EE 622 Assignment #02

Classification of Cat vs. Dog Images

Ankit Kumar Wagadre 130108026

Date: 13/11/2016

Aim:
We have given binary classification problem of images. Given images is composed of Cat and Dog. We have to build a model using Convolutional Neural Network to classify them.

Method and Library used:
For this assignment to build neural network I've used a pretty tensor library which a API for Tensorflow. It provides thin wrappers on tensors so that you can easily build multilayer neural networks. Pretty tensor provides a set of object that behave likes tensors, but also support a chainable object syntax to quickly define neural networks.

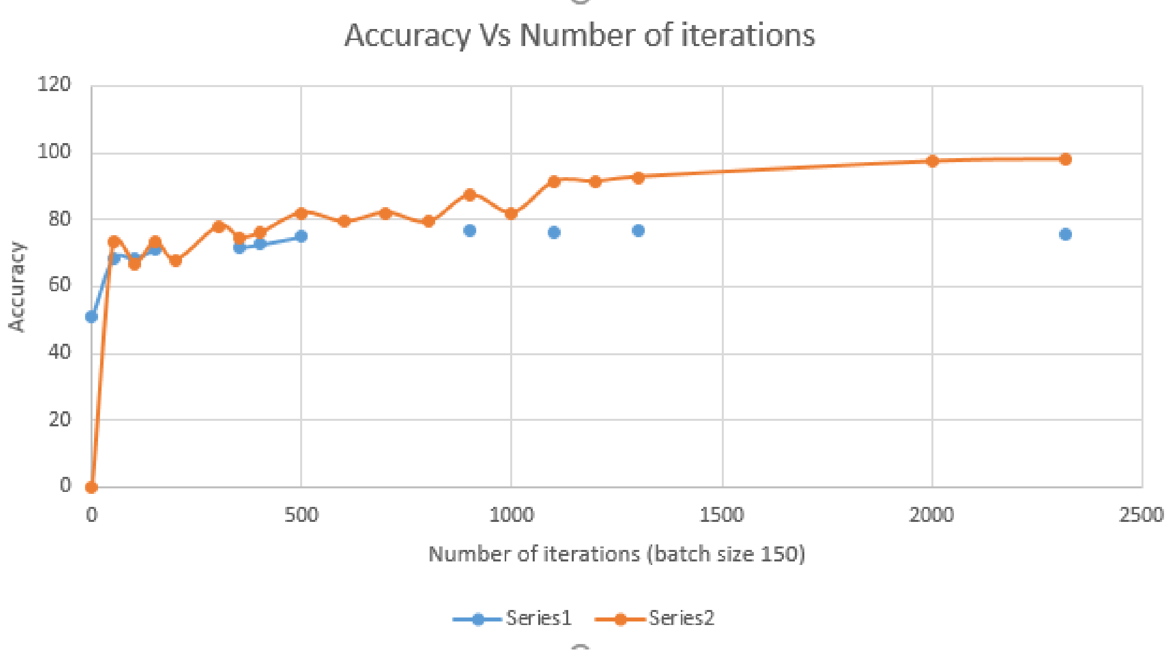
Input Data
Input data is a set of 25000 training images. The data is in vector form every image is a 12288 size array which is equivalent to a 64x64 rgb image. First Training data has been reshaped and divided into two sets training set (1:22000) and validation set (22000:25000). Training data labels are converted into labels one encoded array to use it in tensorflow.
For example, if class number is 2 and number of classes is 3 then label is [0 0 1].
For testing data there is 12500 images are available whose predicted output has been written in "ouput.txt" file.

Main network:
The main network is consist of 2 convolution layer of kernel size 5x5 and of depth 64, followed by a maxpooling of kernel size 2 and stride size 2 each. First input is wrap in a pretty tensor object x_pretty which will be helpful for batch normalization in first layer. After that this object is passed through the two convolution and maxpooling layers. Output of convolution layer is flattened and passed through two fully connected layer of size 256 and 128. In the end a softmax classifier is used. A TensorFlow variable is created "global_step" that keeps track of the number of optimization iterations performed so far. Tensorflow saver object is used to save checkpoints after some optimization so that we don't have to repeat optimization every time we run our system. A randomized input batch of batch size 150 is used as training data per optimization. Training accuracy is printed after every 100 optimizations and saved after every 1000 optimizations.

Results:

I have done over 2000 optimizations with a batch size of 150 images per optimization. With the help of Nvidia GPU 2 GB and 4 GB RAM I was able to complete 2000 iteration in around one and half hour. Validation set accuracy gradually increased with every 100 optimization and become almost 80% after 2000 iteration.

The following a graph of training data and validation data accuracy vs number of optimization.



serires1 is validation set accuracy and series2 is training data accuracy.