FACIAL KEYPOINT DETECTION

CS365A – Artificial Intelligence

Guide : Prof Amitabha Mukerjee

Abheet Aggarwal(12012)
Ajay Sharma(12055)

Abstract

Detecing facial keypoints is a very challenging problem. Facial features vary greatly from one individual to another, and even for a single individual, there is a large amount of variation due to 3D pose, size, position, viewing angle, and illumination conditions. Computer vision research has come a long way in addressing these difficulties, but there remain many opportunities for improvement.

Introduction

We used an approach for estimation of the positions of facial keypoints with three-level carefully designed convolutional networks. At each level, the outputs of multiple networks are fused for robust and accurate estimation. Since the networks are trained to predict all the keypoints simultaneously, the geometric constraints among keypoints are implicitly encoded.

Applications

Building block in several applications, such as:

- tracking faces in images and video
- analysing facial expressions
- detecting dysmorphic facial signs for medical diagnosis
- biometrics/face recognition

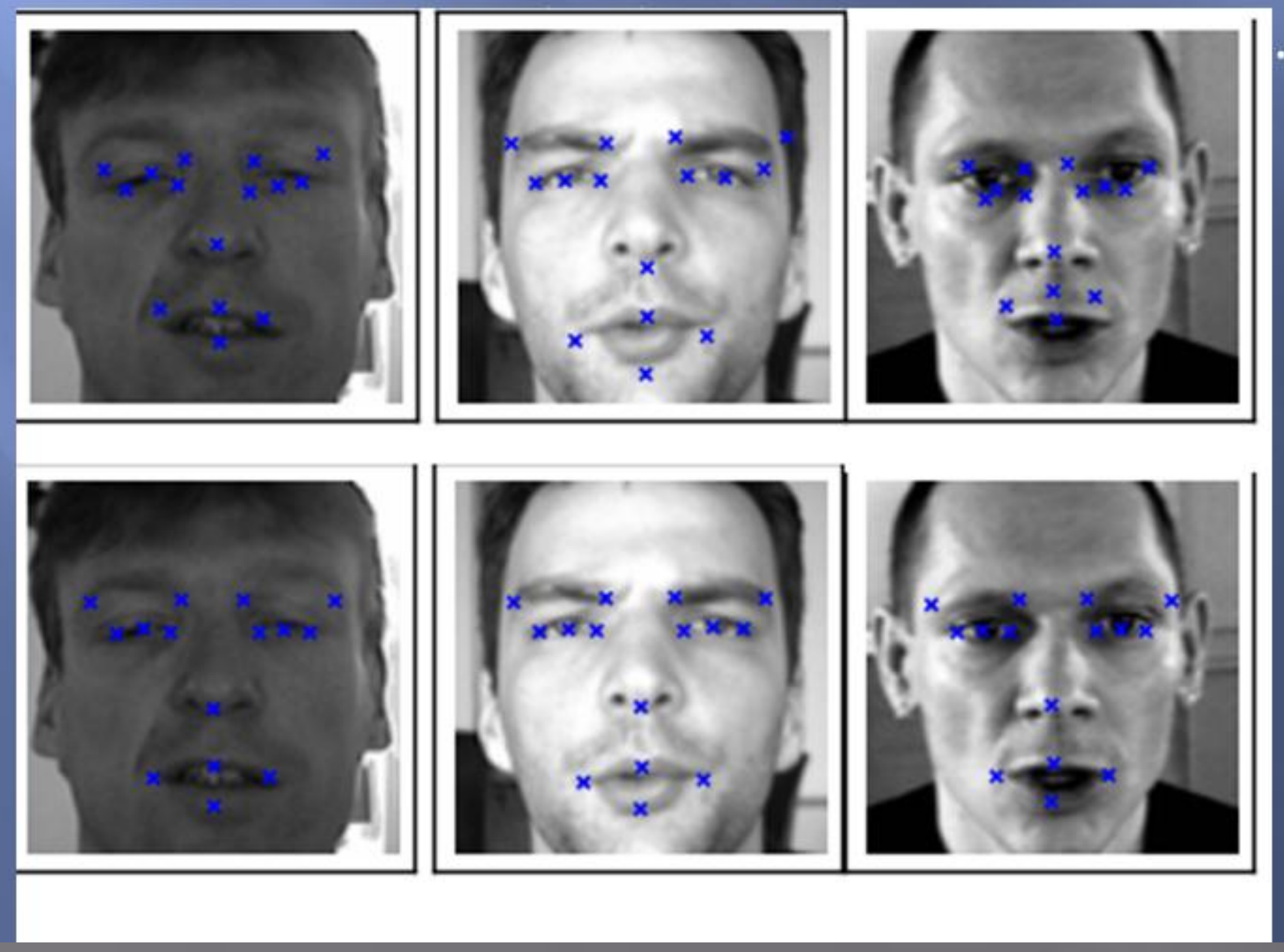
RESULTS AND COMPARISON :

We used two techniques : Simple Neural Network and Deep CNN

We found that the results of simple network were not much satisfying(RMSE=3.8251) while the use of deep CNN made them much better.

We used two hidden layers instead of one in order to implement the non-linearity which increased the accuracy of the output. The number of nodes in the hidden layer are 500 which we chose through cross-validation.

When we used 100 units, a validation loss of 0.004730 implying RMSE = 3.3014. Using 500 units gave less validation loss of 0.004194 implying



Simple NN: The error is large as labels are too far than actual.

Deep CNN : Improvements can be seen easily as keypoints are more accurately labelled.

Dataset

We used the Kaggle dataset which had 7049 training images of 96x96 in a csv file out of which only 2140 had all the 15 keypoints marked. We used 80 percent of these images for training and the remaining 20% for validation.

We deliberately used one of the problematic samples in the test set the results of which were still satisfactory as can be seen in the results section.

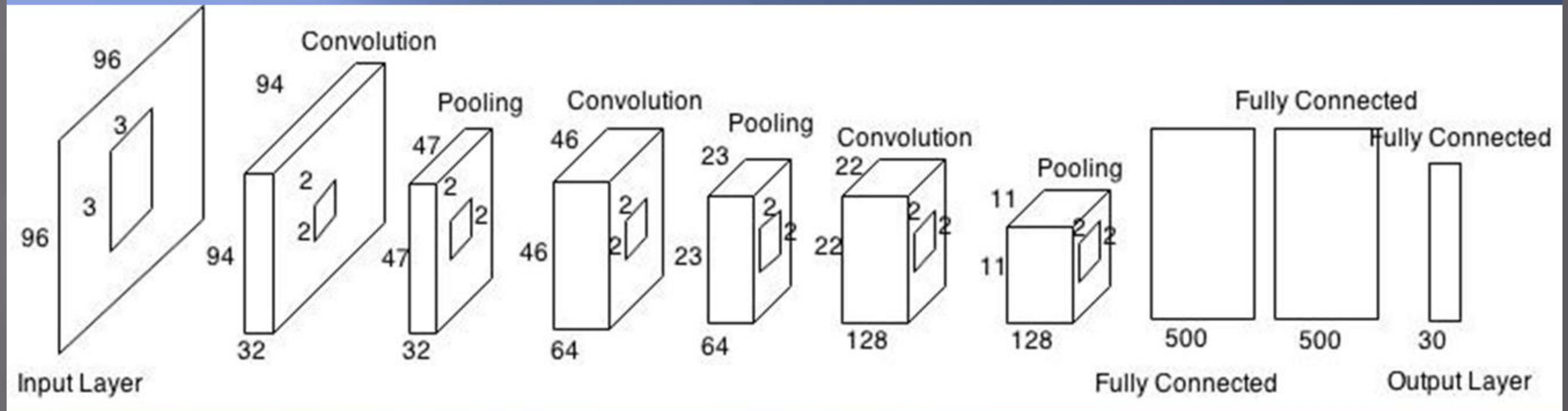
Technique used:

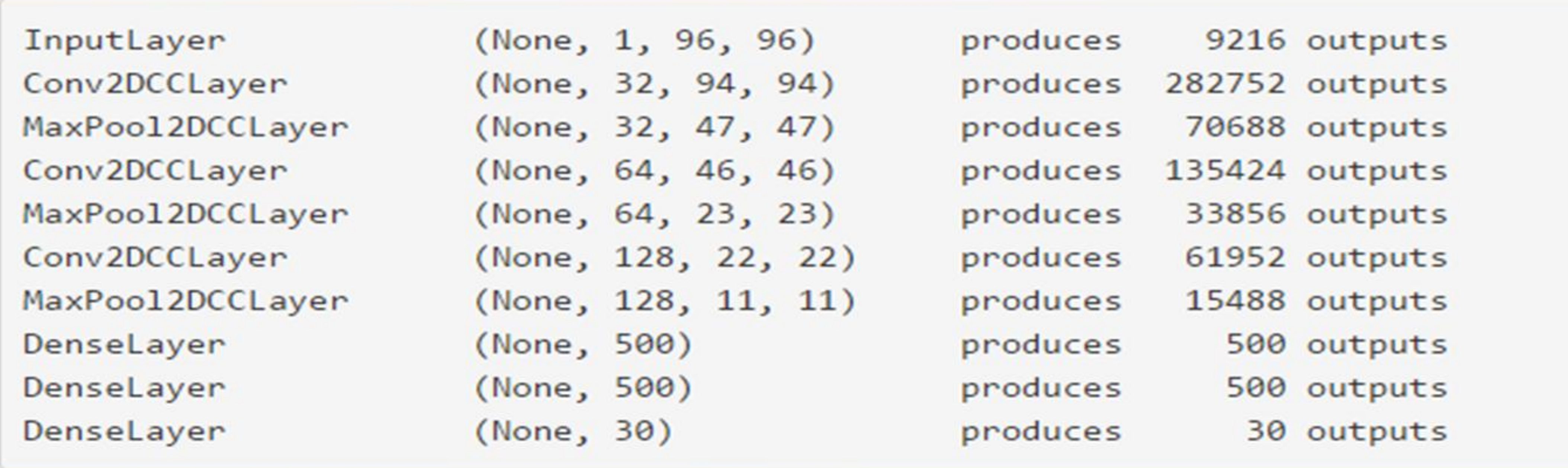
Our network consists of three convolutional layers each one followed by a pooling layer. To implement non-linearity we used the rectifier function over sigmoid function to overcome the vanishing gradient problem [Ref.2].

We used the lasagne and theano libraries in python[Ref.3] to make and train the network.

Initial run of 50 epochs didn't give appealing results. A run over 100 epochs gave much better results.

ARCHITECTURE OF DEEP CONVOLUTIONAL NEURAL NETWORK





```
InputLayer          (None, 1, 96, 96)     produces   9216 outputs
Conv2DCCLayer       (None, 32, 94, 94)    produces 282752 outputs
MaxPool2DCCLayer    (None, 32, 47, 47)    produces  70688 outputs
Conv2DCCLayer       (None, 64, 46, 46)    produces 135424 outputs
MaxPool2DCCLayer    (None, 64, 23, 23)    produces  33856 outputs
Conv2DCCLayer       (None, 128, 22, 22)   produces  61952 outputs
MaxPool2DCCLayer    (None, 128, 11, 11)   produces  15488 outputs
DenseLayer          (None, 500)           produces    500 outputs
DenseLayer          (None, 500)           produces    500 outputs
DenseLayer          (None, 30)            produces     30 outputs
```

Future Work:

- Flipping the images horizontally will allow us to produce an infinite number of examples, without blowing up the memory usage. But one has to be careful as the coordinates of the left and right keypoints also should be flipped.
- For neurons in the convolutional layers, absolute value rectification after the hyperbolic tangent activation function can effectively improve the performance[Ref.1].
- Running the network for more number of epochs can help produce better results.

References :

1. Sun, Y., Wang, X., & Tang, X. (2013). Deep convolutional network cascade for facial point detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
2. Andrew L Maas, Awni Y Hannun, and Andrew Y Ng, "Rectifier nonlinearities improve neural network acoustic models," in ICML Workshop on Deep Learning for Audio, Speech, and Language Processing (WDLASL 2013), 2013.
3. Daniel Nouri's blog – danielnouri.org
4. Kaggle Facial keypoints detection challenge. (May 2013 – Dec 2015)