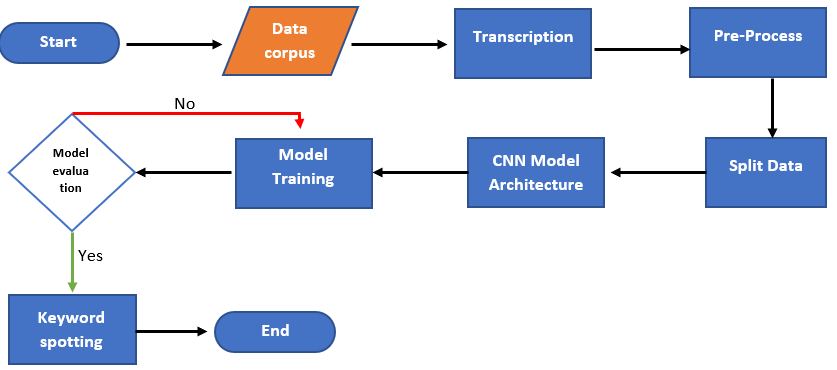
**CHAPTER 3**

# DESIGN AND METHODOLOGY

This chapter contains method, discussions, tools, and illustrations that will be used to develop the proposed study.

## 3.1 Framework of the Study



*Figure 4. framework of the study*

Figure 4 provides a visual overview of the entire ARAVoE development process, serving as a detailed roadmap from start to finish. The evaluation phase is depicted by a red arrow, representing a feedback loop where, if the model produces suboptimal results, it is sent back to the training stage for further improvement. On the other hand, the green arrow indicates successful training results, marking the model’s advancement to the key spotting phase.

## 3.2 Model Design

### 3.2.1 Collected Audio Data

In our data collection approach, we aim to gather 500 audio recordings, accompanied by translations and transliterations of Arabic from Madrasah Education Program (MEP) teachers and their students. If the collected data is insufficient, we will supplement our dataset with recordings sourced from the internet. Our data collection methods include surveys, interviews, and meetings with the MEP focal person from the Department of Education (DepEd). Additionally, we will explore various online platforms to access audio data that can enrich our collection. This multifaceted approach ensures that we have a robust and diverse dataset for our project, enhancing the quality and accuracy of the ARAVoE model.

Learning Arabic is not easy; it involves rules, vowels, and special characters, adding complexity to the language learning process. With the help of the data being gathered, the researchers have a concrete understanding of the Arabic language and the specific needs and preferences of Arabic teachers in Butuan City.

### 3.2.2 Transcription

We utilized a straightforward yet effective method. The teachers provided us with a dictionary of Arabic words and their English translations. To ensure accurate pronunciation and clarity, we had three students record the words individually, one after the other. This approach not only allowed us to capture diverse pronunciations but also enabled us to cross-verify the accuracy of the recordings. Each student was instructed to focus on delivering clear and precise enunciation, while the teachers monitored the process to ensure adherence to correct Arabic pronunciation. This systematic transcription method will contribute to a rich and reliable dataset for the ARAVoE model, facilitating better understanding and representation of Arabic vocabulary in English.

## 3.3 Arabic Recognition Assisted Vocabulary of English (ARAVoE Model)

The ARAVoE Model (Arabic Recognition Assisted Vocabulary of English) is an integrated system designed to translate spoken Arabic audio into English text. This model combines the capabilities of Automatic Speech Recognition (ASR) and Machine Translation (MT) to facilitate seamless communication and understanding between Arabic and English speakers. The ARAVoE Model follows a structured workflow, consisting of data preprocessing, feature extraction, model training, and real-time translation, effectively bridging the gap between the two languages.

### 3.3.1 System Architecture

The ARAVoE Model comprises two primary components: the ASR model and the MT model.

#### 3.3.1.1 ASR Model

The ASR model is responsible for converting Arabic audio input into text. It utilizes advanced feature extraction techniques, specifically Mel-Frequency Cepstral Coefficients (MFCCs), to capture the acoustic features of the speech.

A Convolutional Neural Network (CNN) architecture is employed to learn the relationships between audio features and Arabic phonemes or words. The output of the ASR model is a transcription of the spoken Arabic text.

#### 3.3.1.2 MT Model

Once the Arabic text is generated by the ASR model, the MT model takes over to translate this text into English. The MT model leverages a Transformer architecture, known for its efficiency and accuracy in translation tasks.

The model is trained on parallel corpora of Arabic and English sentences, enabling it to learn the intricacies of both languages.

### 3.3.2 ARAVoE Model

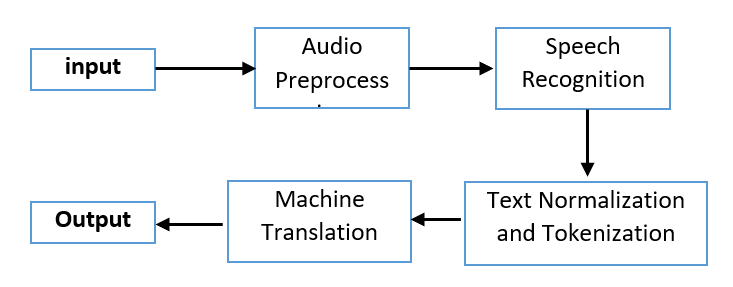
****

Figure # Workflow of the ARAVoE Model

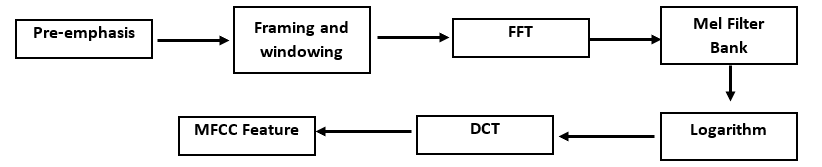
The workflow of the ARAVoE Model begins with the user providing Arabic audio input, which can be captured via a microphone or uploaded as a .wav file. The audio undergoes preprocessing to enhance quality and eliminate background noise, during which the ASR model extracts features using Mel-Frequency Cepstral Coefficients (MFCCs) and normalizes the data for consistency. The preprocessed audio is then fed into the ASR model, which produces the corresponding Arabic text transcription, serving as the crucial source text for the subsequent translation process. Following this, the Arabic text is normalized to remove diacritics and unnecessary symbols, and tokenized into words or subwords in preparation for translation. The normalized and tokenized text is input into the MT model, which translates the Arabic text into English, generating a coherent and contextually relevant output. Finally, the ARAVoE Model displays the translated English text to the user, facilitating a better understanding of the original Arabic speech.

### **3.3.3 Model Training**

#### 3.3.3.1 ****Training the ASR Model****

##### 3.3.3.1.1 Mel-Frequency Cepstral Coefficients Extraction

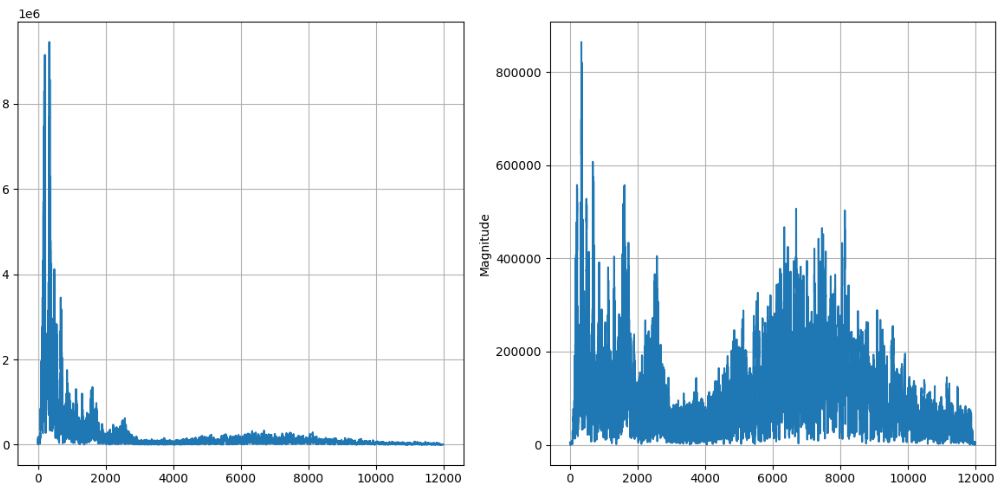
The extraction of Mel-Frequency Cepstral Coefficients (MFCCs) from audio data follows a systematic process. First, the audio files will be loaded using appropriate audio processing libraries, ensuring consistent lengths across all audio signals. The MFCC algorithm will then be applied, considering parameters such as the number of coefficients, FFT window size, and hop length between frames. This extraction will be conducted for each audio file in the dataset. The resulting MFCCs, which capture essential spectral features, will be stored for further analysis or machine learning applications, providing a detailed representation of the audio data.



*Figure 5. MFCC feature extraction Diagram*

Figure 5 shows the MFCC feature extraction method that takes speech and extracts speaker-specific characteristics.

##### 3.3.3.1.2 Pre-emphasis

To flatten the voice signal spectrum, the speech signal is pre-emphasized by approximately 20 dB per decade. This pre-emphasis filter enhances the efficiency of spectral analysis by counteracting the negative spectral slope typically present in spoken signals. By boosting the higher frequencies, this process ensures a more uniform spectral representation, facilitating more accurate analysis in subsequent processing stages. 

*Figure 6. Spectrum before and after Pre-emphasis*

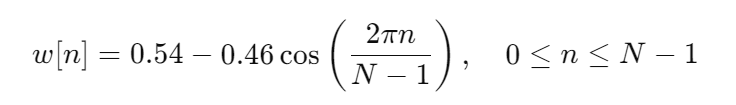
Figure 6 shows the effect of pre-emphasis on the spectrum of a speech signal with the value of 0.97 for a.

##### 3.3.3.1.3 Framing

The speech signal is believed to remain stationary for brief intervals of 20–30 ms. For our study, the input voice signal is thus split into short 25-ms frames that overlap by 15 ms.

##### 3.3.3.1.4 Windowing

To reduce the effects of using a finite-sized segment for feature extraction, the edges of each frame must be tapered to create a window. A Hamming window w[n] is defined by the equation:



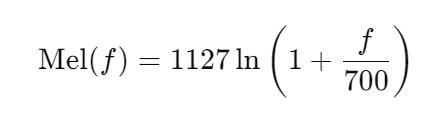
where *N* represents the window size. To extract the signal value at time *n*, the original signal *s[n]* is multiplied by the window function *w[n]*, as expressed by the equation:



##### 3.3.3.1.5 Fast Fourier Transform

The Fast Fourier Transform (FFT) is an efficient algorithm for computing the Discrete Fourier Transform (DFT). It is applied to each frame to obtain the frequency spectrum, commonly referred to as the Short-Time Fourier Transform (STFT). Following this, we calculate the power spectrum, also known as the periodogram.

##### 3.3.3.1.6 Mel Filter Bank

The next step involves applying triangular filters on a Mel scale to extract frequency bands from the power spectrum. The Mel scale is a nonlinear frequency scale, and the following equation calculates the Mel frequency from the frequency in hertz: 

The Mel scale is designed to be more discriminative at lower frequencies and less so at higher frequencies, mirroring the nonlinear way in which human ears perceive sound. Each filter in the filter bank is a triangular filter, characterized by a magnitude of 1 at the center frequency and a linear decrease to 0 as it approaches the center frequencies of the adjacent filters. Frequency domain filtering is performed by multiplying the power spectrum of the signal element-wise with the frequency response of the filters. In our study, we utilize 26 overlapping filter banks.

##### 3.3.3.1.7 Logarithm

The logarithm of the magnitude spectrum must be computed, as humans perceive sound on a logarithmic rather than a linear scale.

##### 3.3.3.1.8 Discrete Cosine Transform

Since the filter banks overlap, the energies from the filter banks are correlated. To decorrelate these energies, we apply the Discrete Cosine Transform (DCT). This process yields cepstral values ranging from the second to the fourteenth coefficient, with the 13th coefficient providing important information about the characteristics of the vocal tract.

##### 3.3.3.1.9 Data Splitting

In our data-splitting process, we utilize a straightforward 80-20 split with the train\_test\_split function. The first step involves randomly shuffling the dataset, with 80% allocated for training features and labels, while the remaining 20% is reserved for testing. Within the training set, we perform an additional split to create a validation set, which is crucial for fine-tuning our machine learning model. To align with the model's input requirements, we also introduce an extra axis to the features. This well-organized split of the dataset facilitates effective learning from 80% of the data and ensures robust evaluation with the remaining 20%.

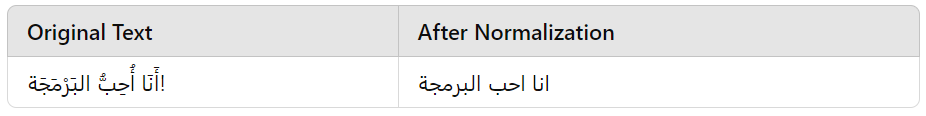
#### 3.3.3.2 MT Preprocess

The preprocessing of text for machine translation (MT) is a crucial step that ensures consistency, clarity, and proper alignment between the source language (Arabic) and the target language (English). Each phase of preprocessing is designed to standardize the data, improve model efficiency, and enhance translation quality. The following sections outline the steps involved in preparing the text for the machine translation pipeline.

##### 3.3.3.2.1 Text Normalization

Arabic text preprocessing requires specific steps to deal with the unique characteristics of the language. Unlike English, Arabic does not have upper or lowercase letters, so lowercasing is irrelevant. The main focus of text normalization is to clean and simplify the text while retaining its meaning. This process involves the removal of diacritics (Tashkeel), which are vowel markings that are typically omitted in written Arabic except in educational texts or the Quran. Diacritics, such as Fatha (ـَ), Damma (ـُ), and Kasra (ـِ), are removed to reduce complexity and enhance the consistency of the text

Table 1: Example of Arabic Text Normalization



as shown in Table 1. Additionally, any special characters, such as Tatweel (ـ) used for elongation, or excess punctuation, are removed. Normalization also includes unifying various forms of the Hamza (ء, أ, إ) to a standard form.

##### 3.3.3.2.2 Tokenization

Tokenization breaks down the text into individual units, known as tokens. For Arabic, tokenization is more complex due to its rich morphology and script. Specialized tokenizers such as Farasa or Moses Tokenizer are employed to split Arabic text into meaningful tokens. For instance, the Arabic sentence "الكتاب كبير" (The book is big) might be tokenized into ["ال", "كتاب", "كبير"]. On the other hand, English tokenization is handled using tools like spaCy or NLTK, which split words and contractions into individual tokens. For example, the phrase "I'm learning" becomes ["I", "'m", "learning"]. This step is critical for both the Arabic and English text, as it standardizes the format for subsequent processing.

##### 3.3.3.2.3 Subword Tokenization

Due to the vast number of possible words in both Arabic and English, rare words can cause issues in translation. To address this, subword tokenization methods like Byte Pair Encoding (BPE) or WordPiece are used. These methods split words into smaller subword units, which helps the model handle out-of-vocabulary words. For example, the English word "playing" might be split into ["play", "ing"], while the Arabic word "كتابي" (my book) can be split into ["كتاب", "ي"] (book, my). Subword tokenization allows the model to generalize better by learning these smaller components instead of entire words.

##### 3.3.3.2.4 Padding and Truncation

Since machine translation models require fixed-length sequences, it is necessary to either pad or truncate the sentences to ensure consistent input length. Sentences that are too short are padded with a special <pad> token until they reach the required length, while sentences that are too long are truncated. Padding ensures that all input sequences are of the same length, which is crucial for batch processing during model training. Truncation, on the other hand, ensures that sentences exceeding the maximum length do not overload the system, making the model more efficient.

##### 3.3.3.2.5 Sentence Pair Alignment

The core of machine translation lies in aligning source sentences (Arabic) with their corresponding target sentences (English). Ensuring this alignment is vital for training the model to accurately map one language to another. Any sentence that does not have a corresponding translation or is mismatched must be excluded from the dataset. For example, the Arabic sentence "أنا أحب البرمجة" should be paired with the English translation "I love programming". Proper alignment ensures that the model can learn from valid pairs, improving its ability to translate correctly.

##### 3.3.3.2.6 Word Embeddings

After tokenization, the words (or subwords) are transformed into numeric vectors through word embeddings. Pre-trained embeddings capture the semantic relationships between words and are used to initialize the model. In this case, Arabic word embeddings like AraVec or FastText can be employed for the Arabic sentences.