Computer Vision

Image Categorization

Student: Yuchang Jiang

Date: 14 Dec. 2020

1. Implementation

1.1 Local feature extraction

$grid\_points.m$ is used to create grid defined by number of points in x, y direction and the border. Here $meshgrid$ matlab function is used to help generate grid points.

Then the descriptor called Histogram of Gradient (HOG) is computed for the defined patch of each grid point. For each patch, it has 4x4 cells (cell is defined as a region of $cellWidth$ x $cellHeight$ pixels) and for each cell, the direction of gradient will be summarized in a 8-bin histogram so in total, there are 4x4x8 bins or a 128-dimensional vector for each grid point.

1.2 Codebook construction

$create\_codebook.m$ is implemented to capture the appearance variability by constructing a visual vocabulary. For each training image (include both positive and negative images), previous built functions are called to get grid points and generate HOG descriptors. With all HOG features, KMeans is performed to find clusters and the centers of clusters are considered as 'visual word' for our vocabulary. There is one important parameter called size of codebook, K, which is the number of cluster stated in KMeans.

1.3 Bag-of-words image representation

$bow\_histogram.m$ is implemented to compute the histogram given the visual words (centers from KMeans) and computed descriptors of image. Here $knnsearch$ matlab function is used to find the closest center to each descriptor of the image. Then a histogram of visual words can be computed for this image. Based on it, $create\_bow\_histogram.m$ is used to generate histogram of all training images and save them in a matrix.

1.4 Nearest Neighbor Classification

The implementation of Nearest Neighbor Classification is quite straightforward here ($bow\_recognition\_nearest.m$): call $knnsearch$ matlab function to find the distance between test image histogram and the closest positive or negative image histogram. If the positive distance is smaller, this image is classified as positive and vice versa.

1.5 Bayesian Classification

In $bow\_recognition\_bayes.m$, test image will be classified as positive if $P(car|hist) > P(!car|hist)$. As these posteriors have the same denominator and priors are equal ($P(car) = P(!car) = 0.5$), we can focus on checking $P(hist|car) > P(hist|!car)$ instead. If each visual word is assumed independent of each other, the probability of all visual words is the product of the probability of visual words. This assumption is used for both positive and negative distribution. To avoid the product of small numeric number, I take a log of probability and turn to compute the sum of log probability. Therefore, after computing the distribution of each visual word in positive and negative images, $normpdf$ matlab function is used to find $P(hist(i)|car(i)), P(hist(i)|!car(i))$ for i-th visual word. The final sum of log probability is compared: if $P(hist|car) > P(hist|!car)$, this image is labeled as positive and vice versa.

2. Results and Discussion

To check the performance of Nearest Neighbor Classifier(NNC) and Bayesian Classifier (BC) with different codebook size, codebook size from 20 to 420 with 40 as interval are tried in the experiment. In Figure 1, the visualization results of codebook with size 20 (left) and 200 (right) are displayed. It is noticeable that the codebook with small size captures more general features while the codebook with large size captures more detailed features.

(a) K=20 (b) K=200

Figure 1: Visualization of codebook

As shown in Figure 2, the performance results with different codebook size for two classifiers are plotted. In general, both of them can achieve more than 90% accuracy in most cases and the performance of Bayesian classifier looks better. It seems the performance of Nearest Neighbor classifier is good for small codebook size while the performance of Bayesian classifier is improved with the increasing of codebook size.

A large codebook size means using more visual words in histogram. As Bayesian classifier uses the visual words to estimate probability, a large number of words can lead to more accurate probability estimation. On the other hand, Nearest Neighbor classifier takes the histogram of visual words as 'point' in latent space so increasing the codebook size means projecting the 'point' to a higher dimensional latent space, which may be not necessary for finding nearest neighbor. Besides, the good performance of Bayesian classifier suggests modelling probability is better than finding nearest neighbor in feature space (if the independence assumption of Bayesian classifier holds true). However, when codebook size is larger than 200, the performance of Bayesian classifier stops increasing. It means choosing a suitable codebook size is important: either too small or too large will make the result worse.

To summarize, Bayesian classifier is better for this exercise and codebook size 200 is preferred.

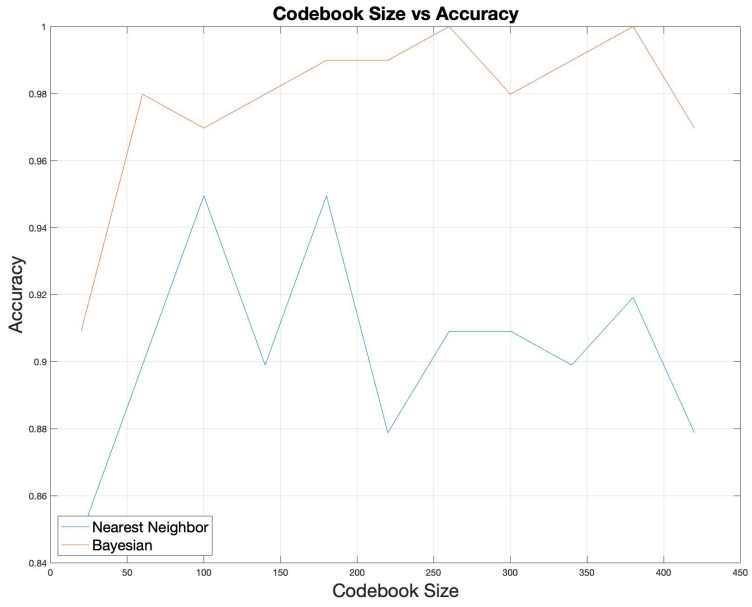

Figure 2: Performance with different codebook size (fix random seed))