IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

VEHICLE DETECTION AND TRACKING USING MACHINE LEARNING TECHNIQUES

Chirag
Student
Indira Gandhi University , Rewari

ABSTRACT

This master's thesis focuses on vehicle detection and tracking. The research tries to detect vehicles in images and videos. It deploys a dataset from Udacity in order to train the algorithms. Two machine learning algorithms; Support Vector Machine (SVM) and Decision Tree have been developed for the detection and tracking tasks. Python programming language have been utilized as the development language for the creation and training of both models. These two algorithms have been developed, trained, tested, and compared to each other to specify the weaknesses and strengths of each of them, although to present and suggest the best model among these two. For the evaluation purpose multiple techniques are used in order to compare and identify the more accurate model. The primary goal and target of the thesis is to develop a system in which the system should be able to detect and track the vehicles automatically whether they are static or moving in images and videos.

Vehicle detection also called computer vision object recognition, basically the scientific methods and ways of how machines see rather than human eyes. The main duty of a vehicle detection system is to localize one or more vehicles in input images. The results showed that SVM outperformed the Decision Tree and has acceptable accuracy for the vehicle detection and tracking tasks.

Keywords: Vehicle detection; Vehicle Tracking; SVM; Decision Tree; Image detection Object ; detection and Tracking .

CHAPTER 1 INTRODUCTION

1.1 Background

Since the population and transport system increase day by day, the demand for managing them increase at the same time. The world is getting populated so fast. Therefore the number of machines from any types including vehicles increased at the same time. That being said, new topics like traffic, accidents and many more issues are needed to be managed. It is hard to manage them with the old methods, new trends and technologies have been found and invented to handle each and every milestone that human kind is trying achieve. One of these challenges is traffic in highways and cities. Many options like traffic light, sign, etc. deployed in order to deal with this phenomena. It seems that these options are not enough or not so efficient alone. New technologies like object detection and tracking are invented in order to utilize automated camera surveillance to produce data that can give meanings for a decision making process. This phenomena have been used for different kind of issues. The new trend Intelligent Transport System (ITS) has many elements which object detection and tracking is one of them. This system is used to detect vehicles, lanes, traffic sign, or vehicle make detection. The vehicle detection and classify ability gives us the possibility to improve the traffic flows and roads, prevent accidents, and registering traffic crimes and violations.

Humans can easily recognize vehicles in videos or images or to identify different types of cars. In computer algorithms and programs it is highly depend on the types of data. Some challenges like the weather or light are also plays important role on making the process easy or much hard. At the same time we have different types and shapes of vehicles. More than that the new challenge could be to identify moving objects in a video in real time where they are different in size and shape.

There are different techniques and methods for vehicle detection and classification. The variety of these techniques are in types of algorithms like Support Vector Machine (SVM), Convolutional Neural Network (CNN), Decision Tree, Recurrent Neural Network (RNN) etc. The field is constantly evolving since the industry is focused on this system or Computer visionary. In this thesis we investigate two algorithms SVM and Decision Tree to identify how they can apply in the field and which one works better than the other.

1.2 The Problem

Object detection attracted the attention in research industry lately. Researchers are trying to explore the topic to reach to an accepted accuracy level. Machine learning is used to detect objects. There are many techniques doing this job but in order to identify the best model among the suggested models, this thesis is going to explore the topic on how to detect the objects while at the same time compare two suggested models and suggest the best one which yields highest accuracy and performance.

The task of detecting and classifying objects in images and videos is suited well in machine learning since the task is a classification task. The reason behind this is from the dataset with complex features. The system is important for many fields especially for traffic and vehicle detection.

The training and analysis was done using a rich dataset which is described in details in chapter 4.

1.3 Aim of the Study

The aim of this thesis is to develop two algorithms, SVM and Decision Tree to detect vehicles in images and videos. It compares the two algorithm with the same dataset and preprocessing methods. Finally suggest the method with high accuracy and performance level. The thesis is going to implement two classifier which are able to predict the class of the image whether its Vehicles or Non-Vehicles. The vehicle detector will also predict the bounding box coordinates of the vehicles.

1.4 Significance of the Study

Since the industrial revolution the number of cars increase day by day. One of the new challenge of the world is the traffic. People in cities wastes most of their time in traffic going somewhere in the city. Having a digitized traffic system which is functioning 24/7 and make the tasks so easy and efficient is crucial for all countries around the globe. Therefore having a computerized traffic system cannot be handled without having an accurate vehicle detection system. The system obviously affects the economy, the life of citizens, the industry, etc.

That being said, it is necessary to contribute in the topic until we reach to a level which the algorithms can accurately detect all kind of objects or vehicles whether they are in images or videos through pipelines.

1.5 The Limitations of the Study

Despite the fact that the thesis reached to its goals, it would have been more accurate if the dataset is improved and the number of objects in vehicles and non-vehicles class increased. At the same time it would be more complete and proper claim if the comparison would have conducted between many more options in the algorithms which are used in computer visionary problems.

1.6 Overview of the Study

The thesis is consist of six chapters in all:

Chapter 1: gives a general introduction and background about the topic. This chapter describes the aim of the thesis, its importance and overview of the research.

Chapter 2: past researches which are related to the topic are reviewed, their techniques and results are also discussed.

Chapter 3: this chapter is focused on machine learning techniques, theory, formula, implementation, and philosophy.

Chapter 4: ML algorithms and models used in this thesis will be explained in details with their backend formulas.
Chapter 5: the result and outcome is presented.

Chapter 6: the conclusion is presented with a general view and future recommendations for research topic improvement.

CHAPTER 2 LITERATURE REVIEW

In the past few decades researchers had a great interest in vehicle detection and tracking. The topic attracted the attention quit much. Different sensing modalities have been used for detecting the objects or specifically vehicles. These modalities are LIDAR, radar, and computer vision. The attraction caused by immense progress of image processing. The very first signs and models of image processing goes back to the 1960s' and 1070s', after that various methods and techniques have been invented and proposed (Chen, 2015). This chapter briefly discuss the recent related works by researchers regarding vehicle detection and tracking.

At the very beginning we can say that any approach in this topic are classified. At a glance we have four sections:

- Object recognition and identification from the appearance
- Classifying the object into one of the categories.
- Object detection or target detection
- Tracking the object or the target

Object detection presents some unique attributes of an object that a computer can identify distinctly from other objects. Meanwhile the object classification intend to identify the similarities of an object with an object category at all. From the object detection result, we can assign an object tracker, the tracker is following the target by re-detecting it in the sequence frame following the first point of the target.

The primary goal and target of the thesis is to develop a system in which the system should be able to detect and track the vehicles automatically whether they are static or moving in images and videos.

Vehicle detection also called computer vision object recognition, basically the scientific methods and ways of how machines see rather than human eyes. The main duty of a vehicle detection system is to localize one or more vehicles in input images. There are two methods in vehicle detection system: sliding window method (Yu & Shi, 2015) and local features method (Noh, Shim & Jeon, 2015). In the local features-based method, the system usually find the features of one object or the group of the objects in the very beginning. After that it tries to categories the founded features into different classes via classification models. This is usually the last step which the system decides which category is the object belongs to. It claims that the biggest strength of this method is that the geometry and features of the targets are known before the detection. While it can be a weakness or limitation as well because it can only detect pre-learned objects. The second method which is sliding window based system, works differently. It scans the input image with a number of windows with different sizes. After that it analyzes whether the target is in the window or not (Su, Sun, Liu, Zhang, & Yu, 2015).

A method proposed by Wang et al (Wang et al, 2014) for detecting objects using edge detection in UAV footage as demonstrated in Figure 2.1. Researcher presents an example of how to use edge detection to recognize artificial objects. An edge detection algorithm have been used in this method to identify the straight lines on a vehicle. Before doing so, the algorithm removes noises from the background via and by the help of higher threshold in the process.

Another method suggested by Sokalski et al (Sokalski et al, 2010) uses color feature and edge detection to separate and distinguish artificial and natural objects. But the difference is in extracting the nine features from the various channels of the color in the original image. Although these features are uses to specify the edges and changes to separate artificial and natural objects like illustrated in Figure 2.2.

The two mentioned researches makes the idea of summarizing clear. It summarizes local descriptors into texture features and color features. There are different cases. Color feature usually works fast in order to detect objects but in scenarios or somehow generally the accuracy is challenged. But texture feature works better with objects in images due to having more information about object in textures. Not to be forgotten both researches used unsupervised classification.

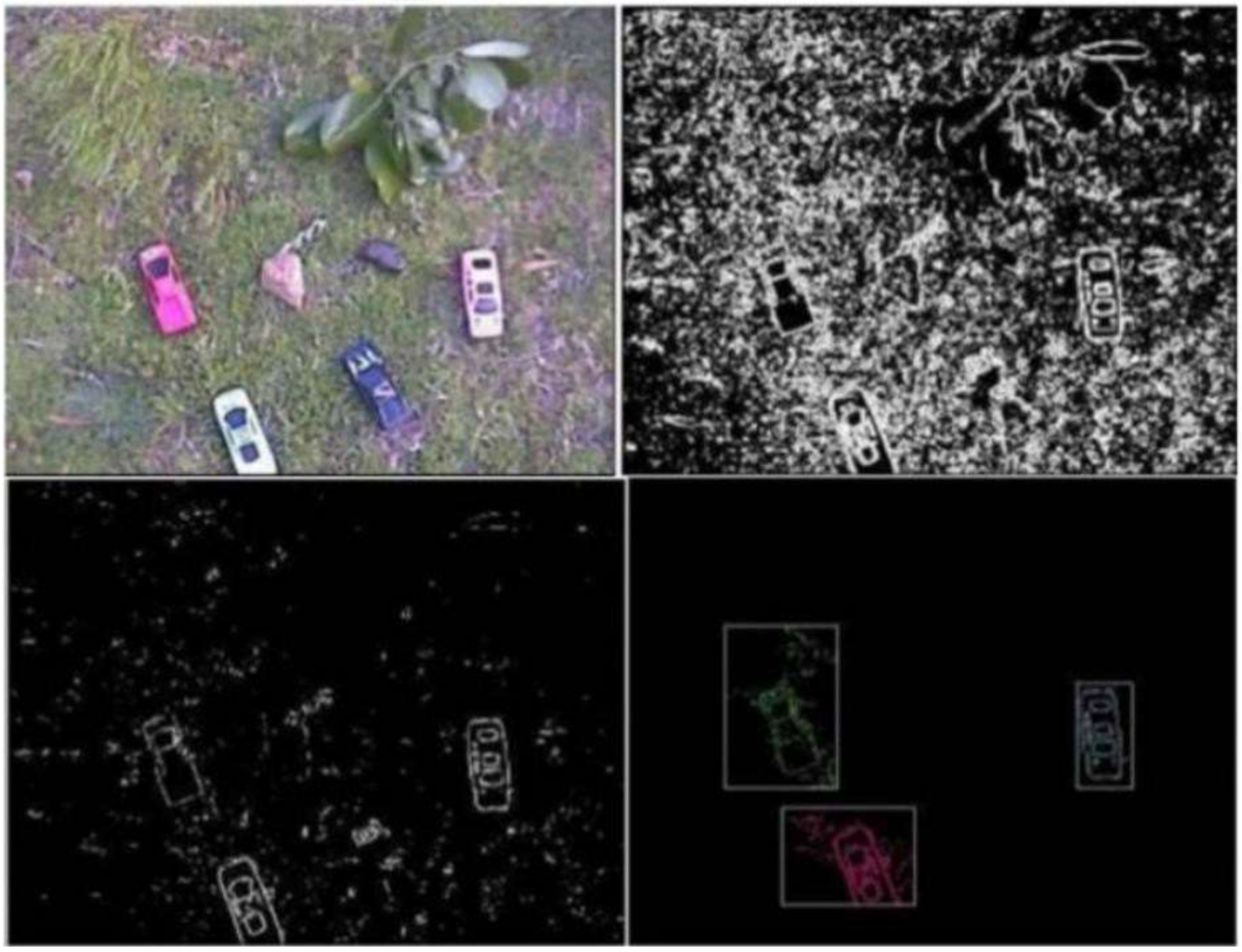


Figure 2.1: different thresholds using edge features (Wang et al, 2014)

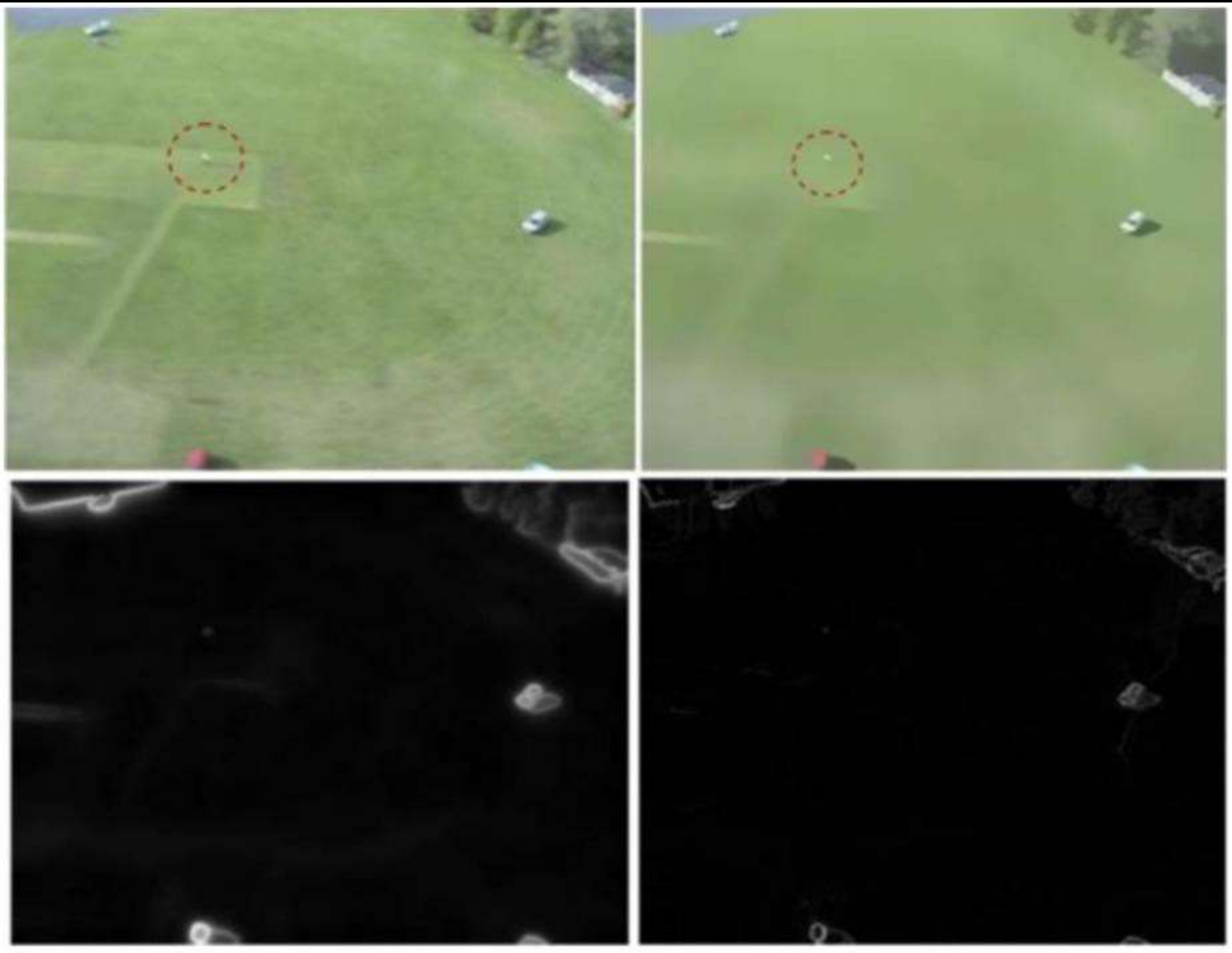


Figure 2.2: edge and color feature for artificial objects (Sokalski et al, 2010)

The two papers above use surface and shading highlights for item recognition, which is average for item identification. Utilizing shading highlights to distinguish objects is the most straightforward technique. N. Baha et al (Baha, 2014). displayed exact milestone acknowledgment investigations of utilizing shading and surface highlights. When tourist spots are perceived from a class of geo-referenced pictures, they can be utilized to gauge the UAV's position and help self-sufficient route. This paper utilizes one lot of ethereal geo-referenced pictures and another arrangement of aeronautical pictures of the equivalent area, gathered at an alternate time, which are not geo-referenced. This work exhibits a milestone acknowledgment framework dependent on the extraction of shading and surface highlights. The proposed strategy utilizes these highlights to give data about surface introduction, shape also, shading. This methodology is being contemplated for application in a PITER (Real-Time Image Preparing) investigate venture that is being done at the Institute for Advanced Studies (IEAv – Instituto de Estudos Avancados), and connected in self-ruling UAV route based on pictures. In this undertaking, 126 examples are utilized for administered realizing, which is a specific sort of AI calculation that permits the forecast of the class of a formerly obscure occurrence dependent on the learning of the class of a preparation test. A HSV shading what's more, Gray Level Co-event Matrix (GLCM) highlight is utilized to prepare the classifier the identification. The strategy utilizes a sliding-window approach in discovery. In this paper appears plainly that the mix of shading and surface highlights can improve the precision of discovery; the utilization of the Supervised Classification strategy can likewise improve location execution.

Saman G et.al (Kanistras et al, 2015) proposed a vehicle identification approach by the thickness estimation. The angle vectors have been determined in the edge guide of the elevated pictures. It is recommended that, the headings of the inclination vectors are changing fundamentally in the limit of the objective and by ascertaining the standard deviation of the slope vectors. So by predefine the edge, the vehicles can be recognized inside the esteem. The assessment utilized the airborne pictures taken from street in Turkey and accomplished 86% exactness in F-measure. Nonetheless, such discovery techniques without the preparation have a typical disadvantage, which the target objects are hard to recognize from the perplexing foundation circumstances.

Peng W et.al (Wei et al, 2015) proposed a vehicle recognition technique that improves foundation extractor. This methodology utilized asphalt division with 8-neighborhood filling to expelling the street markers so as to separating the unpredictable foundation. The outcome shows that the proposed strategy can stay away from the identification blunders brought about by the deceptive of the street markers. In any case, this identification strategy depends on fixed camera discovery, which is in opposition to the motivation behind this theory, however the foundation extractor system can be embraced.

J. Susaki et al. (Susaki, 2015). proposed a two-arrange way to deal with the programmed location of vehicles inside aeronautical symbolism. Their methodology depends on the utilization of different fell Haar classifiers (Viola et al, 2005) for vehicle arrangement and an auxiliary check organize that endeavors to dispose of non-vehicle hopefuls dependent on UAV elevation and vehicle measure limitations. The fell Haar classifier gives a solid locator that is invariant to vehicle shading, type also, setup. To accomplish vehicle introduction invariance, they utilize four separate fell Haar classifiers prepared in test vehicle pictures arranged into four positional introductions. The four classifiers are then assessed utilizing a question picture at various scales and positions utilizing a sliding window approach and numerous classifier location, and identification covers are settled utilizing a spatial combining strategy (Figure 2.3). J. Berni et al. (Berni et al, 2009) later broadened this work with the utilization of extra warm symbolism for warm mark affirmation, which improved execution extensively.

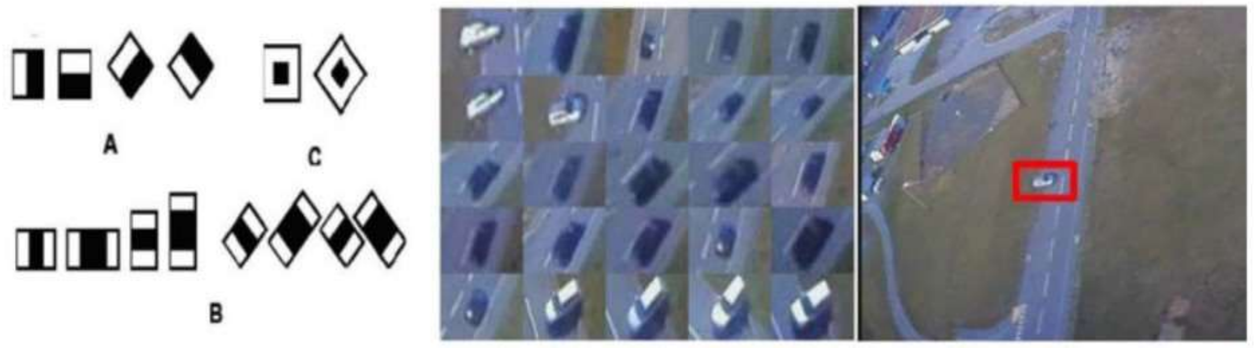


Figure 2.3: features, training samples and detection results (Susaki, 2015)

Gleason et al. (Chen et al, 2015) concentrated on programmed vehicle location in provincial conditions. Their approach comprises of a course location calculation, which is included two phases. In the first stage, a Harris corner indicator is utilized to distinguish highlights of enthusiasm for the pictures; the creators contend that "vehicles specifically have an expansive number of edges and corners looked at to characteristic articles". Next, a productive sliding window approach is utilized to decide districts with an element thickness higher than a foreordained edge. Covering areas are gathered and further refined dependent on the shading qualities of foundation zones. The areas chose in this stage are then put through picture grouping procedures in the second stage, which decide the nearness of a vehicle. These creators' exploration analyzed the execution of two picture fix descriptors, an altered Histogram of Oriented Angles (HoG) highlight and Histogram of Gabor Coefficients highlights. They too researched the execution of three factual grouping strategies; K-Nearest Neighbors (k-NN), Random Forests (RF) and Support Vector Machines (SVM). From the first phase of their calculation, they acquired a normal location rate of 85% and found the top performing classifier to be Random Forests utilizing Histogram of Gabor Coefficients highlights; this was equipped for characterizing 98.9% of vehicles and 61.9% of foundation pictures effectively.

Sebastien R et.al (Razakarivony, 2016) proposed a vehicle location technique in unconstrained conditions with cutting edge object recognition approaches, which is utilizing the sliding window characterization strategy with the SVM classifiers.

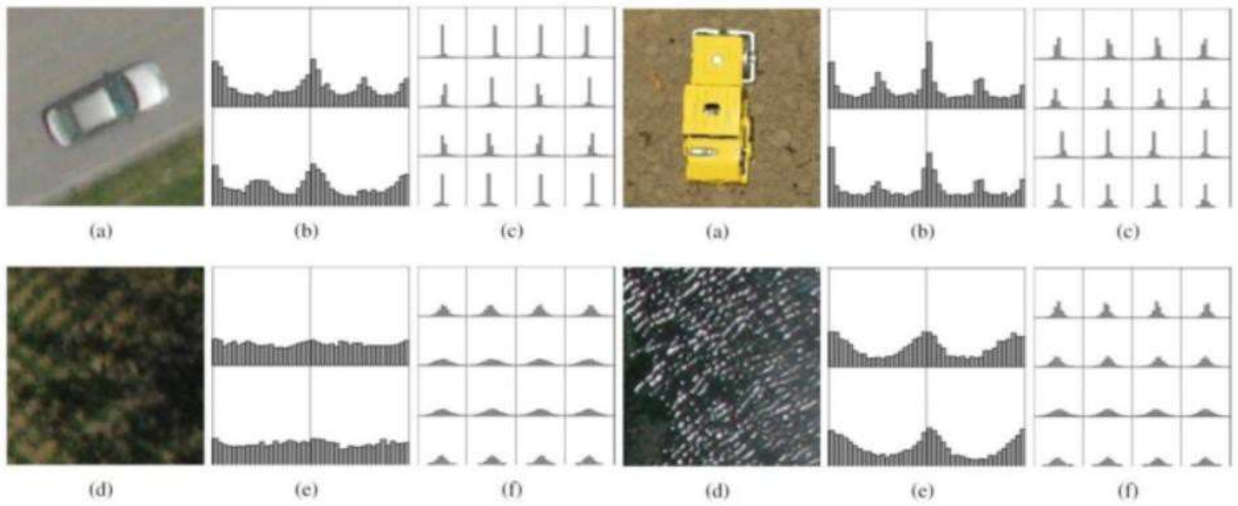


Figure 2.4: HoG features, vectors, histograms, background image (Chen et al, 2015)

Sahli et al. (Sahli et al, 2010) proposed a nearby component based methodology for programmed vehicle identification in low-goals elevated symbolism. Their methodology was created with the point of being free from the imperatives identified with location strategies dependent on a vehicle's visual appearance, for example a vehicle's rectangular shape and the nearness of frontal and additionally back windows. Their approach depends on the extraction of Scale-Invariant Feature Transform (SIFT) highlights from vehicle and foundation pictures. These highlights are utilized to prepare a Support Vector Machine (SVM) classifier to characterize a model that can be utilized to order SIFT highlights separated from the autos and foundation in a question picture. The accumulation of SIFT highlights that are anticipated to have a place with a vehicle are then grouped in the 2D picture space into subsets related with individual vehicles. The creators' bunching strategy depends on an altered Liking Propagation (AP) calculation that is bound by the spatial limitations identified with the geometry of vehicles at the given goals. They got a grouping exactness of 95.2% in aeronautical symbolism of a parking garage containing 105 vehicles, with no false-positive recognitions.

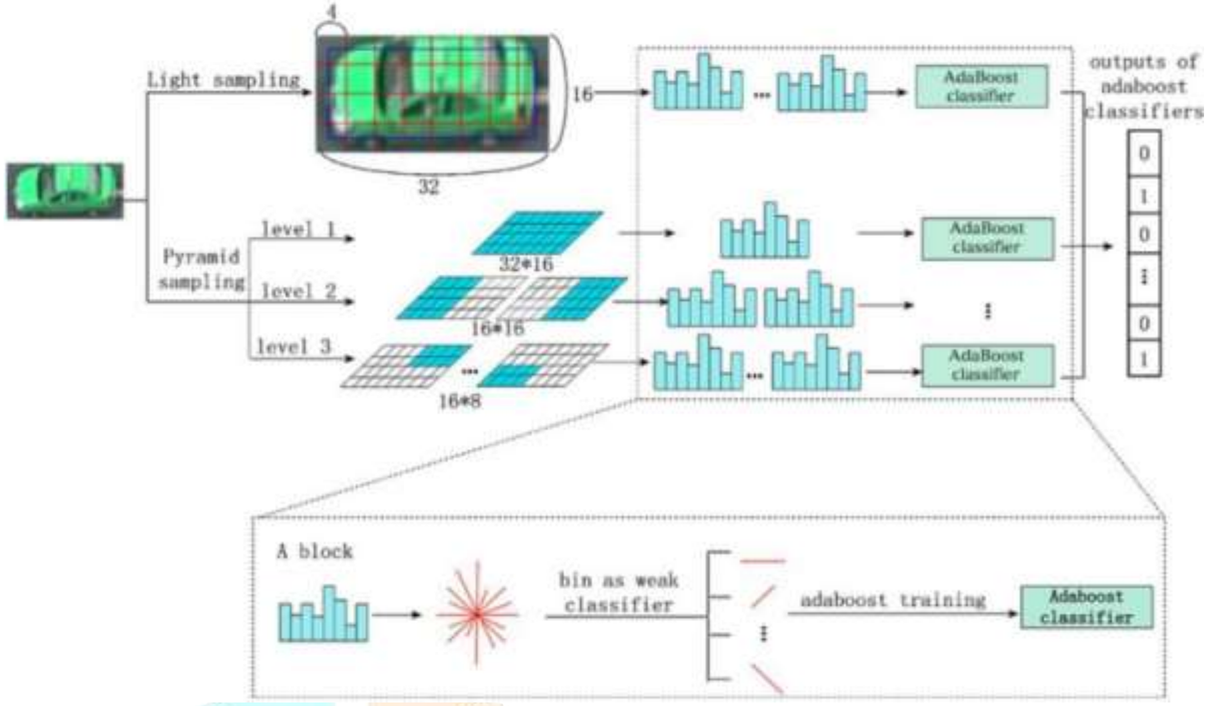


Figure 2.5: extraction process workflow of the bLPS-HoG features (Xu et al, 2011)

CHAPTER 3

MACHINE LEARNING TECHNIQUES

3.1 Machine Learning

The digital revolution has expedited new issues confronted human beings. Quick tech development, human interaction with electric devices and various technologies, electronic records and notable development of people in the web set up hands together to create a gigantic measure of data each second. From the previous two decades associations, colleges, scientists and academicians are attempting to grow new patterns and innovations to use these information for various purposes for example, for detections, recognitions, investigation, distinguishing pieces of proof, recomendations and so on. Practically all businesses these days use AI for the improvement and precision of their working procedures and methods like it is being utilized in Medical, engineering, finance, manufacturing and so forth one of these patterns is Machine learning in computer technology what's more, man-made consciousness which is considered and focused by tech specialists enormously.

The term Machine Learning (ML) is right off the bat utilized by Arthur Samuel in 1959 and being created and finished with gigantic different researchers till now. ML is a piece of Artificial Intelligence (AI) that uses statistic strategies to offer capacity to Computer applications and empower them to gain from data dynamically without earlier directions and characterized programs (Koza et al, 1996). ML is doing that it makes the concealed data known by assessing and perceiving examples and relations between data and occasions. For these purposes ML utilizes computer algorithms, for the most part, models are worked to gain from the data also, make predictions on a similar sort of information which is prepared by. Traditional applications are working in constrained and limited guidelines which are characterized by software engineers. Advancement and self-learning properties of algorithms influence them to conquer conventional applications since building information driven applications like computer visions or email sifting and so forth are nearly infeasible with traditional programming strategies. These algorithms help us to take better decisions and bring dependability. ML has 3 fundamental sorts they are as per the following:

Supervised

- Unsupervised
- Reinforcement

3.1.1 Supervised Learning

In most of the cases, supervised learning is utilized for even minded AI problems. There are two factors in supervised learning, one for inputs of info and one for outputs.

The algorithms are attempting to master mapping between these factors through a mapping work (Russell and Norvig, 2016). The algorithms are prepared with information sources and thought about the result to accessible outputs. In the training stage as a coach or educator, we screen the learning.

At the point when the algorithms anticipate the output, it will check whether the appropriate response is correct or wrong or near output or a long way from the output. These procedures help the algorithms to learn and improve their performance. The learning procedure winds up when it comes to a worthy accuracy level. At the point when the new data comes, it endeavors to foresee the output dependent on the past learnings and estimated mapping among information inputs and outputs.

Supervised learning algorithms are gathered into regression and classification which are examined in the last section. There are a few concerns exist that these algorithms are as it were operational when the data is labeled. Since the using data as another oil on the planet, data isn't free any longer and social events information for learning may be costly. For the commonsense part, a ton of supervised algorithms have been tested. Every one of them had their own qualities and shortcomings. As a base idea in ML, there is no such algorithm that works best for all sort of tasks. Accordingly picking the algorithm is a significant topic that all ought to think about it while working in ML. probably the most utilized managed algorithms are as per the following:

- Support Vector Machines
- Naive Bayes
- Logistic regression
- linear regression
- linear discriminant analysis

- Decision trees
- k-nearest neighbor
- Neural Networks

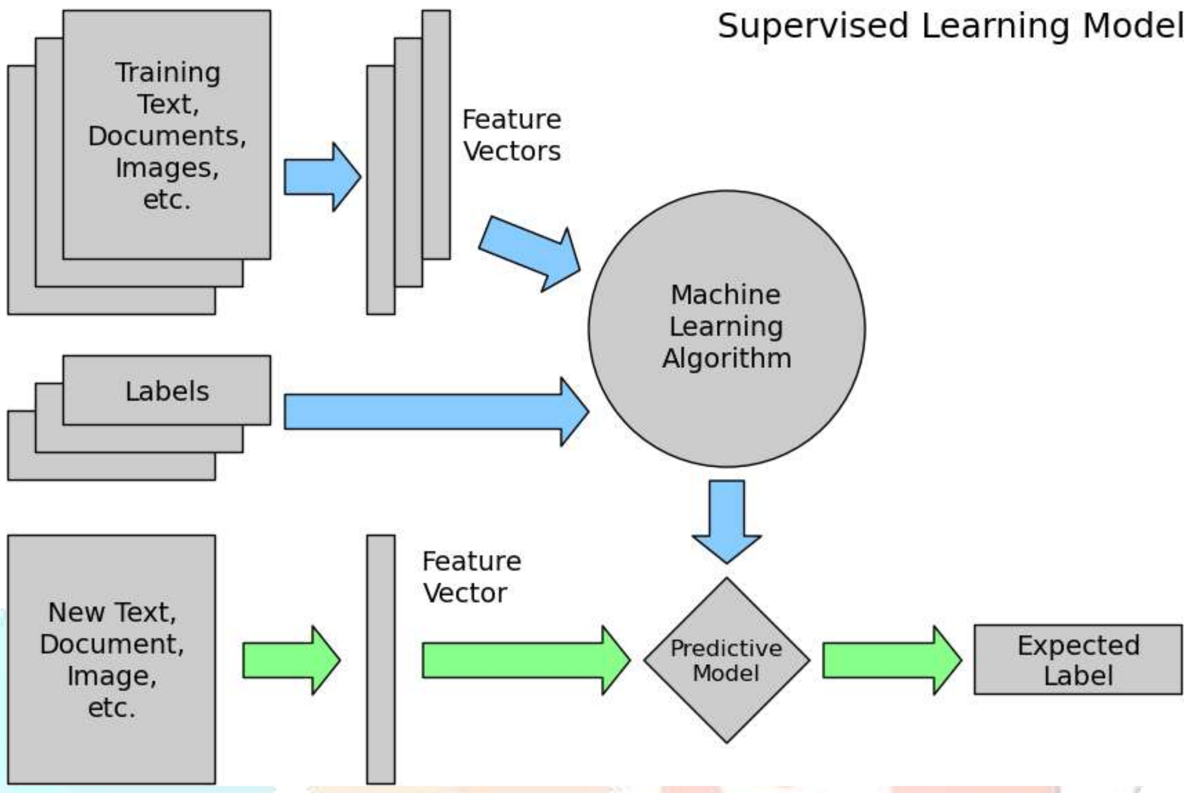

Supervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
Labels
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Expected Label

Figure 3.1: Diagram of the Supervised Learning

3.1.2 Unsupervised Learning

Unsupervised algorithms not at all like supervised don't have right answers. As it were, there is no output variable and a guide or instructor to address botches. The algorithms are attempting to comprehend the information features. They search for covered up and concealed examples in the dataset to predict the output by simply having the input factors. There are no names for them to use so as to learn and improve their predictive capacity (OFOR, 2018). Unsupervised learnings are gathered in clustering and association problems.

Clustering: in this sort of tasks the information is partitioned into gatherings, for example, grouping clients by their purchasing behaviors.

Association: algorithms are attempting to comprehend the standards that can elucidate the expansive bit of the data, for example, client purchases shirt will in general purchase pants as well.

To give some examples of unsupervised learning calculations:

- K-means clustering
- Apriori algorithm association

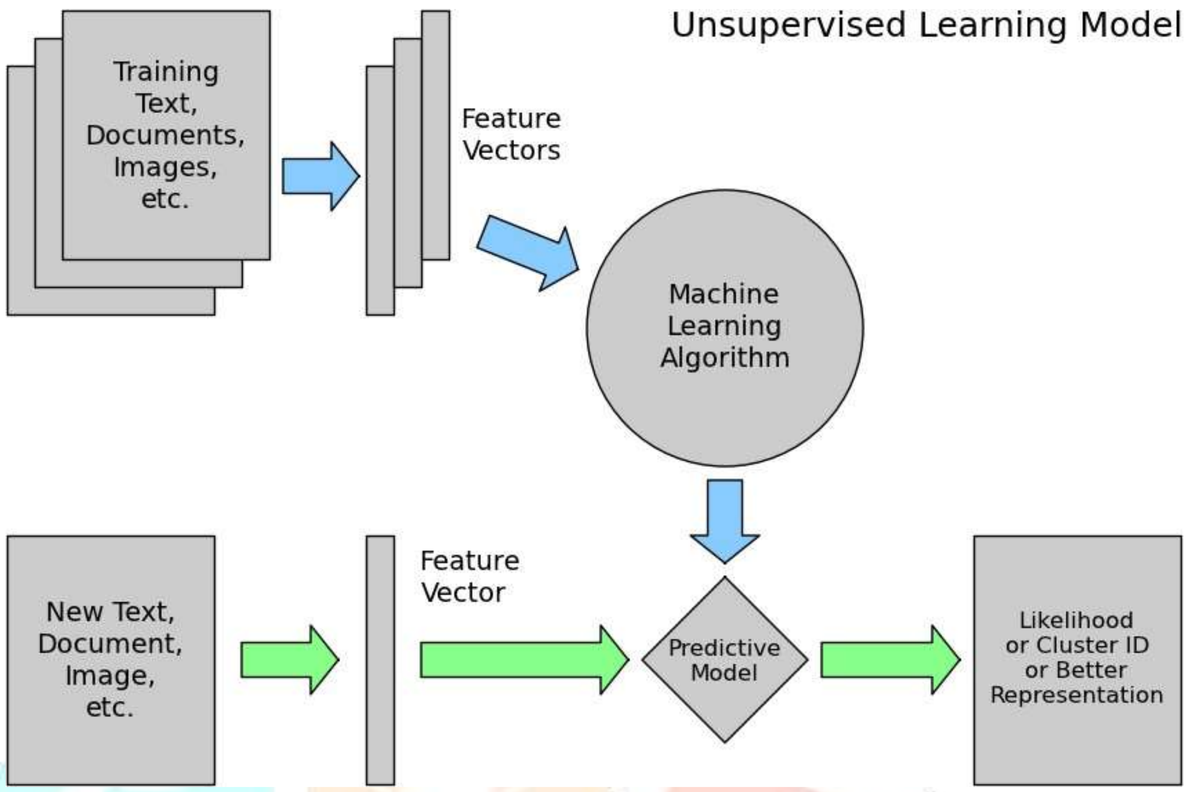

Unsupervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Likelihood or Cluster ID or Better Representation

Figure 3.2: Unsupervised learning model diagram

3.1.3 Reinforcement Machine Learning

The reinforcement learning fundamentally works like a child learning in the beginning period of his/her life. At the point when a kid is working superbly the individual will be induced and when a kid is completing a terrible job, the outcome is by one way or another disciplines or advising for not rehashing that. These algorithms are doing likewise task by an agent fills in as a child here. The agent cooperates with enviroment, it gets remunerate for performing undertakings correctly and punishments for performing it badly. When agent endeavoring to amplify the prizes and limit the punishments.

Think about a self-driving vehicle, if the vehicle touched base in its goal with no accidents, going out of the street or terrible stops it will get rewards yet on the off chance that it did any of the referenced undertakings it will get punishments. In this way next time the vehicle won't do the accidents that it took punishments for. These algorithms are additionally called dynamic programming and immense measure of studies and Researches are being led to improve these algorithms, they will be attainable for a great deal of tasks in the close future.

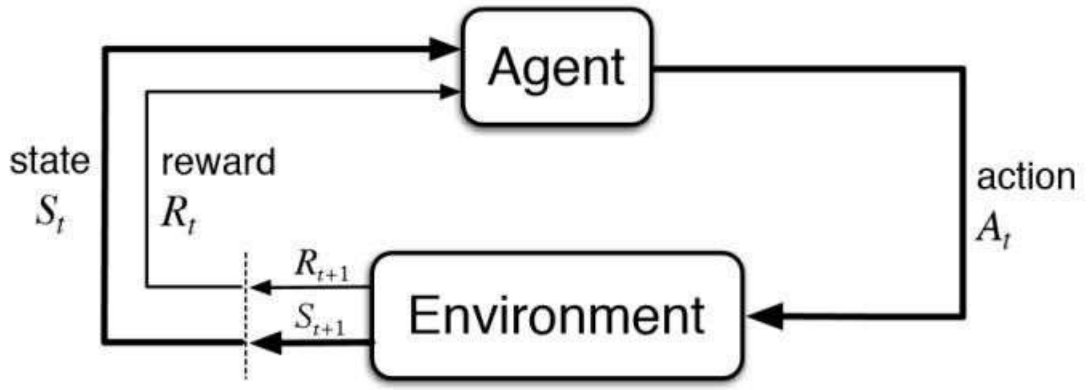


Figure 3.3: Reinforcement Learning model (UCBWiki, 2016)

3.2 Used Machine Learning Techniques

In this thesis, the author implemented and developed two algorithms from supervised category on the same dataset which is described earlier. These algorithms are as follows:

- Support Vector Machine (SVM)
- Decision Trees

3.2.1 Support Vector Machine (SVM)

This is a supervised machine learning algorithm that can be utilized for classification and regression tasks. The general strategy in SVM is to discover a hyperplane which is splitting the dataset into two classes (KDnuggets, 2016) like presented in the Figure 3.6.

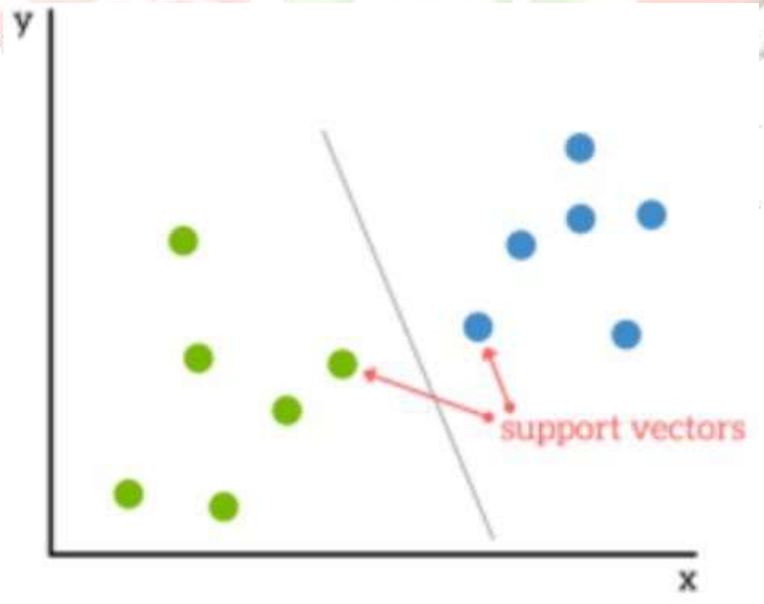


Figure 3.4: SVM splittion

These are the data points which are basically close to hyperplane, for instance if the point is going to be removed from the dataset it will change many things including role of the hyperplane splitter. This is one of the important factors in datasets.

3.2.2 Decision Trees

These type of algorithms are most used supervised algorithms by researchers. It enables developer to build high accuracy models with having a good simplicity in compare to other ML models. The basic outcome in decision tree is whether "YES" or "NO".

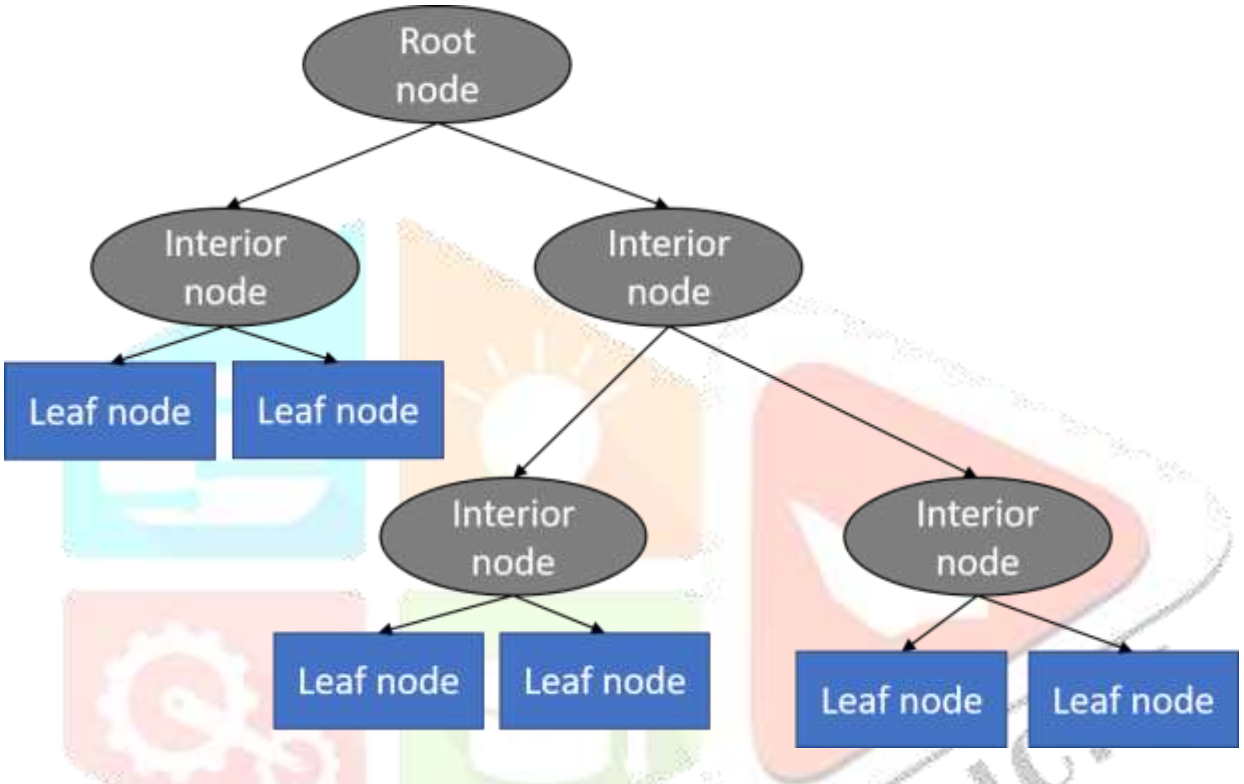


Figure 3.5: Decision tree flow chart

These models are working likewise a tree, as showed in figure above. It basically breaks the whole dataset into smaller portions and the process continuous in the sub portions from the root. It goes to reach to a tree with leaf nodes and decision nodes. The second one is a type of tree which is named root node. The advantage is that it can process both types of data including categorical and numerical data. All types of nodes are presented in Figure 3.5.

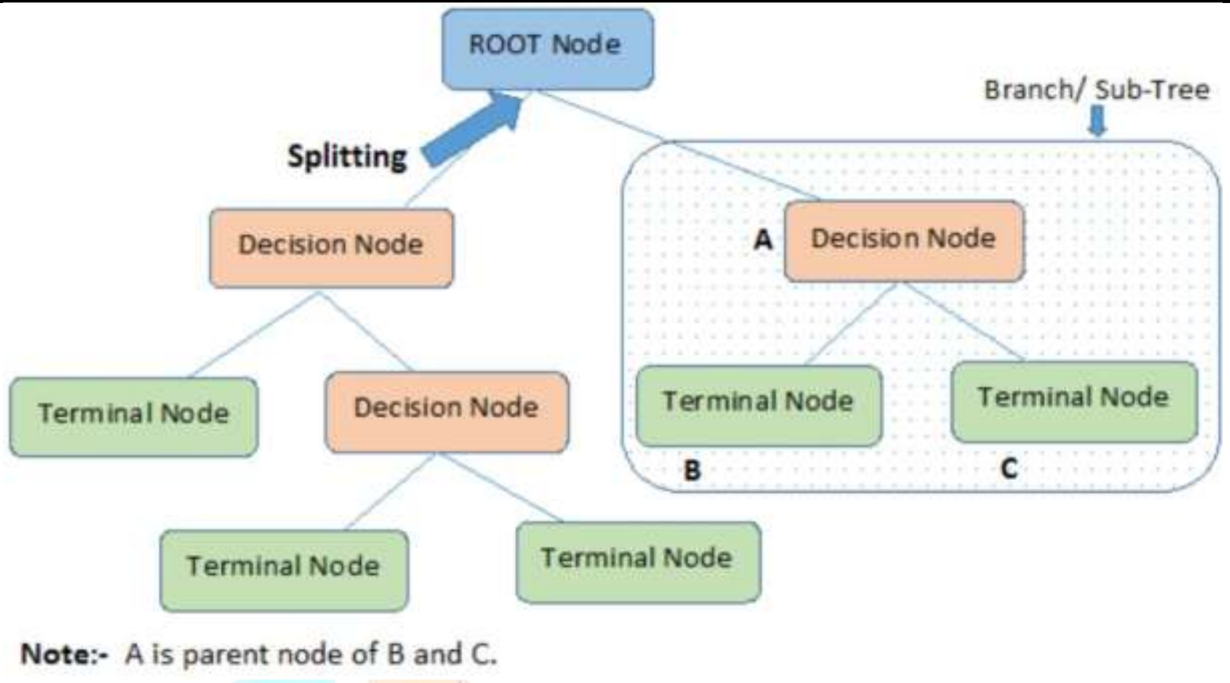


Figure 3.6: Nodes representation

CHAPTER 4 METHODOLOGY

In this chapter, all methods and tools used to detect and track vehicles are being discussed. At the very beginning, tools that are utilized in the research is described in details that how they are being used in the context. After that, the implementation process started with data cleaning, preprocessing and algorithms are presented with a short summary of these issues at the end of the chapter.

4.1 Tools Used

Like every other researches and studies this thesis utilized some applications and tools for creating models and experiments. For such researches many tools needed to be used, for example the author used python as the programming language of the model development or the dataset with a lot of vehicle and non-vehicle images used for training or for example many python libraries which are necessary or we can say useful in order to create ML models. In this chapter all the tools that are being used in the thesis are presented.

4.1.1 Python

Python is a general purpose programming language which is created by Guido van Rossum, it is used for different platforms like mathematic, computer GUI, web and so many large scientific applications. The main aim creating python was to make programming easy so that everyone in the world is able to write code. Therefore it's popular for its simplicity. Python is sensitive in spacing. Despite the fact that python was created for kids, for the time being it bits all programming languages in many fields which is amazing and somehow unbelievable. Python is able to do whatever other programming languages can do. From web to algorithms to desktop applications etc. in the past few years' python became popular in in artificial intelligence and machine learning applications due to having many efficient and handy libraries which makes the job much easier and faster. Experts coming to this field have various backgrounds. Since they don't have programming background, the most easy and convenient language to start with is python. Despite the above mentioned properties, the researcher's knowledge and self-interest caused to choose python as the programming language of the model development. The libraries that are utilized in this thesis are as follows.

4.1.1.1 Numpy

Numpy is an open source library, it does the computing with the help of multi-dimensional matrices and arrays. It contains a number of functions which makes it easy to work with these type of data. In data analysis if we want to make the speed fast and efficient we need to use arrays, therefore this library helps data scientists to work faster with large amounts of data. Usually for detection and forecasting, the models function needs arrays as parameter to operate fast and decrease the training prediction time.

4.1.1.2 Matplotlib

Plotting is growing in all fields to visualize the data and to make it understandable. Therefore Matplotlib is being used as a plotting library to create different kind of graphs and figures for variety of aims. The good thing about matplotlib is that it can produce good plots and graphs with just few lines of codes. So matplotlib is used to extract color features and create histograms.

4.1.2 Jupyter Notebook

Jupyter Notebook is a web based application which makes us able to create and modify live codes, equations, plaintexts and visualizations. This is an open source notebook which supports many programming languages. This is used for different purposes like machine learning, numerical simulation, information visualization etc. the researcher decided to use this notebook for writing readable codes and implementing machine learning algorithms.

4.1.3 Computer

The PC that is being used to train and test the models has the following properties: Model: Dell inspiron

13-5378

RAM: 8 GB

Processor: Core i7 Quad Core Graphic: Intel HD 4 GB

Datasets

Udacity website equips students with the great resources for training the classifiers. Vehicles and non-vehicles samples of the KITTI vision benchmark suite have been used for training as shown in figure 4.1. The dataset is downloaded from Udacity website.

These example images come from a combination of the GTI vehicle image database, the KITTI vision benchmark suite, and examples extracted from the project video itself. You are welcome and encouraged to take advantage of the recently released Udacity labeled dataset to augment your training data.

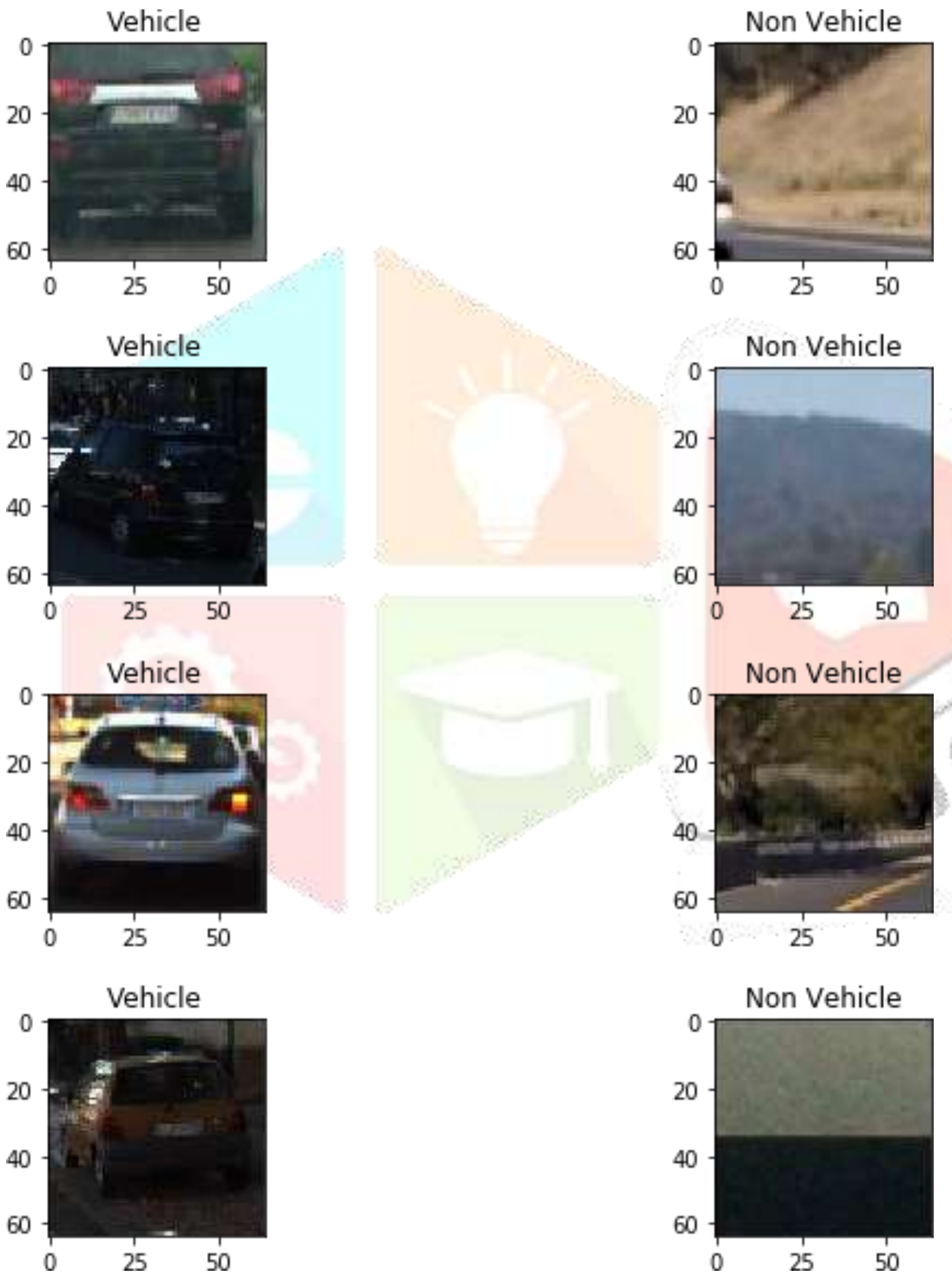


Figure 4.1: Vehicle and Non-Vehicle images

4.2 Implementation

Vehicle detection and tracking is important in self-driving technologies to drive car safely. In this project, goal is to write a software pipeline to detect vehicles in a video.

It can be achieved by following the below tasks:

- Perform a Histogram of Oriented Gradients (HOG) feature extraction on a labeled training set of images and train a classifier Linear SVM classifier.
- Implement a sliding-window technique and use your trained classifier to search for vehicles in images.
- Run pipeline on a video stream and create a heat map of recurring detections frame by frame to reject outliers and follow detected vehicles.
- Estimate a bounding box for vehicles detected.

Vehicle and non-vehicle images as Numpy array are loaded to the separate list using function. There are various feature extraction techniques has been used to train the classifier to detect the cars efficiently.

While it could be cumbersome to include three color channels of a full resolution image, you can perform spatial binning on an image and still retain enough information to help in finding vehicles.

As you can see in the code in appendix, even going all the way down to 32 x 32 pixel resolution, the car itself is still clearly identifiable by eye, and this means that the relevant features are still preserved at this resolution.

A convenient function for scaling down the resolution of an image is OpenCV's cv2.resize(). If you then wanted to convert this to a one dimensional feature vector, numpy's ravel() function can be used.

4.2.1 Color histogram

In photography a histogram is simply a graphical representation of the number of pixels in the image that fall within a certian range, either luminance or color. For example for a normal luminance histogram the graph shows the number of pixels for each luminance or brightness level from black to white. The higher the peak on the graph the more pixels are at that luminance level. With a color histogram the principle is the same but instead of seeing the levels of black graphed you will now see the number of pixels for each of the three main colors. A color histogram is a simply a histogram that shows the color level for each individual RGB color channel.

If we had to, we could differentiate the two images based on the differences in histograms alone. As expected the image of the red car has a greater intensity of total bin values in the R Histogram 1 (Red Channel) compared to the blue car's R Histogram 2. In contrast the blue car has a greater intensity of total bin values in B Histogram 2 (Blue Channel) than the red car's B Histogram 1 features. Differentiating images by the intensity and range of color they contain can be helpful for looking at car vs non-car images.

4.2.2 Histogram of Oriented Gradients (HOG)

A feature descriptor is a representation of an image or an image patch that simplifies the image by extracting useful information and throwing away extraneous information. The technique counts occurrences of gradient orientation in localized portions of an image. In the HOG feature descriptor, the distribution (histograms) of directions of gradients (orientedgradients) are used as features. Gradients (x and y derivatives) of an image are useful because the magnitude of gradients is large around edges and corners (regions of abrupt intensity changes) and we know that edges and corners pack in a lot more information about object shape than flat regions. Next one is to choose the right parameters to train the classifier to predict the image, I have defined the parameter class to define these parameters.

4.2.3 Classifiers

There are two classifiers that have been used in this thesis.

- Support vector machines (SVMs)
- Decision Tree

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outlier's detection. I have decided to use LinearSVC as classifier in this project.

Decision Tree is also a supervised machine learning algorithm which is mostly used to classify the data. The details about classifiers are presented in chapter 3.

Train and Test Split

The StandardScaler() function assumes data is normally distributed within each feature and will scale them such that the distribution is now centred around 0, with a standard deviation of 1. 'x' values are transformed using the function and get the output scaled_X.

There are few helping libraries to split the dataset. 'train_test_split' funtion from 'sklearn' is one of them which help to split the dataset into train and test data for the classifier.

4.2.4 Sliding Window

In the context of computer vision (and as the name suggests), a sliding window is rectangular region of fixed width and height that "slides" across an image. For each of these windows, we would normally take the window region and apply an image classifier to determine if the window has an object that interests us.

Here are three test images and we can see all the bounding boxes for where my classifier reported positive detections. You can see that overlapping detections exist for each of the two vehicles, and in two of the frames, there is a false positive detection on the middle of the road. In this exercise, you'll build a heat-map from these detections in order to combine overlapping detections and remove false positives.

In order to combine overlapping detections and remove false positives, heatmap and threahold limit are used.

The hog sub-sampling is more efficient method for doing the sliding window approach. The code only has to extract hog features once and then can be sub-sampled to get all of its overlaying windows. Each window is defined by a scaling factor where a scale of 1 would result in a window that's 8 x 8 cells then the overlap of each window is in terms of the cell distance. This means that a cells_per_step = 2 would result in a search window overlap of 75%. Its possible to run this same function multiple times for different scale values to generate multiple-scaled search windows. The hog sub-sampling helps to reduce calculation time for finding HOG features and thus provided higher throughput rate.

I have decided to choose stating position of the window search from 350px to 656px and cells_per_step reduced to one to get more accurate result. As explained above, same heatmap and threshold with limit 1 technique is used to combine overlapping detections and remove false positives.

4.2.5 Pipeline video

Finally create the pipeline vide by processing the each frame of the image with above techniques and create the video out of the processed frames.

find_cars_hog_sub function extracts all the bounding boxes detected for the cars in the image. heat_threshold function is used to combine overlapping detections and remove false positives and produce the output with bounding box added to the image.

4.4 Model Development Summary

The entire process of vehicle detection and tracking is described in figure 4.2. Each step needed few actions and tasks to be completed. The flow diagram clearly describes the entire process.

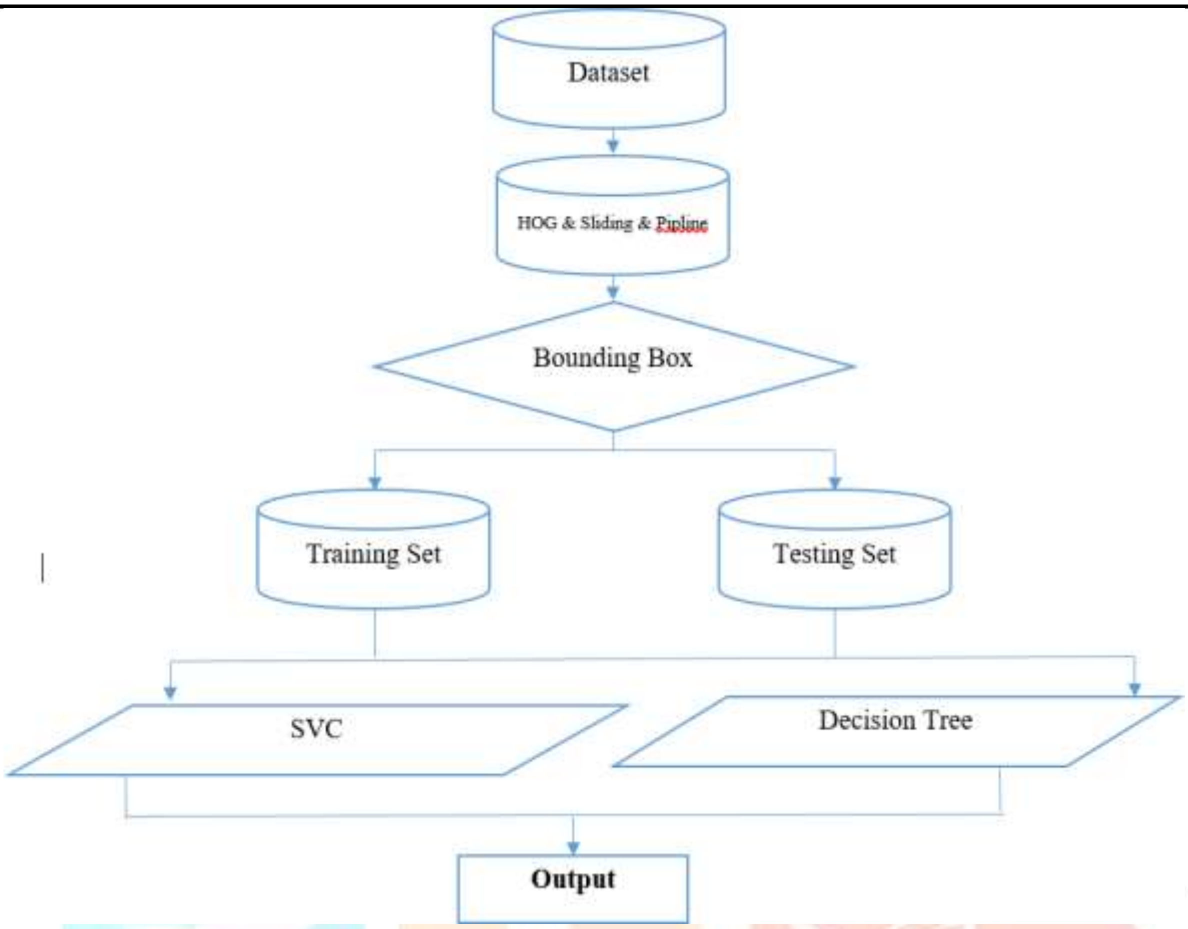


Figure 4.2: Model development summary

CHAPTER 5 RESULTS AND DISCUSSION

5.1 Experimental Setup

While developing Machine learning algorithms there are many options to choose for instance MATLAB, Python, and R programming language. Each option has its own advantages and privileges. Because of that the researcher has decided to choose python for the development due to the easiness and rich libraries available for all kind of tasks. The tools and techniques which have been used in this thesis is discussed in details in chapter 4. Many libraries have been utilized to perform various tasks like Numpy is used for importing and processing data, matplotlib is used for visualization of the extracted color features, Scikit-Learn for splitting data into train and test parts, classifier functions for creating the models and training them with available data in the datasets. In this chapter we are going to discuss the creation of the models along with focusing on the results. Like mentioned in previous chapters, there are two algorithms in the thesis which are used to classify the images and a pipeline to track images in the videos. The tow classifiers are SVM and Decision Tree. At the end of the chapter the result has been described in a comparative way which the best model or classifier is suggest among the two.

Support Vector Machine (SVM) Classification

SVMs algorithm are used mostly for classification tasks. These models are working based on discovering a hyperplane concept which actually perfectly divides the data into two classes (Bambrick, 2016).

- SVM works perfect while dealing with unknown data.
- It is efficient when the data is semi-structured or unstructured like texts, trees, and images.
- SVM properly measure dimensional data

5.2.1 In SVM usually there are no danger of over fitting due to having generalization in practice SVM implementation

Like mentioned earlier, there are bunch of libraries that can help us implement the models easily. First these libraries are needed to be imported in the notebook. Then the dataset needed to be read and load into arrays for further processing. Numpy have been used for these job. For the testing purpose and to understand whether the data is loaded properly or not, few examples of the dataset showed in Figure 5.1.

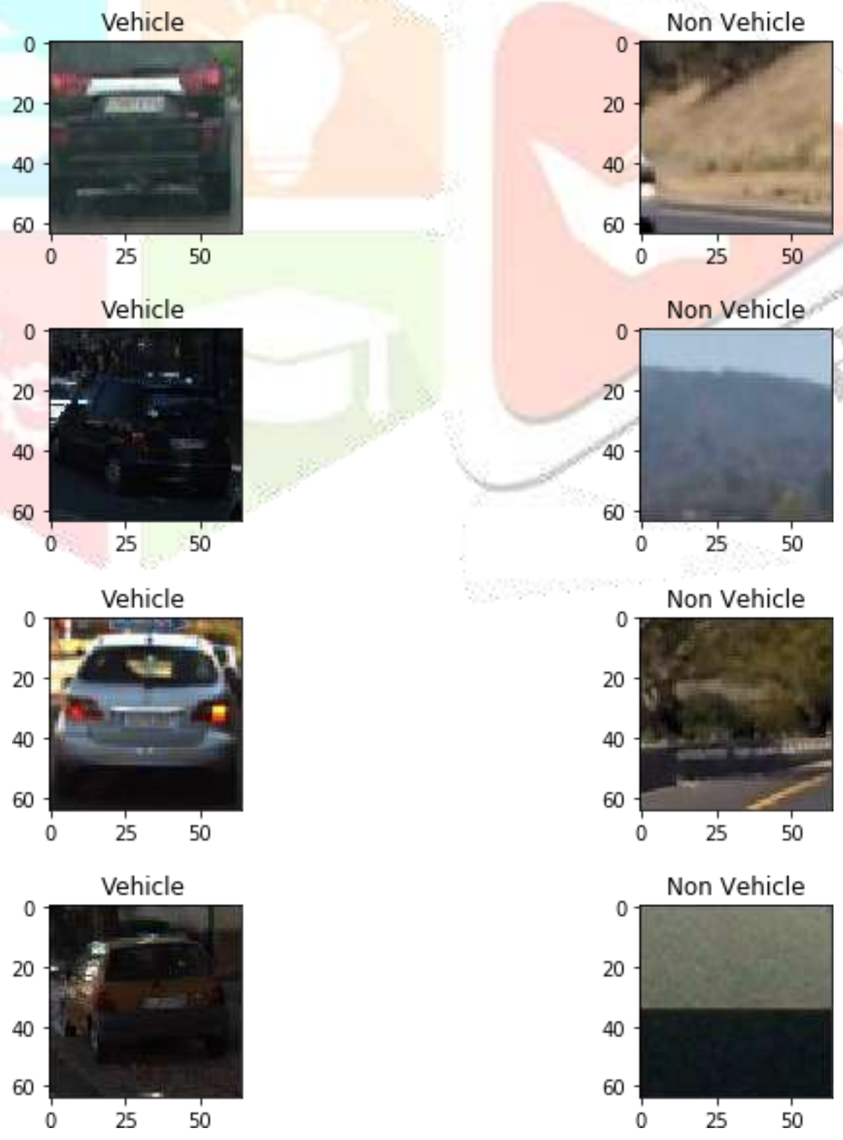


Figure 5.1: Dataset few instances

A histogram is an exact portrayal of the dissemination of numerical information. It is a gauge of the likelihood circulation of a continuous variable (CORAL) and was first presented by Karl Pearson. It varies from a bar graph, as in a bar diagram relates two factors, yet a histogram relates just one. To develop a histogram, the initial step is to "bin" (or "bucket") the scope of qualities—that is, partition the whole scope of qualities into a progression of intervals—and after that check what number of qualities fall into every interim. The receptacles are typically determined as successive, non-covering intervals of a variable. The containers (intervals) must be adjoining, and are frequently (yet are not required to be) of equivalent size. Color features and histogram for vehicles and non-vehicles are presented in Figure 5.2 and 5.3.

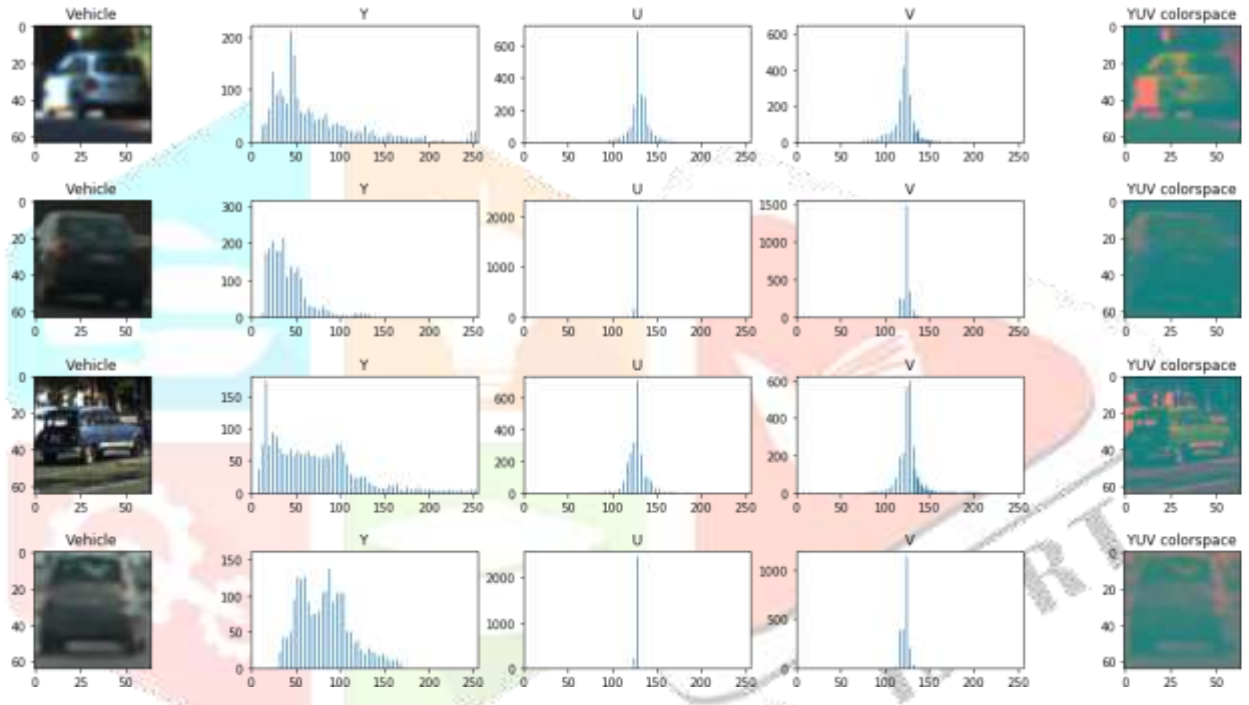


Figure 5.2: Histogram and Color feature of Vehicles

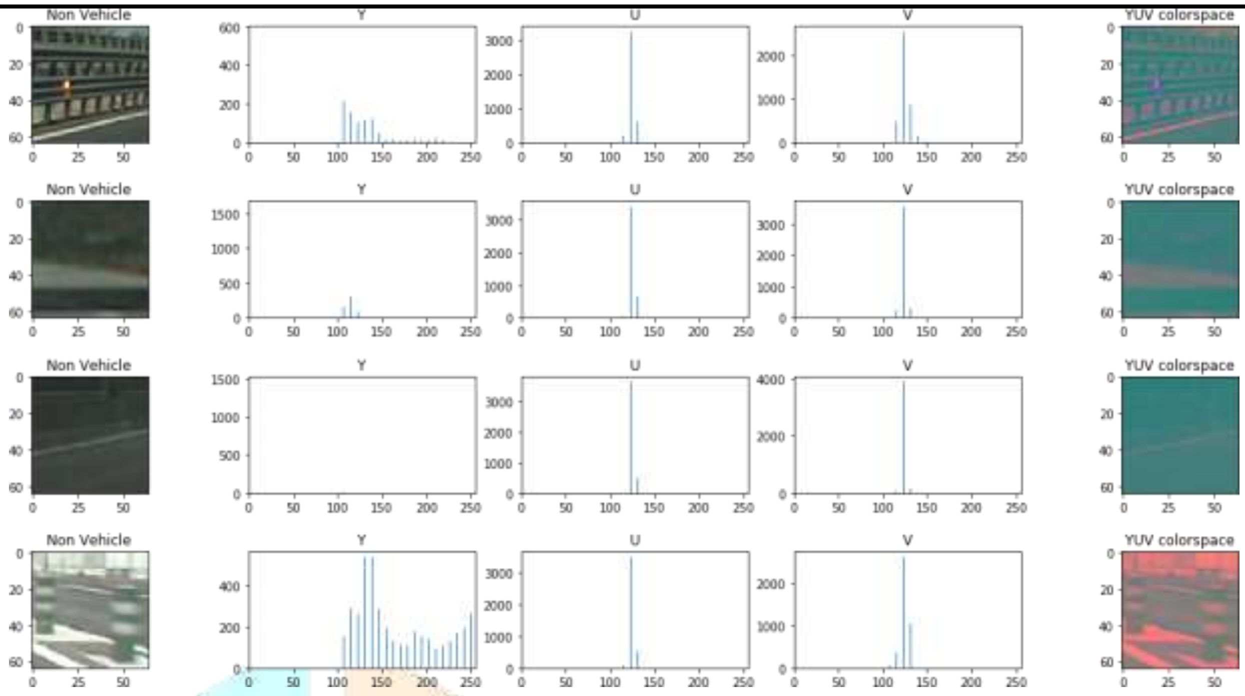


Figure 5.3: Histogram and color feature of non-vehicles

After the extraction of the color feature, the data is going to bin. We have spatial binning which is "This modifier generates a 1-, 2- or 3-dimensional grid covering the simulation domain and assigns each particle into one of the uniformly sized bins. It then performs a reduction operation for a selected particle property, mapping the values of all particles contained in a cell to a single output value. This modifier can thus be used to project the per-particle data to a structured grid, for example to coarse-grain the atomistic data and generate a continuous field representation of a particle property. You can choose between different reduction operations, e.g. sum, average (mean), minimum or maximum. The bin grid can be one-, two- or three-dimensional, i.e. the simulation domain can be subdivided into equally sized bins along one, two or all three of its axes. The spatial bins are always aligned parallel to the simulation cell edges" (Chen, 2015). The action is conducted on the dataset via binning functions and the results are as follows:

No of features before spatial binning 12288 No of features after spatial binning 768
Hereafter the histogram of oriented gradients are extracted. "The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy". The result of the HOG extraction is showed in Figure 5.4 and the following:

Feature Vector Length Returned is 324
No of features that can be extracted from image 4096

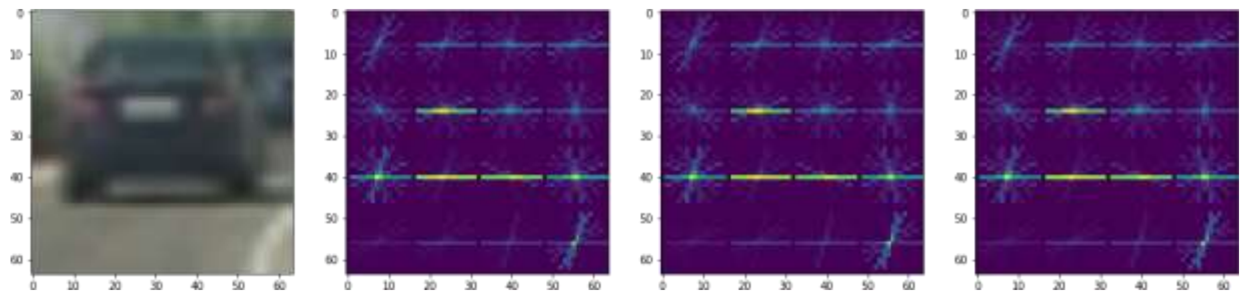


Figure 5.4: HOG features extraction from one sample of the vehicles

The next step is data preprocessing, for training any algorithm we need to prepare data, do some cleaning, scaling etc. to feed the algorithm. Since the dataset here is an imagery dataset, there are no need for cleaning. Therefore, in this step the data splitting is needed to be done.

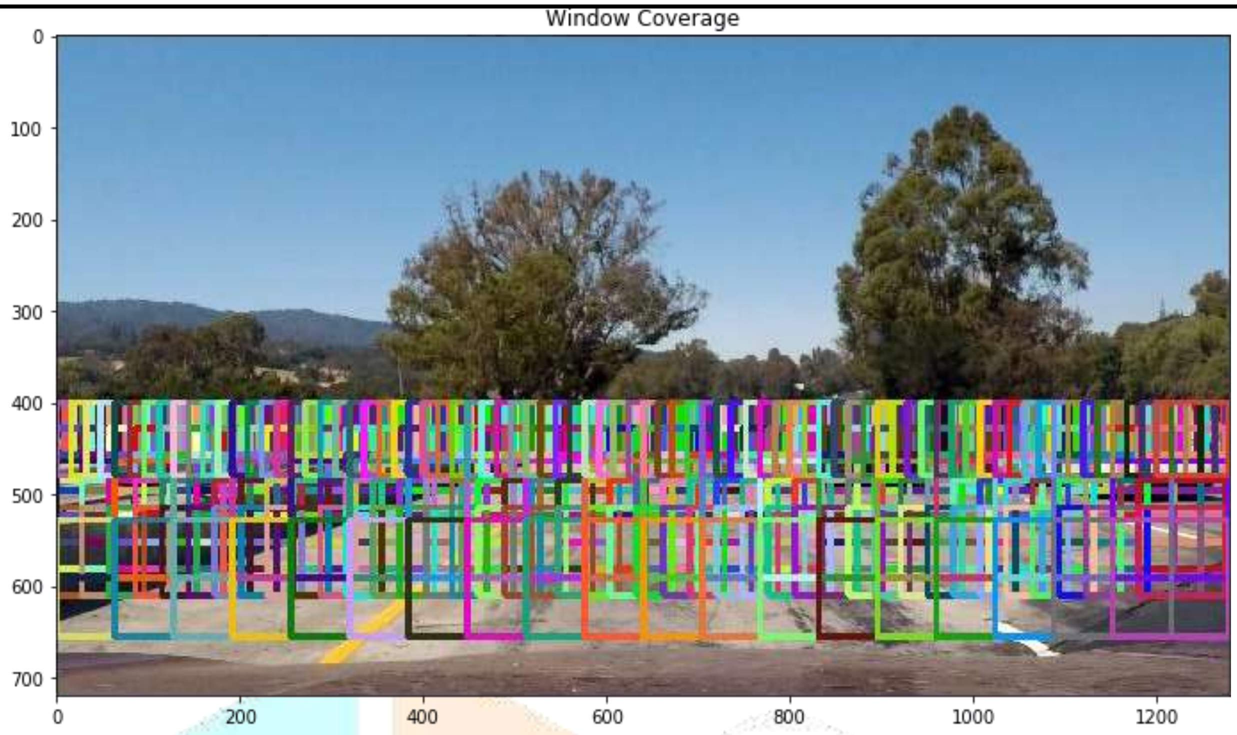
Data is split into two parts training and testing by the help of the train_test_split() function from the Scikit-Learn library. The percentage for both of them are considered like in general 80% for training and 20% for testing. Besides that the data has to be normalized. In the procedure of the SVC classifier the StandardScaler() function has been utilized in order to normalize the data.

After all these processes the SVC classifier has been trained and the result that have been obtained from the algorithms is presented in table 5.1.

Table 5.1: SVC classifier training results

Parameters	**Values**
Time	0.9 second
Error Count	43
Accuracy	0.9879
Mean Squared Error	0.012106
Root Mean Squared Error	0.110027
Mean Absolute Error	0.012106

look in the table the algorithm is trained in 0.9 second and has 98% accuracy. With those values in evaluation matrices. When the model is trained with the training part of the dataset, the sliding window is the next task. There are functions to draw sliding windows. After running the sliding, the system needed to find the windows on which we are going to run the classifier. The function which is returning the refined windows in which the classifier predicts the output to be a car. Then the function which is drawing the main window around the identified cars is needed to be run. The sliding window and the identified car in drawn window around the car in the images are presented in Figure 5.5.



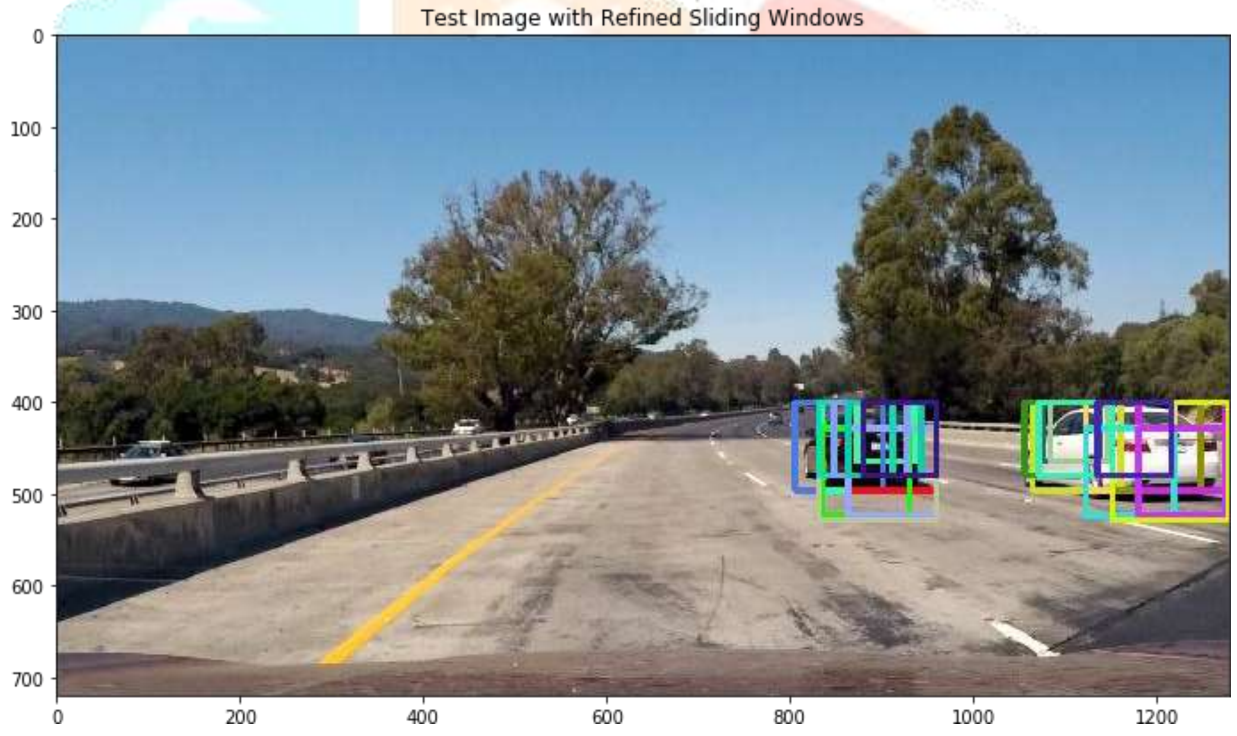


Figure 5.5: Sliding window with the refined sliding windows

The process of drawing boxes around the cars are getting precise. Then after these all the heat map is applied in order to increase the pixel by one inside each box. Then we need to apply a threshold value to the image to filter out low pixel cells and find pixels with each car number and draw the final bounding boxes. The testing heat map images are shown in Figure 5.6.

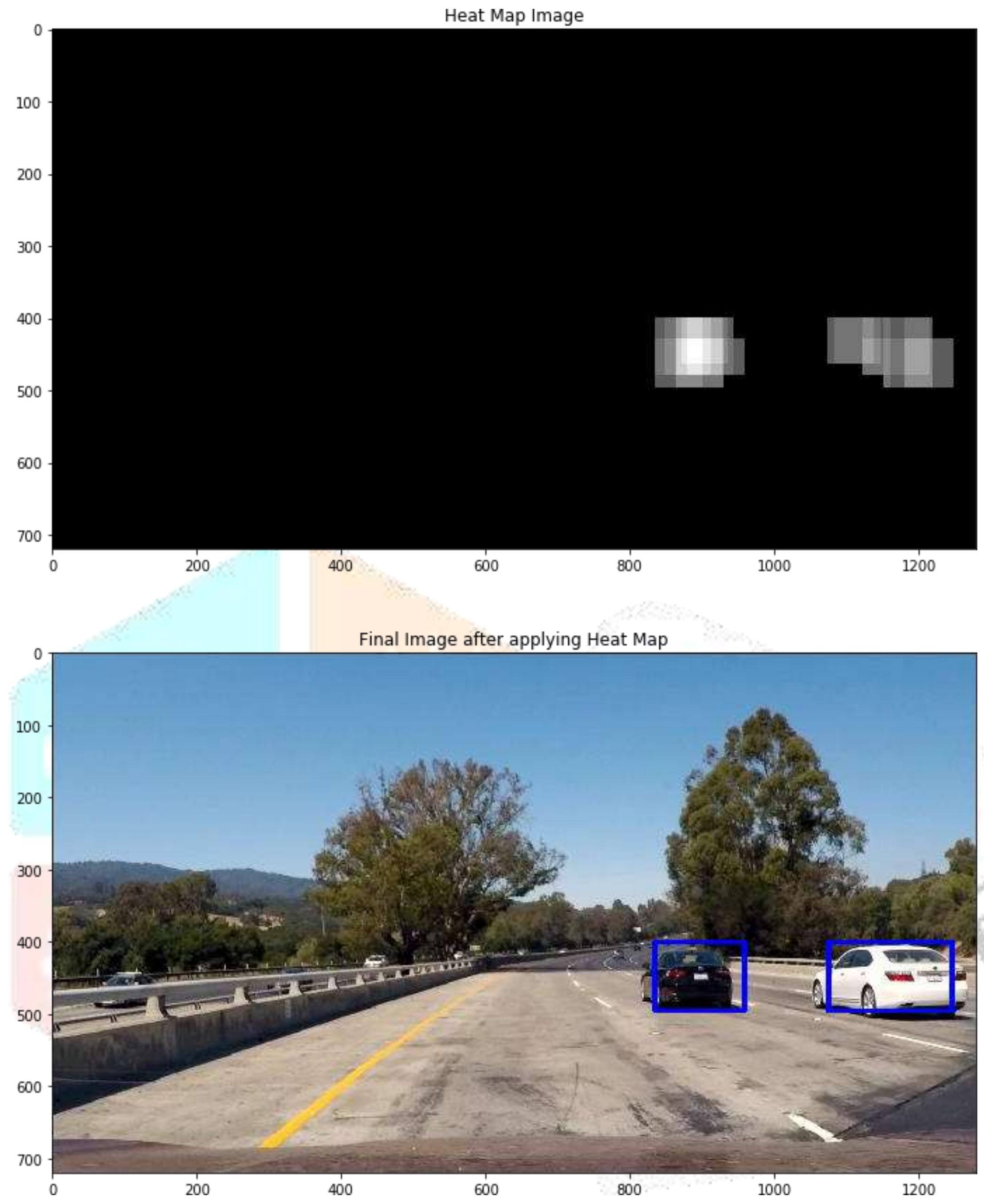


Figure 5.6: Heat Map on testing image

Till here the models are detecting the cars and drawing boxes around them. The main task is complete. Although, it can be said that the system can detect the cars but cannot track them in a video. In order to do that a pipeline needed to be defined. The pipeline creator function needs multiple parameters to load a video. The pipeline function has been created. A sample of the pipeline result on Test images is shown in Figure 5.7.

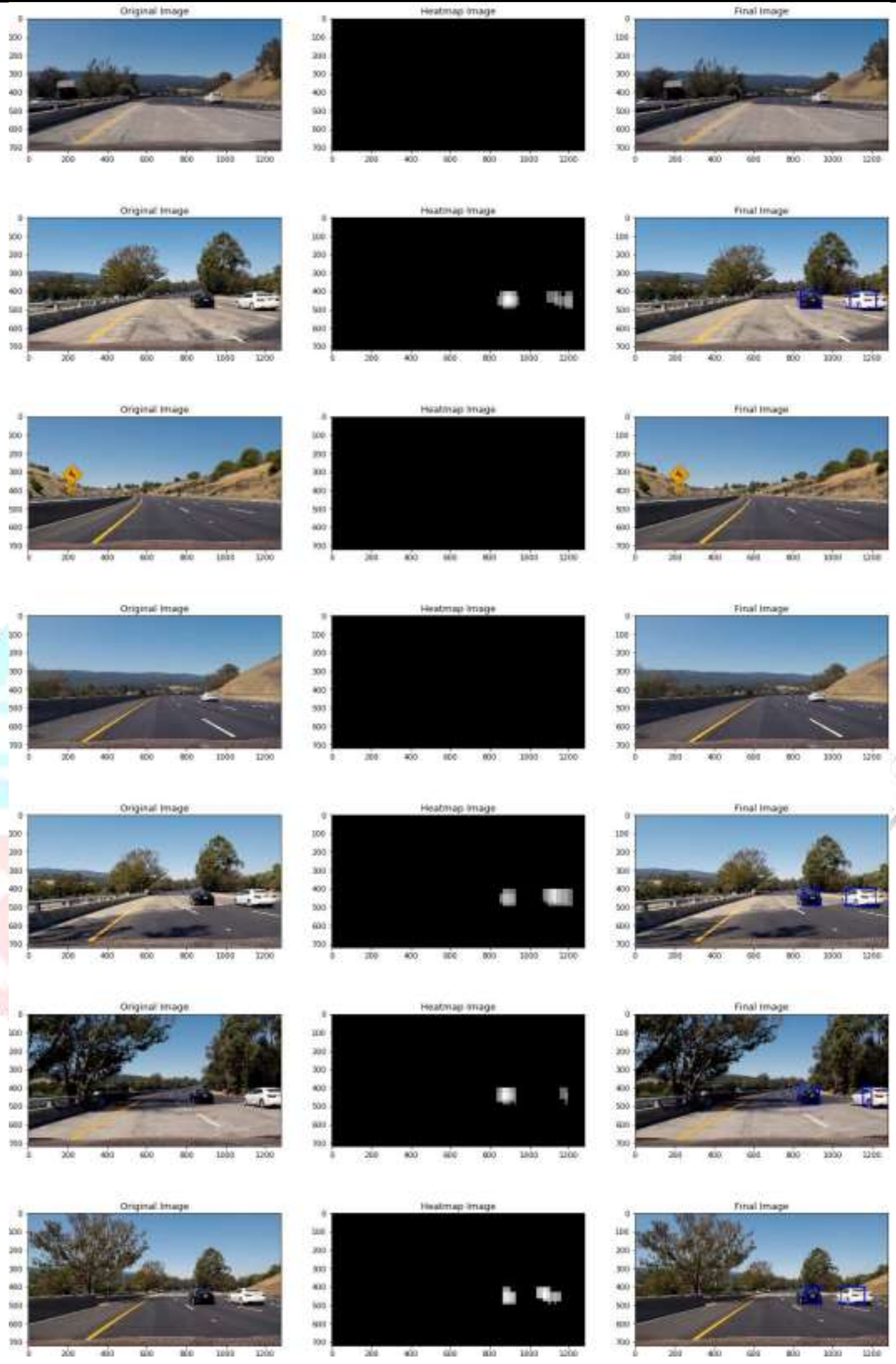


Figure 5.7: Pipeline sample on test images

Same concept is conducted on the videos that are tested in the thesis. Since the video is the series of images. These concept is applied on them and the tracking process is done. While the cars passing in the video the program detect and track the cars with a box around the car.

5.3 Decision Tree

Decision Tree is a drawing or in other words graphical drawing like boxes of the algorithms. The algorithm clarify and describe the decisions which are feasible, utility, cost and consequences of issues. Although, in comparison of these kind of choices it permits one pane (Nayab & Scheid, 2011)

5.3.1 Advantages

The decision tree advantages are listed below:

- Flexibility
- Specificity
- Resilience
- Transparency
- Ease of Use
- Comprehensive

5.3.2 Decision Tree Training

For decision tree everything is the same as previous algorithm except the classifier which is Decision Tree. All preprocessing steps and other techniques which are utilized in SVM is also used in this model. Due to the similarity we skip describing repetitive topics and present the result of the algorithm in Table 5.2. In order to understand how Decision Tree works with these kind of data. And how accurate is the result of prediction.

Table 5.2: Decision Tree classifier training results

Parameters	**Values**
Error Count	205
Accuracy	0.942
Mean Squared Error	0.057714
Root Mean Squared Error	0.240237
Mean Absolute Error	0.057714

Like shown above Decision Tree obtained 94% accuracy in classification prediction of our dataset with 205 errors and those values in evaluation matrices.

5.4 Comparison and the Best Model among the Models

The two models are tested on the same dataset with the same computer even most of the techniques were the same in order to define which algorithm suited for this task. Because of the importance of the vehicle detection and tracking, the topic sensitive and precise to evaluate the results. In order to understand the two models and compare them, the result of the two models are presented in bar charts in a comparative way in Figure 5.8.

Figure 5.8: SVC and Decision Tree Comparison result.

As you can see in the figure, we have three evaluation matrices with accuracy percentage level. Since the evaluation matrices are the error levels, as much as the values for MSE, RMSE, and MAE are low the accuracy level is high and if they are high then the accuracy level is low. As you can see in the figure all MSE, RMSE, and MAE are lower in SVC at the same time accuracy level is 98% for SVC but for Decision Tree the Error levels are high and the accuracy level is 94%. This shows that SVC works better for vehicle detection and tracking tasks.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

By rapid development in car and traffic industries, at the same the growth of population in the world brought the needs for different tools and techniques specially technology solutions in order to manage traffics in cities and populated areas. Meanwhile, object detection can be used in various fields to help humans live easily with comfort and make the world a better place to live in.

Object detection can be used in industries, digitized cities, government, research, academia, environment etc. Vehicle detection and tracking is part of the object detection which is used in traffic, cities etc. the importance of the topic is growing larger. That being said this research is intended to contribute the improvement of the accuracy of these algorithms and models via available techniques and tools.

This Thesis developed two classifier algorithms to detect and track vehicles. These two models are Support Vector Machine (SVM) and Decision Tree. The algorithm selection was based on various studies in literature review. The most suggested models by other researchers were these two model. Therefore, the author decided to choose these models and compare them in order to specify the best model among these two. Many techniques have been deployed to increase the accuracy level and to make the best result possible. The models are trained with the same dataset and the evaluation result showed that SVM performs better than Decision Tree. The result of the models presented both in image and video formats which the system detect the cars that are passing from the screen and tracking them as well.

6.1 Future Works

Looking back to the limitations of the study, there are tasks and options which can be added to this research or possible to work on it separately. The topic is under the attention of researchers and improves day by day. Theoretical development is needed to be tracked and when any theoretical development is published and achieved, the researchers should utilize those concepts practically using algorithms in order to improve the accuracy level of the detection and tracking process. Although, the dataset can be improved, a future work can be testing these models using a better and larger dataset with a massive number of vehicle and non-vehicle images from different places, angels, cars, roads, cameras, distances etc.

Furthermore, other models can be added to the comparison list of models in order to make the comparison more reliable and vast.

REFERENCES

Andrew, W. M. and Victor, M. (2003), Handbook of International Banking (*London: Edward Elgar Publishing Limited*), 350-358

Baha, N. (2014). Real-Time Obstacle Detection Approach using Stereoscopic Images. *International Journal of Information Engineering and Electronic Business*, *6*(1), 42.

Bahrepour, M., Akbarzadeh-T, M. R., Yaghoobi, M., & Naghibi-S, M. B. (2011). An adaptive ordered fuzzy time series with application to FOREX. *Expert Systems withApplications*, 38(1), 475-485.

Bambrick, N. (2016, June 24). *Support Vector Machines for dummies; A Simple Explanation*. Retrieved April 28, 2018, from AYLIEN | Text Analysis API | Natural Language Processing: http://blog.aylien.com/support-vector-machines-for-dummies-a-simple/pp. 1-13.

Basak, D., Pal, S., & Patranabis, D. C. (2007). Support vector regression. *Neural Information Processing-Letters and Reviews*, *11*(10), 203-224.

Berni, J. A., Zarco-Tejada, P. J., Suárez, L., & Fereres, E. (2009). Thermal and narrowband multispectral remote sensing for vegetation monitoring from an unmanned aerial vehicle. *IEEE Transactions on Geoscience and Remote Sensing*, *47*(3), 722-738.

BIS. (2016). Triennial central bank survey: Foreign exchange turnover in April 2016 Monetary and Economic Department.

Chen, X., & Meng, Q. (2015, November). Robust vehicle tracking and detection from UAVs. In *2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR)* (pp. 241-246). IEEE.

Chen. X, (2015), Automatic Vehicle Detection and Tracking in Aerial Video, Doctor of Philosophy Thesis in Loughborough University.

Kalghatgi, M. P., Ramannavar, M., & Dr. Sidnal, N. S. (2015). A Neural Network Approach to Personality Prediction based on the Big-Five Model. *International Journal of Innovative Research in Advanced Engineering, 2(8), 56–63*

Kanistras, K., Martins, G., Rutherford, M. J., & Valavanis, K. P. (2015). Survey of unmanned aerial vehicles (UAVs) for traffic monitoring. *Handbook of unmanned aerial vehicles*, 2643-2666.

Khan, K., Baharudin, B. B., & Khan, A. (2009, June). Mining opinion from text documents: A survey. In *Digital Ecosystems and Technologies, 2009. DEST'09. 3rd IEEE InternationalConference on* (pp. 217-222). IEEE.

Koza, J. R., Bennett, F. H., Andre, D., & Keane, M. A. (1996). Automated design of both the topology and sizing of analog electrical circuits using genetic programming. In *Artificial Intelligence in Design*'96 (pp. 151-170). Springer, Dordrecht.

Nayab, & Scheid, J. (2011, September 2). *Advantages of Decision Tree Analysis*. Retrieved April 28, 2018, from Bright Hub Project Management: https://www.brighthubpm. com/project-planning/106000-advantages-of-decision-treeanalysis/

Noh. S, Shim. D and Jeon. M, (2015), 'Adaptive Sliding-Window Strategy for Vehicle Detection in Highway Environments', *IEEE IEEE Transactions on Intell. Transport.Syst.*, Ofor, e. (2018). *Machine learning techniques for immunotherapy dataset classification* (Master dissertation, Near East University).

Patel, P. J., Patel, N. J., & Patel, A. R. (2014). Factors affecting currency exchange rate, economical formulas and prediction models. *International Journal of Applicationor Innovation in Engineering & Management (IJAIEM)*, 3(3), 53-56.

Pujari, M. V., Sayyed, A. H., Shahani, H., Rupani, D., & Student, B. E. (2018). Forex Trading System. *International Journal of Engineering Science*, 17116.

Razakarivony, S., & Jurie, F. (2016). Vehicle detection in aerial imagery: A small target detection benchmark. *Journal of Visual Communication and Image Representation*, *34*, 187-203.

Russell, S. J., & Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited.

Russell, S. J., & Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited.

Sahli, S., Ouyang, Y., Sheng, Y., & Lavigne, D. A. (2010, April). Robust vehicle detection in low-resolution aerial imagery. In *Airborne Intelligence, Surveillance, Reconnaissance (ISR) Systems and Applications VII* (Vol. 7668, p. 76680G). International Society for Optics and Photonics.

Sayad, S. (2017). Support Vector Machine. *Support Vector Machines. Np, nd Web*, *26*.

Schumaker, R. P., & Chen, H. (2009). Textual analysis of stock market prediction using breaking financial news: The AZFin text system. *ACM Transactions on Information Systems (TOIS)*, 27(2), 12.

Sokalski, J., Breckon, T. P., & Cowling, I. (2010). Automatic salient object detection in uav imagery. *Proc. of the 25th Int. Unmanned Air Vehicle Systems*, 1-12.

Su, A., Sun, X., Liu, H., Zhang, X., & Yu, Q. (2015). Online cascaded boosting with histogram of orient gradient features for car detection from unmanned aerial vehicle images. *Journal of Applied Remote Sensing*, *9*(1), 096063.

Susaki, J. (2015). Region-based automatic mapping of tsunami-damaged buildings using multi-temporal aerial images. *Natural Hazards*, *76*(1), 397-420.

Tenti, P. (1996). Forecasting foreign exchange rates using recurrent neural networks. *Applied Artificial Intelligence*, 10(6), 567-582.

The Federal Reserve Board. (2004) "FRB: Speech, Bernanke--International Monetary Reform and Capital Freedom--October 14, 2004".

Tsai, Y. C., Chen, J. H., & Wang, J. J. (2018). Predict Forex Trend via Convolutional Neural Networks. *arXiv preprint arXiv*:1801.03018.

Tseng, F. M., Tzeng, G. H., Yu, H. C., & Yuan, B. J. (2001). Fuzzy ARIMA model for forecasting the foreign exchange market. *Fuzzy sets and systems*, 118(1), 9-19.

UCBWiki. (2016, February). *Reinforcement Learning with Function approximation.*

Retrieved April 22, 2018, from UCB WIki: http://wiki.ubc.ca /images /b/ bd/Rldiagram1.png

Van Gerven, M., & Bohte, S. (Eds.). (2018). *Artificial neural networks as models of neural information processing*. Frontiers Media SA.

Viola, P., Jones, M. J., & Snow, D. (2005). Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, *63*(2), 153-161.

Vyklyuk, Y., Vukovic, D., & Jovanovic, A. (2013). FOREX prediction with neural network: USD/EUR currency pair. *Актуальні проблеми економіки*, (10), 261-273.

Wang, X., Zhu, H., Zhang, D., Zhou, D., & Wang, X. (2014). Vision-based detection and tracking of a mobile ground target using a fixed-wing UAV. *International Journal of Advanced Robotic Systems*, *11*(9), 156.

Wei, P., Lu, X., Tang, T., Li, C., & Song, J. (2015, August). A highway vehicle detection method based on the improved visual background extractor. In *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)* (pp. 1519-1524). IEEE.

Werbos, P. (1974). Beyond Regression:" New Tools for Prediction and Analysis in the Behavioral Sciences. *Ph. D. dissertation, Harvard University*.

Wernick, M. N., Yang, Y., Brankov, J. G., Yourganov, G., & Strother, S. C. (2010). Machine learning in medical imaging. *IEEE signal processing magazine*, 27(4), 25-38.

Xu, Y., Cao, X., & Qiao, H. (2011). An efficient tree classifier ensemble-based approach for pedestrian detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *41*(1), 107-117.

Yu. X and Shi. Z, (2015), 'Vehicle detection in remote sensing imagery based on salient information and local shape feature', *Optik - International Journal for Light and Electron Optics*.

Zamani, M., & Kremer, S. C. (2011, November). Amino acid encoding schemes for machine learning methods. In *Bioinformatics and Biomedicine Workshops (BIBMW), 2011 IEEE International Conference on* (pp. 327-333). IEEE.

APPENDICES

APPENDIX 1

```
#importing Required Libraries to run code import glob

import cv2 as cv2 import numpy as np

import matplotlib.pyplot as plt import random

from skimage.feature import hog import time

from sklearn.preprocessing import MinMaxScaler from sklearn.svm import LinearSVC

from sklearn.metrics import mean_squared_error from sklearn.metrics import mean_absolute_error from err

import error_count

#reading image paths with glob

vehicle_image_arr = glob.glob('W:/Master NEU/Sipan Thesis/dataset/vehicles/**/*.png') # read images

and append to list

vehicle_images_original=[]

for imagePath in vehicle_image_arr: readImage=cv2.imread(imagePath)

    rgbImage = cv2.cvtColor(readImage, cv2.COLOR_BGR2RGB)

    vehicle_images_original.append(rgbImage)

print('Reading of Vehicle Images Done')

non_vehicle_image_arr = glob.glob('W:/Master NEU/Sipan Thesis/dataset/non- vehicles/**/*.png')

non_vehicle_images_original=[]

for imagePath in non_vehicle_image_arr: readImage=cv2.imread(imagePath)

    rgbImage = cv2.cvtColor(readImage, cv2.COLOR_BGR2RGB)
    non_vehicle_images_original.append(rgbImage) print("Reading of Non Vehicle Images Done")
print("No of Vehicle Images Loaded -"+ str(len(vehicle_image_arr)))

print("No of Non-Vehicle Images Loaded -"+ str(len(non_vehicle_images_original))) # Visualizing the

Vehicle and Non Vehicle Images

f, axes = plt.subplots(4,2, figsize=(10,10)) plt.subplots_adjust(hspace=0.5)

for index in range(4):

    vehicle=random.randint(0, len(vehicle_images_original)-1) non_vehicle=random.randint(0,

    len(non_vehicle_images_original)-1) axes[index,0].imshow(vehicle_images_original[vehicle])

    axes[index,0].set_title("Vehicle") axes[index,1].imshow(non_vehicle_images_original[non_vehicle])

    axes[index,1].set_title("Non Vehicle")

print("Shape of Vehicle Image" +  str(vehicle_images_original[vehicle].shape))
print("Shape of Non Vehicle Image" + str(non_vehicle_images_original[non_vehicle].shape)) ### Extract
Color Space
#creating a Histogram
def ExtractColorHistogram(image, nbins=32, bins_range=(0,255), resize=None): if(resize !=None):
        image= cv2.resize(image, resize)
    zero_channel= np.histogram(image[:,:,0], bins=nbins, range=bins_range) first_channel=
    np.histogram(image[:,:,1], bins=nbins, range=bins_range) second_channel= np.histogram(image[:,:,2],
    bins=nbins, range=bins_range) return zero_channel,first_channel, second_channel
#Find Center of the bin edges
def FindBinCenter(histogram_channel): bin_edges = histogram_channel[1]

bin_centers = (bin_edges[1:] + bin_edges[0:len(bin_edges)-1])/2 return bin_centers
#Extracting Color Features from bin lengths
def ExtractColorFeatures(zero_channel, first_channel, second_channel):
    return np.concatenate((zero_channel[0], first_channel[0], second_channel[0])) # Checking Color
Features for Vehicles
f, axes= plt.subplots(4,5, figsize=(20,10)) f.subplots_adjust(hspace=0.5)

for index in range(4):


    vehicle=random.randint(0, len(vehicle_images_original)-1) non_vehicle=random.randint(0,
    len(non_vehicle_images_original)-1)


    coloredImage= cv2.cvtColor(vehicle_images_original[vehicle],cv2.COLOR_RGB2YUV) r,g,b =
    ExtractColorHistogram(coloredImage,128)


    center= FindBinCenter(r) axes[index,0].imshow(vehicle_images_original[vehicle])
    axes[index,0].set_title("Vehicle") axes[index,1].set_xlim(0,256) axes[index,1].bar(center,r[0])
    axes[index,1].set_title("Y")  axes[index,2].set_xlim(0,256) axes[index,2].bar(center,g[0])
    axes[index,2].set_title("U")  axes[index,3].set_xlim(0,256) axes[index,3].bar(center,b[0])
    axes[index,3].set_title("V")axes[index,4].imshow(coloredImage) axes[index,4].set_title("YUV
    colorspace")
```

```
features = ExtractColorFeatures(r,g,b) print("No of features are "+ str(len(features))) # Checking Color

Features for Non Vehicles f, axes= plt.subplots(4,5, figsize=(20,10)) f.subplots_adjust(hspace=0.5)

for index in range(4):

   non_vehicle=random.randint(0, len(non_vehicle_images_original)-1) coloredImage=
cv2.cvtColor(non_vehicle_images_original[non_vehicle],cv2.COLOR_RGB2YUV) r,g,b =

   ExtractColorHistogram(coloredImage)


   center= FindBinCenter(r) axes[index,0].imshow(non_vehicle_images_original[non_vehicle])

   axes[index,0].set_title("Non Vehicle") axes[index,1].set_xlim(0,256)

   axes[index,1].bar(center,r[0]) axes[index,1].set_title("Y") axes[index,2].set_xlim(0,256)

   axes[index,2].bar(center,g[0]) axes[index,2].set_title("U") axes[index,3].set_xlim(0,256)

   axes[index,3].bar(center,b[0]) axes[index,3].set_title("V") axes[index,4].imshow(coloredImage)

   axes[index,4].set_title("YUV colorspace")


#Resizing Image to extract features, so as to reduce the feature vector size def
SpatialBinningFeatures(image,size): image= cv2.resize(image,size)
   return image.ravel() #testing the spatial binning

featureList=SpatialBinningFeatures(vehicle_images_original[1],(16,16))

print("No of features before spatial binning",len(vehicle_images_original[1].ravel())) print("No of features

after spatial binning",len(featureList))

# General method to extact the HOG of the image

def GetFeaturesFromHog(image,orient,cellsPerBlock,pixelsPerCell, visualise= False,
feature_vector_flag=True):

   if(visualise==True):

      hog_features, hog_image = hog(image, orientations=orient, pixels_per_cell=(pixelsPerCell,

                  pixelsPerCell), cells_per_block=(cellsPerBlock, cellsPerBlock), visualise=True,

                  feature_vector=feature_vector_flag)

      return hog_features, hog_image else:

      hog_features = hog(image, orientations=orient, pixels_per_cell=(pixelsPerCell, pixelsPerCell),

                  cells_per_block=(cellsPerBlock, cellsPerBlock), visualise=False,

                  feature_vector=feature_vector_flag)

      return hog_features #testing HOG on test images
```

```
image=vehicle_images_original[1]

image= cv2.cvtColor(image, cv2.COLOR_RGB2YUV) image_channel_0=image[:,:,0]

image_channel_1=image[:,:,0] image_channel_2=image[:,:,0]



feature_0,hog_img_0=GetFeaturesFromHog(image_channel_0,9,2,16,visualise=True,feature_ve

ctor_flag=True)

feature_1,hog_img_1=GetFeaturesFromHog(image_channel_1,9,2,16,visualise=True,feature_ve
ctor_flag=True)

feature_2,hog_img_2=GetFeaturesFromHog(image_channel_2,9,2,16,visualise=True,feature_ve
ctor_flag=True)

f, axes= plt.subplots(1,4,figsize=(20,10)) axes[0].imshow(vehicle_images_original[1])

axes[1].imshow(hog_img_0) axes[2].imshow(hog_img_1) axes[3].imshow(hog_img_2)

print("Feature Vector Length Returned is ",len(feature_0))

print("No of features that can be extracted from image ",len(hog_img_0.ravel()))

#Convert Image Color Space. Note the colorspace parameter is like cv2.COLOR_RGB2YUV def

ConvertImageColorspace(image, colorspace):

   return cv2.cvtColor(image, colorspace)

# Method to extract the features based on the choices as available in step 2

def ExtractFeatures(images,orientation,cellsPerBlock,pixelsPerCell, convertColorspace=False):

   featureList=[]

   imageList=[]

   for image in images: if(convertColorspace==True):

         image= cv2.cvtColor(image, cv2.COLOR_RGB2YUV)


local_features_1=GetFeaturesFromHog(image[:,:,0],orientation,cellsPerBlock,pixelsPerCell, False, True)


local_features_2=GetFeaturesFromHog(image[:,:,1],orientation,cellsPerBlock,pixelsPerCell, False, True)


local_features_3=GetFeaturesFromHog(image[:,:,2],orientation,cellsPerBlock,pixelsPerCell, False, True)

x=np.hstack((local_features_1,local_features_2,local_features_3)) featureList.append(x)
   return featureList

%%time orientations=9 cellsPerBlock=2 pixelsPerBlock=16

convertColorSpace=True
```

```
vehicleFeatures= ExtractFeatures(vehicle_images_original,orientations,cellsPerBlock,pixelsPerBlock,
convertColorSpace)

nonVehicleFeatures=
ExtractFeatures(non_vehicle_images_original,orientations,cellsPerBlock,pixelsPerBlock,
convertColorSpace)

featuresList= np.vstack([vehicleFeatures, nonVehicleFeatures]) print("Shape of features list is ",

featuresList.shape)

labelList= np.concatenate([np.ones(len(vehicleFeatures)), np.zeros(len(nonVehicleFeatures))])

print("Shape of label list is ", labelList.shape)

# train test split of data

from sklearn.model_selection import train_test_split

X_train,  X_test,Y_train, Y_test = train_test_split(featuresList, labelList, test_size=0.2, shuffle=True)

# normalization and scaling

from sklearn.preprocessing import StandardScaler scaler= StandardScaler()

scaler.fit(X_train)

X_train_scaled= scaler.transform(X_train) X_test_scaled= scaler.transform(X_test)

# Use a linear SVC svc = LinearSVC()

# Check the training time for the SVC t = time.time()

svc.fit(X_train, Y_train) t2 = time.time()

print(round(t2-t, 2), 'Seconds to train SVC...') # Check the score of the SVC

print('Test Accuracy of SVC = ', round(svc.score(X_test, Y_test), 4)) # Check the prediction time for a

single sample

t=time.time() #n_predict = 10

y_pred = svc.predict(X_test)

#print('My SVC predicts: ', svc.predict(X_test[0:n_predict])) #print('For these',n_predict, 'labels: ',

Y_test[0:n_predict])

t2 = time.time()

print(round(t2-t, 5), 'Seconds to predict', X_test.shape[0] ,'labels with SVC') # The performance for Testing

data

err_cnt = error_count(Y_test, y_pred, toler_treshold = 5.0) mse = mean_squared_error(Y_test, y_pred)

rmse = np.sqrt(mse)

mae = mean_absolute_error(Y_test, y_pred) print("Error Count : ", err_cnt)

print("Mean Squred Error : %.6f" % mse) print("Root Mean Squred Error : %.6f" % rmse) print("Mean

Absolute Error : %.6f" % mae)
```

```
# function to draw sliding Windows
```

2. Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier

from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
```

```
# Create Decision Tree classifer object clf = DecisionTreeClassifier()

# Train Decision Tree Classifer clf = clf.fit(X_train,Y_train)

#Predict the response for test dataset y_pred = clf.predict(X_test)

# Model Accuracy, how often is the classifier correct? print("Accuracy:",metrics.accuracy_score(Y_test, y_pred))
```