CMSC733: Homework 0 - Alohomora

Chayan Kumar Patodi
UID : 116327428
Email: ckp1804@terpmail.umd.edu
Using 1 Late Day.

***Abstract*—The homework has two phases. Phase 1 is focused on implementation of boundary probability algorithm using texton,brightness and color information combined with canny and sobel baseline. Phase 2 is implementing multiple deep learning architecture on CIFAR10 dataset.**

I. Phase 1

The idea behind this homework is to determine the edges in a given image.The most used approach for such tasks are Canny and Sobel Edge detector. The algorithm we are working with in this homework (simplified version of Pb algorithm) takes in texture , brightness and color variations across multiple scales to detect the per pixel probability.

A. Filter Bank Generation

The very first step in the given pipeline is to get a filter bank , to capture different information from the input image. There are three different types of filters that are used in this homework : Oriented derivative of Gaussian (DoG) filter, Leung-Malik filter (large and small) and Gabor filter. The output from these filters are shown below:

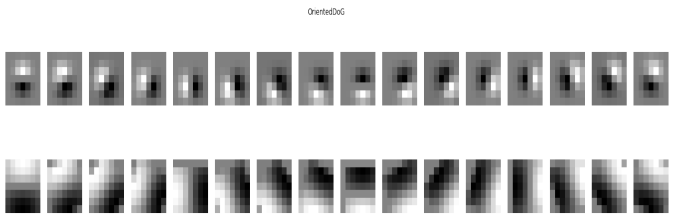

Fig. 1. Oriented DoG filter

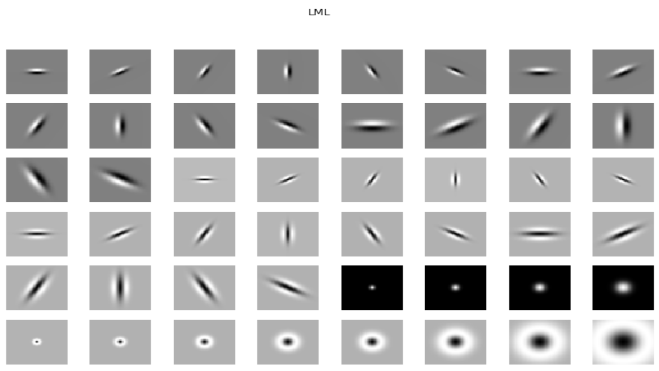

Fig. 2. Leung-Malik large filter

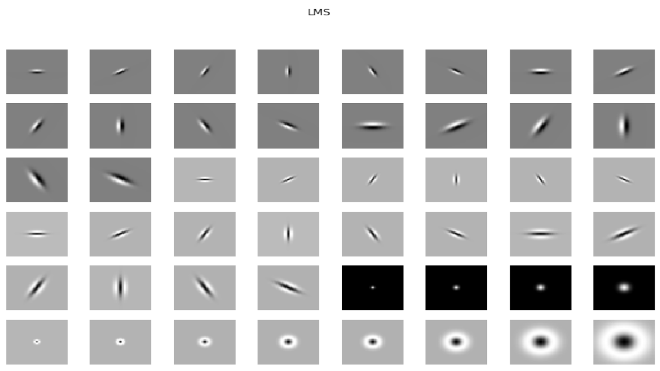

Fig. 3. Leung-Malik small filter

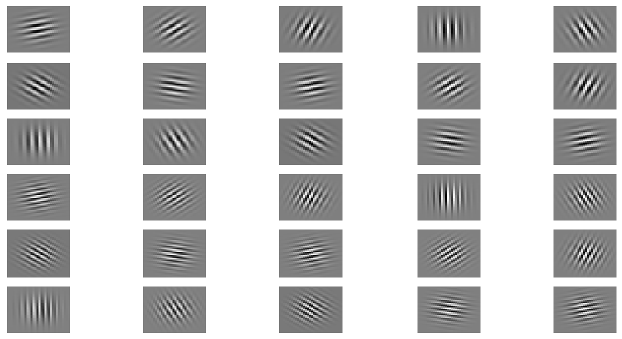

Fig. 4. Gabor filter

B. Texton, Brightness, Color map

The second step in the pipeline is to filter the input image with each element in our generated filter bank.We then use K-means clustering to generate the texture ID's.We replace the pixels in the input image with texton id at that point to generate a texton map. Following the same process, we use grayscale pixel value to generate the brightness map and RGB value to generate the color map.

Texton,brightness and color map of images 3,4,7,10 are given below (See fig. 5 to fig.8).

C. Texton, Brightness, Color Gradient and Half Disc

In order to generate the gradient maps for the above maps , we have to use half disc masks.The half disc masks are simple binary images of half-discs. We compute chi-square distances using half discs and then use that to generate the gradient maps

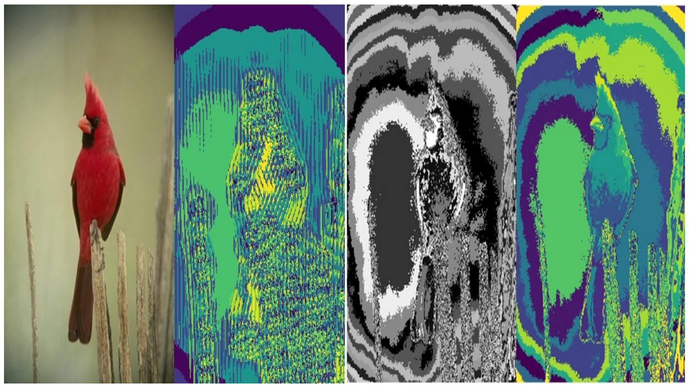

Fig. 5. Original Map,Texton Map,Brightness Map,Color Map

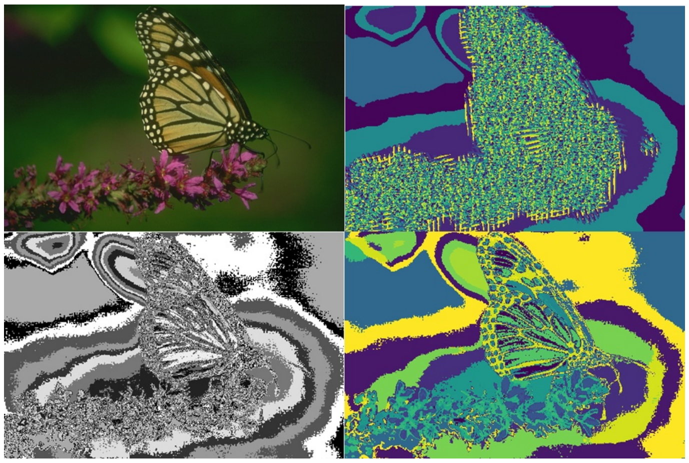

Fig. 6. Original Map,Texton Map,Brightness Map,Color Map

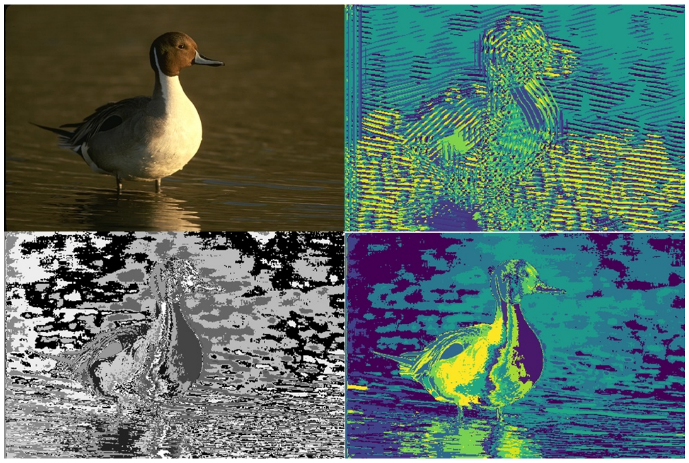

Fig. 7. Original Map,Texton Map,Brightness Map,Color Map

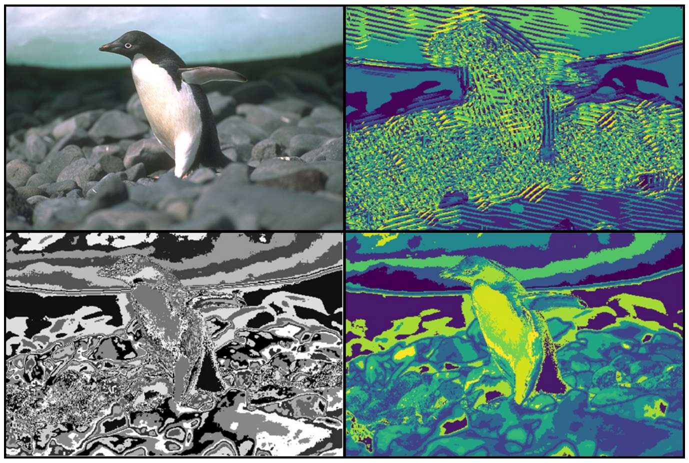

Fig. 8. Original Map,Texton Map,Brightness Map,Color Map

for all the texture, brightness and color information of the input image. Texton,brightness and color gradients of images 3,4,7,10 are given below (See fig.10 to fig.13).

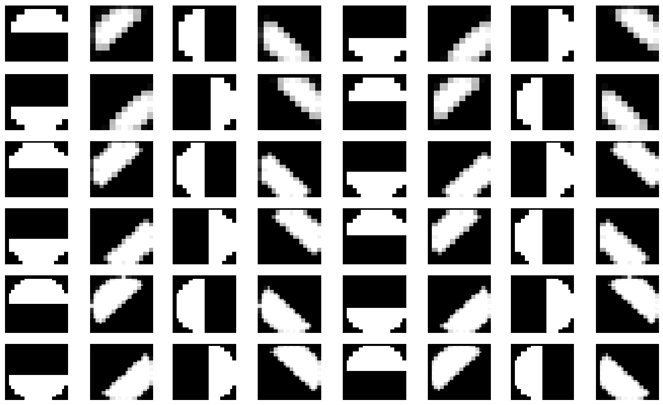

Fig. 9. Half-disc masks for various radious.

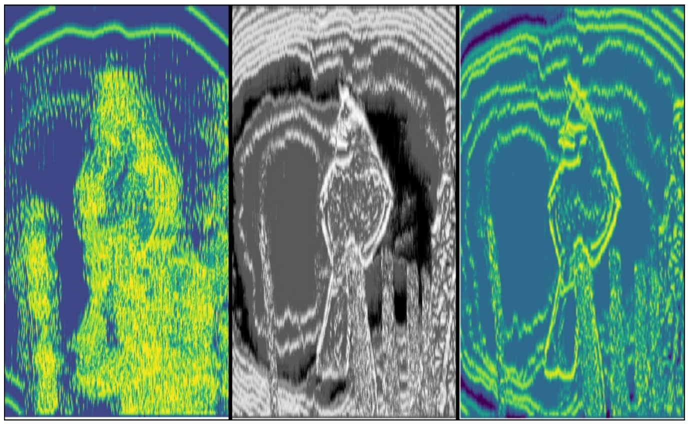

Fig. 10. Texton Gradient,Brightness Gradient, Color Gradient

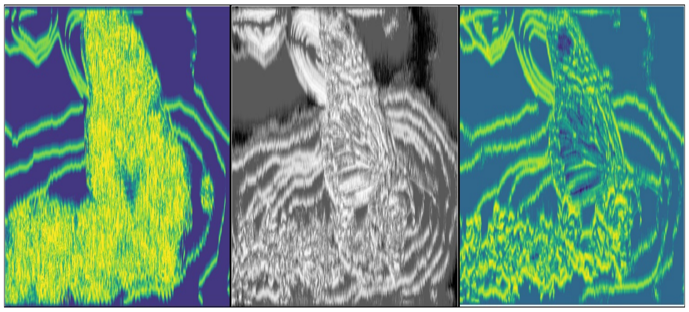

Fig. 11. Texton Gradient,Brightness Gradient, Color Gradient

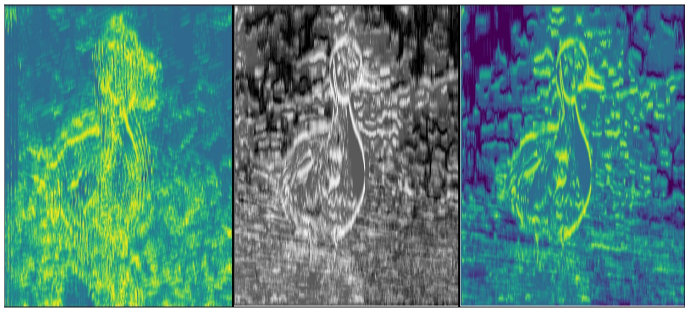

Fig. 12. Texton Gradient,Brightness Gradient, Color Gradient

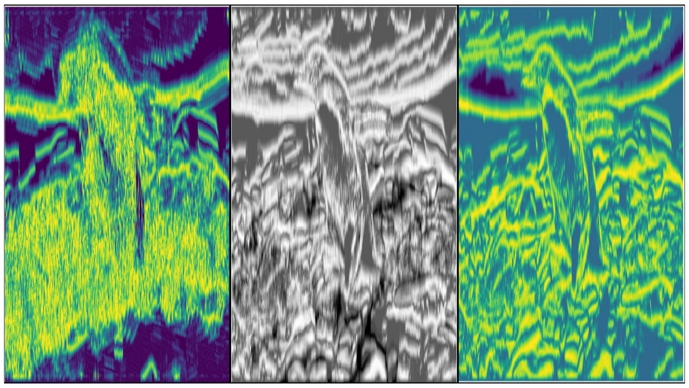

Fig. 13. Texton Gradient,Brightness Gradient, Color Gradient

D. Boundry Detection.

All of the above steps were done to achieve our final motive , the final step which is to combine the traditional (Canny and Sobel baseline) with our generated maps to produce the final output using the following formula:

$$PbEdges = \frac{(\mathcal{T}_g + \mathcal{B}_g + \mathcal{C}_g)}{3} \odot (w_1 * cannyPb + w_2 * sobelPb)$$

The only constraint in this formula is w1 + w2 should be equal to 1.

The results of images 3,4,7,10 are given below(fig.14 to fig.17).

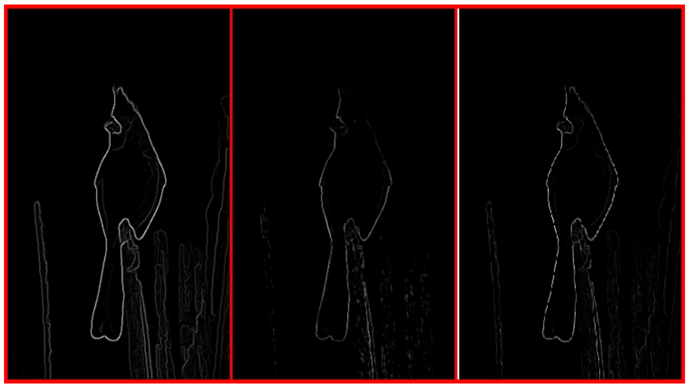

Fig. 14. Texton Gradient,Brightness Gradient, Color Gradient

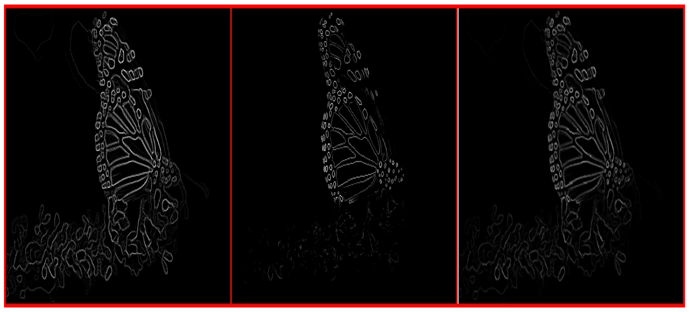

Fig. 15. Texton Gradient,Brightness Gradient, Color Gradient

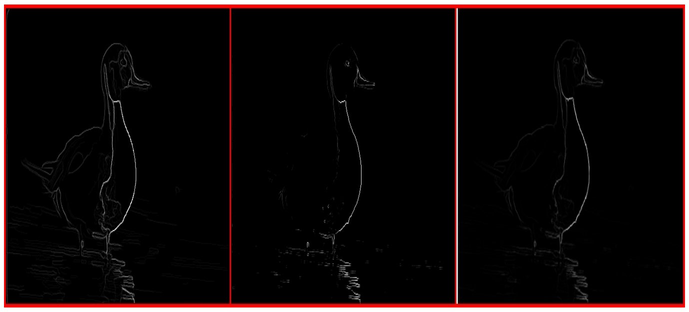

Fig. 16. Texton Gradient,Brightness Gradient, Color Gradient

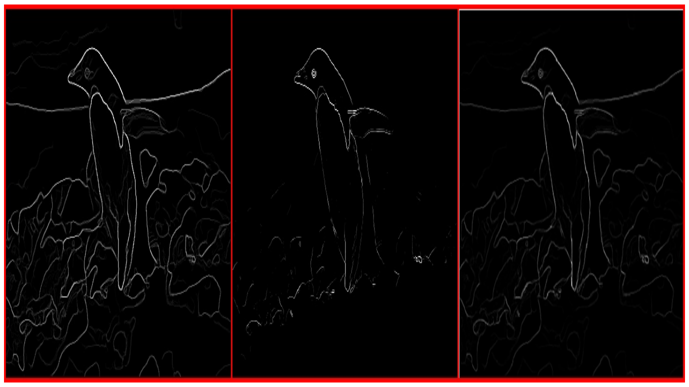

Fig. 17. Texton Gradient,Brightness Gradient, Color Gradient

II. Conclusion

The basic idea was that PB output is much better in finding boundaries as it takes in various other information to detect boundaries as compared to that of Canny and Sobel. My results for the PB lite are not that good and accurate and could be improved by tweaking various parameters in the above steps. We know that Sobel and Canny generate false positives for images with lot of texture and thus pb lite suppress such information to give better results. In much case my implementation would have suppressed important information to some extent and thus the output is not as good as expected.

References

[1] Gaussian Smoothing : https://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm
[2] 2D Convolution : https://johnloomis.org/ece563/notes/filter/conv/convolution.html
[3] Gabor Filter : https://en.wikipedia.org/wiki/Gabor_filter
[4] Gaussian Derivative : https://dsp.stackexchange.com/questions/19175/sobel-vs-gaussian-derivative
[5] filters : http://www.robots.ox.ac.uk/ vgg/research/texclass/filters.html
[6] http://cs.brown.edu/courses/cs143/2011/proj2/