A SPEAKER-DEPENDENT ISOLATED-PHRASE REAL TIME AUTOMATIC SPEECH RECOGNITION SYSTEM USING HIDDEN MARKOV MODELS

FNU Sidharth[1]

[1]Department of Electrical Engineering, University of Washington, Seattle

ABSTRACT

Automatic Speech Recognition (ASR) systems are essential for enabling natural human-machine interactions. This project presents a speaker-dependent, isolated-phrase ASR system designed to recognize a specific user's set of predefined phrases. The system utilizes Hidden Markov Models (HMMs) with a single Gaussian distribution per state and a diagonal covariance matrix to model acoustic features efficiently. A continuous monitoring approach is employed, with a speech/silence detector identifying speech segments. Upon detecting the wake-up phrase, ”Odessa,” the system expects the next segment to contain one of the remaining phrases from the vocabulary. Experimental results demonstrate the system's effectiveness, achieving a mean accuracy of 98.3% across 5-fold cross-validation.

Index Terms— Automatic Speech Recognition, Hidden Markov Models, Speaker-Dependent, Isolated-Phrase, Gaussian Distribution, Diagonal Covariance Matrix

1. INTRODUCTION

Automatic Speech Recognition (ASR) systems have become ubiquitous in modern technology, powering virtual assistants, transcription services, and accessibility features. These systems enable users to interact with devices naturally, using spoken language, enhancing user experience and accessibility.

Numerous approaches have been proposed to improve ASR systems' accuracy and efficiency. Mohamed et al.[1] explored acoustic modeling using deep belief networks, showing promising results in speech recognition tasks. Graves and Schmidhuber[2] introduced bidirectional LSTM networks for framewise phoneme classification, demonstrating the effectiveness of recurrent neural networks in ASR. Hinton et al.[3] discussed the use of deep neural networks for acoustic modeling in speech recognition, highlighting the shared views of multiple research groups on this approach. Rabiner[4] provided a seminal tutorial on hidden Markov models (HMMs) and their applications in speech recognition, laying the foundation for many modern ASR systems.

In this project, I present a speaker-dependent, isolated-phrase ASR system designed to recognize a specific user's set of predefined phrases (”Odessa”, ”Play music”, ”Stop music”, ”What time is it”, ”Turn on the lights”, ”Turn off the lights”). The system employs HMMs with a single Gaussian distribution per state and a diagonal covariance matrix, efficiently modeling acoustic features. A key feature of my system is its continuous monitoring approach, utilizing a speech/silence detector to identify speech segments. Upon detecting the wake-up phrase, ”Odessa,” the system expects the next segment to contain one of the remaining phrases from the vocabulary.

2. DATASET

The dataset used in this study comprises recordings of six distinct phrases, including ”Odessa,” ”Turn on the lights,” ”Turn off the lights,” ”What time is it,” ”Play music,” and ”Stop music.” Each phrase was recorded by a single speaker in various acoustic environments to capture real-world variability. A total of 20 samples were recorded for each phrase, resulting in a dataset of 120 audio samples. The recordings were made at a sampling rate of 44.1 kHz using Audacity recording software.

To simulate different acoustic conditions, recordings were made in environments with varying levels of room reverberations. This variability introduces challenges that an ASR system may encounter in real-world scenarios, making the dataset suitable for testing the robustness of the system.

3. METHODOLOGY

The general pipeline used for the ASR system is shown in Fig.1

The system begins by continuously monitoring the acoustic environment using the Voice Activity Detector (VAD). When speech is detected, the VAD identifies the endpoints of the speech segment, which are then used to calculate the Mel-frequency cepstral coefficients (MFCC) features. These MFCC features serve as the observations for the Gaussian-based Hidden Markov Models (HMMs) used in the system.

Each HMM corresponds to a specific phrase in the vocabulary, such as ”Odessa,” ”Turn on the lights,” ”Turn off the lights,” ”What time is it,” ”Play music,” and ”Stop music.” Upon detecting the wake-up word ”Odessa” using the HMM for that phrase, the system then waits for the next subsequent word. Based on the recognized phrase, the system performs

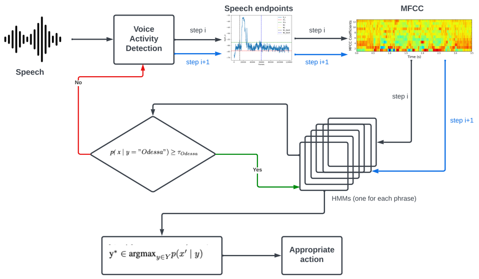
Speech
Voice Activity Detection
step i
step i+1
Speech endpoints
step i
step i+1
MFCC
MFCC Coefficients
Time (s)
step i
step i+1
No
$p(\, x \mid y = \text{"}Odessa\text{"}) \geq \tau_{Odessa}$
Yes
HMMs (one for each phrase)
$\mathbf{y^*} \in \mathbf{argmax}_{y \in Y}\, p(x' \mid y)$
Appropriate action

Fig. 1. General pipeline for the phrase detection

the appropriate action, such as turning on or off lights, playing or stopping music, etc.

This design enables the system to respond to spoken commands in a natural and intuitive manner, enhancing user interaction with the system. The following subsections provide a detailed explanation of the mathematical aspects of these components.

3.1. Voice Activity Detector (VAD)

For VAD, I used Rabiner-Sambur algorithm for speech end point detection[5]. The algorithm uses short-time energy and short-time zero-crossings for end point detection. Short-time energy is defined by the equation (1)

$$E_s(n) = \sum_{i \in [n-5, n+5]} \left\| s(n+i) \right\|^2 \tag{1}$$

Where *s* is the speech signal and n is in ms. In addition to the energy, a zero-crossing measure is used.

$$ZC_s(n) = \frac{1}{L} \sum_{i \in [n-5, n+5]} \left\| f(n+i) \right\|^2 \tag{2}$$

Where L is the number of samples in 10ms window and

$$f(n+i) = \frac{sgn(s(n+i)) - sgn(s(n+i-1))}{2}$$

where *sgn(.)* is the signum function. The algorithm is shown in Algorithm 1.

An example of speech endpoints obtained after using this algorithm is shown in the Fig.2

3.2. MFCC calculation

Mel-frequency cepstral coefficients (MFCCs) are widely used in speech and audio signal processing due to their effectiveness in capturing key characteristics of the human voice. The calculation of MFCCs involves several steps. First, the speech signal is divided into short frames of 25 milliseconds long with 10ms strides, to capture the spectral characteristics over time. Each frame is then windowed using a Hamming window to reduce spectral leakage. Next, the Fast Fourier Transform (FFT) is applied to each windowed frame to obtain the power spectrum.

Algorithm 1 Rabiner-Sambur Endpoint Detection

```
**Input**: $audio\_segment$ - the segment of audio to analyze
**Output**: $start\_point$, $end\_point$ - the estimated start and end points of speech

Compute $E_{sin}$ for $n$ ranging over the segment of audio
Compute $IMX$ and $IMN$, the max and min of $E_{sin}$
Set $ITL = \min(0.003(IMX, IMN), 4IMN)$ and $ITU = ITL * 5$
Set $IZCT = \min(IF, \mu_{IZC} + 2\sigma_{IZC})$, where $IF = 25$, $\mu_{IZC}$ is the mean, and $\sigma_{IZC}$ is the standard deviation of the zero crossings during silence at the beginning of the speech signal
Search from the beginning towards the center until the first frame where $E_{sin}$ goes above $ITL$ and then above $ITU$ without first going below $ITL$. Call this frame $N1$.
Similarly, for the endpoint, start from the end of the signal and follow the same procedure as in the previous step, but searching towards the center.
Search starting at frame $N1$ (the start point) to frame $N2$ (the end point), and if the number of ZCT frames is 3 or more, then the start and end points are changed to the first and last frame, respectively, in that 25 frame interval whose threshold exceeds $IZCT$; otherwise, leave the segment points alone.
```

Following this, the power spectrum is passed through a bank of Mel-spaced triangular filters. The output of these filters represents the energy distribution across different frequency bands. Taking the logarithm of these filterbank energies helps to approximate the nonlinear human perception of sound intensity. Finally, a discrete cosine transform (DCT) is applied to the log filterbank energies to decorrelate the features and reduce dimensionality. I then used the first 13 coefficients as MFCC.

In addition to MFCCs, I also incorporated delta coefficients in my feature set to capture the temporal dynamics of the speech signal. Delta coefficients represent the rate of change of MFCCs between adjacent frames and provide information about the speech signal's dynamics over time. The delta coefficients are calculated as follows

$$\Delta_c(t) = \frac{\sum_{n=1}^{N} n \cdot (c(t+n) - c(t-n))}{2 \cdot \sum_{n=1}^{N} n^2} \tag{3}$$

where c is the MFCC coefficient at time t and N = 2 is the number of frames on each side of the target frame used for regression. Hence I get a feature representation of the speech signal which is of the dimension (26 × T) (13 MFCC + 13 delta).

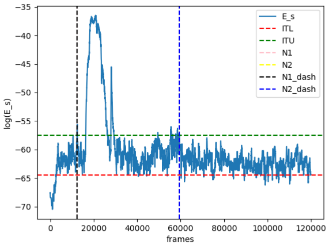

Fig. 2. The black vertical dotted line represents (N1) which is the start point of the speech signal and the blue dotted line (N2) represents the end point of the speech signal.

4. GAUSSIAN - HIDDEN MARKOV MODELS (HMM)

In my work, I employed a single Gaussian Hidden Markov Model (HMM) with 10 states to model the speech signal. Each state in the HMM corresponds to a different phoneme or sound unit in the speech signal. The emission probabilities of the HMM, which represent the likelihood of observing a particular feature vector given a state, were modeled using Gaussian distributions. Specifically, I assumed that the observations within each state were generated from a Gaussian distribution with a diagonal covariance matrix

Let **N** be the number of states in the HMM (in this ase, **N** = 10), **m** be the dimensonality of the feature vectors (26 × T) and T be the length of the observation sequence. The parameters of the HMM include

- Initial state probabilities $\boldsymbol{\pi} = [\pi_1, \pi_2, ..., \pi_N]$, where π_i represents the probability of starting in state i.
- State transition probabilities $\boldsymbol{A} = [a_{ij}]$, where a_{ij} represents the probability of transitioning from state i to state j.
- Mean vectors $\boldsymbol{\mu} = [\mu_1, \mu_2, ..., \mu_N]$ and covariance matrices $\boldsymbol{\Sigma} = [\Sigma_1, \Sigma_2, ..., \Sigma_N]$ of the Gaussian emission distributions.

The Gaussian emission probability density function (pdf) for state i is given by:

$$b_i(o_t) = \frac{1}{(2\pi)^{M/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(o_t - \mu_i)^T \Sigma_i^{-1}(o_t - \mu_i)\right) \tag{4}$$

where o_t is the observed feature vector at time t, μ_i is the mean vector of state i, Σ_i is the covariance matrix of state i, and $|\Sigma_i|$ denotes the determinant of Σ_i.

4.1. Forward and Backward Algorithm

$\alpha_t(i)$ is the forward variable, representing the probability of being in state i at time t and observing the partial sequence $o_1, o_2, ..., o_t$.

Forward Initialization: $\alpha_1(i) = \pi_i \cdot N(o_1 \mid \mu_i, \Sigma_i)$ for $i = 1, \ldots, N$.

Forward Recursion: $\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) \cdot a_{ij} \cdot N(o_{t+1} \mid \mu_j, \Sigma_j)$ for $j = 1, \ldots, N$.

$\beta_t(i)$ is the backward variable, representing the probability of observing the partial sequence $o_{t+1}, o_{t+2}, ..., o_T$ starting from state i at time t.

Backward Initialization: $\beta_T(i) = 1$ for $i = 1, \ldots, N$.

Backward Recursion: $\beta_t(i) = \sum_{j=1}^{N} a_{ij} \cdot N(o_{t+1} \mid \mu_j, \Sigma_j) \cdot \beta_{t+1}(j)$ for $t = T-1, T-2, \ldots, 1$.

4.2. Gamma (State Occupation Probabilities) and Xi (State Transition Probabilities)

$\gamma_t(i)$ is the probability of being in state i at time t given the observed sequence.

$\gamma_t(i) = \frac{\alpha_t(i) \cdot \beta_t(i)}{P(O|\lambda)}$, where $P(O \mid \lambda)$ is the probability of the observed sequence under the current model.

$\xi_t(i, j)$ is the probability of transitioning from state i to state j at time t given the observed sequence.

$\xi_t(i, j) = \frac{\alpha_t(i) \cdot a_{ij} \cdot N(o_{t+1}|\mu_j, \Sigma_j) \cdot \beta_{t+1}(j)}{P(O|\lambda)}$.

4.3. EM Algorithm

E-step: Calculate $\gamma_t(i)$ and $\xi_t(i, j)$ for all t and i, j.

M-step: Update the model parameters:

- $\pi_i = \gamma_1(i)$ (initial state probabilities).
- $a_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$ (state transition probabilities).
- $\mu_i = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot o_t}{\sum_{t=1}^{T} \gamma_t(i)}$ (mean of Gaussian distribution for state i).
- $\Sigma_i = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot (o_t - \mu_i) \cdot (o_t - \mu_i)^T}{\sum_{t=1}^{T} \gamma_t(i)}$ (covariance matrix for state i).

This approach was used to train the Hidden Markov Models (HMMs).

5. RESULTS

In my experiment, I extracted a 26-dimensional global mean and 26-dimensional global variance for each given phrase. From these values, I created a 26 x 26 dimensional diagonal covariance matrix for each phrase. Each phrase was modeled using 10 states, where each state utilized the global mean plus $\frac{1}{8}^{th}$ of the global variance, with the same global variance used for each state within a phrase. I trained six different Hidden Markov Models (HMMs), each corresponding to one of the six different phrases. During the final prediction phase, I assigned the predicted class based on the HMM with the maximum log-likelihood for a given input sequence. To evaluate the performance of my Hidden Markov Models (HMMs), I conducted an experiment using a 5-fold cross-validation strategy with an 80-20 split. The results are shown below

Table 1. Training and Validation accuracies for Fold 1. The values are normalized to 0-1

Phrase	Train	Validation
Odessa	1.0	1.0
Play music	1.0	1.0
Turn on the lights	1.0	1.0
Turn off the lights	1.0	1.0
Stop music	1.0	1.0
What time is it	1.0	1.0

Table 2. Training and Validation accuracies for Fold 2. The values are normalized to 0-1

Phrase	Train	Validation
Odessa	1.0	1.0
Play music	1.0	1.0
Turn on the lights	1.0	1.0
Turn off the lights	1.0	0.75
Stop music	1.0	1.0
What time is it	1.0	1.0

Table 3. Training and Validation accuracies for Fold 3. The values are normalized to 0-1

Phrase	Train	Validation
Odessa	1.0	1.0
Play music	1.0	1.0
Turn on the lights	1.0	1.0
Turn off the lights	1.0	1.0
Stop music	1.0	1.0
What time is it	1.0	1.0

6. CONCLUSION AND CHALLENGES

In this study, I presented a method for training Hidden Markov Models (HMMs) to recognize spoken phrases. By utilizing a 26-dimensional global mean and variance representation, I constructed diagonal covariance matrices for each phrase. Each phrase was modeled using 10 states, with each state incorporating the global mean and 1/8th of the variance. my approach resulted in 6 different HMMs trained to recognize six distinct phrases. For the final prediction, I employed the HMM with the maximum log likelihood as the classification model.

Several challenges were encountered during the experimentation phase. One significant issue was the occurrence of numerical problems, particularly with underflow and overflow of values. To address this, I converted all calculations to the log domain. Additionally, I implemented specific functions, such as logaddexp, to manage numerical stability and ensure accurate computation of probabilities.

Table 4. Training and Validation accuracies for Fold 4. The values are normalized to 0-1

Phrase	Train	Validation
Odessa	1.0	1.0
Play music	1.0	1.0
Turn on the lights	1.0	0.75
Turn off the lights	1.0	1.0
Stop music	1.0	1.0
What time is it	1.0	1.0

Table 5. Training and Validation accuracies for Fold 5. The values are normalized to 0-1

Phrase	Train	Validation
Odessa	1.0	1.0
Play music	1.0	1.0
Turn on the lights	1.0	1.0
Turn off the lights	1.0	1.0
Stop music	1.0	1.0
What time is it	1.0	1.0

7. REFERENCES

[1] Abdel-rahman Mohamed, George E. Dahl, and Geoffrey Hinton, “Acoustic modeling using deep belief networks,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[2] Alex Graves and Jürgen Schmidhuber, “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” *Neural Networks*, vol. 18, no. 5, pp. 602–610, July 2005.

[3] Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[4] L.R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[5] L. R. Rabiner and M. R. Sambur, “An algorithm for determining the endpoints of isolated utterances,” *The Bell System Technical Journal*, vol. 54, no. 2, pp. 297–315, 1975.