Hengyue Liu
4107-2966-75
hengyuel@usc.edu
EE 569 Homework #2
October 9, 2015

Problem 1: Texture Analysis and Classification

a) Texture Classification: Two Classes + Minimum Mean Distance Classifier

I Abstract and Motivation

Texture classification is a hot topic in image processing and computer vison. In lecture, several binary classification methods are introduced. Basically, using original 25-dimension feature vectors, or using reduced feature vectors via Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA) will produce fair results, still, there are pros and cons for each approach, with different error rate. By comparing these methods, one can get better understanding of texture analysis and classification, which is useful for future research and development.

II Approach and Procedures

First step, read raw texture images and do the pre-processing, i.e. mean subtraction to the images. This step is to remove DC component of sub-band channel for obtaining less data fluctuation.

Second step, apply 25 2D Laws filters to each image. Each 2D filter is the tensor product of 2 1D kernels with length 5.

Then, do the local energy computation. Since one image only has one kind of texture, this step is simplified to calculate average energy of whole image.

Next, apply vector normalization. By last step, the 25-D feature vectors are generated. But they have large range, so map the vector values to [0,1] for convenience.

Important step, apply feature reduction by PCA or LDA, also original 25-D vector space is acceptable for comparison. To perform PCA or LDA, it is necessary to group the data into training data and testing data. Testing data can be part of training data too.

Finally, calculate Mahalanobis distance of reduced new feature vector to each class' mean vector, and record wrong labeled number, then calculate error rate.

III Experimental Results

Shown below are the results for part a).

1) Feature Extraction:Figure 1 shows the original straw texture image, and Figure 2 is the mean-subtracted image of Figure 1.

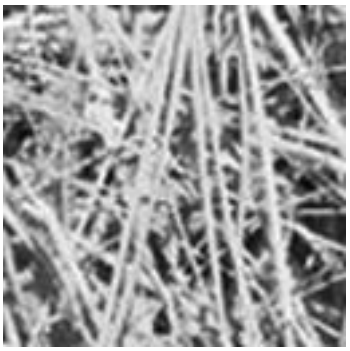


Figure 1 straw_01.raw

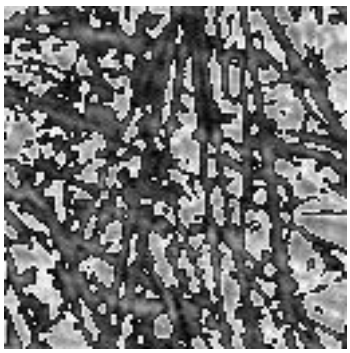


Figure 2 Mean-subtracted one

2) Minimum Mean Distance Classification using 25-D feature vector error rate:

```
Reading trainning images...
Done! Labeling via 25D feature...
Done!
Using 25D feature vector, the error rate is: 25%.
```

Figure 3 25D feature classification result

3) Apply PCA and LDA to the feature vectors, then plot the projected points for 1D or 2D cases:

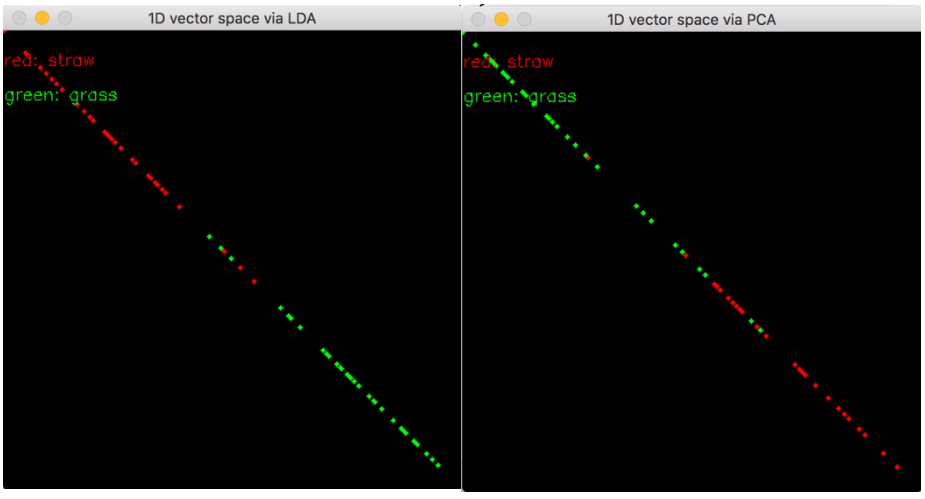


Figure 4 1d vector space via LDA

Figure 5 1d vector space via PCA

2D case for PCA (since LDA can only be reduced to 1 dimension under 2 classes case):

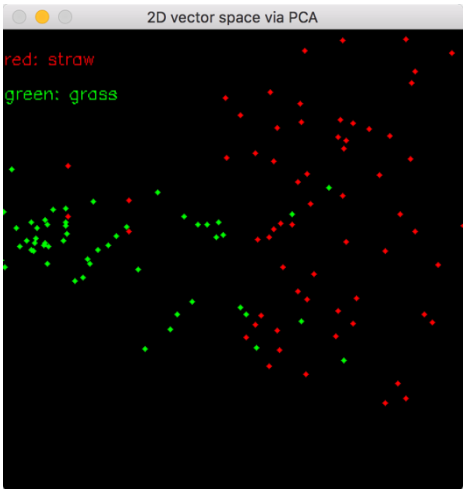


Figure 6 2D vector space via PCA

4) Some results of classifying unknown test data and training data as test data:

```
PCA...
1-D PCA error rate for classifying unknown from 24 testing images is: 0%.
1-D PCA error rate for classifying straw from 36 testing images is: 5.556%.
1-D PCA error rate for classifying grass from 36 testing images is: 16.67%.
LDA...
1-D LDA error rate for classifying unknown from 24 testing images is: 8.333%.
1-D LDA error rate for classifying straw from 36 testing images is: 5.556%.
1-D LDA error rate for classifying grass from 36 testing images is: 8.333%.
```

```
PCA...
2-D PCA error rate for classifying unknown from 24 testing images is: 0%.
2-D PCA error rate for classifying straw from 36 testing images is: 5.556%.
2-D PCA error rate for classifying grass from 36 testing images is: 22.22%.
```

```
PCA...
5-D PCA error rate for classifying unknown from 24 testing images is: 12.5%.
5-D PCA error rate for classifying straw from 36 testing images is: 2.778%.
5-D PCA error rate for classifying grass from 36 testing images is: 25%.
```

```
navigator
trainning images...
beling via 25D feature...
Done!
Using 25D feature vector, the error rate is: 25%.
PCA...
1-D PCA error rate for classifying from 24 testing images is: 0%.
****************
1-D PCA error rate for classifying from 72 training images is: 0.111111%.
****************
LDA...
1-D LDA error rate for classifying from 24 testing images is: 8.33333%.
****************
1-D LDA error rate for classifying from 72 training images is: 6.94444%.
****************
```

Figure 7 2D PCA LDA results

IV Discussion

For P1 (a) 3) , the answer is no. The 25 features are not equally important. Some features are dominant with large variance, some are not and even distractive. So, PCA or LDA are used for reducing the dimension to get better results. For P1 (a) 4) , Figure 4 shows the 1D vector distribution of two classes. The two classes are well clustered and easy to separate.

For For P1 (a) 6). Figure 7 shows the results of PCA and LDA respectively.

Using 24 unknown test data, the PCA has the best performance based on the results. No error occurred in classification, one reason for this is that the straw and grass classes are well-separated, and another reason may be the training data is not enough. However, when increasing the dimension of PCA, for example, 5 dimensions showed in Figure 7, the error rate increases.

When using training data as test data, the advantages of LDA are notable. In Figure 7, 36 testing images are all straw or grass, the LDA results are better than PCA since class labels are using in finding LDA space basis.

Taken together, in 1D case, PCA has better performance than LDA in classifying unknown data; LDA has lower rate in classifying labeled training data. One more thing, note that when error rate of classifying straw increases, the error rate of classifying grass is the other way around. This is caused by the binary call of calculating distance of mean vectors, perhaps.

b) Advanced Texture Classification: Multi-Classes + SVM

I Abstract and Motivation

When more classes involved in the texture analysis, PCA and LDA can only classify test data into two groups, one is class and the other is non-class. So, for 4 classes, 4 times of PCA or LDA are need to run, then calculate the error rate for classifying each class from combing 4 results. As for SVM, hyperplanes are derived to cut the classes. SVM should be more accurate than the other two methods.

II Approach and Procedures

First, random select training data and testing data. Apply 3D PCA and LDA reduction. Then, use 4 classes minimum-distance-to-class-mean classifier individually. In other words, classify test data into straw and non-straw, grass and non-grass, etc.

Second, 4 predicted labels are generated from first step, we can calculate the error rate for each class then. If one class is classified as more than one class, or still classified as unknown, take this test data as random class. Record error rates. However, is this not an optimal solution, one way is to record this data and run the procedure in different training data for classifying the right class.

Then, take each part of training data as test data, and record error rates.

III Experimental Results

Shown below are the results for part b).

For (b) 1), sample 3D new feature vectors are like:

```
new 3-D feature vectors:
[-0.4019714, -0.23693199, 0.1316362]
[-0.37671512, 0.061304536, 0.49078506]
[-0.62976909, -0.45402002, 0.83732319]
[0.21989584, 0.077864885, 0.41025546]
[-0.22958297, 0.34675446, -0.13165237]
[-0.7347551, 0.15021276, 0.10242999]
[-0.0097110793, -0.37303361, 0.048596103]
[-0.86599994, 0.21686243, 0.27311072]
[0.033899676, -0.44219786, -0.37870437]
[0.076420963, -0.43674374, -0.30160499]
[0.76532537, -0.074370518, 0.51033908]
[-0.42987189, 0.42333403, 0.069109067]
[-0.42046383, 0.19516587, -0.33127534]
[0.016117213, -0.0015863858, 0.26453981]
[0.14191961, -0.016472032, 0.19705811]
[-0.038381524, 0.61928284, 0.11009541]
[-0.31235018, 0.43019265, 0.49502492]
[-0.82098353, -0.39598209, -0.18762384]
[0.1000028, 0.53607482, -0.37993088]
[-1.1145042, 0.015221318, -0.41262898]
[-0.17716417, -0.62838519, 0.50657868]
[0.7476989, -0.10150354, 0.1677793]
[0.18135403, -0.39644232, -0.03283786]
[-0.18620226, 0.33998317, -0.15848243]
[-0.038964693, -0.11786316, 0.082210913]
```

Figure 8 3D feature vector from PCA

For (b) 2), all results are shown below:

```
4 classes classifier...
Reading all images...
PCA...
For 48 test images, PCA results:
3-D PCA error rate for classifying straw from 48 testing images is: 27.08%.
3-D PCA error rate for classifying grass from 48 testing images is: 6.25%.
3-D PCA error rate for classifying sand from 48 testing images is: 33.33%.
3-D PCA error rate for classifying leather from 48 testing images is: 20.83%.
For each 36 group of training class images, PCA results:
3-D PCA error rate for classifying straw from 36 testing images is: 10.42%.
3-D PCA error rate for classifying grass from 36 testing images is: 14.58%.
3-D PCA error rate for classifying sand from 36 testing images is: 11.11%.
3-D PCA error rate for classifying leather from 36 testing images is: 6.25%.
For 48 test images, LDA results:
3-D LDA error rate for classifying straw from 48 testing images is: 31.25%.
3-D LDA error rate for classifying grass from 48 testing images is: 14.58%.
3-D LDA error rate for classifying sand from 48 testing images is: 25%.
3-D LDA error rate for classifying leather from 48 testing images is: 27.08%.
For each 36 group of training class images, LDA results:
3-D LDA error rate for classifying straw from 36 testing images is: 6.944%.
3-D LDA error rate for classifying grass from 36 testing images is: 9.722%.
3-D LDA error rate for classifying sand from 36 testing images is: 9.028%.
3-D LDA error rate for classifying leather from 36 testing images is: 4.167%.
Program ended with exit code: 0
```

```
4 classes classifier...
Reading all images...
PCA...
****************
For 48 test images, PCA results:
3-D PCA error rate for classifying from 48 testing images is: 41.6667%.
****************
****************
For 144 traing images as test data, PCA results:
3-D PCA error rate for classifying from 144 training images is: 88.8889%.
****************
****************
For 48 test images, LDA results:
3-D LDA error rate for classifying from 48 testing images is: 50%.
****************
****************
For 144 traing images as test data, PCA results:
3-D LDA error rate for classifying from 144 training images is: 83.3333%.
****************
```

Figure 9 results via 4-class classifier

For (b) 3), all results are shown below:

```
SVM Response is: 0 0
SVM Response is: 0 0
SVM Response is: 3 0
SVM Response is: 0 0
SVM Response is: 3 0
SVM Response is: 0 0
SVM Response is: 2 0
SVM Response is: 0 0
SVM Response is: 0 0
SVM Response is: 3 0
SVM Response is: 3 0
SVM Response is: 3 0
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 2 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 1 1
SVM Response is: 3 2
SVM Response is: 3 2
SVM Response is: 0 2
SVM Response is: 0 2
SVM Response is: 2 2
SVM Response is: 0 2
SVM Response is: 2 2
```

Figure 10 sample SVM response compared with correct labels on the right via 4-class classifier

```
SVM error rate via PCA 3D feature vectors is: 45.83%
```

```
SVM error rate for 144 training images via PCA 3D feature vectors is: 46.53%
```

Figure 11 SVM result 1 via

```
SVM error rate via LDA 3D feature vectors is:
13.89%
SVM error rate for 144 training images via LDA 3D
feature vectors is: 31.25%
```

Figure 12 SVM result 2

IV Discussion

Clearly, SVM has better performance than minimum-distance-to-class-mean classifier. Meanwhile, using LDA features is better than using PCA features in SVM. When number of classes increases from 2 to 4, the classification gets harder too. The MN classifier can only make binary decision on the classification, so when applying 4 times of MN classifier, the error rate grows fast by just "cutting" all data into two classes. On the other hand, SVM performs in a non-linear way so that each class is separated more reasonably than MN classifier.

Problem 2: Edge Detection

a) Sobel Edge Detector and Non Maximal Suppression

I Abstract and Motivation

There has over 40 years' development history for edge detection techniques. From 1970, Sobel, Prewitt, LoG, these 1st and 2nd order local gradient methods built the foundations of edge detection. Then, Canny edge detector achieves better performance, and Machine Learning based detector did even better. Nowadays, Random Forest decision trees and Structured Edge are new techniques that increase the accuracy remarkably than traditional ways. In this part, several mentioned and important edge detectors are implemented and applied. Then based on the F-measure results, compare each method's advantages and disadvantages.

II Approach and Procedures

For part (a), Sobel edge detector is implemented.

For 1): First, read raw image and convert RGB 3 channels to grey channel. Then, design

2D mask of Sobel kernel, and apply it to the gray image. Next, with the gradient map, normalize it into range 0-1, then multiply by 255.

For 2): Sort the gradient values from the largest to lowest, then find the location of 10% and 15% pixels are edge points; get the gradient value of that pixel, then thresholding those below gradient values to zero.

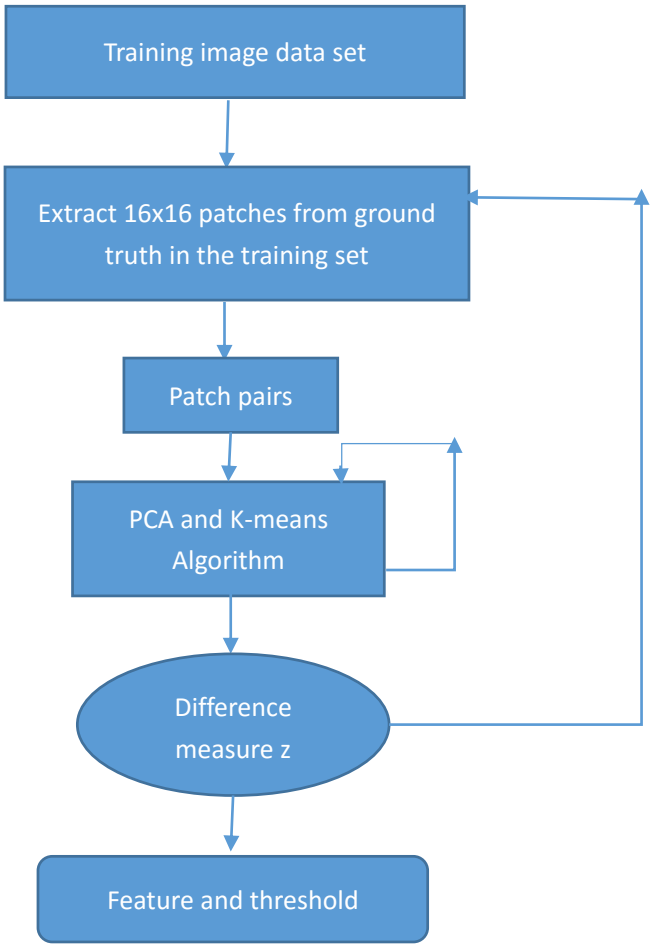
For 3): Non-maximal-suppression (NMS) is designed to reduce the thickness of the edge. Ideally, one edge point is only connected to at most two edge-points. First step of NMS is, for each pixel, record its gradient value, then calculate its neighbors' local maxima; if the pixel gradient value is larger than local maxima, then it is defined as edge point, otherwise set it to zero. This procedure goes through each pixel, the gradient map will be more accurate by NMS.

For part (b): just apply Canny detector in OpenCV with different parameters, then compare the results.

For part (c): ST: K-means clustering to label the Tokens

1) Structure Edge detector algorithm is like this:

The idea is using edge structure directly instead of predefined labels. Extract 16x16 patches from ground truth in the training set to 2^256 structured output for each patch. Then use PCA and K-means to reduce feature to 2 for each patch pair until reaching the leaf node, then feature and threshold can be obtained.

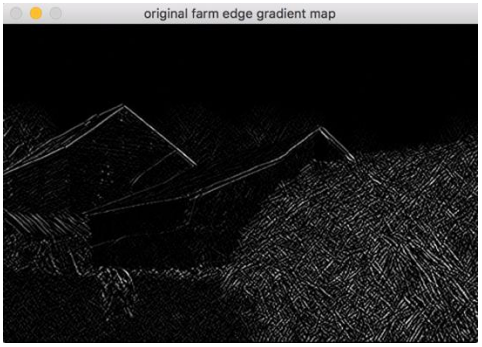


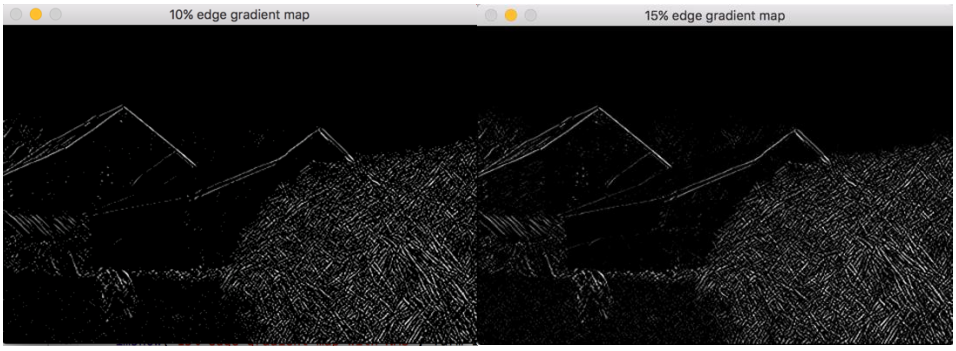
2) Randomly select contour patches and non-contour patches; take random trees to train, each tree calculates entropy of choosing certain feature for each node; trees are trained until reaching the leaf nodes; average trees and calculate the edge probability for each pixel. The advantage of Random Forest is that, it takes data randomly and feature randomly, and the last step of averaging provides the accuracy of prediction.

III Experimental Results

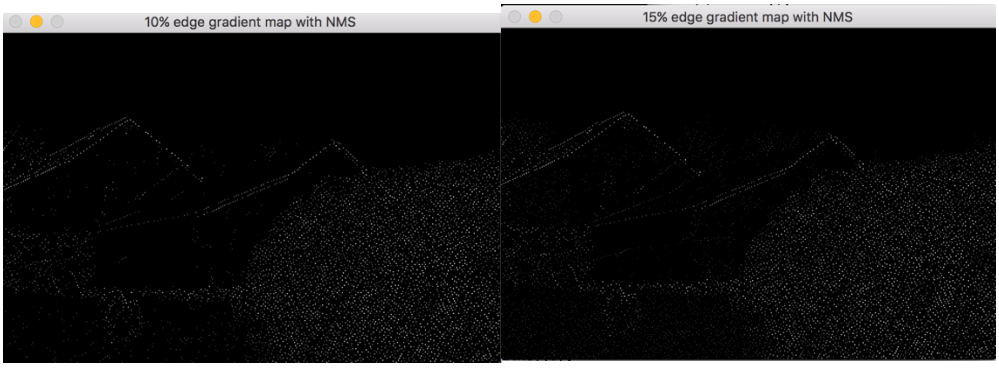
Shown below are the results:

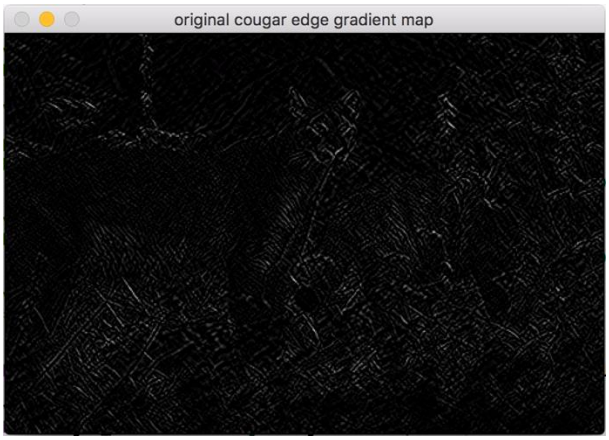
Part 1)

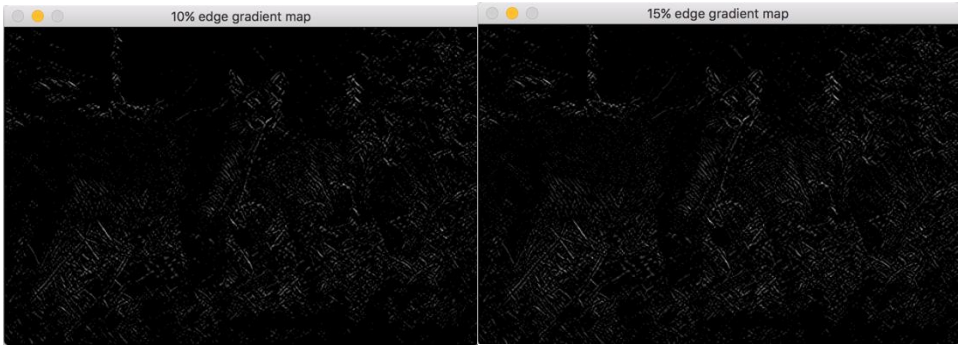




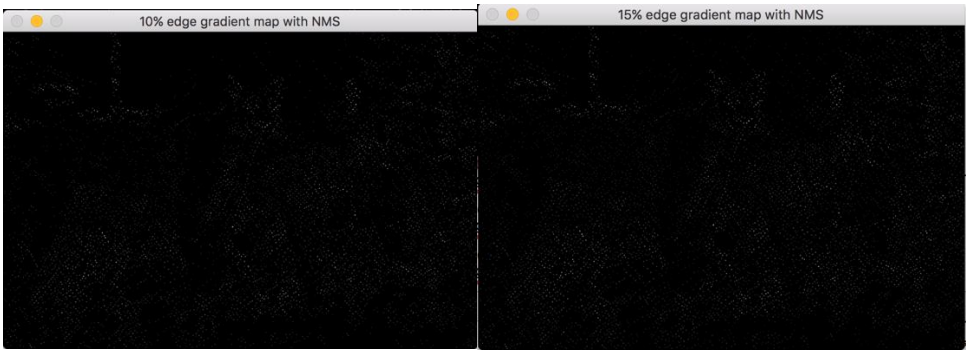

```
Farm image results:
Original gradient map has 44.8307% of points are edge points.
New reduced gradient map has 9.98828% of points are edge points.
Original gradient map has 44.8307% of points are edge points.
New reduced gradient map has 14.9721% of points are edge points.
Program ended with exit code: 0
```



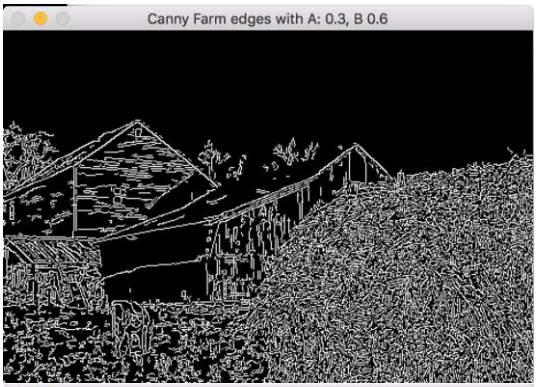


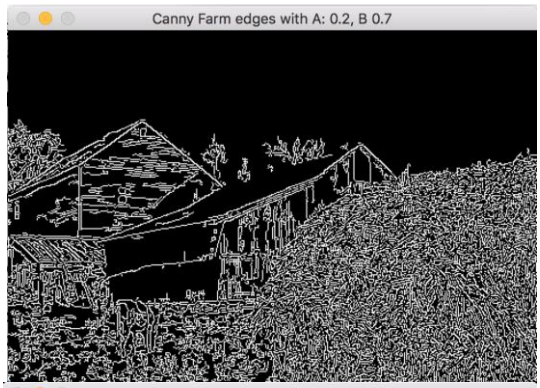


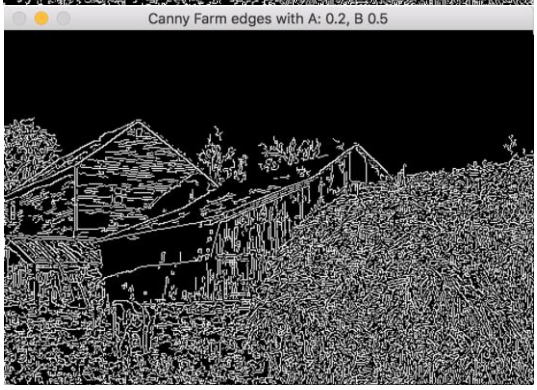

```
Cougar image results:
Original gradient map has 48.2471% of points are edge points.
New reduced gradient map has 9.85356% of points are edge points.
Original gradient map has 48.2471% of points are edge points.
New reduced gradient map has 14.6068% of points are edge points.
Program ended with exit code: 0
```

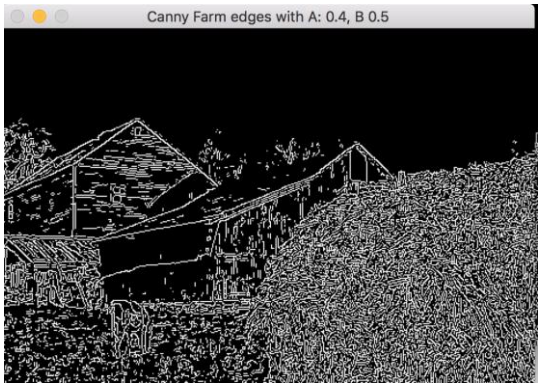


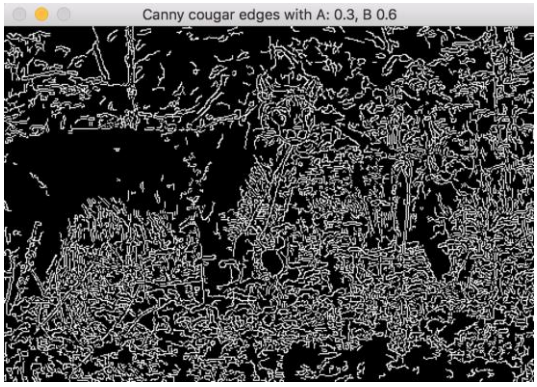
Shown below are the results for part 2)

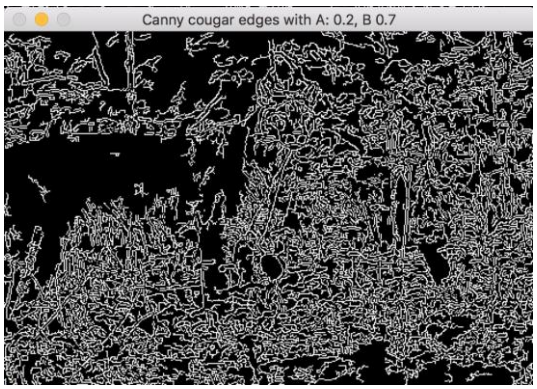


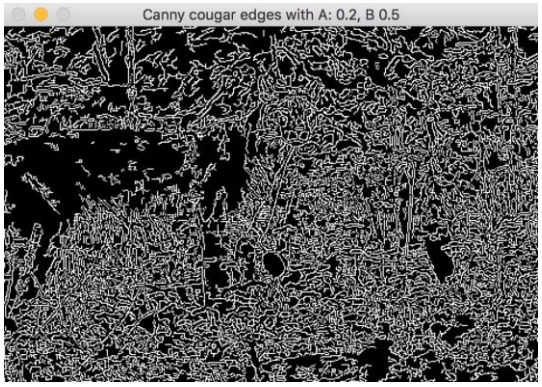


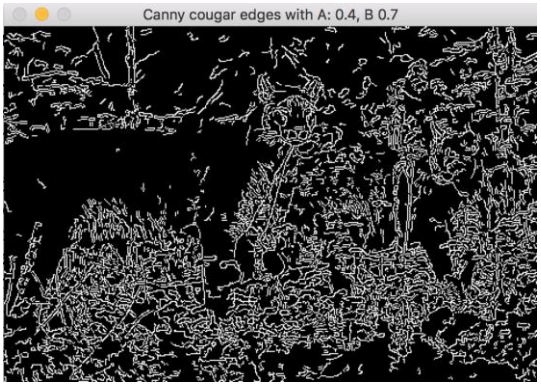


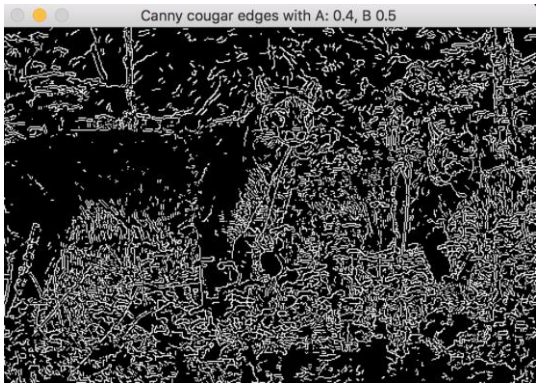










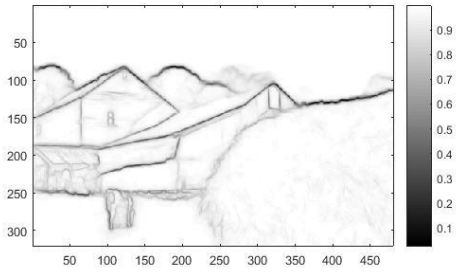


Shown below are the results for part 3)

The F measure is 0.73 with Multiscale = 0.

Parameters of Structure Edge detector

```
12
13    %% set detection parameters (can set after training)
14 -  model.opts.multiscale=1;          % for top accuracy set multiscale=1
15 -  model.opts.sharpen=2;            % for top speed set sharpen=0
16 -  model.opts.nTreesEval=4;         % for top speed set nTreesEval=1
17 -  model.opts.nThreads=4;           % max number threads for evaluation
18 -  model.opts.nms=0;                % set to true to enable nms
19
```

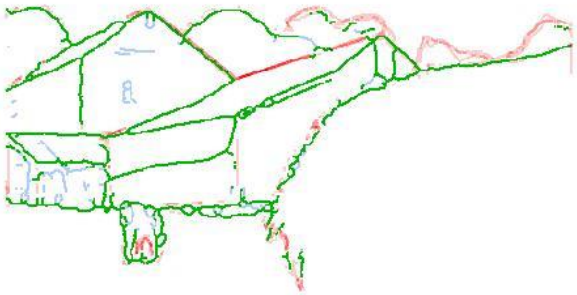


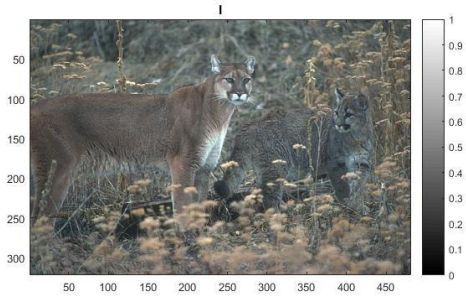
The maxima F-measure visualization edge map of farm image:

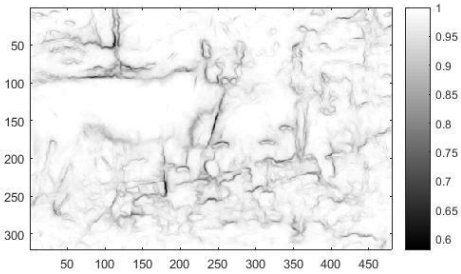
```
>> max(F)

ans =

    0.8129
```





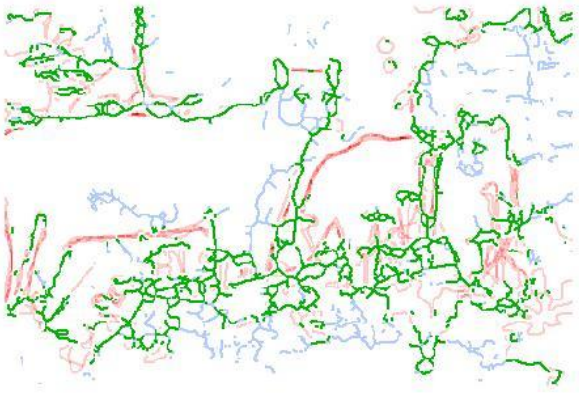


The maxima F-measure visualization edge map of cougar:

```
>> max(F)

ans =

    0.6830
```



These are set detection parameters (can set after training)

model.opts.multiscale=1; % for top accuracy set multiscale=1

model.opts.sharpen=2; % for top speed set sharpen=0

model.opts.nTreesEval=4; % for top speed set nTreesEval=1

model.opts.nThreads=4; % max number threads for evaluation

model.opts.nms=0; % set to true to enable nms

IV Discussion

For (b), the low and high thresholds decide the hysteresis thresholding procedure. Normally the ratio of high threshold over the low one is between 2 and 3 for better performance.

Take results 2) and 3) as comparison. First, the low thresholds are the same for these two results, the high thresholds, one is 0.5 and the other is 0.7. Result 2) is smoother than result 3) since fewer points are set to edge points, in terms of the edge detection of the house shape. However, less edges cause weaker boundary of objects. The roof boundary of result 2) is not preserved well, and in the same situation, result 4) gets worse.

Take results 2) and 4) as comparison then. They both have 0.7 high threshold, but 3) has lower low threshold. For house shape and straws, the results are similar because these textures' gradient changes drastically that can be easily classified as edges. The difference is on grass texture, result 4) has better performance on ignoring the grass, so the image is less messy than

result 2).

For all 5 results, the Canny detector classifies the straw textures as edges. Actually, only the boundary of the straws is important and should be maintained, not the whole content of the textures. This problem is even distractive in the cougar image. All cougar results failed to detect the contour of the cougar, and preserved too much foreground and background information. This reveals that the Canny detector is limited in some particular type of images, like images with many textures or similar foreground and background.

For (c):

Without a doubt, the best one is Structure Edge. The F-measure with proper threshold can achieve even over 0.8 in one single image (OIS). The evaluation image showed good results on both farm and cougar image. From this point of view, it can be concluded that structured edges are well detected in the cougar image, which means it reduces the influence of texture and background. Hence, it is much better than Canny edge. For Canny and Sobel, it really depends. Both are gradient-based methods, Canny is built in with NMS, so it is better than Sobel. If apply NMS with Sobel like the procedure in (a), the contour or shape is clearer than Canny one. Sobel gives better precision, while Canny gives better recall.

For (d):

1)

GroundTruth	Precision	Recall	F-measure
Farm_GT1	0.4417	0.6587	0.5288
Farm_GT2	0.7240	0.7353	0.7296
Farm_GT3	0.8045	0.7374	0.7695
Farm_GT4	0.6456	0.8029	0.7157
Farm_GT5	0.7229	0.7666	0.7441
Cougar_GT1	0.5266	0.5697	0.5473
Cougar _GT2	0.4527	0.6318	0.5274
Cougar _GT3	0.2581	0.4704	0.3333
Cougar _GT4	0.5003	0.6400	0.5616
Cougar _GT5	0.4128	0.5940	0.4871

Structure Edge has highest F-measure, and Canny has lower F-measure, and Sobel's is smallest.

2)

Farm image should have a higher F measure. Based on the image, farm has less distraction than cougar image. The farm image has clear and sharp boundary visually. On the other words, cougar image has many textures, and the background and foreground are similar to the cougar that the cougar blends in the nature. These parts will influence the performance of prediction.

According to the result above, cougar image indeed has lower F measure than one got from farm image.

3)

For predication of test data from training data, the prediction result mush fall into those 4 classes, precision and recall cannot solely decide if the result is good enough. In large data set, precision and recall have a balance or tradeoff. So F measure makes sure the criteria of decision is not too tight or too loose. If precision is significantly high, then the recall will goes to zero, so from the equation, F measure will be very low. Similarly, if recall is much higher, then precision will be low.

$$F = 2 \cdot \frac{P \cdot R}{P + R}$$

If P+R is constant, then P=c-R, F=2*R(c-R)=2cR-2R^2;

take derivative of F, F'=2c-4R; set F'=2c-4R=0, R=c/2;

Thus, R=P.