A Project Report

on

Automatic License Number Plate Recognition using Deep Learning

Submitted in partial fulfillment of the

requirement for the award of the degree of

BTECH IN COMPUTER SCIENCE AND ENGINEERING SPECILIZATION IN CLOUD COMPUTING AND VIRTULIZATION

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of Mr. Mukesh Kumar Jha

Submitted By

JAVED AHMAD

KHAN

(19SCSE1050005 / 19021050098)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERINGGALGOTIAS UNIVERSITY, GREATER NOIDA

INDIA

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"Automatic License Number Plate Recognition using Deep Learning"** in partial fulfillment of the requirements for the award of the **B.Tech CS Cloud Computing and Virtualization** submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of month, Year to Month and Year, under the supervision of Name… Designation, Department of Computer Science and Engineering/Computer Application and Information and Science, of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

JAVED AHMAD KHAN/ 19SCSE1050005

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

Supervisor Name
Designation

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of JAVED AHMAD KHAN 19SCSE1050005 has been held on _____ and his work is recommended for the award of **B.Tech CS Cloud Computing and Virtualization**

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

Signature of Project Coordinator **Signature of Dean**

Date: December, 2021
Place: Greater Noida

ABSTRACT

In this era of fast growing technologies, there is a huge demand among the people for a secure lifestyle and travelling. In the past decade, the number of vehicles on road has been increased. Tracking of individual vehicle becomes a very challenging task with the massive growth in the vehicular sector every day. This paper suggests an automated vehicle tracking system for the fast moving vehicles with the help of the surveillance cameras on the roadside. The process of getting CCTV footage in the real time background is very tedious process. To cater to this problem, an efficient deep learning model such as You Only Look Once (YOLO) is used for object detection. The proposed work consists of four main steps. In the first step, video footage is converted into images and the car is detected from each of the frames. In the next step, license plate is detected from the detected cars. In the final step, the number plate characters reading are recognized from the detected number plates. The proposed deep learning model uses ImageAI library to make the training process easier. Tamil Nadu license plate images are used to analyse the performance of the model. The accuracy of 97% is achieved for car detection, accuracy of 98% is achieved for number plate localization and accuracy of 90% achieved for character recognition.

TABLE OF CONTENTS

1. INTRODUCTION

1.1. INTRODUCTION

The drastic increase in the vehicular traffic on the roadways stimulates a huge demand in the technology for traffic monitoring and management. In this scenario, manual tracking of vehicles running fast on the road is practically not feasible. There will be wastage of man power and time. Even if it is operated manually, that will reflect huge difficulties and enormous errors. There are already available solutions for tracking the vehicles and number plates using machine learning algorithms. But in real time, these algorithms literally fail due to its complexity for processing in real time background. Hence there is an instantaneous necessity to develop an automatic system that will help tracking the vehicles by tracing their number plates in a most efficient way. In this paper, two CNN models are used. Hence two datasets consisting of car images and number plate images are required. For training the car images, Stanford cars dataset from the internet is used. For number plate images, customized dataset was created with the help of internet source and by taking pictures of the cars around. Once the data is obtained, it must be split into train and test images and annotated to the machine readable form.

Formulation of Problem

- To avoid the car theft
- To Prevent the uses of fake number plates on cars
- This can also be used to track any car from control center
- To fine a overspeeding or rash driving`

2. REQUIREMENTS

OpenCV is an open-source machine learning library and provides a common infrastructure for computer vision. Whereas Pytesseract is a Tesseract-OCR Engine to read image types and extract the information present in the image.

Install OpenCV and Pytesseract pip3 python package:

pip3 install opencv-python
pip3 install pytesseract
In this python project, to identify the number plate in the input image, we will use following features of openCV:

- **Gaussian Blur:** Here we use a Gaussian kernel to smoothen the image. This technique is highly effective to remove Gaussian noise. OpenCV provides a cv2.GaussianBlur() function for this task.

- **Sobel:** Here we calculate the derivatives from the image. This feature is important for many computer vision tasks. Using derivatives we calculate the gradients, and a high change in gradient indicates a major change in the image. OpenCV provides a cv2.Sobel() function to calculate Sobel operators.

- **Morphological Transformation**: These are the operations based on image shapes and are performed on binary images. The basic morphological operations are Erosion, Dilation, Opening, Closing. The different functions provided in OpenCV are:
  - cv2.erode()
  - cv2.dilate()
  - cv2.morphologyEx()

- **Contours:** Contours are the curves containing all the continuous points of same intensity. These are very useful tools for object recognition. OpenCV provides cv2.findContours() functions for this feature.

Now, let's dive into the number plate recognition code. Follow the steps below:

1. **Imports:**

For this project we need numpy and pillow python libraries with openCV and pytesseract

```
import numpy as np
import cv2
from PIL import Image
import pytesseract as tess
```

2. Now we will define three functions, to find the unnecessary contours that openCV may identify but it does not have probability of being a number plate.

2.1. The first function to check the area range and width-height ratio:

```
def ratioCheck(area, width, height):
ratio = float(width) / float(height)
if ratio < 1:
ratio = 1 / ratio
if (area < 1063.62 or area > 73862.5) or (ratio < 3 or ratio > 6):
return False
return True
```

2.2. The second function to check average of image matrix:

```
def isMaxWhite(plate):
avg = np.mean(plate)
if(avg>=115):
return True
else:
return False
```

2.3. The third function to check the rotation of contours:

```
def ratio_and_rotation(rect):
(x, y), (width, height), rect_angle = rect
if(width>height):
angle = -rect_angle
else:
angle = 90 + rect_angle
if angle>15:
return False
if height == 0 or width == 0:
return False
area = height*width
```

```
if not ratioCheck(area,width,height):
return False
else:
return True
```

3. Now we will write a function to clean the identified number plate for preprocessing before feeding to pytesseract:

```
def clean2_plate(plate):
gray_img = cv2.cvtColor(plate, cv2.COLOR_BGR2GRAY)
_, thresh = cv2.threshold(gray_img, 110, 255, cv2.THRESH_BINARY)
if cv2.waitKey(0) & 0xff == ord('q'):
pass
num_contours,hierarchy = cv2.findContours(thresh.copy(),cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_NONE)
if num_contours:
contour_area = [cv2.contourArea(c) for c in num_contours]
max_cntr_index = np.argmax(contour_area)
max_cnt = num_contours[max_cntr_index]
max_cntArea = contour_area[max_cntr_index]
x,y,w,h = cv2.boundingRect(max_cnt)
if not ratioCheck(max_cntArea,w,h):
return plate,None
final_img = thresh[y:y+h, x:x+w]
return final_img,[x,y,w,h]
else:
return plate, None
```

4. In this step, we will take an image input. We will perform Gaussian Blur, Sobel and morphological operations. After we find contours in the image and loop through each contour to identify the number plate. We will then clean the image contour and feed it to pytesseract to recognize the number and characters.

```
img = cv2.imread("testData/sample15.jpg")
print("Number input image...",)
cv2.imshow("input",img)
if cv2.waitKey(0) & 0xff == ord('q'):
pass
img2 = cv2.GaussianBlur(img, (3,3), 0)
img2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
img2 = cv2.Sobel(img2,cv2.CV_8U,1,0,ksize=3)
_,img2 = cv2.threshold(img2,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
```

```
element = cv2.getStructuringElement(shape=cv2.MORPH_RECT, ksize=(17, 3))
morph_img_threshold = img2.copy()
cv2.morphologyEx(src=img2, op=cv2.MORPH_CLOSE, kernel=element,
dst=morph_img_threshold)
num_contours, hierarchy=
cv2.findContours(morph_img_threshold,mode=cv2.RETR_EXTERNAL,method=cv2.C
HAIN_APPROX_NONE)
cv2.drawContours(img2, num_contours, -1, (0,255,0), 1)
for i,cnt in enumerate(num_contours):
min_rect = cv2.minAreaRect(cnt)
if ratio_and_rotation(min_rect):
x,y,w,h = cv2.boundingRect(cnt)
plate_img = img[y:y+h,x:x+w]
print("Number identified number plate...")
cv2.imshow("num plate image",plate_img)
if cv2.waitKey(0) & 0xff == ord('q'):
pass
if(isMaxWhite(plate_img)):
clean_plate, rect = clean2_plate(plate_img)
if rect:
fg=0
x1,y1,w1,h1 = rect
x,y,w,h = x+x1,y+y1,w1,h1
# cv2.imwrite("clena.png",clean_plate)
plate_im = Image.fromarray(clean_plate)
text = tess.image_to_string(plate_im, lang='eng')
print("Number Detected Plate Text : ",text)
```

Code for Project GUI

Make a new file gui.py

```
import tkinter as tk
from tkinter import filedialog
from tkinter import *
from PIL import ImageTk, Image
from tkinter import PhotoImage
import numpy as np
import cv2
import pytesseract as tess
def clean2_plate(plate):
```

```
gray_img = cv2.cvtColor(plate, cv2.COLOR_BGR2GRAY)
_, thresh = cv2.threshold(gray_img, 110, 255, cv2.THRESH_BINARY)
num_contours,hierarchy = cv2.findContours(thresh.copy(),cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_NONE)
if num_contours:
contour_area = [cv2.contourArea(c) for c in num_contours]
max_cntr_index = np.argmax(contour_area)
max_cnt = num_contours[max_cntr_index]
max_cntArea = contour_area[max_cntr_index]
x,y,w,h = cv2.boundingRect(max_cnt)
if not ratioCheck(max_cntArea,w,h):
return plate,None
final_img = thresh[y:y+h, x:x+w]
return final_img,[x,y,w,h]
else:
return plate,None
def ratioCheck(area, width, height):
ratio = float(width) / float(height)
if ratio < 1:
ratio = 1 / ratio
if (area < 1063.62 or area > 73862.5) or (ratio < 3 or ratio > 6):
return False
return True
def isMaxWhite(plate):
avg = np.mean(plate)
if(avg>=115):
return True
else:
return False
def ratio_and_rotation(rect):
(x, y), (width, height), rect_angle = rect
if(width>height):
angle = -rect_angle
else:
angle = 90 + rect_angle
if angle>15:
return False
if height == 0 or width == 0:
return False
area = height*width
```

```
if not ratioCheck(area,width,height):
return False
else:
return True
top=tk.Tk()
top.geometry('900x700')
top.title('Number Plate Recognition')
top.iconphoto(True, PhotoImage(file="/home/shivam/Dataflair/Keras
Projects_CIFAR/GUI/logo.png"))
img = ImageTk.PhotoImage(Image.open("logo.png"))
top.configure(background='#CDCDCD')
label=Label(top,background='#CDCDCD', font=('arial',35,'bold'))
# label.grid(row=0,column=1)
sign_image = Label(top,bd=10)
plate_image=Label(top,bd=10)
def classify(file_path):
res_text=[0]
res_img=[0]
img = cv2.imread(file_path)
img2 = cv2.GaussianBlur(img, (3,3), 0)
img2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
img2 = cv2.Sobel(img2,cv2.CV_8U,1,0,ksize=3)
_,img2 = cv2.threshold(img2,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
element = cv2.getStructuringElement(shape=cv2.MORPH_RECT, ksize=(17, 3))
morph_img_threshold = img2.copy()
cv2.morphologyEx(src=img2, op=cv2.MORPH_CLOSE, kernel=element,
dst=morph_img_threshold)
num_contours, hierarchy=
cv2.findContours(morph_img_threshold,mode=cv2.RETR_EXTERNAL,method=cv2.C
HAIN_APPROX_NONE)
cv2.drawContours(img2, num_contours, -1, (0,255,0), 1)
for i,cnt in enumerate(num_contours):
min_rect = cv2.minAreaRect(cnt)
if ratio_and_rotation(min_rect):
x,y,w,h = cv2.boundingRect(cnt)
plate_img = img[y:y+h,x:x+w]
print("Number identified number plate...")
res_img[0]=plate_img
cv2.imwrite("result.png",plate_img)
if(isMaxWhite(plate_img)):
```

```
clean_plate, rect = clean2_plate(plate_img)
if rect:
fg=0
x1,y1,w1,h1 = rect
x,y,w,h = x+x1,y+y1,w1,h1
plate_im = Image.fromarray(clean_plate)
text = tess.image_to_string(plate_im, lang='eng')
res_text[0]=text
if text:
break
label.configure(foreground='#011638', text=res_text[0])
uploaded=Image.open("result.png")
im=ImageTk.PhotoImage(uploaded)
plate_image.configure(image=im)
plate_image.image=im
plate_image.pack()
plate_image.place(x=560,y=320)
def show_classify_button(file_path):
classify_b=Button(top,text="Classify Image",command=lambda:
classify(file_path),padx=10,pady=5)
classify_b.configure(background='#364156', foreground='white',font=('arial',15,'bold'))
classify_b.place(x=490,y=550)
def upload_image():
try:
file_path=filedialog.askopenfilename()
uploaded=Image.open(file_path)
uploaded.thumbnail(((top.winfo_width()/2.25),(top.winfo_height()/2.25)))
im=ImageTk.PhotoImage(uploaded)
sign_image.configure(image=im)
sign_image.image=im
label.configure(text='')
show_classify_button(file_path)
except:
pass
upload=Button(top,text="Upload an image",command=upload_image,padx=10,pady=5)
upload.configure(background='#364156', foreground='white',font=('arial',15,'bold'))
upload.pack()
upload.place(x=210,y=550)
sign_image.pack()
sign_image.place(x=70,y=200)
```

```
label.pack()
label.place(x=500,y=220)
heading = Label(top,image=img)
heading.configure(background='#CDCDCD',foreground='#364156')
heading.pack()
top.mainloop()
```

2.1. Tool and Technology Used

- Programming Languages
  - Python3
  - JavaScript
- IDE
  - Google colab
  - Jupyter Notebook
  - Visual Studio Code
- Cloud Services
  - Google Cloud Platform
- Version Control
  - Git
- Programming Frameworks
  - PyTorch
  - Django

SYSTEM REQUIREMENT

- An Nvidia GPU with at least 2 GB of RAM (FakeApp doesn't support AMD atm)
- An i3 or AMD 9 processor

- 8 GB of RAM/12 GB RAM (if we want better performance)
- 20 GB of free hard drive space

SUPPORTED OPERATING SYSTEM

- **Windows 10** Windows 7 and 8 might work. Your milage may vary
- **Linux** Most Ubuntu/Debian or CentOS based Linux distributions will work.
- **MacOS** GPU support on macOS is limited due to lack of drivers/libraries from Nvidia.

HARDWARE REQUIREMENTS

A POWERFUL CPU

- Laptop CPUs can often run the software, but will not be fast enough to train at reasonable speeds

A POWERFUL GPU

- Currently only Nvidia GPUs are supported. AMD graphics cards are not supported. This is not something that we have control over. It is a requirement of the Tensorflow library.
- The GPU needs to support at least CUDA Compute Capability 3.0 or higher.

3. LITERATURE SURVEY

The automatic license plate recognition system proposed in this research has several limitations. Most major being that the state information position is assumed to be at top part of license plate. Though most of the plates consists of state information at the upper part of license plate, the proposed system will not be able to recognize the state information if the position of state information is changed. Inclusion of recognizing unwanted symbols improves the accuracy of the system. Future work includes localization and detection of state information [43]. It also might locate and detect the license number from the license plate. Future work also includes implementation of license plate segmentation using deep learning techniques, expected to have good accuracy

4. Conclusion

In this article, we have developed a deep learning project to recognize license number plate. We discussed some important features of openCV like Gaussian blur, Sobel operators, Morphological transformations. The application detects number plate text from an image. We have identified and cleaned the number plate using openCV.