HW 4

Problem1. Texture Analysis

a) Texture classification

This part is an application of laws filter. Laws filters contain 25 5 by 5 kernels and general 25 response image. After the filter generation, we can use the average on every kernel to represent the image's texture. Then, we can use PCA for dimension reduction and machine learning algorithms to classify the texture.

Algorithm for texture response:

Step1. Initialize filter bank.

Name	Kernel
L5	[1 4 6 4 1]
E5	[-1 -2 0 2 1]
S5	[-1 0 2 0 -1]
W5	[-1 2 0 -2 1]
R5	[1 -4 6 -4 1]

Construct 25 5 by 5 filters by,

$$\boldsymbol{filter_{ab} = kernel_a^T \times kernel_b}$$

Step2. Get the response from filter bank, the total response is $\boldsymbol{Width \times Length \times 25}$

Step3. Calculate the average of every energy response and obtain $\boldsymbol{Width \times Length}$ response

Algorithm for PCA:

Step1. Centralize the data and calculate the Covariance Matrix of the dataset

$$COV = E[(X - \mu)(X - \mu)^T]$$

Step2. Calculate the Eigenvectors and Eigenvalues of COV matrix and sort the eigenvalues and eigenvector descend.

Step3. Select the eigenvectors descend and calculate the new feature.

$$Dataset_{PCA} = Eigenvector \times (x - \mu)$$

Mahalanobis distance is an advanced version of distance. Here we want to calculate the distance between matrix Y ($M \times V$) and vector X ($1 \times V$).

Step1. Calculate the covariance matrix S and mean μ of matrix Y,

Step2. Calculate the Mahalanobis Distance by

$$D_M(X) = \sqrt{(x-\mu)S^{-1}(x-\mu)}$$

Discriminant Power

Discriminant Power is a method to measure the inter-variance and intra-variance ratio. Normally, if the class have lower intra and higher inter, the feature will be easier to classify.

The overall average: $\overline{y}_{..} = \frac{\sum_i \sum_j y_{ij}}{\sum_i \sum_j 1}$

Intra-class average: $\overline{y}_{i.} = \frac{\sum_j y_{ij}}{\sum_j 1}$

Intra-class variance: $\sum_i \sum_j \left(y_{ij} - \overline{y}_{i.}\right)^2$

Inter-class variance: $\sum_i \sum_j \left(\overline{y}_{i.} - \overline{y}_{..}\right)^2$

$$Discriminantpower = \frac{Intra - variance}{Inter - variance}$$

Experiment Results

From the discriminant power experiment, the largest number is 14.5594, which is from $S5^T L5$. The smallest number is 0.4533, which is from $E5^T E5$. The higher value means that intra is much larger than inter and the feature is not as desired. On the contrary, lower value means that the class can be more separable and is desired.

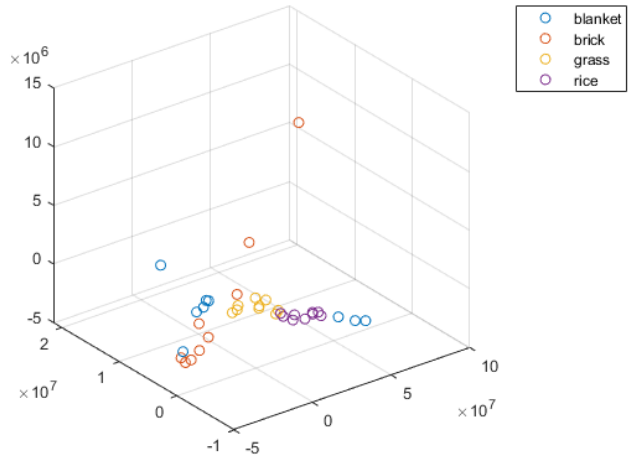


Figure1. PCA texture data visualization

Nearest Neighbor classification prediction	Ground truth
[3 1 1 2 1 1 2 1 4 2 1 1]	[3 1 1 2 4 3 2 4 4 2 1 3]

From the observation, the accuracy of PCA and nearest neighbor classification is 66.7%.

b) Advanced Texture classification

In this part, we implement different classification algorithms, Kmeans, Random Forest and SVM to classify.

Algorithm for Kmeans:

Step1. Determine the cluster number K.

Step2. Initialize K centroids from the data randomly.

Step3. Calculate the distance of datapoints to every centroid and assign label by nearest neighbor.

Step4. Recalculate the centroids by labelled data and repeat **Step3** and **Step4** iteratively.

Algorithms for Random Forest:

Step1. Randomly choose training set from all data.

Step2. Construct Decision Tree model for the selected train set.

Step2. Ensemble the multiple tree models by bagging.

Algorithms for SVM:

Support Vector machine is an algorithm to find the largest margin among the different label data. When introducing kernels such as rbf, polynominal, this algorithm can help classify the linearly inseparable data.

Experiments Results

Owing to unsupervised learning, we cannot compare the predictions label and ground truth label directly.

Table1. Clustering results

Kmeans for 25 features	Kmeans for 3 features	Ground Truth
C1: [1, 6, 12] C2: [2, 7, 10, 11] C3: [3, 5, 8, 9] C4: [4]	C1: [2, 7, 10, 11] C2: [3, 5, 8, 9] C3:[4] C4: [1, 6, 12]	C1: [1,6,12], C2: [2,3,11], C3: [4,7,10], C4: [5,8,9].

From the cluster result, we can find that 25 feature and 3 feature clusters are the same. Besides, there are only three images assigned to the wrong label, which is 4, 2, 11.

For supervised learning, we implemented Random Forest and Support Vector machine (SVM). Here I used build-in function in Matlab. For random forest, I use hyperparameter 50 as the number of trees. For support vector machine, we use the 'rbf' kernel and standardizing the data. From the observations of figure1, the data points cannot be classified linearly, so we used 'rbf' kernel here.

Table2. Supervised learning results

SVM output	Random Forest output	Ground Truth
grass: [1,6,12], blanket: [2,3,10,11], brick: [4,7], rice: [5,8,9].	grass: [1,6,12], blanket: [2,3,10,11], brick: [4,7], rice: [5,8,9].	grass: [1,6,12], blanket: [2,3,11], brick: [4,7,10], rice: [5,8,9].

Finally, we get 11/12 accuracy for both supervised algorithms.

Discussions

For the kmeans, we use 25 features and 3 features respectively. There is no difference of the predictions between the two features. However, reducing the dimension can reduced the computational complexity heavily.

Problem2. Texture Segmentation

a) Basic Texture Segmentation

Texture segmentation is the same as problem1. Here we also use the laws' filter to extract the features. During implement the filters, we didn't substrate global mean here. It is because that different texture may have different global mean and subtract the average mean cannot help to solve the problem.

Algorithm for texture segmentation:

Step1. Extract feature by applying laws filter, here we don't substrate global mean because of various texture.

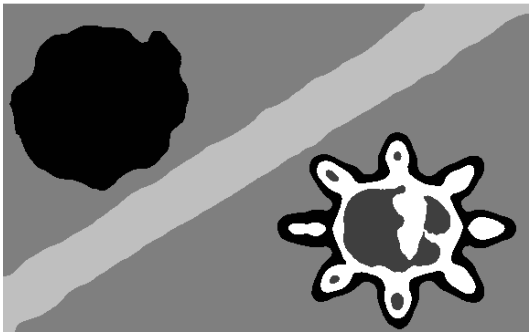
Step2. Calculate the energy by square and get the 25 energy maps with the average with different window type and window size. Here we use the Gaussian window and average window.

Step3. Average pixel-wise features by the feature from $L5^T L5$ and delete that feature.

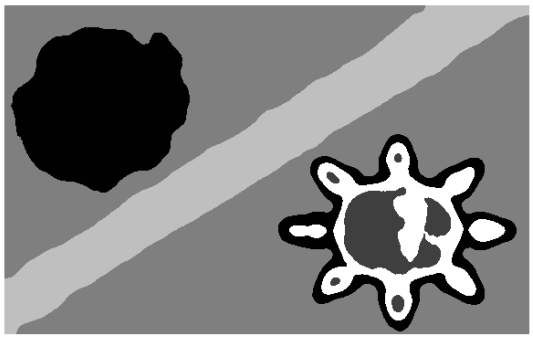
Step4. Use the kmeans algorithm to classify every pixel and assigned new pixel value based on cluster output. Using (0,63,127,191,255) respectively.

Experiments Results

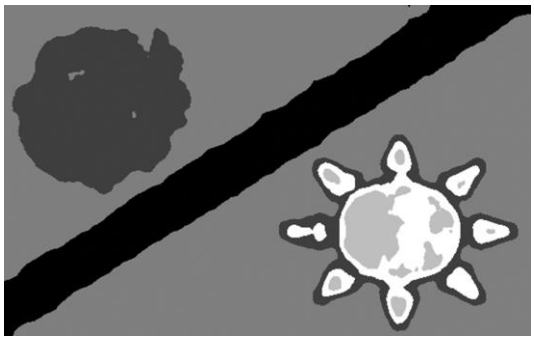
Here we tuned the windows shape for gaussian and average and window size to get the different classification results.



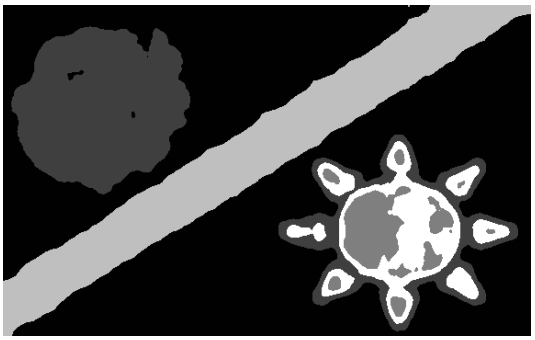
Gaussian 31



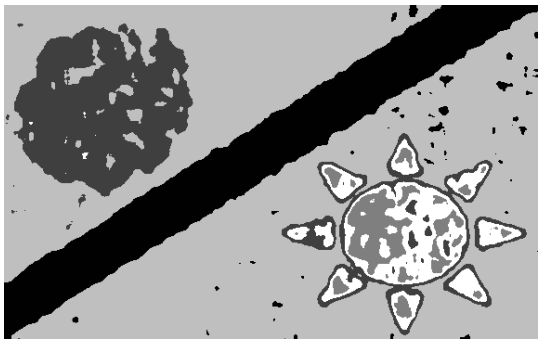
Mean 31



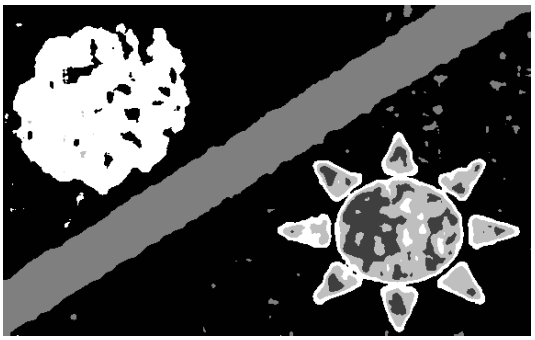
Gaussian 21



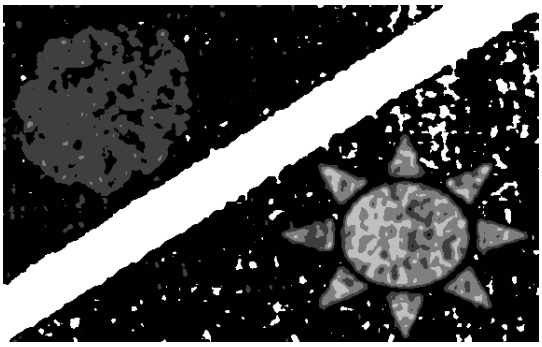
Mean 21



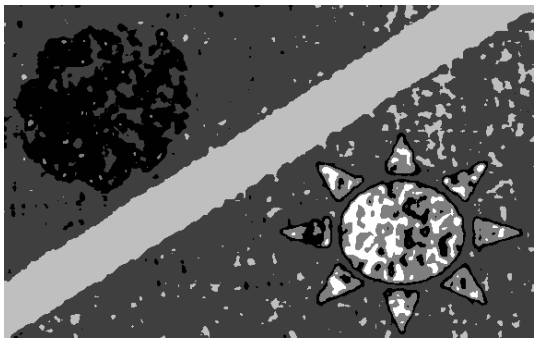
Gaussian 11



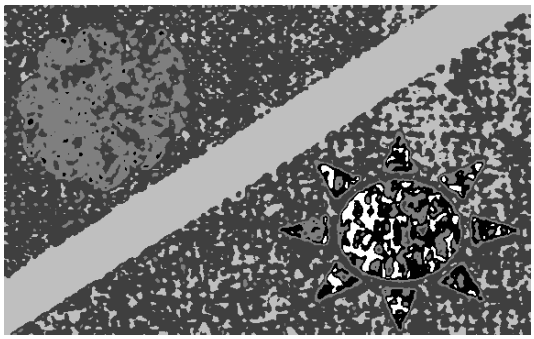
Mean 11



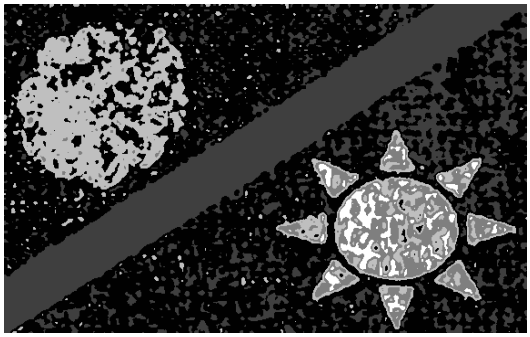
Gaussian 7



Mean 7



Gaussian 5



Mean 5

Figure3. Classification output based on texture

b) Advanced Texture Segmentation

Here, we implement PCA algorithm to detect the features. From the former experiments, it shows that kernel 31 can get more stable results without many defects. From the image with lower kernel, such as 3, the output will preserve more edge information but it will introduce more defects because the energy average process don't calculate in a large scale.

Here, I would like to propose a post-processing method based on kernel to rebuild the texture classification map. From the original image and the output from Problem1 a), we can find that there are two main problems of the output. The first is that there are two regions in the upper and lower diagonal classified as the same region. The second part is the lower diagonal sun like part.

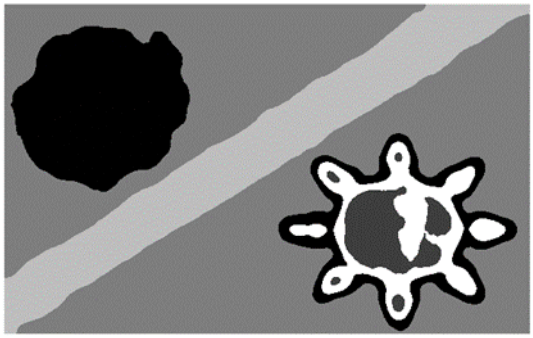


Figure4. Candidate part for advanced processing

To rectify the first misclassified part, I would like to introduce the geometry method to rebuild the upper diagonal method with a new value. To rectify the second misclassified part. I subtract the edge with sobel detector. Then, I dilate the edge of sun like part and rebuild the classification image.

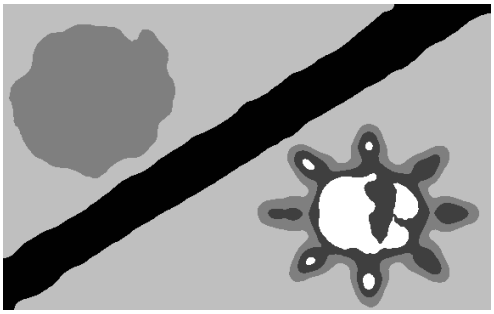
MAKEUP Algorithms

Step1. Find the unused label in the upper part and rebuild the upper diagonal region (out of circle) with one unused label.

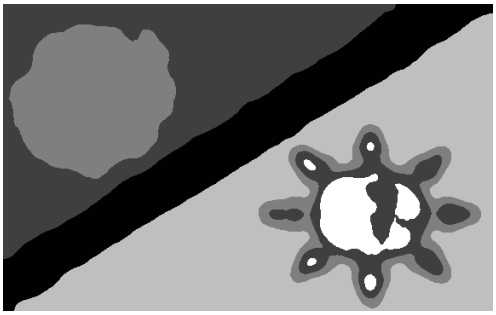
Step2. Extract lower diagonal images' edge and dilate that part of image with ball shape.

Step3. Substitute the dilated map's location in the classification image with the unused label.

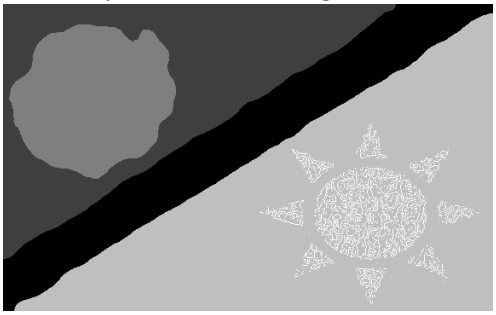
Experiments Results



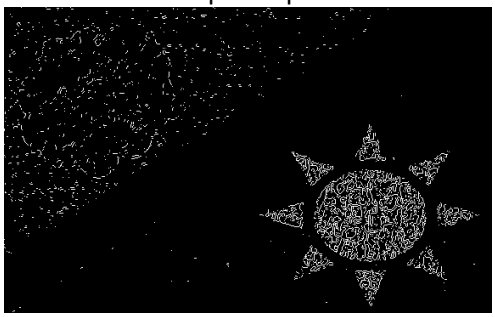
Input of MAKEUP Algorithm



Step1 output



Step2 output



Extracted Edge

Figure5. MAKEUP Algorithm

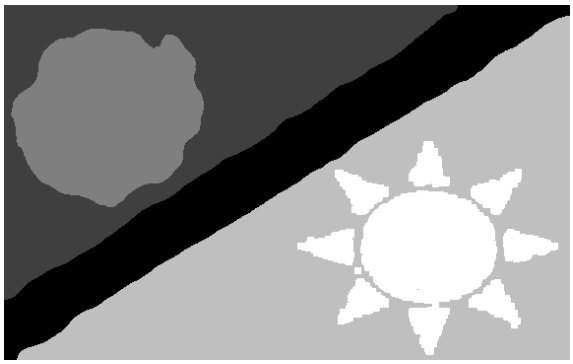
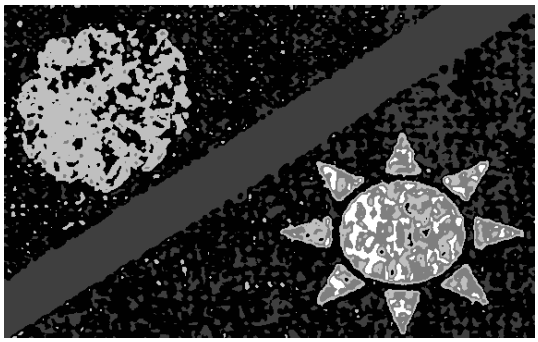


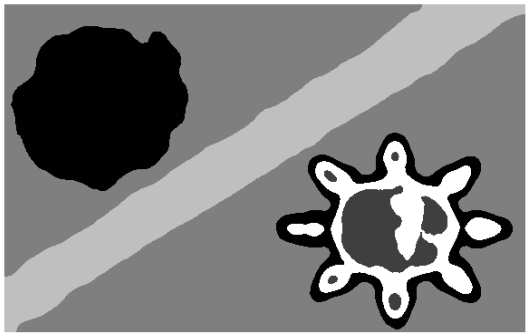
Figure6. Output of advanced classification

Discussions

From tuning the size of average filter. We can find that large kernel may contribute to more unified and blurred image and the small kernel is computation efficiency, but it will introduce defects. Besides, large kernel cannot detect small texture region and the boundary of two different regions, showing in the figure below.



Mean 5



Mean 31

Figure7. Comparison of kernel size

According to the former knowledge, using gaussian kernel can preserved more location information, especially the for the larger kernel size. However, there is no specific difference between the two different windows.

Problem3. SIFT and Image Matching

a) Salient point descriptor
   1) From the abstract, the points are invariant against the image scale and rotation, noise and illuminance.
   2) Scale invariance is achieved by searching for key points across the multiple scaled and find maxima across the different scales.

      Rotation variance is calculated by count the histogram of orientations and present the orientation by high dimension descriptors. Hence, no matter how rotated the images, the distribution of orientation doesn't change and the point can still be detected.

   3) The descriptor is a histogram of image orientations in nearby region. Hence, the brightness of the image will not influence the orientation in the nearby region because it will be normalized to one of total region. During the feature searching, algorithm searches on the region of the difference of image and the difference won't be influenced by the illuminance.
   4) The difference-of-Gaussian is a close approximation to the scale normed LoG. According to previous paper, the extrama of LoG produced more stable features.

5) Descriptor is a 128-dimension vector, which is counted in a 4 by 4 region. Each 4 by 4 region can be counted as an 8 bins for orientation of pixels in that region. Hence, the total will be a $4 \times 4 \times 8$ vector.

b) Image Matching

Here I use VLFeat toolbox in matlab to extract the SIFT features. There are two returns from VLfeat.vl_sift(), [f d]. each key point in f is a 4×1 vector. The first and second elements represent the location of every key point. The third element represents the scale of key point and the last represent the principle orientation. Each key point in d represents by 128×1 vector, which is descriptor.

Algorithm for key point searching
Step1. Extract the key points of two image.
Step2. Find the largest-scale key point in dog3 and calculate the Euclidean distance with descriptors from dog1.
Step3. Find the smallest key points.

Experiment results


Dog3's largest key point


The best match key point

Figure8. key point match

There are too many parameters in vl_sift function. Owing to the two images have huge difference, the key point searching process is bad. However, here are some parameters in the vl_sift function. I can get more reasonable result after selecting hyperparameter 'PeakThresh' = 45, 'EdgeThresh' = 20, 'Levels' = 1.

Dog3's largest key point

The best match key point

Figure8. key point match based on tuned parameters

The orientations of key points show in the image as the radius of the red circle. It is selected from the most frequent orientations based on the nearby region.

For the section part, we directly use the vlfeat build-in function vl_ubcmatch to match key points for the two images and use vl_plotframe to show the selected key point.

Experiment results

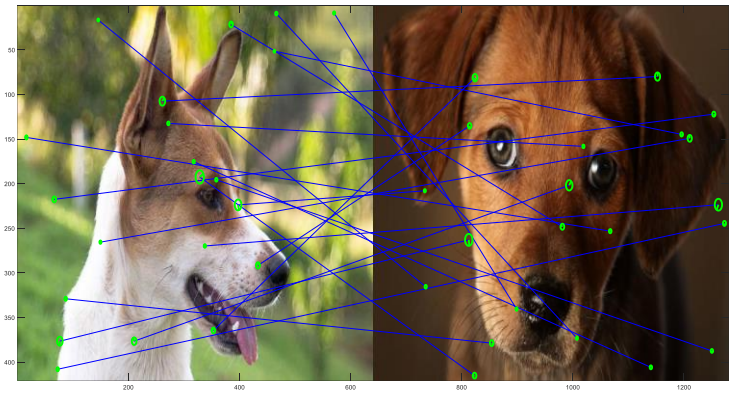


Figure9. matched key points between Dog2 and Dog3

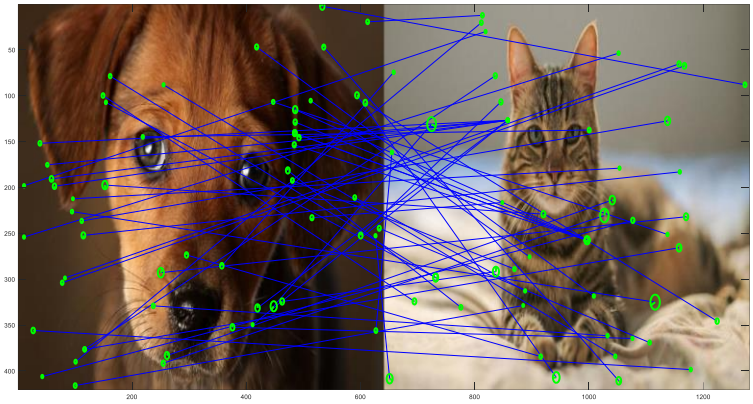


Figure10. matched key points between Dog3 and Cat

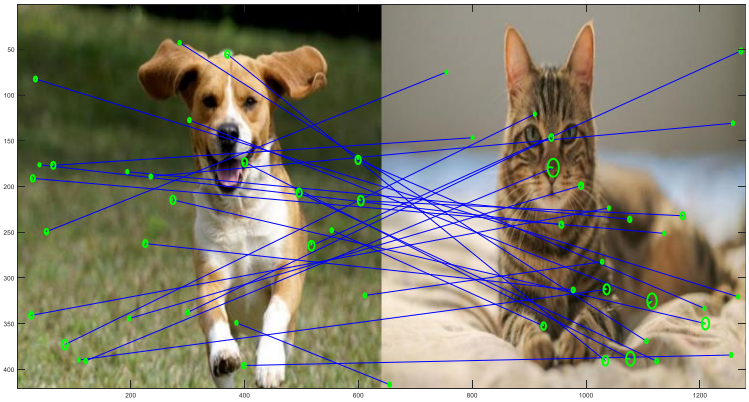


Figure11. matched key points between Dog1 and Cat

Discussion

From the above matching images, we can find that SIFT can not work well in these scenarios. There are two kind of miss matching. The first is that image pairs are unrelated. The second is that one key point match with too many points. The main reason that the two images are totally different. We can find that Dog2 and Dog3 have different scale-size and direction of dogs. The Dog2 and Cat are different animals and different scale.

c) Bag of words

Bag of words is the method to classify image by different image patches based on SIFT features.

Algorithms for Bag of Words

Step1. Find and calculate the SIFT features of all images.
Step2. Implement PCA to reduce the dimension of 128-dim descriptor to 20 dim features **separately**.
Step3. Implement kmeans to classify all key points to eight class
Step4. Calculate the histogram of four images based on eight class.

Experiment Results

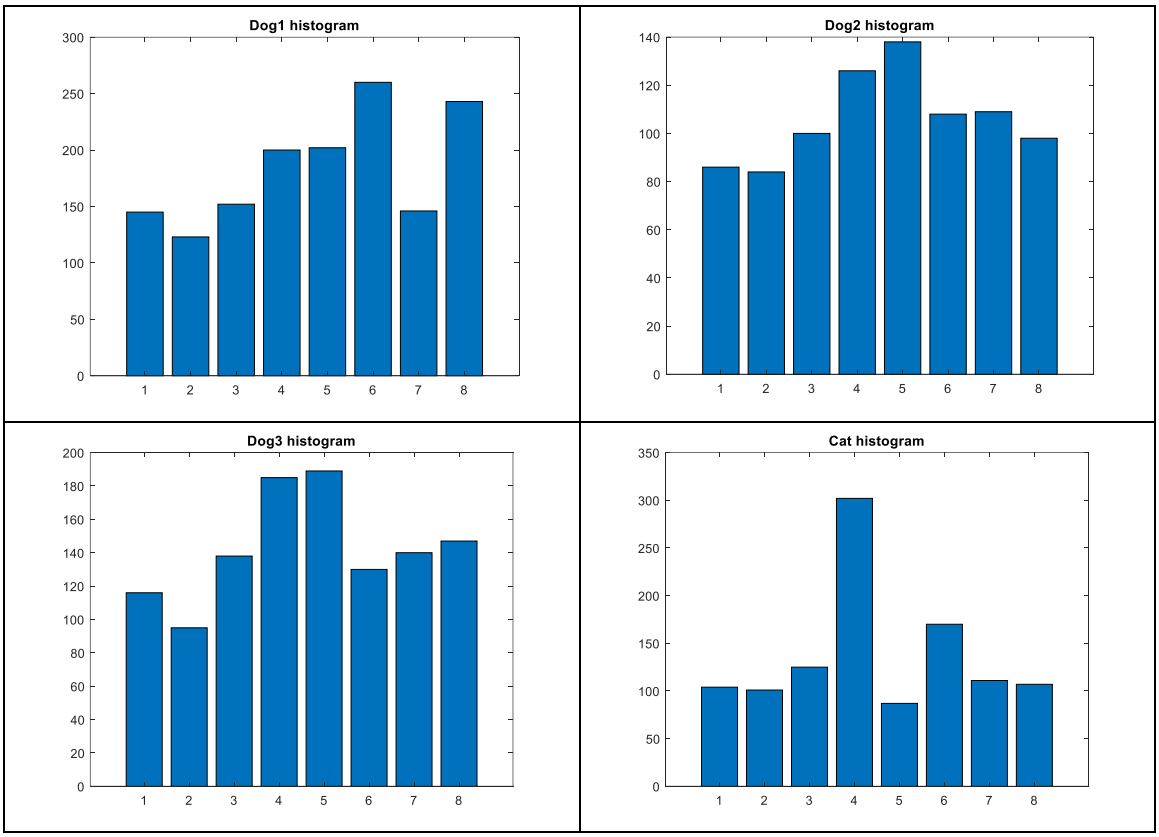


Figure12. Histogram of four images based on bag of words

Table3. The similarity between image pairs

Dog3	1 Trail	2 Trail	3 Trail
Dog1	0.7750	0.7750	0.7678
Dog2	0.7447	0.7386	0.7447
Dog3	0.7245	0.7232	0.8005

Discussion

From the result above, we can find that the similarity between different images are around 0.7-0.8. However, every experiment will introduce the different results. There randomness may be introduced by kmeans. As we know, kmeans can get different results based on the initial centroid selection. Hence the results are reasonable.