System Research & Test Plan

for

Kaldi ASR Team

Prepared by
Adam Gallub, Milan Haruyama, Tabitha O’Malley, David Serfaty, Tahmina Tisha

Name	**Date**	**Reason For Changes**	**Version**
All	10/18/2023	Write wrong information	V1.0
Tabitha	11/20/2023	Added in comments	V2.1
Tisha	11/27/2023	Sections 1, 1.2	V2.2
Tabitha	11/27/2023	Sections 3.3, 3.3.1, 3.3.2, 7, 8	V2.3
Milan	11/27/2023	Section 3.4; Editing all sections	V2.4
Tisha	11/28/2023	Section 3 Editing	V2.5
David	11/28/2023	Section 3.4, 8, 4.1.1, 4.1.2, 4.1.3, 1.2, 3.1.1, 3.3.1, 3.4.1, 3.4.2	V2.6
Tabitha	11/28/2023	Section 1.1 Editing/Reading All Sections Editing 3.3.2, 4, 5	V2.7
Adam	11/28/2023	Section 2, 3.2; Editing, Writing to the sections	V2.8
Milan	11/28/2023	Editing all sections	V2.9
Tabitha	11/29/2023	Section 3.1	V2.10
Tisha	11/29/2023	Section 3.; Edited the section	V2.11
Milan	12/03/2023	Editing all sections	V2.12
Milan	12/04/2023	Editing all sections	V2.13
David	02/10/2024	Sections 3.1, 6, 7	V3.1
Tabitha	02/10/2024	Sections 3.2, 3.3	V3.2
Tabitha	02/12/2024	Sections 3.3.2 , 3.3.3	V3.3
David	02/12/2024	Section 3	V3.4
David	02/15/2024	Section 4.1	V3.5
Tabitha	02/15/2024	Section 5	V3.6
Tabitha	02/15/2024	Section 3, 4, 5	V3.7
Tabitha	02/20/2024	Section 4, 8	V3.8
Milan	02/22/2024	Editing all sections	V3.9
Milan	02/23/2024	Editing all sections	V3.10
Milan	02/24/2024	Editing all sections	V3.11
Milan	02/25/2024	Editing all sections	V3.12

Table of Contents

1. Introduction

1.1 Purpose

This document is a plan for the system testing of the acoustic automatic speech recognition (ASR) model developed by the Kaldi ASR Team. It describes the testing strategies and approaches the team shall use to verify that the model meets the established requirements of the business prior to release.

1.2 Objectives

The objectives for the system testing of the ASR model seeks to ensure the following:

- The ASR model meets the requirements, specifications and the business rules.
- The ASR model supports the intended business functions and achieves the required standards.
- The ASR model satisfies the entrance criteria for User Acceptance Testing.

2. Functional Scope

The modules in the scope of testing for the ASR model are listed as follows:

- System Requirements Document Version 05: SRS_V5
- Section 3.1 of this document

3. Overall Strategy and Approach

3.1 Testing Strategy

System testing for the ASR model shall test the functionality of all possible inputs (valid or invalid), the preparation of valid inputs, the decoding of valid inputs, and all output steps.

3.1.1 Input Testing

Test Objective:
Test all possible inputs.

Technique:

- A MP3 file shall be used as input.
- No input shall be used.
- Multiple files of any type shall be used as input.
- All errors and exceptions thrown shall be compared.
- A WAV file shall be used as input.

Completion Criteria:
The system shall successfully execute the program upon reception of an input that meets all the system requirements. The system shall throw an error and/or exception upon reception of an input that does not meet any of the system requirements.

Special Consideration:
If the program terminates and does not throw an exception or error, this indicates that the model is not functioning properly, and shall require debugging and/or a complete reinstallation.

3.1.2 Preparation Testing

Test Objective:
Test the sequence of steps required to prepare the data preparation.

Technique:
- The audio shall be dithered to ensure there are no points with a frequency of zero or below.
- All frequencies between 8kHz -- 16kHz shall be filtered.
- The audio shall be equalized (equalization range TBD).
- The audio shall be split into 25-millisecond overlapping intervals.
- The audio shall be windowed into frames to prevent spectral leakage.
- A fast Fourier transform (FFT) shall be performed on the frames to convert them from the time domain to the frequency domain.
- The frequencies shall be converted into melodies using mel-filter bank.
- The frequencies shall be filtered with a frequency domain filter.
- A mel-vector shall be created.
- A discrete cosine transform (DCT) shall be performed on the mel-vector to build a mel-frequency cepstrum (MFC).
- The mel-frequency cepstral coefficients (MFCCs) created by the MFC shall be used as input for a Gaussian mixture model (GMM) to maximize the expected outputs.
- The output sequence of states and observations shall be used as input for a hidden Markov model (HMM) to find the most likely hidden state sequence.

Completion Criteria:
The data is prepared and moved to the decoding process and the most likely hidden state sequence to prepare for the decoder step.

3.1.3 Decoder Testing

Test Objective:
Test the sequence of steps required to find the probability of the sequence of states.

Technique:
- The same inputs as the HMM and all outputs from the HMM shall be used as input for the Viterbi decoder at the end of the preparation phase.
- The Viterbi decoder shall determine the probability of state changes based on given observations.

Completion Criteria:
The data is decoded and ready for the final conversion to text.

3.1.4 Output Testing

Test Objective:
Test the sequence of steps required to receive the decoded data and to write the sentences to a text file.

Technique:
- The decoded data from the Viterbi decoder shall be used as input for the weighted finite state transducer (WFST).
- The WFST shall convert the data to triphones using a HMM .
- The WFST shall convert the triphones to monophones using a context-dependency model.
- The WFST shall convert the monophones to words using the lexicon.
- The WFST shall compile the words into sentences using the language model.
- The transcribed sentences shall be written onto a text file.

Completion Criteria:
The system outputs a text file with the completed transcription.

3.2 System Testing Entrance Criteria

A functional ASR model created with the Kaldi ASR Toolkit is required to start system testing. An audio file with a verified transcription shall also be used to test the model.

3.3 Functional Testing

Functional testing of the ASR model shall include testing the input of the audio, preparation of the audio, the decoding of the audio, and the output of the text. A properly working model shall be able to transcribe the audio into text within an acceptable word error rate (WER), where the incorrect transcriptions do not detract from the understanding of the audio.

3.3.1 Input Testing

All possible inputs shall be tested to ensure that the system behaves appropriately. The system shall output a transcription upon receiving a valid input (a WAV file). Upon receiving an invalid input (e.g., a non-WAV file, multiple files of any type), the system shall throw an exception and/or error.

3.3.2 Preparation Testing

In order to produce a successful transcription, all valid audio inputs shall be prepared beforehand. Without any audio preparation, all attempted transcriptions shall be unsuccessful. The preparation process is described in the sequence of steps below.

The audio is dithered in order to prevent any frequencies from equaling zero or below. Since various logarithmic functions are used in later steps, it is important that the audio frequencies are greater than zero, as the logarithm of zero or below is indeterminate. To test for successful dithering, the system shall check that all frequencies are greater than zero. Upon failure of this test, the system shall restart this operation until the test is successful.

Once successfully dithered, the audio is then split into 25-millisecond sliding intervals, then windowed into frames to prevent spectral leakage. The specific time intervals the audio is split into affects the accuracy of the transcription, so it is important that a proper time interval is used. To test for successful windowing, the system shall check that the audio is split into the correct interval length. Upon failure of this test, the system shall restart this operation until the test is successful.

The windowed audio is then converted from the time domain (amplitude over time) to the frequency domain (amplitude over frequency). The new data is then converted into melody frequency numbers. The frequency's energy is also limited in order to vectorize the data and to compute its logarithm.

A discrete cosine transform (DST) is then used to symmetrize each vector with respect to the origin. The first and second derivatives of the data are then derived in order to account for any adjacent data. To test for successful symmetrization, the system shall test to see if each vector is symmetric with respect to the origin. Upon failure, the system shall restart this operation until the test is successful.

Next, a Gaussian mixture model is used to find the maximum expectation of the mel-frequency cepstral coefficient (MFCC) vector with respect to the phone distribution, as each vector can closely represent multiple phonemes.

Finally, a hidden Markov model predicts the most likely hidden state sequence.

3.3.3 Decoder Testing

Once successfully prepared, the data from the previous step shall be input into a Viterbi decoder in order to calculate the probability of the sequence of observables.

3.3.4 Output Testing

Once decoded, the data from the previous step is input into a weighted finite state transducer (WFST) in order to create sentences based on the probability of the sequence of observables. The sentences are created using a hidden Markov model, a context dependency model, a lexicon, and a language model. Once created, the sentences are written onto a text file and outputted from the system. To test for successful output, the system shall check that a non-empty text file is created. Upon failure of this test, the system shall restart this operation until the test is successful.

3.4 Suspension Criteria and Resumption Requirements

This section shall specify the criteria used to suspend all or a portion of the testing activities on the items associated with this test plan.

Testing shall be suspended if the following incidents halt further testing of the system and/or application. Once testing is halted, if any changes are made to the hardware, software, or database, it is the responsibility of the testing manager to determine whether the entire test plan or only a portion of it shall be re-executed.

Suspension Criteria	Resumption Criteria
The ASR model fails to compile or execute.	A new ASR model iteration shall be created, or the current iteration shall be reinstalled.
The Kaldi ASR Toolkit fails to compile or execute.	The Kaldi ASR Toolkit shall be reinstalled and reconfigured.

4. Execution Plan

The following section shall detail the test cases to be executed. The execution plan shall be compiled to ensure that all requirements are covered, and shall accommodate any necessary changes if testing is incomplete. The following test cases to be used for the current ASR model are arranged in a logical order based upon their interdependency.

Audio Files					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
1.1.1	The system shall only accept WAV files as input.	1.1	Non-WAV file	The system shall convert any non-WAV file to WAV before starting the transcription process.	P
1.1.2	The system shall convert other file types into WAV files using the FFMPEG Linux utility.				
Pre-Emphasis					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
2.1.1	The system shall dither the signals.	1.2	Valid input (WAV file)	The system shall add Gaussian white noise to the file to ensure all frequencies are greater than zero.	P
2.1.2	The system shall never let the signal be zero.				
2.2.1	The system shall filter the signal.	1.3	Dithered audio file	The system shall filter high amplitude data out of the signal.	P
2.2.2	The system shall use the formula x[*n*] - ax[*n*-1] to filter the data.				
2.2.3	The system shall define *n* as the sample.	1.4		The system shall define the variables to filter the audio signal.	P
2.2.4	The system shall define x[*n*] as the input signal at *n*.				
2.2.5	The system shall define a as a value between 0.95 and 0.97.				

Signal Framing

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
3.1.1	The system shall partition the audio into overlapping 25-millisecond frames.	1.5	Filtered audio file	The system shall split the audio signal into 25-millisecond overlapping frames.	P
3.1.2	The system shall obtain each frame in 10-millisecond sliding intervals starting from time equals 0.				
3.1.3	The system shall use the formula $[10n, 10n + 25]$ to obtain the closed time intervals of each frame (e.g., [0, 25], [10, 35], [20, 45], …).				
3.1.4	The system shall define the value *n* as any non-negative integer. (i.e., $\{n \in \mathbb{Z} : n \geq 0\}$)				

Windowing

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
4.1.1	The system shall window the audio.	1.6	Framed audio	The system shall window each frame to prevent spectral leakage.	P
4.1.2	The system shall use the formula $w[n] = 0.54 - 0.46cos(\frac{2\pi n}{N-1})$ to window the audio.				
4.1.3	The system shall define the variable *N* as the length of the window.				
4.1.4	The system shall define the variable *n* as *N*-1.				

Fast Fourier Transform (FFT)					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
5.1.1	The system shall use a FFT to convert the audio data within the frames from the time domain to the frequency domain.	1.7	Windowed audio frames	The system shall convert each frame from the time domain to the frequency domain.	P
5.1.2	The system shall use formula $X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{K}kn}$ to convert the audio from the time domain to the frequency domain.				
5.1.3	The system shall define the function x[n] as the windowed speech signal.				
5.1.4	The system shall define *N* is the length of the window.				
5.1.5	The system shall define the variable $K \geq N$.				
5.1.6	The system shall define the variable $j = \sqrt{-1}$.				
5.1.7	The system shall define the variable $k = \frac{\omega_k K}{2\pi}$.				

Mel-Filter Bank					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
6.1.1	The system shall convert the results of the FFT into mel numbers.	1.8	Time domain audio	The system shall convert the audio data into mel-frequency numbers.	P
6.1.2	The system shall use the formula $m = 2595 \cdot log_{10}(1 + \frac{f}{700})$ to obtain the mel numbers.				
	Limit Energy Frequency				
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
7.1.2	The system shall use the formula $a_m = \sum_{k=0}^{K/2+1} F_m[k]\|X[k]\|^2$ to limit the energy frequency of the audio file.				

Mel Vector

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
8.1.1	The system shall vectorize the data.	1.10	Filtered time domain audio, Mel Frequency numbers	The system shall vectorize the data.	P

Logarithm of the Vector

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
9.1.1	The system shall take the logarithm of the vector.	1.11	Audio Vector	The system shall take the logarithm of the vector.	P

Discrete Cosine Transform (DCT)

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
10.1.1	The system shall symmetrize the sequence with respect to the origin.	1.12	Vector	The system shall perform a discrete cosine transform on the vector.	P
10.1.2	The system shall use the formula $c[k] = \sum_{m=0}^{M-1} v[m]cos\,(\pi\,(m + 0.5)\,k/M)$ to symmetrize the sequence.				
10.1.3	The system shall define v[m] as the logarithm of the vector.				
10.1.4	The system shall define *m* as the melody number.				
10.1.5	The system shall define *M* as the number of dimensions from the vector.				
10.1.6	The system shall define *k* as $M - 1$.				

Mel-Frequency Cepstrum (MFC) & Mel-Frequency Cepstral Coefficients (MFCCs)					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
11.1.1	The system shall reduce the dimensions of the vector from the DCT using to 12.	1.13	Vector	The system shall reduce the dimensions of the vector from the DCT to 12.	
12.1.1	The system shall set the thirteenth MFCC to the short time energy of the speech signal.	1.14	Reduced Vector	The system shall derive the first- and second- order derivatives of the 13 MFCCs	P
12.1.2	The system shall use this formula to perform the derivative of the MFCC's, $c_{\Delta}[k, l] = \frac{\sum_{n=1}^{N} n(c[k, l + m] - c [k, l-}{2 \sum_{n=1}^{N} n^{2}}$.				
12.1.3	The system shall derive the first-order derivative of the 13 MFCCs.				
12.1.4	The system shall derive the second derivative of the 13 MFCCs.				
12.1.5	The system shall produce a 39-dimensional MFC represented as an MFCC feature vector.				
12.1.6	The system shall use the first set of thirteen dimensions (i.e., [0, 12]) to represent the thirteen MFCCs.				
12.1.7	The system shall use the second set of thirteen dimensions (i.e., [13, 25]) to represent the first-order derivatives of the MFCCs.				
12.1.8	The system shall use the third set of thirteen dimensions (i.e., [26, 38]) to represent the second-order derivatives of the MFCCs.				

Gaussian Mixture Model (GMM)					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
13.1.1	The system shall initialize three components: the mean of the component μk, the variance of the component Σk, the initial probability of the component πk.	1.15	First- and second-order derivatives of the vector.	The system shall compute the expectation maximization.	P
13.1.2	The systems shall compute the maximization of expectation of the MFCC feature vector matching up with a phoneme distribution.				
13.1.3	The systems shall use the formula $N(\mu_k, \Sigma_k) = \sum_{k=1}^{K} \pi_k N(x \mid \mu_k, \Sigma_k)$ to compute the maximization.				
13.1.4	The system shall compute ɣ(Znk) to find the expected values.				
13.1.5	The system shall use ɣ(Znk) to recalculate μk, Σk, and πk.				
13.1.6	The system shall use the recalculated values to maximize the value of ɣ(Znk).				
13.1.7	The system shall continue to calculate and recalculate as it gets closer and closer to the ideal value.				

Hidden Markov Model (HMM)					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
14.1.1	The system shall define the state space.	1.16	Expectation Maximization	The system shall compute the hidden state sequence.	
14.1.2	The system shall intake the observations from the GMM to define the observation space.				
14.1.3	The system shall intake the initial state transition probabilities πk from the GMM				
14.1.4	The system shall calculate the likelihood of an observation given a state k.				
14.1.5	The system shall calculate the most likely sequence of state changes.				
14.1.6	The system shall then use the output to retrain the HMM and perform steps 14.1.3 through 14.1.5 again until no further improvement.				
14.1.7	The system shall decode the most probable sequence of observable states.				
14.1.8	The system shall evaluate the model and then output the sequence of states and state transition probabilities that were calculated.				

Decoding Operations

Viterbi Decoder

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
15.1.1	The system shall use a Viterbi decoder to find the most probable hidden phoneme state sequence.				
15.1.2	The system shall use the information computed in the GMM and HMM to perform the calculations for the Viterbi decoder.				
15.1.3	The system shall gather the Transition Matrix from the HMM.				
15.1.4	The system shall gather the Emissions Matrix from the HMM.				
15.1.5	The system shall gather the State space from the HMM.				
15.1.6	The system shall gather the Array of initial probabilities from the GMM.				
15.1.7	The system shall gather the sequence of observation space from GMM.				
15.1.8	The system shall gather the Observation Space from GMM.				

Output					
Weighted Finite State Transducer (WFST)					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
16.1.1	The system shall use four WFST models for further processing.				
Hidden Markov Model (HMM)					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
17.1.1	The system shall input the output of the Viterbi decoder into a HMM to output triphones.				
Context-Dependency Model					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
18.1.1	The system shall input triphones to convert the triphone states into monophones.				
Lexicon					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
19.1.1	The system shall use the lexicon to convert monophones into words.				
Language Model					
Req #	**Requirement**	**Test ID**	**Input**	**Expected Behavior**	**P/F**
20.1.1	The system shall use the words from the lexicon out to score the next predicted word.				

Weighted Finite State Transducer (WFST)

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
21.1.1	The system shall combine the four aforementioned models into a single simplified model.				

Text File

Req #	Requirement	Test ID	Input	Expected Behavior	P/F
22.1.1	The system shall output a text file upon completion of all aforementioned operations.				
22.1.2	The system shall only write transcribed words into the aforementioned text file.				
22.2.1	The system shall not write punctuation or grammatical marks into the aforementioned text file.				
22.3.1	The system shall transcribe words found in the lexicon in uppercase.				
22.3.2	The system shall transcribe words not found in the lexicon in lowercase.				

5. Traceability Matrix & Defect Tracking

This section shall cover the importance of each requirement and steps to be taken to ensure proper function.

5.1 Traceability Matrix

5.1.1 Critical Requirements

Req. 3.1.1	The system shall partition the audio into overlapping 25-millisecond frames.
Req. 3.1.2	The system shall obtain each frame in 10-millisecond sliding intervals starting from time equals 0.
Req. 3.1.3	The system shall use the formula [10n, 10n+25] to obtain the closed time intervals of each frame (e.g., [0, 25], [10, 35], [20, 45], …).
Req. 3.1.4	The system shall define the value n as any non-negative integer (i.e., $\{n \in \mathbb{Z} : n \geq 0\}$).

Test Case: Check that the audio is properly split.

Req. 5.1.1	The system shall use a FFT to convert the audio data within the frames from the time-domain to the frequency domain.
Req. 5.1.2	The system shall use formula $X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{K}kn}$.
Req. 5.1.3	The system shall define x $[n]$ is the windowed speech signal.
Req. 5.1.4	The system shall define *N* is the length of the window.
Req. 5.1.5	The system shall define *K* as ≥ *N*.
Req. 5.1.6	The system shall define j as $\sqrt{-1}$.
Req. 5.1.7	The system shall define k as $\frac{\omega_k K}{2\pi}$.

Test Case: Check that the audio data is converted to amplitude over time.

Req. 6.1.1	The system shall convert the results of the FFT into mel numbers.
Req. 6.1.2	The system shall use the formula $m = 2595 \cdot log_{10}(1 + \frac{f}{700})$.

Test Case: Compare the number to mel numbers.

Req. 8.1.1	The system shall vector the data.

Test Case: Check to see if the data is in a vector format.

Req. 11.1.1	The system shall reduce the dimensions of the vector from the DCT using MFC to 12.
Req. 11.1.2	The system shall set the thirteenth MFCC to the short time energy of the speech signal.
Req. 11.1.3	The system shall use the formula $c_{\Delta}[k, l] \ = \ \frac{\sum_{n=1}^{N} n(c[k, l + m] - c [k, l-m])}{2 \sum_{n=1}^{N} n^{2}}$ to perform the derivative of the MFCC's,
Req. 11.1.4	The system shall derive the first-order derivative of the 13 MFCCs.
Req. 11.1.5	The system shall derive the second-order derivative of the 13 MFCCs.
Req. 11.1.6	The system shall produce a 39-dimensional MFC represented as an MFCC feature vector.
Req. 11.1.7	The system shall use the first set of thirteen dimensions (i.e., [0, 12]) to represent the thirteen MFCCs.
Req. 11.1.8	The system shall use the second set of thirteen dimensions (i.e., [13, 25]) to represent the first-order derivatives of the MFCCs.
Req. 11.1.9	The system shall use the third set of thirteen dimensions (i.e., [26, 38]) to represent the second-order derivatives of the MFCCs.

Test Case: Check if the output is a 39-dimension MFCC vector.

Req. 12.1.1	The system shall initialize three components: The mean of the component μk, the variance of the component Σk, the initial probability of the component πk.
Req. 12.1.2	The systems shall compute the maximization of expectation of the MFCC feature vector matching up with a phoneme distribution.
Req. 12.1.3	The systems shall use the formula $N(\mu_k, \Sigma_k) = \sum_{k=1}^{K} \pi_k N(x \mid \mu_k, \Sigma_k)$ to compute the maximization.
Req. 12.1.4	The system shall compute ɣ(Znk) find the expected values.
Req. 12.1.5	The system shall use ɣ(Znk) to recalculate μk, Σk, and πk.
Req. 12.1.6	The system shall use the recalculated values to maximize the value of ɣ(Znk).
Req. 12.1.7	The system shall continue to calculate and recalculate as it gets closer and closer to the ideal value.

Test Case: Check to see if the maximization has been reached.

Req. 13.1.1	The system shall define the state space.
Req. 13.1.2	The system shall intake the observations from the GMM to define the observation space.
Req. 13.1.3	The system shall intake the initial state transition probabilities πk from the GMM
Req. 13.1.4	The system shall calculate the likelihood of an observation given a state k.
Req. 13.1.5	The system shall calculate the most likely sequence of state changes.
Req. 13.1.6	The system shall then use the output to retrain the HMM and perform steps 14.1.3 through 14.1.5 again until no further improvement.
Req. 13.1.7	The system shall decode the hidden state sequence.

Test Case: Unverifiable because of hidden states.

Req. 15.1.1	The system shall use a Viterbi decoder to find the probability of the sequence of events
Req. 15.1.2	The system shall use the information computed in the GMM and HMM to perform the calculations for the Viterbi decoder.
Req. 15.1.3	The system shall gather the Transition Matrix from the HMM.
Req. 15.1.4	The system shall gather the Emissions Matrix from the HMM.
Req. 15.1.5	The system shall gather the State space from the HMM.
Req. 15.1.6	The system shall gather the Array of initial probabilities from the GMM.
Req. 15.1.7	The system shall gather the sequence of observation space from GMM.
Req. 15.1.8	The system shall gather the Observation Space from GMM.
Req. 15.1.9	The system shall evaluate the model and then output the sequence of states and state transition probabilities that were calculated.

Test Case: Check if the output is the probability of the sequence of observable events.

Req. 16.1.1	The system shall use four WFST models for further processing.
Req. 17.1.1	The system shall input the output of the Viterbi decoder into a HMM to output triphones.
Req. 18.1.1	The system shall input triphones to convert the triphone states into monophones.
Req. 19.1.1	The system shall use the lexicon to convert monophones into words.
Req. 20.1.1	The system shall use the words from the lexicon out to score the next predicted word.
Req. 21.1.1	The system shall combine the four aforementioned models into a single simplified model.

Test Case: Check if the sentences match the audio recording.

Req. 22.1.1	The system shall output a text file upon completion of all aforementioned operations.
Req. 22.1.2	The system shall only write transcribed words into the aforementioned text file.

Test Case: Check if the sentences are written to the text file.

5.1.2 Medium Requirements

Req. 2.1.1	The system shall dither the signals.
Req. 2.1.2	The system shall never let the signal be zero.

Test Case: Check if the signal is ever zero or less.

Req. 2.2.1	The system shall filter the signal.
Req. 2.2.2	The system shall use the formula x[n] - ax[n-1] to filter the data.
Req. 2.2.3	The system shall define the n as the sample.
Req. 2.2.4	The system shall define x[n] as the input signal at n.
Req. 2.2.5	The system shall define a as a value between 0.95 - 0.97.

Test Case: Check that there is an accommodation for high frequencies.

Req. 4.1.1	The system shall window the audio.
Req. 4.1.2	The system shall use the formula $w_{Hamm}[n] = 0.54 - 0.46\,cos(\frac{2\pi n}{N-1})$ for the window.
Req. 4.1.3	The system shall define N as the length of the window.
Req. 4.1.4	The system shall define n as N-1.

Test Case: Check that the audio is properly windowed.

Req. 9.1.1	The system shall take the log of the vector.

Test Case: Check if the logarithm was taken.

Req. 10.1.1	The system shall make the sequence symmetric with respect to the origin.
Req. 10.1.2	The system shall use the formula $c[k] = \sum_{m=0}^{M-1} v[m]\,cos\,(\pi\,(m + 0.5)\,k/M)$.
Req. 10.1.3	The system shall define v[m] is the log of the vector.
Req. 10.1.4	The system shall define m as the melody number.
Req. 10.1.5	The system shall define M as the number of dimensions from the vector.
Req. 10.1.6	The system shall define k as M-1.

Test Case: Check if the DCT was taken.

5.1.3 Low Requirements

Req. 1.1.1	The system shall only accept WAV files as input.
Req. 1.1.2	The system shall convert other file types into WAV files using the FFMPEG Linux utility.

Test Case: Check if the system accepts and receives a file.

Req. 22.2.1	The system shall not write punctuation or grammatical marks into the aforementioned text file.
Req. 22.3.1	The system shall transcribe words found in the lexicon in uppercase.
Req. 22.3.2	The system shall transcribe words not found in the lexicon in lowercase.

Test Case: Check the words and grammar based off of the lexicon.

5.2 Defect Severity Definitions

Critical	The defect causes a catastrophic or severe error that results in major problems and the functionality rendered is unavailable to the user. A manual procedure cannot be either implemented or a high effort is required to remedy the defect. Examples of a critical defect are as follows: ● System abends ● Data cannot flow through a business function/lifecycle ● Data is corrupted or cannot post to the database
Medium	The defect does not seriously impair system function and can be categorized as a medium Defect. A manual procedure requiring medium effort can be implemented to remedy the defect. Examples of a medium defect are as follows: ● Form navigation is incorrect ● Field labels are not consistent with global terminology
Low	The defect is cosmetic or has little to no impact on system functionality. A manual procedure requiring low effort can be implemented to remedy the defect. Examples of a low defect are as follows: ● Repositioning of fields on screens ● Text font on reports is incorrect

6. Environment

Since the Kaldi ASR Toolkit was designed to be run on Linux distributions, the testing environment shall be the Ubuntu command line via the Windows Subsystem for Linux (WSL).

7. Assumptions

- The dataset is sufficient to train an ASR model.
- The dataset used to train the ASR model shall be in General American English.
- The dataset used to train the ASR model shall include ATC vernacular.
- The text transcribed by the ASR model shall be based on General American spelling conventions (e.g., “color” versus “colour”).
- All testing performed on the ASR model shall be done under the assumption that the system can compile, train, and run the model.
- All tests shall be run to completion without interruption or changes partway through.
- The system shall retrain the model if the accuracy or runtime do not meet the requirements.

8. Risks and Contingencies

ID	Risk	Impact	Contingency Plan
1	Unable to decrease runtime.	Low	Upgrade GPU or install additional one.
2	Unable to increase accuracy.	Low	Retrain model.
3	Kaldi ASR Toolkit termination error.	High	Reinstall the Kaldi ASR Toolkit.
4	Unable to improve the model.	High	Retrain model.

9. Appendix: 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, United States.
FFmpeg	***Linux utility;*** a portable open-source utility that allows users to decode, encode, transcode, multiplex, demultiplex, stream, filter, and play most human- or machine-made multimedia.
FFT	***Fast Fourier Transform;*** algorithm used to obtain the spectrum or frequency content of a signal.
General American English	The most spoken variety of the English language in the United States.
GMM	***Gaussian Mixture Model;*** used to calculate the distance between the MFC feature vector and the HMM state.
HMM	***Hidden Markov Model;*** used to find the state locations of the phonemes..
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 is understood by the language model.
MFC	***Mel-Frequency Cepstrum;*** a representation of the short-term power spectrum of a sound
MFCCs	***Mel-Frequency Cepstral Coefficients;*** the coefficients that a MFC is comprised of
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
WSL	***Window Subsystem for Linux;*** allows users to run a GNU/Linux environment directly on Windows without the overhead of running the environment through a virtual machine.