Hilispeech: A Hiligaynon Speech Recognition System

A Special Problem

Presented to

the Faculty of the Division of Physical Sciences and Mathematics

College of Arts and Sciences

University of the Philippines Visayas

Miag-ao, Iloilo

In Partial Fulfillment

of the Requirements for the Degree of

Bachelor of Science in Computer Science by

GAVIETA, Don Michael

HONEYMAN, John

SAMSON, Aron Miles

Francis DIMZON

Adviser

Perry Neil FERNANDEZ

Co-Adviser

June 4, 2022

Approval Sheet

The Division of Physical Sciences and Mathematics, College of Arts and Sciences, University of the Philippines Visayas

certifies that this is the approved version of the following special problem:

Hilispeech: A Hiligaynon Speech Recognition System

Approved by:

Name	**Signature**	**Date**
Francis D. Dimzon (Adviser)		June 4, 2022
Perry Neil J. Fernandez (Co-Adviser)		June 4, 2022
Arnel L. Tampos (Division Chair)		June 10, 2022

Division of Physical Sciences and Mathematics

College of Arts and Sciences

University of the Philippines Visayas

Declaration

We, GAVIETA, HONEYMAN, and SAMSON, hereby certify that this Special Problem, including the pdf file, has been written by me/us and is the record of work carried out by me/us. Any significant borrowings have been properly acknowledged and referred.

Name	Signature	Date
Don Michael Y. Gavieta (Student)		June 4, 2022
John A. Honeyman (Student)		June 4, 2022
Aron Miles B. Samson (Student)		June 4, 2022

Dedication

We in Team HiliSpeech dedicate our Special Problem to our beloved family and friends who have supported us in the making of the Hiligaynon speech recognition system. We also dedicate our work to our adviser Prof Francis Dimzon for introducing the concept of speech recognition which gave us an idea on how to apply this to our own native language.

Acknowledgment

In the course of conducting this study, many people made genuine and significant sacrifices for which they need to be gratefully acknowledged.

First, we would like to express our deepest gratitude to our adviser, Francis Dimzon, who guided us in our research. His vast knowledge of our studies, motivation, patience, and expertise helped us overcome the obstacles and hardships we have endured in our studies. This study might not have been feasible without him.

We would also like to thank our co-adviser, Perry Neil Fernandez, who provided us with vital feedback on our documentation by utilizing his knowledge and competence in research.

Lastly, we wish to extend our special thanks to the speakers who took part in our data collection for their time and patience in uttering the several words in our local dictionary. Without them, the completion of our system in our study would be far from beyond being functional and successful.

Abstract

In this paper, the researchers developed a Hiligaynon speech recognition system for recognizing a set of commonly used Hiligaynon words in households and for directions. The Kaldi ASR toolkit was the foundation of the Hiligaynon ASR system. The researchers obtained audio recordings from consented participants for data training and testing. All of the participants are fluent in the language. Each audio file comprises ten words, and each participant was tasked with producing up to 24 audio files. The audio recordings were then cleaned and processed before being used as training and testing data. "Audacity" was used to format the audio recordings. The creation of meta-data for each speaker is a must before training. Sets of multiple acoustic training were employed to train distinct collections of data audio. Monophone, delta-based triphone, delta + delta-delta triphone, LDA + MLLT, SAT, and DNN training were employed for the training models. 5-fold cross-validation was implemented to get the result of each model. Preliminary results showed that while comparing the six training models (Monophone, Triphones(delta and delta + delta-delta), LDA + MLLT, SAT, and DNN), the DNN model had the folds with the lowest word-error rate.

Keywords: speech recognition, acoustic model, training data, phonemes, WER(word error rate).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of the Current State of Technology

The advances of technology today have transformed our way of interacting with our devices. Speech recognition, also known as automatic speech recognition (ASR), is a field of speech science and an efficient tool that enables devices to respond to spoken commands, bridging the gaps between humans and machines through speech and text. It has made hands-free technology possible, building a system that can aid the physically challenged (Gaikwad, Bharti, & Yannawar, 2010). The drawback of this advancement is that its speech input can be limited to frequently spoken languages, which can be a disadvantage to the minority.

The lack of language selections opens for an opportunity for some developers who do not have their language in the options to create and develop systems that could recognize their language and that is where ASR toolkits become handy.

ASR has toolkits that intend to build speech recognition or speech research. An example of this is the Kaldi ASR toolkit. Kaldi is an open-source ASR toolkit written in C++ that is purposely designed to be a modernized and flexible code (Povey et al., 2011). In other words, the toolkit can be easily modified and extended upon by users.

1.2 Problem Statement

The Philippines has a variety of spoken languages, such as Filipino, Cebuano, Ilocano, and Hiligaynon (Casperson, 2010). As mentioned earlier, this creates an opportunity for developers to create speech recognition systems for such languages. Studies have shown that Filipino is one of most commonly used languages in the Philippines for speech recognition. For instance, Dioses' system (2020) detects command words uttered by the user, such as "isa", "ikalawa", "ikatlo", and "patayin", and uses it to control the speed of the fan. Dimzon and Pascual (2020) also created a speech recognizer that analyzes and assesses the Filipino oral reading fluency. For the Hiligaynon language, there are ASR systems that have been developed (Billones & Dadios, 2014). However, with the amount of Hiligaynon language-based ASR systems, developing an effective one has yet to come. With that, having an ASR system for the Hiligaynon language that detects common and command words is yet to be developed.

1.3 Research Objectives

1.3.1 General Objective

This project aims to develop an automatic speech recognition (ASR) system that could recognize a speech input of Hiligaynon words using Kaldi ASR toolkit.

1.3.2 Specific Objectives

Specifically, the goal of the project is to:

1. To develop an ASR system that transcribes the uttered Hiligaynon common and command words of the users;
2. To test and train different speech input of different speakers;
3. To provide an evaluation of the word error rate (WER) and sentence error rate (SER) of each training model;
4. To provide the best training model with the highest speech recognition accuracy.

1.4 Scope and Limitations of the Research

HiliSpeech focused on the Hiligaynon language. The system has a limited Hiligaynon language-based dictionary. Thus, the words used were commonly uttered words (e.g. “huo”, “wala”, “diri”, etc) and basic command words (e.g. “sugod”, “pindot”, “hulat”, etc). Any words outside the dictionary were not included nor recognizable. Furthermore, HiliSpeech only tested and trained speech signals gathered from the participants who are familiar or fluent in this language. In terms of toolkits, the system used the Kaldi ASR toolkit. With that, only the available features, such as language models, training models, and script recipes, in the said toolkit were utilized.

1.5 Significance of the Research

The system is beneficial to the development of the Hiligaynon speech recognition language. One of the main goals of the system was to create a foundation for future researchers to build on if they want to continue developing the Hiligaynon language-based ASR by expanding the system’s dictionary. Furthermore, the development of this system enables other researchers and developers to utilize it in building their software or system.

Chapter 2

Review of Related Literature

2.1 Filipino Speech Recognition

Dimzon and Pascual (2020) utilized HMMs to develop a phoneme-level speech recognizer. Their system used and 60 percent of phoneme positions were within 20 milliseconds of human transcriptions. Using a 5-state model yielded an accuracy rating of 57.47%. This then increased by 10% when the amount of Gaussian mixtures was increased from one to six. Furthermore, incorporating a track length normalization increased its accuracy by 2%. Dimzon and Pascual then stated that to further improve its accuracy, it is recommended to increase the number of HMM's active states and modify the amount of Gaussian mixtures for the training data set. They also recommended integrating a language model, and implementing hybrid Deep Neural Network (DNN)-HMM models.

Brucal et al. (2021) conducted a study on creating a speech-to-text recognition

(STR)system that has a wider range of vocabulary and a higher accuracy rate of Filipino words. Brucal incorporated the convolutional neural network (CNN) in Python language for the system's training and testing stage of the audio files. This feature uses Low Pass Filter, Librosa MFCC Feature Extraction, and Keras Modeling to improve the system's accuracy. Thus, resulted in a percentage of 66.17 for male speakers, 81.64 for female speakers, 38.43 for tests with background noise, and 54.14 for 1kHz monotone.

Dioses (2020) developed a system that automatically controls the electric fan speed with voice control using Filipino language. This was done with the help of a speech recognition system and the use of an Arduino and a smartphone. The words that were used for testing and training are the fan speed and off command - "isa", "ikalawa", "ikatlo", and "patayin". Results showed that the voice command "ikalawa" and "ikatlo" showed the highest accuracy rate of 100%, while "isa" and "patayin" scored 50% and 60%, respectively.

Filipino ASR's are included in this study because they show similarities in terms of phonology and words (Casperson, 2010) . Casperson's research provided evidence showing the similarities of Hiligaynon to other neighbouring languages. He compared the International Phonetic Alphabet (IPA) of different English words translated into the following languages: Hiligaynon, Cebuano, Tagalog, Waray-Waray, and Ilocano. The words he used were "eye", "egg", and "small". His results showed that for "eye", their IPA's are similar. For "egg", Waray-Waray differed from the rest. Lastly, all the languages differed from each other when comparing the IPA for "small". His results concluded that most words in different languages have similarities, especially in Filipino and Hiligaynon. For instance, the Hiligaynon and Filipino word and IPA for "egg" is "itlog", and "eye" is "matah".

2.2 Hiligaynon Speech Recognition

A study by Billones and Dadio (2014) focused on creating a system that uses a 5-word vocabulary Hiligaynon speech recognition to help spread awareness of breast cancer among the local female population of Western Visayas.The researchers used 5 different Hiligaynon words commonly used as directions as the 5 target chromosomes which are “idalom”, “ibabaw”, “wala”, “tuo”, “patiyog”. To symbolize the 200 chromosomes, they recorded 40 audio samples for each word, totalling 200 audio samples. The system uses Mel frequency cepstrum coefficients (MFCC)’s feature extraction and genetic algorithm’s pattern recognition. Furthermore, the system incorporates an adaptive database which helps in increasing the accuracy of the training and classification of the Hiligaynon words. Combining these models alongside the adaptive database resulted in an accuracy of 97.50% in recognizing the different Hiligaynon words.

2.3 Multilingual Speech to Text Recognition System

The study of Shadiev et al. (2017) determined if multilingual communications in cross-cultural learning can be supported with the aid of a STR and computer-aided translation (CAT) system. STR was used to generate texts from their participants’ voice inputs, while CAT translated the obtained STR-texts to English. For the STR portion, the lowest accuracy rating was for Mongolian (94.37%) and Filipino language (94.60%), while the highest was Spanish (98.15%), Russian (98.02%), and French (97.95%).

2.4 Speech Recognition Systems utilizing Kaldi

Naeem et al. (2020) created an ASR system that utilizes the features of the Kaldi toolkit, specifically Subspace Gaussian Mixture Model (SGMM). The study's purpose was to improve on the current architecture of the statistical automatic speech recognition system for Urdu, an under-resourced language in South Asia. Their system underwent five phases: Monophone Model, Tri1-Model, Tri2-Model, Tri3-Model, and SGMM2. After refining the models, results showed that the Monophone Model has the worst minimum WER (33.24), followed by the three triphone models (16.70, 16.74, and 13.63, respectively), and lastly SGMM2 with the best minimum WER (9.64). Naeem et al. also tested different approaches on common voice datasets containing more than 1000 hours of speech along both sexes with various age groups. These approaches their resulted WER are: Sphinx (39.82), Kaldi monophone (52.06), Kaldi triphone (with delta and double delta) (25.06), Kaldi LDA + MLLT (21.69), Kaldi LDA + MLLT + SAT with 20,000 samples (22.25), Kaldi LDA + MLLT + SAT with a complete dataset(17.85), Kaldi TDNN (4.82). Among these approaches, Kaldi Time Delay Neural Network has the best WER of 4.82%. As for their recommendations, the researchers suggested that in future studies, testing different n-values for the language models and combining them with DNNs may increase the system's WER.

Upadhyaya et al. (2017) and their system focused on creating a continuous Hindi ASR Model based on Kaldi. Their AMUAV database consisted of 100 speakers, each uttering 10 random short Hindi sentences, but two sentences are common to each speaker. With a total of 1000 audio files, 90% are put in the training, while the remaining 10% were for testing. The researchers then followed the Metadata preparation steps mandatory for the Kaldi toolkit. The results showed that after comparing the performance between the MFCC and PLP features by using the monophone training model with a 2-gram, 3-gram, 4-gram language model, MFCC ranked better than PLP. The same result goes for testing the performance of the two features by using the triphone training model with the same n-gram language models. For the monophone model, the 3-gram had the best recognition rate among the three n-gram models. Increasing it to 4-gram decreased its rating. For the triphone model, on the other hand, the 2-gram language model had the best recognition rate. Increasing its n-value, however, decreased its performance. After their research, they recommended that implementing DNN improves the performance of their speech recognition system.

2.5 HMM-based Speech Recognition

Bautista and Kim (2014) developed a Filipino speech recognition system using HTK System tools. Phonetic Hidden-Markov Model (HMM) based acoustic models were applied in their study to measure the parameters of the Filipino speech corpus that was used in both the training and testing section of their system. This speech corpus was developed by the Digital Signal Processing Laboratory (DSP Lab) of the University of the Philippines-Diliman. The results showed an average accuracy rate of 80.13 for a single-Gaussian mixture model. When a phoneme-alignment was implemented, the result increased to 81.13. An increased

Gaussian-mixture weight model gave a score of 87.29. The highest result came from a 5-state model with six Gaussian mixtures, with a score of 88.70%. Pascual et al. (2017) developed a language learning system that utilized an HMM-based ASR system. The speech corpus contained 248 voice-recorded phrases that were assessed as regularly employed for survival. Further, the corpus were created using phoneme-level transcriptions of 10 native Ilokano speakers in their middle years (ages 18-28). The results revealed that the training data was suitable for a system that was not dependent on the user, having a miscue detection error rate of 34.25% (equivalent to a detection rate of 65.75% at a false alarm rate of 25.16%) using the Reading Miscue Detector (RMD). With a total of 2,480 speech recordings, Malaay then recommended to have a larger scale of speech corpus that is more gender and phonetically balanced to have a higher detection rate and lower false alarm rate in such systems.

2.6 DNN-based Speech Recognition

Kipyatkova and Karpov's study (2016) focused on deep neural network (DNN) based acoustic models of the speech recognition systems for the Russian language. Kaldi ASR toolkit was used in their program for training and testing their ASR system. With Kaldi, they were able to implement the DNN feature offered by Kaldi. The researchers initially started their experiments with GMM-HMM acoustic models, followed by experimenting on Russian ASR using the DNN-based acoustic models. With the tanh function on their DNN, they have three to five hidden layers, each layer having 1024-2048 units. There was only a slight improvement in performance from the GMM-HMM AMs to the DNN-based AMs.

In their final experiment, the p-norm output dimension and p-norm input dimension parameters were used. They tested p-norm input and output dimensions of 2000/200 and 4000/400 respectively, showing slightly better results compared to the second experiment. Overall, the best WER that they achieved was from the p-norm DNN with six hidden layers and an input/output dimension of 2000/200, with a score of 20.30%.

Yin et al.(2015) conducted experimental research for deep neural network (DNN) based speech recognition using the Kaldi toolkit. They used noisy training to test the DNN's adaptability to noise in speech recognition. Noisy training, also known as noise injection, consists of two simple steps. The first step was to collect real-world noise signals from locations such as parks, buses, train stations, cafeterias, and restaurants. The next step was combining the noise signals with the original clean training data to produce "corrupted" audio. The researchers concluded that by introducing different levels of noise into the training data, noise signal patterns can be learned and can provide significant performance improvements for DNN-based speech recognition.

The research of Dahl et al. (2012) used a hybrid DNN-HMM architecture and yielded a result of a sentence accuracy improvement of 5.8% and 9.2%, which outperformed GMM-HMMs.

Seide, Li, and Yu's study (2011) also used DNN-HMMs together with other HMMs such as ANN-based HMMs combined with tied-state triphones and deep-belief network pre-training. With the application of DNN-HMMs, the results showed the error rate reduced the score of a standard GMM-HMM of 27.4% to 18.5%. Their results then concluded that standard GMM-HMM underperforms compared to DNN-based models.

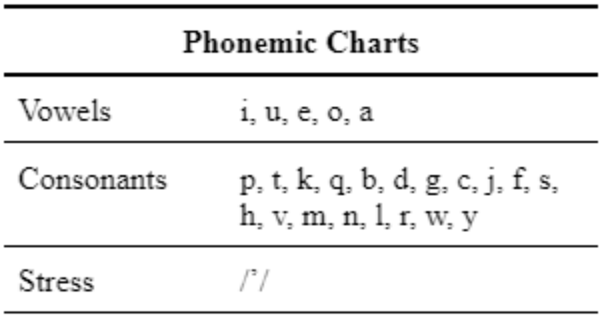
Maas et al. (2017) decided on using DNN-based models in their study as he quoted that DNN models are "a central component of nearly all state-of-the-art speech recognition systems". He described GMMs as complex models which yield unsatisfactory results. Hence, a DNN-based model was used in place of GMM.

2.7 Hiligaynon Phonology

The Hiligaynon Language, commonly referred to as Ilonggo, is an Austronesian language (Michel, Hangya, & Fraser, 2020) and it is the lingua franca of most of the portions in Western Visayas, especially in the islands of Negros, Panay, and Romblon (Wolfenden, 2019). The Hiligaynon Language is the 5th most spoken language in the Philippines, following Ilocano, Cebuano, Tagalog, and Filipino (Casperson, 2010). Hiligaynon is most commonly used in the provinces of Negros Occidental, Iloilo, and Capiz (Robles, 2012).

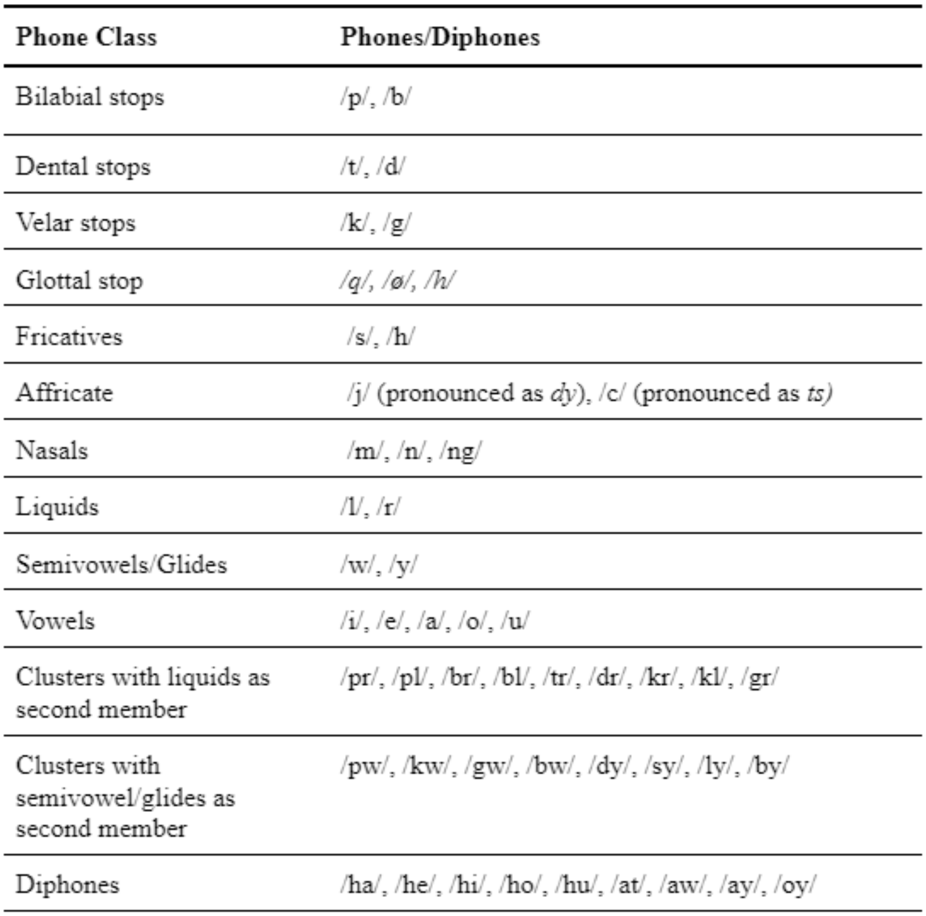
The Hiligaynon alphabet was derived from the Tagalog spelling system (Ager, n.d.). The pronunciation of most of the consonants is similar to the phonetic value of how they are pronounced in English. Vowel sounds, however, can differ from English vowels. The vowels a, i, and u are Hiligaynon letters native to the language, while e and o are adopted from English and Spanish (Motus, 2019).

The Hiligaynon has a total of 25 letters in the alphabet, with five vowels and twenty consonants (Wolfenden, 2019). The following are the Phonemic Charts of the Hiligaynon Language:



Phonemic Charts	
Vowels	i, u, e, o, a
Consonants	p, t, k, q, b, d, g, c, j, f, s, h, v, m, n, l, r, w, y
Stress	/ˈ/

The following are the phonological symbols of the Hiligaynon language's alphabet (Wolfenden, 2019):



Phone Class	**Phones/Diphones**
Bilabial stops	/p/, /b/
Dental stops	/t/, /d/
Velar stops	/k/, /g/
Glottal stop	*/q/, /ø/, /h/*
Fricatives	/s/, /h/
Affricate	/j/ (pronounced as *dy*), /c/ (pronounced as *ts)*
Nasals	/m/, /n/, /ng/
Liquids	/l/, /r/
Semivowels/Glides	/w/, /y/
Vowels	/i/, /e/, /a/, /o/, /u/
Clusters with liquids as second member	/pr/, /pl/, /br/, /bl/, /tr/, /dr/, /kr/, /kl/, /gr/
Clusters with semivowel/glides as second member	/pw/, /kw/, /gw/, /bw/, /dy/, /sy/, /ly/, /by/
Diphones	/ha/, /he/, /hi/, /ho/, /hu/, /at/, /aw/, /ay/, /oy/

Chapter 3

Research Methodology

As indicated in the title, this chapter described the research methodology. In more detail, in this part, the author outlined data gathering techniques and data sets, implementations of training models, the toolkit, and the specifications employed.

3.1 Standard Speech Recognition System

Speech recognition is the process of using various algorithms implemented as a computer program that will automatically recognize a speech signal and convert it into a sequence of words (Gaikwad et al., 2010). These algorithms included statistical pattern recognition, communication theory, signal processing, linguistics, and many more, which were important to analyze and process the given speech signals (Shrawankar & Mahajan, 2013). A recording device, such as a microphone or a telephone, is required to collect speech or acoustic signals. According to S and Chandra (2016), the acoustic front-end, lexicon, language models, acoustic model, and decoder are the five major components of a standard speech recognition system as shown in Figure 3.1.

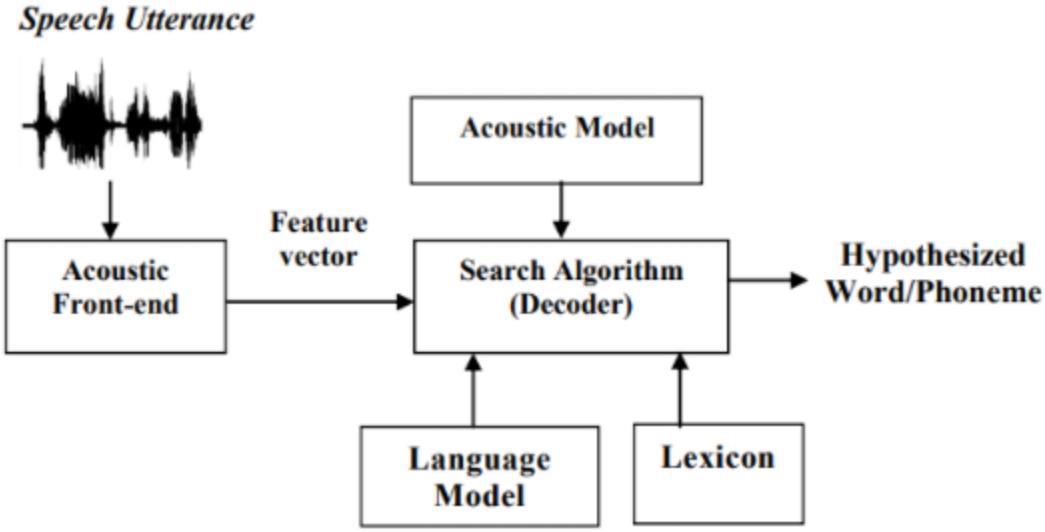


Figure 3.1: Chandra's Standard Speech Recognition System

The acoustic front-end functioned as a feature extractor where it converts the speech / acoustic signal into a sequence of small fixed-size acoustic vectors that is still reliable and effective for recognition (Ayaz & Shaukat, 2021; S & Chandra, 2016). A large number of class samples (acoustic vectors of training data) were used to estimate the parameters of the classification model (Gaikwad et al., 2010; S & Chandra, 2016). The decoder worked as a recognition or matching process between the test and the trained data. It searched for all the possible word sequences that match the test data. An acoustic model was used to calculate the probability and be determined by a language model (S & Chandra, 2016).

3.2 Different ASR toolkits

3.2.1 Kaldi

Kaldi is an open-source ASR toolkit written in C++ that is purposely designed to be a modernized and flexible code (Povey et al., 2011). In other words, the toolkit can be easily modified and extended by users. The process in Kaldi for training and testing uses deep neural networks for acoustic modeling and Gaussian mixture models. The tools used for decoding were the weighted finite-state transducers. Linear and affine transformations were used in modeling arbitrary phonetic-context sizes, acoustic modeling, and Gaussian mixture models (Sahu & Ganesh, 2015).

3.2.2 Julius

Julius is written in C and has been widely used in both academic research and industrial applications. This toolkit can run a 60k-word dictation task on PCs with low specifications and a small footprint (Piero, 2020). It is also capable of managing different language models such as the N-gram model and Hidden Markov Model (HMM) as an acoustic model.

3.2.3 Hidden Markov Model Toolkit (HTK)

HTK, similar to Julius, is another portable toolkit written in C. This tool can build on HMM for training, testing, and analysis of results. It has been applied in different fields such as speech synthesis, character recognition, and DNA sequencing (Young et al., 2015). HTK also has scripts that can be useful for acoustic modeling (Sahu & Ganesh, 2015) and this can be modified to be used on other ASR software.

3.3 Kaldi Toolkit

The Kaldi ASR toolkit is the toolkit used in this study. Other toolkits were put into consideration, such as Julius, and Hidden Markov Model Toolkit (HTK). The researchers decided to incorporate Kaldi into their system due to its popularity, features, and ease of use (Povey et al., 2011). The Kaldi toolkit is a piece of open-source software for speech recognition. This toolkit's code was developed in C++ and released under the Apache License v2.0 (Povey et al., 2011). The benefit of using Kaldi to build a speech recognition application is that it generates high-quality lattices that are fast enough for real-time recognition. (Povey et al., 2011). The internal structure of the Kaldi ASR toolkit is shown in Figure 3.2.

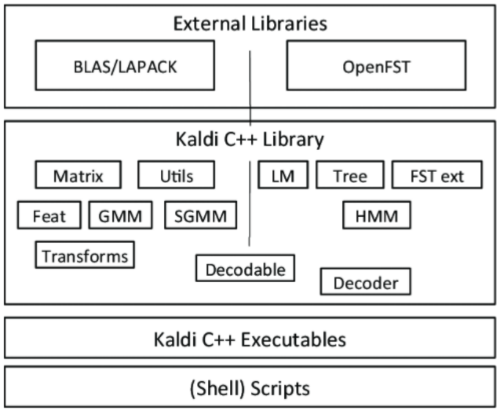


Figure 3.2: Internal structure of Kaldi ASR toolkit

3.4 Data Collection

For this study, audio files of the participants were recorded. The researchers utilized recording software on a variety of devices, including smartphones and computers, to retrieve audio files from the participants. To reduce background noise, the recording took place in a secluded room.

The chosen participants were either Hiligaynon native speakers or people who can speak Hiligaynon fluently. In total, there were ten (10) speakers, five (5) males, and five(5) females), ranging from the age of 18 to 65 years old. The researchers performed a brief survey, asking a few members of their individual families for a list of words they used commonly. These words were picked at random and are often used on a regular basis. The researchers compiled a list of 240 widely used Hiligaynon words. These words were compiled utilizing a Hiligaynon dictionary and Hiligaynon lessons based on (Naeem et al., 2020). These words were then chosen at random and combined to form a collection of 10 words for each utterance. The researchers used a variety of hardware, including headsets and microphones, to record the participants' voices. Each audio file contains ten words chosen at random from the collection of utterance, and each participant was tasked with creating each audio file, for a total of 24 audio files, each with a ten-word utterance.

3.4.1 Data phonemes

One important key component for the Kaldi to function is its various phonemes. These phonemes were paired with words from the system's local dictionary. The list of phonemes used in the study were shown in Table 3.1.

Table 3.1: Utilized data phonemes

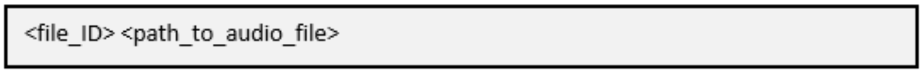
Phone Class	Phones/Diphone
Bilabial stops	/p/, /b/
Dental stops	/t/, /d/
Velar stops	/k/, /g/
Africate	/j/
Fricatives	/s/, /sh/, /v/, /z/, /f/
Nasals	/m/, /n/, /ng/
Liquids	/l/, /r/
Semivowels/Glides	/w/, /y/
Vowels	/i/, /e/, /a/, /o/, /u/
Diphones	/ha/, /he/, /hi/, /ho/, /hu/, /at/, /aw/, /ay/, /oy/

3.5 Preprocessing

The researchers used the software “Audacity” as the tool for cleaning and processing the audio files. Audacity is an easy-to-use, multi-track audio editor and recorder for Windows, macOS, GNU/Linux, and other operating systems. Developed by a group of volunteers as open source (Crook, n.d.). Noises and incomprehensible utterances were removed from the data. This stage fixes any record errors as well as stuttering. The data was sent out on a single channel at a fixed sample rate of 16 kHz. The recordings of each speaker were continuously captured and exported in WAV format.

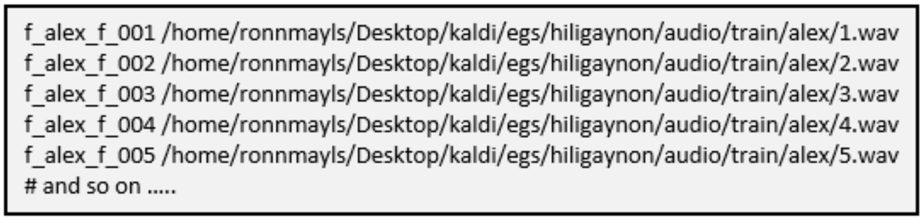
Construction of the meta-data of each speaker is needed before proceeding into acoustic model training and testing. To save time and avoid errors, this meta-data was constructed with the help of a Python script. The following is the meta-data format for acoustic data:

1. **wav.scp:** This file contains the set of path of the recorded audio file together with its file ID (speaker name, gender, id). This file can be described in this format:



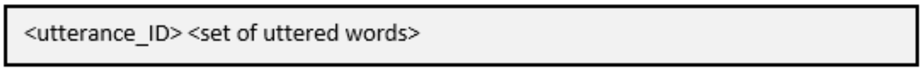
```
<file_ID> <path_to_audio_file>
```

   Example of wav.scp file:



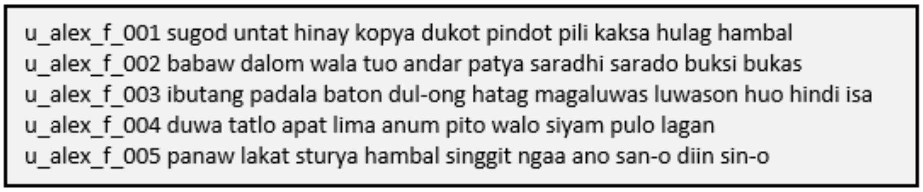
```
f_alex_f_001 /home/ronnmayls/Desktop/kaldi/egs/hiligaynon/audio/train/alex/1.wav
f_alex_f_002 /home/ronnmayls/Desktop/kaldi/egs/hiligaynon/audio/train/alex/2.wav
f_alex_f_003 /home/ronnmayls/Desktop/kaldi/egs/hiligaynon/audio/train/alex/3.wav
f_alex_f_004 /home/ronnmayls/Desktop/kaldi/egs/hiligaynon/audio/train/alex/4.wav
f_alex_f_005 /home/ronnmayls/Desktop/kaldi/egs/hiligaynon/audio/train/alex/5.wav
# and so on .....
```

2. **text:** This file contains an utterance ID that corresponds with its text transcription. This file can be described in this format:



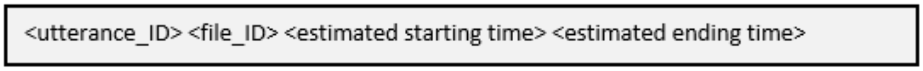
```
<utterance_ID> <set of uttered words>
```

   Example of text file:



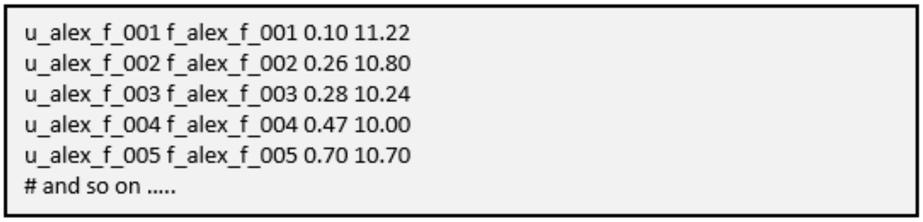
```
u_alex_f_001 sugod untat hinay kopya dukot pindot pili kaksa hulag hambal
u_alex_f_002 babaw dalom wala tuo andar patya saradhi sarado buksi bukas
u_alex_f_003 ibutang padala baton dul-ong hatag magaluwas luwason huo hindi isa
u_alex_f_004 duwa tatlo apat lima anum pito walo siyam pulo lagan
u_alex_f_005 panaw lakat sturya hambal singgit ngaa ano san-o diin sin-o
```

3. **segments:** This file contains the file id, utterance id, estimated starting time of the first word and the ending time of the last word utterance. This file can be described in this format:



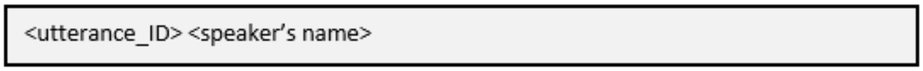
```
<utterance_ID> <file_ID> <estimated starting time> <estimated ending time>
```

Example of segments file:



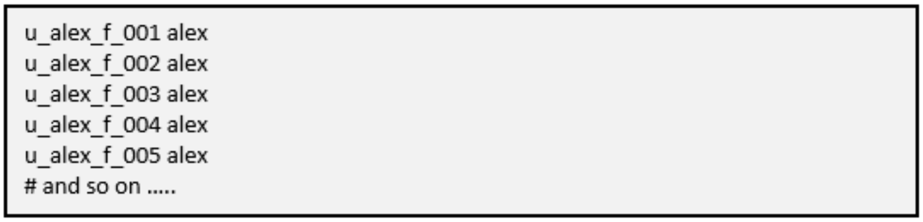
```
u_alex_f_001 f_alex_f_001 0.10 11.22
u_alex_f_002 f_alex_f_002 0.26 10.80
u_alex_f_003 f_alex_f_003 0.28 10.24
u_alex_f_004 f_alex_f_004 0.47 10.00
u_alex_f_005 f_alex_f_005 0.70 10.70
# and so on .....
```

4. **utt2spk:**This file contains the mapping of the utterance and the speaker. This file can be described in this format:



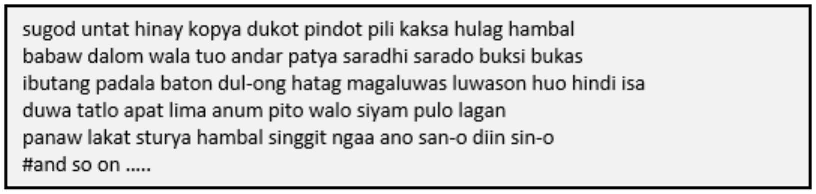
```
<utterance_ID> <speaker's name>
```

Example of utt2spk file:



```
u_alex_f_001 alex
u_alex_f_002 alex
u_alex_f_003 alex
u_alex_f_004 alex
u_alex_f_005 alex
# and so on .....
```

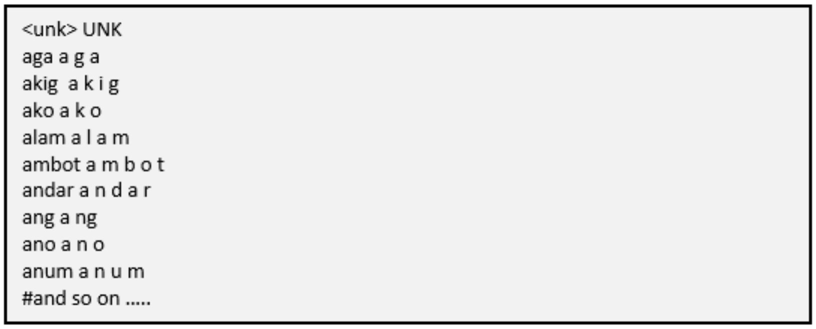
5. **corpus.txt:** This file contains all of the utterance transcriptions utilized in building the model. This is an example of corpus.txt file:



```
sugod untat hinay kopya dukot pindot pili kaksa hulag hambal
babaw dalom wala tuo andar patya saradhi sarado buksi bukas
ibutang padala baton dul-ong hatag magaluwas luwason huo hindi isa
duwa tatlo apat lima anum pito walo siyam pulo lagan
panaw lakat sturya hambal singgit ngaa ano san-o diin sin-o
#and so on .....
```

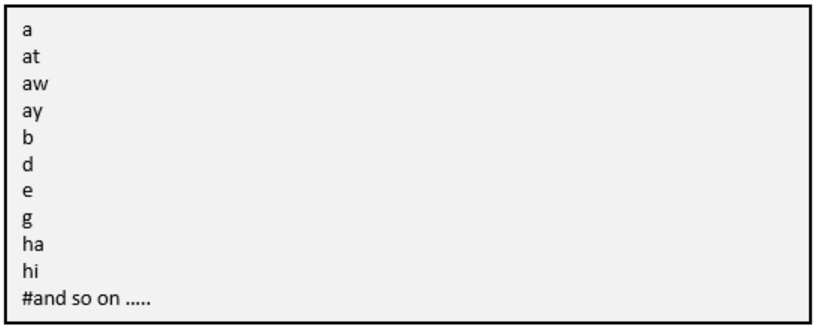
The following is the meta-data format for language data:

6. **lexicon.txt:** This file contains the phone transcriptions of every word. This is an example of lexicon.txt:



```
<unk> UNK
aga a g a
akig  a k i g
ako a k o
alam a l a m
ambot a m b o t
andar a n d a r
ang a ng
ano a n o
anum a n u m
#and so on .....
```

7. **nonsilence_phones.txt:** This contains all the phones that are used in the language. This is an example of nonsilence_phones.txt:



```
a
at
aw
ay
b
d
e
g
ha
hi
#and so on .....
```

8. **silence_phones.txt:** This file contains the silence and short pause phone. This is an example of silence_phones.txt:

```
SIL
UNK
```

3.6 Model Building

3.6.1 Acoustic Model

The acoustic model was in charge of deciding the relationship between acoustic features and phonetic units that must be recognized (Bhatt, Jain, & Dev, 2020). It is a file that holds the statistical representation of each sound that makes up a word, and it's crucial for automatic speech recognition (S & Chandra, 2016). Acoustic models, according to (Mansikkaniemi, 2010), are statistical models that estimate the likelihood of a phoneme being said in a recorded audio segment.

Acoustic modeling is a crucial procedure in voice recognition since it performs the majority of the statistical calculations owing to feature extractions that affect the recognition process. In this process, choosing the best-suited classification methods was needed to be implemented. There are different classification methods to create an acoustic model such as the hidden Markov model (HMM), deep neural networks (DNNs), and sequence to sequence acoustic modeling.

Hidden Markov models (HMMs) and Gaussian mixture models are the most often used approaches in audio modeling (GMMs). These models were employed in statistical parametric techniques that use acoustic feature sequences to produce low-level speech waveforms from high-level inputs. However, these approaches have limitations, which is why some researchers and innovators employ other methods such as deep neural networks (DNNs), which have been effectively used in the development of an automatic speech recognition system. (Ling et al., 2015).

3.6.2 Language Model

The language model is a set of limitations on the sequence of words that can be used in a speech. It analyzes the text in the data to compute the statistical likelihood of the terms.(Gaikwad et al., 2010). The statistics of each word on the list of terms estimated on a training corpus might be the basis for having these limits in place.

Language modeling is an essential component of every speech recognition system because it limits the acoustic analysis, guides the search through numerous candidate word strings, and assesses the acceptability of the speech recognizer's final output.(Chen & Chen, 2011). There are different types of language models, such as the Statistical Languages Model (SLM) and Neural Languages Model (NLM). The SLM is the most commonly used language model in voice recognition systems because it incorporates several statistical approaches such as n-grams variants(unigram, bigram, trigram) and Hidden Markov Models (HMM).

The simplest language model is the N-gram, which calculates the likelihood of a sequence of n-words. N-grams can be unigrams (one-word patterns like "hi"), bigrams (two-word like "hi friend"), or trigrams (three-word sequences like "hi best friend"). By using only the last few words of the audio data that were trained, this model can approximate the probability of the word occurrence.

3.7 Training Acoustic Model

The goal of an automatic speech recognition system is to recognize an input signal speech or test data set by comparing and matching it with various trained acoustic model parameters. Different acoustic training algorithms were used to train the different collections of data audio. Examples are monophone, triphone, LDA + MLLT, SAT, and DNN training.

3.7.1 Monophone

The first acoustic model trained was the monophone model. Monophone is considered a building block model for the triphone model, and it doesn't use any of the contextual data of the previous or subsequent phone. (Chodroff, 2015). For example, the word "five" has three phonemes of /f ay v/, its HMM structure for this model is shown in Figure 3.11.

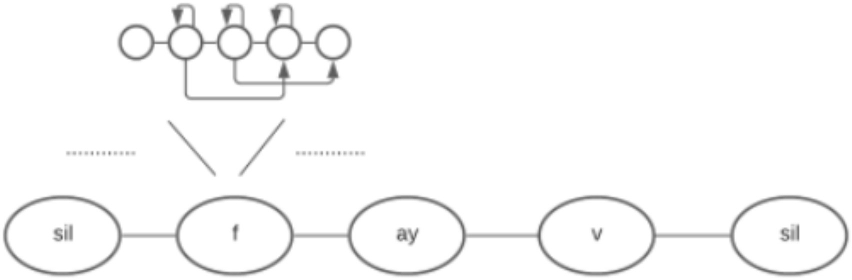


Figure 3.3: HMM Structure of a Monophone Model

3.7.2 Triphone

In contrast to the monophone model, the triphone model considers the contextual data of the preceding and following phones, allowing it to function better. This model has a drawback in that it is larger than the monophone, making it more complicated and time-consuming to construct. . The triphone model's HMM structure is depicted in Figure 3.12, which can be used to compare to the monophone model.

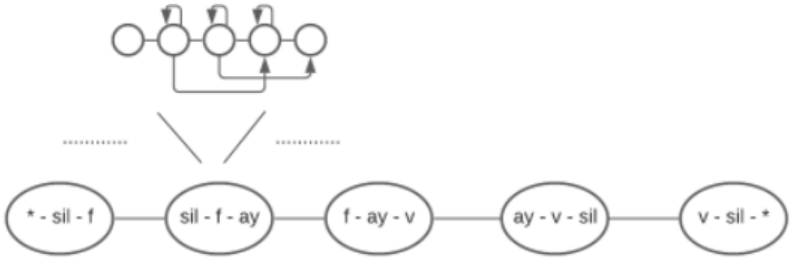


Figure 3.4: HMM Structure of a Triphone Model

A number of algorithms may be used in conjunction with triphone training to increase the performance of the acoustic model. These algorithms include Delta+delta-delta training, LDA-MLLT, SAT

3.7.3 Delta + delta-delta

Delta and double-delta is a training algorithm that computes delta and double-delta features, also known as dynamic coefficients to boost the MFCC features. Delta and delta-delta characteristics are numerical estimates of the signal's first and second-order derivatives (features) (Chodroff, 2015).

3.7.4 LDA + MLLT

The Linear Discriminant Analysis - Maximum Likelihood Linear (LDA-MLLT), is a mix of two algorithms that can be used to refine the first triphone model. LDA is used to reduce the feature space of the data when taking feature vectors and building the HMM states. On the other hand, MLLT uses the reduced feature space of the LDA and normalizes the different speakers by minimizing their differences.(Chodroff, 2015).

3.7.5 SAT

The Speaker Adaptive Training (SAT), just like LDA, also manages the speaker and noise normalization by applying a data transform to each speaker. It is typically used in Hidden-Markov-Model (HMM) voice recognizer that is associated with Gaussian Mixture Models (GMMs) that generates standardized data, which allows the model to focus on predicting variance related to the phoneme rather than the speaker or recording environment.t (Chodroff, 2015; Miao, Zhang, & Metze, 2015).

3.7.6 DNN

The Deep Neural Network (DNN) is a distinct algorithm that can learn speech patterns and is naturally discriminative when trained with an appropriate objective function. It is very adaptable to structures with a large number of parameters that are shared among feature dimensions and targets such as phones or states (Yin et al., 2015). The previous models typically begin with utterance-level transcription. In contrast, the DNN model starts with the labeled frames (phoneme-to-audio alignments) generated by the previous GMM-HMM system.

3.8 Gathering of results

3.8.1 Cross Validation

Cross-validation (CV) is a technique for evaluating and testing the performance of a machine learning model (or accuracy). It entails reserving a specific sample from a data set on which the model has not yet been trained. The model is then tested on this sample to see how well it works (Joby, n.d.).

In this study, the researchers implemented the k-fold cross-validation method. It is a type of cross-validation that guarantees that every observation from the data set has a chance of showing up in the training and test sets. K-fold cross-validation divides that data set into k splits and performs it k times. In this study, the researchers used 5-folds (k = 5), which means the data set (speakers) was divided into five (5) batches, with two (2) speakers serving as test data and the remaining eight (8) serving as training data. The first batch was made up of speaker1 and speaker2 as test data, while the rest were training data. Speaker3 and speaker4 were used as test data in the next batch, while the rest served as training data. This method was repeated until all speakers by pair served as test data. Additionally, k-fold cross-validation was implemented in all acoustic models to gather the average WER and SER results.

Chapter 4

Results and Discussions

This section summarizes and analyzes the various results obtained from decoding the various training models (monophone, triphones (delta, delta + delta-delta), LDA + MLLT, SAT, and DNN). It contains the data result from each decoding model's WER files output, including its WER and SER (word and sentence error rates), the number of insertions, deletions, and substitutions, and the model's average WER and SER percentage.

To draw comparisons, the researchers implemented the 5-fold cross validation to the different training models. In the preliminary results, ten speakers were employed. Two speakers were utilized for testing, while the remaining were used for training. The preliminary findings are as presented in the subsection below.

4.1 Results of Monophone model

Table 4.1 shows the results of the first fold decoding of the monophone model. Word and sentence errors are only found at wer_7. Furthermore, the average WER and SER percentages of this fold are 0.40 and 0.38, respectively.

Table 4.1: Results of first fold decoding of monophone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	0.42	4.17	1	0	1
wer_8	0	0	0	0	0
wer_9	0	0	0	0	0
wer_10	0	0	0	0	0
wer_11	0	0	0	0	0
wer_12	0	0	0	0	0
wer_13	0	0	0	0	0
wer_14	0	0	0	0	0
wer_15	0	0	0	0	0
wer_16	0	0	0	0	0
wer_17	0	0	0	0	0
Mean	0.4	0.38			

Table 4.2 shows the results of the second fold decoding of the monophone model. Word and sentence errors start from wer_7 to wer_17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.53% and 5.30%, respectively.

Table 4.2: Results of second fold decoding of monophone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	1.04	10.42	2	0	3
wer_8	1.04	10.42	2	0	3
wer_9	0.83	8.33	2	0	2
wer_10	0.62	6.25	1	0	2
wer_11	0.62	6.25	1	0	2
wer_12	0.62	6.25	1	0	2
wer_13	0.21	2.08	0	0	1
wer_14	0.21	2.08	0	0	1
wer_15	0.21	2.08	0	0	1
wer_16	0.21	2.08	0	0	1
wer_17	0.21	2.08	0	0	1
Mean	0.53	5.30			

Table 4.3 shows the results of the third fold decoding of the monophone model. Word and sentence errors start from wer_7 to wer_17 following descending order. Furthermore, the average WER and SER percentages of this fold are 3.24% and 21.59%, respectively.

Table 4.3: Results of third fold decoding of monophone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	5.62	31.25	7	5	15
wer_8	4.79	29.17	6	5	12
wer_9	4.58	27.08	6	5	11
wer_10	3.75	22.92	4	5	9
wer_11	3.54	22.92	3	5	9
wer_12	3.33	22.92	3	5	8
wer_13	2.71	18.75	3	5	5
wer_14	2.29	18.75	2	5	4
wer_15	1.67	14.58	1	5	2
wer_16	1.67	14.58	1	5	2
wer_17	1.67	14.58	1	5	2
Mean	3.24	21.59	3.36	5	7.18

Table 4.4 shows the results of the fourth fold decoding of the monophone model. Word and sentence errors start from wer_7 to wer_17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.66% and 6.25%, respectively.

Table 4.4: Results of fourth fold decoding of monophone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	1.46	12.5	2	1	4
wer_8	1.46	12.5	2	1	4
wer_9	1.04	10.42	2	1	2
wer_10	0.83	8.33	1	1	2
wer_11	0.62	6.25	1	1	1
wer_12	0.62	6.25	1	1	1
wer_13	0.42	4.17	1	1	0
wer_14	0.21	2.08	0	1	0
wer_15	0.21	2.08	0	1	0
wer_16	0.21	2.08	0	1	0
wer_17	0.21	2.08	0	1	0
Mean	0.66	6.25	0.91	1	1.27

Table 4.5 shows the results of the fifth fold decoding of the monophone model. Word and sentence errors start from wer_7 to wer_14 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.57% and 5.11%, respectively.

Table 4.5: Results of fifth fold decoding of monophone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.08	16.67	3	0	7
wer_8	1.46	12.5	2	0	5
wer_9	0.62	6.25	0	0	3
wer_10	0.62	6.25	0	0	3
wer_11	0.62	6.25	0	0	3
wer_12	0.42	4.17	0	0	2
wer_13	0.21	2.08	0	0	1
wer_14	0.21	2.08	0	0	0
wer_15	0	0	0	0	0
wer_16	0	0	0	0	0
wer_17	0	0	0	0	0
Mean	0.57	5.11	0.45	0	2.18

Among the five (5) folds, the fold with the highest average WER and SER percentage is the third fold with 3.24 and 21.59, followed by the fourth fold with 0.66 and 6.25, the fifth fold with 0.57 and 5.11, the second fold with 0.53 and 5.30, and the first fold with 0.4 and 0.38 respectively. With this, the monophone model gives a WER percentage ranging from 0.40 to 3.24 and an SER percentage of 0.38% to 21.59%.

4.2 Result of Triphone model

Table 4.6 shows the results of the first fold decoding of the triphone model. Word and sentence errors start from wer_7 to wer_11 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.38% and 5.42%, respectively.

Table 4.6: Results of the first fold decoding of triphone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	1.88	16.67	3	0	6
wer_8	1.04	10.42	1	0	4
wer_9	0.42	4.17	1	0	1
wer_10	0.42	4.17	1	0	1
wer_11	0.42	4.17	1	0	1
wer_12	0	0	0	0	0
wer_13	0	0	0	0	0
wer_14	0	0	0	0	0
wer_15	0	0	0	0	0
wer_16	0	0	0	0	0
wer_17	0	0	0	0	0
Mean	0.38	5.42	0.64	0	1.18

Table 4.7 shows the results of the second fold decoding of the triphone model. Word and sentence errors start from wer_7 to wer_17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.70% and 6.44%, respectively.

Table 4.7: Results of the second fold decoding of triphone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	1.88	14.58	7	0	2
wer_8	1.46	12.5	6	0	1
wer_9	1.04	10.42	4	0	1
wer_10	0.83	8.33	3	0	1
wer_11	0.62	6.25	2	0	1
wer_12	0.62	6.25	2	0	1
wer_13	0.42	4.17	1	0	1
wer_14	0.21	2.08	0	0	1
wer_15	0.21	2.08	0	0	1
wer_16	0.21	2.08	0	0	1
wer_17	0.21	2.08	0	0	1
Mean	0.70	6.44	2.27	0	1.09

Table 4.8 shows the results of the third fold decoding of the triphone model. Word and sentence errors start from wer_7 to wer_17 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.61% and 12.50%, respectively.

Table 4.8: Results of the third fold decoding of triphone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	3.54	20.83	2	4	11
wer_8	2.5	18.75	1	4	7
wer_9	1.67	14.58	0	4	4
wer_10	1.25	10.42	0	4	2
wer_11	1.25	10.42	0	4	2
wer_12	1.25	10.42	0	4	2
wer_13	1.25	10.42	0	4	2
wer_14	1.25	10.42	0	4	2
wer_15	1.25	10.42	0	4	2
wer_16	1.25	10.42	0	4	2
wer_17	1.25	10.42	0	4	2
Mean	1.61	12.50	0.27	4	3.45

Table 4.9 shows the results of the fourth fold decoding of the triphone model. Word and sentence errors start from wer_7 to wer_17 following descending order. Furthermore, the average WER and SER percentages of this fold are 2.92% and 22.92%, respectively.

Table 4.9: Results of the fourth fold decoding of triphone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.92	22.92	8	1	5
wer_8	2.08	18.75	5	1	4
wer_9	1.25	10.42	2	1	3
wer_10	1.25	10.42	2	1	3
wer_11	0.83	6.25	1	1	2
wer_12	0.42	4.17	1	1	0
wer_13	0.42	4.17	1	1	0
wer_14	0.42	4.17	1	1	0
wer_15	0.42	4.17	1	1	0
wer_16	0.21	2.08	0	1	0
wer_17	0.21	2.08	0	1	0
Mean	2.92	22.92	2	1	1.55

Table 4.10 shows the results of the fifth fold decoding of the triphone model. Word and sentence errors start from wer_7 to wer_17 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.10% and 10.23%, respectively.

Table 4.10: Results of the fifth fold decoding of triphone model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.5	22.92	5	0	7
wer_8	1.88	16.67	4	0	5
wer_9	1.88	16.67	4	0	5
wer_10	1.46	12.5	2	0	5
wer_11	1.04	10.42	2	0	3
wer_12	0.83	8.33	2	0	2
wer_13	0.83	8.33	2	0	2
wer_14	0.62	6.25	2	0	1
wer_15	0.42	4.17	1	0	1
wer_16	0.42	4.17	1	0	1
wer_17	0.21	2.08	2.27	0	3
Mean	1.10	10.23			

Among the five (5) folds, the fold with the highest average WER and SER percentage is the fourth fold with 2.92% and 22.92%, followed by the third fold with 1.61% and 12.50%, the fifth fold with 1.10% and 10.23%, the second fold with 0.70% and 6.44%, and the first fold with 0.38% and 5.42% respectively. With this, the monophone model gives a WER percentage ranging from 0.38% to 2.92% and an SER percentage of 5.42% to 22.92%.

4.3 Result of Triphone Delta + Delta-Delta model

Table 4.11 shows the results of using the first fold decoding of the triphone delta + delta-delta model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.83% and 7.58%, respectively.

Table 4.11: Results of the first fold decoding of triphone delta + delta-delta model

	WER	SER	Insertion	Deletion	Substitution
wer_7	3.33	29.17	6	0	10
wer_8	2.71	22.92	4	0	9
wer_9	1.25	12.5	2	0	4
wer_10	0.83	8.33	1	0	3
wer_11	0.83	8.33	1	0	3
wer_12	0.21	2.08	1	0	0
wer_13	0	0	0	0	0
wer_14	0	0	0	0	0
wer_15	0	0	0	0	0
wer_16	0	0	0	0	0
wer_17	0	0	0	0	0
Mean	0.83	7.58	1.36	0	2.64

Table 4.12 shows the results of using the second fold decoding of the triphone delta + delta-delta model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.95% and 7.95%, respectively.

Table 4.12: Results of the second fold decoding of triphone delta + delta-delta model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.5	18.75	9	0	3
wer_8	2.08	18.75	7	0	3
wer_9	1.67	14.58	5	0	3
wer_10	0.83	6.25	1	0	3
wer_11	0.83	6.25	1	0	3
wer_12	0.83	6.25	1	0	3
wer_13	0.62	6.25	1	0	2
wer_14	0.42	4.17	1	0	1
wer_15	0.21	2.08	0	0	1
wer_16	0.21	2.08	0	0	1
wer_17	0.21	2.08	0	0	1
Mean	0.95	7.95	2.36	0	2.18

Table 4.13 shows the results of using the third fold decoding of the triphone delta + delta-delta model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.51% and 12.69%, respectively.

Table 4.13: Results of the third fold decoding of triphone delta + delta-delta model

	WER	SER	Insertion	Deletion	Substitution
wer_7	3.12	20.83	1	4	10
wer_8	2.29	18.75	0	4	7
wer_9	1.67	14.58	0	4	4
wer_10	1.46	12.5	0	4	3
wer_11	1.25	10.42	0	4	2
wer_12	1.25	10.42	0	4	2
wer_13	1.25	10.42	0	4	2
wer_14	1.25	10.42	0	4	2
wer_15	1.04	10.42	0	4	1
wer_16	1.04	10.42	0	4	1
wer_17	1.04	10.42	0	4	1
Mean	1.51	12.69	0.09	4	3.18

Table 4.14 shows the results of using the fourth fold decoding of the triphone delta + delta-delta model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.81% and 7.76%, respectively.

Table 4.14: Results of the fourth fold decoding of triphone delta + delta-delta model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.5	20.83	5	1	6
wer_8	1.46	14.58	3	1	3
wer_9	1.25	12.5	3	1	2
wer_10	0.62	6.25	1	1	1
wer_11	0.62	6.25	1	1	1
wer_12	0.62	6.25	1	1	1
wer_13	0.62	6.25	1	1	1
wer_14	0.62	6.25	1	1	1
wer_15	0.21	2.08	0	1	0
wer_16	0.21	2.08	0	1	0
wer_17	0.21	2.08	0	1	0
Mean	0.81	7.76	1.45	1	1.45

Table 4.15 shows the results of using the fifth fold decoding of the triphone delta + delta-delta model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.72% and 14.20%, respectively.

Table 4.15: Results of the fifth fold decoding of triphone delta + delta-delta model

	WER	SER	Insertion	Deletion	Substitution
wer_7	4.38	35.42	7	1	13
wer_8	2.92	22.92	4	1	9
wer_9	1.88	16.67	3	1	5
wer_10	1.67	14.58	3	1	4
wer_11	1.46	12.5	2	1	4
wer_12	1.25	10.42	2	1	3
wer_13	1.25	10.42	2	1	3
wer_14	1.04	8.33	2	1	2
wer_15	1.04	8.33	2	1	2
wer_16	1.04	8.33	2	1	2
wer_17	1.04	8.33	2	1	2
Mean	1.72	14.20	2.82	1	4.45

Among the five (5) folds, the fold with the highest average WER and SER percentage is the fifth fold with 1.72% and 14.20%, followed by the third fold with 1.51% and 12.69%, the second fold with 0.95% and 7.95%, the first fold with 0.83% and 7.58%, and the fourth fold with 0.81% and 7.76% respectively. With this, the triphone delta + delta-delta model gives a WER percentage ranging from 0.81% to 1.72% and an SER percentage of 7.76% to 14.20%.

4.4 Result of LDA + MLLT model

Table 4.16 shows the results of using the first fold decoding of the LDA + MLLT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.74% and 6.63%, respectively.

Table 4.16: Results of the first fold decoding of LDA + MLLT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.29	20.83	2	0	9
wer_8	1.88	16.67	1	0	8
wer_9	1.46	12.5	0	0	7
wer_10	1.25	10.42	0	0	6
wer_11	0.83	8.33	0	0	4
wer_12	0.21	2.08	0	0	1
wer_13	0.21	2.08	0	0	1
wer_14	0	0	0	0	0
wer_15	0	0	0	0	0
wer_16	0	0	0	0	0
wer_17	0	0	0	0	0
Mean	0.74	6.63	0.27	0	3.27

Table 4.17 shows the results of using the second fold decoding of the LDA + MLLT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.95% and 6.82%, respectively.

Table 4.17: Results of the second fold decoding of LDA + MLLT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.08	16.67	5	0	5
wer_8	1.46	10.42	3	0	4
wer_9	1.46	10.42	3	0	4
wer_10	1.25	10.42	3	0	3
wer_11	1.25	10.42	3	0	3
wer_12	0.83	6.25	2	0	2
wer_13	0.42	2.08	0	0	2
wer_14	0.42	2.08	0	0	2
wer_15	0.42	2.08	0	0	2
wer_16	0.42	2.08	0	0	2
wer_17	0.42	2.08	0	0	2
Mean	0.95	6.82	1.73	0	2.82

Table 4.18 shows the results of using the third fold decoding of the LDA + MLLT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.55% and 12.69%, respectively.

Table 4.18: Results of the third fold decoding of LDA + MLLT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	3.33	25	2	4	10
wer_8	1.88	14.58	2	4	3
wer_9	1.46	12.5	1	4	2
wer_10	1.46	12.5	1	4	2
wer_11	1.46	12.5	1	4	2
wer_12	1.25	10.42	1	4	1
wer_13	1.25	10.42	1	4	1
wer_14	1.25	10.42	1	4	1
wer_15	1.25	10.42	1	4	1
wer_16	1.25	10.42	1	4	1
wer_17	1.25	10.42	1	4	1
Mean	1.55	12.69	1.182	4	2.27

Table 4.19 shows the results of using the fourth fold decoding of the LDA + MLLT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 2.03% and 11.93%, respectively.

Table 4.19: Results of the fourth fold decoding of LDA + MLLT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	4.38	25	9	1	11
wer_8	3.96	20.83	9	1	9
wer_9	3.12	18.75	8	1	6
wer_10	2.08	12.5	4	1	5
wer_11	1.67	8.33	3	1	4
wer_12	1.67	8.33	3	1	4
wer_13	1.25	8.33	2	1	3
wer_14	1.25	8.33	2	1	3
wer_15	1.25	8.33	2	1	3
wer_16	0.83	6.25	1	1	2
wer_17	0.83	6.25	1	1	2
Mean	2.03	11.93	4	1	4.73

Table 4.20 shows the results of using the fifth fold decoding of the LDA + MLLT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.70% and 6.63%, respectively.

Table 4.20: Results of the fifth fold decoding of LDA + MLLT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.29	18.75	3	0	8
wer_8	1.04	10.42	2	0	3
wer_9	0.83	8.33	2	0	2
wer_10	0.83	8.33	2	0	2
wer_11	0.83	8.33	2	0	2
wer_12	0.62	6.25	2	0	1
wer_13	0.42	4.17	1	0	1
wer_14	0.21	2.08	1	0	0
wer_15	0.21	2.08	1	0	0
wer_16	0.21	2.08	1	0	0
wer_17	0.21	2.08	1	0	0
Mean	0.70	6.63	1.63	0	1.72

Among the five (5) folds, the fold with the highest average WER and SER percentage is the fourth fold with 2.03% and 11.93%, followed by the third fold with 1.55% and 12.69%, the second fold with 0.95% and 6.82%, the first fold with 0.74% and 6.63%, and the fifth fold with 0.70% and 6.63% respectively. With this, the LDA + MLLT model gives a WER percentage ranging from 0.70% to 2.03% and an SER percentage of 6.63% to 11.93%.

4.5 Result of SAT model

Table 4.21 shows the results of using the first fold decoding of the SAT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.81% and 8.14%, respectively.

Table 4.21: Results of the first fold decoding of SAT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	2.08	20.83	2	0	8
wer_8	2.08	20.83	2	0	8
wer_9	1.67	16.67	1	0	7
wer_10	1.04	10.42	0	0	5
wer_11	1.04	10.42	0	0	5
wer_12	0.62	6.25	0	0	3
wer_13	0.42	4.17	0	0	2
wer_14	0	0	0	0	0
wer_15	0	0	0	0	0
wer_16	0	0	0	0	0
wer_17	0	0	0	0	0
Mean	0.81	8.14	0.45	0	3.45

Table 4.22 shows the results of using the second fold decoding of the SAT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.14% and 8.33%, respectively.

Table 4.22: Results of the second fold decoding of SAT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	3.33	20.83	7	0	9
wer_8	2.5	14.58	6	0	6
wer_9	2.08	14.58	5	0	5
wer_10	1.67	12.5	4	0	4
wer_11	0.83	8.33	1	0	3
wer_12	0.83	8.33	1	0	3
wer_13	0.42	4.17	0	0	2
wer_14	0.21	2.08	0	0	1
wer_15	0.21	2.08	0	0	1
wer_16	0.21	2.08	0	0	1
wer_17	0.21	2.08	0	0	1
Mean	1.14	8.33	2.18	0	3.27

Table 4.23 shows the results of using the third fold decoding of the SAT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 2.48% and 16.67%, respectively.

Table 4.23: Results of the third fold decoding of SAT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	5	25	3	5	16
wer_8	4.17	25	3	5	12
wer_9	2.92	18.75	3	5	6
wer_10	2.29	16.67	1	5	5
wer_11	2.29	16.67	1	5	5
wer_12	2.29	16.67	1	5	5
wer_13	2.08	14.58	1	5	4
wer_14	1.67	12.5	1	5	2
wer_15	1.67	12.5	1	5	2
wer_16	1.46	12.5	1	5	1
wer_17	1.46	12.5	1	5	1
Mean	2.48	16.67	1.55	5	5.36

Table 4.24 shows the results of using the fourth fold decoding of the SAT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.80% and 14.77%, respectively.

Table 4.24: Results of the fourth fold decoding of SAT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	5	35.42	9	1	14
wer_8	3.75	27.08	6	1	11
wer_9	2.71	22.92	5	1	7
wer_10	2.08	18.75	4	1	5
wer_11	2.08	18.75	4	1	5
wer_12	1.25	10.42	2	1	3
wer_13	0.83	8.33	1	1	2
wer_14	0.62	6.25	1	1	1
wer_15	0.62	6.25	1	1	1
wer_16	0.42	4.17	0	1	1
wer_17	0.42	4.17	0	1	1
Mean	1.80	14.77	3	1	4.64

Table 4.25 shows the results of using the fifth fold decoding of the SAT model. Word and sentence errors start from wer 7 to wer 17 following descending order. Furthermore, the average WER and SER percentages of this fold are 2.18% and 15.34%, respectively.

Table 4.25: Results of the fifth fold decoding of SAT model

	WER	SER	Insertion	Deletion	Substitution
wer_7	5.42	35.42	3	0	23
wer_8	5	31.25	3	0	21
wer_9	3.54	25	1	0	16
wer_10	3.12	22.92	1	0	14
wer_11	2.71	18.75	1	0	12
wer_12	1.88	12.5	0	1	8
wer_13	0.83	8.33	0	1	3
wer_14	0.62	6.25	0	1	2
wer_15	0.42	4.17	0	1	1
wer_16	0.21	2.08	0	1	0
wer_17	0.21	2.08	0	1	0
Mean	2.18	15.34	0.82	0.55	9.09

Among the five (5) folds, the fold with the highest average WER and SER percentage is the third fold with 2.48% and 16.67%, followed by the fifth fold with 2.18% and 15.34%, the fourth fold with 1.80% and 14.77%, the second fold with 1.14% and 8.33%, and the first fold with 0.81% and 8.14% respectively. With this, the SAT model gives a WER percentage ranging from 0.81% to 2.48% and an SER percentage of 8.14% to 16.67%.

4.6 Result of DNN model

Table 4.26 shows the results of using the first fold decoding of the DNN model. Word and sentence errors start from wer 9 to wer 11 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.07% and 0.69%, respectively.

Table 4.26: Results of first fold decoding of DNN model

	WER	SER	Insertion	Deletion	Substitution
wer_9	0	0	0	0	1
wer_10	0	0	0	0	0
wer_11	0.21	2.08	0	0	0
Mean	0.07	0.69	0	0	0.33

Table 4.27 shows the results of using the second fold decoding of the DNN model. Word and sentence errors start from wer 9 to wer 11 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.00% and 0.00%, respectively.

Table 4.27: Results of second fold decoding of DNN model

	WER	SER	Insertion	Deletion	Substitution
we_9	0	0	0	0	0
we_10	0	0	0	0	0
we_11	0	0	0	0	0
Mean	0	0	0	0	0

Table 4.28 shows the results of using the third fold decoding of the DNN model. Word and sentence errors start from wer 9 to wer 11 following descending order. Furthermore, the average WER and SER percentages of this fold are 1.46% and 11.81%, respectively.

Table 4.28: Results of third fold decoding of DNN model

	WER	SER	Insertion	Deletion	Substitution
wer_9	1.67	12.5	1	6	1
wer_10	1.04	10.42	1	6	1
wer_11	1.67	12.5	0	5	0
Mean	1.46	11.81	0.67	5.67	0.67

Table 4.29 shows the results of using the fourth fold decoding of the DNN model. Word and sentence errors start from wer 9 to wer 11 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.21% and 2.08%, respectively.

Table 4.29: Results of fourth fold decoding of DNN model

	WER	SER	Insertion	Deletion	Substitution
wer_9	0.21	2.08	0	1	0
wer_10	0.21	2.08	0	1	0
wer_11	0.21	2.08	0	1	0
Mean	0.21	2.08	0	1	0

Table 4.30 shows the results of using the fifth fold decoding of the DNN model. Word and sentence errors start from wer 9 to wer 11 following descending order. Furthermore, the average WER and SER percentages of this fold are 0.00% and 0.00%, respectively.

Table 4.30: Results of fifth fold decoding of DNN model

	WER	SER	Insertion	Deletion	Substitution
wer_9	0	0	0	0	0
wer_10	0	0	0	0	0
wer_11	0	0	0	0	0
Mean	0	0	0	0	0

Among the five (5) folds, the fold with the highest average WER and SER percentage is the third fold with 1.46% and 11.81%, followed by the fourth fold with 0.21% and 2.08%, the first fold with 0.07% and 0.69%, and the second and fifth fold with 0.00% and 0.00% respectively. With this, the DNN model gives a WER percentage ranging from 0.00% to 1.46% and an SER percentage of 0.00% to 11.81%.

4.7 Summary of 5-fold cross validation

Table 4.31 shows the summary of the WER results of the 5-fold cross-validation. It shows the average Word Error Rate (WER) for each acoustic models for every fold (f1, f2, f3, f4, f5) and the mean of the means. Where every fold has 2 speakers, e.g. first fold (f1) has speaker 1 (s1) and speaker 2 (s2). From this table, it shows that the DNN model is the best model among the rest having the lowest average WER means of 0.35%.

Table 4.31: Summary of 5-fold cross validation for WER results

Model Method	F1(s1,s2)	F2(s3,s4)	F3(s5,s6)	F4(s7,s8)	F5(s9,s10)	Mean
Monophone	0.04	0.53	3.24	0.66	0.57	1.01
Triphone	0.38	0.70	1.61	0.95	1.1	0.95
$\Delta + \Delta\Delta$ Triphone	0.83	0.95	1.51	0.81	1.72	1.16
LDA + MLLT	0.74	0.95	1.55	2.03	0.7	1.19
LDA + MLLT + SAT	0.81	1.14	2.48	1.8	2.18	1.68
DNN	0.07	0	1.46	0.21	0	0.35

Table 4.32 shows the summary of the SER results of the 5-fold cross-validation. It shows the average Sentence Error Rate (SER) for each acoustic models for every fold (f1, f2, f3, f4, f5) and the mean of the means. Where every fold has 2 speakers, e.g. first fold (f1) has speaker 1 (s1) and speaker 2 (s2). From this table, it shows that the DNN model is the best model among the rest having the lowest average SER means of 2.92%.

Table 4.32: Summary of 5-fold cross validation for SER results

Model Method	F1(s1,s2)	F2(s3,s4)	F3(s5,s6)	F4(s7,s8)	F5(s9,s10)	Mean
Monophone	0.38	5.30	21.59	6.25	5.11	7.73
Triphone	5.42	6.44	12.50	8.15	10.23	8.55
$\Delta + \Delta\Delta$ Triphone	7.58	7.95	12.69	7.76	14.20	10.04
LDA + MLLT	6.63	6.82	12.69	11.93	6.63	8.94
LDA + MLLT + SAT	8.14	8.33	16.67	14.77	15.34	12.65
DNN	0.69	0	11.81	2.08	0	2.92

Chapter 5

Conclusion and Recommendation

5.1 Conclusion

In this study, the researchers developed a Hiligaynon speech recognition system called HiliSpeech using the Kaldi ASR toolkit that transcribes uttered common and command Hiligyanon words. The system corpus has 240 recordings with five (5) males and five (5) females, all fluent in the Hiligaynon language. The HiliSpeech was assessed based on the Word Error Rate (WER) and Sentence Error Rate (SER).

The evaluation process consisted of various training models such as monopohone, triphone(delta, delta + delta-delta), Linear Discriminate Analysis Maximum Likelihood Linear Transform (LDA+MLLT), Speaker Adapting Training (SAT), and Deep Neural Network (DNN) model. In addition to these models, a statistic method, five-fold cross-validation, was used in assessing the results of each model.

The system's overall performance ranges from 99.65% to 98.32%, depending on the training method used. When the means of the five-fold cross-validation were compared, the best model that yielded the lowest word error rate (WER) was the DNN with 0.35%, followed by the triphone delta-based model with 0.95%. Furthermore, the monophone produced 1.01% WER, triphone delta + delta-delta with 1.16%, the LDA + MLLT with 1.19%, and the SAT model with 1.68%. With these results, the researchers found out that the best acoustic training model to be used in this system with the highest speech recognition accuracy was the DNN model. This aligns with the statements mentioned in the second chapter, where DNN-models outperform GMM acoustic models (Dahl et al., 2012; Seide et al., 2011; Maas et al., 2017)

5.2 Recommendation

Further studies can be done by adding more words that might enhance the vocabulary of the system. Additionally, expanding the training data set with more structured sentences might improve the accuracy of the WER and SER, since other language models other than the monophone model can yield better results if every utterance is more grammatically structured. Furthermore, this study can be used by future researchers if they want to continue expanding the study, or it can be used for their research.

Chapter 6

References

Ager, S. (n.d.). *Hiligaynon (ilonggo).* Retrieved from `https://omniglot.com/writing/hiligaynon.htm`

Ayaz, A., & Shaukat, S. (2021, 12). Neural network solution for secure interactive voice response.

Bautista, J. L., & Kim, Y. (2014). An automatic speech recognition for the filipino language using the htk system..

Bhatt, S., Jain, A., & Dev, A. (2020). Acoustic modeling in speech recognition: A systematic review. *International Journal of Advanced Computer Science and Applications*, *11*(4). Retrieved from `http://dx.doi.org/10.14569/IJACSA.2020.0110455` doi: 10.14569/IJACSA.2020.0110455

Billones, R. K. C., & Dadios, E. P. (2014). Hiligaynon language 5-word vocabulary speech recognition using mel frequency cepstrum coefficients and genetic algorithm. In *2014 international conference on humanoid, nanotechnology, information technology, communication and control, environment and man-*

agement (hnicem) (p. 1-6). doi: 10.1109/HNICEM.2014.7016247

Brucal, S. G. E., Dabo, J. V. M., Lagunay, K. M. L., Yong, E. D., Samaniego, L. A., Dela Cruz, G. B., & Martin, J. N. L. (2021). Filipino speech to text system using convolutional neural network. In *2021 fifth world conference on smart trends in systems security and sustainability (worlds4)* (p. 176-181). doi: 10.1109/WorldS451998.2021.9513991

Bushofa, B., & Bazina, N. (2014, 01). User authentication based on his/her speech..

Casperson, T. (2010, Dec). *The phonology of hiligaynon.*

Chen, K.-Y., & Chen, B. (2011, 05). Relevance language modeling for speech recognition. In (p. 5568-5571). doi: 10.1109/ICASSP.2011.5947621

Chodroff, E. (2015, Jul). *Kaldi tutorial.* Retrieved from `https://www.eleanorchodroff.com/tutorial/kaldi/index.html`

Crook, J. (n.d.). *Copyright.* Retrieved from `https://www.audacityteam.org/copyright/`

Dahl, G., Yu, D., Deng, l., & Acero, A. (2012, 02). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, *20*, 30 - 42. doi: 10.1109/TASL.2011.2134090

Dave, N. (2013, 07). Feature extraction methods lpc, plp and mfcc in speech recognition. *International Journal For Advance Research in Engineering And Technology(ISSN 2320-6802)*, *Volume 1*.

Dimzon, F. D., & Pascual, R. M. (2020). An automatic phoneme recognizer for children's Filipino read speech. In *2020 ieee international conference on teaching, assessment, and learning for engineering (tale)* (p. 1-5). doi: 10.1109/TALE48869.2020.9368399

Dioses Jr, J. (2020, 06). Androiduino-fan: A speech recognition fan-speed control system utilizing Filipino voice commands. *International Journal of Advanced Trends in Computer Science and Engineering*, *9*, 3042-3047. doi: 10.30534/ijatcse/2020/84932020

Gaikwad, S., Bharti, W., & Yannawar, P. (2010, 11). A review on speech recognition technique. *International Journal of Computer Applications*, *10*. doi: 10.5120/1462-1976

Joby, A. (n.d.). *What is cross-validation? comparing machine learning models.* Retrieved from https://learn.g2.com/cross-validation

Kipyatkova, I., & Karpov, A. (2016, 08). Dnn-based acoustic modeling for russian speech recognition using kaldi. In (p. 246-253). doi: 10.1007/978-3-319-43958-7_29

Ling, Z.-H., Kang, S., Zen, H., Senior, A., Schuster, M., Qian, X.-J., ... Deng, L. (2015). Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends. *IEEE Signal Processing Magazine*, *32*, 35-52. Retrieved from http://dx.doi.org/10.1109/MSP.2014.2359987

Maas, A. L., Qi, P., Xie, Z., Hannun, A. Y., Lengerich, C. T., Jurafsky, D., & Ng, A. Y. (2017). Building dnn acoustic models for large vocabulary speech recognition. *Computer Speech Language*, *41*, 195-213. Retrieved from https://www.sciencedirect.com/science/article/pii/S0885230816301930 doi: https://doi.org/10.1016/j.csl.2016.06.007

Mansikkaniemi, A. (2010). *Acoustic model and language model adaptation for a mobile dictation service* (Master's thesis). Retrieved from http://urn.fi/URN:NBN:fi:aalto-201203131407

Miao, Y., Zhang, H., & Metze, F. (2015). Speaker adaptive training of deep neural network acoustic models using i-vectors. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *23*(11), 1938–1949. doi: 10.1109/taslp.2015.2457612

Michel, L., Hangya, V., & Fraser, A. (2020, May). Filipino speech to text system using convolutional neural network. In *2021 fifth world conference on smart trends in systems security and sustainability (worlds4)* (Vol. Proceedings of the 12th Language Resources and Evaluation Conference, p. 2573–2580). European Language Resources Association.

Motus, C. L. (2019). *Hiligaynon dictionary.* University of Hawai'i Press. Retrieved from `http://www.jstor.org/stable/j.ctv9hvsxq`

Naeem, S., Iqbal, M., Saqib, M., Saad, M., Raza, M. S., Ali, Z., … Arshad, M. U. (2020). Subspace gaussian mixture model for continuous urdu speech recognition using kaldi. In *2020 14th international conference on open source systems and technologies (icosst)* (p. 1-7). doi: 10.1109/ICOSST51357.2020.9333026

Pascual, R., malaay, e., Cabatic, R., Castillo, A., & Cabotaje, A. (2017, 04). Prototyping a computer-aided ilokano language learning (caill) system for tagalog speakers..

Piero. (2020, Mar). *Julius.* Retrieved from `https://www3.pd.istc.cnr.it/piero/ASR/julius.htm#references`

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., … Vesely, K. (2011, December). The kaldi speech recognition toolkit. In *Ieee 2011 workshop on automatic speech recognition and understanding.* IEEE Signal Processing Society. (IEEE Catalog No.: CFP11SRW-USB)

Robles, C. Y. (2012, Feb). *Hiligaynon: an endangered language?* Retrieved from `https://mlephil.wordpress.com/2012/02/26/hiligaynon`

-an-endangered-language/

S, K., & Chandra, E. (2016, 04). A review on automatic speech recognition architecture and approaches. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, *9*, 393-404. doi: 10.14257/ijsip.2016.9.4.34

Sahu, P. K., & Ganesh, D. S. (2015). A study on automatic speech recognition toolkits. In *2015 international conference on microwave, optical and communication engineering (icmoce)* (p. 365-368). doi: 10.1109/ICMOCE.2015.7489768

Seide, F., Li, G., & Yu, D. (2011). Conversational speech transcription using context-dependent deep neural networks. In *Twelfth annual conference of the international speech communication association.*

Shadiev, R., Reynolds, B. L., Huang, Y.-M., Shadiev, N., Wang, W., Laxmisha, R., & Wannapipat, W. (2017). Applying speech-to-text recognition and computer-aided translation for supporting multi-lingual communications in cross-cultural learning project. In *2017 ieee 17th international conference on advanced learning technologies (icalt)* (p. 182-183). doi: 10.1109/ICALT.2017.20

Shrawankar, U., & Mahajan, A. (2013, 05). Speech: A challenge to digital signal processing technology for human-to-computer interaction.

Soltys, M. (2018). Dynamic programming. In *An introduction to the analysis of algorithms* (p. 71-93). Retrieved from https://www.worldscientific.com/doi/abs/10.1142/9789813235915_0004 doi: 10.1142/9789813235915_0004

Upadhyaya, P., Farooq, O., Abidi, M. R., & Varshney, Y. V. (2017). Continuous hindi speech recognition model based on kaldi asr toolkit. In *2017* *international conference on wireless communications, signal processing and networking (wispnet)* (p. 786-789). doi: 10.1109/WiSPNET.2017.8299868

Vyas, G., & Kumari, B. (2013, 06). Speaker recognition system based on mfcc and dct. *International Journal of engineering and advanced technology*, *2*, 167-169.

Wang, W. (2015, Mar). *Formula coding.* Retrieved from `http://wantee.github.io/2015/03/14/feature-extraction-for-asr-preprocessing/`

Wolfenden, E. P. (2019). Hiligaynon reference grammar. doi: 10.2307/j.ctv9hvst8

Yin, S., Liu, C., Zhang, Z., Lin, Y., Wang, D., Tejedor, J., ... Li, Y. (2015). Noisy training for deep neural networks in speech recognition. *EURASIP Journal on Audio, Speech, and Music Processing*, *2015*(1). doi: 10.1186/s13636-014-0047-0

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., & Povey, D. (2015). *The htk book.* Entropic Cambridge Research Laboratory.

Appendix A

Code Snippets

```
***TRAINING GMM BASED MODELS: train_gmm.sh***
echo
echo "===== MONO TRAINING ====="
echo

steps/train_mono.sh --boost-silence 1.25 --nj $nj --cmd "
    $train_cmd" \
data/train data/lang exp/mono

echo
echo "===== MONO ALIGNMENT ====="
echo

steps/align_si.sh --boost-silence 1.25 --nj $nj --cmd "$train_cmd
    " \
data/train data/lang exp/mono exp/mono_ali || exit 1;

echo
```

```
echo "===== TRI1 (DELTA-BASED TRIPHONE) TRAINING ====="
echo

steps/train_deltas.sh --boost-silence 1.25 --cmd "$train_cmd" \
100 200 data/train data/lang exp/mono_ali exp/tri1 || exit 1;

echo
echo "===== TRI1 (DELTA-BASED TRIPHONE) ALIGNMENT ====="
echo

steps/align_si.sh --nj $nj --cmd "$train_cmd" \
data/train data/lang exp/tri1 exp/tri1_ali || exit 1;

echo
echo "===== TRI2A (DELTA + DELTA-DELTA TRIPHONE) TRAINING ====="
echo

steps/train_deltas.sh --cmd "$train_cmd" \
100 200 data/train data/lang exp/tri1_ali exp/tri2a || exit 1;

echo
echo "===== TRI2A (DELTA + DELTA-DELTA TRIPHONE) ALIGNMENT ====="
echo

steps/align_si.sh  --nj $nj --cmd "$train_cmd" \
--use-graphs true data/train data/lang exp/tri2a exp/tri2a_ali
   || exit 1;

echo
echo "===== TRI3A (LDA-MLLT TRIPHONE) TRAINING ====="
echo
```

```
steps/train_lda_mllt.sh --cmd "$train_cmd" \
100 200 data/train data/lang exp/tri2a_ali exp/tri3a || exit 1;

echo
echo "===== TRI3A (LDA-MLLT TRIPHONE) ALIGNMENT with FMLLR ====="
echo

steps/align_fmllr.sh --nj $nj --cmd "$train_cmd" \
data/train data/lang exp/tri3a exp/tri3a_ali || exit 1;

echo
echo "===== TRI4A (SAT TRIPHONE) TRAINING ====="
echo

steps/train_sat.sh  --cmd "$train_cmd" \
100 200 data/train data/lang exp/tri3a_ali exp/tri4a || exit 1;

echo
echo "===== TRI4A (SAT TRIPHONE) ALIGNMENT with FMLLR ====="
echo

steps/align_fmllr.sh  --cmd "$train_cmd" \
data/train data/lang exp/tri4a exp/tri4a_ali || exit 1;

***TRAINING DNN MODEL: train_dnn.sh***
echo
echo "===== DNN TRAINING ====="
echo

mkdir -p $experiment_dir
```

```
steps/nnet2/train_simple.sh \
    --stage -10 \
    --num-threads "$num_threads" \
    --feat-type raw \
    --splice-width 4 \
    --lda_dim 65 \
    --num-hidden-layers 2 \
    --hidden-layer-dim 50 \
    --add-layers-period 5 \
    --num-epochs 10 \
    --iters-per-epoch 2 \
    --initial-learning-rate 0.02 \
    --final-learning-rate 0.004 \
    --minibatch-size "$minibatch_size" \
    data/train \
    data/lang \
    exp/tri3a_ali \
    $experiment_dir \
    || exit 1;

echo
echo "===== END DNN TRAINING ====="
echo


***TESTING GMM BASED MODEL: test_gmm.sh***
echo
echo "===== MONO TESTING ====="
echo

utils/mkgraph.sh --mono data/lang exp/mono exp/mono/graph || exit
```

```
    1
steps/decode.sh --config conf/decode.conf --nj $nj --cmd "
   $decode_cmd" exp/mono/graph data/test exp/mono/decode

echo
echo "===== TRI1 (DELTA-BASED TRIPHONE) TESTING ====="
echo

utils/mkgraph.sh data/lang exp/tri1 exp/tri1/graph || exit 1
steps/decode.sh --config conf/decode.conf --nj $nj --cmd "
   $decode_cmd" exp/tri1/graph data/test exp/tri1/decode

echo
echo "===== TRI2A (DELTA + DELTA-DELTA TRIPHONE) TESTING ====="
echo

utils/mkgraph.sh data/lang exp/tri2a exp/tri2a/graph || exit 1
steps/decode.sh --config conf/decode.conf --nj $nj --cmd "
   $decode_cmd" exp/tri2a/graph data/test exp/tri2a/decode

echo
echo "===== TRI3A (LDA-MLLT TRIPHONE) TESTING ====="
echo

utils/mkgraph.sh data/lang exp/tri3a exp/tri3a/graph || exit 1
steps/decode.sh --config conf/decode.conf --nj $nj --cmd "
   $decode_cmd" exp/tri3a/graph data/test exp/tri3a/decode

echo
echo "===== TRI4A (SAT TRIPHONE) TESTING ====="
echo
```

```
utils/mkgraph.sh data/lang exp/tri4a exp/tri4a/graph || exit 1
steps/decode.sh --config conf/decode.conf --nj $nj --cmd "
   $decode_cmd" exp/tri4a/graph data/test exp/tri4a/decode

echo
echo "===== GMM TESTING DONE ====="
echo

***TESTING DNN MODEL: test_dnn.sh***
echo
echo "===== DNN TESTING ====="
echo

utils/mkgraph.sh data/lang exp/nnet2/nnet2_simple exp/nnet2/
   nnet2_simple/graph || exit 1

rm -r $experiment_dir/decode
mkdir $experiment_dir/decode

steps/nnet2/decode_simple.sh \
    --num-threads "$num_threads" \
    --beam 18 \
    --max-active 1000 \
    --lattice-beam 10 \
    --nj 2 \
    exp/nnet2/nnet2_simple/graph \
    data/test \
    $experiment_dir/final.mdl \
    $experiment_dir/decode \
    || exit 1;
```

```
for x in ${experiment_dir}/decode*; do
    [ -d $x ] && grep WER $x/wer_* | \
        utils/best_wer.sh > nnet2_simple_wer.txt;
done

echo
echo "===== END TESTING ====="
echo
```