Detection of Vehicles From High Resolution Imagery Data Using K-Nearest Neighbors Classifier

Kiana Alikhademi

***Abstract*—Vehicle identification is an important and challenging task in Intelligent Transportation System, which could be used for many different applications. The problem of detecting red cars based on the color in the high resolution imagery data has been studied in this paper. This problem is one of the example of imbalanced learning in which the number of points in the majority class is more than that of the minority class. The image data has been processed to form our ground truth data by labeling the center of red car coordinates and eventually the pixels which formed the red car pictures. By considering simplicity and the ability to apply different distance metrics, the KNN classifier has been chosen to detect red cars in the image by using the color distance metric between two distinct pixels Red Green Blue(RGB) values [2]. Using the validation data, our method returned a precision score of 70%, recall of 71%, an F-measure of 0.7, and an accuracy score of 70%. These results show that our method is able to identify red cars in the test data. Major findings from the experiment I ran is summarized in the discussion section. The most influential factor in the KNN models performance could be related to the the imbalanced nature of data.**

I. Introduction

MANY civilian or military applications are depending on vehicle identification, including, but not limited to, highway traffic surveillance control [14]. Intelligent Transportation System (ITS)is significantly depending on automated vehicle identification techniques which could be used for different purposes such as vehicle counts, tracking and traffic analysis [8], [9]. Software based classification system have substituted the hardware ones due to the real-time feedback and better performance by looking at high resolutions image or videos [9]. In this project, our objective is to detect red cars in a high resolution satellite image. Among the methods we could use there are two different learning methods suitable for this job, namely, K-Nearest Neighbor(KNN) and K-Means algorithm. Their advantages and disadvantages will be discussed first, then the rationale for picking KNN over K-Means will be given, and lastly the literature review will be briefly discussed.

The K-Means method is an iterative algorithm that updates the clustering of different features, which in our case is the clustering of data based on RGB color combination. However, in this case, the number of pixels labeled as red is very small in compare with other colors in the background. Therefore, it is very unlikely that K-Means method ends having a cluster around red cars accurately. This is due to the fact that most of the time K-Means produce clusters of uniform size despite the skewed nature of the given data. This could affect the results significantly while it could be neglected from the developer side.

On the other hand, KNN makes no assumptions about the functional form of the problem being solved and it is very easy to implement and understand. Like the K-Means algorithm, it works very well when data is not skewed toward any specific classes' distribution. However, it is challenging to apply KNN to large data sets due to huge time complexity of comparing test instances with the whole training data. In addition, when we are dealing with imbalanced data like ours, the result of using this algorithm is very biased toward the majority class. Therefore, the majority class is most of the time winner in the major voting process. More over, KNN, as an instance-based supervised learning method; therefore, instead of learning a model explicitly, it would memorize all samples in the training data and then compare it with instances of test data to predict the classes to which each instance belongs. This memorization aspect categorized KNN as a memory-based learning algorithm, which is the influential factor in the increase of time complexity as the size of training data increases [4].As Frigui et al. [7] pointed out KNN classifiers are "appealing because of their simplicity, ability to model non parametric distributions, and theoretical optimality as the size of the training data goes to infinity". However, The main drawback of KNN is low efficiency with respect to time and dependency on the value of k.

By considering all the advantages and disadvantages of KNN and K-Means, I consider to proceed with KNN instead of K-Means due to couple of reasons. First, many research suggested that KNN is a good initial step to obtain some preliminary knowledge about your data and problem before switching to more complex ones [4], [5]. Second, in a vehicle detection problem it would be beneficial to use a non-parametric method to avoid misleading assumptions and priors about the imbalanced data distributions. Lastly, as Frigui et al [7] suggested KNN results are easy to interpret which makes it a viable solution when the data and problem is complex in nature. In this project, as image data was imbalanced and complex it is important to choose a method to interpret the results easily. Sarikan et al. [9] developed an automated vehicle classification to detect the vehicles from the highway image sample. They computed the heat map for different type of vehicles. These heat maps would be tested with the test image to measure the similarity. I did not follow their method for couple of reasons. First, they used OpenCV mainly for the processing of image which we could not use for this project. Second, they are not considering the color as a feature for classification. However, KNN method and its parameters tuning process is the same between their work and mine. Frigui and Gader [7] developed a probabilistic KNN method to assign the confidence with the goal to detect the land mines from

the sensory data generated by the ground-penetrating radar. I picked KNN as it would be more manageable for me in the context of this project. I have implemented the conventional KNN, which has been introduced in [15].
The rest of this paper is organized as follow: Section II discusses the details of the experiment, such as processing the initial raw imagery data, parameter tuning, cross validation and results of our experiment. Discussion provides the findings and describes what could be done to improve this method potentially. Finally, conclusions will be summarized in the Section IV.

II. Implementation

Our procedure to detect the red cars in the image consists of multiple stages which is described in details.

A. *Processing Data*

Like any other supervised classification task, the first stage is to process the raw data and make it ready to be used as a basis for the training. As the major part of the initial pictures are background objects, certainly it is not time efficient to use the whole high resolution image data for the fitting purposes of KNN. While reducing the image's dimension helps to reduce the time and cost of applying the KNN algorithm, it is still not helpful. Therefore, it is vital to come up with a method to generate numbers of random points from the raw imagery data. As the main goal of this project is to detect red cars, I pick these random training points from a piece of picture including high density of red cars. Aside from the coordinate, the RGB values of each pixel would be stored, which will be used later to evaluate the color similarity or distance between two distinct pixels.
One way to deal with imbalanced data is to use the concept of over sampling. The ground truth file provided in the discussion by Bo Hu, were used initially to find the centers of all the red cars. Each car in ground truth is represented as only one pixel. But, in reality each red car consists of several red pixels in its neighborhood. That is main reason the ground truth data is expanded by adding a window of 25 pixels around each red car and tag all of these pixels as red. Despite the expansion of the ground truth for red cars, the number of red pixels in training set is still minimal. If we randomly sample through the original image to divide into training and validation points, there is a very high chance that we are not including sufficient red cars in the data and this is not helping to reach the goal of this project. Therefore, we gather random points and random rows of ground truth data to form the training and validation data. We ended up into 4000 and 1200 points in total from both red cars and background for the training and validation set, respectively.
The dimension of the test data is also a problem which needs to be fixed before training the model. Therefore, I do the same procedure as training set to generate random points from the test data with only difference. Here, the ground truth underlying the test data is not given. Therefore, random ones are generated without knowing if they are red cars or not. The output of my KNN model would be the car coordinates and the labels corresponding to it.
The oversampling of training and validation has been explained. However, the rationale for picking this method of oversampling was not explained. To validate my rationale, I ran an experiment that what happens if the points are coming always from random distribution. The accuracy was low around 0.17 and almost all of the time the validation instances were labeled as 1, or non red car. This result give me the intuition that KNN would not be able to predict the test data without looking through sufficient red cars in the training data. Therefore I pre process the data in a way so that there are sufficient number of red cars in each validation and training set.

B. *KNN Approach*

KNN method was discussed properly in the introduction by covering all of its advantages and disadvantages. The details of our KNN implementation would be discussed here to help others in replicating it. The KNN model which it was developed for the second homework in Fundamentals of Machine Learning course has been used. The main reason for using that implementation over the one from Sci-kit Learn library was to get better control over the problem and related metrics. We previously mentioned that KNN is a non-parametric approach which computes the similarity between the points of test and training data. Therefore, we need to talk that how this similarity will be measured. There are many different distance metrics which have been used thorough the literature including but not limited to Manhattan, Mahalanobis and Euclidean. The distance metric in KNN approach should be selected based on the application. In this particular problem, there is no difference between different metrics since one only needs to measure the similarity between RGB values of pixels. However, Aggrawal et al. [18] suggested to use Manhattan distance over the Euclidean when you are working with high dimensional data. Therefore, Manhattan distance pixel wise was chosen to use on the training and test data set. According to the he Manhattan distance between two pixels of "M" and "P" is defined in Equation 1. The output of the KNN approach would be coordinates of the red cars.

$$d(M,P) \equiv |M_x - P_x| + |M_y - P_y| \qquad (1)$$

C. *Parameter Tuning*

The next step after making the test and training data ready is to tune the parameters for our KNN model. As we previously mentioned, KNN does not have a huge training phase and it is only computing the distance between test instances and training data to compute the majority vote which would be the final prediction. In order to obtain better results for KNN, we need to tune the "k" parameter based on the data complexity and number of classes. In KNN classifier, a small "k" might lead to an over-fitting result, while a large k might lead to under-fitting. Therefore, we choose the parameter k through cross validation technique. I use cross validation technique to increase "k" over a specified range and compute the classification accuracy with respect to each ”k”. The plot from this phase has been shown in Figure 1. As you could see in the figure the optimum k for our data

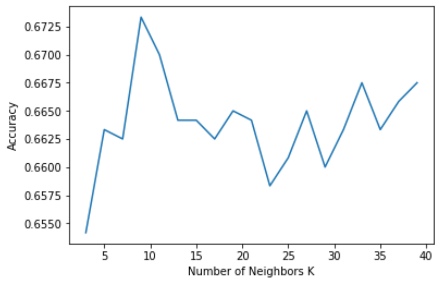

Fig. 1. Accuracy over range of different K values

would be k = 9 as there is a peak in the accuracy. The best parameter k would be used to predict the red cars on the test data.
As the random splitting of the data based on the indices does not seem efficient for this problem, we did not use K-Fold cross validation to test different folds of data as training or validation. Kfold and train-test-split gives the random split over the training data which may not be the best approach in this problem. As Zakka [5] suggested the value of ”k” gives us some control about the decision boundary in the process of prediction. You could see that increasing k from small to large could have different outcomes in case of accuracy. Once you go with small values of k, the noise will have a higher influence on the classification result. On the other side, large values are computationally expensive as KNN is a non-parametric and instance-based approach. Parameter k could be considered as a tool which helps us in focusing on a part of the sample data points to decide what is the label of the unknown input (Test data). Figure 1 would help us to visualize the impact of k’s range of values on the classification results. If k is too small (such as k=1), we are not obtaining really that much information from the whole distribution which could be helpful in getting better results. On the other hand, if k is too large we are including more neighbors in our labeling process which they could be actually outliers. The large values of k introduce biases into the problem which may affect the classification results significantly. As Figure 1 shown , by increasing k from small to large the results are getting better most of the times until we reach the optimum k.
After finding the optimum k for our problem we would run KNN algorithm on validation data to compute the most well-known accuracy metrics. The results along with the findings from our experiment would be explained in the next section.

III. Experiment

A. Results

Computing the accuracy does not suffice to interpret your predictive model’s strength especially when you are dealing with imbalanced data. If you are dealing with imbalanced data, like in ours, it would be the best to just ignore couple of red car’s points and predict all the points as background which could give you a very high accuracy. However, this is not aligned with our goal of the project which is detecting the red cars on the image. Therefore, we incorporate other metrics to evaluate our model’s performance with respect to both positive and negative cases. In Figure 2True Positive (TP), False Negative (FN), False Positive (FP) and True Negative (TN) are visualized. This shows that our model is doing reasonably well in predicting the positive and negative cases with accuracy of 70%. However, it definitely could be improved. The values of precision, recall and F-1 score has been shown in Table I.

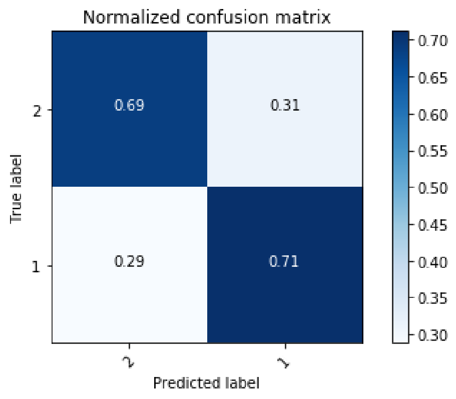

Fig. 2. Confusion Matrix for visualizing the classification metrics

TABLE I
Classification metrics

metric	Precision	Recall	F1-Score
value	0.7	0.71	0.7

B. Discussion

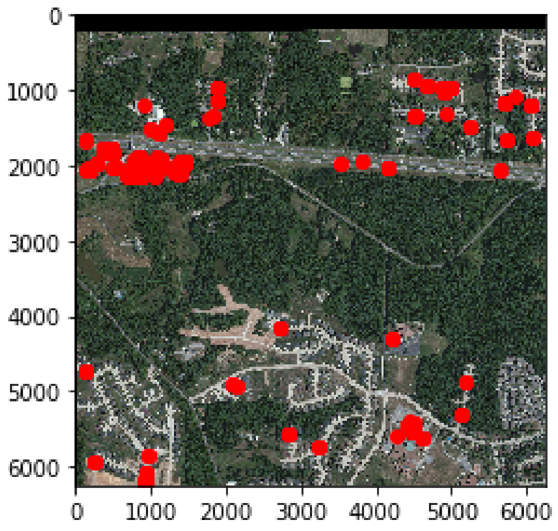

Fig. 3. Mapping the predicted coordinates

One of the goal of our experiment was to evaluate the impact of imbalanced data on the predictive model(KNN). As shown in Table I and Figure 2, the same number of points from the background and red cars led to almost same functionality with respect to the positive and negative cases. According to Table 2, the KNN model is able to classify most of the red points but there are around 30% points which were not classified correctly. There are multiple factors which could have led to this problem. First, we should not forget that our data is really skewed toward the background so just simple sampling through the training and validation is not solving the problem completely although it makes it better. It was found that there are more complex oversampling which might help regarding this problem such as SMOTE [16]. Second, it would be beneficial to have more features which you could decide if it is car or not and then compare the RGB values to classify it as red car. This would help with the cases that RGB value is so similar to the red cars but it is not red car actually. The coordinates which were predicted as red cars are shown in Figure 3 with the red colors. When the classes' distribution are skewed, there is a high chance that points from the minority class does not even enter the majority voting process. This problem exists when samples of the data were selected randomly. Samory Kpotufe et al.[17]introduced an approach named Gradient Weighting, which took two-pass directional derivatives on the data to gives the weight which could outperform the regression or classification models' performance. Due to time constraints, this method was not implemented, but it could definitely be considered as an improvement to our work..

Time complexity is always a challenge for KNN. The time will grow exponentially as the number of pixels involved in the experiment increases. This could be related to the fact that for each test instance we are comparing it with all the training dataset no matter they are related or not. To cope with time limitation one needs to generate limited number of random points for both validation and training data. In addition, As Zakka [5] suggested using efficient data structures such as K-D tree may help in boosting the time complexity and eventually the processing of a broader range of data. While we applied the first solution, we were not able to use more efficient data structure in this project.

Another finding from this experiment was related to the distance function. The distance metric should incorporate all the features of data and it should seem reasonable for the specific application. For instance, we tried Manhattan and euclidean distance between the points but it did not seem reasonable as the distance of two points in different part of picture does not necessarily mean that they are representing the same object or they have the same color. Therefore, we switched to consider the Manhattan distance of RGB values to find the points in training data which have closer color pattern to the test instance. This shift in the distance metric changed our accuracy significantly from 0.17 to 0.65.

Space and time complexity could be considered as the drawbacks if desiring to use the whole image data. Two different ways to overcome memory limitation and time complexity is to use a small randomly generated data training and enlarge the small scale set of red cars which is known as oversampling. Finally, using a more powerful method may have helped to detect the red cars more accurately in both validation and test set.

IV. Conclusion

In this project, we focused on the task of red car detection in the raw high resolution satellite image data. Although there are many promising solutions to solve this problem, KNN method which is a non-parametric and instance based approach has been used. The main rationale for choosing the KNN over others would be the ease of the interpreting the results. The classification accuracy of our method was around 70% which shows high number of true positive and low number of false negative. Our performance metrics showed that there is definitely way to improve the system. Therefore, the major findings from our experiment has been discussed to interpret the possible future directions for this work. For future work, it definitely consists of using a pre processing method to distinguish the different objects in the image. Moreover, it would be beneficial to come up with a more suitable method for oversampling the data by considering the imbalanced nature of it. Th findings of this experiment provides a proper baseline to move toward a better solution for vehicle identification using high resolution image data.

Notes

I discussed my original thought with Diandra Prioleau to clarify my confusions. However, I guarantee that all the works in this report and coding is %100 my own personal work without any help.

References

[1] Hu, L.Y., Huang, M.W., Ke, S.W. and Tsai, C.F., 2016. The distance function effect on k-nearest neighbor classification for medical datasets. SpringerPlus, 5(1), p.1304.

[2] Color Difference between 2 colors using Python, Hanzra Tech. [Online]. Available: http://hanzratech.in/2015/01/16/color-difference-between-2-colors-using-python.html. [Accessed: 05-Oct-2018].

[3] Subset K-Means Approach for Handling In-balanced Distributed Data

[4] Deekshatulu, B.L. and Chandra, P., 2013. Classification of heart disease using k-nearest neighbor and genetic algorithm. Procedia Technology, 10, pp.85-94.

[5] A Complete Guide to K-Nearest-Neighbors with Applications in Python and R. [Online]. Available: https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/. [Accessed: 08-Oct-2018].

[6] 8 Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset. [Online]. https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/ [Accessed: 08-Oct-2018].

[7] Frigui, H. and Gader, P., 2009. Detection and discrimination of land mines in ground-penetrating radar based on edge histogram descriptors and a possibilistic k-nearest neighbor classifier. IEEE Transactions on Fuzzy Systems, 17(1), pp.185-199.

[8] El Hamra, W. and Attallah, Y., 2011. The role of vehicles identification techniques in transportation planningModeling concept. Alexandria Engineering Journal, 50(4), pp.391-398.

[9] Sarikan, S.S., Ozbayoglu, A.M. and Zilci, O., 2017. Automated Vehicle Classification with Image Processing and Computational Intelligence. Procedia Computer Science, 114, pp.515-522.

[10] Melgani, F. and Bruzzone, L., 2004. Classification of hyperspectral remote sensing images with support vector machines. IEEE Transactions on geoscience and remote sensing, 42(8), pp.1778-1790.

[11] Keller, J.M., Gray, M.R. and Givens, J.A., 1985. A fuzzy k-nearest neighbor algorithm. IEEE transactions on systems, man, and cybernetics, (4), pp.580-585.

[12] Karaimer, H.C., narolu, . and Batanlar, Y., 2015. Combining shape-based and gradient-based classifiers for vehicle classification.

[13] Khairdoost, N., Pour, M.R.B., Mohammadi, S.A. and Jajarm, M.H., 2015. A ROBUST GA/KNN BASED HYPOTHESIS VERIFICATION SYSTEM FOR VEHICLE DETECTION. International Journal of Artificial Intelligence & Applications, 6(2), p.21.

[14] Hadi, R.A., Sulong, G. and George, L.E., 2014. Vehicle detection and tracking techniques: a concise review. arXiv preprint arXiv:1410.5894.

[15] Cover, T. and Hart, P., 1967. Nearest neighbor pattern classification. IEEE transactions on information theory, 13(1), pp.21-27.

[16] https://www.kaggle.com/qianchao/smote-with-imbalance-data

[17] Kpotufe, S., Boularias, A., Schultz, T. and Kim, K., 2016. Gradients weights improve regression and classification. The Journal of Machine Learning Research, 17(1), pp.672-705.

[18] Aggarwal, C.C., Hinneburg, A. and Keim, D.A., 2001, January. On the surprising behavior of distance metrics in high dimensional space. In International conference on database theory (pp. 420-434). Springer, Berlin, Heidelberg.