Speaker Signals Recognition

TuanKhai Nguyen

tran.tuan.khai.nguyen@vanderbilt.edu

EECE 5356 – Digital Signal Processing

1. **Overview**

We built:

**main.m** – big script contain all the helper functions. Run main.m by *Run sections*, not full script, to get what we want.

**createDatabase.m** – create a matrix that concatenates training signals of 1 subject.

Input: NONE, but user have to choose folder with all voice signals of a subject when prompted.

Output: train matrix of that subject.

**processVoice.m –** Do all the Fourier transform, spectrogram, and normalization for 1 signal.

Input: file name for voice signal of 1 subject

Output: processed signal of 1 subject

**calcScore.m** – calculate the difference score between a voice signal and a voice matrix of 1 subject

Input: 1 unknown signal, 1 train matrix of a subject

Output: difference score

**classify.m** – classify sound signals of unknown subject in train set.

Input: a) All train matrices b) folder of unknown subject to classify

Output: Classification + score of each file.

**classifyUnknown.m –** similar to classify.m but for unknown subjects NOT in train set.

We design it this way since voice signals do not have the same lengths which did not allow concatenation / grouping of all processed voices into 1 big matrix. Every subject has their own training matrix.

1. **Classify unknowns of subjects included in train set.**

**Step 1**: Run first section of **main.m** to get all training matrices. We have 15 training subjects so 15 matrices to be obtained:

Graphical user interface, text, application

Description automatically generated

**Step 2**: We can save training data and reload for future use. Run the next section of **main.m**:

Graphical user interface, text, application

Description automatically generated

**Step 3:** Run the ‘classify’ section in **main.m,** which calls **classify.m** and prompts user to pick a folder containing the ‘.wav’ files.

Graphical user interface, text, application

Description automatically generated

When chosen, **classify.m** will (1) pick out the files used for testing, (2) compute the difference score against all 15 training matrices (3) Print out which training set/matrix it belongs to:

Graphical user interface, text, application

Description automatically generated

We did the same for all other folders, results are shown below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Filename** | **Class/score** | **Filename** | **Class/score** | **Filename** | **Class/score** |
| Alex4 | Alex(train1)/ 0.088494 | Ben4 | Ben(train2)/  0.080054 | Jacob4 | Jacob(train3)/  0.121352 |
| Alex5 | Alex(train1)/ 0.089521 | Ben5 | Ben(train2)/  0.084463 | Jacob5 | Jacob(train3)/  0.125160 |
| Alex6 | Alex(train1)/ 0.099285 | Ben6 | Ben(train2)/  0.070069 | Jacob6 | Jacob(train3)/  0.111311 |
| Maddie4 | Maddie(train4)/  0.103010 | Matt4 | Matt(train5)/  0.086573 | Akash4 | Akash (train6)/  0.047665 |
| Maddie5 | Maddie(train4)/  0.104005 | Matt5 | Matt(train5)/  0.095349 | Akash5 | Akash (train6)/  0.046104 |
| Maddie6 | Maddie(train4)/  0.114670 | Matt6 | Matt(train5)/  0.089152 | Akash6 | Akash (train6)/  0.047501 |
| Daniel4 | Daniel(train7)/  0.065633 | Eliza4 | Eliza(train8)/  0.067229 | Google4 | Google(train9)/  0.034247 |
| Daniel5 | Daniel(train7)/  0.062741 | Eliza5 | Eliza(train8)/  0.085958 | Google5 | Google(train9)/  0.032982 |
| Daniel6 | Daniel(train7)/  0.072744 | Eliza6 | Eliza(train8)/  0.069994 | Google6 | Google(train9)/  0.036938 |
| Steven4 | Steven(train10)/  0.045543 | Brian4 | Brian(train11)/  0.059301 | Comp4 | Comp(train12)/  0.036301 |
| Steven5 | Steven(train10)/  0.066062 | Brian5 | Brian(train11)/  0.059043 | Comp5 | Comp(train12)/  0.045199 |
| Steven6 | Steven(train10)/  0.052815 | Brian6 | Brian(train11)/  0.060871 | Comp6 | Comp(train12)/  0.032081 |
| Grant4 | Grant(train13)/  0.061090 | Jay4 | Jay(train14)/  0.057707 | Sara4 | Sara(train15)/ 0.085842 |
| Grant5 | Grant(train13)/  0.055018 | Jay5 | Jay(train14)/  0.060371 | Sara5 | Sara(train15)/  0.079005 |
| Grant6 | Grant(train13)/  0.064005 | Jay6 | Jay(train14)/  0.047832 | Sara6 | Sara(train15)/  0.080588 |

Full log is displayed here:

(I went to Temple before Vanderbilt so the Temple OneDrive folder)

Text

Description automatically generated

Text

Description automatically generated

1. **Classify unknowns of subjects NOT included in train set.**

Some initial experiments show that our classification method from **part I.** produces relatively consistent results on which train set where subjects, NOT coming from the train set, are the closest to, except for the variance for DavidR.

We can come up with a simple way that make use of the scoring function developed from part I.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Alex (train1) | 0.092433± 0.005956 | Ben (train2) | 0.078195±  0.007375 | Jacob (train3) | 0.119274±  0.007155 |
| Maddie (train4) | 0.107228±  0.006464 | Matt (train5) | 0.090358±  0.004511 | Akash (train6) | 0.047090±  0.000858 |
| Daniel (train7) | 0.067039± 0.005148 | Eliza (train8) | 0.074394±  0.010110 | Google (train9) | 0.034723±  0.002020 |
| Steven (train10) | 0.054807± 0.010404 | Brian (train11) | 0.059739±  0.000990 | Comp (train12) | 0.037860±  0.006696 |
| Grant (train13) | 0.060038±  0.004585 | Jay (train14) | 0.055303±  0.006606 | Sara (train15) | 0.081812±  0.003579 |

The table shows the mean and standard deviation of the score of 3 test samples from part 1 computed against their respective train matrix. For example, jay4, jay5, jay6 has a mean score of 0.055303 and standard deviation 0.006606 to the big Jay train matrix, as shown in table.

Thus, alternatively, we can view **a score** as the **distance** from the center of a cluster (ex: Jay) to a point of voice sound. This distance has the mean and standard deviation of the score. This distance (with standard deviation) can be viewed as a radius of a sphere representing the cluster if we choose to visualize in 3D but it can be at higher dimensions.

Diagram

Description automatically generated

Figure . Simple illustration of distance-from-cluster-center idea.

1. So we can reuse the **score** function, determine the minimum score and assign it to a train set. This score will be the distance of the point to the center of the train set.
2. Run the section in from **main.m** with classifyUnknown to get the scores for unknown sounds in speakerDataSet7.
3. Then, we check if it is within the radius of the sphere i.e. belongs to the cluster, using the mean and standard deviation of the distance from each cluster’s center (from the table above).

We try out with WillR’s, which the algorithm from part 1 classifies them to belong to Jay cluster.

Text

Description automatically generated

Figure . We have 50% of these scores are inside Jay’s cluster (train14) with radius   
0.055303 ± 0.006606 = [0.048697 0.061909]. Accuracy 50%

Results for other subjects and accuracy of classifying whether it belongs to the cluster is shown below:

Text

Description automatically generated

Figure . We have NONE of these scores are inside 0.055303 ± 0.006606 = [0.048697 0.061909]. So all are outside of Jay’s cluster (train14). Accuracy 100%

Text

Description automatically generated

Figure . BrandiC got classified as belonging to Eliza (train8), radius 0.074394 ± 0.010110 = [0.064248 0.084504]. So NONE of BrandiC sounds are inside the sphere. Accuracy 100%.

Text

Description automatically generated

Figure . ErinL classified as belonging to Eliza (train8), radius 0.074394 ± 0.010110 = [0.064248 0.084504]. So 2 of ErinL sounds are inside the sphere. Accuracy 66.6%.

Text

Description automatically generated

Figure . DavidR classified as belonging to Grant (train13) radius 0.060038 ± 0.004585, with max radius **0.064623** and Jay (train14) radius 0.055303 ± 0.006606 with max radius **0.061909**. So NONE of DavidR sounds are inside any of these 2 spheres. Accuracy 100%.