ASR with Real-Time Speech Processing

June 5, 2024

This document details the development of our Automatic Speech Recognition (ASR) system designed to process real-time audio input for recognizing specific commands. The system employs Mel-Frequency Cepstral Coefficients (MFCC) for feature extraction and Gaussian Mixture Model (GMM) based Hidden Markov Models (HMMs) for sequence modeling. A real-time component detects and responds to commands upon recognizing the wake word "Odessa".

1 Introduction

This project aims to develop an ASR system that can recognize specific voice commands and execute corresponding actions in real-time. The system uses MFCC for feature extraction from audio signals and GMM based HMMs for modeling the temporal sequences of these features.

2 Data Loading and Preprocessing

Audio files are loaded from the file system, with each command category having 30 distinct set of files. The audio data is loaded using the `librosa` library, and the filenames are organized based on a specific naming pattern.

3 Feature Extraction

MFCC features and their delta features are computed for each audio signal. These features are then concatenated to form the final feature set used for training the HMM.

4 Gaussian Mixture Model (GMM) based HMM

In many real-world applications, the emission probabilities are better modeled by Gaussian Mixture Models (GMMs) instead of single Gaussian distributions. A GMM can capture the multimodal nature of the data by combining several Gaussian components.

4.1 Components of GMM-HMM

1. **States (S):** As in the standard HMM, we have a finite set of states $\{s_1, s_2, \ldots, s_N\}$.
2. **Observations (O):** The observations are generated from a mixture of Gaussians, providing a more flexible and accurate representation of the data distribution.
3. **Initial State Probabilities (π):** The initial state distribution remains the same.
4. **Transition Probabilities (A):** The state transition probabilities remain unchanged.

5. **Emission Probabilities (B):** Each state's emission probability is modeled as a GMM. A GMM is defined by a set of parameters $\{w_k, \mu_k, \Sigma_k\}$ for each component k, where w_k are the mixture weights, μ_k are the means, and Σ_k are the covariance matrices.

$$b_i(o_t) = \sum_{k=1}^{M} w_{ik} \mathcal{N}(o_t \mid \mu_{ik}, \Sigma_{ik})$$

where M is the number of Gaussian components in the mixture.

4.2 Training GMM-HMM

Training a GMM-HMM involves estimating the parameters $\lambda = (\pi, A, B)$ using the EM algorithm. The process is similar to training a standard HMM but requires additional steps to update the GMM parameters.

4.2.1 Expectation Step

In the E-step, we calculate the responsibilities of each Gaussian component for each observation. This involves computing the posterior probabilities of the states and the responsibilities of each mixture component.

$$\gamma_t(i) = P(q_t = s_i \mid O, \lambda)$$

$$\gamma_t(i,k) = \frac{w_{ik} \mathcal{N}(o_t \mid \mu_{ik}, \Sigma_{ik})}{b_i(o_t)}$$

4.2.2 Maximization Step

In the M-step, we update the parameters to maximize the expected log-likelihood of the data. This includes updating the transition probabilities, the mixture weights, the means, and the covariances.

$$w_{ik} = \frac{\sum_{t=1}^{T} \gamma_t(i,k)}{\sum_{t=1}^{T} \gamma_t(i)}$$

$$\mu_{ik} = \frac{\sum_{t=1}^{T} \gamma_t(i,k) o_t}{\sum_{t=1}^{T} \gamma_t(i,k)}$$

$$\Sigma_{ik} = \frac{\sum_{t=1}^{T} \gamma_t(i,k)(o_t - \mu_{ik})(o_t - \mu_{ik})^T}{\sum_{t=1}^{T} \gamma_t(i,k)}$$

In our ASR, GMM-HMMs have shown to improve performance in terms of likelihood and classification accuracy over traditional HMMs.

5 Speech Detection

The speech detection process utilizes a state machine with states including 'silence', 'possible_start', 'speech', and 'possible_end'. The transitions between these states are governed by the energy thresholds ITL and ITU, and the zero-crossing threshold $IZCT$. Here's a detailed explanation of how each segment is identified and validated:

5.1 Start of Speech Detection

1. **Initial Search**: The algorithm starts at the beginning of the speech signal and searches toward the center. It looks for the first frame where the energy $E(n)$ first exceeds ITL and then goes above ITU without dropping below ITL again. This identified frame is denoted as $N1$.

2. **Refinement Using Zero-Crossings**: Starting from frame $N1$, the algorithm checks 25 frames backwards (covering a 250 ms interval assuming a 10 ms frame size). Within this interval, it counts the frames where the zero-crossing rate exceeds $IZCT$.

   - If three or more frames exceed $IZCT$, the starting point $N1$ is adjusted to $N1'$, which is the first frame in this interval where $IZCT$ is exceeded.
   - If fewer than three frames exceed $IZCT$, the starting point $N1$ remains unchanged.

5.2 End of Speech Detection

1. **Initial Detection**: From the state 'speech', when the energy falls below ITU, the system transitions to 'possible_end'. It then looks for the energy to drop below ITL, marking a potential end of speech at frame $N2$.

2. **Refinement Using Zero-Crossings**: Starting from frame $N2$, the algorithm examines the next 25 frames (again, a 250 ms interval).

   - It counts how many of these frames have a zero-crossing rate that exceeds $IZCT$.
   - If the count is three or more, $N2$ is adjusted to $N2'$, which is the first frame in this interval where the zero-crossing threshold is exceeded.
   - If fewer than three frames exceed $IZCT$, $N2$ remains unchanged as the endpoint.

5.3 Validation of Detected Speech Segments

Each detected speech segment, defined by start $N1$ or $N1'$ and end $N2$ or $N2'$, is validated based on its duration and peak amplitude:

- The duration of the segment must be at least 75 ms.
- The peak amplitude within the segment must exceed a predefined threshold k_4 which we set as 0.05.

Only segments that meet both criteria are considered valid speech and are subsequently logged.

5.4 Real-Time Operation

The system operates in an infinite loop, continuously capturing and processing audio data. When a valid speech segment is detected, the MFCCs of the segment are computed and given as an input to our trained GMM-HMMs. Based on the log-likelihood value outputs of the six trained GMM-HMMs, we classify this speech segment into the label of the GMM-HMM corresponding to highest log-likelihood.

6 Flow

Training audio filenames are loaded and MFCC and delta MFCC features are computed for these files and concatenated. The concatenated features are organized into categories for various commands like "Odessa," "turn_on," "turn_off," "play_music," "stop_music," and "time." A class `HMM_GMM` is defined for training Hidden Markov Models with Gaussian Mixture Models (GMMs). Functions for the forward-backward algorithm, normalization, and stochastic matrix conversion are implemented. The GMMs are trained on provided data and then the HMMs are trained using the EM algorithm to optimize model parameters. A HMM-GMM model is trained for each of the commands using the pre-processed and organized training data. Functions to compute energy and zero crossings of audio signals are defined, which are used for speech activity detection. A real-time speech detection function is implemented that processes audio input, detects speech segments based on energy and zero-crossing thresholds, and manages the audio buffer. When speech is detected, the buffer is processed to identify segments and classify them using the trained HMM-GMM models. Audio is continuously recorded, processed in chunks, and speech segments are detected. MFCC features are extracted from detected speech segments and classified using the trained HMM-GMM models. A function to compute log-likelihoods for different models is defined, and the label of detected speech segments is predicted based on the highest log-likelihood.

7 Results and Evaluation

The models are evaluated on a validation set, with the accuracy computed for each command category. The overall goal is to optimize the model parameters such that the likelihood of the observations is maximized, enabling accurate modeling and prediction of the underlying processes generating the data. The errors obtained are tabulated below in Table 1.

Label	**Training Error**					**Validation Error**				
	Fold 1	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**
Odessa	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
turn_on	0.00	0.12	0.06	0.00	0.00	0.17	0.17	0.00	0.17	0.00
turn_off	0.06	0.04	0.08	0.00	0.00	0.17	0.00	0.17	0.00	0.00
play_music	0.00	0.00	0.00	0.00	0.12	0.00	0.00	0.00	0.00	0.00
stop_music	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08
time	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Overall	0.01	0.05	0.02	0.00	0.02	0.06	0.03	0.03	0.03	0.01

Table 1: Training and Validation Errors for Different Labels Across 5 Folds

8 Conclusion

This ASR system effectively processes real-time audio input to recognize specific commands using MFCC feature extraction and HMMs. The system demonstrates accurate recognition performance on the evaluated dataset and provides real-time responses to recognized commands.

References

[1] Medium: Analytics Vidhya, *Baum-Welch Algorithm for Training a Hidden Markov Model (Part 2 of the HMM Series)*, `https://medium.com/analytics-vidhya/baum-welch-algorithm-for-training-a-hidden-markov-model-part-2` 2021.

[2] Lecture slides.