Phrase detection Neural Network

Final Project Proposal

Jun Lin Tan ([tan65@wisc.edu](mailto:tan65@wisc.edu))

**Introduction**

Voice recognition and natural language processing have found its way into our daily routine. This is made possible by recent technological advancements, primarily powerful computing power that is able to support state of the art voice recognition artificial intelligence algorithms, efficiently enough to fit all of that in a device no bigger than the size of your palm. In this project, I aim to built a Deep Neural Network capable of identifying a few audio phrases. The goal is to be able to imitate the “Hey Siri” detection seen in Apple devices, but with “Hey Bucky” instead.

**Background**

The program should ideally be able to take in a continuous live audio stream from a computer mic and process the time stamped audio frequencies into a more interpretable format through Discrete Fourier Transformation. A trained Neural Network will be used to detect the phrase from the preprocessed audio signal and provide an output accordingly.

**Dataset**

The dataset will consist of my voice recorded through a microphone from my phone. Currently, I plan to use only voice samples from myself to eliminate additional complications that needs to be considered while training and testing the neural network. However, I will also experiment with voice samples from different users in regard to both training and testing the DNN model. I plan to have a dataset of 2000 voice samples from myself, where 200 of them will carry the positive label, and an additional 1000 voice samples from a friend with a similar positive label ratio. Voice samples with background noise and different white noise will be used as well.

1. **Dataset distribution**

A typical scenario is where the model will be used to continuously sample audio signal from the environment. Thus, the model needs to be able to detect when “hey bucky” is mentioned precisely. To mitigate this issue, the full data set will have a 9:1 ratio of negative: positive voice samples.

**Method**

1. **Preprocessing audio signal**

(<http://asr.cs.cmu.edu/spring2014/lectures/class3.featurecomputation.pdf> )

I plan to use either the Sphinx audio feature extraction library developed by CMU, or wav2letter++ by Facebook AI for the preprocessing step of my audio data. Currently, I am leaning more towards Sphinx because of the resources I am able to find about using it.

It will not be possible to extract any important information regarding the audio signal from just the frequency and time stamp of each frequency value. There are many variables that has to be considered when extracting features from the audio signal. For example, the amplitude of the frequency, the speed of the phrase sample, the distribution of the signal across the detected time frame. In order to derive meaningful features from the audio signal, the following step will be taken:

* **Discrete Fourier Transform and windowing**

The DFT coefficients of an audio signal can be calculated through Fast Fourier Transform. However, the DFT coefficients calculated from an audio signal assumes that the signal length is infinite. An appropriate approach to extracting the DFT coefficients will be to slice up the signal into slices. This concept where the signal is extracted at ascending intervals in reference to the time signature is called windowing

* **Parameterization**

This step will consider the characteristics of the human auditory system. The signal will be warped such that lower frequencies have more importance and sensitivity than higher frequency to mimic the frequency resolution of the human ear. Next, frequencies higher and lower than the human auditory frequency range will be filtered

* **Normalization**

This step aims to remove the variability that arises from noise and speaker variation. The Cepstra mean will be deducted from the Cepstra vector of the entire audio singal. A good way to conceptualize this step is normalizing a dataset to make sure each feature holds equal importance.

* **Feature extraction**

The acceleration between the Cepstra vector within the entire ceptrum of the signal can be calculated to yield an acceleration feature that can be used as the input to the neural network.

1. **Neural Network**

(<https://escholarship.org/uc/item/6mm160gq> )

A deep neural network will be implemented and trained to detect “Hello Bucky” from the features extracted from the previous step. The architecture of the neural network has yet to be decided. This part of the project will have strong references from Guo’s Et al. dissertation paper on Neural Network Based speech and speaker recognition.

The paper discusses in detail the procedures used for a full end to end speech recognition system. Many of the intricacies discussed in the paper can be omitted in my case. The most important aspect that I can learn from the paper is discussed in 3.3 (Feature learning in the frequency domain).

1. **Training and Testing**

The data sample carrying the positive label will be up-sampled (probably using Synthetic Minority Oversampling Technique, SMOTE) to produce a balanced training data set. The actual evaluation metric of the model will be based on the original data set with the 1:9 label ratio, prioritizing on both the recall and precision.

1. **Deployment**

live audio feed will be registered using Pyaudio. A set period (t) of recorded audio will be continuously stored, preprocessed, and fed into the neural network at a reasonable short time interval. An easy way to visualize this concept is that a window of period t will be continuously sampled from the live audio feed at a short time interval.

A picture containing graphical user interface

Description automatically generated

**Project planning**

Currently, I aim to proceed with this project individually but I am open to working in a team as well. Both the preprocessing step and neural network step hold equal weight in importance. The first few weeks of the project will be dedicated to building a robust preprocessing script that will yield a decently large dataset to train the neural network. The Neural network will focus on the essential components required to detect short and simple phrases.