Bag of Visual Words

Egor Badich, Zhengtao Wang, Gabrielle Kuntz, Suyang Yan, Chujun Qi
Department of Electrical and Computer Engineering
Boston University

Abstract

Bag of words is one of the most commonly used methods for categorizing objects. The main idea behind the bag of words is to generate visual words using K-means. Then extract key points into one of the visual words and construct a histogram based on the occurrences of the words. This report shows the whole process of image classification based on the bag of words model. All of the steps are implemented by matlab.

Introduction

In recent years, the bag of words model shows its power in image classification and text recognition. It includes different detectors and descriptors so that users can choose different ways to extract features from images. The basic idea of a bag of visual words model is to detect features from images and quantize them into “visual” words. Each part of the image that contains a feature would be assigned a vector to generate a description matrix. After generating the visual words, use histogram to count the number of visual words for each image and pick the word that appears the most frequently, which means the bag of words model will pick the word and assign it as the single vector for the image.

The process we go through to build our bag of visual words model is to build a feature extraction method, one k means algorithm, and a multiclass SVM classifier. The purpose of the feature extraction algorithm is to build visual words for each part of the input image. After getting all the visual words of one image, use k means algorithm to cluster different words. For instance, one image may contain different types of features, so they would obtain different descriptions. Thus, we need k means to separate them and then use histogram to find the exact word for the target image. After getting all the visual words of the training set and image set, use multiclass SVM to train the classifier and then match each image in the test set to a word.

Pre-process

Before starting feature extraction and training data, organizing and separating the dataset should be done. Based on the dataset we used, each image has a roughly size of 300 x 200 pixels, and each category has at least 40 images. For our testing, we choose to classify three different categories, cup, camera, and airplanes. Three categories contain different numbers of images, so we need to find the smallest category and use the image number to create our test set and training set. Based on this condition, we choose to train 35 images of each category and test them using 15 images of each. We combine all the categories into one image set, and then divide the training set and test set with the proportion of 7:3. Moreover, in order to send images into the SIFT algorithm, we need to transfer images to RGB type(M*N*3) and then to grayscale for feature detection.

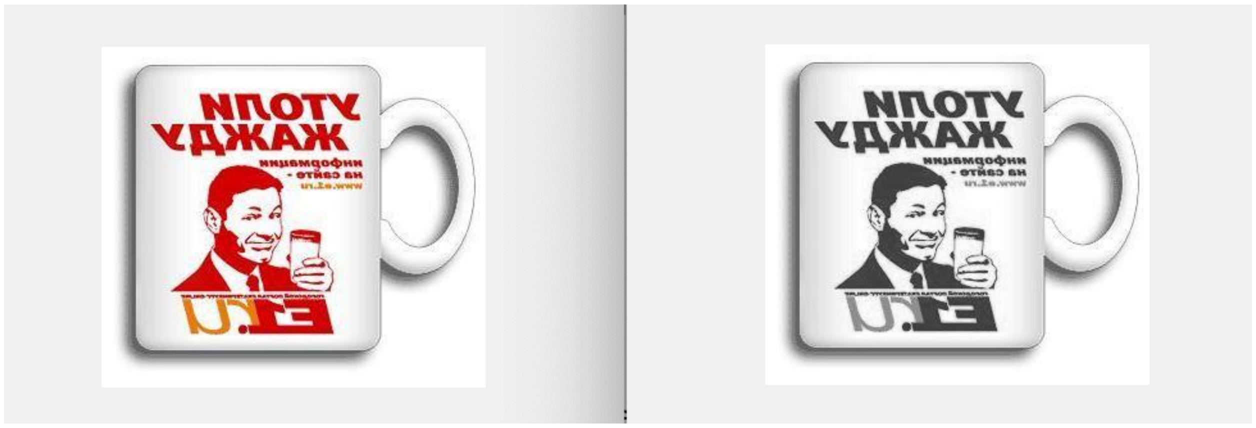


Figure 1: Use rgb2gray() method to transfer an RGB image to grayscale.

SIFT Feature Extraction

In this practice, we use the SIFT(Scale-Invariant Feature Transform) algorithm to detect and describe local features in images. It locates key points in the image (detector) and gives them quantitative information (descriptors). SIFT features not only have scale invariance, even if the rotation angle, image brightness or shooting angle of view are changed, a good detection effect can still be obtained. So it’s often used in image feature extraction.

Building scale space is an initialization operation, the purpose of the scale space theory is to simulate the multi-scale characteristics of image data. Because the Gaussian convolution kernel is the only linear kernel that implements scaling, we apply Gaussian Convolution to blur the obtained grayscale images to build the scale space. Convolutions are made for each pictures with increasing standard deviation. After convolution, one of the pictures would be chosen to be down-sampled and start convolution from that. Processing until the picture is too small to proceed, we will get multiple layers of the image which are called octaves since the sampling rate is decreased by a factor of two per stage.

Now, every octave is a continuous space with three dimensions, the coordinate x,y of the pixels and the standard deviation. Then we will use the Difference of Gaussian(DoG) to enhance the features which would be better for us to find the extreme gray value. The input image through DoG will be smoothed with a Gaussian kernel and sampled.[1]

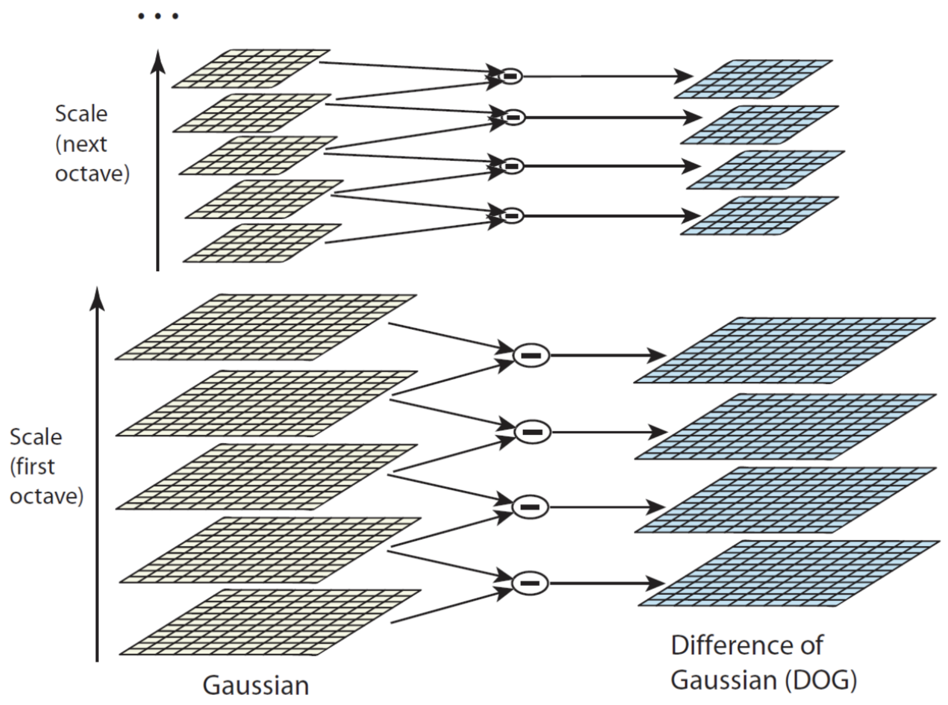


Figure 2: The process of DoG[2]

Once the final DoG is produced, all images in the DoG will be searched by local extrema. In order to detect the local maxima and minima after the process, each example point is compared with its eight neighbors in the current level and every nine neighbors of its upper scale and lower scale. Points with small absolute values will be discarded to prevent noises. Now, we get plenty of key points.

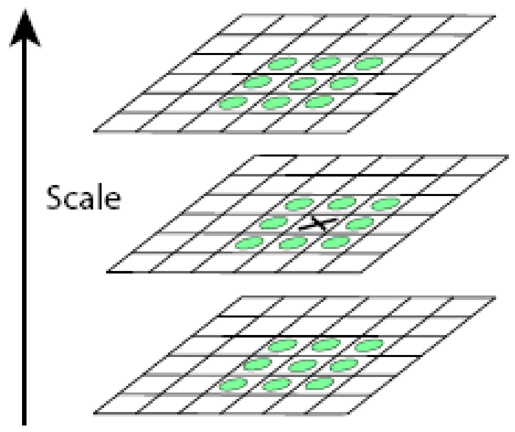


Figure 3: Detect extrema of DoG images[2]

Next, Taylor Expansion of spatial scale will be implemented to remove the points with low contrast. Due to the sensitivity of edges when processed by DoG, we also need to apply Hessian Matrix to eliminate the edge responses. Finally we get the needed accurate key points.

$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^{\mathbf{T}} \frac{\partial^2 D}{\partial \mathbf{x}^2} \mathbf{x}$$

Figure4. Taylor Expansion of Spatial Scale

$$\mathbf{H} = \left[\begin{array}{cc} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{array} \right]$$

Figure5. Hessian Matrix

The next step is orientation assignment. We should calculate the gradient of sample points within a region around the keypoints. Then divide the 360 degrees into 36 small grids every 10 degrees. Count the number of key points that's in each direction grid, and generate a histogram of degree distribution. The peak degree will be selected and assigned to the orientation of the key point.

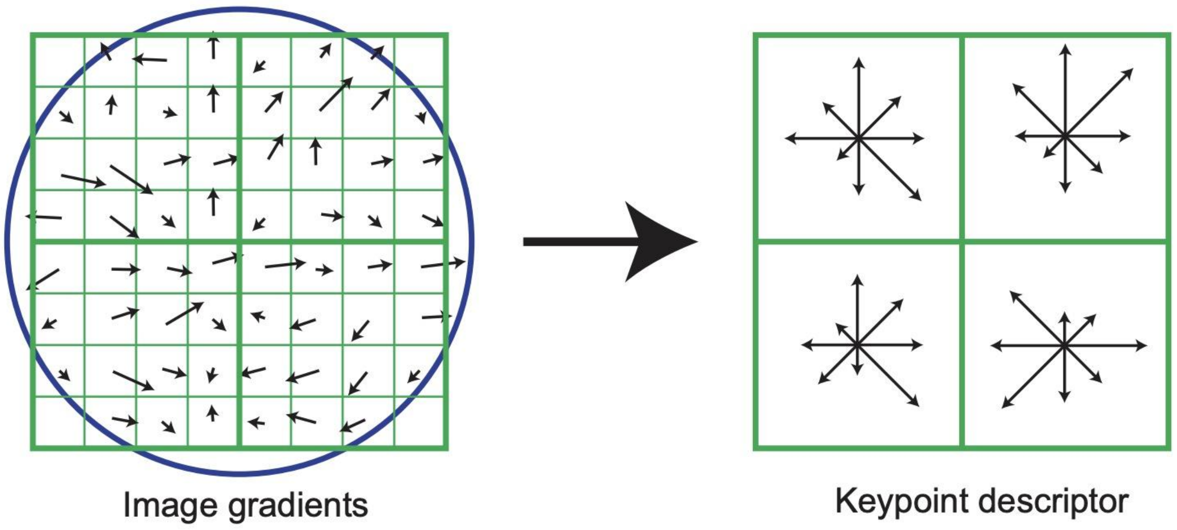


Figure6 Keypoint descriptor

The last thing we need to do is generate the descriptors of the keypoints. First we rotate the axes to the orientation of the key points to ensure rotation invariance. Then we take an 8×8 window centered on the keypoint. The center of the left part of the figure is the position of the current keypoint, each small cell represents a pixel in the scale space where the key point neighborhood is located, and the gradient magnitude and gradient direction of each pixel are obtained by formula, and the direction of the arrow represents the gradient of the pixel direction, and the length of the arrow represents the gradient modulus value, which is then weighted with a Gaussian window. The blue circle in the figure represents the range of Gaussian weighting (the pixel gradient direction information that is closer to the key point contributes more). Then calculate the gradient direction histogram of 8 directions on each 4×4 small block, and draw the accumulated value of each gradient direction. Finally we get the 128 dimensional descriptors for keypoints.

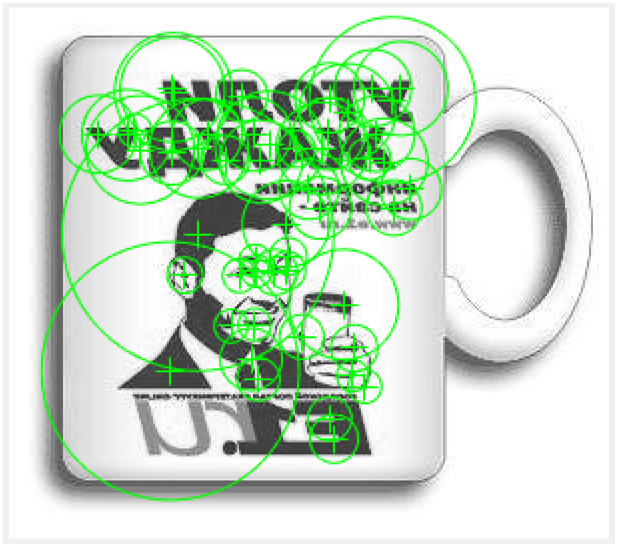


Figure 7: Use SIFT Algorithm to detect features on the image. It shows the strongest 50 features that the algorithm can detect.

Visual Words and Histogram Formation

For each training image get some SIFT features. The features allow the image patches to be represented as numerical vectors. Once you have the SIFT features cluster with K-means. For each image, testing and training, get the SIFT features again. Assign each feature using K-means. K-means groups visually similar features into one cluster. The number of clusters is dictated by the size of the vocab, or the number of features we extract the first time SIFT is performed. The size of the vocab is an important parameter in this model. A vocab that is too small results in the visual words not being a good representative of all features. If the vocab is too big then overfitting is likely to occur. Next, construct a histogram that indicates how many times each visual word appears in the image.

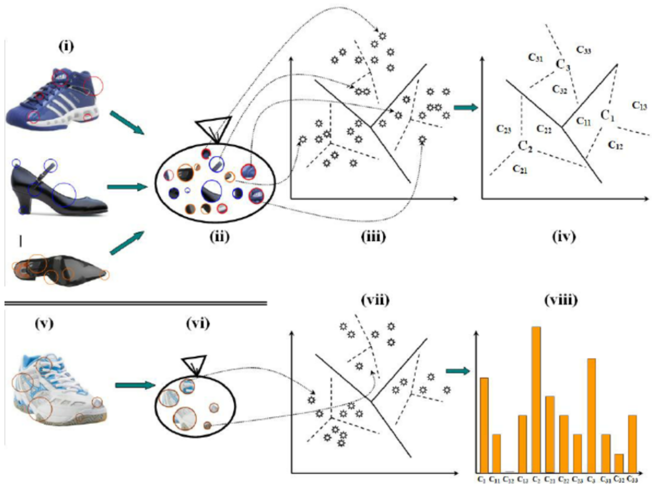


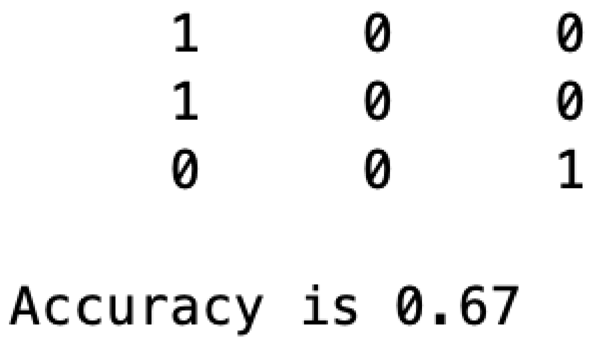
Figure 8: Shows the process of extracting features from an image, assigning those features to a cluster which is mapped to a visual word, and then creating a histogram.

Training and Testing(SVM)

Several methods can be utilized for image and vocabulary classification; however, most of the traditional approaches can not handle high dimensionality of the feature space. That is, large data sets indicate a high number of sections of the tested image that need to be extracted in order to perform image classification on the data set. SVM, however, is an optimal solution for image classification if the feature space is represented by histograms. In the case of Bag of Features, as described above, SIFT is performed in order to convert features into histograms. Therefore, multiclass SVM can be trained on the data histogram models in order to classify the training set. Each label of the SVM corresponds to one of the visual words. Soft margin SVM is utilized due to the possibility of inseparable data. A soft margin is implemented in order to relax the separating conditions. Furthermore, Matlab has a built-in function to solve quadratic programming functions that are used in the SVM.

Conclusion

In one word, we implement the whole bag of words model using SIFT algorithm and k means algorithm. In order to test our result, we build a linear multiclass SVM to train and test our final approaches. The test accuracy of our bag of words model can reach 0.6667.



```
1    0    0
1    0    0
0    0    1

Accuracy is 0.67
```

Figure 9: Confusion matrix and accuracy we got from our bag of words model

Based on our research, the common accuracy that a linear multiclass SVM can get is around 0.6. Nowadays, the bag of words model is still powerful in image classification, but it is not the best. Many algorithms, like Convolutional Neural Networks, can reach a high accuracy in image classification. Overall, the bag of visual words model has a high flexibility for customization and it is easy for us to understand. But it still has some limitations. For example, it does ignore spatial relationships.

Contributions

For the pre-process part, Chujun Qi and Zhengtao Wang built this together. The structure of SIFT parts was built by Gabrielle Kuntz, Zhengtao Wang and Chujun Qi. Egor Badich and Suyang Yan built the multiclass SVM. The presentation and report was worked on by all members.

References

[1] Jialu Liu. Image Retrieval based on Bag-of-Words model. In Arxiv, page 3, 2013.
[2] Scale-Invariant Feature Transform. Link: https://kobiso.github.io/research/research-sift/
[3] SIFT -Scale-Invariant Feature Transform. Link: http://weitz.de/sift/
[8] Process of Histogram Formation.
Link:https://www.researchgate.net/figure/Illustration-of-the-bag-of-visual-words-approach-that-we-use-for-classification-The_fig3_221301237

Chapelle, O., Haffner, P. & Vapnik, V.N., 1999. Support vector machines for histogram-based image classification. *IEEE transactions on neural networks*, 10(5), pp.1055–1064.

Lowe D G. Distinctive image features from scale-invariant keypoints[J]. International journal of computer vision, 2004, 60(2): 91-110.