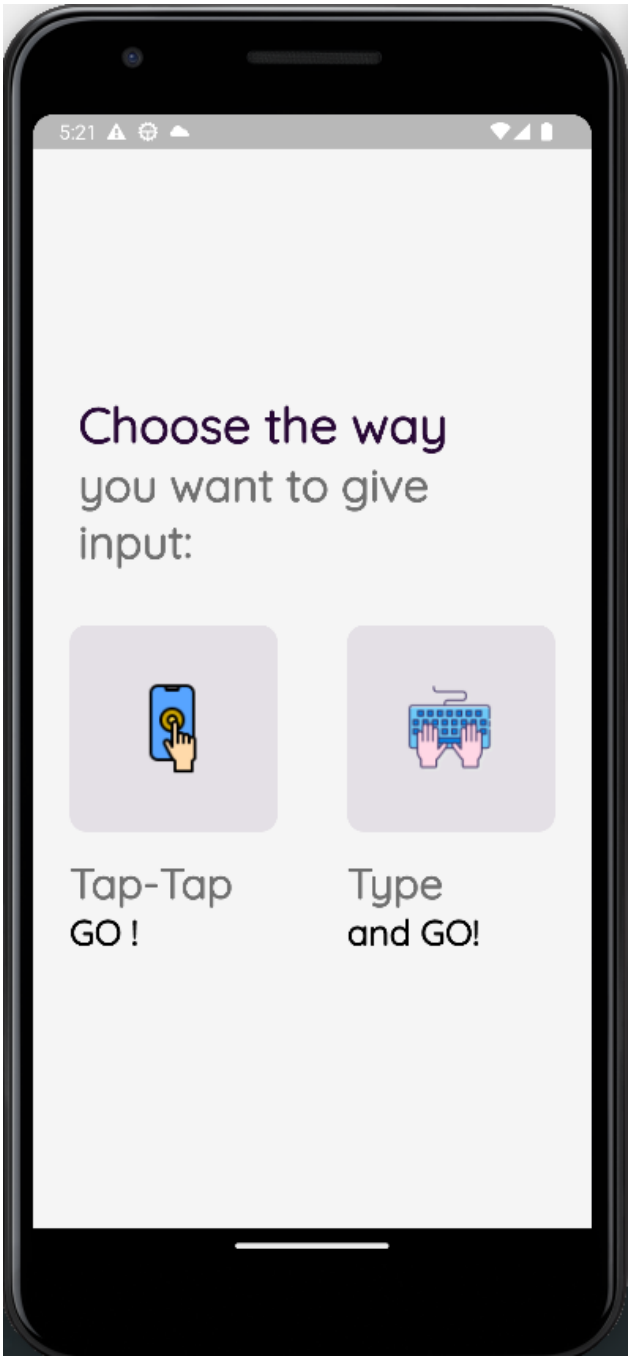

Figure 5.3: Default application landing page with accessibility choice

Similarly, in terms of text prompts, it provides large text prompts that are easy to select along with the feature of adding suffixes to them, as shown in **Figure 5.4**.

Figure 5.4: Text prompt with discrete choices

Augmentation (Lexical Configuration)

In terms of augmentation, the application provides the lexical configuration of text in the form of "Subject+Object+Verb" as ruled by Nepali grammar. As a result, it forms an augmentative platform where the Nepali lexicon is adhered to, resulting in concise understandable utterances. This configuration is shown in **Figure 5.5**.

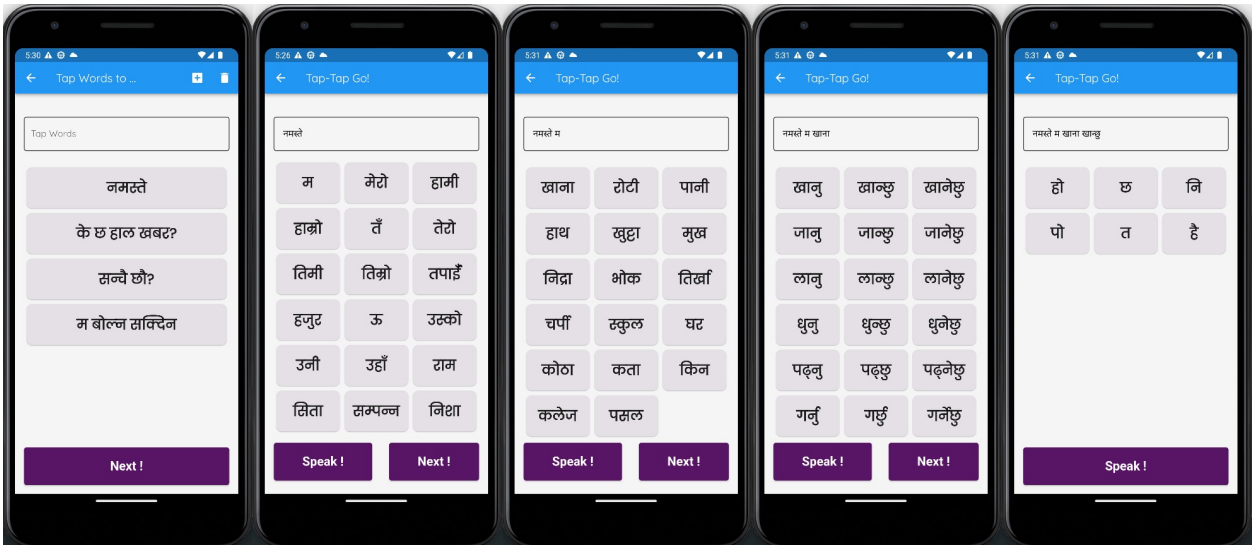

Figure 5.5: TTS text prompts in lexical configuration

5.5 Third Party Integrations

The Aawaj TTS model is being integrated into the AsTeRICS Grid web application as an open-source contribution to the field of Accessible Technology (AT). It is being integrated to add a Nepali language mode to the already existing pool of AAC features for languages such as English, German, French, Spanish, Chinese, etc. Integrating with an application such as AsTeRICS Grid, with more than 5000 daily users, is an important step towards reaching the secondary objective of the project to open-source its technologies and applications to the target population of speech-impaired Nepalese.

Because it is being integrated with a larger, open-source community of developers, the AsTeRICS Grid application helps provide eye-tracking, head-tracking as well as alternative modes of control to the user. It also provides the presence of Aawaj's in-house TTS in tablets and computers, as opposed to the solitary mobile application described in this project. As a result, it enhances the ability of the application to take advantage of the client machine's superior computing capability to run eye-tracking, head-tracking, and alternative forms of control input. Similarly, it also provides a more icon-oriented communication platform for people with cognitive as well as physical disabilities.

5.6 Application Screenshots

The screenshots of the current application that has been built so far are given below:

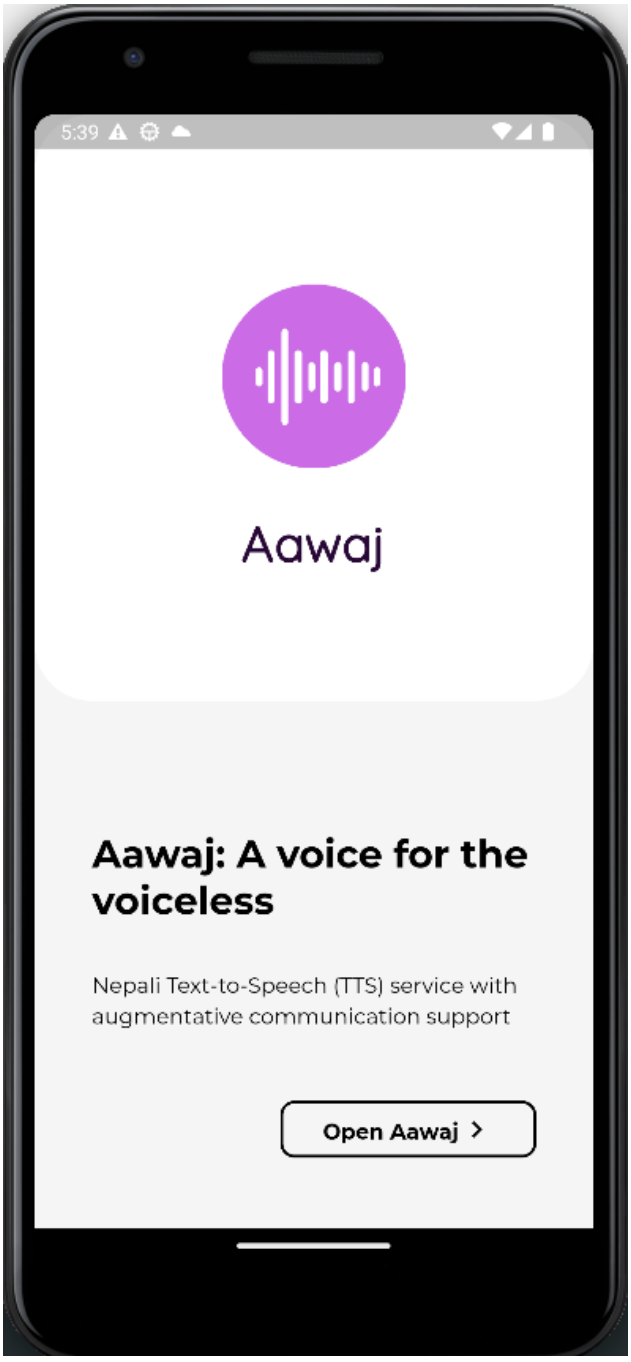

Figure 5.6: Default application landing page

Figure 5.7: Custom user-specific text prompts

Figure 5.8: Adding custom user-specific text phrases

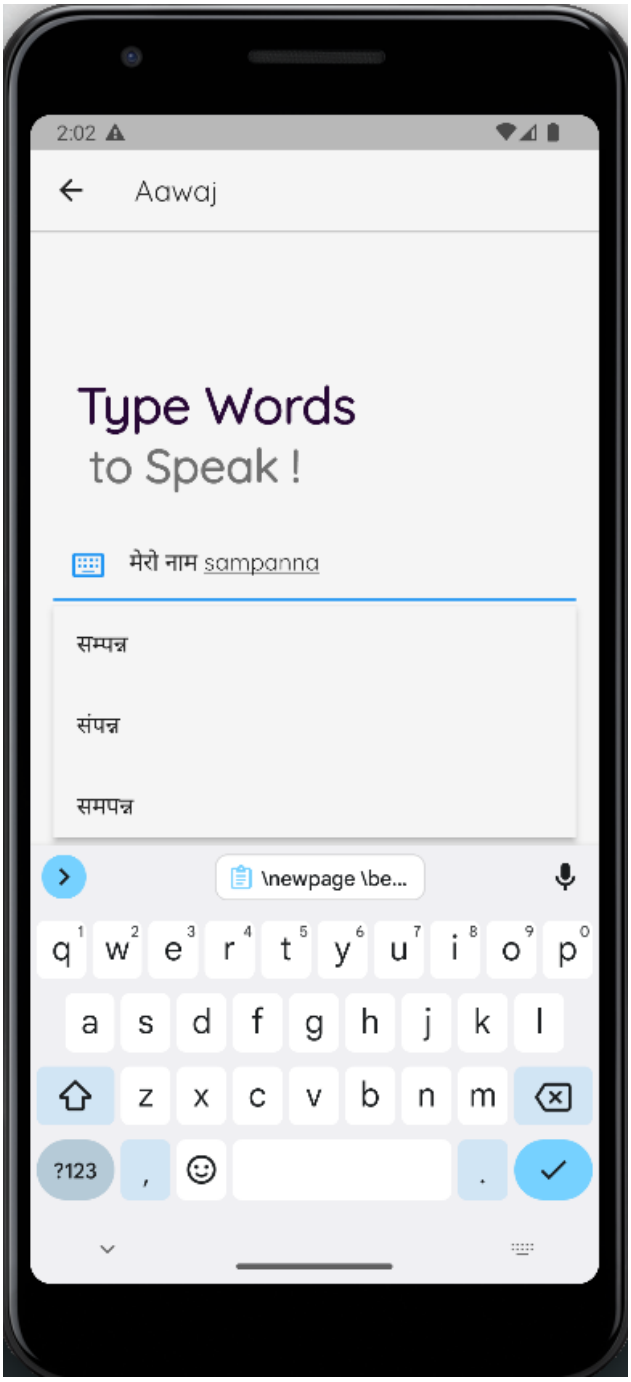

Figure 5.9: Romanized Nepali typing supported by transliteration

Figure 5.10: Speech synthesis screen

6. Conclusion

In conclusion, this project strives to address and provide a much-needed intervention in the space of Nepali TTS as well as speech impairment accessibility in mobile applications. The combination of a Nepali TTS application with special accessibility features for the physically impaired addresses the two striking problems in the context of the speech-impaired populace of Nepal.

The project utilizes a state-of-the-art TTS model for Nepali voice synthesis and proposes the creation of an ingenious accessibility feature for the physically impaired to gain access to a trustworthy TTS engine.

The successful completion of the primary objectives of this proposed project has resulted in a vital piece of technology that could not only improve the use of Nepali TTS engines for voice synthesis but also improve the accessibility features that are provided to the physically impaired in the context of technology specific to the Nepalese domain. Open-sourcing this model with further help in future endeavors of developers to provide AAC features for the speech-impaired population of Nepal.

Similarly, with an operational NLP system to recommend text prompts, it also provides unique accessibility features to the users in the form of next-word prediction, optimizing the input flow from the user. Furthermore, the accessibility-oriented UI provides a distinctive interface for the user to choose communication prompts that help ease day-to-day interactions while also providing an avenue for 2-way communications with people who may not be fluent in sign language. It provides a more meaningful platform that helps relieve some of the burdens faced by people with speech disabilities in their day-to-day activities in the context of Nepal.

Integration with third-party platforms like AsTeRICS also puts this project in a unique position where it can maneuver the resources provided by a large open-source community, in the form of model deployment, application reach, and open-source innovation in the domain of Nepali TTS and AAC in the context of Nepal.

All things considered, this project has met its primary objective of creating a Nepali TTS with AAC features in a mobile application, along with its secondary objectives of open-sourcing the TTS model and striving to deploy it as a free-to-use application to the target populace.

7. Limitations and Future Enhancement

7.1 Limitations

Despite having a working mobile application, the project still has some limitations. These limitations are listed below:

- **Lack of iOS Support:** Despite the application running on iOS, the actual TTS engine fails to run on iOS platforms due to the strict API access policies and the requirement of SSL certificates by Apple devices.
- **Need of active Internet connection:** Since the TTS model is very heavy, it doesn't run on mobile devices natively, causing the need for an external API to fetch the TTS feature, disabling offline usage.
- **Ad-hoc Server:** Due to the extremely niche requirements of the TTS engine as well as heavy GPU usage, the model is very expensive to be deployed in the cloud, causing the need for an ad-hoc server run on a personal computer.
- **Lack of diversity in TTS voice:** The TTS is trained on a solitary female dataset, hence not providing diversity in the voice.

7.2 Future Enhancements

The future enhancements that can come with the application in future iterations are listed below:

- **Catering to cognitive disability:** The addition of icons to each text prompt can make the application open to people with cognitive disabilities as well as speech impairment.
- **iOS Support:** With the addition of credible SSL certificates to the Flask API, the application can be run on iOS too.
- **Native TTS usage:** Optimization of the TTS model in future iterations can make it available natively to the user, providing offline usage.
- **Cloud deployment:** The application can be deployed in the cloud to ensure proper content delivery.

- **Different voicelines, accents:** The addition of different accents, dialects, and voices can be a future enhancement to the TTS engine.
- **Eye tracking accessibility:** The use of eye-tracking and head-tracking can be a feature in future iterations, catering to physically disabled people with ALS, spinal cord injuries, and so on.
- **Transition to other languages:** The application can then be expanded to include languages other than Nepali too.

References

[1] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, Rif A. Saurous, Yannis Agiomvrgiannakis, and Yonghui Wu. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783, 2018.

[2] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. *CoRR*, abs/1811.00002, 2018.

[3] Mukti Prakash Thapaliya. A Report on Disability in Nepal. *Australian Himalayan Foundation (AHF)*, 2016.

[4] Disability Research Center. Disability Atlas Nepal.

[5] World Health Organization. Priority Assistive Product List of Nepal.

[6] Pitt Hopkins Research Foundation. Augmentative Alternative Communication.

[7] Thierry Dutoit. *An Introduction to Text-to-Speech Synthesis*. Springer, 1997.

[8] Richard Sproat. *Multilingual Text-to-Speech Synthesis: The Bell Labs Approach*. Kluwer Academic Publishers, 1998.

[9] Thierry Dutoit. High-quality text-to-speech synthesis : an overview. 1:4–5, 2004.

[10] Sumit Saha. A comprehensive guide to convolutional neural networks, Oct 2018.

[11] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.

[12] Chalermpol Tapsai, Phayung Meesad, and Choochart Haruechaiyasak. Tls-art: Thai language segmentation by automatic ranking trie. Dec 2016.

[13] Keshan Sodimana, Knot Pipatsrisawat, Linne Ha, Martin Jansche, Oddur Kjartansson, Pasindu De Silva, and Supheakmungkol Sarin. A Step-by-Step Process for Building TTS Voices Using Open Source Data and Framework for Bangla, Javanese, Khmer,

Nepali, Sinhala, and Sundanese. In *Proc. The 6th Intl. Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU)*, pages 66–70, Gurugram, India, August 2018.