Speech Digit Recognition with Hidden Markov Models

https://github.com/11jeonghy/SSP2023_assignments

Hyeonbin Jeong
Instructed by Gil-Jin Jang
May 1, 2023

I. Preliminaries

1. Short-Time Speech Analysis

Speech signal is highly dynamic and analysis of the full duration of speech signal is not a recommended option. When analyzing speech signal in time-domain, short-time windows are utilized when extracting parameters. Just as video data can be split into frames, audio signal is divided into sequences of windows. Windows are required to be at most as short as 5 to 10ms for accurate analysis.

Windowing is a element-wise multiplication, the result being the transformed signal weighted by the window. Windows are shifted along the signal to let the entire speech signal be processed. There are some features of windows that need to be put under consideration. Length of the window will affect the calculation amount and the shape of the window will affect the spectral representation of the signal.

2. Time-Domain Parameters

There are some import time-domain parameters that can be utilized when processing speech signals. Time-domain parameters are more straightforward, or intuitive to understand as the original speech signal need not be transformed into another domain. Yet they are important in areas like speech segmentation, speech synthesis. For this assignment, where end-point detection and noise reduction are one of the main challenges, time-domain parameters are one of the major concerns.

The first parameters to be inspected are short time energy and magnitude. These two parameters are closely related(the square of magnitude and energy). One notable application of short time energy or magnitude is segmentation between voiced and unvoiced regions. In case of the current assignment, where we are to recognize isolated digits, or words, short time energy aids in determination of the starting and ending points of the pronunciation of the word, in other words, endpoint detection.

Zero-crossing Rate also provides much information about the speech signal. Put simply, high ZCR may relate to signals with high frequency. Threshold of ZCR can be set to detect voiced and unvoiced regions of the signal.

3. Frequency-Domain Parameters

Frequency-domain parameters, also referred to as spectral parameters, are closer to the human nature of hearing. The human auditory system works by sensing signals of different frequencies, reacting more sensitively to low-frequency signals.

One of the most popular spectral analysis methods is filter-bank analysis. A filter bank refers to a set of bandpass filters, which of each analyzes a different region of the frequency field. The resolution of analysis can be changed easier by adjusting the number of bandpass filters or the coverage of each filter. Although it is possible to create a linear filter bank, it is widely known that a logarithmic approach to speech signal processing is better. Thus, the covering range of filters are set in a logarithmic manner, the filters being concentrated on the low-frequency region to improve low-frequency resolution and being sparsely distributed in the high-frequency region, inversely. This way, a logarithmic-scale filter bank, or a Mel-scale filter bank can be designed.

Short time Fourier Transform is another decent tool for analyzing speech signal in the frequency domain. For signals which can be represented as a function of time, Fourier Transform is a good way of analyzing it to find out amplitude and phase information. However, because speech signals are highly time-variant, simply applying Fourier Transform does not give much useful information. This is why, again, just like time-domain analysis of speech signals, windows are crucial in analyzing speech signals. Fourier Transform applied with the concept of windowing becomes Short Time Fourier Transform. Although the normal Fourier Transform would also give spectral information about the signal, the spectral information at a point of time cannot be determined as it the spectral information gained from Fourier Transform handles the speech signal as whole. Thus, using Short-Time Fourier Transform, it is possible to determine the spectral

information at each frame, or interval, determined by the size of the window, better describing the time-variant nature of speech signals.

II. Program Setup

1. Audio Data

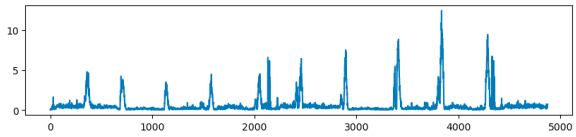
The audio database utilized for this assignment is from the participants of Speech Signal Processing Lab(DEEE0725). Each participant read aloud 10 digits, 0 to 9, each 10 times and segmented the recorded files so that each recorded file would have a digit spoken out. The test dataset consists of 8 randomly chosen participants, and the validation dataset 3 participants. The recording environment differed from person to person, but the sampling rate and the format of the file were uniform over the whole database, each being 16kHz and WAV(WAVefile Audio Format). The validation dataset was then mixed with 2 types of noise signal, with SNRs of 10dB, 0dB, and -10dB to form 6 variations. The test dataset was mixed under the same conditions to form, again, 6 variations, but unlike the test dataset and the validation dataset, consisted of unsegmented files.

2. HMM Model Setup

When writing the initial program, I referenced the GitHub source file[1] written for spoken digit recognition, but made a few changes in order to fit the program with this assignment. First of all, the MFCC feature extraction function was edited to make further utilization convenient and the DataLoader function had to be redefined to match the current project files' directory. The prediction function was also added to make assessment of the model's accuracy a straightforward task. Implementation of the Hidden Markov Model was done with hmmlearn library[2] and audio data handling was done with librosa library[3].

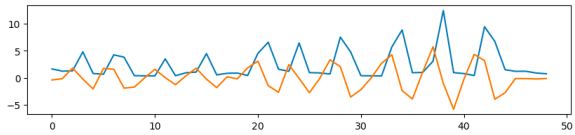
III. Experiments/Results

1. Test Data Segmentation



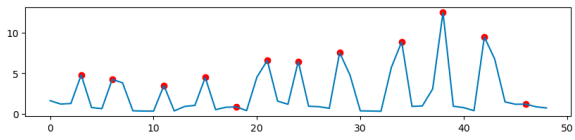
Short-Time Energy of a speech signal

The segmentation of the test dataset was done with End-Point Detection based on short time energy. I focused on the point that there is a local maximum in short time energy for each of the speech segments. For ideal unsegmented speech signal, it would thus be possible to segment the signal by finding out the local maxima and utilizing them to detect endpoints. However, the input signal was too jiggly due to the presence of noise, and several additional methods had to be applied to gain local maxima that correspond to the speech segments.



Intervalized short-time energy and its derivative

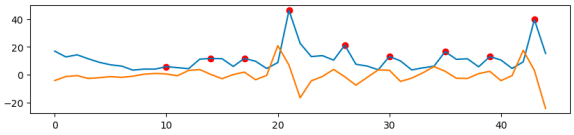
The first and the most straightforward method was to perform short time analysis again on the short time energy. The size of each interval was set to 100 and the mean value of short time energy in each interval was retrieved and used to represent each interval. I'll refer to this process as intervalization of short time energy for the rest of the report. It can be simply noted as short-time energy gained with bigger frame size, but since short-time analysis is valid under the condition that it represents the dynamic time-variance of speech signals and the purpose of intervalization of short-time energy is to ignore the time-variance of the speech signal to make analysis easier, I thought it would be useful to separate these, at least in this written report.



Local maxima of the intervalized short-time energy

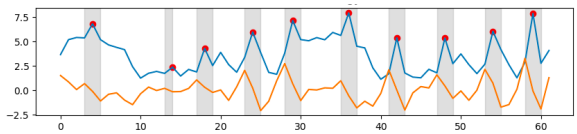
Still, the local maxima of the intervalized short time energy did not perfectly fit to the speech segments, so another method was utilized to determine whether a local maximum is an appropriate one. I utilized the derivative of the

intervalized short time energy. The index of the appropriate local maxima that we are willing to find also represents zero-crossing points of the derivative of the intervalized short time energy, although it may not correspond in a one-to-one manner. Adding this determination process, it was able to determine most of the local maxima corresponding to the speech segments in unsegmented test speech signals. Even for signals with -10dB SNR, the proposed method was able to determine proper local maxima of the short time energy effectively.

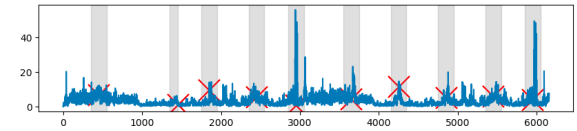


Local maxima gained via intervalization and derivative-based determination under -10dB SNR condition

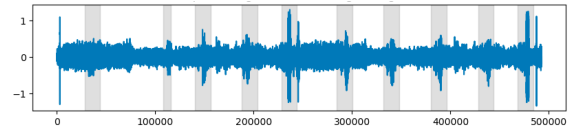
The next problem to be solved in the segmentation process was to determine the starting point and the ending point of each speech segment. As the local maxima, which correspond to the approximate center of speech segment, have been determined, the nearest endpoints of each local maximum had to be figured out. I focused again on the derivative of the intervalized short time energy, and utilized the fact that a spike of energy would yield a sine-like structure on the derivative. Thus, the local maxima and minima of the derivative of intervalized short time energy were calculated. Starting points of the speech segments were determined by finding out the local maxima of the derivative closest to the approximated center of the speech segments, and the ending points of the speech segments were determined finding out the local minima of the derivative closest to the approximated center of speech segments. In case the estimation result contained more than 10 speech segments, the segments were thresholded to cross out improper segments.



Estimated speech segments on the intervalized short time energy-magnitude scale, under -10dB SNR condition



Estimated speech segments on the short time energy-magnitude plane, under -10dB SNR condition



Estimated speech segments on the original speech signal-amplitude plane, under -10dB SNR condition

2. Assessment of the Database

The initial setup showed 32.90% accuracy on the segmented test files and 56.67% accuracy on the segmented test files without noise synthesis. The first thing to be checked to improve the accuracy was appropriateness of the audio data. All the files were sampled at 16kHz, so there was no defect regarding the format of the files, but some mislabeled data were found, so the training process was conducted again, with the mislabeled files excluded(to be specific, directory ‘shin3875’ was excluded in the process of creating the training dataset). The result of experiment conducted with mislabeled training data excluded did yield some improvement. It showed unsatisfactory accuracy in validation datasets but showed some improvement in the test dataset accuracy. The overall test accuracy was improved to 40.09% and showed 68.89% accuracy on the segmented test files without noise synthesis. Thus, the remaining experiments were all conducted with the mislabeled data excluded in order to maximize the test accuracy of the model.

```
predicting the train dataset...
Accuracy:  63.797 %
predicting the original validation dataset...
Accuracy:  30.333 %
predicting the nbnSNR10 validation dataset...
Accuracy:  10.0 %
predicting the nbnSNR0 validation dataset...
Accuracy:  22.0 %
predicting the nbnSNR-10 validation dataset...
Accuracy:  10.0 %
predicting the wbnSNR10 validation dataset...
Accuracy:  9.333 %
predicting the wbnSNR0 validation dataset...
Accuracy:  20.333 %
predicting the wbnSNR-10 validation dataset...
Accuracy:  9.333 %
predicting the test dataset...
Accuracy:  32.902 %
predicting the clean dataset...
Accuracy:  56.667 %
```

Digit prediction,
Mislabeled data included

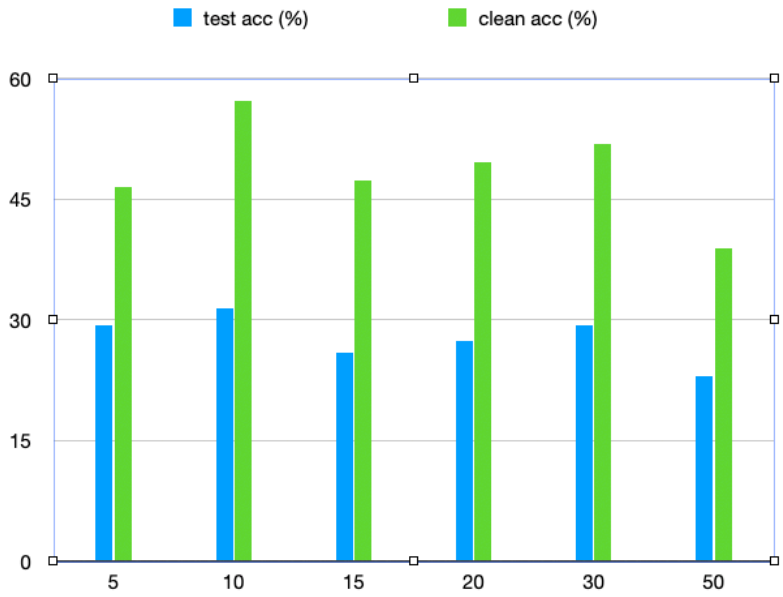
```
predicting the train dataset...
Accuracy:  67.101 %
predicting the original validation dataset...
Accuracy:  37.0 %
predicting the nbnSNR10 validation dataset...
Accuracy:  10.0 %
predicting the nbnSNR0 validation dataset...
Accuracy:  23.667 %
predicting the nbnSNR-10 validation dataset...
Accuracy:  10.0 %
predicting the wbnSNR10 validation dataset...
Accuracy:  14.667 %
predicting the wbnSNR0 validation dataset...
Accuracy:  17.0 %
predicting the wbnSNR-10 validation dataset...
Accuracy:  14.667 %
predicting the test dataset...
Accuracy:  40.086 %
predicting the clean dataset...
Accuracy:  68.889 %
```

Digit prediction,
Mislabeled data excluded

3. MFCC

The next option was to adjust the number of MFCC features to be extracted. It is known that the number of MFCC features is usually set to a value around 10~20, the bigger the better for more comprehensive applications such as speech recognition or cases where the signals are noisy. The extracted number of MFCC features were set to 5, 10, 15, 20, 30, 50 and 5 tests were conducted on each case. The result is saved as an additional PDF file, and for each of the n_mfcc sections, the 5 non-highlighted rows are the actual test results and the highlighted row refers to the average value of the 5 tests' results. The visual representation of the test data is shown below.

It was thus able to find out that higher number of MFCC features did not necessarily relate to better test accuracy. The ideal MFCC for this assignment seem to be around 10, although the exact figures would have to be calculated with more data and higher precision for accurate determination.



Although the test results were not perfectly satisfactory, the highest test accuracy measured being just over 30%, the model seemed to be able to identify a spoken digit if no noise were synthesized. With the models trained with 10 MFCC figures configuration, it was able to identify roughly 6 out of 10 spoken digits under clean surroundings. The expected accuracy for this assignment if random guesses were to be made is only 10%, so the MFCC features did help us make algorithms that would identify different speech.

V. References

[1]https://github.com/msnmkh/Spoken-Digit-Recognition/blob/master/SDR.py

[2]https://hmmlearn.readthedocs.io/en/latest/

[3]https://github.com/librosa