Facial Key-point Detection
Update Report

Sarthak Agarwal
Cognitive Computing and AI

Problem statement:
The objective is to train a model that can predict the key-point positions on face images. Facial key-points are centers and corners of the eyes, eyebrows, nose and mouth, among other facial features.

The work done so far can be found at the following Github link:
https://github.com/agsarthak/Cognitive-Computing-and-AI/tree/master/Midterm_Project

Summary of what I have done and have not done.

Things that I have successfully accomplished:

1. **Data pre-processing and transformation**
   - Handle missing values.
   - Flatten, scale and stack the images in an array.
   - Convert the test image to greyscale, resize it, reshape it and store it in a 2d array.

2. **Setup environment on AWS EC2 instance with GPU2xlarge.**
   - Connected Jupyter notebook to AWS instance.
   - Connected FileZilla to AWS instance for file transferring.

3. **Built a baseline neural network model.**
   - Built a model with one hidden layer and 100 nodes.
   - Ran it for 100 epochs on a CPU. Took 5mins to run.
   - Tried different momentum, learning rate and optimizers and compare the learning curve by plotting the graph on training and validation data.

4. **Built a Convolutional Neural Network.**
   - Built a neural network with 3 CNN layers and 3 dense layers.
   - Ran it on GPU at AWS for 500 epochs that took 40mins to run.
   - Observed significant decrease in loss when compared to baseline model.
   - Saved the model and weights for future prediction.

5. **Researched and implemented data augmentation technique.**
   - Flip the images and map the key-points to the flipped image in training dataset. This is considered new data by the algorithm.

- Ran the CNN on the updated set of data and observed decrease in loss.
- Ran it on GPU at AWS for 500 epochs. It took 50mins.

6. **Prediction**
   - Wrote a script that takes a real-life image – resizes it, converts it into 2d greyscale image and stores it in an array.
   - Predicted the facial key-points on the real image.

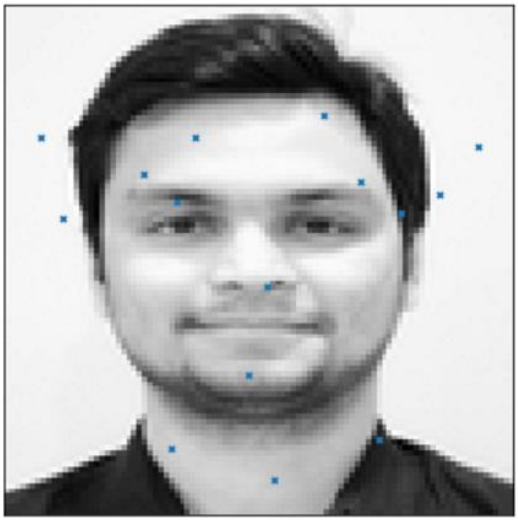
Outstanding items / Future Work:

1. Build a statistical model and compare the performance with Neural Networks.
2. Run the above CNN models for more epochs.
3. Build a deeper convolutional neural network like DenseNet and ResNet.

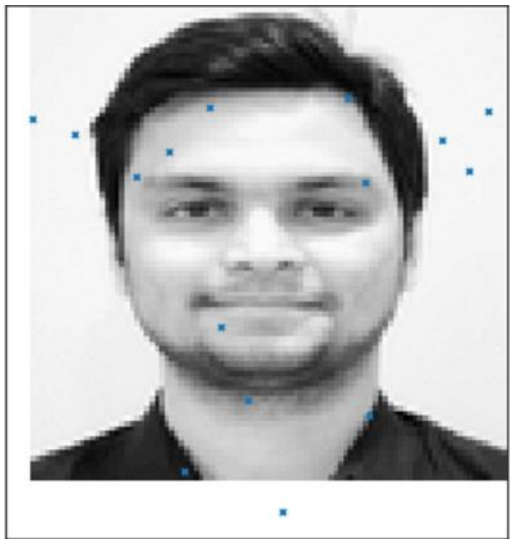
Results:

Model	Model Description	Epoch	Time	Train Loss	Val Loss
Model 1	NN with 1 hidden layer, 100 nodes	100	3mins CPU	0.0024	0.0037
Model 2	Model 1 + momentum=0, decay=0, nesterov=False	100	3mins CPU	0.0070	0.0080
Model 3	CNN	500	40mins GPU	0.0012	0.0020
Model 4	CNN – Flipped images	Yet to run			

Model 1 Prediction:



Model 2 Prediction:



Model 3 Prediction: