ECE 417: Multimedia Signal Processing
Fall 2018

Machine Problem #2

Due: Tuesday, October 16, 2018

---

1 Overview

In this machine problem, you will get a chance to develop a speech recognizer using Hidden Markov Model(HMM) to recognize certain words. You are provided the data (waveform) of four people, speaking 5 different words ("CNN", "DNN", "ASR", "TTS" and "HMM"), each five times. The audio features extracted from the waveform are provided to you. You will develop your recognizer using the Gaussian HMM model based on the audio features.

2 Procedure

1. **Extracting audio features**: You are provided both the data of waveform and the audio features extracted from the waveform. The audio features provided are simply obtained from the Matlab mfcc function.

   [OPTIONAL] There is a potential increase in the recognizer's word recognition accuracy by improving the audio features. You will get a chance to learn and explore the concepts of audio feature extraction, and will receive 10% extra credit if you can come up with another feature set, explain why you think this feature set will work better, implement your new feature vector and report the results. You will need to beat the reference implementation accuracy results to receive the full extra credit. Partial extra credit is also possible if you give reasonable explanation.

2. **Splitting training and testing data**: You should split the feature set into training set and testing set. You can concatenate the audio features in training set for each word as inputs to train your Gaussian HMM for that word.

   You will need to split the data in two ways:

   1. Consider the recordings from DG, LS, YX as training data, and recordings from MH as testing data. (75 training utterances, 25 testing utterances)

   2. Select one utterance of each word from each person's recordings as testing data, and consider the remaining as training data. (80 training utterances, 20 testing utterances)

   The goal here is to see the accuracy difference between a speaker dependent and speaker independent recognizer.

3. **Training the Gaussian HMM model for speech recognition**: You will train a left-to-right non skip Gaussian HMM model for each of the five words. Training a Gaussian HMM is essentially a procedure of estimating the parameters of that model to best represent the word. An HMM is normally identified with parameter set $(\pi, \mathbf{A}, \mathbf{B})$ where $\mathbf{A}$ is a state transition matrix, $\mathbf{B}$ is an observation matrix, and π is an initial state distribution. In this lab, we model the observation matrix as a Gaussians with parameters (μ, σ). Therefore, the training problem becomes determining the parameters of $(\pi, \mathbf{A}, \mu, \sigma)$.

   You can initialize your transition probability matrix as [0.8 0.2 0 0 0; 0 0.8 0.2 0 0; 0 0 0.8 0.2 0; 0 0 0 0.8 0.2; 0 0 0 0 1], the number of hidden states for HMM as 5, and π as uniform distribution across 5 states. For each hidden state, μ can be initialized as the mean across the audio features for that word, and σ can be initialized as the co-variance matrix across the audio features for that word.

Then you will need to code the "forward-backward"/EM algorithm for learning the optimal parameters $(\pi, \mathbf{A}, \mu, \sigma)$ of HMM.

[OPTIONAL] The observation matrix can also be modeled as likelihood function other than Gaussian. For example, you might use a Gaussian mixture, a K-nearest-neighbors likelihood estimator, or a neural network. You can use somebody elses code to compute the likelihood function if you wish (e.g., the neural network), but you need to write your own code to integrate their function with your HMM, and you need to explain why you think that this method will work. 10% extra credit is given if you implement the idea and it gives better results than the baseline; extra credit up to 8% is given if you implement the idea and explain convincingly why it should work and report results, but the results dont beat the baseline.

4. **Evaluating the model**: After obtaining the parameters of the HMM model, you will then compute the likelihood of a word utterance in test set given the model parameters. The test word utterance will be classified as the word with the maximum likelihood. You will then compute and report the average classification accuracy on all the word utterances in your testing data.

3 Experiments

1. Train your HMM using the first four examples of each word, from each of the four training speakers (4x4x5=80 training utterances). Test using the fifth example from each speaker (4x5=20 test utterances). This is called a speaker-dependent speech recognizer, because each of the test speakers was also in the training data. Give your results in the form of a confusion matrix: a 5x5 matrix in which the (m,n)th element specifies the conditional probability that the recognizer chose the nth word, given that the mth word was correct. Hint: your overall accuracy (average of the diagonal elements in the confusion matrix) should be above 85%.

2. Train your HMMs using all utterances from speakers DG, LS, and YX (3x5x5=75 training utterances), and test using all utterances from speaker MH (5x5=25 test utterances). This is called a speaker-independent speech recognizer, because the test speaker was not in the training data. Give your results in the form of a confusion matrix. Hint: your overall accuracy (average of the diagonal elements in the confusion matrix) should be above 45%.

3. Record your own voice, saying five examples of each of the five words in your vocabulary (5x5=25 test utterances). Use the HMMs that you trained for experiment 2 in order to recognize the sample utterances that you recorded, and report the confusion matrix.

4. EXTRA CREDIT: [Up to 10%] Using a new feature set that you design, perform experiment 1 and/or experiment 2 again.

   [Up to 10%] Using a new observation likelihood function that you design, perform experiment 1 and/or experiment 2 again.

4 Notes

- You can use MATLAB or Python for your experiments.
- If the input speech file has two channels, use the the first channel as input to your algorithm.

5 Submission

You will submit (1) a report in PDF format, and (2) a zip file containing your code along with a Readme file to Compass. You must name your report as <Lastname>_<Firstname>_report.pdf and your zip file as <Lastname>_<Firstname>_code.zip

If you are working as a team, only one person should upload the report. Please make sure that the title page includes the names of all the team members.