System Design Document

For

RTube Kaldi Research Team

Team members:
Tabitha O'Malley, Milan Haruyama, David Serfaty, Tahmina Tisha, Adam Gallub

Revision History

Version/Author	Notes	Date
V1 Tabitha, Milan, David, Max, Tisha, Adam		09/29/23
V2.1 Tabitha	Writing Section:1.2.1	10/24/23
V2.2 Tisha,Tabitha, Milan	Rewriting the section: 2.2	10/24/23
V2.3 Tabitha, Tisha, Milan	Writing the section, Rewriting, and editing: 1.2	10/24/23
V2.4 Tabitha	Writing Sections: 2.1, 1.5	10/25/23
V2.5 Tabitha	Writing Sections: 1.1, 1.2.2, 1.3	10/26/23
V2.6 David	Writing Sections: 1.2, 1.5, 3.1, 3.2	10/26/23
V2.7 Milan	Writing Sections: 1.1, 1.2, 1.5	10/26/23
V2.8 Adam	Asking TA: 2.1 Write Section: 4.1	10/28/23
V2.9 Tisha	Asking TA: 2.1 Writing Section: 2.1, 5.1	10/28/23
V2.10 Tabitha	Writing/Rewriting Sections: : 1.2.1, 2.1, 2.2, 3.1, 3.2, 4, 4.1, 4.2, 5.2	10/29/23
V2.11 David	Writing/Rewriting Sections: 1.2.1, 1.2.2, 1.2.3, 2.1. 2.2, 3.1, 3.2, 4, 4.1, 5, 5.1, 6	10/29/23
V2.12 Tisha	Writing/Rewriting Section: 2.1	10/29/23
V2.13 Milan	Editing All Sections	10/29/23
V2.14 Milan	Editing All Sections	10/30/23
V2.15 Tabitha	Rewriting Section: 2.1	10/30/23
V2.16 Tabitha	Updating Models Editing Sections Writing Section: 4.2	10/31/23
V2.17 David	Rewriting sections: 1.5, 5.2 Editing sections Updating models (context, use case, DFD)	10/31/23
V2.18 Milan	Editing All Sections	10/31/23

TABLE OF CONTENTS

SYSTEM DESIGN DOCUMENT

1 INTRODUCTION

1.1 Purpose and Scope

As an aircraft pilot, learning to communicate with Air Traffic Control (ATC) is a daunting task. Despite being designed to mitigate miscommunication, aviation phraseology is highly intricate and requires hundreds of hours of training to learn its idiosyncrasies. While there exist a few resources (such as the website LiveATC) for student pilots to study aviation phraseology, these resources do little to accommodate the major learning curve present, especially since they do not provide live transcriptions of speech for student pilots to read.

As such, the RTube web application shall bridge the learning gap faced by many student pilots by providing the ability to transcribe live ATC transmissions into text in real time, as well as providing a live interface for users to track the flight paths of aircrafts. The live speech-to-text transcription is performed using an automatic speech recognition (ASR) model developed using the Kaldi ASR Toolkit. With the ability to transcribe speech to text in real time, student pilots can dramatically reduce the amount of training required to understand spoken aviation phraseology. In addition, the live interface helps students better understand different contexts that specific phrases are used in.

1.2 Project Executive Summary

The RTube Kaldi Research Team solely focuses on the development of the ASR model that shall be utilized by the RTube web application. Included in this section of the document is an overview of the sample ASR model provided by the Kaldi ASR Toolkit, design constraints for future ASR models, and general project contingencies.

1.2.1 System Overview

The sample ASR model transcribes audio into text. The ASR model utilizes phone, triphone, and word databases to convert the received audio into words. The ASR model then uses a predictive model to build the sentence structure for the converted words. After the audio has been fully transcribed, the ASR model returns the text to the user.

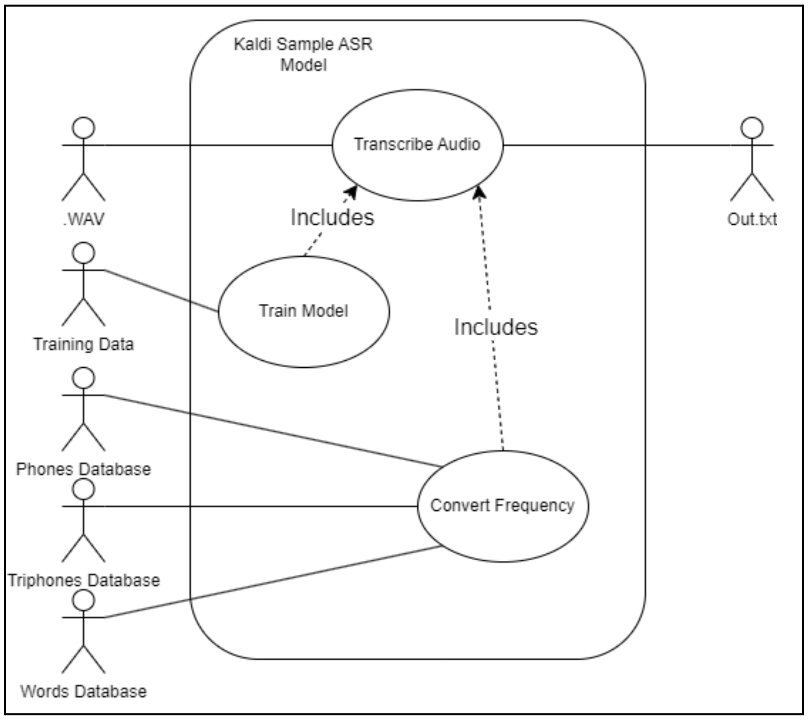


Figure 1: Sample ASR Model Use Case Diagram V1.6

1.2.2 Design Constraints

There are two constraints that limit the scope of any development done by the RTube Kaldi Research Team. Those are the complexity of speech recognition and the scale of the Kaldi ASR Toolkit. There are many challenges that speech recognition brings to the table, variability in speech, tone and pitch, accents, and even the speed at which the speaker talks. Each one of these brings a unique challenge. When it is speech recognition for an ATC application these challenges increase tenfold and even more challenges are added including the idiosyncrasies of ATC communications and static from the radios.

Due to the complexity of speech recognition the magnitude of the Kaldi ASR Toolkit is enormous. The average desktop computer does not have the capability to run and sometimes even download the toolkit. The scale of Kaldi also brings the challenge of understanding. This is because the developers of Kaldi did not put a lot of time into writing documentation that could be understood. This means that as the research team goes through the Kaldi code it is a steep learning curve and requires a great deal of teamwork and documentation to learn how the toolkit works.

1.2.3 Future Contingencies

The Kaldi Research Team is hindered by a lack of coordination and connection with the NeMo team, outdated hardware that is incapable of running the Kaldi ASR Toolkit.

1.3 Document Organization

This document shall discuss system architecture, human-machine interfaces, detailed design, external interfaces, and system integrity control in their own respective sections. The System Architecture section shall discuss the high-level structure and functionality of the sample ASR model included in the Kaldi ASR Toolkit. The Human-Machine Interfaces section shall discuss the inputs and outputs of the aforementioned ASR model. The Detailed Design section shall discuss the lower level of detail of the ASR model. External interfaces shall have a basic description of how other systems interface with the Kaldi ASR Toolkit. System integrity control shall cover any data that is sensitive information that if modified could threaten the integrity of the system.

1.4 Project References

Chester F. Carlson Center for Imaging Science | College of Science | RIT, www.cis.rit.edu/class/simg716/Gauss_History_FFT.pdf. Accessed 1 Nov. 2023.

Kaldi Team Product Vision Statement. Kaldi Team. 19 September 2023. Kaldi Drive.

Ravihara, Ransaka. "Gaussian Mixture Model Clearly Explained." *Medium*, Towards Data Science, 11 Jan. 2023, towardsdatascience.com/gaussian-mixture-model-clearly-explained-115010f7d4cf.

"About Pandas." *Pandas*, pandas.pydata.org/about/. Accessed 26 Oct. 2023.

"A Python Library to Read/WRITE EXCEL 2010 Xlsx/XLSM Files¶." *Openpyxl*, openpyxl.readthedocs.io/en/stable/. Accessed 26 Oct. 2023.

"LiveATC FAQ." *LiveATC.Net - Listen to Live Air Traffic Control over the Internet!*, www.liveatc.net/faq/. Accessed 26 Oct. 2023.

"What Is Kaldi?" *Kaldi*, www.kaldi-asr.org/doc/about.html. Accessed 26 Oct. 2023.

"What Is Speech Recognition?" *IBM*, www.ibm.com/topics/speech-recognition. Accessed 26 Oct. 2023.

1.5 Glossary

Term	Definition
ATC	***Air Traffic Control;*** the service that elicits communications between pilots and helps to prevent air traffic accidents.
ASR	***Automatic Speech Recognition;*** the ability for computers to recognize and translate spoken speech.
CLI	***Command-Line Interface;*** text-based interface that allows interaction from the user to the computer program.
DNN	***Deep Neural Network;*** a machine learning technique that represents learning and processing data in artificial neural networks.
ERAU	***Embry Riddle Aeronautical University;*** an aviation-centered university located in Daytona Beach, Florida.
FFT	***Fast Fourier Transform;*** algorithm used to obtain the spectrum or . frequency content of a signal.
GMM	***Gaussian Mixture Model;*** used to model the transition of phones.
HMM	***Hidden Markov Model;*** used to model transitions of tri-phones and words.
IPA	***International Phonetic Alphabet:*** an alphabetic system of phonetic notation developed by the International Phonetic Association; used to represent speech sounds in a standardized format
Lexicon	*pertaining to speech;* a library of words that are understood by the language model
MFC	***Mel-Frequency Cepstral;*** algorithm used to transition from change-over-frequency to a 39 dimension array.
MFCC	***Mel-Frequency Cepstral Coefficients;*** the units used by the MFC.
NLP	***Natural Language Processing;*** the culmination of computer science, linguistics, and machine learning.
Phone	*pertaining to speech;* a distinct speech sound or gesture;
Phoneme	*pertaining to speech;* a set of phones that can distinguish one word from another
Triphone	*pertaining to speech;* a sequence of three consecutive phonemes
WER	***Word Error Rate;*** the rate at which error in words occurs

2 SYSTEM ARCHITECTURE

2.1 System Software Architecture

FFT: Fast Fourier Transform - responsible for partitioning audio into frames, and converting audio data from change-over-time to change-over-frequency.

- Inputs: .WAV files
- Outputs: Frames

MFC: Mel-Frequency Cepstrum - responsible for converting frames into an array of MFCCs.

- Inputs: Frames
- Outputs: Array of MFCCs.

GMM: Gaussian Mixture Model - responsible for converting the MFCCs to phone indices.

- Inputs: Array of MFCCs
- Outputs: Phone Index

HMM: Hidden Markov Model - responsible for converting phone indices to triphones indices, and triphones indices to word indices.

- Inputs: Phone Index
- Outputs: Word Index

Language Model: responsible for predicting words based off of previously predicted words in order to create sentences.

- Inputs: Word Index
- Outputs: Sentences

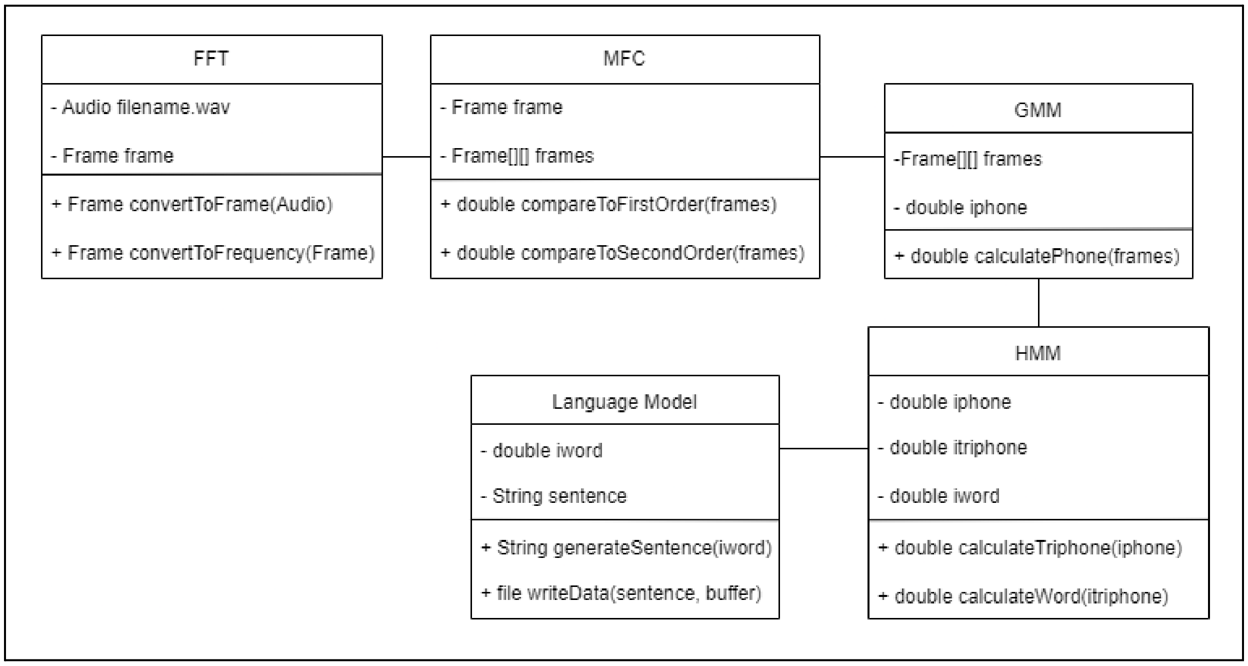

FFT
- Audio filename.wav
- Frame frame
+ Frame convertToFrame(Audio)
+ Frame convertToFrequency(Frame)
MFC
- Frame frame
- Frame[][] frames
+ double compareToFirstOrder(frames)
+ double compareToSecondOrder(frames)
GMM
-Frame[][] frames
- double iphone
+ double calculatePhone(frames)
HMM
- double iphone
- double itriphone
- double iword
+ double calculateTriphone(iphone)
+ double calculateWord(itriphone)
Language Model
- double iword
- String sentence
+ String generateSentence(iword)
+ file writeData(sentence, buffer)

Figure 2: Sample ASR Model Class Diagram V1.3

2.2 Internal Communications Architecture

The Kaldi ASR Toolkit provides users with a sample ASR model that receives audio files as input, and outputs text transcriptions of the audio. The model exclusively accepts .WAV format audio files and exclusively outputs text files. The text files shall only contain words based on General American English spelling conventions -- no punctuation or grammatical marks shall be included. The words shall be capitalized if they are found in the lexicon. If not, they are in lowercase. A more thorough analysis of the sample ASR model provided by the Kaldi ASR Toolkit is illustrated in Figure 6.

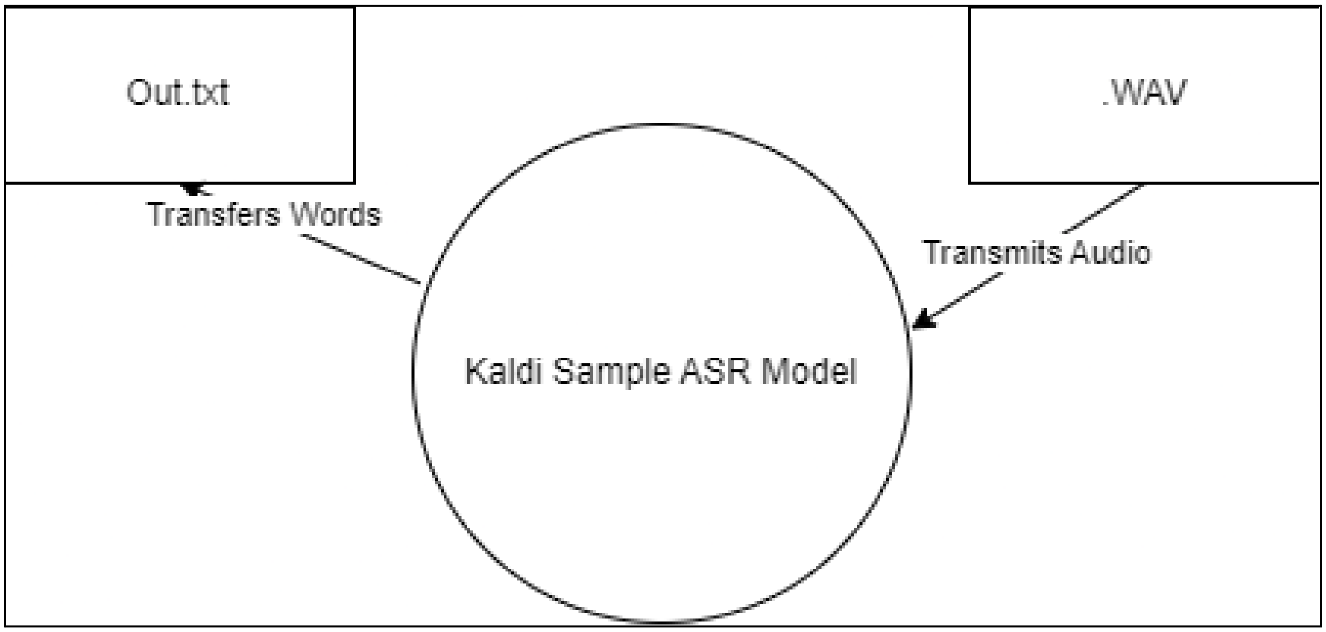
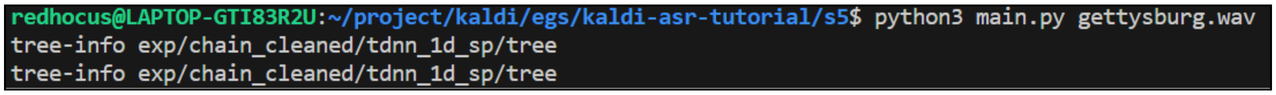


Figure 3: Sample ASR Model Level 0 Data Flow Diagram V1.2

3 HUMAN-MACHINE INTERFACE

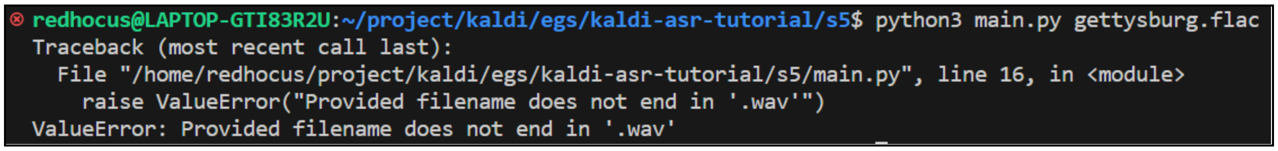
3.1 Inputs

The sample ASR model shall only accept .WAV files as input. Any other file format (e.g., .flac) shall be converted into .WAV beforehand. Input shall be given as a terminal command in the format, “python3 main.py file_name.wav” (*Figure 4*). The string, “python3”, is the calling and execution of a Python3 file. The string, “main.py”, is the file used to initiate the execution of the ASR model. Lastly, the string, “file_name.wav”, is the audio file that shall be transcribed. Any input attempt that does not adhere to this format shall not execute any code -- only an exception shall be returned (*Figure 5*).



```
redhocus@LAPTOP-GTI83R2U:~/project/kaldi/egs/kaldi-asr-tutorial/s5$ python3 main.py gettysburg.wav
tree-info exp/chain_cleaned/tdnn_1d_sp/tree
tree-info exp/chain_cleaned/tdnn_1d_sp/tree
```

Figure 4: Sample ASR Model Successful Input (with code execution snippet)



```
⊗ redhocus@LAPTOP-GTI83R2U:~/project/kaldi/egs/kaldi-asr-tutorial/s5$ python3 main.py gettysburg.flac
  Traceback (most recent call last):
    File "/home/redhocus/project/kaldi/egs/kaldi-asr-tutorial/s5/main.py", line 16, in <module>
      raise ValueError("Provided filename does not end in '.wav'")
  ValueError: Provided filename does not end in '.wav'
```

Figure 5: Sample ASR Model Unsuccessful Input

3.2 Outputs

The ASR model shall exclusively output all data into a .txt file called "out.txt" stored in the folder "/kaldi/egs/mini_librispeech/s5/". This .txt file shall allow users to read the transcription performed. The output shall be words based on General American English conventions. The words shall be capitalized if they are found in the lexicon -- if not, they shall be in lowercase. No grammatical or punctuation marks shall be included (*Figure 7*).

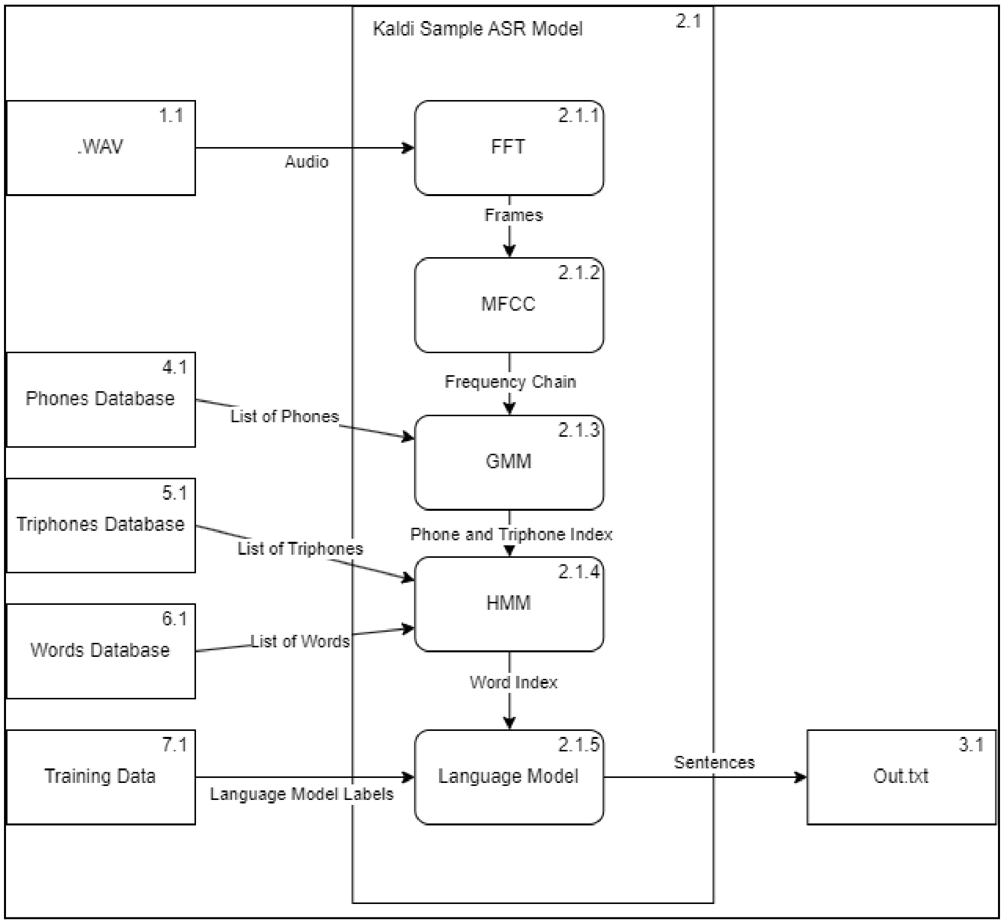
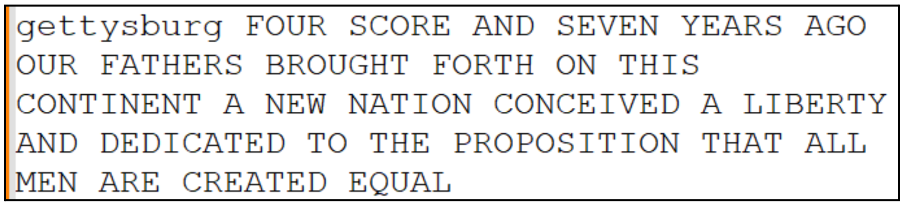


Figure 6: Sample ASR Model Level 1 Data Flow Diagram V2.5



```
gettysburg FOUR SCORE AND SEVEN YEARS AGO
OUR FATHERS BROUGHT FORTH ON THIS
CONTINENT A NEW NATION CONCEIVED A LIBERTY
AND DEDICATED TO THE PROPOSITION THAT ALL
MEN ARE CREATED EQUAL
```

Figure 7: Sample ASR Model Output

4 DETAILED DESIGN

4.1 Software Detailed Design

This section provides an overview of the use cases utilized by the ASR model in the form of a use case diagram below.

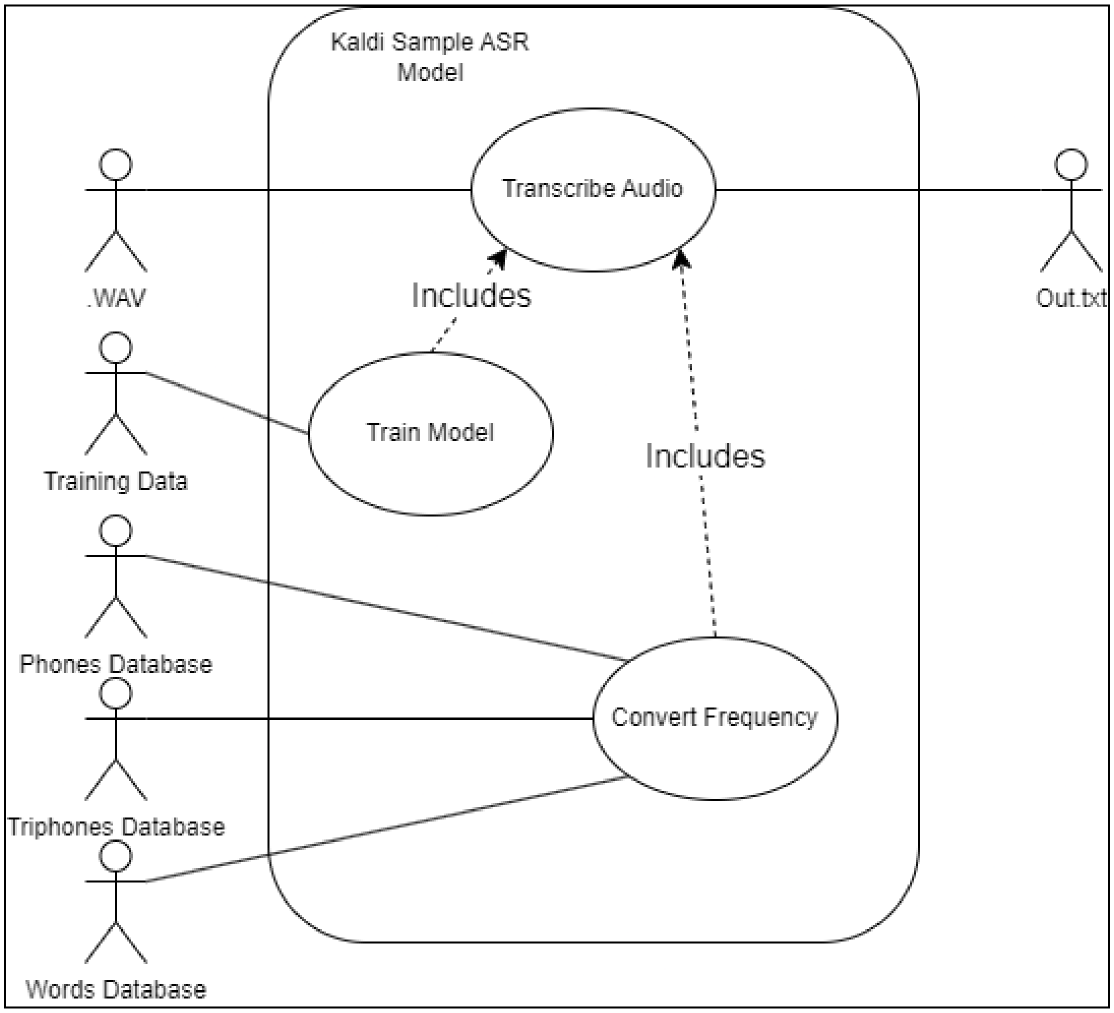


Figure 8: Kaldi ASR Toolkit Sample ASR Model Use Case Diagram V1.6

The use case diagram above displays how the users and other entities interact with the ASR model.

Use Cases:

Use Case ID: UC1

Use Case Name: Transcribe Audio
Goal: User obtains text transcription of audio file
Actors: User, Phones Database, Triphones Database, Words Database, Training Data
Pre:

- Model is properly installed and trained

Post:

- out.txt is stored in /kaldi/egs/mini_librispeech/s5/

Main Success Scenario:

Step	Actor Action	Step	System Reaction
1	Input .WAV	2	Divide into frames of 25 milliseconds, every 10 milliseconds, allowing for 3 frames to overlap
		3	Converts frames from time-change to frequency-change
		4	Calculates the changes in frequency between frames from up to 2 time derivations before and after
		5	Calculates the most probable phone based on the changes in frequency
		6	Builds triphones using the most probable phones
		7	Builds words using the most probable triphones
		8	Uses the previous two words to predict the following word
		9	Uses the language model to assemble the words into sentences
11	Receive out.txt	10	Transfers sentences to a text file

Exceptions (Alternative Scenarios of Failure):

- User inputs .FLAC
- User does not input anything

Add Reqs for Final Version

4.2 Internal Communications Detailed Design

The Kaldi ASR Toolkit is used for developing ASR models. The toolkit comes packaged with a sample ASR model that the user can access using the Ubuntu terminal. The user starts the process by entering the command, “python3 main.py file_name.wav”. Upon activation of the Python3 file “main.py”, the .WAV file (*Figure 9:1.1)* is processed by the ASR model (*Figure 9:2.1)*. Note that the ASR model only accepts .WAV files as input -- all other file types must be converted into a .WAV file.

The sample ASR model then inputs the .WAV files into the FFT (*Figure 9: 2.1.1*) . The FFT partitions the file into 25 millisecond frames using the formula [10n, 10n+25]. The audio data is then converted from change-over-time to change-over-frequency.

The MFC (*Figure 9: 2.1.2*) partitions the aforementioned frames into 12 segments. It also appends an additional “common” segment. The MFC proceeds to convert each of the 13 segments into MFCCs. The first order and second order derivatives of each MFCC are derived afterwards. Finally, a 39-dimensional array is created, where each dimension is indexed by time in 10 millisecond increments (e.g., n = 1 represents 10 milliseconds). The first set of thirteen dimensions represents the 13 MFCCs. The second set of thirteen dimensions represents the first order derivatives. The last set of thirteen dimensions represents the second order derivatives.

The system inputs one index from the MFC into the GMM (*Figure 9: 2.1.3*) which takes the 39 MFCC numbers and predicts the phone index. The GMM compares the 39 MFCC numbers to the phone database(*Figure 9: 4.1*) that has many points for each phone and returns the closest phones index.

Three phone index numbers are taken into the HMM (*Figure 9: 2.1.4*) and it calculates the closest triphone index based on the triphone database (*Figure 9: 5.1*). The HMM takes the triphone index numbers and compares them to the word database (*Figure 9: 6.1*) to find the closest word and returns the index for it.

The Language Model (*Figure 9: 2.1.5*) takes the two previous word index numbers and predicts the next word using the lexicon, which is created using the training data (*Figure 9: 7.1*). The Language Model then strings together the capitalized words from the lexicon and the lowercase words that are created from the phones. Finally, the sentences are written into “out.txt”, which is returned to the user.

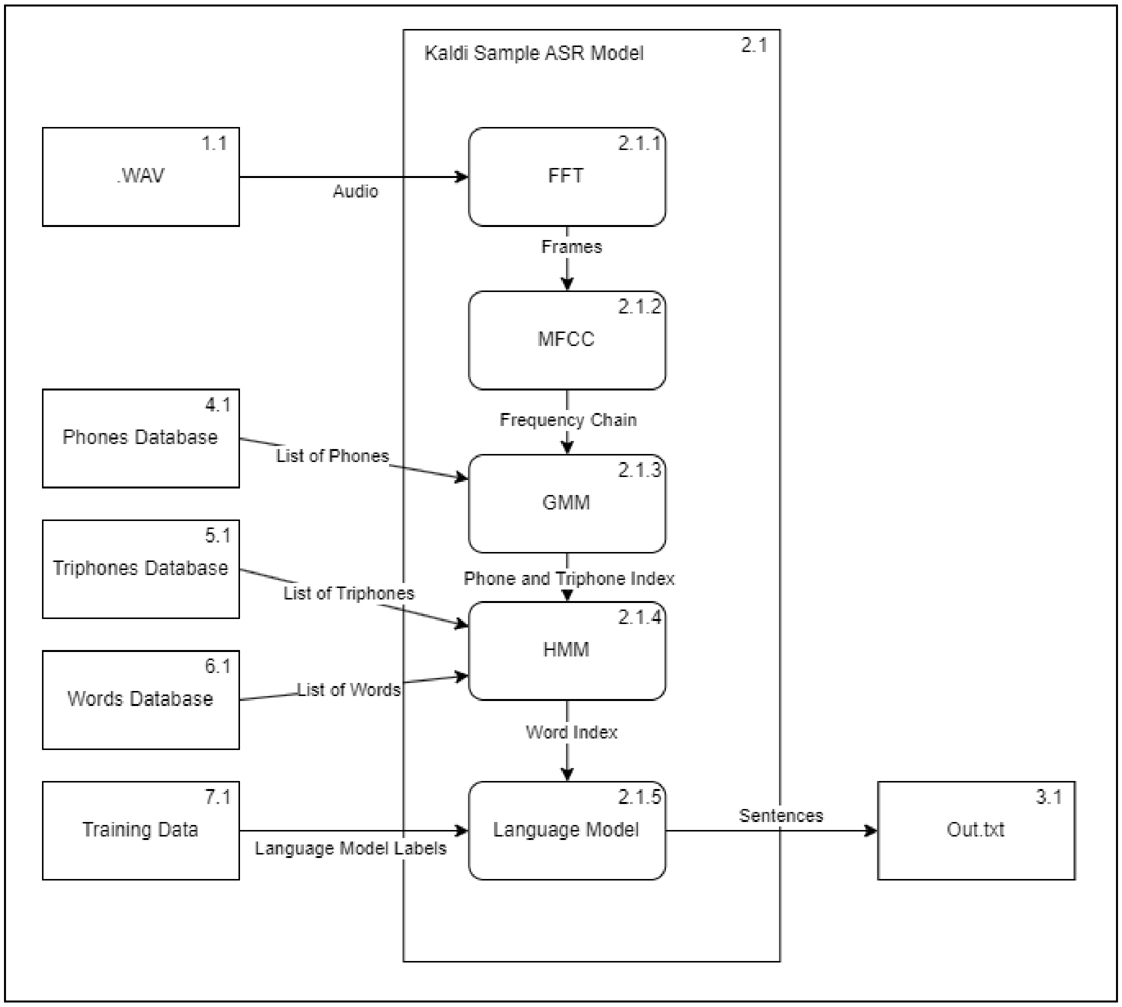


Figure 9: Kaldi ASR Toolkit Sample ASR Model Level 1 Data Flow Diagram V2.5

5 EXTERNAL INTERFACES

Outside the purview of the Kaldi Research Team is the development of the RTube web application. Once completed, the RTube web application shall integrate the data gathered by the Kaldi Research Team at the end of development.

5.1 Interface Architecture

The only interface between the ASR model developed by the RTube Kaldi Research Team and an external interface is the RTube web application being developed by the RTube NeMo Team. This interface shall allow for the ASR model generated by the RTube Kaldi Research Team into the RTube web application through the Python API Flask.

5.2 Interface Detailed Design

The Kaldi ASR Toolkit uses a command-line interface (CLI) to accept and generate input. It shall exclusively accept .WAV files as input and the only input a user shall have is selection of the input file. There are two commands of importance: “ffmpeg file_name.flac file_name.wav”, and “python3 main.py filename.wav”. The first command is used to convert .FLAC files to .WAV files. The second command is used to initiate the model and transcribe the audio. The first argument in the second command, “python3”, calls a Python file. The second argument, “main.py”, calls the main Python file that initiates the transcription process. Lastly, the final argument, “filename.wav,” is the chosen audio file to transcribe into text.