Final Project Review

SDP Team 23

SDP Team 23

Liam Shea

Phillip Ashe

Jason Nguyen

Paul Mahoney

Why Voice Biometrics?

- Benefits of Voice Biometrics
    - Easier to use than e.g. iris scanner
    - Low cost (no specialty hardware needed)
    - Can be used as an added layer of security in many applications

Project Goals

Create a reliable, portable system that could provide near real-time speaker identification.

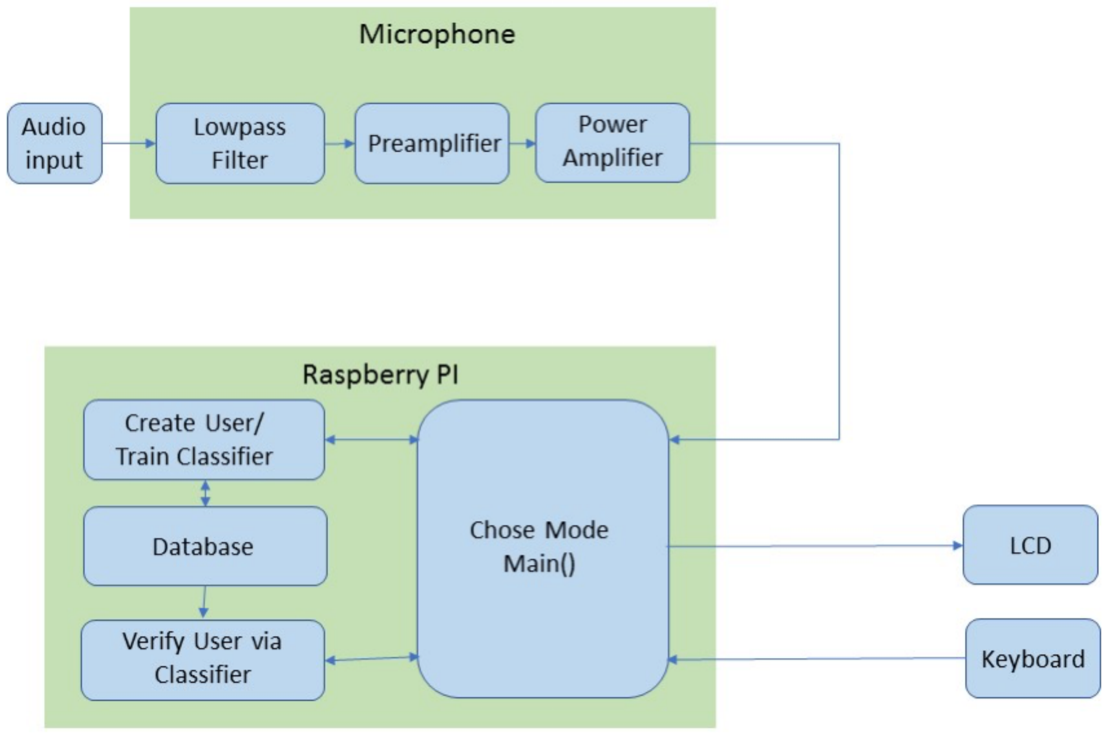
Should exist as a pair of platforms:

- A mobile platform that records and process voice signals, and output identification features
- A server platform to store to the features and match features against those stored in database

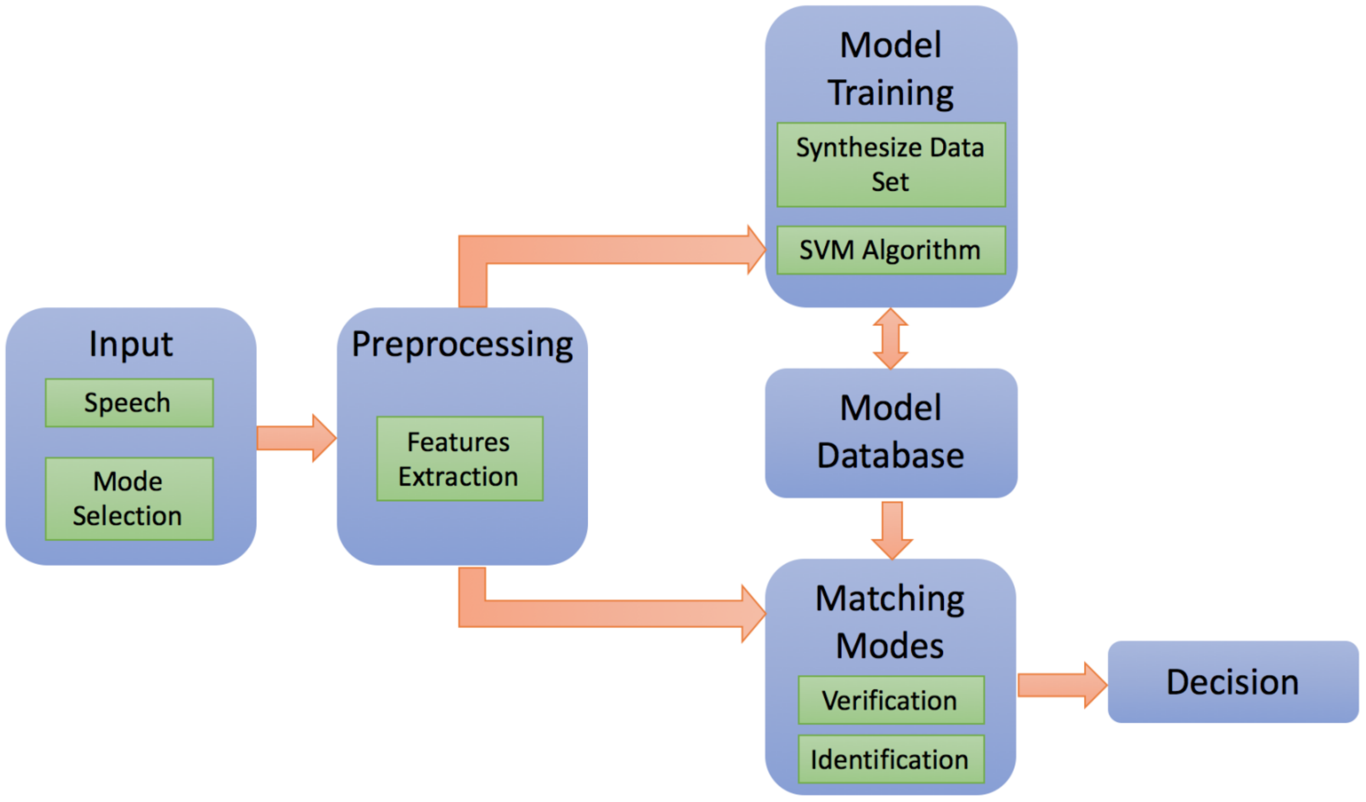
New goals after drawbacks:

- Software working properly with at least 75% accuracy
- Entire system runs on a PC
- Program runs entirely in MATLAB.

Original Block Diagram



Current (software) block diagram



System Overview

- User starts the program in MATLAB
- User is prompted to choose one of three modes of operation
    - Enter **training mode** if the input is “New User”
        - Creates a new profile in the database consisting of a .csv file containing user voice samples and a support vector machine (SVM) model trained from those samples. (see slide 16)
    - Enter **verification mode** if the input is an existing username
        - Checks a voice sample against a single specified profile and returns a positive or negative
    - Enter **identification mode** if the input is “whoiam”

Training mode

- Record a user speaking a scripted phase (e.g "bookcase")
    - Acquire 5 samples for SVM's training mode
    - Increase functionality with larger database
- Perform silence/white space truncation from input signal
- Extract features from each recording
- Artificially generate larger data set for each user (see slide 15)
- Use data set to train an SVM for that user (see slide 16)

Verification Mode

Take Recording

Remove whitespace/silence from recordings

Extract features

Check against SVM model associated with that user

Return result (positive or negative match)

Identification Mode

Take Recording

Remove whitespace/silence from recordings

Extract features

Checks sample against all users in the database and returns most likely user.

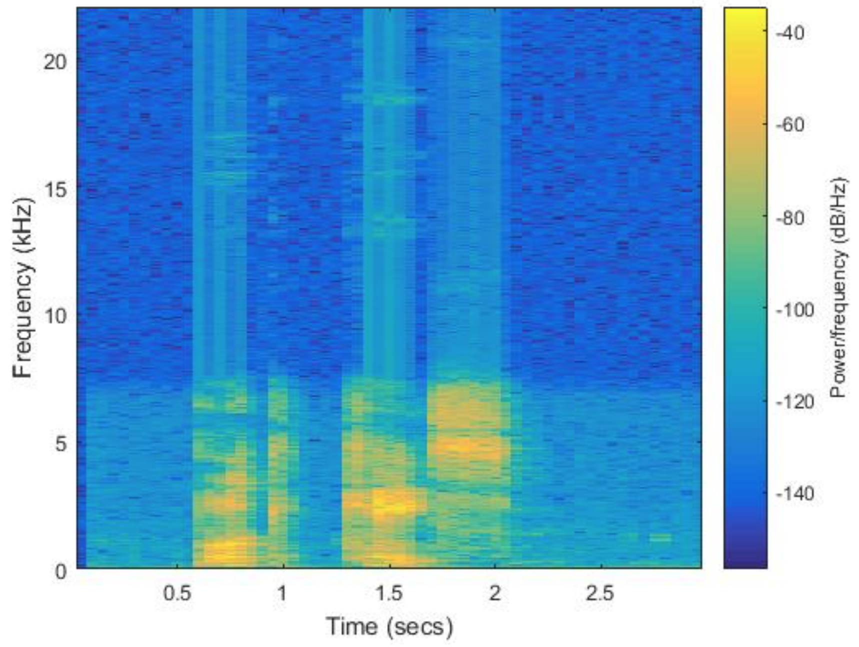
Closest match: user whose SVM classifier gives the largest positive classification score

Silence/ Whitespace Truncation

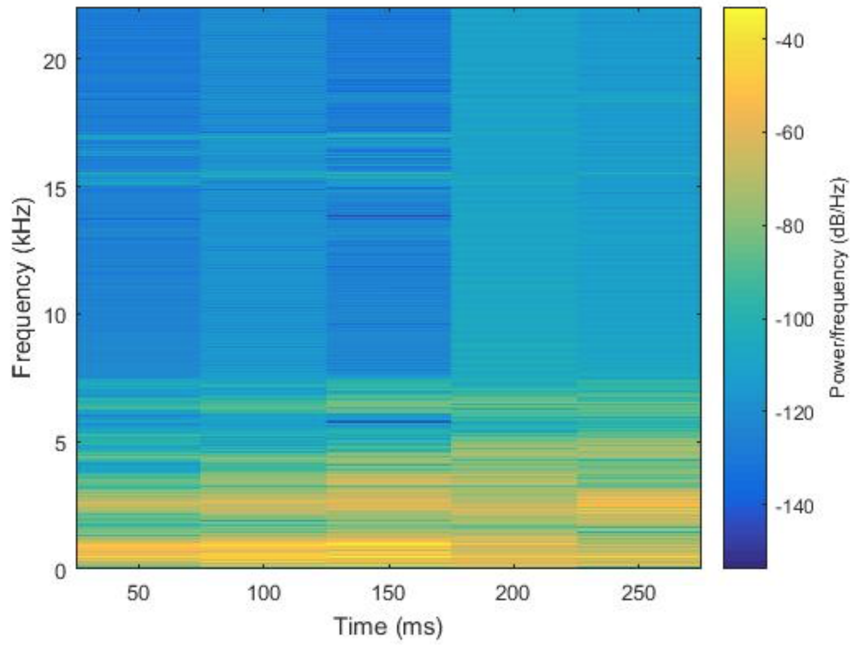
Sets a threshold power and if the sample is below that power it is removed.

Eliminates blank space before speech so that the samples lineup in time.

Spectrogram: Before vs. After Truncation



Phil’s Voice Sample before Silence Truncation



After

Feature Extraction

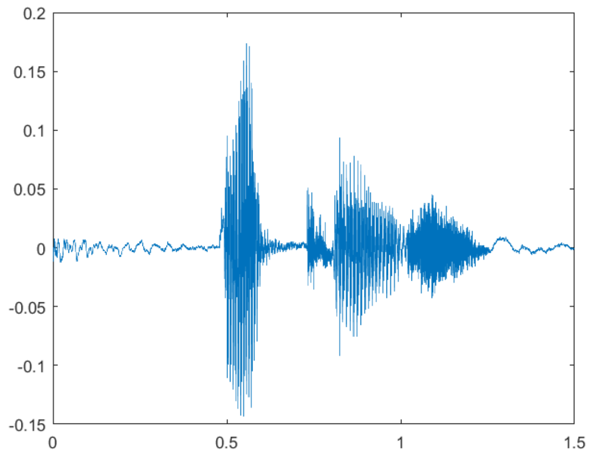
Create spectrogram of input signal

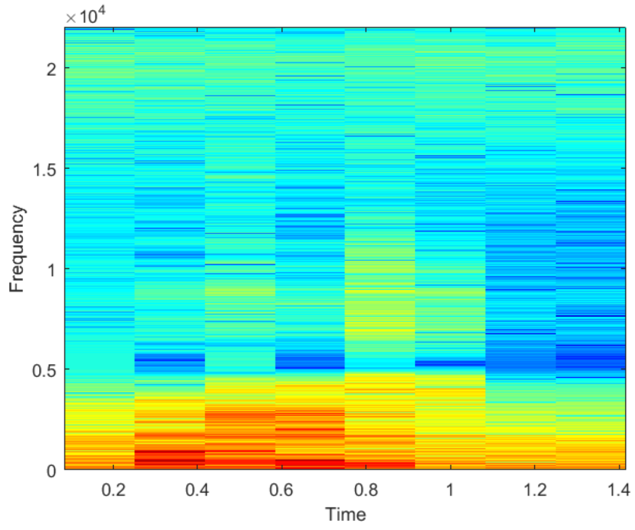
- Divide input signal into 8 frames, 50% overlap
- Apply Hamming window to reduce freq. leakage
- Calculate FFT for each frame

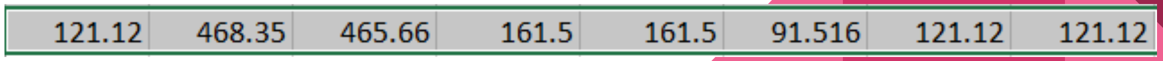
For each frame, find frequency with highest power

A 8d feature vector is obtained by finding for each frame the frequency bin with the highest power content

Feature Extraction







121.12	468.35	465.66	161.5	161.5	91.516	121.12	121.12

Creating synthetic data points

Take feature vectors extracted from recordings

Fit normal distribution parameters (mean and variance)

X_i ~ N(mean_i, var_i)

Generate many additional samples within those parameters

Training the Support Vector Machine (SVM)

The SVM is a binary classifier and is used for supervised machine learning

Each user has their own SVM associated with them

To train the SVM:

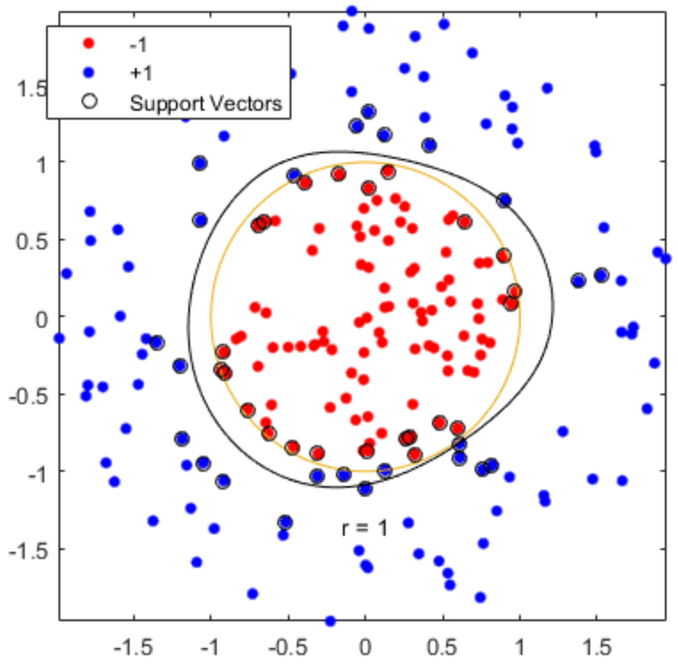
- Read in extracted features for that user. These are part of the “positive” class
- Read in extracted features for all other users. These are part of the “negative” class

Each feature vector is plotted as a coordinate in an n-dimensional hyperspace

The MATLAB SVM algorithm fits a boundary between the positive and negative classes

To verify users, a new unclassified sample is checked against the user's SVM

Training the SVM (MATLAB)



What does our system look like now?

Control flow of program works

Recording voice samples works

Feature extraction gives inconsistent results

Adding synthetic data points works…

- But gives some extraneous results due to input variance

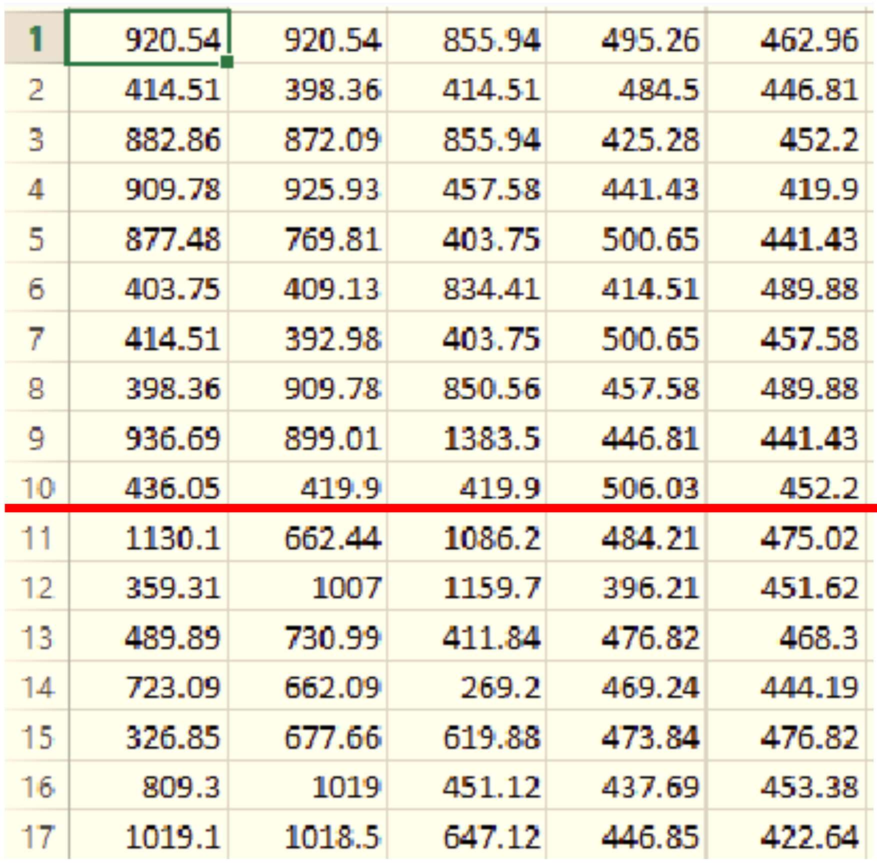
Why do we think our system failed?

- Even with the same person saying the same phrase, there are variances between recording instances, even for the same user.
  - E.g. saying the phrase slightly faster or slightly slower, different inflection
  - People get “impatient” during the training phase

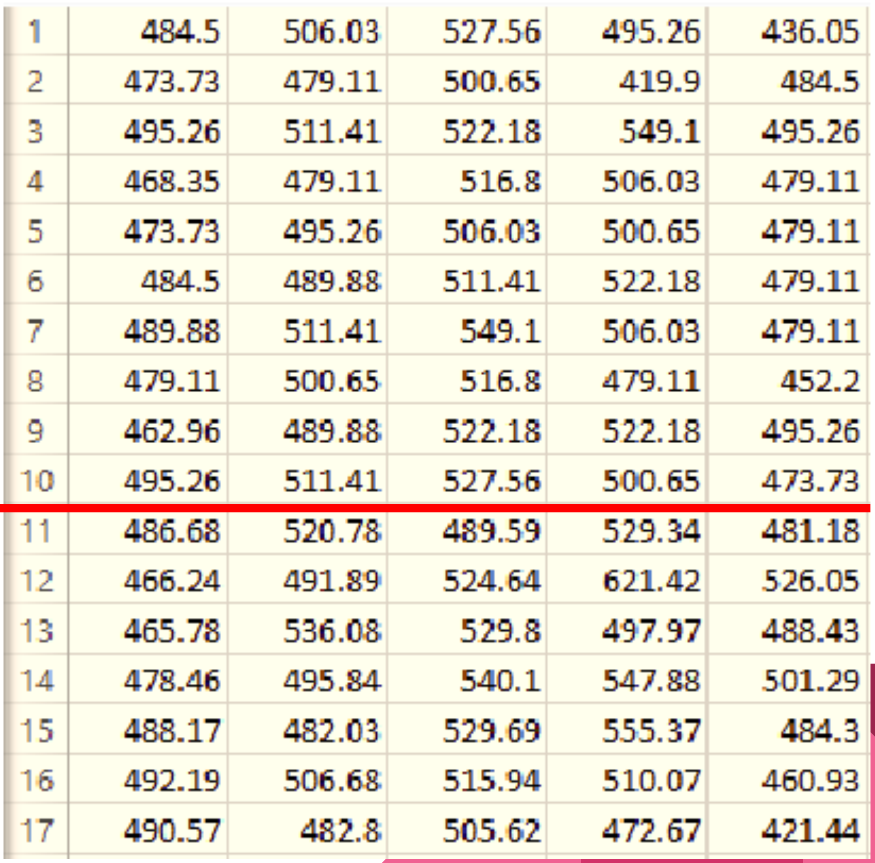
Artificial generation of data points did not compensate instances of high variance in user samples

Normal distribution might not be the best model for generating new samples

Example of data with high variance (left) versus low variance (right)



1	920.54	920.54	855.94	495.26	462.96
2	414.51	398.36	414.51	484.5	446.81
3	882.86	872.09	855.94	425.28	452.2
4	909.78	925.93	457.58	441.43	419.9
5	877.48	769.81	403.75	500.65	441.43
6	403.75	409.13	834.41	414.51	489.88
7	414.51	392.98	403.75	500.65	457.58
8	398.36	909.78	850.56	457.58	489.88
9	936.69	899.01	1383.5	446.81	441.43
10	436.05	419.9	419.9	506.03	452.2
11	1130.1	662.44	1086.2	484.21	475.02
12	359.31	1007	1159.7	396.21	451.62
13	489.89	730.99	411.84	476.82	468.3
14	723.09	662.09	269.2	469.24	444.19
15	326.85	677.66	619.88	473.84	476.82
16	809.3	1019	451.12	437.69	453.38
17	1019.1	1018.5	647.12	446.85	422.64



1	484.5	506.03	527.56	495.26	436.05
2	473.73	479.11	500.65	419.9	484.5
3	495.26	511.41	522.18	549.1	495.26
4	468.35	479.11	516.8	506.03	479.11
5	473.73	495.26	506.03	500.65	479.11
6	484.5	489.88	511.41	522.18	479.11
7	489.88	511.41	549.1	506.03	479.11
8	479.11	500.65	516.8	479.11	452.2
9	462.96	489.88	522.18	522.18	495.26
10	495.26	511.41	527.56	500.65	473.73
11	486.68	520.78	489.59	529.34	481.18
12	466.24	491.89	524.64	621.42	526.05
13	465.78	536.08	529.8	497.97	488.43
14	478.46	495.84	540.1	547.88	501.29
15	488.17	482.03	529.69	555.37	484.3
16	492.19	506.68	515.94	510.07	460.93
17	490.57	482.8	505.62	472.67	421.44

Where’s our hardware?

Our project became very software-oriented

Possible hardware components

- Wireless Microphone with pcb power supply
- Implementation of algorithm(s) on Raspberry Pi

Specs

Category	Proposed	Achieved
Classification accuracy	>90% accuracy	<50% accuracy
Database	SQL database	Unstructured data
Latency	<1s	~1s

Moving forward...

New Specs (for post-FPR)

Category	Proposed
Classification accuracy	Able to recognize Paul, Liam, Phil, Jason with >80% accuracy
Power supply	2.5A @ 5V

Questions?