E207 Final Report: Digit Audio to Text Converter

Jimmy Fernandez and Christina Lau, May 13 2020

Abstract — This paper describes the team's efforts to create an audio to text system for audio containing numerical digits from 0 to 9. The team created a system to process digits that split queries into individual words, calculated Mel-Frequency Cepstrum Coefficient features, and compared these features to audio references of known digits using the subsequence dynamic time warping algorithm. After several iterations and experimenting with other processing methods, the final system correctly classified digits 94.02% of the time, with a Macro F1 Score of 94.04% and a speed of 5.12 digits/second.

I. INTRODUCTION

In this project, the team was tasked with applying some of the signal processing tools they had learned in Harvey Mudd College's ENGR207: Signal Processing, Modeling, and Classification course to a problem of their choice. After some consideration, the team decided to investigate a simple Automatic Speech Recognition (ASR) System.

An ASR system is designed to convert input speech into a form that can be understood and used by computers. Modern ASR software is designed to recognize a vast vocabulary of words using neural networks and other advanced tools [1]. However, because the team had only worked with classical signal processing methods, they decided to narrow their scope in hopes of achieving similar performance on a smaller scale.

It was ultimately decided that the input vocabulary would be limited to only the digits 0 to 9, and the system would only evaluate audio from a single speaker, one of the team members. The system would take in an audio recording of a sequence of digits spoken aloud, and output its predictions of the digit sequence as text. The system would be evaluated based on its ability to correctly classify the digits, as well as how quickly it was able to classify them.

Initial design of the system as well are discussed in Section II. Further experimentation on the results and changes made to improve the system are discussed in Section III. The results of the system and other interesting parts of the team's experiments are discussed in Section IV, and Section V lists the team's conclusions.

II. SYSTEM DESIGN

A. Data Collection and Database Creation

There were two types of audio files that needed to be recorded: reference audio, audio recordings of a single digit to be used as a base for comparison, and query audio, audio recordings of sequences of digits. As previously mentioned, for the scope of the project, the team limited the audio recordings to a single person, one of the team members. Furthermore, each digit spoken in the query audio was separated by approximately one second of silence to facilitate distinguishing digits from each other. For the project, the team member recorded 10 audio files of single digits from 0 to 9 for the reference audio and 100 audio files of 2 to 10 digit long sequences for the query audio.

Once data collection was completed, the team created a database for the reference audio. As storing entire audio files would use a sizable amount of space and take long periods of time to parse for later comparison, the team decided to represent each audio file as a feature matrix, where each column of the matrix is a representation of the important features of the audio at a particular point in time. The team chose to use the Mel-Frequency Cepstrum Coefficients (MFCCs) as their features, which are described later in the report. Once the features were calculated for each reference audio file, the feature matrices were stored in a Python dictionary. A block diagram of this process is shown in Figure 1.

B. General System

The system was built to utilize subsequence dynamic time warping (DTW), an algorithm used to measure the similarity between one set of time series data to a subsection of another set of time series data. To do this, each query audio file is first split into segments containing the audio of an individual digit. Each segment is then translated into a feature matrix, where the features are the same type as the features in the database. Subsequence DTW is then used to find the similarity between each audio segment feature matrix and the reference feature matrix of each digit stored in a database. The reference segment that has the highest degree of similarity with the query segment is chosen as the digit spoken in that segment. This process is repeated for every segment of a query file to calculate all the digits spoken in the query. A block diagram of the overall system is shown in Figure 2.

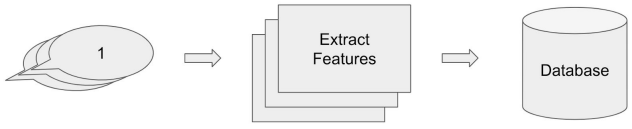

Figure 1. Database Creation Block Diagram

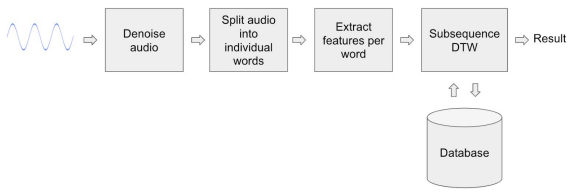

Figure 2. Proposed System Block Diagram

Denoising was added as a preprocessing step after initial attempts to automatically split the query audio were hindered by noise. As the team was familiar with the Librosa library from past assignments, the team chose to use Librosa's split function [2], which returns the start and end times of non silent segments within an audio file. Noise in the query files were being detected as non-silent segments, which resulted in the splitting function returning more segments than the expected amount of digits. To remove this problem, all the query audio files were denoised with Audacity's Noise Reduction Tool [3]. To maintain consistency, the team also denoised the reference audio files for the database.

C. Improvements to Query Audio Digit Splitting

Even after noise reduction, the automatic audio segmentation still occasionally returned an incorrect number of splits. These cases resulted from words that had periods of silence within the word, mainly the digits 6, 7, and 8. An example of this is the word "six," where the audio can be split into the "sik" and "ss" portion. Figure 3 shows an audio recording of the word "six."

Initially, the team inputted the number of digits to expect per query into the system. As a result, the system would remove the shortest segments until the number of segments matched the number of expected digits. This method however had two major flaws: first, the team had to input extra knowledge about the query into the system and second, the deletion of the smallest segment negatively impacted the performance of the split digits (6, 7, and 8) as parts of the full audio of the digits were being discarded. To address this, the team implemented a minimum silence length.

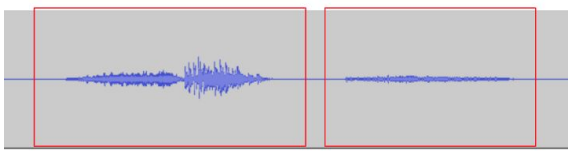

Figure 3. Reference Audio of "six," which would be split into two nonsilent segments rather than one

Thus, if the time between two consecutive segments were below the threshold, the segments would be combined into one. The minimum silence length was set to be 0.24 seconds, which was determined by measuring the longest split between segments of digits in the reference and adding a generous margin for variation. After this, all 100 queries were correctly split into the desired number of segments.

D. MFCC

After some initial research in the field, the team decided to use MFCC as the features for each frame of the audio. Mel Frequency Cepstrum Coefficients are commonly used when designing ASR software, as they represent energy in the frequency spectrum in a nonlinear way that more accurately reflects how humans speak [4]. Using online sources, the team implemented their own version of the MFCC calculation.

E. Initial results

The team tested both Euclidean distance and cosine similarity metrics. As can be seen in Table 1, the system using the cosine similarity metric performed slightly better with a Macro F1 Score of 77.90% and full digit accuracy of 78.12%.

III. SYSTEM MODIFICATIONS

A. Reference and Query Preprocessing Modifications

The team first decided to experiment with different types of references. In the initial system, the references were also denoised to remain consistent with the denoised queries. As an experiment, the team tested the system performance using the noisy references. While the system using Euclidean distance did worse, the system using cosine similarity did much better, with a Macro F1 score of 93.91% and full system accuracy of 93.85%.

After this, the team decided to also test performance with the noisy references and noisy queries using split times from the denoised queries. While this increased the performance for the Euclidean distance metric, it drastically decreased the performance for the cosine similarity metric compared to only using the noisy references. The results of these tests are shown in Table 2.

Table 1. Initial Test Results

Metric	Macro F1 Score	Full Digit Accuracy
Euclidean	77.75%	77.26%
Cosine	77.90%	78.12%

Table 2. Noisy References and Queries Test Results

Metric	Macro F1 Score	Full Digit Accuracy
Noisy Ref, Euclidean	49.38%	50.60%
Noisy Ref, Cosine	93.91%	93.85%
Noisy Ref and Queries, Euclidean	69.89%	69.40%
Noisy Ref and Queries, Cosine	75.15%	74.18%

From a recommendation, the team also tried to "desilence" the references, or remove silent periods before and after the digit was spoken in the reference audio. It was hypothesized that the silent periods might reduce accuracy; however, in all cases, both the Macro F1 score and system accuracy decreased. Thus, the team decided not to pursue this any further.

B. MFCC Changes

After doing more research, the team found that ASR systems commonly make a number of changes to their features on top of the base MFCC features. The team implemented a number of these changes in an effort to improve the performance of their system.

Firstly, it is common to calculate a set of 26 Cepstrum coefficients, but cut out the set of top coefficients (corresponding to higher frequency energy), as this has been found to improve performance for ASR [4, 5]. In their second iteration of the feature matrix calculation, the team decided to keep the bottom 12 coefficients of the 26 that were calculated.

Another common addition to the feature matrix is to append what is called "delta" and "delta-delta" features on top of the base features. These features are approximations of the first and second derivatives of the original features, respectively. They are calculated based on how each element of the original feature changes between frames [4, 5]. These features improve the system performance by adding information about the dynamics of speech [5]. The team used Librosa's delta function to calculate and append the first and second derivatives to their feature matrix in their second iteration of the feature calculation [6].

After completing the second iteration of the MFCC calculation, the team reran their benchmarks for the various different reference sets discussed in Section III part A. Overall, the team found that the new features matrix slightly decreased performance in most categories and slightly increased performance in the noisy reference category (the highest performer). The team ultimately decided to make this the standard feature calculation in their system

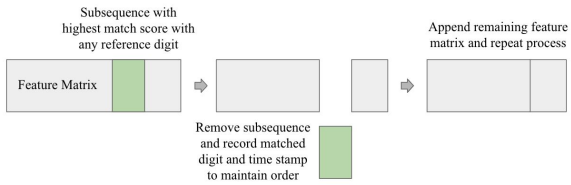

Figure 4. Alternative Digit Matching Process

despite having very small decreases in accuracy due to the far more robust theory behind the features, with it containing information on dynamics of speech via the delta and delta-delta features [4, 5].

C. No Split Method

The team decided to investigate an alternative to splitting the query audio with silence detection. In this alternative, the system would use subsequence DTW on the entire query audio rather than the reference audio and try to find the subsequence in the query audio that best matched a reference digit. After finding the subsequence with the highest match score to any of the reference digits, the system would record the matched digit and the timestamp of the subsequence. The subsequence would then be removed from the query, and the process would be repeated until the query passed a minimum length threshold. This process is shown in Figure 4.

The alternative system faced many problems with guessing the correct number of digits as it was difficult determining thresholds to stop the digit matching process. At best, the system was able to get the correct number of digits for only 74 of the 100 queries. To address this, the team created one final iteration of the MFCC feature calculation. This feature matrix contains an extra base feature per frame representing the total energy in that frame. This was to add information about the volume of each frame to help the system distinguish between silence and actual speech. Unfortunately, the team found that this feature did not increase the performance of the No Split method and actually decreased performance instead. As a result, the team decided not to further pursue this alternative method as the system performance was drastically hindered by this issue with predicting the correct number of digits.

IV. RESULTS AND DISCUSSION

Based on the team's experiments, the final system used noisy references and denoised queries. To find the match score, the system calculated the MFCC features with delta-delta features and performed subsequence DTW using the cosine similarity metric.

Digit	Precision	Recall	F1-Score
0	0.9388	1.0000	0.9684
1	0.8472	0.8841	0.8652
2	1.0000	0.9595	0.9793
3	0.9683	0.9531	0.9606
4	0.9500	0.9194	0.9344
5	1.0000	0.9531	0.9760
6	0.9149	0.9556	0.9348
7	0.8551	0.9833	0.9147
8	1.0000	0.9444	0.9714
9	0.9524	0.8511	0.8989

Macro F1 Score: 0.9404
Full Digit System Accuracy: 0.9402
Total Run Time: 114.1403 seconds
Digits Per Second: 5.1253

Figure 5. Final Results

As seen in Figure 5, the final full system accuracy was 94.02% with a Macro F1 score of 94.04% and a run time of around 114 seconds, or about 5 digits per second.

Throughout the project, the runtime remained relatively consistent, with very minor speed improvements in the final iteration. The team never considered the runtime an issue that warranted substantial investigation into improving however, since it was always at a rate faster than a human would reasonably speak.

Examining digit-wise performance leads to more insights about the system's performance. In particular, the digits 1, 7, and 9 performed poorly compared to other digits. By examining the confusion matrix used to generate the final result metrics, the team was able to determine that 1 and 9 were being commonly guessed as 7. This fact is reflected in 7's high precision and lower recall. The team is unsure as to how this discrepancy arises.

In general, the cosine similarity metric performed better than the Euclidean distance metric. This result matches the team's intuition, as noise in the feature vectors would likely alter the Euclidean distance more substantially than the angle between them.

Using the noisy references increased the system performance by over 15%; however, using noisy audio for both the references and query decreased system performance. This is likely because denoising removed some distinguishing features between digits from the reference audio which in turn decreased the performance. On the other hand, using the noisy queries introduced too much noise to the system and interfered with the digit matching process. Thus, using the noisy references and denoised queries provided the best result.

Desilencing the reference audio surprisingly decreased the system performance. The team hypothesizes that this occurred because the desilencing was done manually while the query audio segmentation was done automatically. This may have resulted in a small amount of silence at the start and end of each query audio segment which negatively impacted the match with the desilenced references.

V. CONCLUSION

In this project, the team was able to create an Automatic Speech Recognition system that achieved high performance in both precision and recall using only classical signal processing tools. Though modern ASR systems using more refined signal processing tools can often achieve higher performance on a larger scope, the team's successful system shows the power of classical signal processing tools. Classical tools certainly have their uses in real world systems.

If the team were to further develop this system, they would recommend looking into two potential areas of interest. The first would be to expand the system to be speaker independent, so any individual could speak a digit that could be accurately converted. The second area for improvement is looking into alternative ways of processing queries, besides the words splitting and "no split" methods described in this report. The team feels that these unexplored avenues could reveal interesting insights into designing ASR systems.

ACKNOWLEDGEMENTS

The team would like to thank Professor Tsai for his encouragement and recommendations on improving the system. The team is also grateful for Librosa's development team, as the Librosa library was extremely helpful in developing this implementation.

REFERENCES

1. "Speech Recognition." Internet: https://en.wikipedia.org/wiki/Speech_recognition, May 5, 2020 [May 10, 2020].
2. "liborsa.effects.split." Internet: https://librosa.github.io/librosa/generated/librosa.effects.split.html, n.d. [Apr. 14, 2020].
3. "Noise Reduction." Internet: https://manual.audacityteam.org/man/noise_reduction.html, n.d., [Apr. 14, 2020].
4. J. Lyons. "Mel Frequency Cepstral Coefficient (MFCC) tutorial." Internet: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/, n.d., [Apr. 14, 2020].
5. J. Hui. "Speech Recognition - Feature Extraction MFCC & PLP." Internet: https://medium.com/@jonathan_hui/speech-recognition-feature-extraction-mfcc-plp-5455f5a69dd9, Aug. 28, 2019 [May 9, 2020].
6. "librosa.feature.delta." Internet: https://librosa.github.io/librosa/generated/librosa.feature.delta.html, n.d. [May 8, 2020].