# EE 554 Computer Vision ‖

# Project 1

CHONGJIA YIN

MINGJIE WU

# Content

# Overview

## ➢ Abstract

This is an object recognition project based on bag of words. General idea applied in this project is based on vector quantization of affine invariant descriptors of image patches. The naïve Bayes classifier is used here to classify classes. For experiment, we use three different class to test out model. And the data set is download on website: http://www.robots.ox.ac.uk/~vgg/data/data-cats.html. The experiment result is fairly good that we can say that we can distinguish the difference within classes and produce fair categorization accuracy even without exploiting geometric information.

## ➢ Introduction

Given an appropriate categorization of image contents, the task of accurate classification is always important. Thus, a general problem of visual categorization is in front of us. Also, the process here should also can handle the variations in view, imaging, lighting and occlusion because in real world, the images taken can varies among these aspects on the same kind of objects. The task-dependent and evolving nature of visual categories motivates an example based machine learning approach. This paper presents a *bag of words* approach to visual categorization. A *bag of words* corresponds to a histogram of the number of occurrences of particular image patterns in a given image. The main advantages of the method are its simplicity, its computational efficiency and its invariance to affine transformations, as well as occlusion, lighting and intra-class variations.

The *bag of words* approach is applied on text categorization first and then it is evolved to the case of image categorization. In the case of images categorization, the *bag of words* approach uses clustering to obtain quite high-dimensional feature vectors for a classifier but not a relatively low-dimensional histograms in text categorization. The method proposed below is based on SIFT descriptor and its application. We'll talk a little more about SIFT descriptor in the following content.

# Outline

Main steps of method are following:

- Feature extraction of image patches
- Clustering features with a vector quantization algorithms
- Constructing *bag of words* model
- Classifying images with multiclass classifier

Due to the variation of environment in which images are taken, the descriptors extracted should be invariant to variations that are irrelevant to the classification task. Or to say the descriptors should be invariant to images transformation, lighting variation or even occlusions. Here are some details about each step.

## ➢ Feature extraction of image patches

In computer vision area, local descriptors have been proved well enough to matching and recognition task. This means if there is a transformation between two instances of an object, corresponding points are well detected and hopefully, identical descriptor values are obtained for each instance. In this object recognition case, this is rather important because the various instance as well as shape of same kind of object. Robust descriptors should be applied here to deal with this situation.

✧ Scale Invariant Feature Transform (SIFT) descriptors

SIFT descriptors are multi-image representations of an image neighborhood. They are Gaussian derivatives computed at 8 orientation planes over a 4x4 grid of spatial locations, giving a 128-dimension vector. Given SIFT's ability to find distinctive keypoints that are invariant to location, scale and rotation, and robust to affine transformations (changes in scale, rotation, shear, and position) and changes in illumination, they are usable for object recognition.

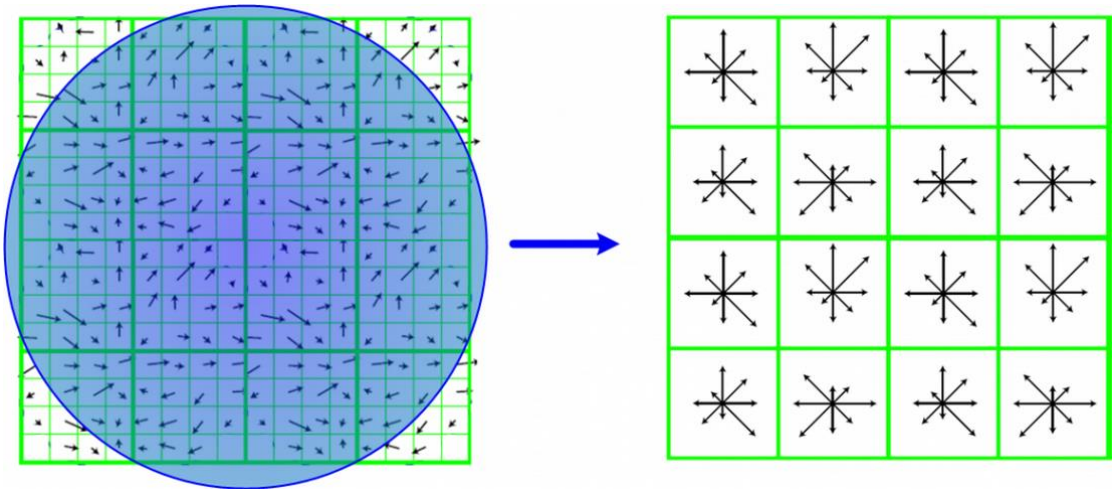

Figure 1. How SIFT descriptor is constructed

Figure 1 shows some brief idea of how SIFT descriptor is constructed. First a set of orientation histograms is created on 4x4 pixel neighborhoods with 8 bins each. These histograms are computed from magnitude and orientation values of samples in a 16 x 16 region around the keypoint such that each histogram contains samples from a 4 x 4 sub-region of the original neighborhood region. The magnitudes are further weighted by a Gaussian function with \sigma equal to one half the width of the descriptor window. The descriptor then becomes a vector of all the values of these histograms. Since there are 4 x 4 = 16 histograms each with 8 bins the vector has 128 elements. This vector is then normalized to unit length in order to enhance invariance to affine changes in illumination.

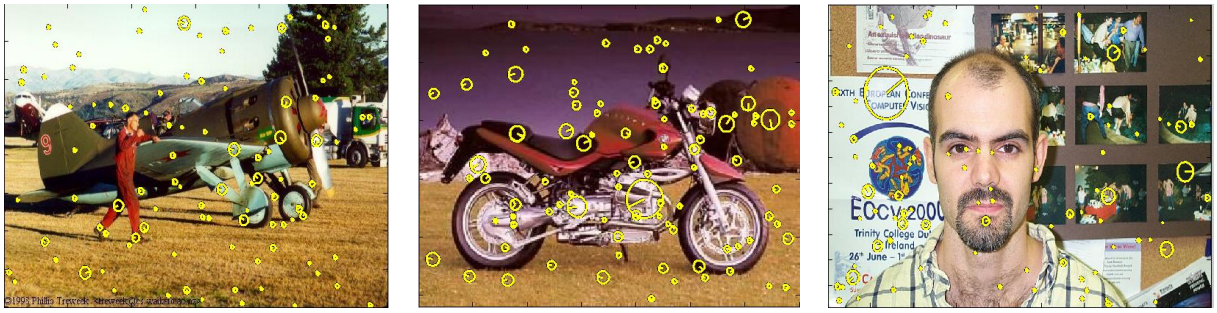

Figure 2. Examples of descriptors extracted in images

Figure 2 shows some examples of descriptors extracted in images selected from dataset used in this project. 3 images are from three differently classes, namely airplanes, motorbikes and faces. Some of the SIFT descriptors are chosen to show in the images above. Actually, SIFT descriptors are pretty dense descriptors. We prefer SIFT descriptors to alternatives such as steered Gaussian derivatives or differential invariants of the local jet for the following reasons:

1. simple than linear Gaussian derivatives;
2. simple Euclidean metric in the feature space seems justified;
3. 128-dimension is more discriminative.

In our case, we have the labels information of the train images. for each class, we extracted the SIFT descriptors of each image by using 'vl_sift' function. Finally, the 'vocabulary' is built by combining all the descriptors together.

For different size images, the number of total SIFT descriptors extracted are different. Usually, the numbers are proportional to the size of images. For bigger size image, there are always more descriptors. For rather small image, there are pretty few descriptors. So SIFT descriptors are not so precise in some small size image cases. And in our project, we also tried some images classes with small size, it turned out that the classification result is not good. We'll talk more about this in the experiment part. And later we'll talk a little more about the dataset we used.

## ➢ Clustering features with a vector quantization algorithms

We applied the simplest square-error partitioning method: k-means here to do the clustering job. This algorithm proceeds by iterated assignments of points to their closest cluster centers and re-computation of the cluster centers. Two difficulties are that the k-means algorithm converges only to local optima of the squared distortion, and that it does not determine the parameter k. we already learned much about k-mean, so I'll skip more details about this approach.

The function 'vl_kmens' is used here to cluster all the descriptors. Totally, there are over 400,000 descriptors which are extracted from the previous part. The value 'k' is not determined. And apparently, the k should be big enough to have the discriminative power. However, bigger k, huger computation complexity. We tried several k to balance performance of the model and time consuming.

After clustering the 'vocabulary', several centers as well as the assignment of the centers are decided. We choose to use *maximum a posterior* (MAP) to learn the necessary parameters of the model. The parameter of each class can be compute as:

$$u_k^{MAP} = \frac{N_k + a_k - 1}{\sum_{m=1}^{K}(N_m + a_m - 1)}$$

Where $a_m$ is decided by the Dirichlet smoothing parameter. We choose the Dirichlet smoothing parameter as 2000 here.

## ➢ Classifying images with multiclass classifier

Once descriptors have been assigned to clusters to form feature vectors, we reduce the problem of generic visual categorization to that of multi-class supervised learning, with as many classes as defined visual categories. The categorizer performs two separate steps in order to predict the classes of unlabeled images: training and testing. During training, labeled data is sent to the classifier and used to adapt a statistical decision procedure for distinguishing categories. Among many available classifiers, we applied the Naïve Bayes classifier for its simplicity and its speed.

To mention, for every SIFT descriptor extracted in test data set, it should be assigned to a proper class. The nearest neighbor is used here to do the assignment job. The Euclidean distance between a special descriptor and k centers are computed, and the descriptor will be assigned to the nearest cluster. We call the function 'leastdistance' to assign descriptors to certain cluster.

The classification score can be compute using

$$socre(Q, I) = c\ P(Q|u_1, \dots, u_k) = u_1^{q_1}\ u_2^{q_2} \dots\ u_k^{q_k}$$

Where c is only a constant, when maximizing the score, the constant c can be ignored. With the extracted descriptors of image, the distribution of these descriptors are computed in the form of $q_1, q_2, \ldots, q_k$. And then with all the parameters calculated, the matching score can be obtained. The category which maximizes the matching score will be the assigned category of the test image. For each category, we also compute the classification accuracy of this class.

# Experiment

## ➢ Dataset

The image data is download from the database of University of Oxford. The website is http://www.robots.ox.ac.uk/~vgg/data/, under the category of 'Object Categories'.

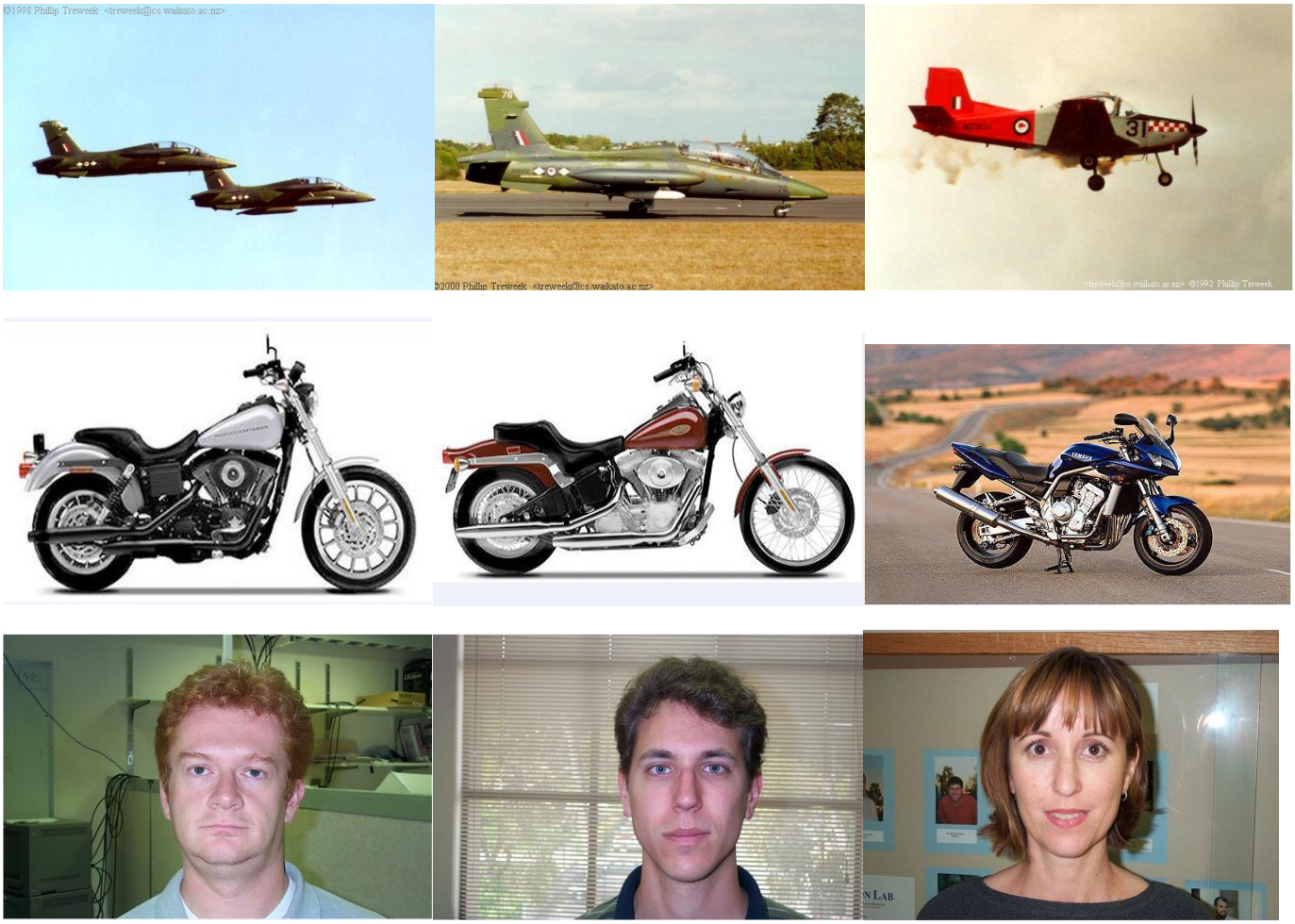

Figure 3. Examples of 3 classes in the dataset. Airplane, motorbikes, and faces.

Figure 3 shows some images in the dataset we used. Namely, airplanes, motorbikes, faces. The shape of the same objects really varies. Let alone the outside environment. So object recognition is not a simple problem. During our experiment, we tied many classes, and one interesting thing is that image class with smaller pixel size always get worse performance. We discussed and reach the conclusion that this is because of the characteristic of SIFT descriptors. For larger pictures, SIFT always get more descriptors. So when you put

different size pictures together, especially when picture size of a certain class is much smaller, then the total descriptors of this class will be much smaller and thus this class is hard to distinguish. This is a drawback of our method. The 3 classes we choose here is with the around picture size so we can obtain a better performance.

Also, the code running process really costs much time. For a 1,000 images data set, it takes about 15 minutes to go. This is also why we only choose 3 class to test. The feature extracting of each picture costs main part of the time. But the retribution robustness of SIFT descriptors is also neat.

## ➢ Observation

In this case we choose three categories：Airplanes, motorbikes and faces. During experiment, slightly tuned parameters and observe how the classification accuracy changed.

✧ Number of clusters.

We set different k numbers in k-means and compare the difference in accuracy outcomes. The number of test image is 100. Results are shown in Table 1.

Table 1 Recognition accuracy with respect to different k values (length of training = 300)

| | Airplane | Motorbike | Face | Average |
|---|---|---|---|---|
| K=200 | 70 | 44 | 83 | 65.67 |
| K=300 | 72 | 47 | 83 | 67.33 |
| K=350 | 75 | 48 | 87 | 70 |
| K=400 | 75 | 50 | 83 | 69.33 |

✧ Length of training set.

Although it is generally acknowledged that more training data involved, more accurate the classification will be, we still want to figure out to what extent the number of training data can affect the experiment result. So we change the number of training images and observe.

Table 2 Recognition accuracy with respect to different training length. (K=300)

| | Airplane | Motorbike | Face | Average |
|---|---|---|---|---|
| Train=200 | 61 | 33 | 86 | 60 |
| Train =250 | 62 | 48 | 75 | 61.67 |
| Train=300 | 72 | 47 | 83 | 67.33 |
| Train=350 | 64 | 31 | 97 | 64 |

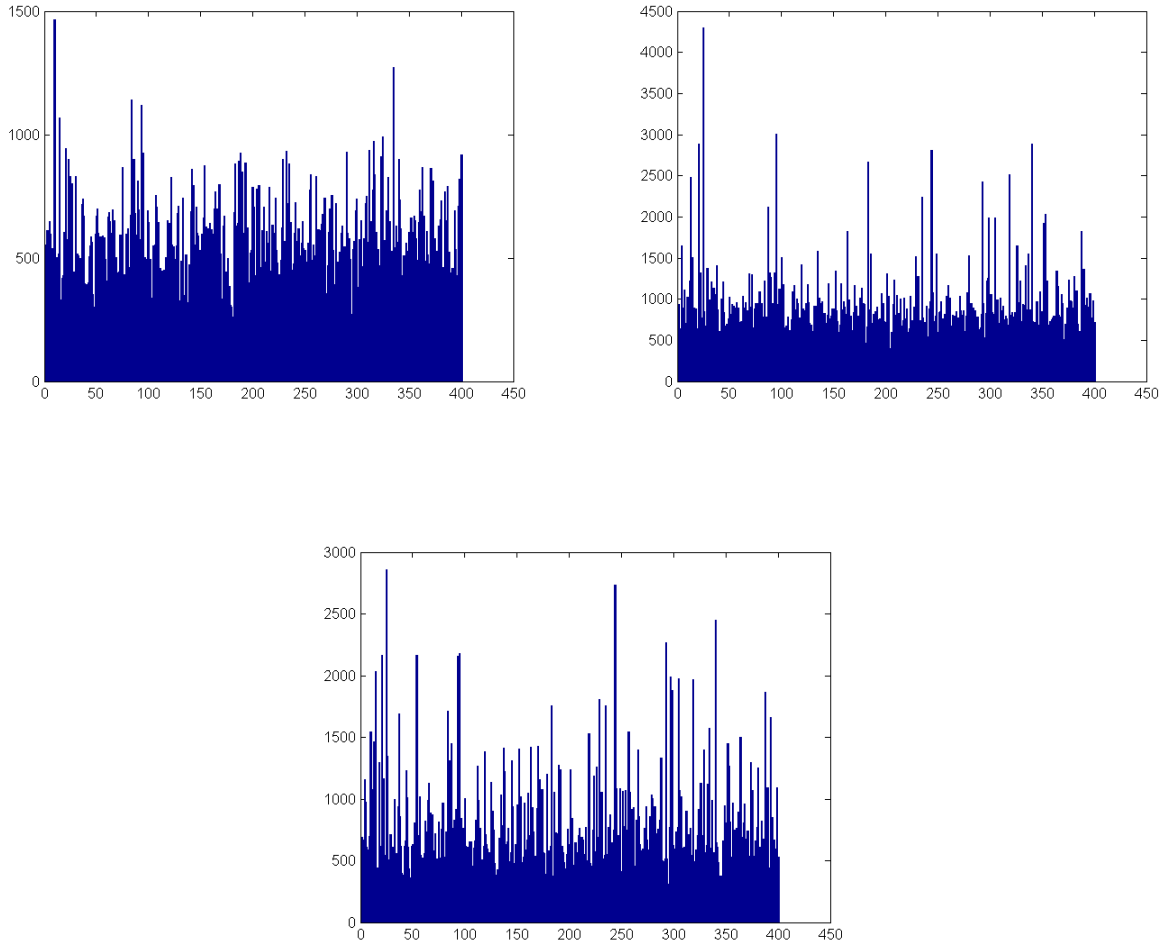


**Figure 3** Categorical distribution against visual words when k=400 for three categories. From top to bottom, left to right: airplane, motorbike, face.

The best performance achieved when training length equals 300 and k equals 350. And as is shown above, it is not true that the more training data involved， the higher accuracy it will be.

## Conclusion

In this project, we accomplished the 3-category object classification problem with supervised learning. First, we extract SIFT descriptors from all training images. In the pool of all descriptors, we use K-means to cluster all the descriptors to form k visual words. By tuning k to different numbers, the outcomes would differ.

The K-means algorithm assigns each descriptor to one specific visual word, and by that we can generate the categorical distribution against k visual words for different category. Then we learn the parameter about that categorical distribution with MAP method, which outperformed the MLE method (in case that some visual words may have no descriptors belonging to). After we got the visual word distribution of different categories, we can compare them with a new incoming test image data. As what was did before, we extract the SIFT descriptors of the new image and assigned them to the nearest visual words (the centers of clusters) by computing the Euclidean distance. Generate the new distribution in the same way and learn the new parameter for that. Then we can compute the similarity score for the query image and classify it to one of three categories who has the largest score value.

To make our outcome more detailed and convincing, we separately compute the classification accuracy for each categories and found out the outcome vary a lot with respect to categories. We assume that the reason may be

the descriptors of some categories can be easier to be extracted so they may have some slight advantages in dominating the cluster result. This can be verified from the mediate MATLAB results: the number of descriptors found in each category is proportional to the accuracy of it. Besides, the average size of images in each category may also affect the result since bigger sized images may include more descriptors of that kind. One way to improve it is to adjust the threshold when extracting SIFT descriptors to accommodate to different situation. This will be studied in our future works.

As for the contribution part, we are a small group with 2 members. So, we contribute to this project about equally. Chongjia wrote the feature extraction part as well as the clustering part of the code. Mingjie did the parameter computing and the classifying part. And the report is wrote by both of us.