RBE 549 HW0: Alohmora

Aabha Tamhankar
Worcester Polytechnic Institute, Robotics Engineering
astamhankar@wpi.edu

Fig. 1. Derivative of Gaussian Filter Bank

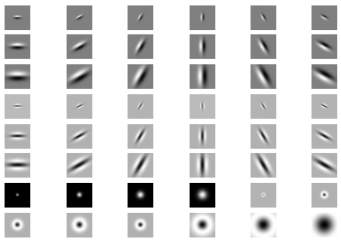

Fig. 2. Leung-Malik Filter Bank

I. Phase 1: Shake my Boundary

The goal of this phase was to detect boundary detection on images using a method called PB lite (Probability of Boundary) Detection Algorithm. This method uses information texture, colour and brightness information present in the given images and predicts the possibility of boundaries in them. In this particular assignment, ten images were taken as inputs for PB lite algorithm and the results were used to improve the results with other boundary detection methods like Canny and Sobel.
The steps taken to perform this algorithm are presented below.

A. Filter Banks

The first step was to filter the image with a set of filter banks. This helps in measuring texture properties and aggregating regional texture and brightness distributions. Here, three filters have been used namely, Difference of Gaussian Filters (DoG), Leung-Malik Filters, and Gabor Filter.

1) Difference of Gaussian Filters (DoG): The DoG filter is formed by convolving a simple Sobel filter and a Gaussian kernel and then rotating the result. For this filter bank, 2 scales (1 and 2) with 16 orientations (between 0 to 360 degrees) were used. The Gaussian kernel can be found out using the formula mentioned below.

$$G(x,y) = \frac{1}{2\pi\sigma^2}\exp -(\frac{x^2}{2\sigma^2})\exp -(\frac{y^2}{2\sigma^2}) \tag{1}$$

Thus, a filter bank of size 2 X 16, which is given in the figure 1.

Identify applicable funding agency here. If none, delete this.

2) Leung-Malik Filter Bank: The LM 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 first and second derivatives of Gaussians are used with 3 scales and 16 orientations so, 36 DoG, 8 LoG and 4 Gaussian filters are used in the LM filter bank, making a total of 48 LM filters.
The Gaussians in this filter were calculated in a similar process as DoG filters, but they had two different standard deviations (σs) along the x and y axes; σ_x and σ_y, such that σ_x = 3σ. The Gaussian kernels were then convolved with Sobel filter to get a First Order Gaussian. Furthermore, these first order Gaussians were convolved another time with Sobel filter to get the Second Order Gaussian.
The Laplacian of Gaussian was calculated using the formula

$$L(x,y) = -\frac{1}{\pi\sigma^4}[1-\frac{x^2+y^2}{2\sigma^2}]\exp -(\frac{x^2+y^2}{2\sigma^2}) \tag{2}$$

at σ_x and $3\sigma_y$

Two such LM filter banks (LM small and LM large) were applied to image, where the LMS scales were [1,$\sqrt{2}$ and 2], while the LML scales were [$\sqrt{2}$, 2 and $2\sqrt{2}$]. The filters can be seen in the figure 2.

3) Gabor Filters: Gabor Filters are designed based on the filters in the human visual system. A gabor filter is a gaussian kernel function modulated by a sinusoidal plane wave. They can be calculated by using the formula

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp -(\frac{x'^2+\gamma y'^2}{2\sigma^2}) - \cos(\frac{2\pi x'}{\lambda}+\psi) \tag{3}$$

where,

$$x' = x\cos(\theta) + y\sin(\theta) \tag{4}$$

$$y' = -x\sin(\theta) + y\cos(\theta) \tag{5}$$

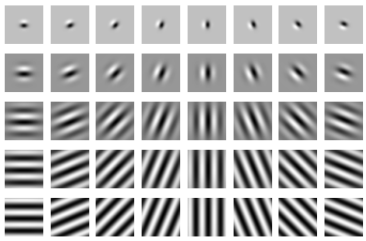

Fig. 3. Gabor Filter Bank

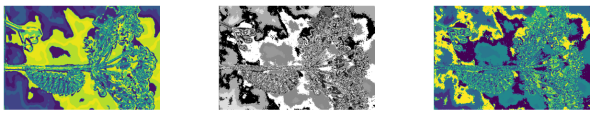

Fig. 4. Texton, Brightness and Colour Maps for Image1

Gabor Filters are seen in figure 3.

B. Texton, Brightness and Color Maps

The next step is to create the texton, brightness and colour maps for these images, in order to map the differences in these parameters in the image. Since 3 filters are used in the filtering process, it gives us 3 filter responses at each pixel. Then, KMeans clustering can be used on the vectors, to get the required maps.

At the end of filtering process, we get a image stack with size m X n X N, where m X n is the size of image, and N is the number of filters applied in the process. Then, they are clustered into K textons using K-Means. This generates an image which captures the texture changes in the original image.

The Brightness maps the intensity change in the pixels of image. We can use K-means on the grayscale images to map the output of brightness maps.

The Colour maps the changes in colour values in the image, and can be plotted similar to the brightness maps.

The texton, brightness and colour maps of each of the ten images have been plotted in the figure **??**

C. Half-Disk Masks for Gradients

The texton, brightness and color maps are further used to calculate the respective gradients, Tg, Bg, and Cg, which give the texture, brightness and color changes at each pixel.

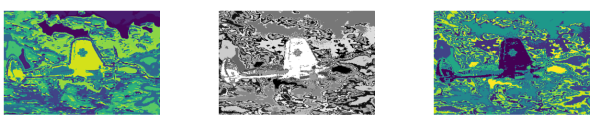

Fig. 5. Texton, Brightness and Colour Maps for Image2

Fig. 6. Texton, Brightness and Colour Maps for Image3

Fig. 7. Texton, Brightness and Colour Maps for Image4

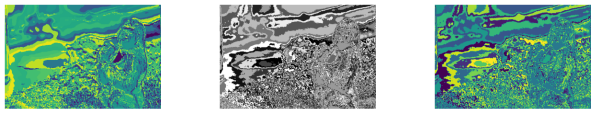

Fig. 8. Texton, Brightness and Colour Maps for Image5

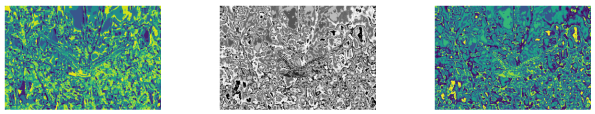

Fig. 9. Texton, Brightness and Colour Maps for Image6

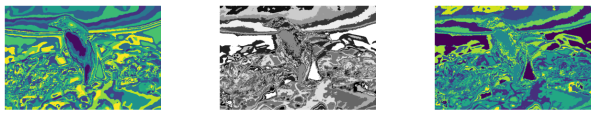

Fig. 10. Texton, Brightness and Colour Maps for Image7

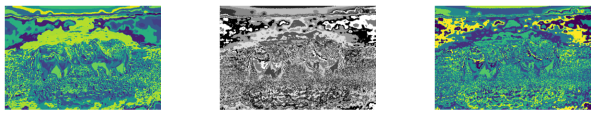

Fig. 11. Texton, Brightness and Colour Maps for Image8

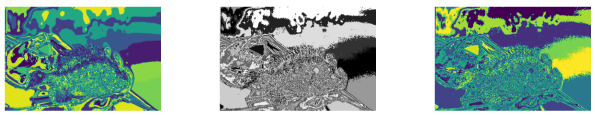

Fig. 12. Texton, Brightness and Colour Maps for Image9

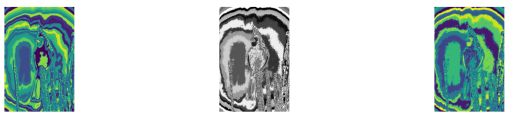

Fig. 13. Texton, Brightness and Colour Maps for Image10

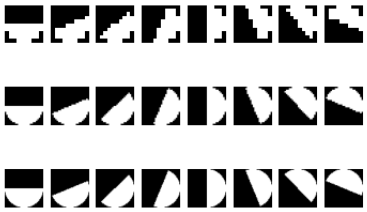

Fig. 14. Left Half-Disks

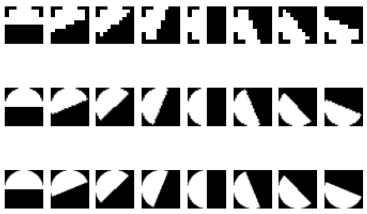

Fig. 15. Right Half-Disks

For this, left and right half-disks are used which compare the values at each pixels. Using this along with Chi-square distance, we get the images in gradients which are plotted in figure 15

D. Canny and Sobel Baselines

The gradients found in the earlier section were then combined with the canny and sobel baselines given in the problem. This was done using the Hadamard matrix product with two weights for canny and sobel that should together be added up to 1.

E. Results

The PB lite method helps with flexibilty of information withdrawn from an image. By changing parameter, we can control edge visibilty, which is a good advantage over methods of Canny and Sobel. The Canny and Sobel and PBLite Edge Detections are compared in the figure 25 for all images.

II. Phase 2: Deep Dive on Deep Learning

The goal of this phase was to perform deep learning operations of convolution neural network (CNN) on a CIFAR-10 dataset for image classification.

A. Creating a Neural Network

The CNN Network created consists of three convolutional layers with reLu activation and max-pooling to down-sample the dimensions of the image. The softmax cross entropy loss function was used on the network logits and it was optimized using the Adam optimizer. This was trained over 30 epochs with mini batch size of 64. The accuracy obtained was 42.38% The loss over epoch can be seen in the figure 26

Fig. 16. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image1

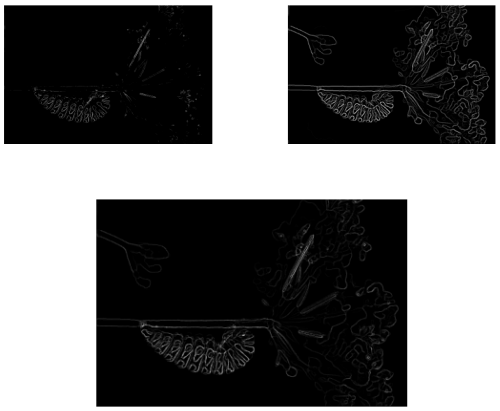

Fig. 17. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image2

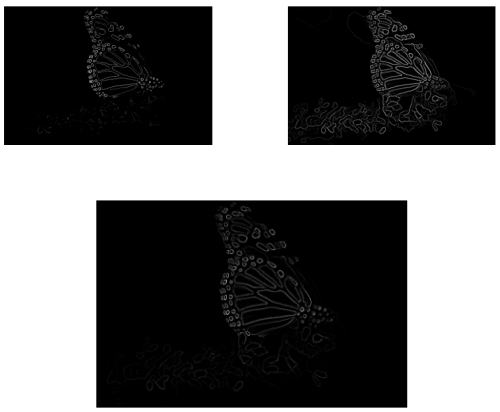

Fig. 18. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image3

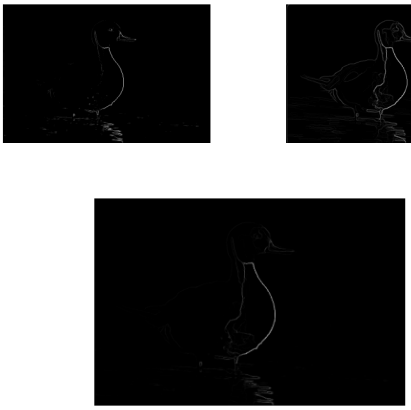

Fig. 19. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image4

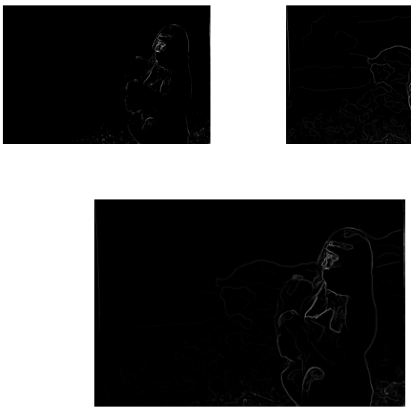

Fig. 20. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image 5

Fig. 21. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image6

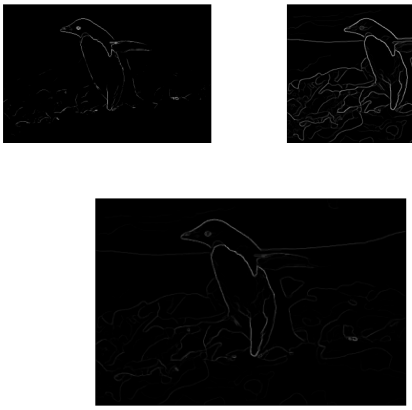

Fig. 22. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image7

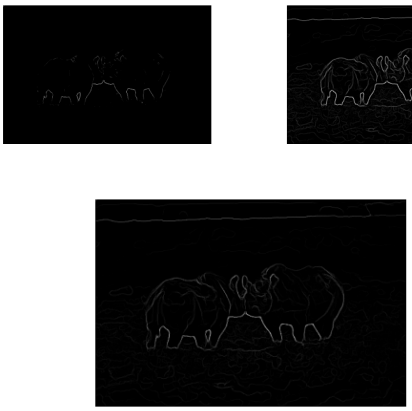

Fig. 23. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image8

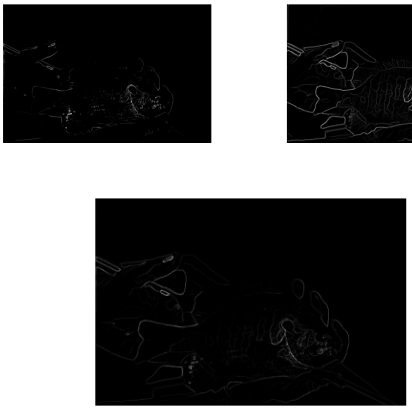

Fig. 24. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image9

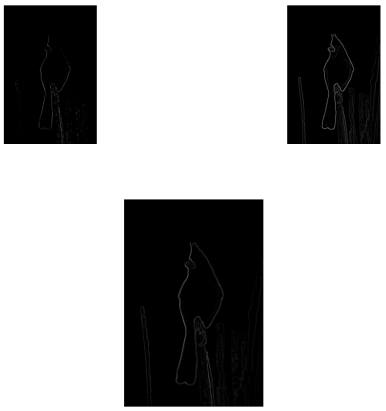

Fig. 25. Canny(top-right), Sobel(top-left) and PBLite(bottom) Outputs of Image10

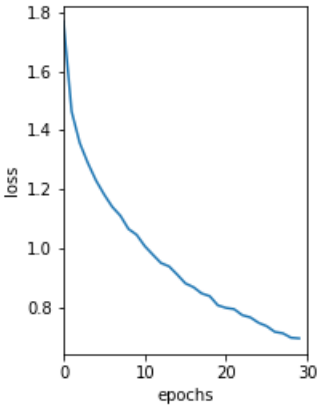

Fig. 26. Training Loss vs Epoch

The confusion matrix for test is in the figure 27. This is a matrix which presents actual and predicted outcomes in a matrix form. Here, each row maps instances in actual class, and each column maps instances in the predicted class. The matrix makes it easy to differentiate the true and false positives or negetives.

B. Improving the Neural Network

Since a simple network has been implemented, we do not see much accuracy. However, there are other methods we can use to increase this accuracy. Some of these methods are:

1) Standardize the data input.
2) Decay the learning rate while training or increase batch size while training.
3) Augment the data to artificially make dataset larger.
4) Add Batch Normalization between layers.
5) Change the hyperparameters in the architecture such as number of layers, number of neurons.
6) Use of innovative neural networks like ResNet, ResNeXt or DenseNet

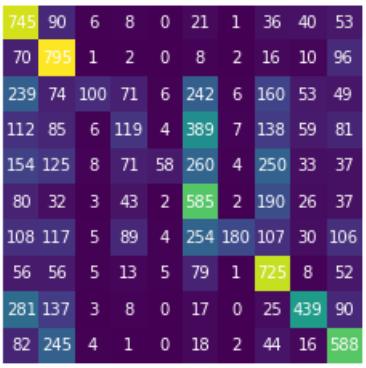

Fig. 27. Testing Confusion Matrix Results

C. Other Neural Networks

1) Residual Network (ResNet): esNet is a neural network architecture which solves the problem of vanishing gradients by applying skip connections in a general residual block. his allows the network to accommodate deep layers without having the vanishing gradient problem.

2) ResNeXt: ResNeXt is a homogeneous neural network which reduces the number of hyperparameters required by conventional ResNet. This is achieved by their use of ”cardinality”, an additional dimension on top of the width and depth of ResNet. Cardinality defines the size of the set of transformations.

3) Densely Connected Convolutional Networks(DenseNet): In DenseNet each layer connects to every other layer in a feed-forward fashion. If there are n number of filters, DenseNet will have n(n+1)/2 connections. For each layer the feature maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs to all subsequent layers.

III. Conclusion

In Phase 1, we can see that the method of PB lite edge detection is significantly adaptable to the user while plotting the required edges. Thus, by changing many of the parameters in the functions used in this method, an image with required highlights can be obtained.

In Phase 2, deep learning was used in classification of an image data of 60000 images consisting of 10 classes. Though only one basic network was used to do so, there are better implementations which increase the accuracy to the maximum.

References

[1] Pablo Arbelaez, Charless Fowlkes, ”Contour Detection and Hierarchical Image Segmentation”
[2] https://www.robots.ox.ac.uk/ vgg/research/texclass/filters.html
[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, ”Deep Residual Learning for Image Recognition”
[4] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, Kaiming He, ”Aggregated Residual Transformations for Deep Neural Networks”
[5] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger ”Densely Connected Convolutional Networks”