CS669 Assignment 2

Submitted by: Prachi Sharma

Branch: MSc Applied Mathematics

Roll No.: V21078

Date: 21/11/2022

In this assignment, you will develop a language identification (LID) system using Gaussian mixture models (GMM.) 39-dimensional (d = 39) Mel frequency cepstral coefficients (MFCCs) are used as the feature representation. These MFCC features are of varying length (depending on the duration of the speech utterance from which they were derived.) This is achieved by using a voice activity detector algorithm (which itself is based on a GMM.)

1. **System 1.** This system uses a GMM to model each class conditional density. This is done using the EM algorithm.

   Link to code notebook:

   https://colab.research.google.com/drive/1SQ5PR2eG7JXZKxOEzKuj3NfIRTfMBEbv?usp=sharing

2. **System 2.** This is a UBM-GMM system. Pool the data of all classes to form a large GMM, called the universal background model (UBM.) From the UBM, class-specific GMMs are built using MAP adaptation. Only the means are to be adapted, and other parameters (Σk, πk) are used as such from the UBM.

   Link to code notebook:

   https://colab.research.google.com/drive/1fjfnU_oXpdwZ3dlw2cgehZH_rcaaDX0h?usp=sharing

Questions.

1. Which system (1 or 2) performs better and why?

   Ans. Accuracy values for system 1 are better than system 2. The reason could be that we have used N=13 for separate data and N=39 for pooled data which is very large in size. For higher values, like N=128, 256 or 512, GMM-UBM model may have reached a better accuracy. Due to system limitations, it was not possible to apply such high values of N.

   Following table has accuracies for GMM system and GMM-UBM system.

Language	GMM (N=13)		GMM-UBM (N=39)	
	PB_test	YT_test	PB_test	YT_test
0.Asm	0.9498607242339833	0.26666666666666	0.707520891364902	0.26666666666666
1.Ben	0.9441340782122905	0.14444444444444	0.877094972067039	0.14444444444444
2.Eng	0.8879310344827587	0.15873015873015	0.896551724137931	0.15873015873015
3.Guj	0.9776536312849162	0.0	0.837988826815642	0.0
4.Hin	0.88268156424581	0.0	0.435754189944134	0.0
5. Kan	0.8629441624365483	0.03314917127071	0.796954314720812	0.03314917127071
6. Mal	0.8622448979591837	0.03333333333333	0.688775510204081	0.03333333333333
7. Mar	0.7948717948717948	0.07563025210084	0.794871794871794	0.07563025210084
8. Odi	0.9396984924623115	0.0	0.798994974874371	0.0
9. Pun	0.4838709677419355	0.0	0.604838709677419	0.0
10. Tam	0.776	0.017094017094	0.744	0.017094017094
11. tel	0.9072164948453608	0.0	0.407216494845360	0.0

2. How does performance vary with the number of mixtures in the GMM? Give a meaningful plot.

Ans. As can be seen from the plot, with an increasing number of mixtures, accuracy is improved for each language in all PB and most of the YT test data.

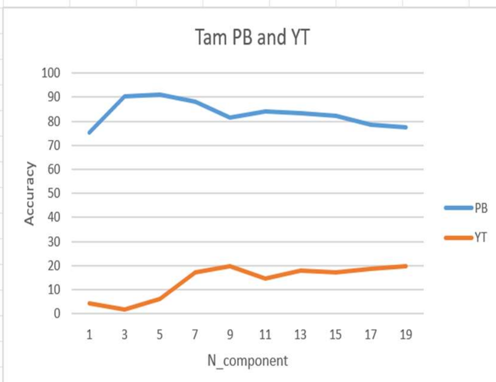
Accuracy for different mixtures in PB test data

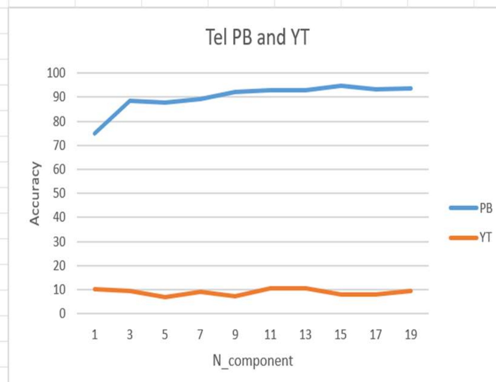
	0	1	2	3	4	5	6	7	8	9	10	11
1	5	7.777778	9.52381	0	1.104972	2.209945	10.55556	5.042017	32.77778	0	4.273504	10.11236
3	15.55556	11.11111	12.69841	0	0	11.04972	9.444444	0	30	0	1.709402	9.550562
5	7.777778	7.777778	16.66667	0	0	14.91713	5	3.361345	38.33333	0	5.982906	6.741573
7	12.22222	6.666667	12.69841	0	0	9.392265	6.111111	3.361345	31.66667	0	17.09402	8.988764
9	12.77778	4.444444	8.730159	0	0	6.629834	7.222222	4.201681	31.11111	0	19.65812	7.303371
11	13.33333	6.111111	16.66667	0	0	7.734807	6.666667	4.201681	26.66667	0	14.52991	10.67416
13	16.11111	6.666667	12.69841	0	0	10.49724	5.555556	5.042017	24.44444	0	17.94872	10.67416
15	19.44444	7.222222	13.49206	0	0	13.25967	3.888889	5.042017	21.11111	0	17.09402	7.865169
17	13.88889	7.222222	11.90476	0	0	12.1547	2.777778	5.042017	22.77778	0	18.80342	7.865169
19	15.55556	7.777778	23.80952	0	0	7.734807	3.888889	5.042017	17.22222	0	19.65812	9.550562

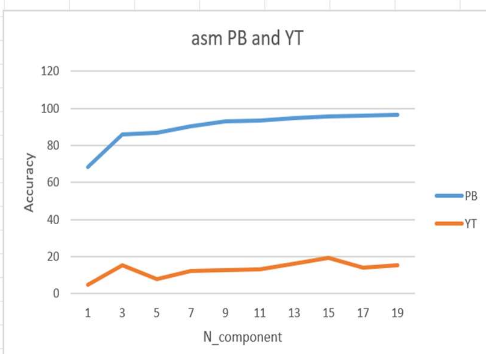
Accuracy for different mixtures in YT test data

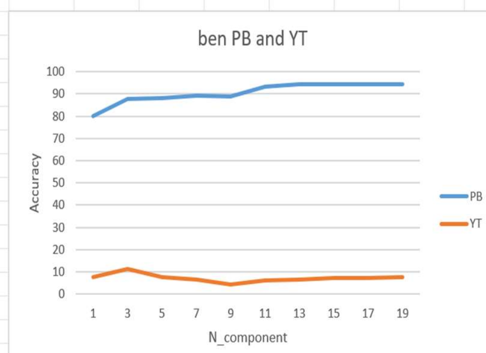
	0	1	2	3	4	5	6	7	8	9	10	11
1	68.24513	79.88827	84.48276	84.9162	69.27374	65.98985	70.40816	58.11966	90.95477	48.3871	75.2	74.74227
3	85.79387	87.7095	87.93103	89.94413	81.00559	76.64975	77.55102	77.77778	88.94472	48.3871	90.4	88.65979
5	86.90808	88.26816	87.93103	93.85475	84.35754	82.2335	82.65306	72.64957	92.46231	48.3871	91.2	87.62887
7	90.52925	89.38547	88.7931	97.2067	86.03352	86.80203	83.67347	76.92308	94.47236	48.3871	88	89.17526
9	93.03621	88.82682	88.7931	96.64804	84.9162	84.26396	82.65306	78.63248	94.47236	48.3871	81.6	92.26804
11	93.59331	93.29609	90.51724	98.32402	88.26816	86.29442	85.71429	75.21368	93.46734	48.3871	84	92.78351
13	94.98607	94.41341	88.7931	97.76536	88.82682	86.29442	86.22449	79.48718	93.96985	48.3871	83.2	92.78351
15	95.54318	94.41341	90.51724	97.76536	88.26816	86.80203	87.7551	82.05128	95.47739	48.3871	82.4	94.84536
17	96.10028	94.41341	90.51724	97.2067	87.15084	86.80203	86.22449	80.34188	93.96985	48.3871	78.4	93.29897
19	96.65738	94.41341	89.65517	97.2067	86.03352	89.84772	87.2449	78.63248	92.96482	48.3871	77.6	93.81443

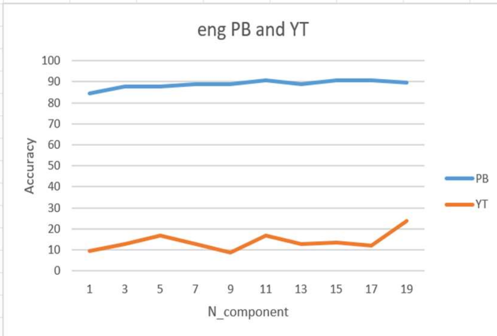
Accuracy plots for each language with Accuracy on Y-axis and Number of mixtures on X-axis.

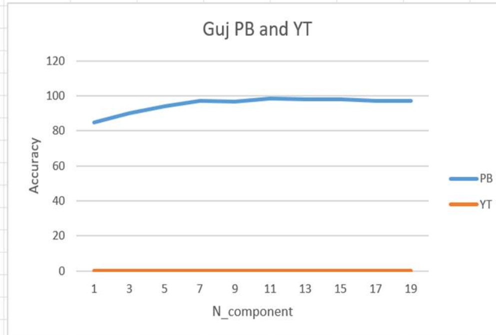


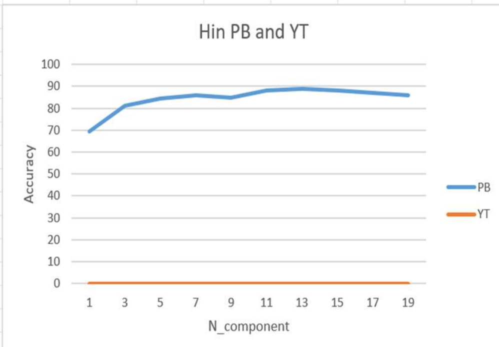


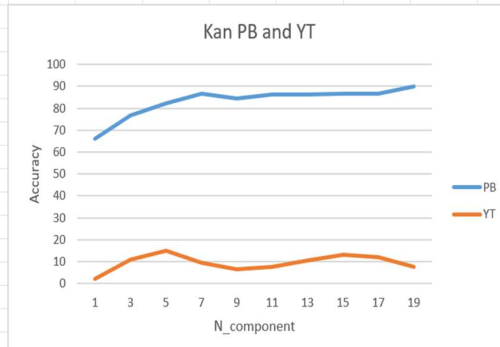


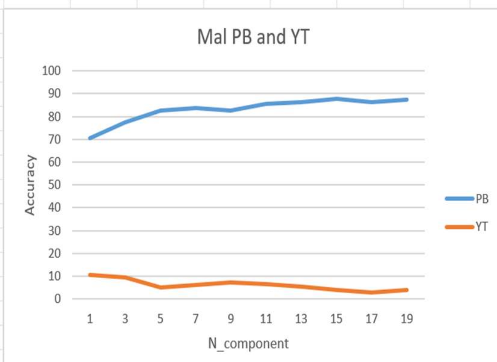


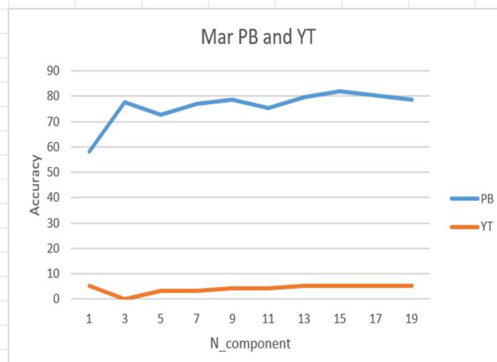

eng PB and YT
Accuracy
100
90
80
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT

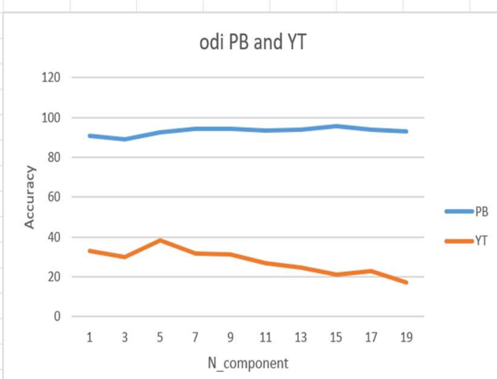

Guj PB and YT
Accuracy
120
100
80
60
40
20
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT

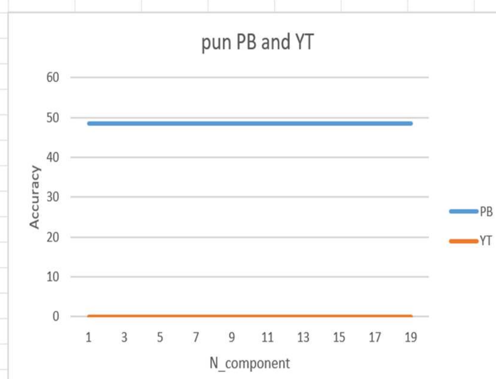

Hin PB and YT
Accuracy
100
90
80
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT


Kan PB and YT
Accuracy
100
90
80
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT


Mal PB and YT
Accuracy
100
90
80
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT


Mar PB and YT
Accuracy
90
80
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT


odi PB and YT
Accuracy
120
100
80
60
40
20
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT


pun PB and YT
Accuracy
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19
N_component
PB
YT

3. Is it better to use a full covariance matrix or a diagonal covariance matrix in the GMM?

In case of full covariance matrix, results are better when compared to diagonal covariance matrix. In many cases, diagonal covariance matrix can give rise to singularities.

Language	Diagonal Covariance, GMM (N=13)		Full Covariance, GMM (N=13)	
	PB_test	YT_test	PB_test	YT_test
Asm	0.7047353760445683	0.2	0.9498607242339833	0.26666666666666
Ben	0.837988826815642	0.133333333	0.9441340782122905	0.14444444444444
Eng	0.879310344827586	0.166666666	0.8879310344827587	0.15873015873015
Guj	0.888268156424581	0.0	0.9776536312849162	0.0
Hin	0.575418994413407	0.0	0.88268156424581	0.0
Kan	0.715736040609137	0.08287292817679558	0.8629441624365483	0.03314917127071
Mal	0.714285714285714	0.11111111	0.8622448979591837	0.03333333333333
Mar	0.623931623931623	0.050420168067226	0.7948717948717948	0.07563025210084
Odi	0.909547738693467	0.272222222	0.9396984924623115	0.0
Pun	0.483870967741935	0.0	0.4838709677419355	0.0
Tam	0.904	0.025641025641025	0.776	0.017094017094
tel	0.711340206185567	0.016853932584269	0.9072164948453608	0.0

4. Compare the performance on PB test and on YT test. Why is there a difference?

Ans. PB test data has better accuracy as compared to YT data at all levels of gaussian mixtures as can be seen through the plots as well as accuracy table. There could be two possible reasons. Firstly, the training has been done on PB data. So due to more similarity, PB test data shows better results. Secondly, YT test data has discrepancies like for instance, Assamese language data also contains other language data, which might’ve caused less accuracy.

5. Which languages are confusable and why?

Similar languages like Assamese, Bengal, and Odia are confusable. Apart from that, the confusion matrix (columns have predicted label, rows have true label) is showing that Punjabi is being confused for Gujrati, Bengali for Assamese and so on for PB data. Similar inferences can be seen for YT test data.

```
Confusion matrix for PB data

   0     1    2    3    4    5    6    7    8    9   10   11
0:[[341.   0.   6.   1.   0.   0.   0.   0.   4.   0.   4.   3.]
 1:[  5. 169.   0.   0.   2.   0.   0.   0.   3.   0.   0.   0.]
 2:[  4.   0. 103.   0.   3.   1.   1.   1.   3.   0.   0.   0.]
 3:[  1.   0.   0. 175.   0.   0.   0.   0.   0.   0.   1.   2.]
 4:[  1.   2.   9.   2. 158.   0.   0.   3.   1.   1.   0.   2.]
 5:[  4.   1.   7.   4.   0. 170.   1.   0.   7.   0.   1.   2.]
 6:[  8.   1.   3.   2.   2.   0. 169.   1.   9.   0.   0.   1.]
 7:[ 19.   0.   0.   0.   0.   0.   1.  93.   2.   0.   0.   2.]
 8:[  1.   2.   1.   0.   1.   0.   1.   1. 187.   0.   0.   5.]
 9:[  0.   0.   0.  60.   3.   0.   0.   0.   0.  60.   0.   1.]
 10:[  0.   3.   0.  16.   0.   2.   1.   0.   4.   0.  97.   2.]
 11:[  2.   1.   1.   4.   2.   0.   3.   0.   4.   0.   1. 176.]]
```

```
Confusion matrix for YT data

      0  1    2   3   4   5   6   7  8    9  10  11
0: [[29. 16. 22.  0.  0.  1. 25.  0. 73.  0.  0. 14.]
1:[ 4. 12. 40.  0.  0. 14. 34. 15. 38.  0.  0. 23.]
2: [ 7.  3. 16.  0.  0. 35. 27.  0. 35.  0.  0.  3.]
 3:[ 1.  0. 24.  0. 13. 16. 83. 18.  8.  0.  7. 11.]
 4:[ 2.  1.  3.  0.  0. 27. 59.  0. 66.  0.  0. 23.]
 5:[22.  7. 37.  0.  3. 19. 28. 17. 40.  0.  0.  8.]
 6:[ 1. 35. 10. 25. 12. 20.  9. 19. 21.  2. 18.  8.]
 7:[ 8.  0. 19.  0.  0.  2. 42.  6. 26.  0.  0. 16.]
 8:[32. 21. 29.  0.  0.  2. 10. 35. 44.  0.  0.  7.]
 9:[13.  0.  5.  0.  1.  0. 13. 11. 75.  0.  0.  3.]
 10:[ 5.  4.  4.  3. 27.  0. 15.  0. 33.  5. 21.  0.]
 11:[ 0. 11.  4. 11.  3. 11. 40. 32. 33.  0. 16. 17.]]
```