Facial Expression Recognition with CNN

Xu Yan
Jing Si
Hanbo Li

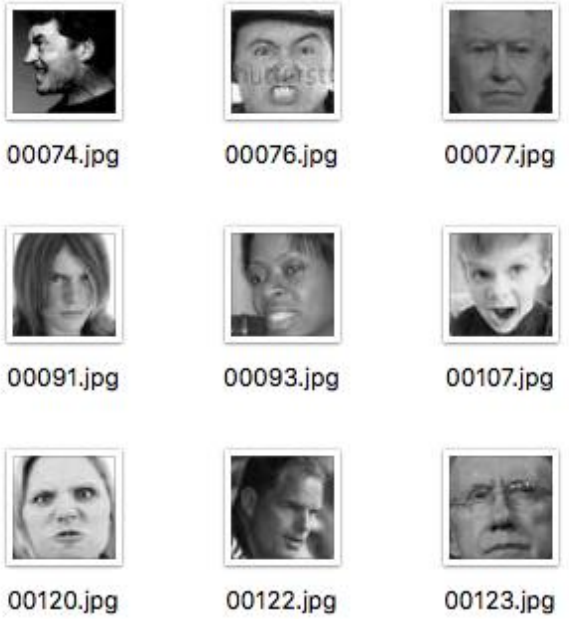
Tool -caffe

1. Lmdb format data set
2. Train.prototxt, solver.prototxt
3. Interface to parse log file and create graphs

Dataset -fer2013

Obtained from Kaggle, consists of 48x48 pixel grayscale images of faces. Each face is categorised to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).

- Training set: 28,709
- Validation set: 3,589
- Test set: 3,589



This dataset was prepared by Pierre-Luc Carrier and Aaron Courville, as part of an ongoing research project

Data Prepare

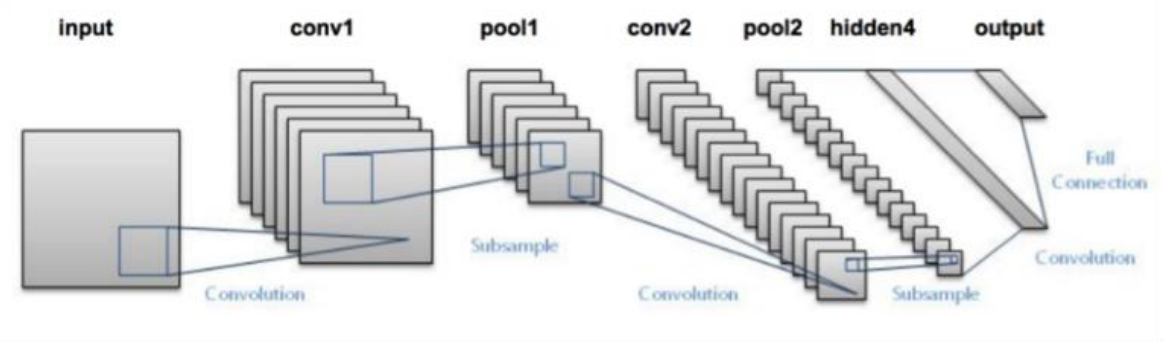
- Convert to JPG and store images to train, val, test folders
- Create auxiliary data- this is used as indexes when generate lmdb data.
- Generate lmdb files- modify create_imagenet.sh
- Generate image mean file -normalize the image data

Training models

- LeNet
- AlexNet

LeNet

- The LeNet architecture was first introduced by LeCun et al. in their 1998 paper
- The network of LeNet is quite simple: two convolution layers and one fully connected layer.

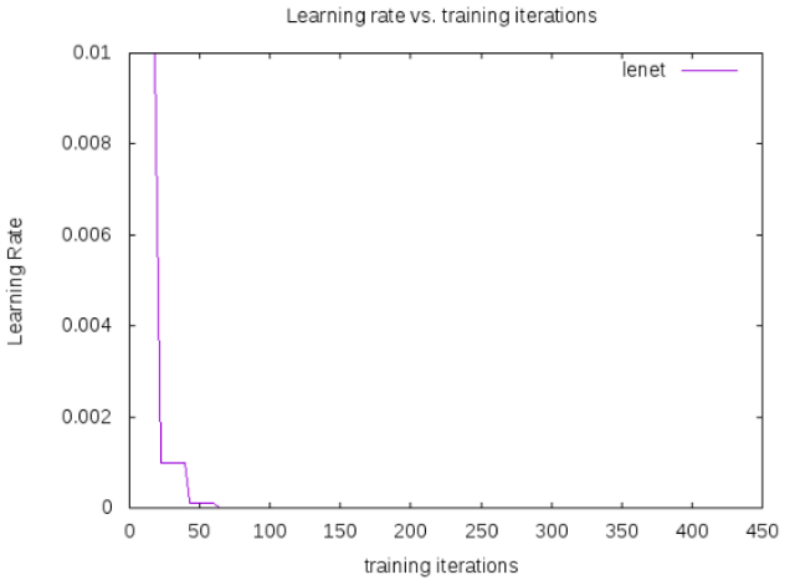


lenet_architecture-768x226.png

Training LeNet

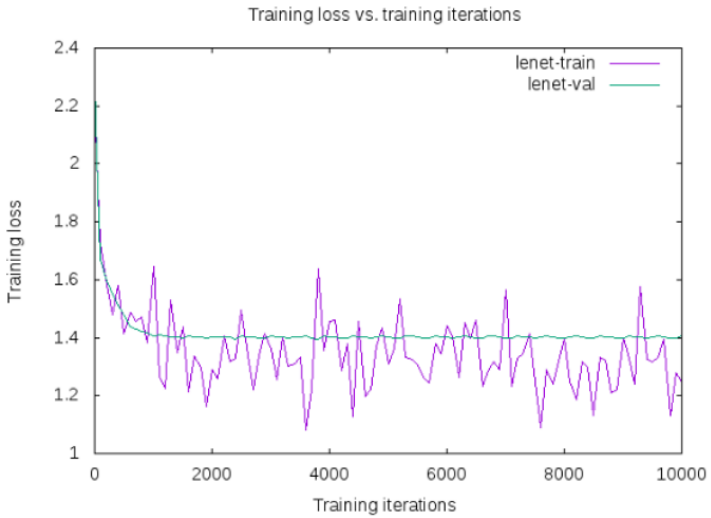
We first use the solver file we used in exam2 and use max iteration equals 10000.

As we can see, the learning rate drops two fast and we can hardly know if the network still learning something after 100 iteration.

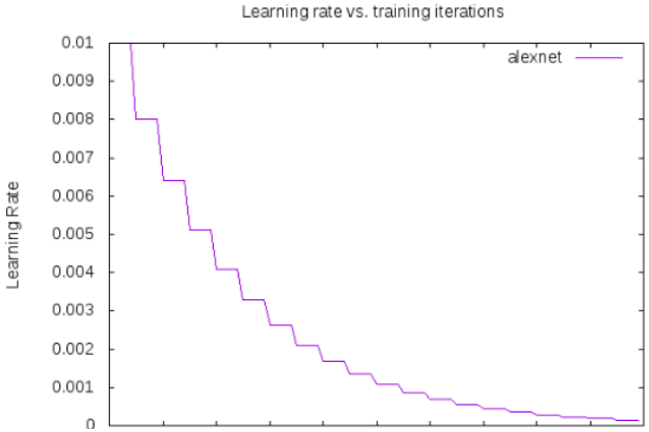


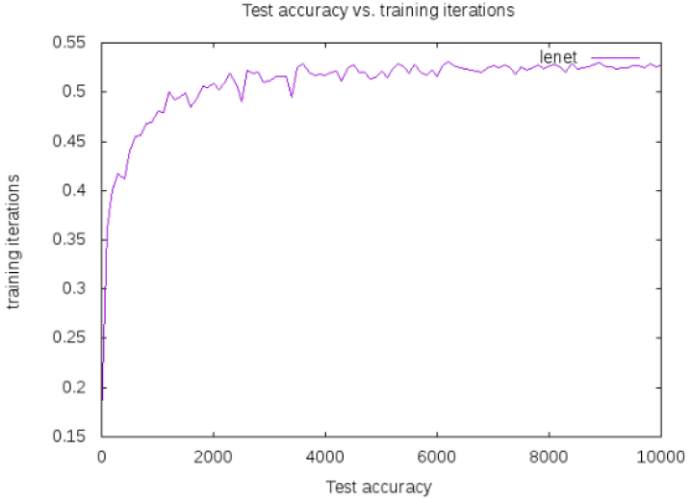
Training LeNet

This bad learning rate also leads to a very bad result.



Training LeNet



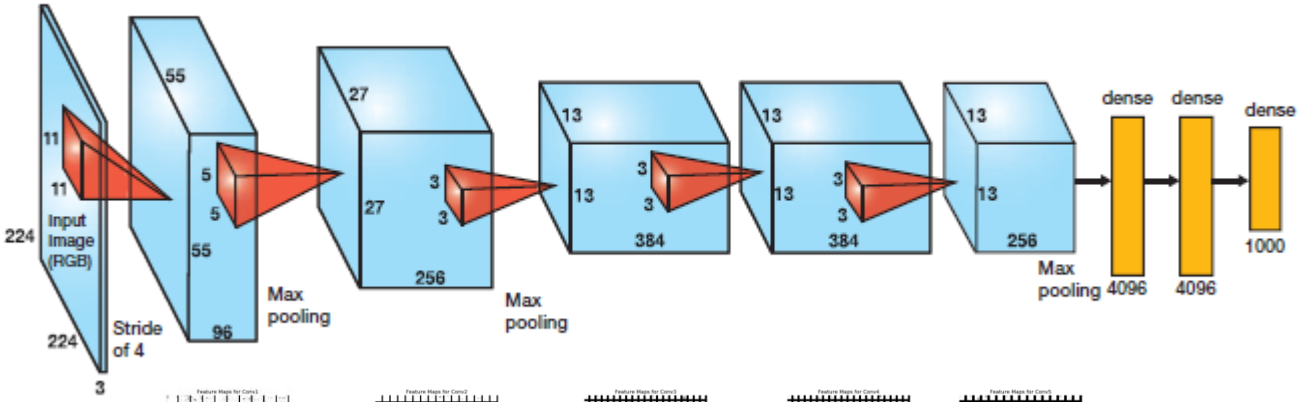


AlexNet

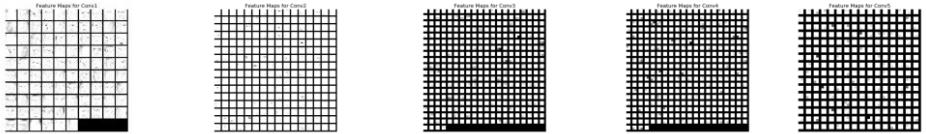
- The first work that popularized Convolutional Networks in Computer Vision was the **AlexNet**, developed by Alex Krizhevsky, Ilya Sutskever and Geoff Hinton.
- The AlexNet was submitted to the ImageNet ILSVRC challenge in 2012 and significantly outperformed the second runner-up (top 5 error of 16% compared to runner-up with 26% error).
- The Network had a very similar architecture to **LeNet**, but was deeper, bigger, and featured Convolutional Layers stacked on top of each other (previously it was common to only have a single CONV layer always immediately followed by a POOL layer).

AlexNet

- 5 convolution layers and 3 fully connected layers
- Uses ReLu instead of a Tanh or Sigmoid function
- Using a Dropout layer after every FC layer



Feature map of each conv-layer:



Training

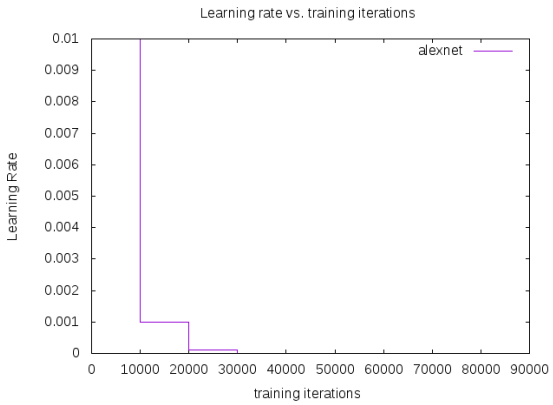
3 AlexNets with different parameters:

	Iterations	**Training batch size**	**Gamma (lr_policy:step)**
AlexNet 1	50000	100	0.1
AlexNet 2	50000	**256**	0.1
AlexNet 3	50000	100	**0.95**

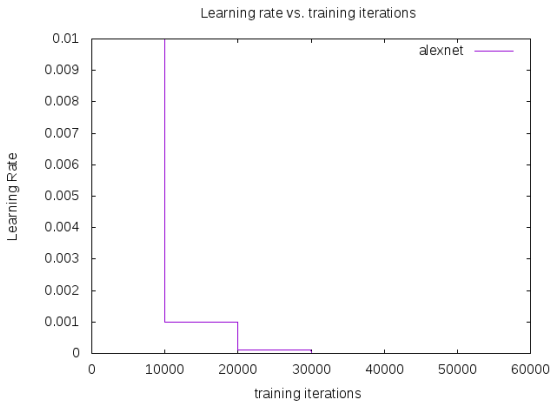
Lr_policy: step
  base_lr * gamma ^ (floor(iter / stepsize))

Training

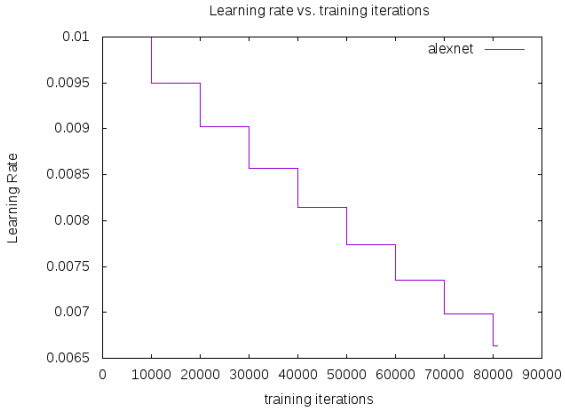
Learning rate vs. training iterations



AlexNet 1



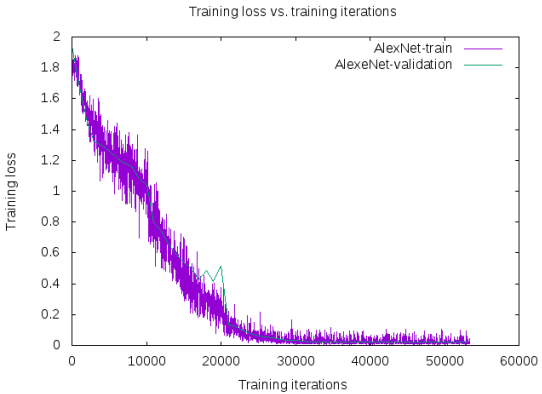
AlexNet 2



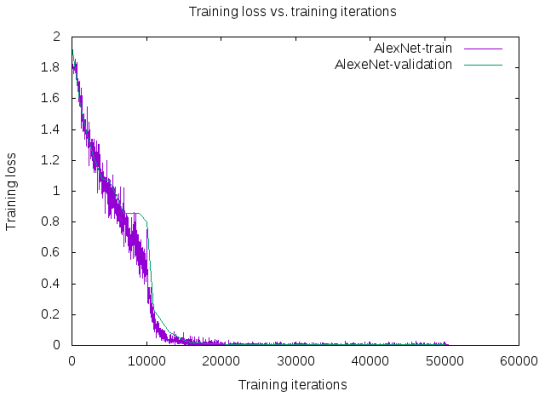
AlexNet 3

Training

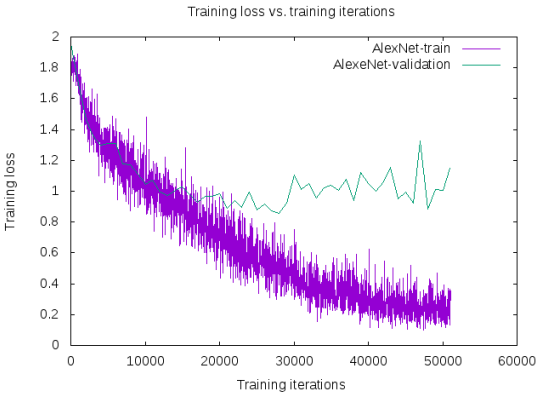
Training loss vs. training iterations



AlexNet 1



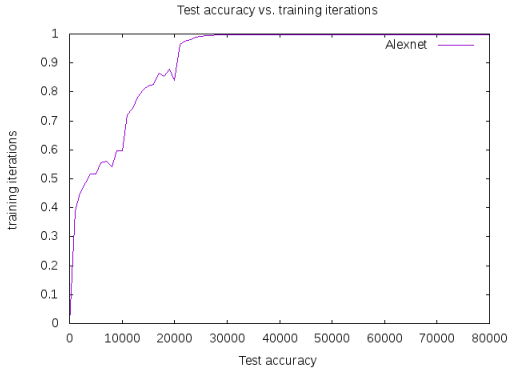
AlexNet 2



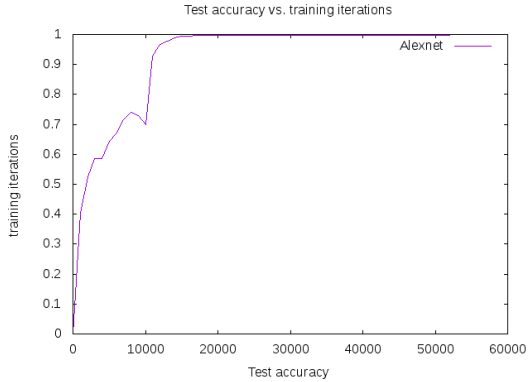
AlexNet 3

Training

Test accuracy vs. training iterations



AlexNet 1



AlexNet 2

AlexNet 3

Prediction Result

The best performance we got:

Accuracy: 54.8620785734%

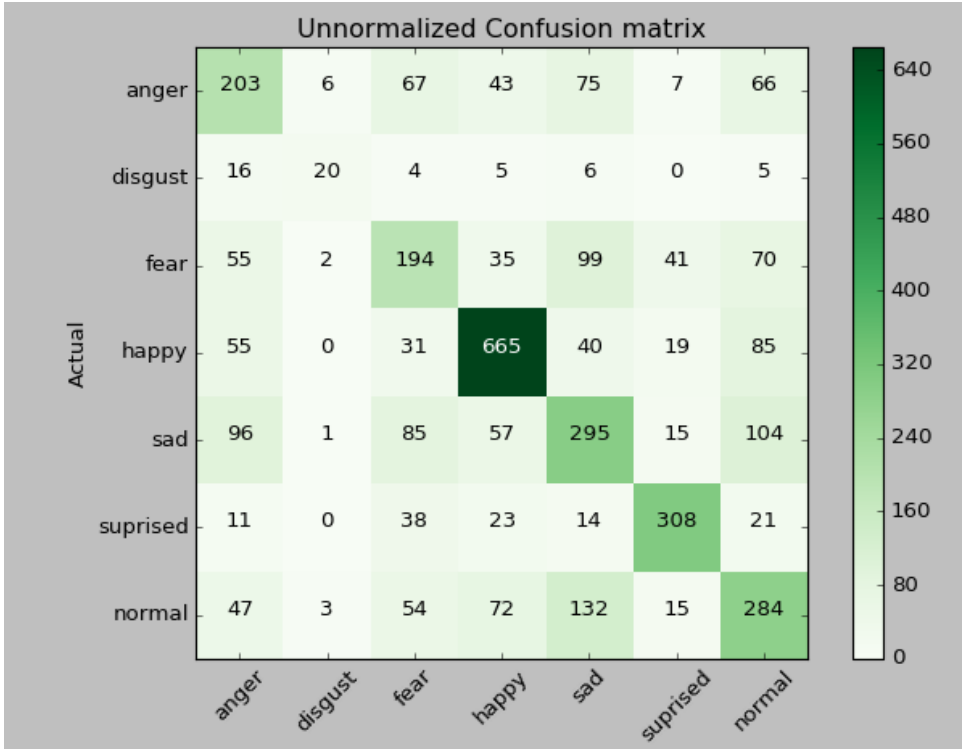
Misclassification Rate: 45.1379214266%

- Confusion matrix
- ROC Curve & AUC Value

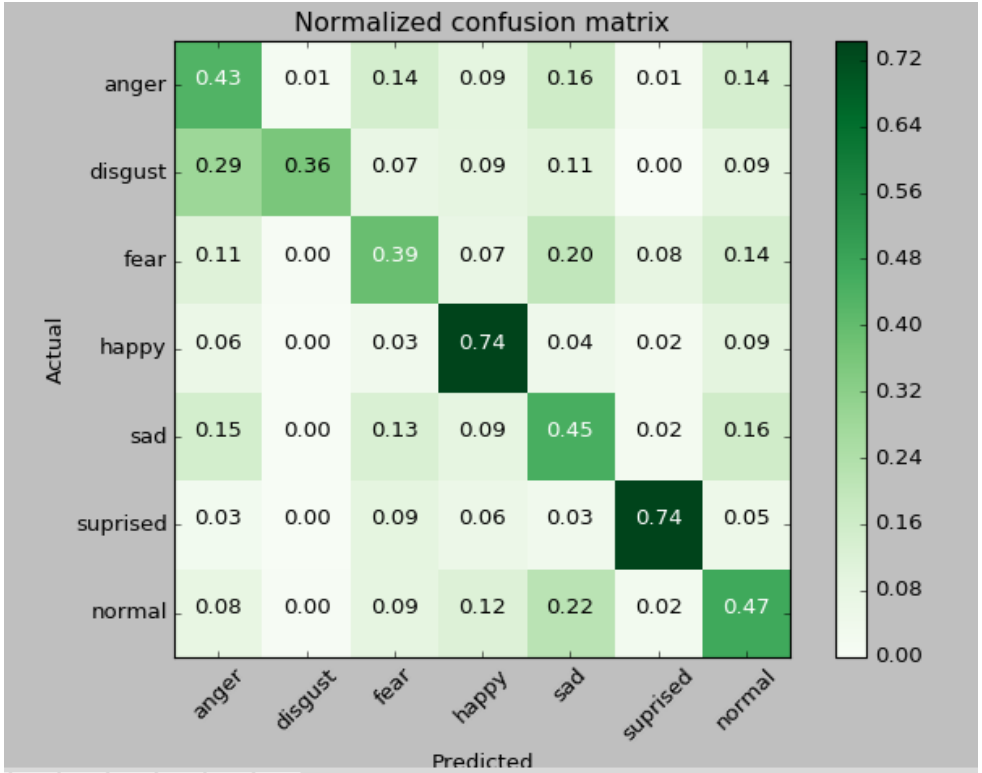
Confusion matrix

```
[[203   6  67  43  75   7  66]
 [ 16  20   4   5   6   0   5]
 [ 55   2 194  35  99  41  70]
 [ 55   0  31 665  40  19  85]
 [ 96   1  85  57 295  15 104]
 [ 11   0  38  23  14 308  21]
 [ 47   3  54  72 132  15 284]]
```

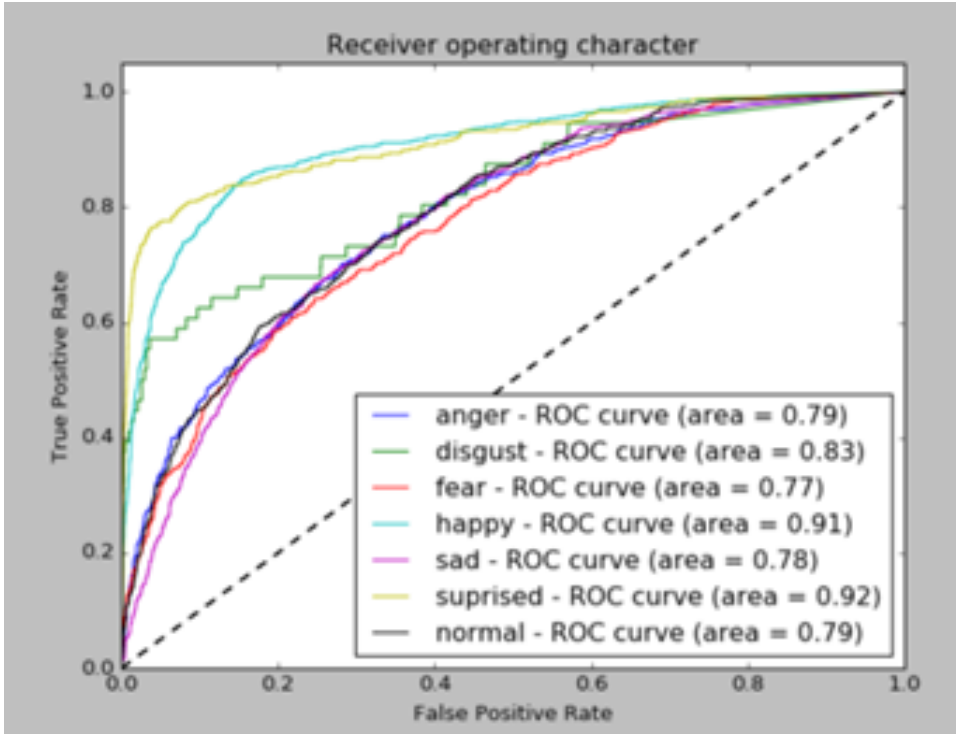
Predicted across the top: Each column of the matrix corresponds to a predicted class.
Expected down the side: Each row of the matrix corresponds to an actual class.



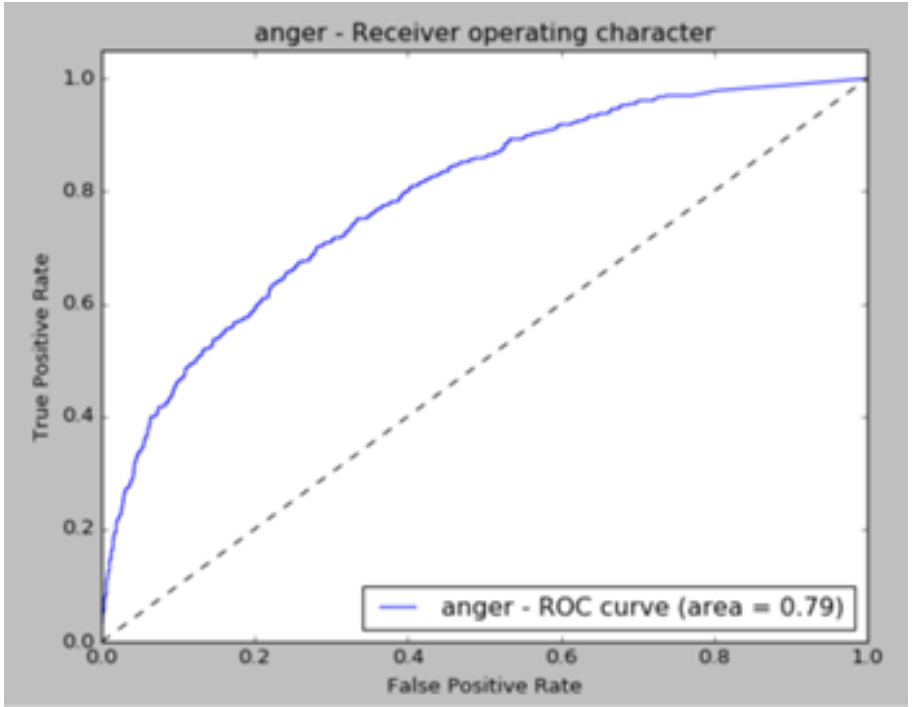
Normalized confusion matrix

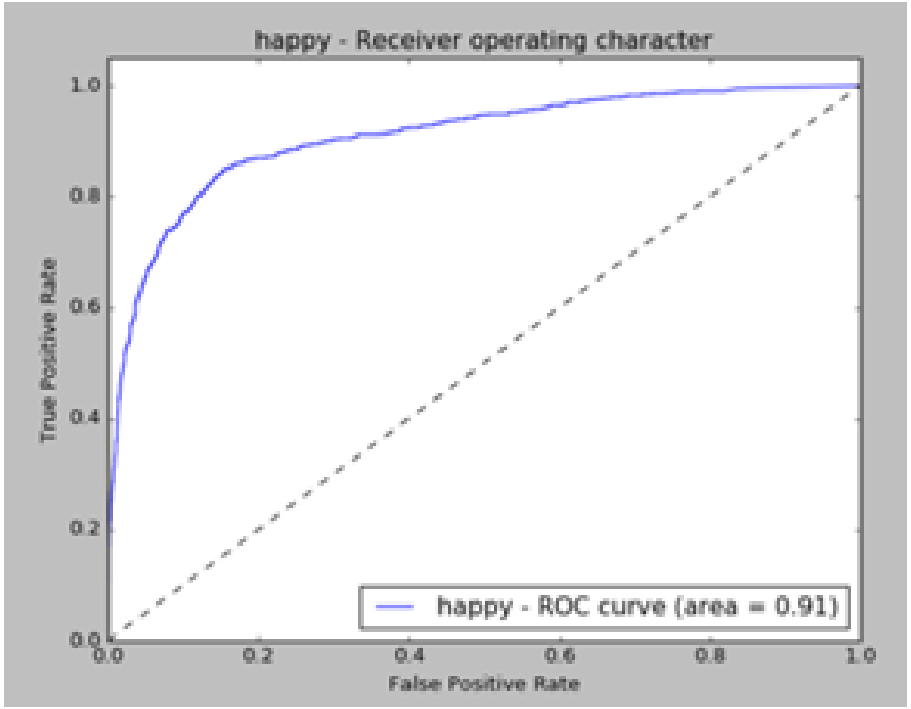


ROC Curve & AUC Value

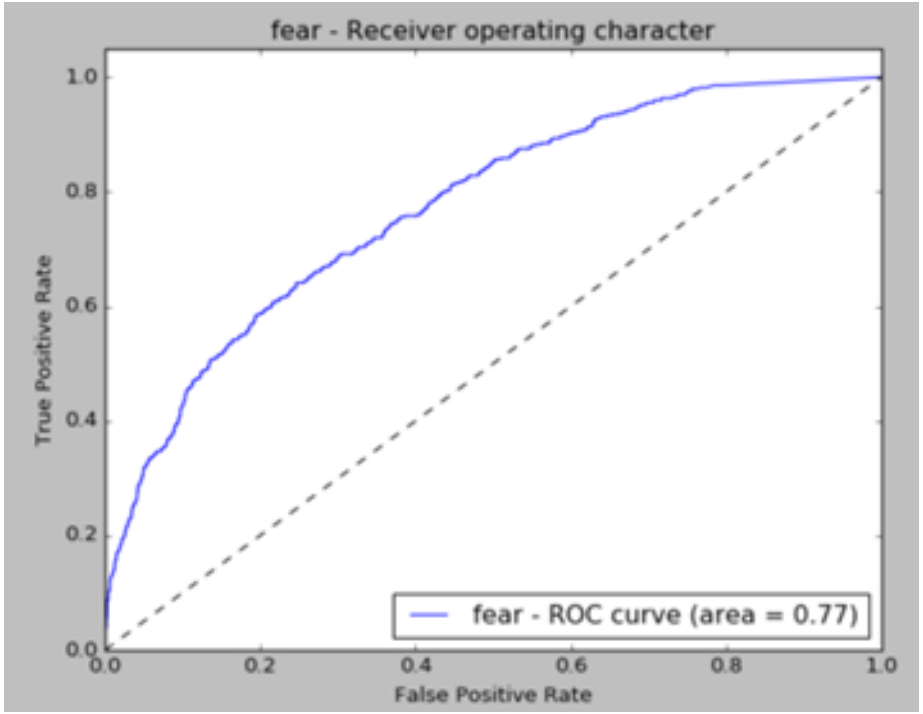


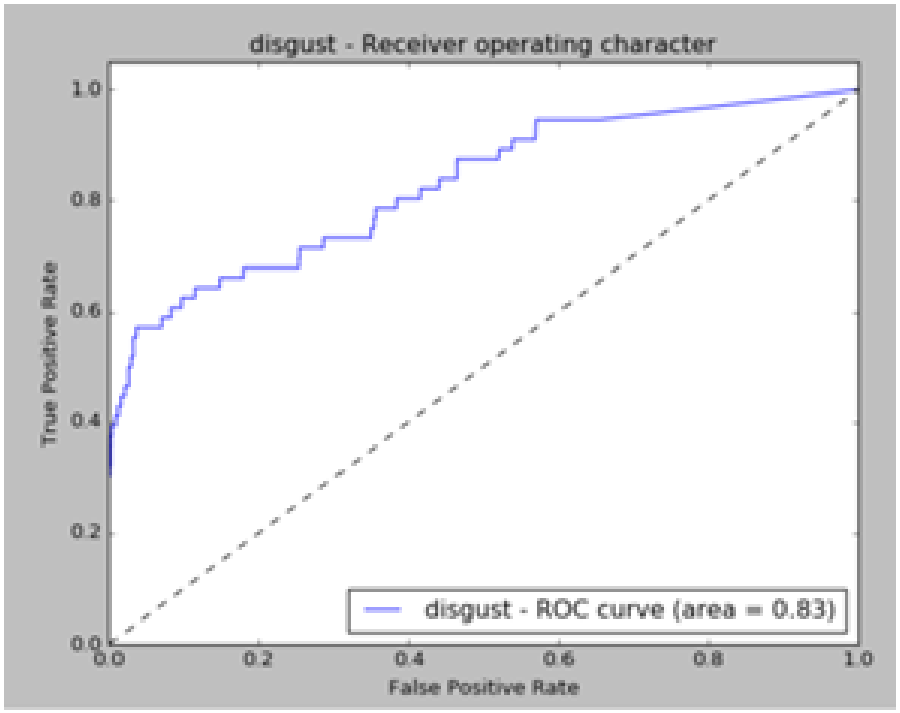
Anger & Happy


anger - Receiver operating character
True Positive Rate
False Positive Rate
anger - ROC curve (area = 0.79)
0.0
0.2
0.4
0.6
0.8
1.0

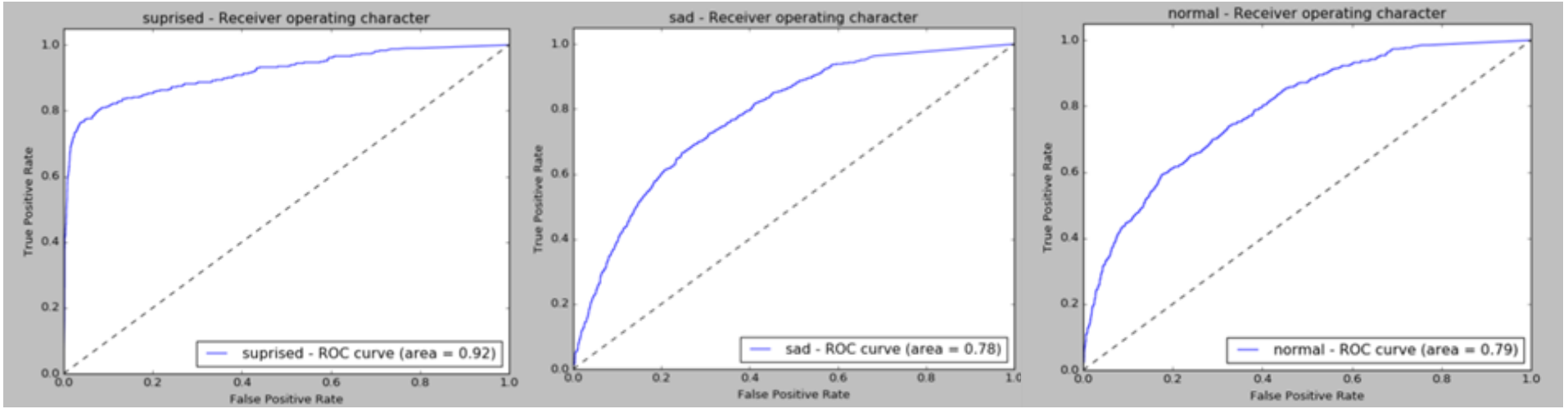

happy - Receiver operating character
True Positive Rate
False Positive Rate
happy - ROC curve (area = 0.91)
0.0
0.2
0.4
0.6
0.8
1.0

Fear, Disgust





Surprise, Sad, Normal



Summary

Overall accuracy: **55%**
Over mismatch classification rate: **45%**

Anger accuracy: 43%
Disgust accuracy: 36%
Fear accuracy: 39%
Happy accuracy: 74%
Sad accuracy: 45%
Surprised accuracy: 74%
Normal accuracy: 47%

Future

Network:

- Googlenet
- Resnet

Framework:

- Pytorch
- Tensorflow

Really thank you for your patient teaching Amir!!

Thank you.