**AI HACKATHON REPORT**

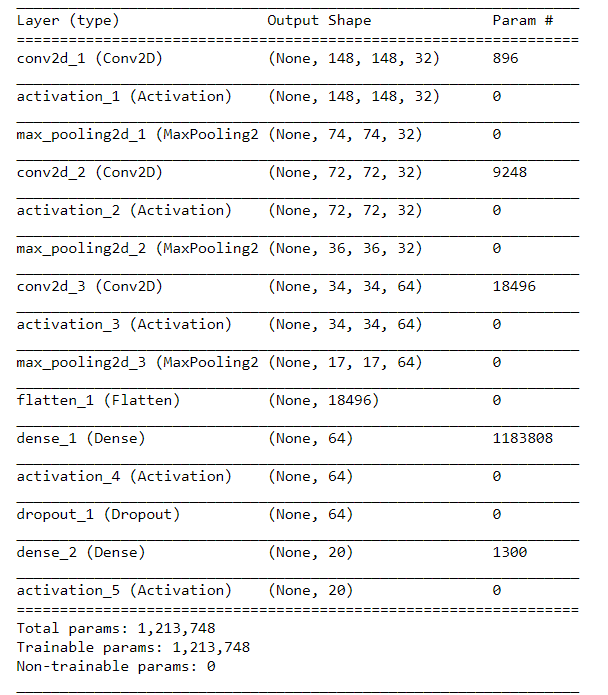
Praveen C Naik -01FB15ECS213

Parashara R -01FB15ECS202

Rahul Pillai -01FB15ECS224

**PROBLEM 1(Detect multiple objects in a image and classify them)**

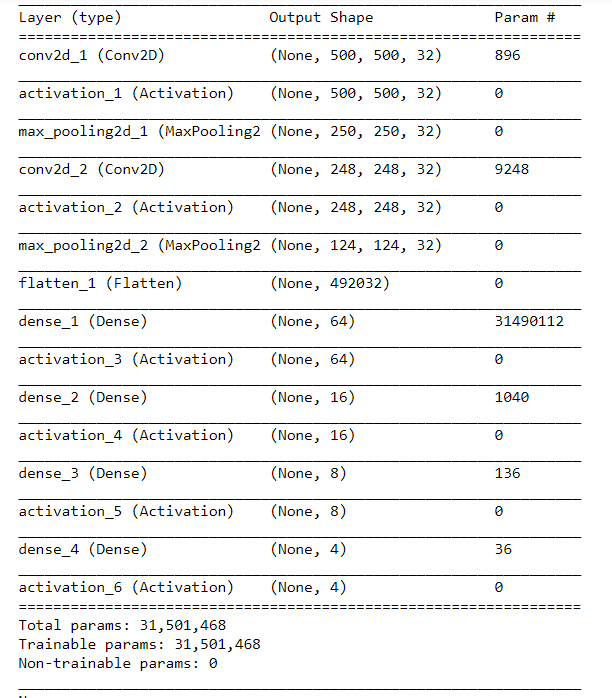
* **Training data** :
  + Our training data consisted of al the 11k images in the dataset
  + Our xtrain was a numpy array of the image in the shape (11300,150,150,3) after resizing all images to the size of 150\*150
  + Our ytrain was a one hot vector of size 20 with the ith bit =1 implying that we have the classified it as the ith class among the given 20 clases
* **Model overview and architecture**:
  + The model has the following structure



* + The last output layer is a dense layer with 20 units with an activation of sigmoid
  + Each convolutional layer has an activation of Relu and max Pooling layer, each filter has a dimension of (3x3) kernel
* **Results and Issues**:
  + we are getting around 92% training accuracy and prediction happens very well as our test accuracy is around 94%
  + Our precision and recall values were around 0.8094 and 0.9232 respectively
  + we tried to train the same data with the vgg16 model but we were not able to train beyond 1 epoch as it takes a long time but we tested with that we were able to classify it again with a very good training accuracy of 97%
  + Although the given problem was only for single object classification in our model when given an image with multiple objects in it when predicting we get a one hot vector where we get the corresponding k’s bit value as 1 for each of the k objects in the given test image

**PROBLEM 2(Detect the bounding box for an object in a image)**

* **Training data and approach** :
  + Our training data consisted of only a subset of the 11k images in the dataset which is around 5k images (ie only those with a single object in the image)
  + Our xtrain was a numpy array of the image in the shape (5000,500,500,3) after padding all images to the size of 500\*500 so that the bounding box coordinates need not be scaled down later(we used the PIL library to do the preprocessing)
  + Our ytrain is a numpy array of the 4 points determining the bounding box of shape (5000,4)
  + We have separate regressors for each different class and we save each of those models separately as a .h5 file in our local machine
* **Model overview and architecture**:
  + The model has the following structure

**

All the activations have been made to Relu as we felt that relu activations would be faster to compute for such a convolutional network as ours

* **Results and Issues**:
  + we are getting around 62% training accuracy and around 51% testing accuracy .
  + we tried to train the same data with the vgg16 model along with our dense output of 4. but we were not able to train beyond 1 epoch as it takes a long time and accuracy for the bounding box is also very reasonably good around 71% accuracy
  + We were facing a lot of issues when trying to fit the model as we were facing ‘Resource Exhausted issues due to the size of the model and the parameters.

**TESTING PHASE**

* **APPROACH**:
  + We have a seperate file to test both the classification and the bounding box so our approach was given an image we first apply the classification model and from that based on the class we got we used that particular class’s regression model and predicted the bounding box for that image and below are few of the images we got after trying it for a few pictures.

