Daniel Shannon
Assignment 3: Proposal
CIS 663 Biometrics
Syracuse University
February 21st, 2024

The effects of malapropism on voice feature extraction and the whisper-tiny speech to text model.

Abstract

Here we explore the common features in everyday phrases that lead to malapropisms in human conversation. With the extracted features, we aim to deterministically fool the OpenAI voice-to-text whisper-tiny model. Beginning with a list of 40 phrases, we record ourselves saying these phrases, find similar phrases in open source recorded voice repositories, and use generated voices. We then process the samples to remove noise, center and pad the waveforms, and employ voice activity detection. The distance is then measured between the phrases and audio is produced using the features within the threshold distance. The feature generated audio samples are passed through the whisper-tiny model and the results are compared to the original features of the phrases using the Levenshtein distance.

Introduction

Malapropisms often go unnoticed or unrealized by the speaker and listener. A malapropism is “the usually unintentionally humorous misuse or distortion of a word or phrase.” We will explore and identify the recurring features across processed and padded audio recordings and generated recordings. Our hypothesis is that if we can isolate the features across phrase pairs that make up malapropisms or other similarly related phrases, then we may be able to structure these features in a way to deterministically fool the OpenAI whisper-tiny speech-to-text model [1].

We start with data generation, which involves gathering data from the Common Voice corpus 16.1 [5], self-recording, and some AI-generated recordings. Common Voice recordings that have short Levenshtein distances will be paired together under the assumption that similar phrases might share similar features. The whisper-tiny model was trained using only voice recorded data [2]. We will attempt to generate these features with recorded voice and AI-generated voice.

We must process the samples to ensure that we are sampling voices and that the phrases are aligned. First we derive the waveform and resample to match the sample rate of 1600 Hz, as done in the whisper model. After resampling, we use a Voice Activity Detection algorithm to detect voice, and center and pad the detected voice so the two samples align.

After alignment we can take the Euclidean Distance between the waveforms and process this distance into a Mel Spectrogram and separately, as audio. The more useful distance to measure is the Mel Frequency Cepstral Coefficients (MFCC) [6]. With the differentiated MFCCs we can use the Griffin Lim to turn this back into a waveform. Another option is to classify MFCC differentiated features and then extract those features from a word and then generate a phrase that has multiple meanings. We can do this using k-means clustering and iterating through combinations of features to pass through the Whisper model.

Previous Work

The model we are using to test the speech-to-text generation is a large-scale unsupervised model known as Whisper. Whisper was trained on 680,000 hours of unsupervised human audio recordings paired with transcripts and has 5 model sizes ranging from tiny (39M parameters) to large (1550M parameters). Because machine generated transcripts have been shown to decrease performance [8], Whisper used heuristics in the data grooming process to identify and remove machine generated transcripts.

The Whisper model was trained on data that is processed into 80-channel log-magnitude Mel spectrograms that has been resampled to 16,000 Hz. The mel spectrogram is computed on a 25 millisecond window with a stride of 10 milliseconds. This will guide our own processing decisions as we will attempt to match the processing parameters used to generate the mel spectrograms.

Rather than working directly with Mel Spectrograms, past studies have had success working with the Mel Frequency Cepstral Coefficients (MFCC). MFCCs allow us to reduce the dimensionality of the original waveform and subsequent Mel Spectrogram by taking power that varies significantly over time and reducing the frame of change. The steps for MFCC are Pre-emphasis, Framing and Windowing, DFT (Discrete Fourier Transform), Mel-Frequency filter bank, logarithm, DCT(Discrete Cosine Transform), to finally the MFCC [8]. This results in a power log scaled histogram with stable voice data in frames of information.

More on MFCC and their uses.[8] How to use k-means to classify using MFCC [9].

There are common python libraries that we can use for the transforms we are using such as MFCC, mel spectrogram, VAD, and band pass filtering.

- **A section on language and voice detection**
- **A section on ai text-to-speech models**

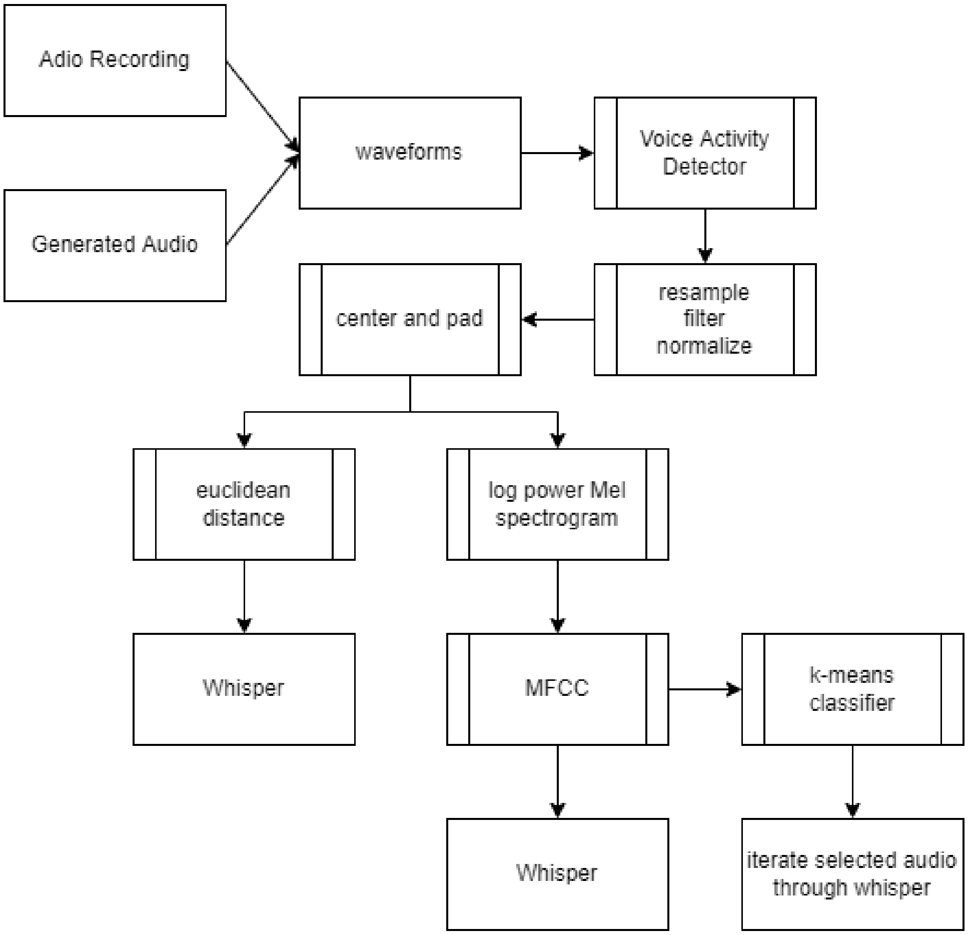
Experiment Design

We propose extracting the features used in the voice to text model whisper-tiny [1][2] by comparing common malapropisms and mispronunciations. We use a combination of recorded voices and generated audio. For the generated audio, we compare the levinstein distance between the input text and the output text and remove recordings that do not meet the minimum threshold distance and iterate on this process until we have a well sized corpus of data (50 samples of each phrase to start). The generated audio is the output of the torchaudio text to speech model Tacotron2 [3].

To process the waveforms we apply the torchaudio Voice Activity Detector (VAD)[4] and then center, pad, and normalize the waveforms. The waveforms are then passed through a bandpass filter to filter out interference that is not in the same frequency range as human speech. We then resample the waveform to 16,000 Hz. This normalized, centered, filtered and resampled data is considered the processed data. Finally we create a log Mel spectrogram of the processed data with a 25 millisecond window and a stride of 10 milliseconds. These sample rates and windows match those used to train Whisper.

With the processed data, we can measure the euclidean distance between the malapropism pairs. We need to filter out values higher than a threshold amplitude because we want to extract features that are close in distance. This gives us a starting point to compare distances to Whisper. Whisper does recognize these differences as voice and will generate text. Ideally, the generated text is prominent phonetics that compose a phrase.

To improve on measuring the distance between the processed waveforms, we can extract the MFCCs from the log power Mel spectrogram. This converts the waveform from the time domain to the frequency domain. The Mel spectrogram is a more realistic visualization of human speech because humans do not hear frequency intensity linearly. Now we can measure the distance between the realistic audio characteristics of the phrases.



This is honestly where I'm a little unsure of what to do…
With the filtered, distanced MFCC we can begin to classify the features of speech that make phrases have meanings with one similar overall sound.

We can use k-means clustering[9] to classify the extracted features and relate them back to the original phrases. With the features identified, we then transform the feature extracted from the mel spectrogram back into the time domain. We test combinations of these features on whisper to test if the model can regularly produce two of the same phrases.

The progress so far can be seen here, in the feature-extraction branch of voice-attack [10].

1. openai/whisper-tiny · Hugging Face
2. [2212.04356] Robust Speech Recognition via Large-Scale Weak Supervision
3. Text-to-Speech with Tacotron2 — Torchaudio 2.2.0 documentation
4. https://pytorch.org/audio/stable/generated/torchaudio.functional.vad.html
5. https://commonvoice.mozilla.org/en/datasets
6. https://www.researchgate.net/profile/Golam-Rabbani/publication/255574793_Speaker_Identification_Using_Mel_Frequency_Cepstral_Coefficients/links/55f05d5908ae0af8ee1d1894/Speaker-Identification-Using-Mel-Frequency-Cepstral-Coefficients.pdf
7. [2109.07740] Scaling Laws for Neural Machine Translation
8. Mel Frequency Cepstral Coefficient and its Applications: A Review | IEEE Journals & Magazine
9. Speaker Identification Using K-means Method Based on Mel Frequency Cepstral Coefficients
10. https://github.com/radioxeth/voice-attack/tree/feature-extraction