ETH zürich | Computer Vision and Geometry Lab

December 22, 2018

Computer Vision

Lab 10: Image Categorization

Sumeet Gyanchandani

In this lab, I implement a bag-of-words image classifier that decides whether a test image contains a car or not.

1 Local Feature Extraction

1.1 Feature detection

I implemented a very simple local feature point detector: points on a grid. I calculated and rounded up a step size in both X and Y dimensions. That helped me get a mesh of points.

1.2 Feature description

Next, I described each feature (grid point) by a local descriptor. I used the histogram of oriented gradients (HOG) descriptor. I implemented a function for hog descriptors which should computes one 128-dimensional descriptor for each of the N 2-dimensional grid points. This was achieved by calculating the top corner of the cell for every point in **vPoints** and offsetting in both the directions to get the cell around the points. Then, I found the histogram and image patch for every cell.

2 Codebook construction

2.1 Nearest neighbour matching

I created a function that returns for each feature vector in D1 the index of the nearest neighbor in D2, as well as the Euclidean distance to this nearest neighbor.

2.2 K-means clustering

Here, I used *findnn* function to find the association between features and cluster centers and then calculated the cluster centers as mean of the associated features. I repeated this process *numiter* times.

2.3 Visualize codebook

Figure 1 shows a sample output of this function

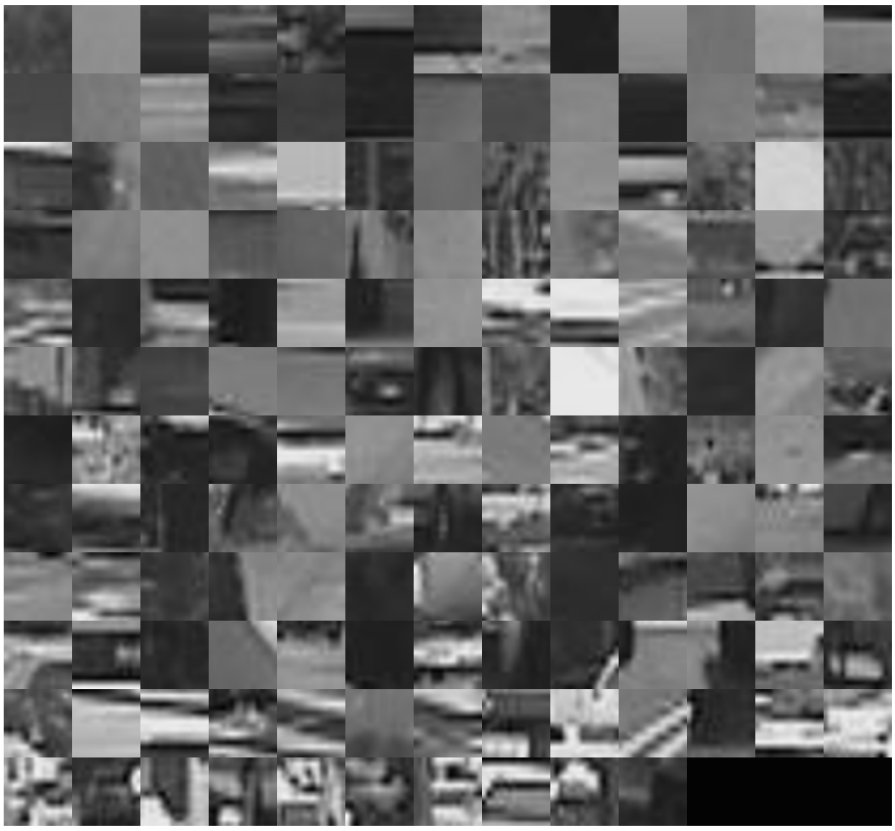

Figure 1. Visual Words of the Codebook

3 Bag-of-words Histogram

The function *bow_histogram* computes a bag-of-words histogram corresponding to a given image. The function takes as input a set of descriptors extracted from one image and the codebook of cluster centers. To compute the histogram to be returned, it assigns the descriptors to the cluster centers and counts how many descriptors are assigned to each cluster.

4 Nearest Neighbor Classification

For classifying a new test image, we compute its bag-of-words histogram, and assign it to the category of its nearest-neighbor training histogram. It compares the bag-of-words histogram of a test image to the positive and negative training examples and returns the label 1 (car) or 0 (no car), based on the label of the nearest-neighbour.

5 Bayesian Classification

Here, I calculate the required probabilities for the Bayes Theorem. While calculating the values of **P(hist|Car)** and **P(hist|!Car)**, I evaluate the sum of the logarithm of each probability term instead of a product and I make the value of column in product expression 1, if it has **NaN** values. Finally, if P(hist|Car) * P(Car) > P(hist|!Car) * P(!Car), I set the label to 1 (car). Otherwise, I set it to 0 (no car).

6 Results

Codebook Size	Nearest Neighbor Classification	Bayesian Classification
120	0.9823	0.9091
160	0.9242	0.8914
200	0.9545	0.8788
240	0.9343	0.8611
280	0.9646	0.8005
320	0.9495	0.7753

Table 1. Categorization accuracies in various scenarios

I ran multiple tests for a single Codebook size with different random seeds. Then, I averaged these accuracies. Table 1 shows the accuracies for various codebook sizes averaged over multiple runs.

My conclusion:

- The overall accuracies for Nearest Neighbour Classification for the given images is better than the Bayesian Classification
- When we increase the value of k, Nearest Neighbour accuracies may increase or decrease. In codebook sizes that I have considered, they alternate between increasing and decreasing
- In contrast, with the increase in the value of k, the accuracy of Bayesian Classification decreases