Red Cars detection in a High-Resolution Satellite Imagery using K-Nearest Neighbors (KNN) Approach (October 2018)

Keerthana Bhuthala, *Master's in Computer Engineering, University of Florida*

***Abstract*— The KNN algorithm with the distance metric as Euclidean distance is used to find the Red cars in a High-resolution Satellite Imagery. The KNN approach is extremely easy to implement and since the algorithm requires no training before making predictions, new data can be added seamlessly. The RGB(Red Green Blue) values of the pixels are taken as the training data and distance between the RGB values of the training data and test data is calculated to make predictions. The K nearest points are selected where K is an Integer. The prediction is made such that the data point belongs to the class to which most of the K data points belong. In this way, the test data points are grouped to 'Red car' class and 'Not a Red car' class. The locations of the corresponding RGB values that belong to 'Red car' class are easily obtained. Determining K values plays a vital role in getting satisfactory results for the given data. After conducting several experiments, by varying train and test data size and varying K, it is observed that the KNN algorithm has a high prediction cost for the large data sets.**

***Index Terms*—KNN, Euclidean distance, RGB values, Prediction, Varying K, Prediction cost**

I. Introduction

FEATURE extraction and classification are the part of Image Processing system which have various applications in many fields including Engineering, Medicine and Science. Color matching system is one of the applications that can be used for several industries. In this paper, the implementation of the K-Nearest Neighbors (KNN) Algorithm to find the red color cars in a high-resolution satellite imagery is described. The KNN Algorithm is one among the supervised machine learning algorithms. KNN can perform quite complex classification tasks. As it does not have a specialized training phase, it is considered as the lazy learning algorithm. It uses all the data for training while classifying a new data point or instance.

KNN is a non-parametric learning algorithm. It is a robust and versatile classifier and is used as a benchmark for more complex classifiers such as Artificial Neural Networks and Support Vector Machines. Despite its simplicity, the KNN algorithm can outperform more powerful classifiers and is used in various applications such as economic forecasting, data compression and genetics.

In this project, the RGB values of each pixel in the high-resolution satellite imagery is taken as the training data. Based on the Ground Truth provided, each pixel is classified into two labels – one representing the red car, the other not a red car. Training and Cross Validation are performed and the KNN Algorithm is used to predict the class/label of each pixel in the test data. Thereby, the locations of the red cars in the high-resolution satellite imagery are found.

II. Implementation

In this project, the ground-truthed imagery that indicates the locations of the red cars (data_train.npy and ground_truth.npy) and some imagery that does not have corresponding ground-truth (data_test.npy) are provided. Basically, the provided data is pre-processed to obtain the training data and the corresponding target labels.

A. Generate training data and validation data

The RGB values of each pixel is extracted from the data_train.npy file and stored as the training data. To obtain the labels of the corresponding pixel, a new array of the size same as the data_train is created and all the pixels that are present in the ground_truth are grouped to class label 1. The remaining pixels are grouped to class label 0. Hence, the pixel with class label 1 represents the location of the red car in the given ground_truthed satellite imagery.

B. Train and Cross Validation

The data obtained from the input – the modeling/training variables and target variables are split into training and test(validation) sets. The KNN algorithm is used to fit the model using the training data and the target values. The intuition behind the KNN algorithm is one of the simplest of all the supervised machine learning algorithms. The distance of the new data point to all the other training data points is calculated. The distance metric is chosen as per the data set. A popular choice is the Euclidean distance, but the other measures can be more suitable as per the requirement and include Manhattan, Hamming distance and Chebyshev. In this project, the distance metric used is the Euclidean. So, Euclidean distance between the RGB values of the validation data set and the RGB values of all the training data set is measured. Then, the K-nearest data

Keerthana Bhuthala is with the Department of Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611, USA (Phone: 352-667-9054; email: kbhuthala@ufl.edu).

points are selected where K can be any integer. Finally, the data point is assigned to the class to which most of the K data points belong.

The new data point of the validation set is classified to either label 0 (not a Red car) or label 1 (Red car). Suppose, the value of K is 3, the KNN Algorithm starts by calculating the distance of the new data point from all the other points. It finds the 3 nearest points with least distance to the new data point.

In the final step of the KNN Algorithm, predictions are made on the test data. The new point is assigned to the class to which majority of the three nearest points belong. The accuracy is tested between the predicted label and an already existing label of the data point.

Cross Validation is used to estimate the test error associated with the learning method to evaluate its performance, or to select the appropriate level of flexibility. It is used to tune the parameter K. The training set and the validation set are used for the cross validation using different K. The average accuracy for different K is measured and hence the K with best accuracy is selected and used to determine the class label of the test data point from the test set.

C. Predictions for the Test data

The parameters that are learned from the training and cross validation are used to provide the predictions for the test data. The optimistic K value which is obtained from the training is used to fit the Model and output the labels/classes of the test data accordingly.

III. Experiments

A. Construction of Training, Validation and Test data Sets

The training data is basically the RGB values of the pixels of the high-resolution satellite imagery. As mentioned in the Implementation section above, the labels corresponding to these pixels are stored in a separate array. The labels are provided based on the Ground truth imagery provided. Additionally, the X and Y coordinates in the Ground truth which does not represent the red color is eliminated. The RGB values of these co-ordinates is not added in the training data.

The Validation data set is constructed by splitting the training data into two parts (test_size is 0.4). This is carefully determined, such that there are red cars (label 1) available in both the training set and the validation set.

The RGB values of the pixels of the high-resolution test imagery is provided as the Test data Set. The predictions of the KNN classifier are stored to determine the location of the Red cars.

B. Testing

As the input data provided is very large, the subset of the training data is used for testing. The subset is chosen such that it has the RGB values that correspond to both the labels, 0 and 1 representing not a red car and red car respectively. The labels of the corresponding pixels are provided as the target vectors.

For example, the X and Y coordinates are chosen from the subset (850, 970) and (4517,4985) respectively such that a few of the coordinates would match the coordinates given in the ground truth. The labels of the corresponding coordinates are obtained.

This subset of the training data is split into training data and validation data. The split is determined such that the labels 0 and 1 are available in both the training and validation sets. After predicting using the KNN, the RGB values of the pixels are checked to see if its really a red car or not that is the results are verified to see if it’s grouped into a correct class are not. The accuracy when the test train split is with the test data as 0.4 is 0.9998575498575498. The accuracy is checked with the different test data values, but it is always ensured that the training and validation sets has the label values that belong to both 0 and 1.

The cross validation is performed by using different values of K and the accuracy with the varying K values is plotted. Since there are a very few pixels that belong to class 1 when compared to class 0, the accuracy is more for less value of K. More accuracy is noticed for the K values 1 and 3.

Hence, the K value is chosen between 1 and 3. The K value in this project is 1 and it is passed from train.py file to test.py file. The test data is split into subsets for testing. For example, the X and Y coordinates are chosen from the subset (50, 170) and (1517,1985) respectively. The RGB values corresponding to these X and Y coordinates are considered as the test data.

With value of K as 1, the labels for the subset of the test data is predicted. The X and Y co-ordinates corresponding to these class labels is retrieved which thereby denotes the locations of the red cars in the given test high-resolution imagery. The predicted values are verified by obtaining the corresponding RGB values. It is ensured that the subset of the test data used for the testing has the Red cars and not Red cars. Only then, we can determine that the predictions are correctly made. For the above subset of the test data provided, 7 red cars are predicted. The red car locations and their corresponding RGB values are given in Table 1.

TABLE I
Red Car Locations in the Test Data

S.No	Location (X,Y)	RGB values
1	(50,1533)	(127,96,103)
2	(50,1534)	(91,37,56)
3	(50,1535)	(115,44,77)
4	(50,1536)	(122,44,78)
5	(50,1537)	(122,45,74)
6	(50,1538)	(138,74,96)
7	(50,1539)	(165,128,138)

C. Advantages in this approach

As you can already tell from the previous section, one of the most attractive features of the K-neighbor algorithm is that it is simple to understand and easy to implement. Furthermore, the non-parametric nature of KNN gives it an edge in certain settings where the data may be highly unusual. Compared to other algorithms, KNN is robust to noisy data.

D. Disadvantages in this approach

One of the major drawbacks of the KNN algorithm is that the computationally expensive testing phase since we need to compute distance of each query instance to all the training samples. The parameter K must be determined correctly to get the accurate results.

E. Improvements to this approach

One improvement that can be made to this approach is that, the RGB values can be converted to Lab values. Lab values are usually defined by the lightness and the color-opponent dimensions a and b which are based on the compressed Xyz color space coordinates. By calculating the Euclidean distances of the Lab values, the accuracy can still ne improved and hence the better results are obtained.

IV. Conclusions

KNN is a simple yet powerful classification algorithm. It requires no training for making predictions, which is typically the most difficult parts of the machine learning algorithm. Using KNN algorithm with distance metric as Euclidean to find the red cars in the high-resolution satellite image gave satisfactory results. But, the computational cost is very high and by varying the test and training size along with the K value, the results vary drastically in this approach as the number of labels of the input training data set has a huge difference.

Further, the results can still be improved by implementing the extension of KNN algorithm. To improve the accuracy and to avoid the computation cost, a different distance metric can be chosen or the extension of the KNN algorithm such as Fuzzy KNN Algorithm can be used.

References

[1] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," IEEE Trans. Inform. Theory, vol. IT-13, pp. 21-27, Jan. 1967. W.-K. Chen, *Linear Networks and Systems.* Belmont, CA, USA: Wadsworth, 1993, pp. 123–135.

[2] K. Fukunaga and L. D. Hostetler, "ΛΓ-nearest neighbor Bayes risk estimation," IEEE Trans. Inform. Theory, vol. IT-21, no. 3, pp. 285-293, 1975. E. P. Wigner, “Theory of traveling-wave optical laser,” *Phys. Rev.*, vol. 134, pp. A635–A646, Dec. 1965.

[3] T. M. Cover, "Estimates by the nearest neighbor rule," IEEE Trans. Inform. Theory, vol. IT-14, no. 1, pp. 50-55, 1968.

[4] B. V. Dasarathy, "Visiting nearest neighbor—A survey of nearest neighbor classification techniques," in Proc. Int. Conf. Cybern. Soc, 1977, pp. 630-636.

[5] A. N. Chadha, M. A. Zaveri and J. N. Sarvaiya, “ Optimal feature Extraction and Selection Technigues for Speech Processing: A Review,” International Conference on Communication and Signal Processing (ICCSP), IEEE. Melmaruvathur. India, pp. 1669-1673, April 2016.