Flipkart_Grid - 2019 - AInsteins

Mehul Kasliwal, Saurav Pradhan, Aditya Ketkar

The basic idea of object detection using bounding box is to detect a real life independent object by specifying coordinates of bounding box for the object.

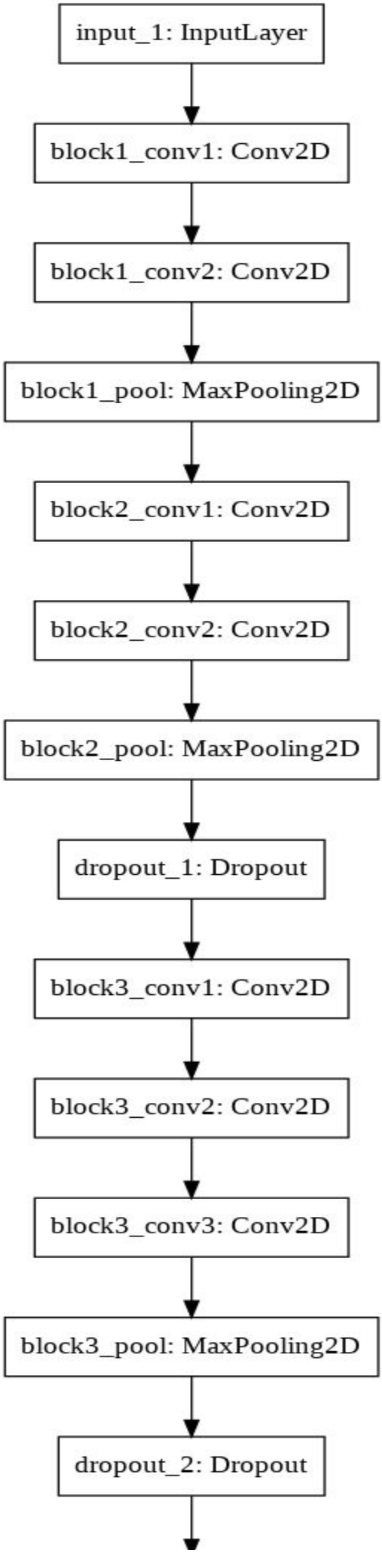
Pre-Processing :

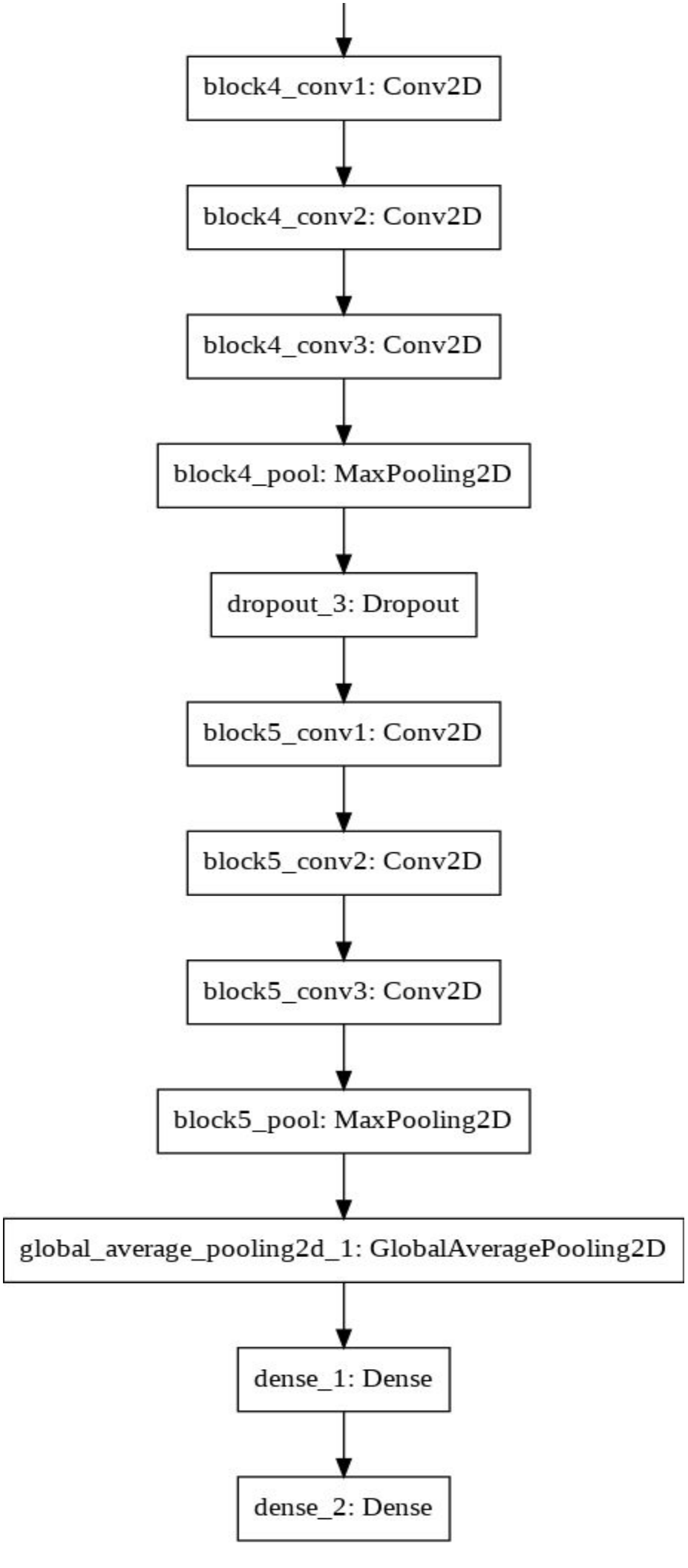
The images were pre-processed by the following steps :

1. To scale the RGB values from [0, 255] to [-1, 1], the values were subtracted from mean and divided by standard deviation.
2. Next we have grayscale the images , as they are more suitable and also reduces the amount of features by reducing the total no of data channels used from three to one, which would roughly reduce the size of the data by 3.
3. Finally the last step of preprocessing include reshaping the size of the image to a suitable value such that training it on a large network is feasible within reasonable bounds of data loss. The dimensions that we chose was 192*192 pixels ,reshaped from the original picture of size (640*480) . We also tried modelled the data using smaller dimensions of 64*64, 128*128 , however due to higher data losses , the model suffered from lower accuracies. Also ,now that we have reshaped the images we made suitable changes to our xmin,x,ymin,ymax in the training set by multiplying them with the requisite scaling factor.

Training :

A sequential deep learning model was designed keeping in mind the complexity and volume of data to be learned. The block diagram on the next page summarizes the architecture of the whole neural network.


input_1: InputLayer
block1_conv1: Conv2D
block1_conv2: Conv2D
block1_pool: MaxPooling2D
block2_conv1: Conv2D
block2_conv2: Conv2D
block2_pool: MaxPooling2D
dropout_1: Dropout
block3_conv1: Conv2D
block3_conv2: Conv2D
block3_conv3: Conv2D
block3_pool: MaxPooling2D
dropout_2: Dropout


block4_conv1: Conv2D
block4_conv2: Conv2D
block4_conv3: Conv2D
block4_pool: MaxPooling2D
dropout_3: Dropout
block5_conv1: Conv2D
block5_conv2: Conv2D
block5_conv3: Conv2D
block5_pool: MaxPooling2D
global_average_pooling2d_1: GlobalAveragePooling2D
dense_1: Dense
dense_2: Dense

Features Used :

- EarlyStopping was used with a minimum threshold of 0.0001 for delta to prevent model from overfitting.
- Callbacks feature of Keras was used to store the timesnap of model weights after epoch.
- Adam was chosen as optimizer for the model because of its excellent record with regression data.
- MSE loss function was used to calculate the deviation from true value of the predicted value.

Testing :

With the model trained , we finally put it to test with 12800 test samples. Similar to the Training Model, we first reshape the size of the image to 192*192 pixels. Next, we load the weights trained by the model previously on the training set. Next , we start predicting the output and finally export the predictions in the corresponding csv format.

Our model reported an accuracy of ~88%.