A Project/Dissertation ETE FINAL Report

on

Vehicle Number Plate Recognition using ANPR

Submitted in partial fulfillment of the

requirement for the award of the degree of

Bachelor of Technology – Computer Science and Engineering

Under The Supervision of

Dr. Nitin Mishra

Professor

Submitted By :

Parth Saraogi – 18021011580

Nipoorva Yadav – 18021011652

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY, GREATER NOIDA

INDIA

October, 2021

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project

entitled " Vehicle price prediction using ANPR" in partial fulfillment of the requirements for the award of the BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of JULY-2021 to DECEMBER-2021, under the supervision of Mr.Nitin Mishra, Professor, Department of Computer Science and Engineering of School of Computing Science and Engineering , Galgotias University, Greater Noida.

The matter presented in the project has not been submitted by me/us for the award of any other.

degree of this or any other places.

18SCSE1010348 - PARTH SARAOGI

18SCSE1010421 - NIPOORVA YADAV

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Supervisor

(Mr.NITIN MISHRA, Assistant Professor)

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of 18SCSE1010348 - PARTH SARAOGI, 18SCSE1010421 - NIPOORVA YADAV has been held on _____ and his/her

work is recommended for the award of BACHELOR OF TECHNOLOGY IN COMPUTER

SCIENCE AND ENGINEERING.

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date:

Place:

Abstract

Predicting the value of a car, especially when using a car and not directly from the factory, is an important and important task. With the increase in demand for used cars and a drop of up to 8 percent in demand for new cars in 2013, more and more car buyers are finding new ways to buy new cars directly.

People prefer to buy cars by rent which is a legal agreement between the buyer and the seller. A class seller includes a direct seller or third party, a business company or an insurance company. Under the lease agreement, buyers pay regular installments for the item purchased. These rental installments depend on the value of the vehicle and as a result, honest sellers are interested in knowing the estimated value of their vehicles.

Come up with a way to upgrade the Android system that allows you to see the car plate quickly and easily with a smartphone or tablet camera, the app detects the camera number inside the camera image and uses the information to extract the subscription. Vehicle details with the Regional Transport Office via the web and process this information to predict the current car price.

The License Plate Recognition System essentially consists of four main processes: Image Acquisition, Subtraction of Plate Number, Sorting and Character Recognition. Visual digits are processed to extract registered vehicle details with the e-pravahan agency. Returning information such as Chassis Number, Engine Number, Owner Name, Vehicle Class, Type of Fuel, Car Manufacturer / Model, Registration Towards Expiration Date Insurance.

Issuance of a car price estimation feature. Predicting the price of a car with high accuracy.

Since it is not possible to judge which method is best, different papers, based on the steps mentioned in Fig.1, are evaluated and categorized based on the methods of each method. In each case whenever the parameters available such as speed, accuracy, performance, image size and field are reported. Sales product surveys are beyond the scope of this paper as sometimes these products require more than actual accuracy for advertising purposes. Part of this paper is divided as follows: Section 2 contains a study of different strategies for

obtaining plate numbers. Strategies for classifying characters are reviewed in section 3 and section 4 contains a discussion of ways to identify characters. The report concludes with a discussion of what has not been done and what kind of research is possible in ANPR.

List of Figures

Figure No.	Table name	Page number
1.	UML diagram	7
2.	Data Flow Diagram	8
3.	Classification of images	17
4.	Scanned images	17

Acronyms

ALPR	**Automatic Licence Plate Reader**
ANPR	**Automatic Number Plate Recognition System**
ASCII	**American Standard Code for Information Interchange**
AVR	**Automatic Vehicle Recognition**
CPR	**Car Plate Recognition**
IDE	**Integrated Development Environment**

Contents

Title	Page no.
Abstract	I
List of Tables	II
List of Figures	III
Chapter I. Introduction	
1.1 Introduction	7.
1.2 Formation of Problem	7.
1.2.1 Tools and Technology used	8.
Chapter II. Literature Survey/Project Design	9-13.
Chapter III. Functionality/Working of project	14-15.
Chapter IV. Results and Discussions	16-20.
Chapter V. Conclusion and Future Scope	17.
5.1 Conclusion	17.
Reference	24-25.

Chapter I.

Introduction

With the increase in the number of vehicles in the world, it has become increasingly difficult to find a suitable car park. This problem is particularly important in large organizations and educational institutions such as schools and colleges. And all this leads to nothing more than increased traffic. Still in India, traffic congestion is controlled by traffic police and people themselves are not responsible for their road behavior. The parking lot is still tense with security guards not keeping track of vehicles entering and leaving the building. The whole situation is getting worse due to the increase in traffic.

India is one of the fastest growing economies in the world. The income of the people of central India is growing and thus the number of private cars is increasing.

Formulation of Problem

Although public transport is widely available in India, it is not yet suitable for Indian people. Especially in big cities, public transportation is often congested. Therefore, to travel peacefully people prefer to travel in their own cars. And as a result there are more cars coming to the roads.

Lack of parking is one of the major problems in the traffic congestion in India. Due to the lack of parking spaces people are forced to park their cars in front of buildings that often enter the streets. This leads to a less useful road.

Poor road conditions are also one of the causes. This often leads to repairs and thus reduced usable space.

But with our app we aim to solve this problem. Come up with a way to upgrade the Android system that allows you to see the car plate quickly and easily with a smartphone or tablet camera, the app detects the camera number inside the camera image and uses the information to extract the subscription. Vehicle details with the Regional Transport Office via the web and process this information to predict the current car price.

Parking halls in shopping malls and business areas often face traffic congestion due to overcrowding. Drivers should move

to find available parking without knowing if it is available or not.

Thus the upgraded Car Park Control System can display parking details on the Liquid Crystal Display (LCD) screen. Before using the specified parking space the driver will be given a bar code card. They must scan the card as they enter and exit the parking lot.

At the same time, the system will search for empty parking and a message will be displayed to indicate whether it is available or not. Nowadays the method of displaying information using the LCD screen is not widely used in the parking lot. Usually the information will be displayed using the Light Emitting Diode (LED) screen and is limited as the message displayed is not clear.

In this paper, we present our research in the field of automated parking monitoring system. Within the Computer Vision and Robotics (CoVisBot) research team at UTeM, we aim to develop a parking system monitoring system using ammunition detectors.

Tools and Technology Used

Jupyter Notebook - JupyterLab is a web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design allows for extensions that expand and enrich functionality.

ANPR - Automatic Number Plate Recognition (ANPR) covers **a wide range of camera technology that automatically reads vehicle number plates then records information about that plate**, or uses it to cross-reference elsewhere to set off an 'action'.

OCR - Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example: from a television broadcast).

Smartphone's camera - A camera phone is a mobile phone which is able to capture photographs and often record video using one or more built-in digital cameras. It can also send the resulting image wirelessly and conveniently. The first color commercial camera phone was the Kyocera Visual Phone VP-210, released in Japan in May 1999.

Most camera phones are smaller and simpler than the separate digital cameras. In the smartphone era, the steady sales increase of camera phones caused point-and-shoot camera sales to peak about 2010 and decline thereafter. The concurrent improvement of smartphone camera technology, and its other multifunctional benefits, have led to it gradually replacing compact point-and-shoot cameras.

Modules used:

Module 1 -: Scanner: Scans Image from camera

Module 2-: Plate Recognition: Recognizes Characters from

the image through

API calls.

Module 3-: Fetch Details: Fetch details through API Call.

Module 4 -: User Input: Get the fuel price as input from the user.

Module 5-: Price Prediction: Predict the price and display it to the

Driver Code :

```
%tensorflow_version 2.x
import tensorflow as tf
print(tf.__version__)
```

2.4.1

```
import cv2
import random
import os
import numpy as np
import matplotlib.pyplot as plt
image = cv2.imread("car4.jpg")
image = cv2.imread("car4.jpg")


gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)


 def plot_images(img1, img2, title1="", title2=""):
    fig = plt.figure(figsize=[15,15])
    ax1 = fig.add_subplot(121)
    ax1.imshow(img1, cmap="gray")
    ax1.set(xticks=[], yticks=[], title=title1)

    ax2 = fig.add_subplot(122)
    ax2.imshow(img2, cmap="gray")
    ax2.set(xticks=[], yticks=[], title=title2)
```

In [ ]:

```
plot_images(image, gray)blur = cv2.bilateralFilter(gray, 11,90,
90)plot_images(gray, blur)


edges = cv2.Canny(blur, 30, 200)plot_images(blur, edges)cnts, new =
cv2.findContours(edges.copy(), cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)image_copy = image.copy()_ =
cv2.drawContours(image_copy, cnts, -1, (255,0,255),2)plot_images(image,
image_copy)


 cnts = sorted(cnts, key=cv2.contourArea, reverse=True)[:30]
```

In [ ]:

```
image_copy = image.copy()_ = cv2.drawContours(image_copy, cnts, -1,
(255,0,255),2)plot_images(image, image_copy)plate = Nonefor c in cnts:
    perimeter = cv2.arcLength(c, True)
```

```
    edges_count = cv2.approxPolyDP(c, 0.02 * perimeter, True)
    if len(edges_count) == 4:
        x,y,w,h = cv2.boundingRect(c)
        plate = image[y:y+h, x:x+w]
        break


cv2.imwrite("plate.png",plate)plot_images(plate, plate)



edges1 = cv2.Canny(plate, 30, 200)plot_images(plate, edges1)cnts1, new1 =
cv2.findContours(edges1.copy(), cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)image_copy1 = plate.copy()_ =
cv2.drawContours(image_copy1, cnts, -1, (255,0,255),2)plot_images(plate,
image_copy1)


!sudo apt install tesseract-ocr!pip install pytesseractimport pytesseractimport
shutilimport osimport randomtry:
  from PIL import Imageexcept ImportError:
  import Image
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libnvidia-common-460
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  tesseract-ocr-eng tesseract-ocr-osd
The following NEW packages will be installed:
  tesseract-ocr tesseract-ocr-eng tesseract-ocr-osd
0 upgraded, 3 newly installed, 0 to remove and 34 not upgraded.
Need to get 4,795 kB of archives.
After this operation, 15.8 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tesseract-ocr-en
g all 4.00~git24-0e00fe6-1.2 [1,588 kB]
```

```
Get:2 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tesseract-ocr-os
d all 4.00~git24-0e00fe6-1.2 [2,989 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tesseract-ocr am
d64 4.00~git2288-10f4998a-2 [218 kB]
Fetched 4,795 kB in 1s (3,796 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based fronten
d cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 76, <> li
ne 3.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package tesseract-ocr-eng.
(Reading database ... 160706 files and directories currently installed.)
Preparing to unpack .../tesseract-ocr-eng_4.00~git24-0e00fe6-1.2_all.deb ...
Unpacking tesseract-ocr-eng (4.00~git24-0e00fe6-1.2) ...
Selecting previously unselected package tesseract-ocr-osd.
Preparing to unpack .../tesseract-ocr-osd_4.00~git24-0e00fe6-1.2_all.deb ...
Unpacking tesseract-ocr-osd (4.00~git24-0e00fe6-1.2) ...
Selecting previously unselected package tesseract-ocr.
Preparing to unpack .../tesseract-ocr_4.00~git2288-10f4998a-2_amd64.deb ...
Unpacking tesseract-ocr (4.00~git2288-10f4998a-2) ...
Setting up tesseract-ocr-osd (4.00~git24-0e00fe6-1.2) ...
Setting up tesseract-ocr-eng (4.00~git24-0e00fe6-1.2) ...
Setting up tesseract-ocr (4.00~git2288-10f4998a-2) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Collecting pytesseract
  Downloading https://files.pythonhosted.org/packages/a0/e6/a4e9fc8a93c13185
40e8de6d8d4beb5749b7960388a7c7f27799fc2dd016/pytesseract-0.3.7.tar.gz
Requirement already satisfied: Pillow in /usr/local/lib/python3.7/dist-packages
(from pytesseract) (7.1.2)
Building wheels for collected packages: pytesseract
  Building wheel for pytesseract (setup.py) ... done
```

```
  Created wheel for pytesseract: filename=pytesseract-0.3.7-py2.py3-none-any.
whl size=13945 sha256=580c8b3943e3775194a2e4df6fb21fd50302846b5fcfac
08ee47acaf0c477c00
  Stored in directory: /root/.cache/pip/wheels/81/20/7e/1dd0daad1575d5260916
bb1e9781246430647adaef4b3ca3b3
Successfully built pytesseract
Installing collected packages: pytesseract
Successfully installed pytesseract-0.3.7
                                                                    In [ ]:
 text = pytesseract.image_to_string(plate, lang="eng")print(text)
MH12DE1433


import osimport numpy as np
from sklearn.svm import SVC
from sklearn.model_selection
import cross_val_scorefrom sklearn.externals
import joblibfrom skimage.io
import imreadfrom skimage.filters
import threshold_otsu


letters = [
          '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D',
          'E', 'F', 'G', 'H', 'J', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T',
          'U', 'V', 'W', 'X', 'Y', 'Z'
      ]


 def read_training_data(training_directory):
    image_data = []
    target_data = []
    for each_letter in letters:
       for each in range(10):
          image_path = os.path.join(training_directory, each_letter, each_letter +
'_' + str(each) + '.jpg')
          img_details = imread(image_path, as_gray=True)
          # converts each character image to binary image
          binary_image = img_details < threshold_otsu(img_details)
          flat_bin_image = binary_image.reshape(-1)
```

```
            image_data.append(flat_bin_image)
            target_data.append(each_letter)

    return (np.array(image_data), np.array(target_data))
```

In [ ]:

```
def cross_validation(model, num_of_fold, train_data, train_label):

    accuracy_result = cross_val_score(model, train_data, train_label,
                                cv=num_of_fold)
    print("Cross Validation Result for ", str(num_of_fold), " -fold")

    print(accuracy_result * 100)
```

In [ ]:

```
from google.colab import drivedrive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [ ]:

```
!ls
```

```
drive  sample_data
```

In [ ]:

```
training_dataset_dir="drive/MyDrive/data"
```

In [ ]:

```
training_dataset_dir
=training_dataset_dir+"/train20X20"print(training_dataset_dir)
```

```
drive/MyDrive/data/train20X20
```

In [ ]:

```
image_data, target_data =
read_training_data(training_dataset_dir)print('reading data completed')
```

```
reading data completed
```

In [ ]:

```
svc_model = SVC(kernel='linear', probability=True)
cross_validation(svc_model, 8, image_data, target_data)
print('training model')
svc_model.fit(image_data, target_data)
```

```
Cross Validation Result for  8  -fold
[ 95.34883721  95.34883721 100.         97.6744186  100.
```

```
 95.23809524 100.        100.        ]
training model
```

Out[ ]:

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
    max_iter=-1, probability=True, random_state=None, shrinking=True, tol=0.0
01,
    verbose=False)
```

In [ ]:

```
import pickleprint("model trained.saving model..")filename =
"drive/MyDrive/data"+"/model.sav"pickle.dump(svc_model, open(filename,
'wb'))print("model saved")
model trained.saving model..
model saved
```

In [ ]:

```
##plotting
```

In [ ]:

```
class plotting():
  import matplotlib.pyplot as plt
  def gen_plot(nr,nc):
    fig,axis=plt.subplots(nr,nc)
    return fig,axis

  def plot_car_image(image,fig,axes):
    axes.imshow(image,cmap="gray")



  def add_borders(border,fig,axes):
    axes.add_patch(border)

  def show():
    plt.show()
```

In [ ]:

```
from skimage.io import imreadfrom skimage.filters import
threshold_otsufrom skimage import measurefrom skimage.transform import
```

```
resizeimport matplotlib.patches as patchesimport numpy as npimport
matplotlib.pyplot as plt
```

In [ ]:

```
class Preprocess():
    def __init__(self,image_path):
        self.car_image = imread(image_path, as_gray=True)*255
        threshold_value = threshold_otsu(self.car_image)
        self.binary_car_image = self.car_image > threshold_value
        self.fig,(self.axis,self.axis1)=plotting.gen_plot(2,1)


#LICENSE PLATE DETECTION
    #  heuristics for license plate dimensions
    # height of plate is within [5-20]% of image height and width within [15-
60]% of image_width
    # region area >50
    # plate height more than 20% plate_width
    # sum of pixels in plate greater than 60 % of total pixels
    # only 10 cc in actual plate


    def plate_detection(self):
        label_image = measure.label(self.binary_car_image)
        image_height,  image_width=label_image.shape

plate_dim=(0.05*image_height,0.2*image_height,0.15*image_width,0.6*ima
ge_width)

        plotting.plot_car_image(self.car_image,self.fig,self.axis)

        self.lp_cands=[]
        self.lp_cand_dimension=[]
        for region in measure.regionprops(label_image):
            minRow, minCol, maxRow, maxCol = region.bbox
            (region_height,region_width)=(maxRow-minRow,maxCol-minCol)
```

```
        if(region.area < 50 or region_height<0.2*region_width ):
            continue

candidate=np.invert(self.binary_car_image[minRow:maxRow,minCol:maxCol]
)
        if(region_height>=plate_dim[0] and region_height <=plate_dim[1]
and region_width>=plate_dim[2] and region_width<= plate_dim[3]):

            if self.elimininate_candidate(candidate):
                continue

            rectBorder = patches.Rectangle((minCol, minRow), maxCol-minCol,
maxRow-minRow, edgecolor="red", linewidth=2, fill=False)
            self.lp_cands.append(candidate)
            self.lp_cand_dimension.append(((minRow,minCol),(maxRow-
minRow,maxCol-minCol)))
            plotting.add_borders(rectBorder,self.fig,self.axis)

    def elimininate_candidate(self,candidate):
        r,c=candidate.shape
        return np.sum(candidate) > 0.3*r*c

## CHARACTER SEGMENTATION  ## Heuristics for character## character
height [35-90]%  and width [2-10]%

    def character_segmentation(self):
        segmented_characters=[]
        idx=0
        for idx in range(len(self.lp_cands)):
            cand=self.lp_cands[idx]
            plotting.plot_car_image(cand,self.fig,self.axis1)
            char_dim = (0.30*cand.shape[0], 0.90*cand.shape[0],
0.02*cand.shape[1], 0.1*cand.shape[1])

            labelled_cand = measure.label(cand)
```

```
        cnt=0
        border=[]
        temp_chars=[]
        for region in measure.regionprops(labelled_cand):
            minRow, minCol, maxRow, maxCol = region.bbox
            (region_height,region_width)=(maxRow-minRow,maxCol-minCol)
            if(maxRow==self.lp_cand_dimension[idx][1][0]):
                continue
            #print(region_height,region_width)
            if(region_height>=char_dim[0] and region_height <=char_dim[1]
and region_width>=char_dim[2] and region_width<= char_dim[3]):
                rectBorder = patches.Rectangle((minCol, minRow), maxCol-
minCol, maxRow-minRow, edgecolor="red", linewidth=2, fill=False)
                border.append(rectBorder)
                temp_chars.append((minRow,maxRow,minCol,maxCol))
                plotting.add_borders(rectBorder,self.fig,self.axis1)



        if(len(border)==10):
            for borders in border:
                plotting.add_borders(borders,self.fig,self.axis1)
            dim=self.lp_cand_dimension[idx]

            for val in temp_chars:
                r1=dim[0][0]+val[0]
                r2=dim[0][0]+val[1]
                c1=dim[0][1]+val[2]
                c2=dim[0][1]+val[3]

segmented_characters.append((val[2],resize(np.invert(self.binary_car_image[r
1:r2,c1:c2]),(20,20))))
        return segmented_characters
```

In [ ]:

```
if __name__=="__main__":
    inp="drive/MyDrive/data/test_image/car4.jpg"
```

```
    env=Preprocess(inp)
    env.plate_detection()
    segmented_characters=env.character_segmentation()

    plotting.show()
```

##Prediction

```
import plottingimport matplotlib.pyplot as pltimport pickleimport osimport sys
                                                                    In [ ]:
model_pat="drive/MyDrive/data"+"/model.sav"
                                                                    In [ ]:
model= pickle.load(open(model_pat,"rb"))
                                                                    In [ ]:
segmented_characters.sort()
                                                                    In [ ]:
segmented_characters


ans=[]for char in segmented_characters:
   #print(plt.imshow(char[1]))
   ans.append(model.predict(char[1].reshape(1,-1)))license_plate= []for val in
ans:
   license_plate.append(val[0])

for idx in range(len(license_plate)):
   if(idx==0 or idx==1 or idx==4 or idx==5):
      if(license_plate[idx]=='0'):
         license_plate[idx]=str('O')
      elif(license_plate[idx]=='1'):
         license_plate[idx]=str('I')
      elif(license_plate[idx]=='2'):
         license_plate[idx]='Z'
   else:
      if(license_plate[idx]=='O'):
         license_plate[idx]='0'
      elif(license_plate[idx]=='I'):
         license_plate[idx]='1'
```

```
        elif(license_plate[idx]=='Z'):
            license_plate[idx]=str('2')
  license_plate="".join(license_plate)print("Recognized License Plate
is:")print(license_plate)
```

Chapter II.

Literature Review

The process of keeping track of vehicle details is not new. From time immemorial, if there are a lot of vehicles, someone should track them so that they do not get lost or easily found whenever needed. Guards have been hired, or someone else for this purpose.

According to [1] - ANPR uses visual acuity (OCR) in camera images. When Dutch car registration plates switched to a different style in 2002, another change was made to the font, with smaller spaces added to other characters (such as P and R) to make them more distinctive and easy to read in such programs. Some license plate settings use different font sizes and shapes - ANPR systems must be able to deal with such variations in order to be truly effective. The most complex systems can handle international diversity, or most programs are designed for each country.

According to [2] - Used cameras can be either street cameras or shut down television cameras, as well as mobile units, which are usually attached to vehicles. Some systems use infrared cameras to take a clear picture of the plates.

The software feature runs on home computer hardware and may be connected to other applications or data. It first uses a series of image tricks to locate, identify and enhance a number image, and then detects a character (OCR) to extract license plate numbers. ANPR programs usually distribute in one of two basic ways: one allows the whole process to take place in a real-time environment, while the other transfers all images from multiple channels to a remote computer and performs OCR process there over time. When done in line location, the information entered on the alphanumeric plate, time of day, route identification, and other required information is completed in approximately 250 milliseconds. This information can be easily transferred to a remote computer for processing and, if necessary, or stored online for later retrieval. In another setting, there are a large number of PCs used on a server farm to handle a large number of tasks, such as those found in a London congestion project. Often in such programs, there is a need to transfer images to a remote server, and this may require large media bandwidth transfer.

Determining the price on a list of used cars is a daunting task, given the many factors that drive the price of a used car on the market. The focus of this project is to develop machine learning models that can accurately predict the number of used vehicles based on its features, so you can buy an expert. We use and evaluate various learning methods in a database that includes sales figures of different types and models across cities in the United States. Our results show that the Random Forest model and the K-Means merge with the reversal of the line bring the best results, but it survives. Regular line breakdown has also produced satisfactory results, with the benefit of a much shorter training time compared to the methods mentioned above. Inspiration Determining whether a used car is worth the amount posted when you see a listing online can be difficult. Few items, including mile, shape, model, year, etc. they can affect the real value of a car. From a dealer's point of view, it is also a problem to call a used car the right way. Based on the available data, the aim is to use machine learning algorithms to improve the pricing models of used cars. Data set In this project, we are using a database for the sale of used cars from all over the United States, located at Kaggle. Features found in this database are Mileage, VIN, Make, Model, Year, State and City. Pre-processing To better understand data, we have created a data histogram. We noticed that the database had a lot of output, mainly due to the sensitivity of the large numbers of used vehicles. In general, the most recent and low-end models sold at a premium, however, there were many data points that did not match this. This is because the history of the accident and the situation can have a profound effect on the value of the vehicle. Since we did not have access to vehicle history and status, we set up our database into three standard deviations to export. We have transformed Make, Model and State into hot vectors. Having more than 2000 different cities in the database, we have established a boolean city unit that was set up when the population of the city was above a certain limit i.e. the metropolitan area.

Automatic number-plate recognition (ANPR) is a technology used to identify bright letters in pictures to read car registration plates.

Rapid technological advances have allowed DaHua (for example) to enable camera integration of ANPR technology. This means that it can be widely used to improve efficiency and security in a variety of solutions. In addition, the ANPR camera side is more expensive than standard server analytics solutions.

Automated ANPR is an essential solution for many applications, including:

• Road Safety Act, Automatically detects speed violations and red light

• Compliance with Law, Vehicle Licensing and Registration. Provides an automatic warning when vehicles with the restricted list pass

• Electronic Toll Collection, Enables free flow and holds car license plate number

• Parking Guide, Automatically view available parking spaces and direct cars to the best route

• Access and Exit Control, Raise restricted vehicle restrictions

Through the use of high-speed image with supporting light, the detection of characters within the given images, the confirmation of the sequence of characters such as those from a car license plate, the attention of the characters to convert the image into text; therefore it ends with a set of metadata that identifies an image that contains the vehicle's license number and the text associated with the code of that plate.

One of the most common tasks in computer screening and image processing is the recognition of an image or object in an image. Among these functions, OCR is a popular research topic. The OCR aims to enable computers to detect visual signals without human intervention. This is achieved by searching

similarities between the features extracted from the icon image and the image model library. Computer systems installed in such an OCR system improve installation speed, reduce some human errors and enable integrated storage, faster retrieval and other tricky files The OCR system is used in many applications such as 3D object recognition, automatic guidance.

vehicles (AGV), digital libraries, invoices and receipt processing, and more recently the personal digital assistant (PDA). The OCR covers important pattern problems

recognition, common in other related topics, for example the image retrieval system.

We have made car price estimates based on historical data available

collected in daily newspapers. They used supervised machine learning methods to predict the price of cars. Many other algorithms such as multiple line loops, algorithms for the closest neighbor, naive

based, and other decision tree algorithms have also been used. All four algorithms are compared and available for the best prediction algorithm. They face some difficulty in comparing the algorithms, which in some way they hold.

In the present system, the pricing of both two-wheeled and four-wheeled vehicles, multiple data mining algorithms and machine learning algorithms were widely used. The main effect of this existing system is that they need additional features to predict the value of the vehicle.

In accordance to [4] - Additional comparative methods must be used to achieve the desired result. It is very difficult to find enough data sets that are widely distributed around the world. Data sets can only be collected online. But not in offline mode. It is not possible for everyone to collect data sets using Internet mode especially in rural areas.

Data sets will not contain vehicles that have not been used for a long time and even standard model vehicles may or may not be included in the data sets. The main barriers to the existing system The system is very slow due to many functions related to keyword query analyzing individual points, and is not suitable for many programs that require analysis of different car points groups. There are no quick and low return queries due to SVM shortages under Constraints.

As mentioned in [7] - The most important thing in this paper is to scan the vehicle number. And this is done using a smartphone camera. A smartphone camera can have very low resolution but still we want the algorithm to work properly without any problem. The license plate number is a metal plate mounted on the front and rear of the vehicle for official identification purposes. The registration code is a number or alphanumeric that identifies a vehicle within a specific output region. In some countries, the registration code varies from country to country, and in others it varies from place to place. As each country follows its own car plate format rules. One can classify them based on the background - background color, size (one or two lines) and text type (depending on language). The front colors of the plate numbers can be black, white, blue and red while the background colors can be black, white, and yellow depending on the laws of the country. For example in India, black letters are

used for commercial or non-commercial vehicles and the white letter represents foreign embassies.

The size of a number plate depends on whether the letters and / or number are one or two lines long. The language of number plates is English or a specific country. For example, India uses the English language while Iran and Japan use Persian and Japanese respectively. The number plates of most countries can include two or three types of

letters, such as English and numeric letters on the Indian number plate and Chinese characters, English, and the numbers on the Chinese number plate.

Many numerical algorithms fall into more than one category based on different strategies. Determining the license plate number should take into account the following factors:

(1) .Pat size: plate may have a different size in car image.

(2) .Loc: The plate can be found anywhere in the car.

(3) .Bear plate: The plate may have different background colors depending on the type of vehicle. For example a government vehicle number may have a different domain than other public vehicles.

(4) .Slover: The plate may have a screw and may not be considered a character.

The plate number can be extracted using the image separator method. There are many ways to separate the images found in various books. Many methods are used to make binary image. Some writers use Otsu's method of making a picture frame to turn a colored image into a gray scale. Some plate separation algorithms are based on color separation. Location license plate research based on color separation. The following sections describe the most common numerical methods, followed by a detailed discussion of the image separation techniques used in various ANPR or LPR publications.

Chapter III.

Functionality/Working of Project

(UML Diagrams)

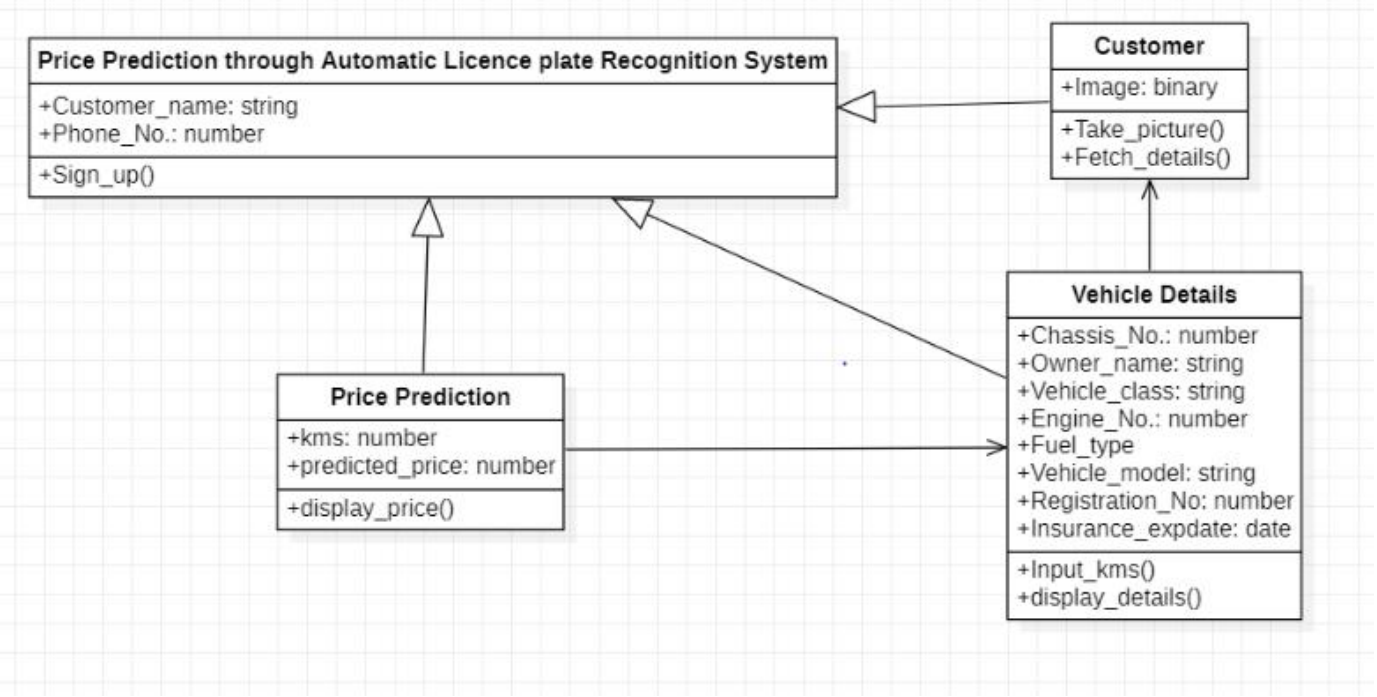

Fig. no. 1 UML Diagram

There are 4 entities created that are basically tables that will be created in the database. They are basically as follows –

Price Prediction through Automatic Licence plate Recognition system – This will include the basic details of the customer like customer name which will be of type string.

Customer phone number that will be of type number.

The methods that will be taking the use of this entity will be –

Sign_up()

Customer – The customer entity takes in the image that will be of type binary.

The methods that will be taking use of this entity will be –

Take_pictures(), Fetch_details().

Price Prediction – This entity takes in the number of kilometres the vehicle has driven that will be of type number and the predicted price.

Vehicle Details – This entity takes in the vehicle details that includes the vehicle no., chassis no., engine no., owner name, vehicle class, fuel type, vehicle model, registration no., and insurance expiry date. The methods that will be taking the use of this entity will be –

Input_kms() and display_details().

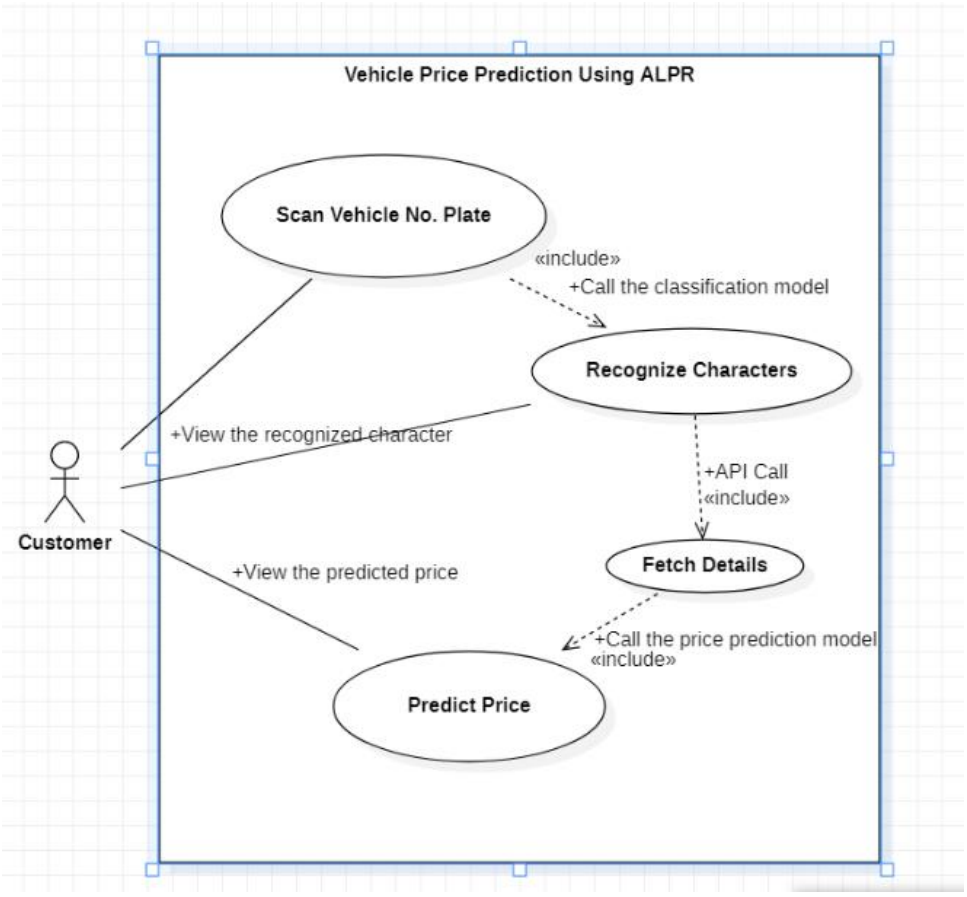

Fig. no. 2 Data Flow diagram

Here are some of the algorithms that are used in the working of this model -

Connected Component Analysis (CCA)

CCA or blob extraction is a method of labeling a separate set of small sets of connected parts based on a given heuristic. Scan the binary image and record pixels as an individual connection to current pixels like North-East, North, North-West pixel current (8-connection). 4- The connection is only used for the north and west neighbor of the current pixel. The algorithm provides better performance and is very useful for automatic image analysis. This method can be used to separate plates and separate letters.

Mathematics morphology

Mathematical morphology is based on established theory, lattice theory, topology, and random functions. It is often used in digital photography but can also be used in other local architecture. It was originally designed to process binary images and was expanded to process gray scale functions and images. Contains basic operators like erosion, extension, opening, closing.

Character Recognition

As discussed in Section 2, alphabetical identification helps to identify and translate image text into edited text. As many algorithms detect numeric numbers using a single letter identification method. In this section, each method is described.

Artificial Neural Network

The Artificial Neural Network (ANN) is sometimes known as the mathematical neural network, which contains synthetic neurons. Several algorithms such as [5], [6], [7], [8] are based on ANN.

In [5] A two-layer neural probabilistic network with a topology of 180-180-36. The character recognition process is performed at 128ms. In [6] multi-layered perceptron (MLP) model ANN is used for character classification. Contains a set of input decision-making entries, a hidden input component for complex organizations and a decision-making framework. The Feed forward back-propagation (BP) algorithm was used for training

Chapter IV.

Results and Discussions

The code first scans the image of the number plate. It can classify the images into four different parts and and accordingly we can see the how the ML model will treat the images so that it can scan the numbers that are written. It is very important to differentiate the car chassis with the numbers that are written.

Fig. no. 3 It shows the classification of the images according to the model

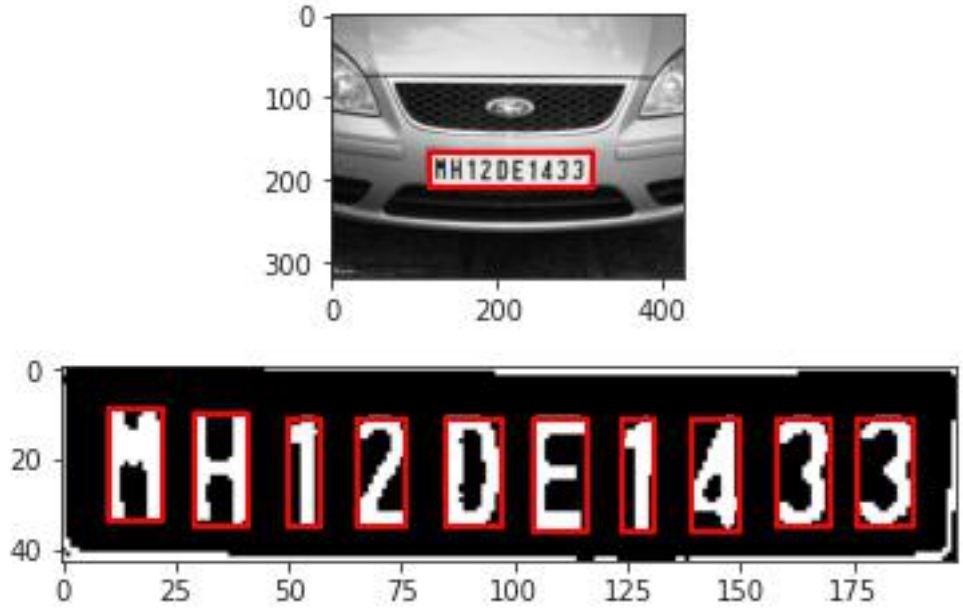

Fig. no. 4 Scanned image shows the number in the number plate

Performing Image Binarization

Image Binarization is the process of converting an image to black and white. In this method, a certain limit is chosen to distinguish certain black pixels from certain pixels as white. But the biggest problem is how to choose the right threshold value for a particular image. Sometimes it is very difficult or impossible to choose the right threshold value. Adaptive Thresholding can be used to overcome this problem. The threshold can be selected by the user manually or it can be selected by the default algorithm known as the default threshold.

Edge discovery

Edge discovery is an important way to find a feature or remove an element. Usually the result of using an algorithm acquisition edge is the boundary of an

object with connected curves. It is very difficult to apply this method to complex images as it may result in a borderline of unconnected objects. Various algorithms / operators such as Canny, Canny-Deriche, Differential, Sobel, Prewitt and Roberts Cross are used for edge detection.

Hough Transform

It is a method of extracting a feature originally used for line detection. Later it has been expanded to accommodate a sloping shape such as a circle or oval. The original algorithm was generally developed by D.H. Ballard

Blob Detection

Blob detection is used to detect different points or regions in light or color compared to the surrounding area. The main purpose of using this method is to find easily accessible circuits that can be detected by the edge detection or existing algorithms. Other common blob detectors are Laplacian of Gaussian (LoG), Difference of Gaussians (DoG), Determinant of Hessian (DoH), strongest regions and a circuit detector based on Principle curvature.

Output :

Recognized License Plate is:

MH12DE1433

Chapter V.

Conclusion

Therefore, the conclusion to this is that we can use the proposed model whenever someone needs to get the details of a car, let alone the vehicle number or engine number. or chassis number, etc.

The main purpose of the study was to develop a system that could reflect current parking knowledge in the parking lot. Information shows include available space and total car parking space. The system works with hardware integration such as OCR and webcam. Based on the test result, the system passes the test with a 100% success rate. It proves that the system can be used in the real situation of parking cars especially in the indoor environment. To minimize system control personnel, a few sensors can be installed to connect to the system to detect the entry and exit vehicle. With sensor installation, as expected, the system can run automatically without the control of the controller.

It is clear that ANPR is a complex system due to the different number of categories and it is currently not possible to achieve 100% complete accuracy as each category is based on the previous category. Factors such as different lighting conditions, vehicle shadow and unusual size of number letters, distinctive font and background color affect ANPR performance. Some systems work only in these limited cases and may not produce the right amount of accuracy in extreme cases. Some of the programs are being developed and implemented in a particular country, summarized in Table 3. Plans that can be discussed in the country are not included in Table 3. As in Figure 3, it is clear that there are very few ANPRs. . There is therefore a wide range of development plans similar to the one in India.

This paper provides an in-depth study of the latest developments and future trends in ANPR, which can be useful to researchers who have contributed to such improvements.

Predictability performance should be enhanced through data purification processes. But in this paper, an inadequate set of complex data is the result here. We will get only 50 percent result in using a single machine algorithm.

Therefore, we have proposed several teams of machine learning algorithm to obtain additional accuracy and obtain 93 percent efficiency. This one-group comparisons with multiple machine learning algorithms are important. It also overcomes the drawback of the single machine algorithm provided for the proposed system.

Although, this program has achieved significant performance in vehicle price prediction, our aim for future work is to test this system to work effectively with various data sets.

References

1. Abhishek Kashyap, B. Suresh, Anukul Patil, Saksham Sharma, Ankit Jaiswal,Soma Biswas. “Automatic Number Plate Recognition” ISBN: 978-1-5386-4119-4/18/$31.00 2018 IEEE.

2. Used Car Price Prediction using K-Nearest Neighbor Based Model By-K. Samruddhi, Dr R.Ashok KumarVol. 4 (3), September 2020, www.ijirase.com

3. Vishal Jain, Zitha Sasindran, Anoop Rajagopal, Soma Biswas, Harish S Bharadwaj, K R Ramakrishnan.

   “Deep Automatic Licence Plate Recognition system” 2016 ACM. ISBN 978-1-4503-4753-2/16/12

4. S.E.Viswapriya, Durbaka Sai Sandeep Sharma, Gandavarapu Sathya kiran.

   ”Vehicle Price Prediction using SVM Techniques”.

   International Journal of Innovative Technology and Exploring Engineering (IJITEE)

   ISSN: 2278-3075, Volume-9 Issue-8, June 2020

5. S.Kranthi, K.Pranathi, A.Srisaila .“Automatic Number Plate Recognition”

   Vol 2, No 3 (July 2011) ©IJoAT International Journal of Advancements in Technology

6. International Journal of Computer Applications (0975 – 8887) Volume 69– No.9, May 2013 21 Automatic Number Plate Recognition System (ANPR): A Survey by Dipti Shah, PhD. , Chirag Patel ,Atul Patel, PhD

7. Nitish Monburinon, Prajak Chertchom, Thongchai Kaewkiriya, Suwat Rungpheung, Sabir Buya, Pitchayakit Boonpou. “Prediction of Prices for Used Car by Using Regression Models” ISBN:978-1-5386-5254-1/18/$31.00 2018 IEEE

8. Tarun Kumar Dept. of CSE MNNIT, Suraj Gupta Dept. of CE NSIT Delhi An Efficient Approach for “Automatic Number Plate Recognition for Low Resolution Images”