Computer Vision Project 4
Aravind Chandradoss
Face Recognition
(Identity, 50 male-50 Female)

The approaches tried
- one shot learning (Siamese network)
- model based (Correlation)

I tried using entire image as features, but accuracy were poor, so moved to Deep features.

In One shot learning approach, I tried 4 different cases. I used FaceNet (pretrained) to get the facial features (512 feature - facial encoding) and later used the encoding for identity recognition.

CASE A1: I used leave one out approach and used the difference (mahalanobis distance) in encoded features. Since, there were 25 images for comparison, **I got 100% classification accuracy** always **(results for randon test image repeated 10 time for each identity).**
--- Later on digging deeper, i found that all the model (Identity) are significantly separated by using **T-SNE plot** (reduced-PCA-2D plot is also shown for ref).

CASE A2: I used only one image as reference (*-001-*.bmp) and did the same. I got **accuracy around (91-94%).** Confusion matrix is shown below.

CASE A3: where all images in 1st session as ref showed **87-95% accuracy.** Test image from 2nd session, and ref image from 1st session

CASE A4: I used did the same, but used only image in 1st session as reference and test images are taken from 2nd session. **I got accuracy of 88 - 93%**. plots are shown.

Model Based: In this approach, I used the encoded image to create a model for each identity and used reconstruction error as the metric for identification.
The model was created using embedding from the first session and test image are taken from second session.
In model based, I tried 4 cases.

CASE B1: With all eigen bases as the model, and used reconstruct error as metric. I got accuracy around **95-99%.** (Due to random sampling while testing, I got a wide range, some time 91%)

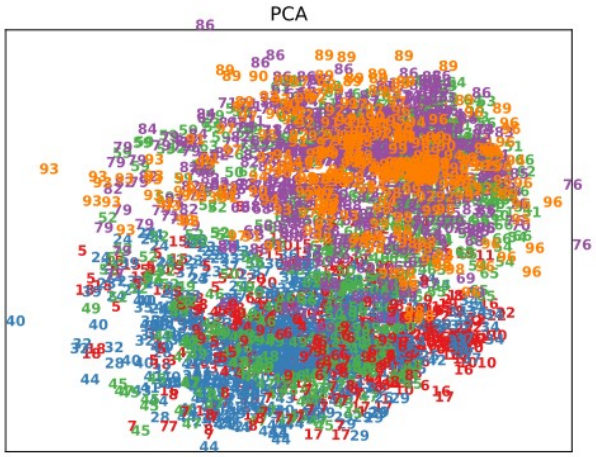
CASE B2: With only the first 10, I got around **97-98%.** Confusion matrix and results are shown.

CASE B3: With first 5 vectors, I got around **95-98%** classification accuracy.

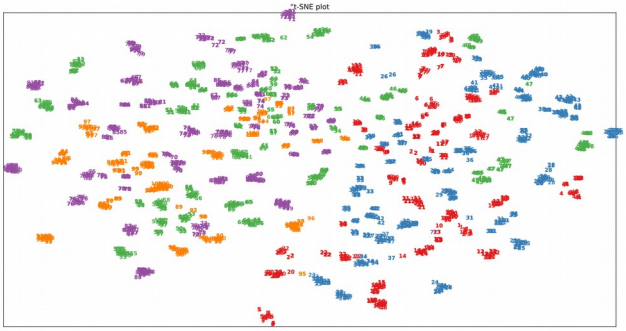
CASE B4: With first vector along, I got **94-97%** classification accuracy.

RESULTs:
Images are in svg format (feel free to look at attached file) (might be black in this pdf)

PCA 2D plot (less interpret able)

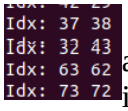


TSNE plot



We can clearly see the each identity is well separated in high dimension (512 dimension). This is reason for high accuracy in CASE A1 (distance metric gave the closest match).

RESULTS and INFERENCES:



Reason for CASE A1 to give high accuracy. (Image :: Left : index of test image, Right : Index of closest image). We can see that algorithm picked the most similar image (in fact the adjacent image) for identification. This is not the case, when I used Single image (*-001-*.bmp) or Model for classification.

The results for each cases as confusion matrix (Images might be small (have to plot for 100 identities), feel free to refer the attached svg file for more details)

One more thing, increase your brightness and "*clean*" your screen if you "could not see" mis-classified image in confusion matrix

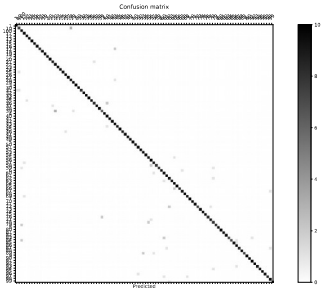
---

NOTE: Since, I used deep features, I did not plot the eigen faces

---

CASE A1: Not soo great plot (got 100% classification, refer Tsne plot)

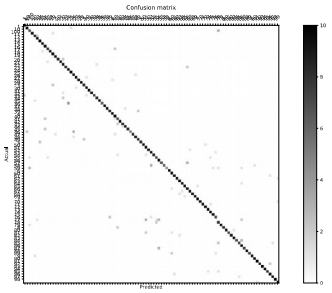
CASE A2:



Test: Random (both sess)
Ref: Single image

```
[100 rows x 100 columns]
correct: 938 incorrect: 62 = 0.938 %
```

CASE A3:

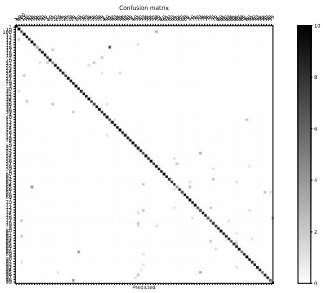


Test: 2nd sess (random)
Ref: 1st sess (all)

Acc= 87 - 95 %

```
[100 rows x 99 columns]
correct: 870 incorrect: 130 = 0.87 %
```

CASE A4:

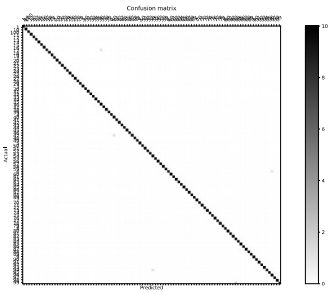


Test: 2nd Sess (random)
Ref: 1st sess (single image)

Acc = 88 – 93%

```
[100 rows x 100 columns]
correct: 888 incorrect: 112 = 0.888 %
```

CASE B1:



```
[100 rows x 100 columns]
correct: 994 incorrect: 6 = 0.994 %
```

Refer attachment to see the confusion matrix
Here, I have shown only the accuracy results

CASE B2:

```
[100 rows x 100 columns]
correct: 988 incorrect: 12 = 0.988 %
```

CASE B3

```
[100 rows x 100 columns]
correct: 959 incorrect: 41 = 0.959 %
```

CASE B4:

```
[100 rows x 100 columns]
correct: 941 incorrect: 59 = 0.941 %
```