RBE/CS 549 Computer Vision

HW0 - Alohomora

Noopur Koshta
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, MA, USA
nkoshta@wpi.edu
Using 1 late day

***Abstract*—The assignment consists of two parts: A)"Shake my Boundary" where we use a probability based edge detection by calculating Texture, Brightness and Color Map and gradients along with Sobel and Canny Baselines B) "Deep Dive on Deep Learning " where we compare multiple deep learning architectures to classify objects from CIFAR10 BSDS500 dataset.**

***Index Terms*—Edge Detection, Sobel, Canny, CIFAR10, BSDS500, ResNet, DenseNet, ResNeXt**

I. PHASE 1 : SHAKE MY BOUNDARY

The objective of the first phase is to implement the PbLite edge detection algorithm. 'Pb' stands for probability of boundary and accordingly gives the probability of each pixel of an input image, belonging to a true edge in the image. The algorithm is implemented in 4 steps: A) Filtering the input image to get textures B) Quantizing the per pixel texture, brightness and color attributes C) Finding the gradients of texture, brightness and color for each pixel D) Combining the result with the baseline Canny or Sobel edge detection results

In this assignment, we use a probability based edge detection which consists of three different parameters:texture, brightness as well as color variations to detect boundaries along with three different filters: Oriented Derivative of Gaussian, Leung-Malik (LM), Gabor Filter-banks.

A. Oriented Derivative of Gaussian Filter Bank

As the name suggests, this filter bank has the first order derivative of the 2D gaussian function. The first order derivative is computed by convolving the gaussian kernel with the Sobel filter We obtain the Oriented DOG Filter, Convolution of a Sobel filter over a Gaussian kernel, rotating the kernel with 2 different scales and 16 orientations.

Equation of a Gaussian operator :

$$g(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)}$$

B. Leung-Malik (LM) Filter Bank

This filter bank consists of 4 different types of filters - the gaussian filter, first order derivative of gaussian, second order derivative of gaussian and the laplacian of gaussian. The derivative of gaussian filters have different standard deviation along the X and Y axes giving the filter an elongated shape. Further, these elongated filters are rotated to have 8 different orientations. A fixed set of 4 standard deviation values was used to create the filters and depending on those fixed set of values, 2 LM filter banks were implemented - LM Small and LM Large. Derivative of gaussian filters were created only for the first 3 scale values of the set. Gaussian and laplacian of gaussian were created for all the 4 scale values.

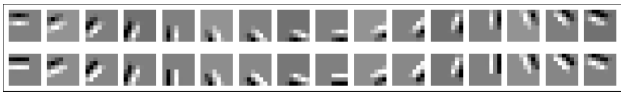

Fig. 1. Oriented DOG Filter-bank

Leung-malik filter-banks are formed by multi-scale, multi-orientation filter-bank consisting 48 different filters. There are three different types of Leung-malik filters. In first type of filters, first and second derivative filters occur at the first 3 scales with an elongation factor of 3, i.e. $sigma_x = sigma$ $sigma_y = 3 * sigma_x$. In second type of filter, Leung-malik small filters occurs at basic scales, sigma =$1, \sqrt{2}, 2, 2\sqrt{2}$. In third type of filter, Leung-malik large filters occurs at basic scales, sigma = $\sqrt{2}, 2, 2\sqrt{2}, 4$.

Leung-Malik filters are obtained by combining 4 different combinations of filters: 1) First Derivative of Gaussian Filter 2) Second Derivative of Gaussian Filter 3) Laplacian of Gaussian Filter 4) Gaussian Filter

C. Gabor Filter Bank

The gabor filter has a gaussian kernel modulated with a sinusoidal plane kernel. The bank consists of gabor filters having different standard deviation for the gaussian and different frequency for the sinusoids, which are further rotated to get different orientations. Gabor filters with 3 different standard deviation values and 2 different sinusoid frequencies were implemented.

Gabor filters mostly occur in the human visual system. Gaussian kernel function modulated by a sinusoidal plane wave. It analyses whether there is any specific frequency change.

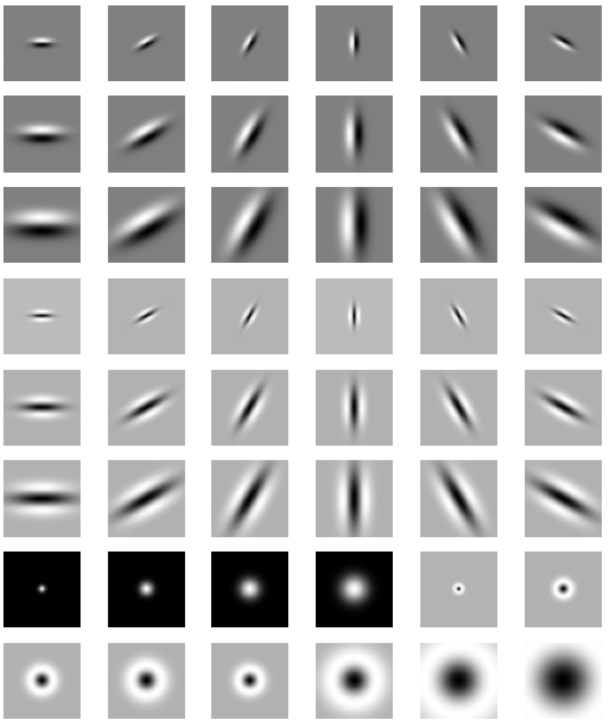

Fig. 2. Leung-Malik Small Filter-bank

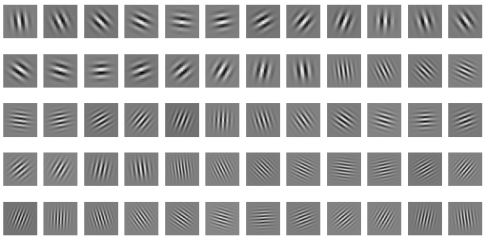

Fig. 3. Leung-Malik Large Filter-bank

D. Texton Map, Brightness Map, Color Map

1) Texton Map: We find Texton Map by capturing the texture changes in the image and cluster the texture variations with an N-dimensional vector for clustering all the responses at all pixels in the image for K textons using Kmeans.

The texton map or the texture map encodes the information of a certain pixel being a part of a texture in the image. This is done by convolving the image with the individual filters

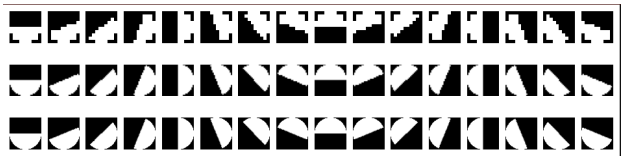

Fig. 4. Leung-Malik Filter-bank

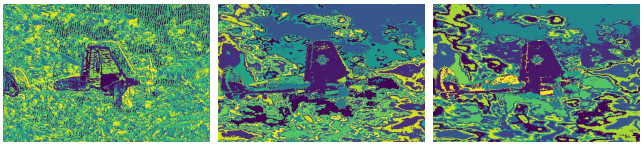

Fig. 5. Image 1 (a) Texton Map (b)Brightness Map (c) Color Map

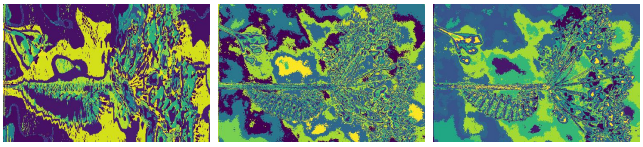

Fig. 6. Image 2 (a) Texton Map (b) Brightness Map (c) Color Map

in the filter banks generated in I-A. The brightness and color maps accordingly encode the intensity and color values for each pixel. These maps are then processed with dimensionality reduction operation to get a lower dimension space for the maps. This is done using the kmeans clustering algorithm. For the texture map, convolution was performed for all the 3 image channels - R,G,B. The color maps were also generated for all the 3 image channels. In the resulting texton and color maps, the 3 chanels were merged to create an RGB image. Figures 5 and 6 show the corresponding maps where the Derivative of Gaussian filter bank was used.

2) Brightness Map: We find Brightness Map by capturing the brightness change in the image and cluster the brightness values for gray-scale equivalent of a color image using Kmeans clustering by choosing a set of cluster bins.

The next step is to find the gradients of the texture, brightness and color at each pixel. This step is executed efficiently using half disk masks of varying sizes and orientations. Half disk masks are a set of pairs of masks having a semicircular disk centred at the centre of the mask, refer Figure 7. By applying pairs of half disk masks at each pixel of the maps generated in I-B via convolution, and computing the difference between the results of left mask and right mask application, we get a measure of the gradient of the attribute at the pixel.

3) Color Map: We find Color Map by capturing color changes or chrominance content in the image and cluster the color values (3 values per pixel (RGB color channels)) using Kmeans clustering by choosing a set of cluster bins.

E. Half Disc Masks

Half Disc Masks refer to pairs of binary images of Half-Discs using equation of circles constraining either x and y or both within a particular range and variation of angles.

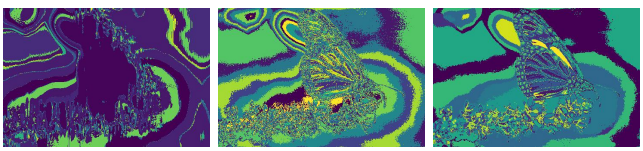

Fig. 7. Image 3 (a) Texton Map (b) Brightness Map (c) Color Map

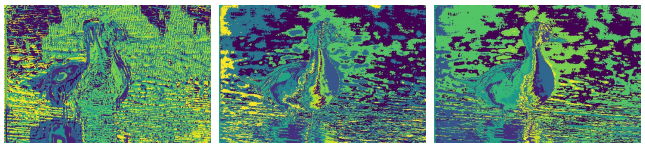

Fig. 8. Image 4 (a) Texton Map (b) Brightness Map (c) Color Map

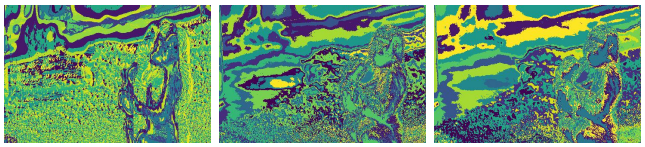

Fig. 9. Image 5 (a) Texton Map (b) Brightness Map (c) Color Map

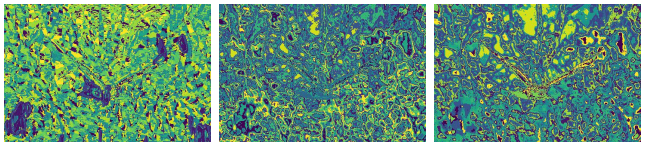

Fig. 10. Image 6 (a) Texton Map (b) Brightness Map (c) Color Map

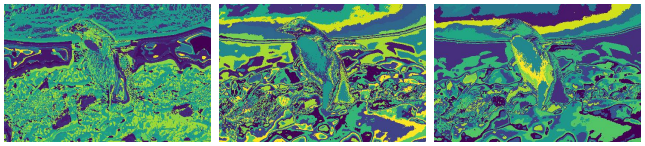

Fig. 11. Image 7 (a) Texton Map (b) Brightness Map (c) Color Map

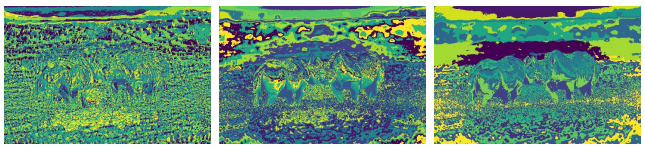

Fig. 12. Image 8 (a) Texton Map (b) Brightness Map (c) Color Map

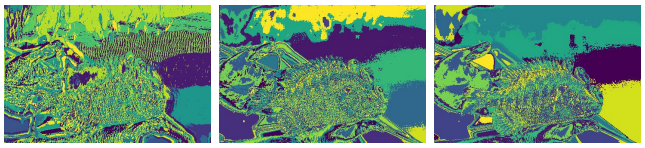

Fig. 13. Image 9 (a) Texton Map (b) Brightness Map (c) Color Map

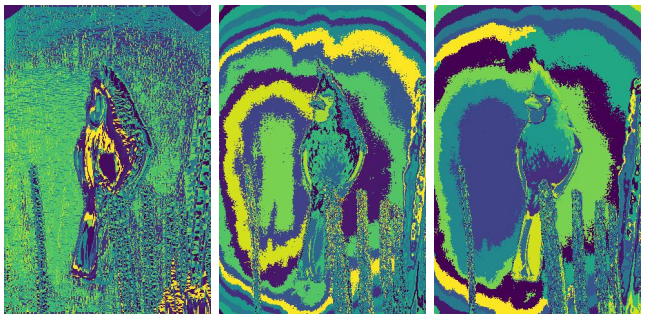

Fig. 14. Image 10 (a) Texton Map (b) Brightness Map (c) Color Map

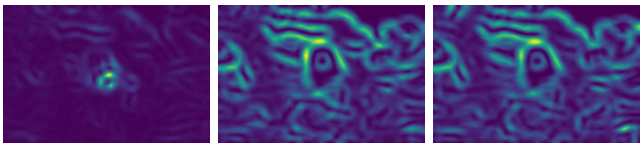

Fig. 15. Image 1 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

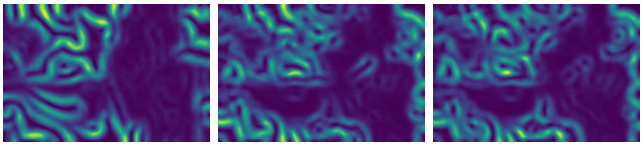

Fig. 16. Image 2 (a) Texton Gradient (b)Brightness Gradient (c) Color Gradient

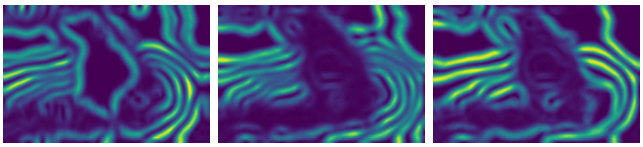

Fig. 17. Image 3 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

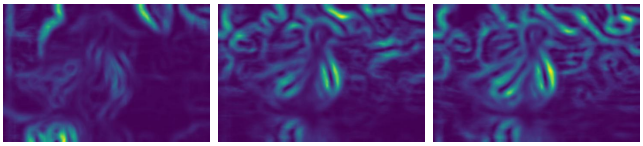

Fig. 18. Image 4 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

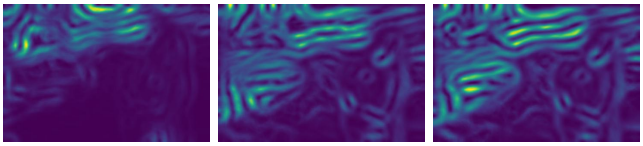

Fig. 19. Image 5 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

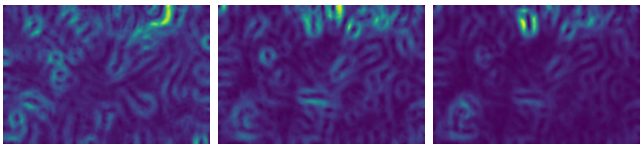

Fig. 20. Image 6 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

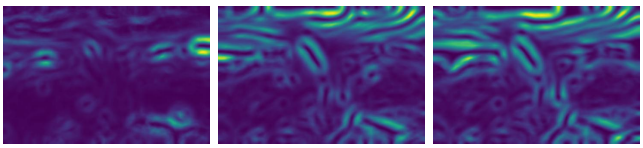

Fig. 21. Image 7 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

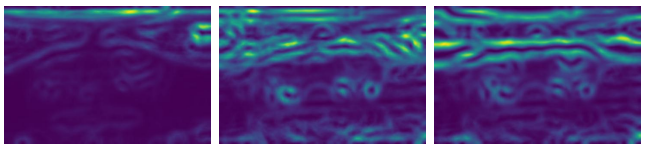

Fig. 22. Image 8 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

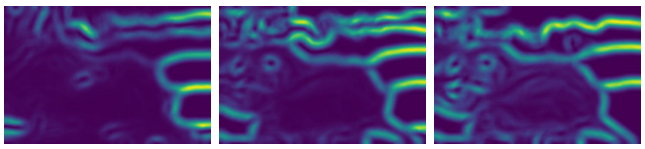

Fig. 23. Image 9 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

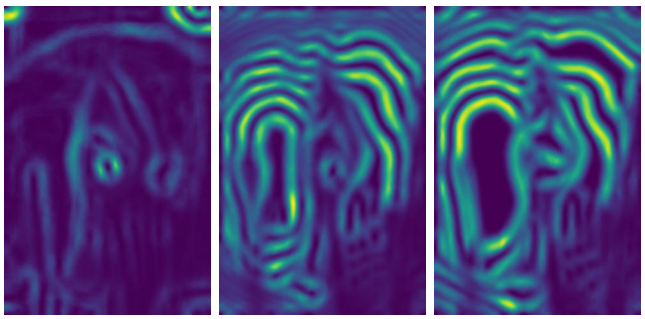

Fig. 24. Image 10 (a) Texton Gradient (b) Brightness Gradient (c) Color Gradient

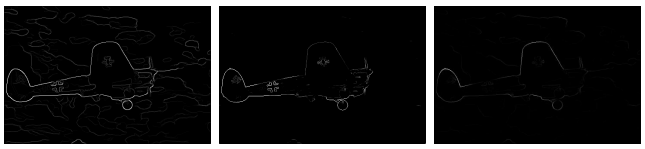

Fig. 25. Image 1 (a) Canny (b) Sobel (c) Pblite

Fig. 26. Image 2 (a) Canny (b) Sobel (c) Pblite

Fig. 27. Image 3 (a) Canny (b) Sobel (c) Pblite

Fig. 28. Image 4 (a) Canny (b) Sobel (c) Pblite

Fig. 29. Image 5 (a) Canny (b) Sobel (c) Pblite

Fig. 30. Image 6 (a) Canny (b) Sobel (c) Pblite

Fig. 31. Image 7 (a) Canny (b) Sobel (c) Pblite

Fig. 32. Image 8 (a) Canny (b) Sobel (c) Pblite

Fig. 33. Image 9 (a) Canny (b) Sobel (c) Pblite

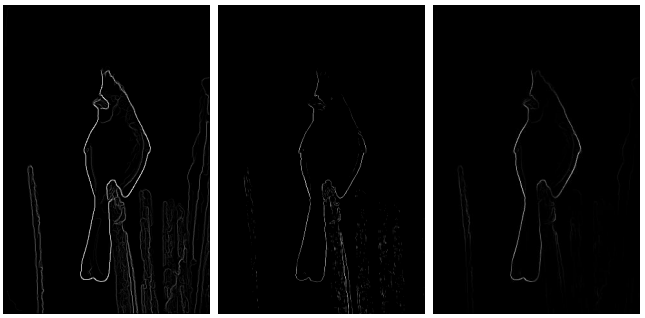

Fig. 34. Image 10 (a) Canny (b) Sobel (c) Pblite

F. Chi Square Distance

Chi-square distance is a statistical method to measure similarity between 2 feature matrices (h, g) and used in many applications like similar image retrieval, image texture, feature extractions. It has the property of distributional equivalence, meaning that it ensures that the distances between rows and columns are invariant. We use chi-sqaure distance to find the various gradient values by comparing each map with particular bins against half disk filter bank.

$$\chi^2(g,h) = \frac{1}{2}\sum_{i=1}^{K}\frac{(g_i - h_i)^2}{g_i + h_i}$$

G. K-means Clustering

K-means algorithm clusters data by trying to separate samples in group of equal variance by minimizing inertia or within cluster sum of squares.

Kmeans algorithm divides a set of N samples X into K disjoint clusters C, each described by mean u_i of samples in the cluster.

We first start with initialising the number of clusters and randomly initialise the centroid within the clusters and compute new centroids of each cluster by assigning each point to its closest centroid until the centroid positions remain constant and unaffected by further iterations.

H. Gradient Maps

The Maps generated above are used to calculate gradient maps for texture, brightness and color. These maps encode the texture, brightness and color distributions changing at each pixel. These are generated by comparing the values at each pixel by convolving the image with a left/right half-disc pair centered at the pixel. The basic concept behind this is that if the values are similar the gradient should be small and if the values are dissimilar, the gradient will be large. The half-disks are generated by multiplying an array of size equal to the radius/scale of the circular disk with all values which lie inside this radius equal to 1 and rest 0, with an array of equal size but where one half of the array is 0s and the other half consists of 1s. This multiplication results in a half-disk which can be rotated to produce the desired half-disk mask. Here if you rotate the disk after you've multiplied the two arrays will result in pixel voids. This can be avoided by rotating the rectangular block matrix of 0s and 1s and then by applying a "logical OR" operator on them.

II. PHASE 2 : DEEP DIVE INTO DEEP LEARNING

A. Convolutional Neural Network

This is an initial deep learning model. It is a simple neural network with 4 convolution layers followed by 2 fully connected layers and a softmax output layer. The architecture is shown in figure. This architecture did not use any data augmentation while training and also no standardization technique was used. The network reached about 80% training accuracy in about 30 epochs. The importantthing to note here is that even though the training accuracy is pretty high, the test accuracy

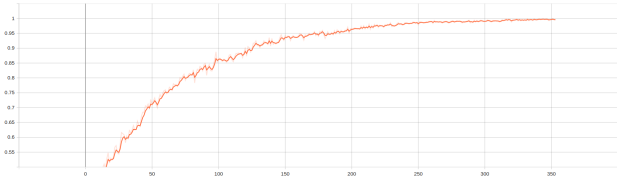

Fig. 35. CNN with tuned parameters Accuracy Graph

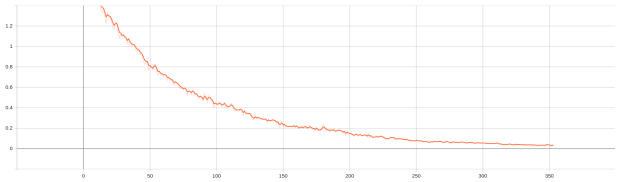

Fig. 36. CNN with tuned parameters Validation Loss

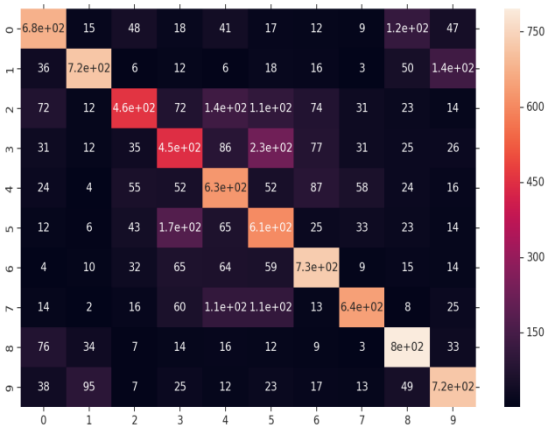

Fig. 37. Confusion Matrix

reaches a maximum of 54% for the same number of epochs as the train set.

B. Convolutional Neurqal Network with Improved Accuracy

The previous network gives acceptable results but its accuracy can be improved by slightly tweaking the architecture. We first normalize the dataset within values [-1,1]. We can also apply data augmentation techniques wherein we do random noise addition as well random left and right image rotation, to improve accuarcy. The final touch is adding batch normalization layers after each convolution layer. By doing this we force the input of every layer to have approximately the same distribution in every training step. This prevents the system from the internal covariate shift problem and decrease training time. We get considerable improvement in the test accuracy as the accuracy improves over the previous model with a maximum of 60%.

C. Residual neural Network (ResNet)

Keeping the standardization and data augmentation as it is we implement the ResNet architecture. A Residual Network,

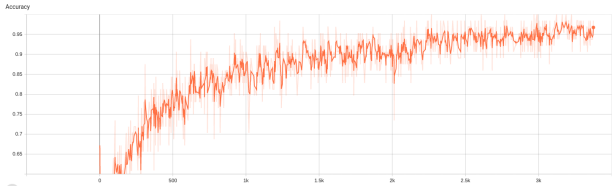

Fig. 38. ResNet Accuracy Graph

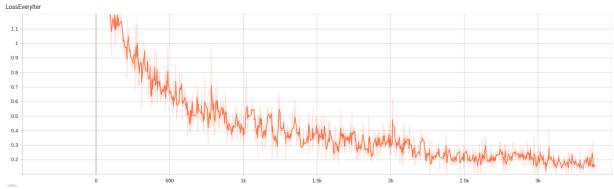

Fig. 39. ResNet Validation Loss

or ResNet is a neural network architecture which solves the problem of vanishing gradients in the simplest way possible, i.e., by applying skip connections in a general residual block. This allows the network to accommodate deep layers without having the vanishing gradient problem. This network was only trained for 25 epochs. Only 3 residual layers with 2 convolution layers each were used in this ResNet network. As you can see the training accuracy reaches about 90% for only 5 epochs.

D. ResNext

ResNext is a recent improvement on the ResNet architecture. In addition to utilizing the concept of residual learning framework from ResNet, the concept of ”Cardinality” is introduced in this network. Cardinality is the size of the set of split transformations an input goes through before to is passed on to the fully connected layers, as shown in fig(44). In ResNext architecture, the input is split into different paths(number of split paths is equal to the cardinality) and convolutions are performed in each of these split paths. The outputs of these split layers is then concatenated and added to the input itself, followed by application of non-linearity. It is empirically shown in the paper that even under the restricted condition of maintaining complexity, increasing cardinality is able to improve classification accuracy. Moreover, increasing cardinality is more effective than going deep

E. DenseNet

In this architecture, skip connections are used in different way, the output from one layer is added to all the next layers output and feeded as input. In the final modification, the enhanced network was added with 4 more convolution layers to give a total of 6 convolution layers. 3 of these layers have the same number of input and output channels while the other layers have the same number of input and output channels different from the first 3 layers. Each of the 3 layers was connected to all its subsequent networks of the same input size, thus yielding 7 connections in 6 networks, which in the normal scenario would have been 5 connections only, as

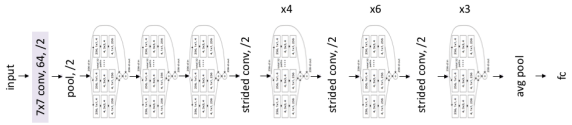

Fig. 40. ResNext Architecture

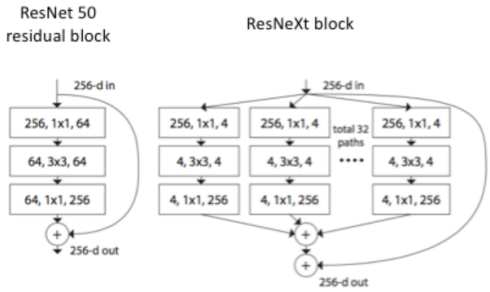

Fig. 41. ResNext Architecture

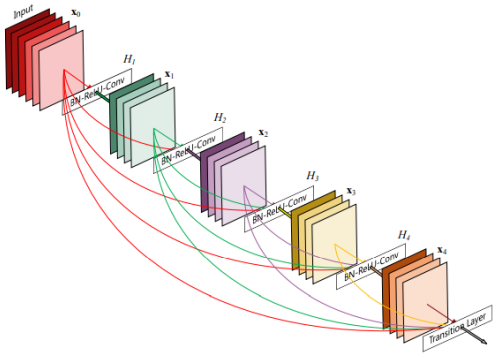

Fig. 42. DenseNet Architecture

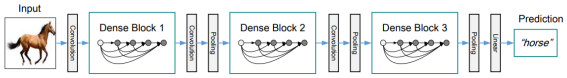

Fig. 43. DenseNet Architecture Layers

shown in Figure 18. This implementation is not a full-fledged replication of the DenseNet architecture, but just a toy model.

REFERENCES

[1] https://www.tensorflow.org/tutorials/images/deepcnn
[2] Samuel L. Smith, Pieter-Jan Kindermans, Chris Ying, Quoc V. Le, Don't Decay the Learning Rate, Increase the Batch Size
[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition
[4] https://dblp.org/rec/bib/journals/corr/HeZRS15
[5] Saining Xie, Ross B. Girshick, Piotr Dollar, Zhuowen Tu, Kaiming He ´ Aggregated Residual Transformations for Deep Neural Networks https://arxiv.org/abs/1611.05431
[6] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger
[7] Densely Connected Convolutional Networks https://arxiv.org/abs/1608.06993