# k-Nearest Neighbors Algorithm to Find Red Car in A Random Image

First A. Author, *Tianyu Zhou, IEEE*

***Abstract*—It Is hard to find something in one large image by hand, so it is necessary to find a way to use computer to help us finding object in images or other thing. The purpose of this project is to help people find red cars in a picture. I use the principles of machine learning, find the characteristics of the red car. And then use these features to find other red cars in the picture.**

***Index Terms*—machine learning, find, red car, characteristic**

## I. Introduction

IN this project I use the method of machine learning to find red cars in one image. I use python programing to build some code, which can easily call library to help us finish the job. And I use k-nearest neighbors algorithm way to distinguish two classes based on characteristics I find. After that, I use training data to compute the accuracy and find the most suitable k for our experience. At last, I will use this k to test the data to find the red car.

## II. Implementation

If I want to use machine learning method to achieve this project, there are some parameter I must find. Those are the characteristics of the red car and the characteristics of those object which are not red car. After I get these characteristics, I can compare every object to these characteristics that can help me to identify if it is a red car. I can also add characteristics for red car to improve the accuracy.

So if I want to finish this project I need to complete the following steps. Manually find numbers of red cars, and something which are not red car. Determine the characteristics of the red car that different from other thing. Use the KNN [2] (k-Nearest Neighbors) method to specified two class and train the computer to specify the red car. I can also change k to get accuracy that can find the most suitable k value. Finally, use that k for KNN to find other red cars.

If can not be 100% accuracy, so if we have time, we can get some method to improve the accuracy. We can try some other method to find the red car, add other characteristics to the red car, add a filter let the good data pass, etc.

## III. Experiments

First, I need to determine the method I need to use. There are many methods I can use, such like squared error objective, probabilistic generative classifiers, k-nearest neighbors, and k-means. I will discuss the advantages and disadvantages of each method in the following paper.

### *A. Method*

a. Probabilistic Generative Classifiers

I won't discuss squared error objective separately, because squared error objective separately is a very simple method, and it is limitations are large and the estimated eigenvalues are inaccurate. [3] So probabilistic generative classifiers combine squared error objective and other methods. Compared to squared error objective, probabilistic generative classifiers have a big improvement.

However, I do not want to use probabilistic generative classifiers. Because in our previous experiments, I find out that probabilistic generative classifiers will help 2D and 7D data get high and stable accuracy, k-nearest neighbors will help HS the high accuracy. Which means probabilistic generative classifiers for low-dimensional data and k-nearest neighbors for high-dimensional data can achieve higher accuracy.

Since in this project, I need to judge high-dimensional data, so I choose k-nearest neighbors.

b. k-means

K-mean also has certain limitations. That is, I need to know how many classes it needs to be divided into first. However, I do not know the specific number of class I need to divide into. And k-mean use the class to identify the red color directly, it only finds the red point, not the red car. So there is no process for machine learning, just a process to find red directly.

Maybe I can combine the KNN to k-mean, but I do not know how to do it, so I choose KNN method to train my data.

### *B. Choose train and test data*

From the beginning, I want to test a very large image form the train data, but I found out that it will cost a large amount of time deal with these data and I failed. So cut out a part from the train data is a good way to save time.

To the red car data, I find the center point of it. To the random data, I find some random point except the red car point. I can use these data to the following step and can save many time.

### C. *Find Characteristics*

I find some point last step, but I find out that if I only use the center point of the red car, it can only show the RGB data of that point to me. It is useless, because the red car not only have color data but also have shape and size. So, it is necessary to get the data from point around the center point.

Then I cut out a 10*10 size rectangle based on the position of the center point both red car and random point. If I cut a larger rectangle, it will contain much more information, and will confuse the train data. So 10*10 is suitable for me.

Now I have enough data to define the class. I define red car rectangles as class 1. Then the random rectangles as class 2. There data can do the KNN train.

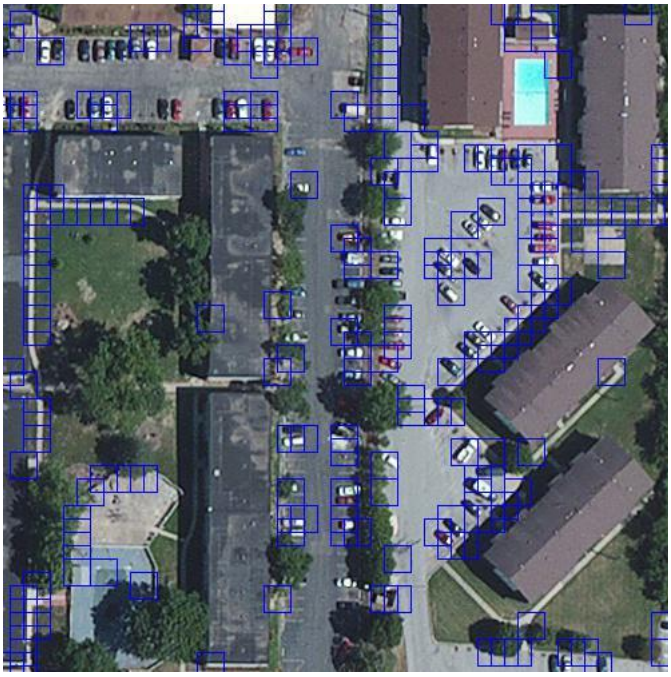

Fig. 1. The bad result to find red car in data_test image.

This is a bad result for data test image. Because I do not choose the right characteristics for red car, the result not only circles cars of various colors but also the road, roof, etc. So the accuracy of this result will be very low.

From this result, I know the importance of choosing the good characteristics.

### D. *k-Nearest Neighbors Algorithm Train (cross validation)*

I now have two classes of data. I combine them into one set of data, and then I can do the cross validation. I disrupt the data. Extract some data from it to train, the rest of the data is used to test the result of the train. After that I can use the result of these rest of data compare with the labels of these data to get the accuracy.

After that, I can change the value of k to test which k get the highest accuracy. Choosing this k for out test data.

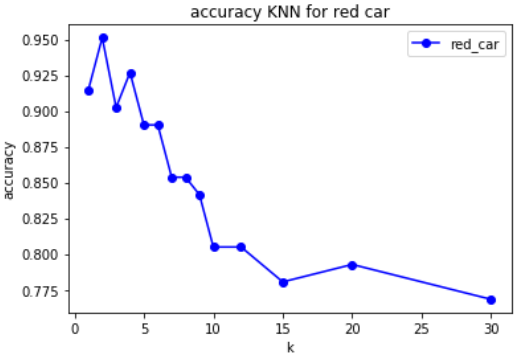


Fig. 2. The red car KNN accuracy change from K change.

Form the figure we can find that k=2 maybe the best answer, but if k is even it may be some problem. So maybe choose k = 3 is better.

### E. *Test the data*

If I want to use this method to test the data, I should build the test data to a 10*10 size rectangle just like the red car data which can improve the accuracy. So I cut the image into several parts, each part is a rectangle of 10*10 size. Comparing each of these part to class 1 and class 2. And classify it as class 1 or class 2. The rectangles classified as Class 1 are all red cars.

Then I can find out where the test red car rectangle is located. Circle these positions with a blue rectangle. Add these blue rectangles to the corresponding test image.

Finally, I can see the red car circled in the test image.

### F. *Color filter*

In the process of circling the red car. I found that there are still many other rectangles that are encircled into the range of red car. These rectangles include some other color car, so a color filter is necessary.

To build the color filter, I need to know the range of red color. But it is difficult to definition red color range from RGB value. So I change the RGB value to HSV value.

| | 黑 | 灰 | 白 | 红 | | 橙 | 黄 | 绿 | 青 | 蓝 | 紫 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| hmin | 0 | 0 | 0 | 0 | 156 | 11 | 26 | 35 | 78 | 100 | 125 |
| hmax | 180 | 180 | 180 | 10 | 180 | 25 | 34 | 77 | 99 | 124 | 155 |
| smin | 0 | 0 | 0 | 43 | | 43 | 43 | 43 | 43 | 43 | 43 |
| smax | 255 | 43 | 30 | 255 | | 255 | 255 | 255 | 255 | 255 | 255 |
| vmin | 0 | 46 | 221 | 46 | | 46 | 46 | 46 | 46 | 46 | 46 |
| vmax | 46 | 220 | 255 | 255 | | 255 | 255 | 255 | 255 | 255 | 255 |

Fig. 3. [1] The color range for HSV. H: 0 – 180 S: 0 – 255 V: 0 – 255

From the figure 2 I can find out that the red color range is:

0<= h <= 10 or 156<= h <= 180 and 43<= s <= 255 and 46<= v <= 255.

I can use this range to definition the red color. I judge that if there are enough red pixel in one rectangle that I can sign it as red car. So only red car can pass this filter.

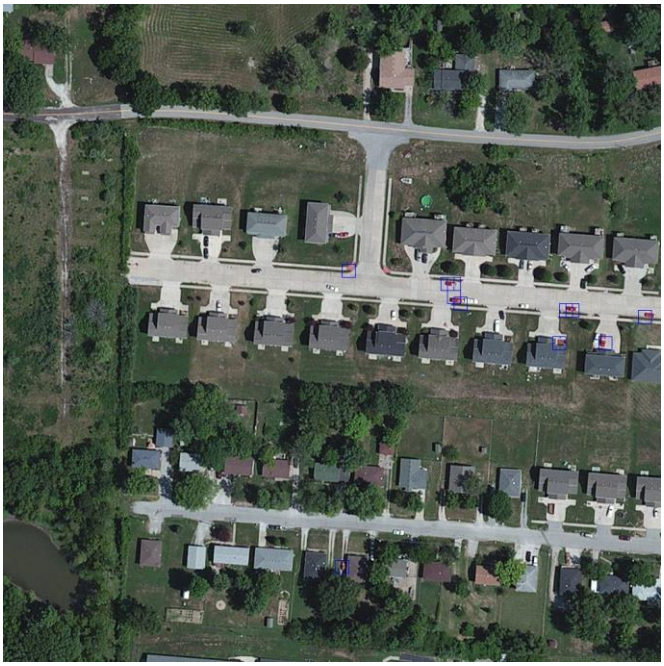

Fig. 4. The result to find red car in data_train image.

This is the result for test the data_train, from this image we can see that there still have some problem with the study process.

we can see that it actually circles most of the red cars, but there still have some red cars not be circled. And if the rectangles divide the red car to two red cars, the computer will judge there are two red cars. And there are some red things which not red car are also be judged to red car.

This method also can not use for a large amount of data, it will cost a lot of time.

So there still have some part need to be improve.

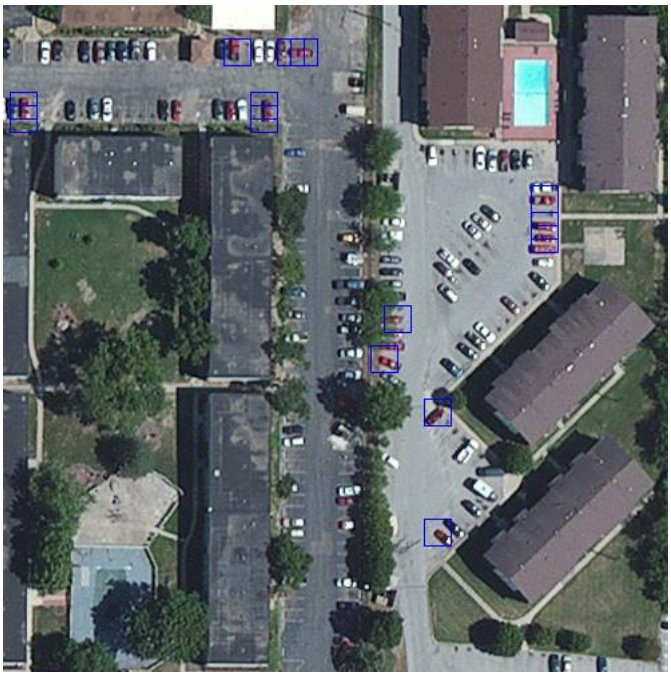

Fig. 5. The result to find red car in data_test image.

This is the image for data_test (unseen) image. We can see that it work good in some way. The computer avoid to show some red rectangles such like red swimming pool and red roof.

## IV. Conclusions

I learned the following through this project.

### A. *Optimizer*

Because this project involves a large amount of data operations and processing. If the optimization is not done well, it will cause the program to run such a long time. There are some method to improve my program.

First, do not choose a large data to train my computer. Second, optimize the for loop. Do not nest too many for loops and do not do too many operations in the for loop. Third, define more function, that can save a lot of time.

### B. *KNN method*

From this project, I find out that the more data for class data, the more accuracy result is. But if I get too much data, the code will be slow. So the suitable number of data can get good result.

Find good characteristics is also important to get good result, clear contrast is necessary.

Also, KNN may be good for shape identification, but it is not good for color identification. Maybe we can add other characteristics or combine it with another method to get good result.

I think combine KNN with k-mean may be a good way to try.

### C. *K-mean method*

I think k-mean is not suitable to use alone. I try it, but it just highlights the red point, not the red car. But it is useful to find the red car color in RGB. So I think I can combine it to KNN next time.

### D. *Answer the questions*

1. I have already done this in the beginning of Experiments.
2. I choose parameter k from the figure 1.
3. I think my method can find most of the red car. For the unseen test data, the result shows in figure5. But there still have some problem. Not all red car be circled. Some red car circle two times. Some red thing be judged as red car. Large amount of data will cost a lot of time.

## References


[1] Bishop C M. Pattern Recognition and Machine Learning (Information Science and Statistics)[J]. 2006(4):049901.

[2] https://blog.csdn.net/Taily_Duan/article/details/51506776

[3] Lecture 07 Non-parametric K-Nearest Neighbors Algorithm

[4] https://github.com/FoundationsMachineLearning-Fa18/homework-02-Tianyu-Zhou