ETH ZURICH, D-INFK

Computer Vision Exercise 8: Image Categorization

Xingchang Huang

December 14, 2017

1 LOCAL FEATURE EXTRACTION

The features we use here is combinations of local features. For a whole image, we would divide it into grids and extract features (HoG) on grids. After that, we should use kmeans clustering algorithms to construct bag of visual words and then construct a histogram based on visual words as a feature descriptor for that image. Then we can use a classifier to train and classify images based on images' descriptors.

1.1 FEATURE DETECTION - FEATURE POINTS ON A GRID

For each image, we would divide it into 100 parts with 100 (10×10 for x, y axis) grid points (points in the center of grids). For each grid point, we will use its surrounding 16×16 points or region to generate the descriptor of it. Therefore, we should leave 8 pixels for each grid point in all 4 directions to ensure that the region is still inside the original image.

1.2 FEATURE DESCRIPTION - HISTOGRAM OF ORIENTED GRADIENTS

For the surrounding 16×16 pixels, we need to divide them into 16 parts further, which may have the region of 4×4 pixels. Inside that 4×4 region, we need to get the gradients of both x, y direction and compute the angles using atan2 and magnitudes of gradients. The angles of gradients are used to decide which bin it belongs to and the magnitudes should be considered the added value of that bin. Here we get the histogram with 8 bins generated uniformly from angle 0 to 360. In this way, with 16 regions for a grid totally, we can have $8 \times 16 = 128$ bins in a whole histogram to represent a local descriptor of that image.

With 100 grid points, we can have totally 100×128 feature descriptors for each image. Thus, for N images in training set, we can get a $(N \times 100) \times 128$ matrix to represent features of training images. Finally, we also need to extract image patches (4×4 region) from original images for visualization as shown in the following figures. Here we need to subtract one from each direction because the grid point and its surrounding pixels have totally 17×17 pixels. From the figures, we can find that if the number of centers can not be defined well, the patches that closest to the center visualized would become a totally dark region, which cannot show the features of the car well and thus have negative influence on the classification accuracy.

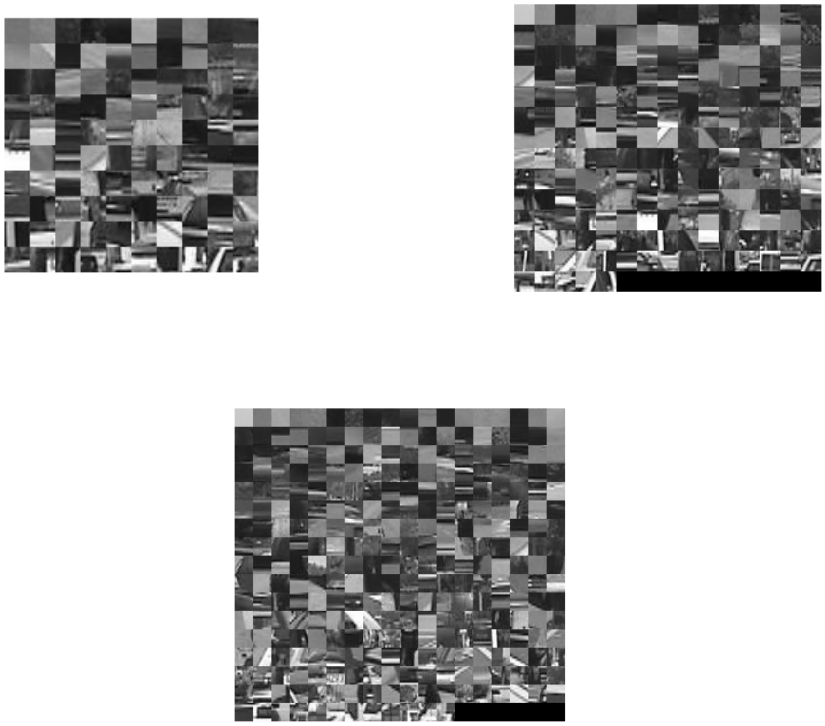

Figure 1.1: Examples of visualization when codebook size is 100, 200 and 300, respectively.

2 CODEBOOK CONSTRUCTION

After constructing histogram as feature descriptors for images in the training data set, we then use kmeans clustering algorithm to find the centers of those descriptors (128-dim vector), which is called codebook construction. Then we can assign each descriptor a cluster center and construction a new histogram, where the bin number is the number of clusters and the value is the count of features belonging to that cluster center. With 200 codebook

size, we will finally construct a new histogram with 200 bins to represent the final descriptor for each image used for classification later. The function "findnn" used in kmeans is basically to find the nearest center for each image feature descriptor. Thus, it returns a vectors with size of number of images.

It should be noticed that some bins may be 0 because the corresponding cluster centers may have not features assigned to it when the codebook size is large.

3 Bag-of-words image representation

As discussed above, we will use cluster centers of all $(N \times 100) \times 128$ descriptors to construct a new bag-of-visual-words histogram representation. The original representation is a 100×128 matrix and the new representation will become a vector of size 200 according to the cluster centers, where 200 is the number of visual words. For each center, we should find the features belonging to it and then add the counting to the histogram.

4 Nearest Neighbor Classification

For a test image, we just need to construct histogram of it using bag of visual words representation and find the nearest (Euclidean Distance) histogram among all training data (positive and negative), to see its nearest neighbour is positive or negative and assign the predicted label to it. Here I just use min function in MatLab to find the nearest item. The comparison of nearest neighbor and bayesian classifier should be given in the next part.

5 Bayesian Classification

Here, we use bayesian classifier to do classification. Basically, this is a Naive Bayes model because each dimension of feature descriptor or histogram are considered independent of each other. Also, each dimension (column) has continuous value because we cannot know how many feature descriptor would be assigned to a cluster center. There, in order to find a posterior of those continuous values, we need to use a gaussian distribution to fit the values and then for a new value we can find the corresponding value according to the estimated gaussian distribution by training data.

Here, for each dimension in the bag of words histogram, we use the descriptors of training images to estimate the mean and standard deviation for each dimension to determine a specific gaussian distribution for each dimension. Then in the test step, we have a test image with its histogram and we can know the value of P(hist|car) or P(hist|!car), with probability from the existing estimated gaussian distribution. And then we do cumulative product for all dimension and then we can find the probability of the histogram given car or not car, which is used to compute the probability of car or not car (P(car|hist) or P(!car|hist)) by bayesian rule.

It should be noticed that the cumulative product here may cause overflow and numerical problems when it meets with very small number, so I use log first and sum over all probability and finally use exp function to get the needed probability. If we get the probability P(car|hist), we can actually compare its value with 0.5 to determine the label, with no need to calculate the P(!car|hist) as well. The following table shows the results and parameters for classifiers. Notice that with different size of codebook, I fix the number of iterations for kmeans as 10.

From the table, we can conclude that the best performance for bayesian classifier with codebook size 100, whose visualization contains more reasonable features without obvious dark regions. Here NN classifier performs better. I suppose that the reason for this phenomenon is that when we do cumulative product on probability, if one of the number of probability in the histogram is small, it would make great influence on the final calculated probability to be a very small number. This may come from the choice of cluster centers using kmeans and the value of histogram would be small because some cluster centers may not have surrounding features.

codebook	classifier	k=1	k=2	k=3	mean acc.
100	Bayesian	0.86869	0.84848	0.89899	0.87205
	NN	0.94949	0.93939	0.90909	0.93266
200	Bayesian	0.89899	0.81818	0.82828	0.84848
	NN	0.93939	0.9697	0.94949	0.95286
300	Bayesian	0.74747	0.88889	0.76768	0.80135
	NN	0.9596	0.89899	0.9697	0.94276

Table 5.1: Classification results comparisons of nearest neighbor (NN) and bayesian classifiers with different size of codebook with repetitions (k=1, 2, 3).

6 QUESTIONS ANSWERING

1. What is your classification performance using the nearest neighbour classifier? The classification result of nearest neighbor classifier is from around 0.93 to 0.95, with size of codebook from 100 to 300, respectively. So the mean performance of nearest neighbor classifier may be around 0.94.

2. What is your classification performance using the Bayesian classifier? Is it better or worse? How do you explain this difference? The mean classification accuracy of bayesian classifier is worse than nearest neighbor one, varying from around 0.8 to 0.87 and cannot reach over 0.9. As discussed in the previous section, I suppose the the reason may be that when we do cumulative product on probability using bayesian rule, if one of the number of probability in the histogram is small, it would make great influence on the final calculated probability to be a very small number. Those small numbers may come from the unsuitable choice of cluster centers using kmeans and the value of histogram would be small because some cluster cen-

ters may not have surrounding features (e.g., dark regions).

3. Vary the number of cluster centres in your codebook, k. How does your categorization performance vary with k? Can you explain this behaviour? Again, as shown in the table, when the size of codebook k=100, the performance would be the best among them. When we increase k, the performance of classification would decrease. The explanation can come from the visualization of different size of codebook. When k=200 or 300, we would find some regions that are not meaningful to the features of car, which may have negative influence on the result while k=100 can bring much better visualization on features.

7 BONUS: OWN DATASET

I use a small dataset collected from CASIA-Webface database and it is constructed manually, containing 48 images for man and 241 images for women in training data and 6 images for both man and women in testing data. Therefore, this is also a binary classification task for deciding the gender, whether a image is a man or a woman. I construct the bag of visual words based on training data of man. Because the images of man is far more less than women, it would be a potential problem when constructing the bag of visual words based on man.

The reason for me to choose the data set of human faces is that the features of different genders would be more subtle than the features of car and not car, which may bring more difficulty for classification. Also, the bag-of-words here is hard to construct due to subtle difference by kmeans, which may show the deficit of this method used in human faces. The example image in the dataset is shown as the following. The result decreased sharply in this dataset but the bayesian classifier performs better. In this way, we can conclude that for subtle difference and more difficult tasks, bayesian model considers the distribution of features, which would give better result when nearest neighbor classifier just classify by distance.

Therefore, the bayesian classifier could reach at most 0.66 while the nearest neighbor classifier reach at most 0.61. More details of visualization of image patches and classification results have been put in the following figures and table. From the visualization, you can find image patches of key part on human faces.

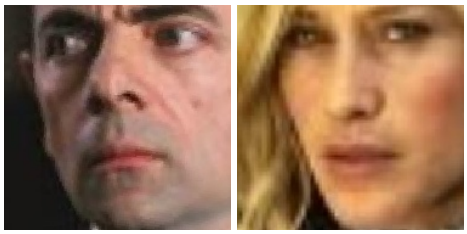

Figure 7.1: Examples of images in my faces gender dataset.

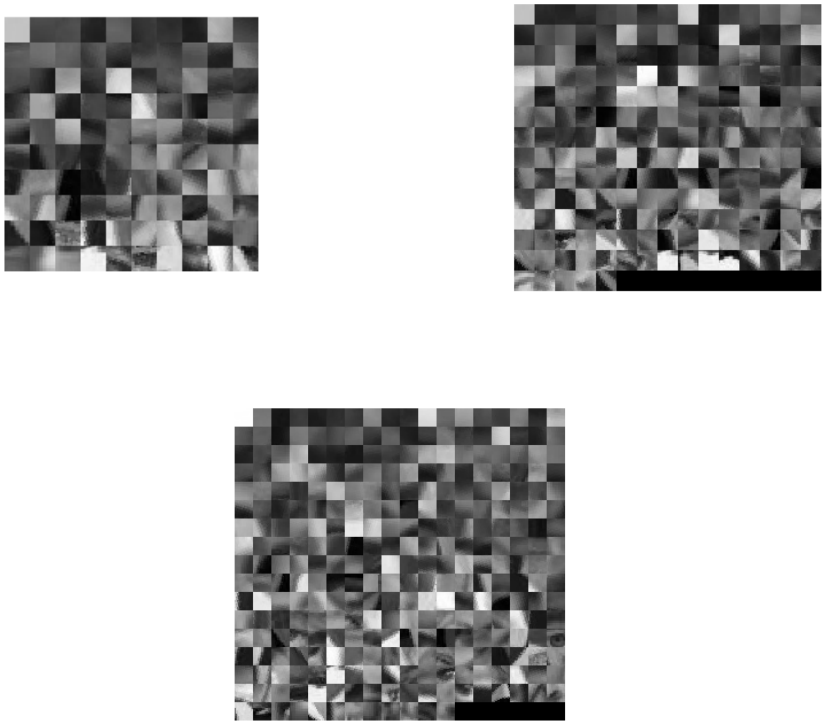

Figure 7.2: Examples of visualization in my dataset when codebook size is 100, 200 and 300, respectively.

codebook	classifier	k=1	k=2	k=3	mean acc.
100	Bayesian	0.58333	0.75	0.58333	0.63889
	NN	0.66667	0.66667	0.5	0.61111
200	Bayesian	0.66667	0.58333	0.75	0.66667
	NN	0.41667	0.5	0.58333	0.5
300	Bayesian	0.66667	0.58333	0.66667	0.63889
	NN	0.58333	0.58333	0.58333	0.58333

Table 7.1: Classification results comparisons of nearest neighbor (NN) and bayesian classifiers on my faces gender dataset.