Object recognition

Qi Ma, 20-960-225, qimaqi@student.ethz.ch

December 13, 2021

Abstract

This assignment teach me to use two way for object recognition. The bag-of-words method is very intuitive and implementation is very quick and the CNN-based method is highly accurate but hard to train.

1 Bag-of-words Classifier

The first task uses bag-of-words classifier to determine whether a car or not.

1.1 Local Feature Extraction

1.1.1 Feature detection - feature points on a grid

This part ensures we sample points uniformly distributed around the images and all possible feature will be counted.

1.1.2 Feature description-histogram of oriented gradients

In this part all detected grid points will be surrounded by 16 grid while each grid have 16 pixels inside. Since we do not consider the rotation problem so we delete the part of finding main direction and de-rotate the patch. Inspired by the discussion in Moodle question forum. We can try building the HOG with the vote of gradient magnitude. In the final section we will compare the result of magnitude voting and equal-weight voting and also normalized magnitude.

1.2 Codebook construction

For this part we cluster the large set of all local descriptors from the training images into a small number of visual words (which form the entries of the codebook). What need to be done is determining the number of cluster k and also iterations. The discussion will be in the final section.

1.3 Bag-of-words Vector Encoding

After building the codebook, what we need to do now is encoding the image from positive and negative dataset and use it for distance calculating for test samples. The basic idea is encoding each image with a histogram which number of bins is number of visual word in codebook.

1.4 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. This method will in another word finding the most similar image in training set.

1.5 Discussion

1.5.1 How to choose K

The number of cluster center K can be understand as how many important difference between car and non-car image. Since each descriptor is small (16x16 pixels around one grid center point), so we can treat the visual words as "Auto Exclusive Features". Because most of the pictures are of the rear of the car so we can say important features should be "Car Tail Light", "Car license plate","Car wheels","Car Rear Window", "Car Rearview Mirror", "Car bumper edge". So based on there prior knowledge we can set k to be 10 and make some room for mistake (missing centers will influence more than extra centers). The result we have shows the result we assumed. Too little cluster can not distinguish the car and non-car scenario, and as soon as we achieve critical value, the accuracy will become stable as K increasing. In following discussion we will set k as 70.

Figure 1: The rear of the car.

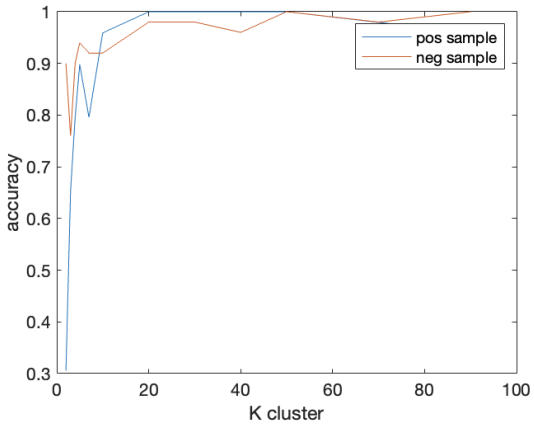

Figure 2: The performance of different K.

1.5.2 How to build HOG

The standard way for build HOG is voting with gradient magnitude since the large gradient value contains the information about the speed of how pixels value changes so the HOG based on gradient magnitude should perform better than HOG which voting equally. However from the experiment below it shows that the best performance comes from the equally voting and the magnitude based voting have bad performance in negative sample recognition. This can be explained by the fact that the illumination for each sample is not equal, so considering the magnitude of gradient will bring in environment based factor instead of vehicle itself. If we just consider the gradient direction instead of magnitude, we may be vote wrongly because background may have smaller gradient magnitude compared to the vehicle but in our case the background is road. So road have quite different magnitude direction with vehicle so it's good if we vote equally.

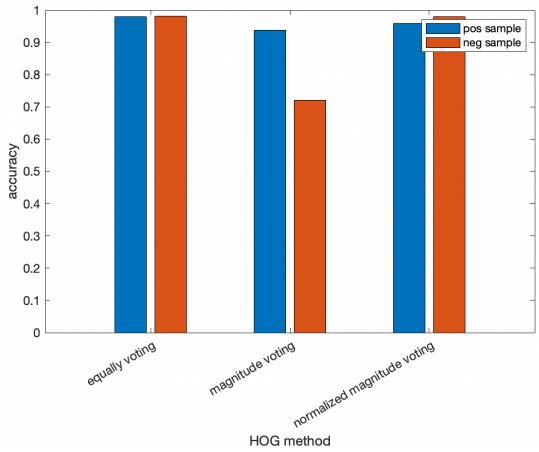

Figure 3: Different way to calculate the HOG.

2 CNN-based Classifier

This tasks oriented at the most exciting part of machine learning while computer beat human in image classification.

2.1 A Simplified version of VGG Network

Figure 4: VGG Net structure from paper.

In this part we use a simple version of VGG network which compared to the original VGG we have less convolution layers.

2.2 Training and Testing

First we show the tensorboard result which shows constantly decreasing of training loss.

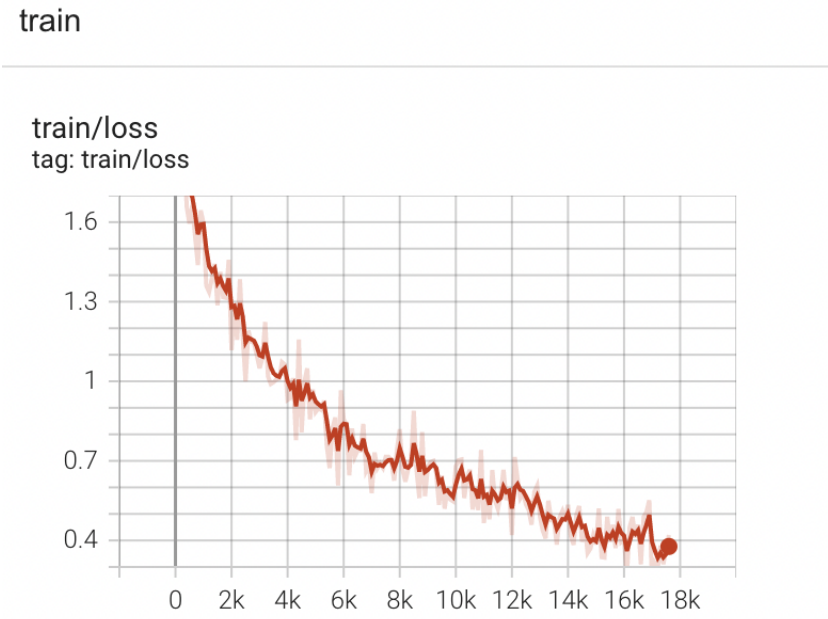

Figure 5: Training loss converge.

Then we show the validation accuracy increase and achieve 0.82 finally.

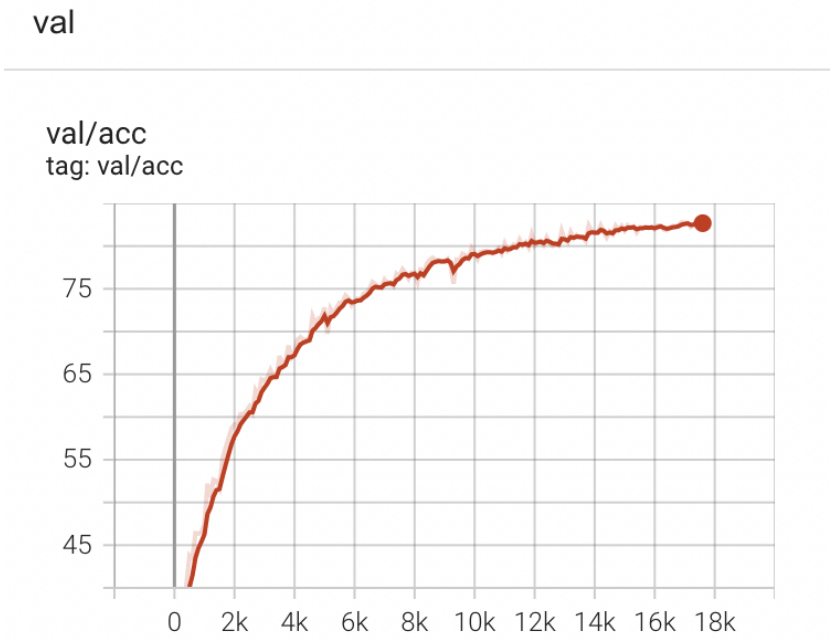

Figure 6: Validation loss converge.

In testing part we achieve 82.16 accuracy which pass the requirement.

```
[INFO] test set loaded, 10000 samples in total.
79it [00:40,  1.96it/s]
test accuracy: 82.16
```

Figure 7: Final testing result.