EE569 HW4

Problem 1: Texture Analysis

Motivation

Texture are certain pattern on different objects. By analyzing textures, we can easily distinguish one object from another. Moreover, by comparing with existing texture dataset, we are very likely to recognize what type of object it is. In this part, we use Law filter to construct texture feature vector, and then try to build classifiers.

Procedure

a). Texture Classification – Feature Extraction

1. Use the 5 kernels to generate 25 Law filters, and extract the reponse vector in every train image. To let the filters works, I do symmetric padding.

Table 1: 1D Kernel for 5x5 Laws Filters

Name	Kernel
L5 (Level)	[1 4 6 4 1]
E5 (Edge)	[-1 -2 0 2 1]
S5 (Spot)	[-1 0 2 0 -1]
W5 (Wave)	[-1 2 0 -2 1]
R5 (Ripple)	[1 -4 6 -4 1]

2. Computer energy feature. Each image will have a 25-D feature vector. Compute and decide which feature has the strongest discriminant power, which has the weakest.
3. Form a 36*25 matrix with all these feature vectors. Use PCA to reduce to 3-D, and plot. Use nearest neighbor method to classify the test data.

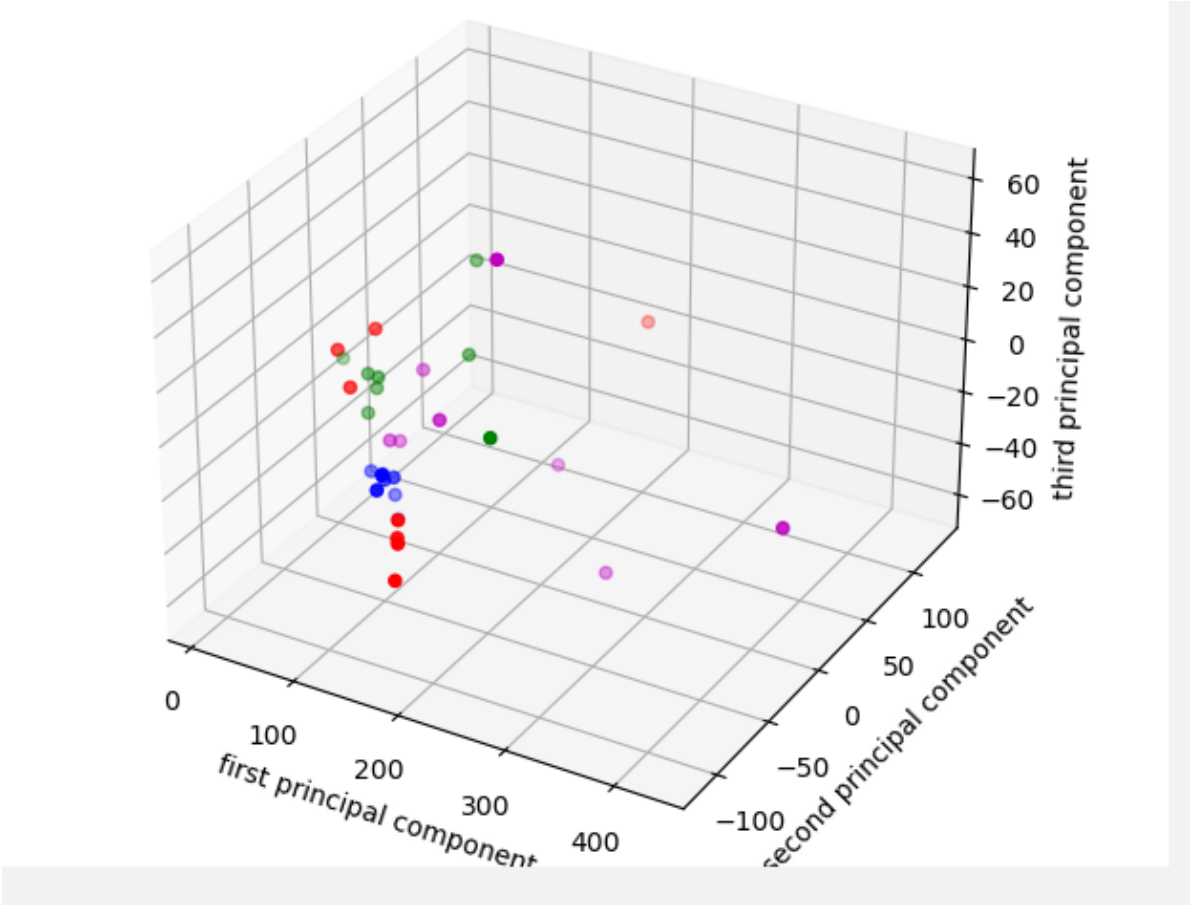
b). Advanced Texture Classification --- Classifier Exploration

1. Use Kmeans on the 3D and 25D test images feature vectors to realize unsupervised classifier.
2. Train random forest and support vector machine with 3D train image feature vectors and apply to test data to realize supervised classifier.

Result

The discriminant power of the 25 features are 18.9889, 0.955592, 3.45098, 7.66155, 4.3046, 2.27234, 1.36839, 2.69716, 8.22138, 5.31794, 27.4044, 2.84191, 11.3691, 12.2852, 11.1008, 21.5647, 6.30918, 13.3234, 11.3923, 10.9031, 2.93228, 2.87744, 5.59401, 8.17438, 4.45562

The strongest discrimant power is thus L5S5, the weakest is L5E5



The Mahalanobis distance is computed with

$$d_m^2(x,y) = (x-y)^T \Sigma^{-1} (x-y)$$

where x is a test point, y is a train point, and Σ is the covariance matrix of the train data.

The nearest neighbour classify result is grass

1. grass √
2. blanket √
3. blanket √
4. stones √
5. grass
6. grass √
7. blanket
8. brick
9. stones
10. brick √
11. brick
12. grass √

7 out of 12 are classified correctly.

For Kmean classifier, the 25D cluster result is [0 0 0 3 0 0 2 0 0 0 1 0], purity for four class: 33.3%, 100%, 100%, 100%, error rate: 50%

the 3D cluster result is [0 0 3 2 0 0 1 0 0 0 2 0], purity for four class: 37.5%, 100%, 50%, 100%, error rate: 50%

For random forest classifier, the test results are

1. grass √
2. blanket √
3. blanket √
4. brick
5. stones √
6. grass √
7. blanket
8. brick
9. blanket
10. brick √
11. brick
12. grass √

7 out of 12 are correct

For SVM the test results are

1. grass √
2. blanket √
3. blanket √
4. stones √
5. stones √
6. grass √
7. brick √
8. brick
9. grass
10. brick √
11. stones
12. grass √

9 out of 12 are correct

Discussion

For L5L5 filter, because it's mean is not 0, and we only care about the variance, so we should minus the mean before we square it to compute energy.

PCA step leads to only 1 result change in 12 test data point, while significantly reduces the computation load. So it is worthy when there're too many features.

The random forest has 100% train set accuracy, but the test set accuracy is not that high. And it does not make any changes when I adjust the parameters I think 36 samples are too few for training a random forest. And also it is possible that the model doesn't generalize well.

The SVM works better. By properly adjust the parameter, the performance has obvious improvement. Although it doesn't achieve 100% accuracy on train set, it generalize well to the test set.

Compare these advanced classifier, supervised ones do better than unsupervised. And SVM works better than random forest since it can more flexibly adjust extents of fitting by adjusting the parameters.

Among all these classifier, although non of them reaches a very high accuracy, but I think there're possible excuses. For example, the test point 7, 8, 9, 11, almost every classifier gets wrong result on them. It is very likely that the 36 train data points are too few, and not representative enough, or these test points are not very typical.

Problem2 Texture Segmentation

Motivation

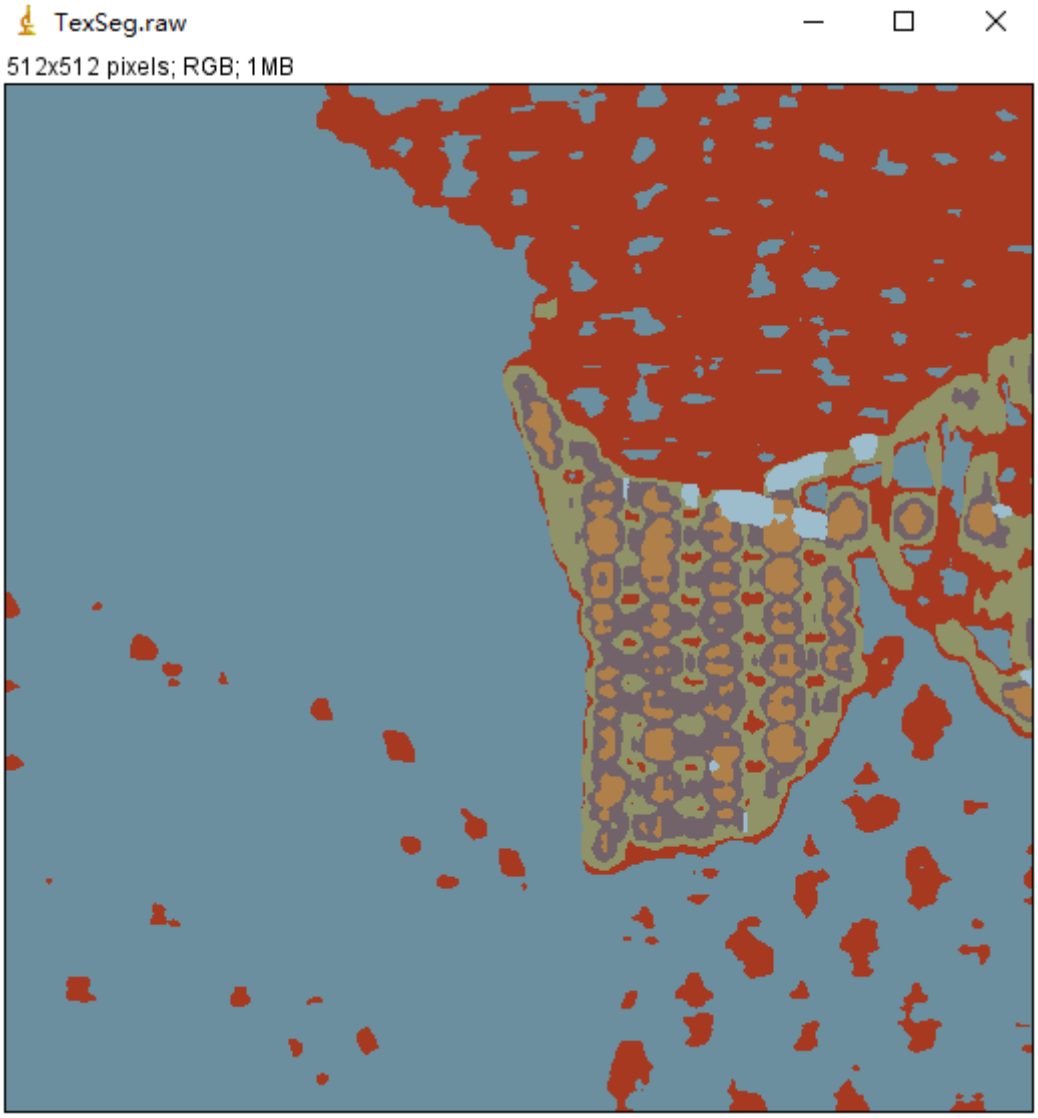
In the previous part we have known how to distinguish texture when there's only one single texture in the picture. However, it's more likely that multiple texture mix in one image, and we need to segment it. The key difference is, now we no longer calculate the total energy for the whole picture, instead, we compute it for every pixel in a local window. The window size is inconstant, which should be determined according to the image pattern. Each pixel now has a feature vector. After that we use Kmeans to cluster those features.

Procedure

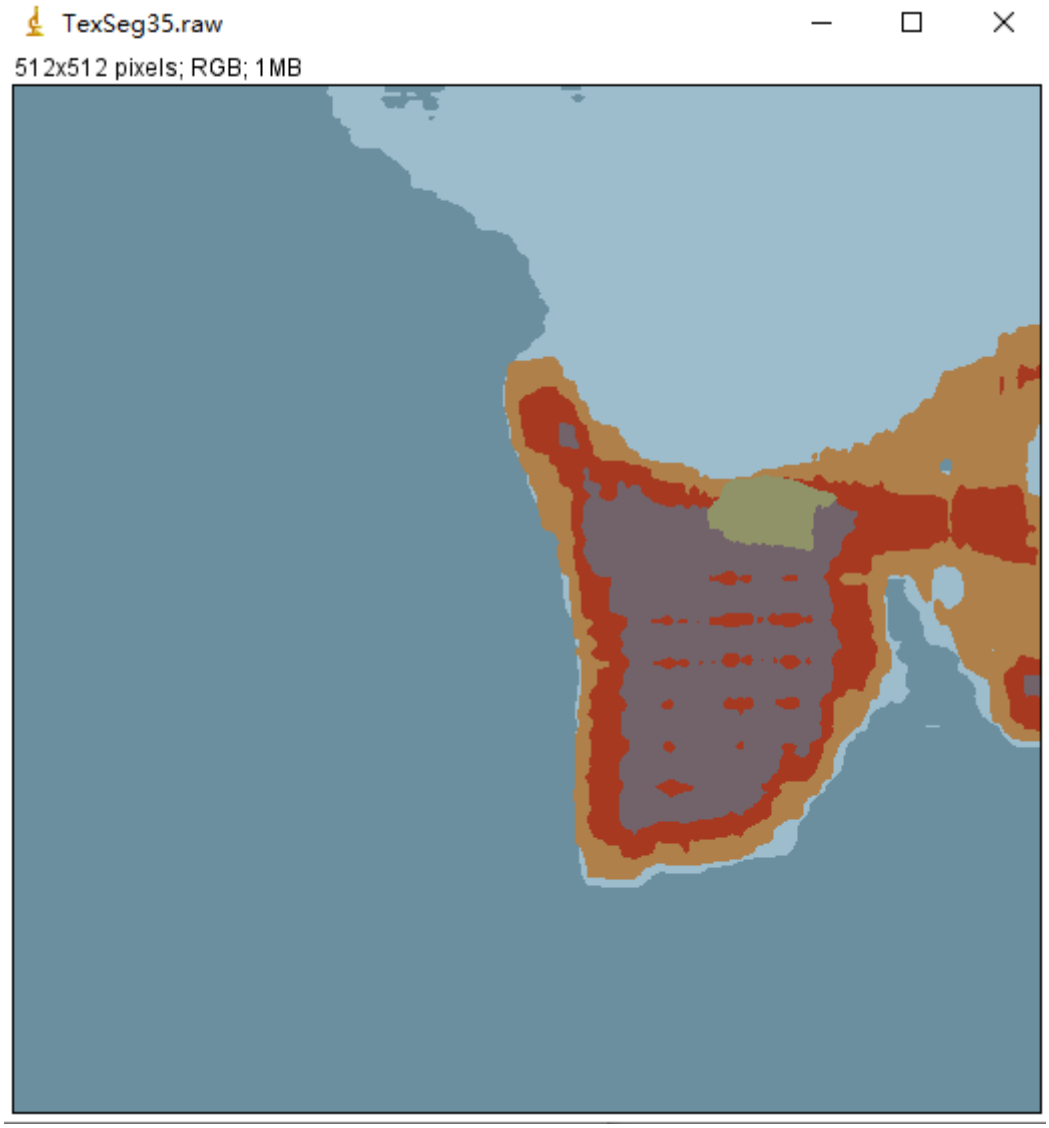
1. Compute filter bank response for every pixel as in Problem 1. L5L5 filter is no longer applied.
2. Calculate the energy feature for every pixel with in a local window. For example, if window size is 15*15, then we sum up all the energy in this range and get a feature vector for this pixel.
3. Use Kmeans to cluster all the pixels. Dye the image to 6 colors according to their class.

Result

Outcome of 15*15 window

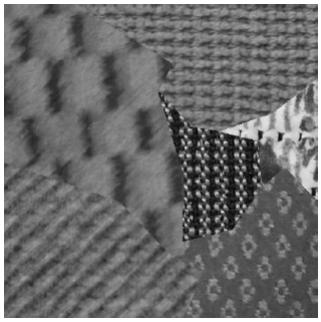


Outcome of 35*35 window



Discussion

The repetitive pattern in the test mosaic image is huge. For example, the upper left corner, I think the minimum repetitive unit size is 80*80, therefore I believe we should use big window. And you can see when the window size is 15 * 15, some flowers and stripes are isolated from textures. With 35*35 windows, those small elements are no longer segmented.



In the lecture slide, it is said the window size is usually 13,15,17. But in our case, our image is big, and the texture pattern is also big, so small window size does not perform very well (as shown above). However, big window size will cost significantly more time to compute, and my computer can only afford 35*35 in reasonable amount of time. So it is a balance.

Problem 3

Motivation

Scale Invariant Feature Transform SIFT is based on keypoint description. It is highly robust to affine distortion, addition of noise, and change in illumination. The main steps are scale-space extrma detection, keypoint localization, orientation assignment, and keypoint descriptor.

Bag of Words use features to represent an image and cluster them. A histogram of frequency of features is generated, and the histogram is used for classification. BoW consist of following steps: detect features, then represent through filter bank responses or SIFT descriptors or other methods, compute the frequency of features and do comparison with other images.

Procedure

Result

Discussion

SIFT is robust to image scale and rotation.

Assigning a consistent orientation to each keypoint can help to invariance to image rotation. Repeated convolution with Gaussian to create different scale space image help to invariance to scale.

The local image descriptor contribute to conquering illumination change and 3D viewpoint. The gradient in a 4*4 region will together generate one keypoint descriptor, which mean one gradient sample can shift up to 4 pixels while still contributing to the same descriptor. That improve 3D robustness. Moreover, contrast change will be canceled by feature vector normalization, and breightness change will not influence the gradient values. So the model is robust to illumination change.

DoG is more efficient to compute, and it provides a close approximation to the scale-normalized LoG.

4x4x8 = 128 element feature vector

OpenCV SIFT is no longer supported. Matlab vlfeat->vl_sift and Computer Vision Toolbox->detectSIFTFeatures just do not work. GREAT!