Automatic Vehicle Number Plate Recognition

Project report submitted in partial fulfillment of the requirement
for the degree of Bachelor of Technology

In

Computer Science and Engineering/Information Technology

By

SAGEET YADAV (171317)

Under the supervision
of

MR. PRAVEEN MODI

Department of Computer Science & Engineering and
Information Technology
Jaypee University of Information Technology Waknaghat,
Solan-173234, Himachal Pradesh

CERTIFICATE

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this report entitled "**Automatic Vehicle Number Plate Recognition**" in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering/Information Technology submitted in the department of Computer Science &amp; Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2021 to May 2021 under the supervision of (**Parveen Modi**) (Assistant Professor and CSE&IT).

The matter embodied in the report has not been submitted for the award of

any other degree or diploma.

Sageet yadav 171317

[illegible]

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Mr. Praveen Modi

Assistant Professor (Grade-I)

Department of computer science & engineering and information technology,

Jaypee University of Information Technology Waknaghat, Solan-

173234, Himachal Pradesh. 05-12-2020

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to our teacher and mentor **Mr. Praveen Modi** who gave us the golden opportunity to do this project on the topic Automatic Vehicle Number Plate Recognition, who also helped us in doing a lot of Research and we came to know about so many new things. We are thankful to him. Secondly, we would also like to thank Lab assistants who helped us a lot in finalizing this project within the limited time frame.

I am grateful to, Head of Dept. of Computer Science & Engineering and Information Technology for his excellent support during our work. We would also like to thank all the professors & other supporting members of the department of Computer Science & Engineering and Information Technology for their generous help in various ways for the completion of this work. I am also obliged to all my faculty members for their valuable support in their respective fields which helped me in reaching at this stage of our project.

Sageet yadav(171317)

[illegible]

INDEX

LIST OF FIGURES

LIST OF ABBREVIATIONS

S.No	Name of Abbreviation	Full Form
1.	OSINT	Open source intelligence
2.	ALGOL	Algorithmic Language
3.	API	Application Program Interface
4.	ASCII	American Standard Code for Information Interchange
5.	AUI	Attachment Unit Interface
6.	BIOS	Basic Input Output System
7.	CMD	Command
8.	CPI	Clock / Cycle Per Instruction
9.	DAC	Data Acquisition and Control
10.	DBMS	Data Base Management System

AUTOMATIC LICENSE PLATE DETECTION & RECOGNITION

ABSTRACT

Traffic light and vehicle proprietor ID turned into a critical disadvantage in everywhere on the world. Regularly it gets problematic to detect a vehicle proprietor World Health Organization disregards traffic rules and drives excessively speedy. Accordingly, it's unreachable to get and punish those styles of people because of the traffic individual may not be prepared to recover the vehicle range from the moving vehicle attributable to the speed of the vehicle. Along these lines, there's a craving to build up AN Automatic reach Plate Recognition (ANPR) framework together of the answers for the current drawback. Their region unit shifted ANPR frameworks available nowadays. These frameworks zone unit upheld various procedures anyway still, it's a very troublesome undertaking as some of the elements like rapid of the vehicle, non-uniform vehicle range plate, the language of vehicle range and diverse lighting conditions will affect an incredible arrangement inside the general acknowledgment rate. The vast majority of the frameworks work underneath these constraints. During this paper, various ways to deal with ANPR is referenced by considering picture size, achievement rate and time stretch as boundaries. Towards the tip of this paper, AN expansion to ANPR is typically suggested.

KEYWORDS: Automatic Number Plate Recognition (ANPR), Number Plate, Optical Character Recognition, Artificial Neural Network (ANN), Character Segmentation Image Segmentation.

CHAPTER 1
INTRODUCTION

1.1 Introduction

The Automatic number plate acknowledgment (ANPR) is the strategy which is utilized for mass reconnaissance and fundamentally, it utilizes optical character acknowledgment on pictures to peruse the tags on vehicles. These days google tesseract is the best character recognizer on the lookout. We can likewise make our character recognizer. The least difficult character can be made by Open-Cv and py-tesseract libraries in python. Another method for creating character recognizer is the neural network or we can say that convolutional neural network will be the best. Using CNN method we will first train the model based on training dataset which consists of thousands of vehicle's images. The ANPR system is used for security purposes by various police forces and it is also used as a method of electronic toll collection on pay-per-use roads and we can also monitor traffic activities, such as red light adherence in an intersection.

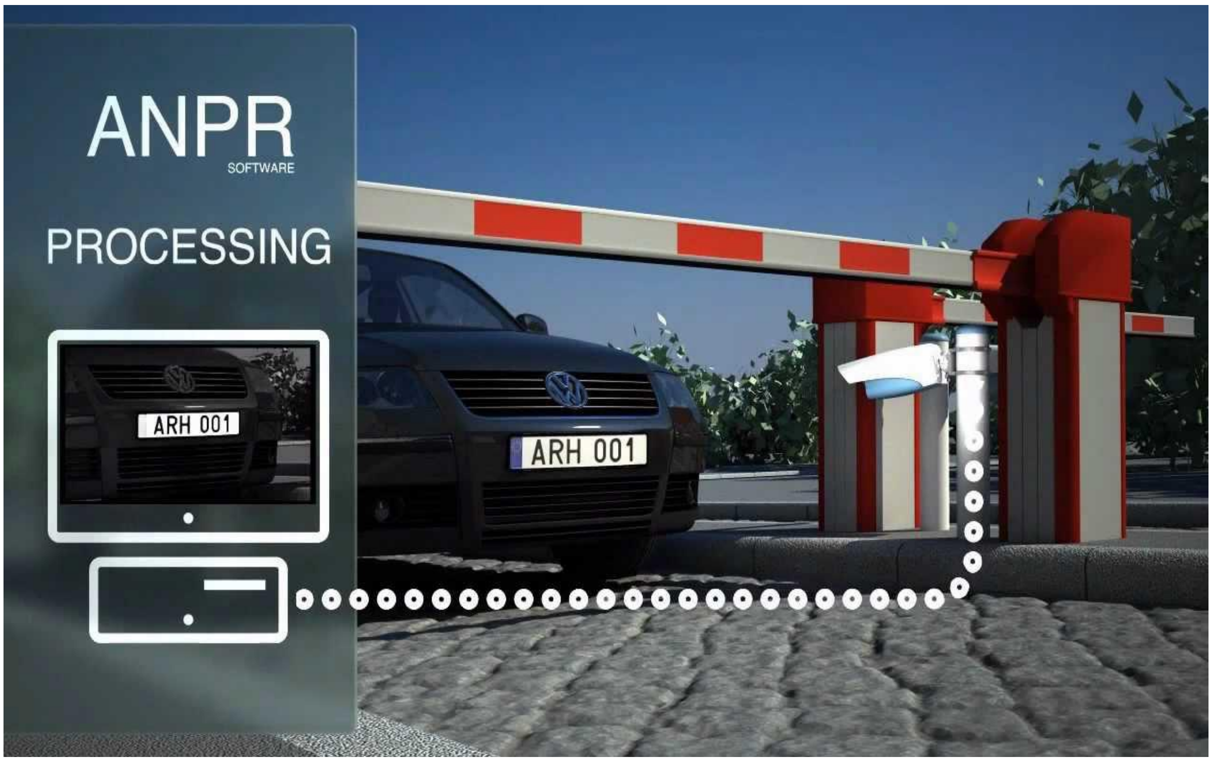


Fig 1: Simple representation of ANPR system.

Automatic Number Plate Recognition system generally divides into four major steps:

1) Vehicle Image Capture.
2) Number Plate Detection or Image Segmentation:
    a. Plate Localization
    b. Plate Orientation and Sizing.
    c. Normalization
3) Character Segmentation.
4) Character Recognition.

ANPR can be used to store the images that are captured by the cameras than detect the number plates properly as well as recognize the text from the license plate correctly and efficiently, we can modify the system to store a photograph of the driver along with the number plate and for the higher level, we can also know the details of owners of the vehicles at the same time. Systems commonly use infrared lighting which allows the camera to take the image of the vehicle at any time of the day. A powerful flash is included in at least one version of the intersection monitoring cameras, serving both to illuminate the picture and to make the offender aware of his or her mistake. ANPR technology tends to be region-specific, owing to plate variation from place to place.

1.2 Objective:

The objective of this project and this research is to successfully detect standard number plate, segment characters and then recognize the characters. The system must deal with different angles, Light conditions, Weather conditions, distances, scales, resolutions, day, night and illumination conditions.

The Important objective of my final year project is to implement an approach to be able to recognize Number Plates of different vehicles automatically in an efficient way and creating a Graphic User Interface system which is capable of detecting vehicle number plates automatically and efficiently. So we can use this system for car parking management, traffic management, parking security, automatic tolling and Intelligent transport systems.

Throughout the years, researchers working on image processing are attempting to figure out an efficient method which will work efficiently to recognize and detect number plates of different shape, size and colours. There are lots of algorithms to recognize the numbers of plates. So, our goal is to implement our project from two-three algorithms and investigate the efficiency of the different algorithms in different conditions.

1.3 Motivation:

In the present time, inner security is a major concern for all the countries in the world. Automatic number plate recognition system helps them to deal with the problems related to vehicles, the security of vehicles, traffic management and at tolling. So we want to develop our system which will work on an efficient approach of Automatic detection and recognition system. Because it is such a problem that does not have an ideal solution. Recognition of number plates needs proper detection of number plates. So we have to focus on both the detection part and recognition part so we get an efficient approach. In ANPR, we generally focus on plate detection and character recognition so we can maintain a database. We generally use ANPR in traffic management, car parking management, police security checkpoints, tolls and other public places for security purposes.

Hence, we have to create a system which can be used in every public place so we can efficiently detect and recognize the vehicle number plates in an efficient manner.

1.4 Language Used

1.4.1 Python

- Python is a famous programming language.
- It was made by Guido van Rossum and It was discharged in 1991.
- We can use python for:
    - Improving server-side.
    - Web development.
    - Framework scripting, science
    - Artificial intelligence.
    - Machine learning
    - Framework scripting.

1.4.2 What would python be able to do?

- We can use python on a server to make web applications.
- Python can be associate with database frameworks.
- Python can be used to deal with large information like Big Data, Data Science.
- Python can also be used to perform complex arithmetic.
- It can be used for fast prototyping.

1.4.3 Why Python?

We are utilizing python language since Python can be utilized on the vast majority of the stages like Windows, Mac, Linux, Raspberry Pi, etc. Python is a stage free language. Python is comparative and as straightforward as the English language. Python bolsters bunches of libraries and it has a straightforward linguistic structure like English while java and C++ have complex codes. Python programs have less number of lines than some other programming language. That is the reason we use Python language for man-made consciousness, AI, Dealing with large information. Python is an article arranged language. Uh oh, the idea in python are classes, objects, polymorphism, exemplification, legacy and reflection.

1.5 Technical Requirements:

1.5.1 Hardware Requirements

- Core i3 or higher, (Cache – 3MB or 4 MB recommended)
- 256 GB RAM
- GB hard free drive space

1.5.2 Software Requirements

- Python 3

  Python Libraries used are:

  1. Tkinter
  2. Keras
  3. Pytesseract
  4. Tensorflow
  5. Imutils
- Pycharm
- Google collab
- Web Browser: Google Chrome (recommended)
- Operating System: Window 10 (recommended)

1.6 Application Areas:

A portion of the utilization of ANPR are:

- Parking robotization and stopping security:
  - a. Ticketless stopping expense the executives.
  - b. Parking access robotization.
  - c. Vehicle area direction

d. Car robbery counteraction.

e. Partially or completely robotized instalment measure.

- We can utilize ANPR in Intelligent Transportation System prepared which can give:

a. A framework which can consequently gather expressway cost.

b. Analysis of hefty traffic in urban communities.

c. Keep a record of substantial vehicles like Trucks.

d. For the counteraction of vehicles from criminals.

e. To watch that Public is keeping the laws or not.

f. To discover who is disrupting norms.

g. To control monitor vehicles entering from outskirts, and so forth.

- Other potential applications include:

a. We can likewise make an information base of vehicle developments.

b. We can utilize ANPR at checkpoints for observing the security of streets.

c. Vehicle reconnaissance at service stations, inns, and eateries and so forth

d. Keeping the record of date and season of vehicles.

e. State fringe control is one of the main utilizations of ANPR.

CHAPTER-2

LITERATURE SURVEY

2.1 “AI-powered Indian License Plate Detector” By Sarthak Vajpayee in Sep 7,2019

For my project, I have studied this research paper “AI- powered Indian License Plate Detector” by Sarthak Vajpayee on Sep 7, 2019. One day a guy hit author’s car and got away so auther get inspired and decided to implement an AI-powered Indian License Plate detector.

Approach:

He wants to build a system which is capable of:

- First, we will take image or video from source camera. In hardware, we need a camera and a computer. For software, Sarthak Vajpayee has used python(3.6.7) language and python libraries such as OpenCV(4.1.0) for his project..
- He has used Haar cascade and Pre trained model for the detecting number plate from the image. So after that we can recognize an Indian license plate.
- He has used OpenCV, OpenCV's grayscale, erode, dilate, threshold, contour detection etc so he can generate information from the number plate. This information can be used in further stages. (We will get 8 out of 10 characters if an image has a lot of noise and distortion. Then to enter the next stage we have to remove noise from the image.
- In the next stage, we will do character segmentation which means segmenting and separating the alpha-numeric characters from the number plate. We can noise-free the image by thresholding, eroding, dilating and blurring that image. After that’s to extract the characters we will use contour detection.
- In the final stage, he will try to recognize the characters one by one. For recognizing he has created a CNN model and pass those characters one by one from the trained model. He has used the karas python library for his convolutional neural network CNN model. At last, he gets the combined result of those characters as a string.

Prerequisites:

- Python: Programing Language. He has used python version 3.6.7 here.
- OpenCv: Open Source Python Library for Real time Computer Vision. He has used Open-Cv version 4.1.0 for his project.
- IDE: He has used Jupyter Notebook.
- Keras: Open Source Python library for making deep learning simple. It is used to create neural networks.

- Haar Cascade = ML Algorithm for Object detection and Identification.

2.2 "Automatic Number Plate Recognition System" by Amr Badr, Mohamed M. Abdelwaheb, Ahmed M. Thabet and Ahmed M. Abdelsadek in 2011.

For my project, I have also studied this research paper 1.1 "Automatic Number Plate Recognition System" by Amr Badr, Mohamed M. Abdelwaheb, Ahmed M. Thabet and Ahmed M. Abdelsadek in 2011. They are creating a system which can be efficiently used in tracking stolen cars, traffic management, parking toll etc.

Approach:

He wants to build a system which is capable of:

- **Pre-Processing:** In this, he has enhanced the input number plates. First, they read the image, and resize the image then they have applied minimum filter to remove the noise from the input image. Then they have used segmentation and morphological so differentiate between foreground and background.
- **License Plate Localization:** In this step, they have found the location of the number plate. First, they set a rectangle containing the number plate but with extra areas. Then they used Sobel edge detection filter for the proper detection of image and removing extra areas.
- **Character Segmentation:** In this stage, they do segmentation of the characters from the number of plates. In this, they have separate all the characters from the background of the image.
- **Character Recognition:** The main goal this stage is recognizing and classify all the characters which they have separated in the previous stage. In the last of this stage, every character has a label and a factor which show error and a value will be predefined and if the error factor will be greater than this pre-defined value than that character will be rejected and marked as false character accidentally came or passed from the previous steps. They used a convolutional neural network for the recognition of these characters.

Prerequisites:

- Python: Programing Language.
- OpenCv: Open Source Python Library for Real time Computer Vision. He has used Open-Cv version 4.1.0 for his project.
- IDE: He has used Jupyter Notebook.
- Keras: Open Source Python library for making deep learning simple. It is used to create neural networks.
- Haar Cascade = ML Algorithm for Object detection and Identification.

CHAPTER-3

SYSTEM DEVELOPMENT

3.1 ANPR

In this section, we will discuss briefly about the different sections of Automatic number plate recognition. There are four main sections in the process of automatic number plate recognition and they are:

1.) Pre-Processing.
2.) License Plate Localization.
3.) Character Segmentation.
4.) Character Recognition.

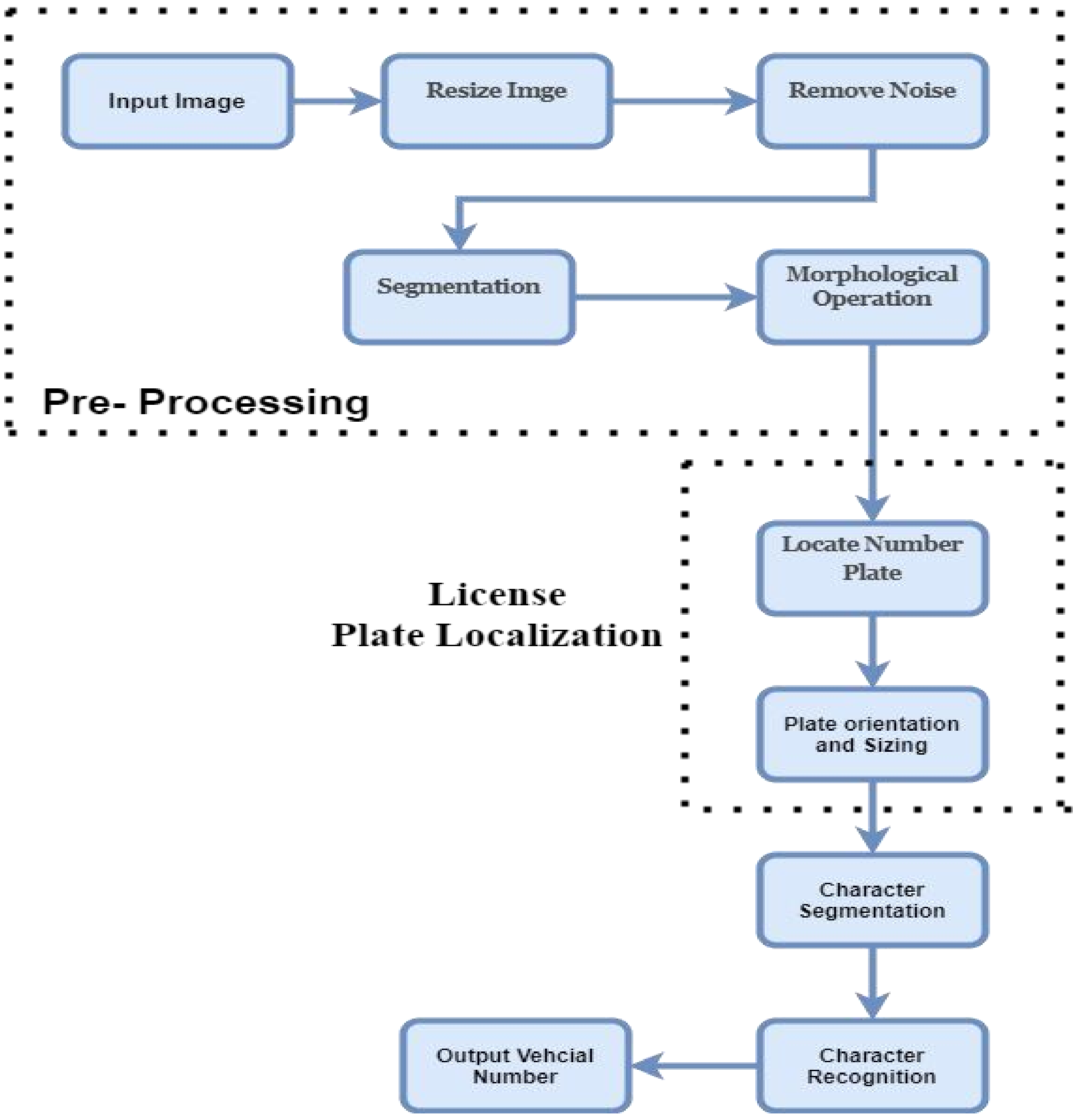


Fig 2: Steps in ANPR system.

3.1.1 Pre-Processing

Data preprocessing is a technique in which we use different algorithms to transform the raw data into useful and efficient information. The Pre-Processing involves handling of missing data, noisy data, dealing with outliers and smoothing of images etc.

Steps in pre-processing are:

3.1.1.1 Read image:

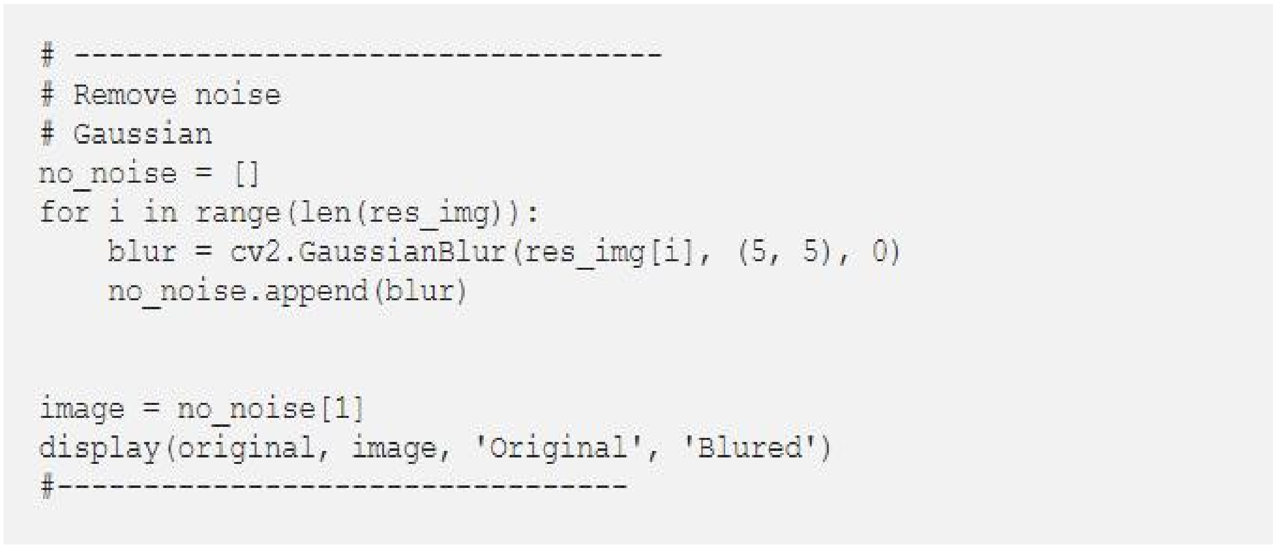
The first step of Pre-processing is to take an image from the Source. We have to store the path of the directory or the path to the image dataset and pass that path into a variable then we can create a function which can load those images in the dataset or in that folder into arrays. We have to import the libraries so that we can use and store those images. We use “os” and “NumPy” libraries to do this work.

3.1.1.2 Resize image:

All the images captured by the cameras have different size. So, We have to use a suitable size for all the images in the dataset. That’s Why we resize all the images in the dataset so we get images of same base size. As we can see in the image shown below that we have resized the image 220 x 220.

3.1.1.3 Remove noise (Denoise):

In this step, We remove the noise from the dataset Images. In this step, we smooth our image to remove unwanted noise. We mainly do this step using gaussian blur. This effect is widely used in graphic software, mainly to reduce noise from images. Gaussian smoothing is used as a pre-processing step in digital image processing algorithm for enhancing image structures at different scales.

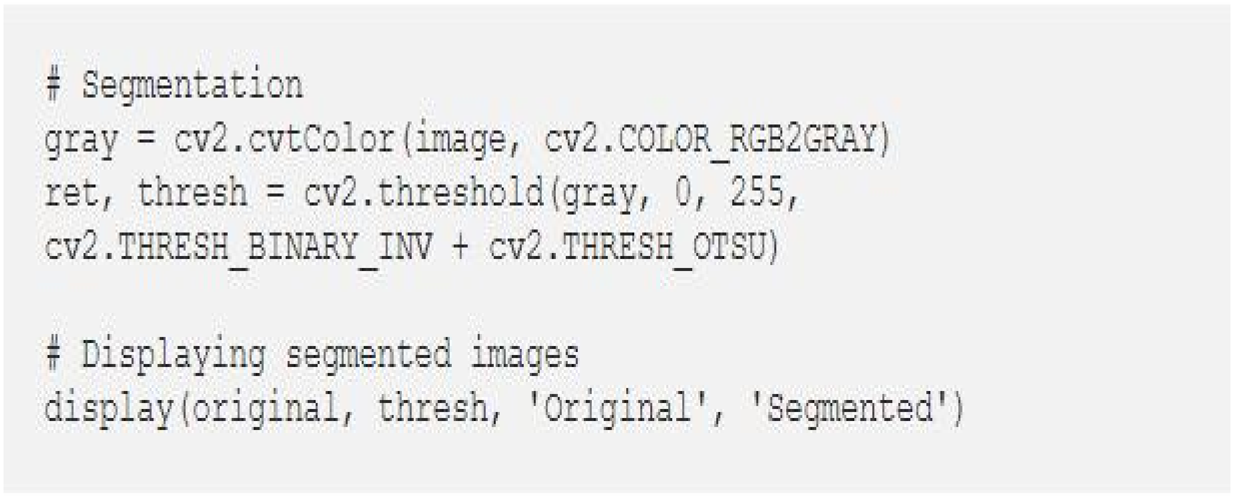


```
# ----------------------------------
# Remove noise
# Gaussian
no_noise = []
for i in range(len(res_img)):
    blur = cv2.GaussianBlur(res_img[i], (5, 5), 0)
    no_noise.append(blur)


image = no_noise[1]
display(original, image, 'Original', 'Blured')
#----------------------------------
```

Fig 3:code for noise removal

3.1.1.4 Segmentation:

In this step, we will segment the image which means we are separating the background from foreground objects. And for further improvement in segmentation, we will do more noise removal in the image.



```
# Segmentation
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
ret, thresh = cv2.threshold(gray, 0, 255,
cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)

# Displaying segmented images
display(original, thresh, 'Original', 'Segmented')
```

Fig 4: Code for segmentation of an image.

3.1.1.5 Morphology:

In this step, we use morphologic operations on the image. It is used to extract the meaningful objects from the image or we can say that using morphological operations we extract meaningful objects from the foreground.

Now, we can separate all the different objects in the image with markers. As shown in the image below:

3.1.2. License Plate Localization

In this stage, we will locate the license plate from the given image. The Final Output of this stage will be an image that contains the image of the number plate only. This stage mainly has two steps:

3.1.2.1. ***Locating a bounding rectangle over the vehicle number plate***.

In this step, we first focus on the location where the number plate is located. In this, we will represent the location of number plate with a rectangle and this rectangle may contain some extra areas or portions from the four sides of number plate.

This area of an image under this rectangle will be used as the input for the next step for further processing such as removing the extra parts, then character segmentation and at last recognition.

First, We will use Sobel vertical edge detector on the input image. Then we will set a threshold of 36 in such a manner that every edge whose magnitude is less than 36 is not considered to be an edge and its value will be set to 0.

Then we will use a vertical projection which lies on the y-axis on the edge detected image and we will use average filter to smooth this image with width equals to 9. We know that the number plate along with characters of number plate has strong vertical edges. These steps help to locate only the number plate in the next step.

3.1.2.2. ***Determining the exact location of the number plate.***

In this step, we will rectangle image from the previous step in which the yellow rectangle contains the number plate. But in this image, we have some extra areas along with will the number plate. So, now we focus only on the number plate and remove the extra areas from sub-image we get from the last step. Most of the time we have seen that the number plate may be skewed. This is because of the angle of the source camera. That's why this step of de-skewing the number plate to its original orientation become an important step. So in this stay, we will first focus on de-skew the original orientation of number plate and then resizing of the number plate and removing the extra parts of the sub-image from the last stage. For this, we will cut the parts having false edges and only focusing on the number plate. First, we will get rid of extra parts from top and bottom and then we will cut the extra parts from the left and right and at last, we will get our number plate and we will extract that number plate from the image. At this moment, finally, we have our number plate.

3.1.3. Character Segmentation

In this stage, We do segmentation of the characters from the number of plates. In this, we will separate all the characters from the background of the image. The output of this stage will be a set of monochromic images for each and every character present in the number plate. The first step in this stage is to use an adaptive threshold with a window of size 11 to convert the number plate to a binary image. After this step, we will remove noise and for this, we will use flood fill to get the connected components based on the 8-neighbourhood from the binary image. So, Firstly we do binarization of the plate image and then we will remove the noise from it. Each character in the number plate has an aspect ratio between some range and a certain range of several pixels. The maximum filter is applied after removing noise. This is applied to make the effect of thinning the characters in the number plate to make sure that these components will not merge. We will cut those components individually by detecting the boundaries between the characters. The peaks correspond to the gaps b/w the characters. Since a true number plate has a fixed range of gaps b/w characters. Then according to the peaks, the characters will be cut. So finally we got the binary image in which all the characters are cut individually. These sub-images will be used as an input image in character recognition.

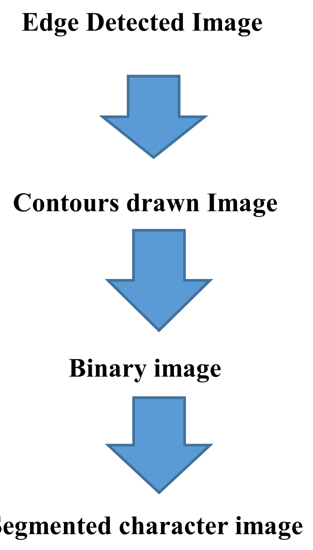


Fig 5:segmentation image flow

3.1.4. Character Recognition

The main goal of this stage is to recognize and classify all the characters which we have separated in the previous stage. In the last of this stage, every character has a label and a factor which show error and a value will be predefined and if the error factor will be greater than this pre-defined value than that character will be rejected and marked as false character accidentally came or passed from the previous steps. For the classification of the characters, some features of those characters must be collected.

We are using **a convolutional neural network** for the recognition of these characters. In this, we are creating a machine learning model and training that model for these characters.

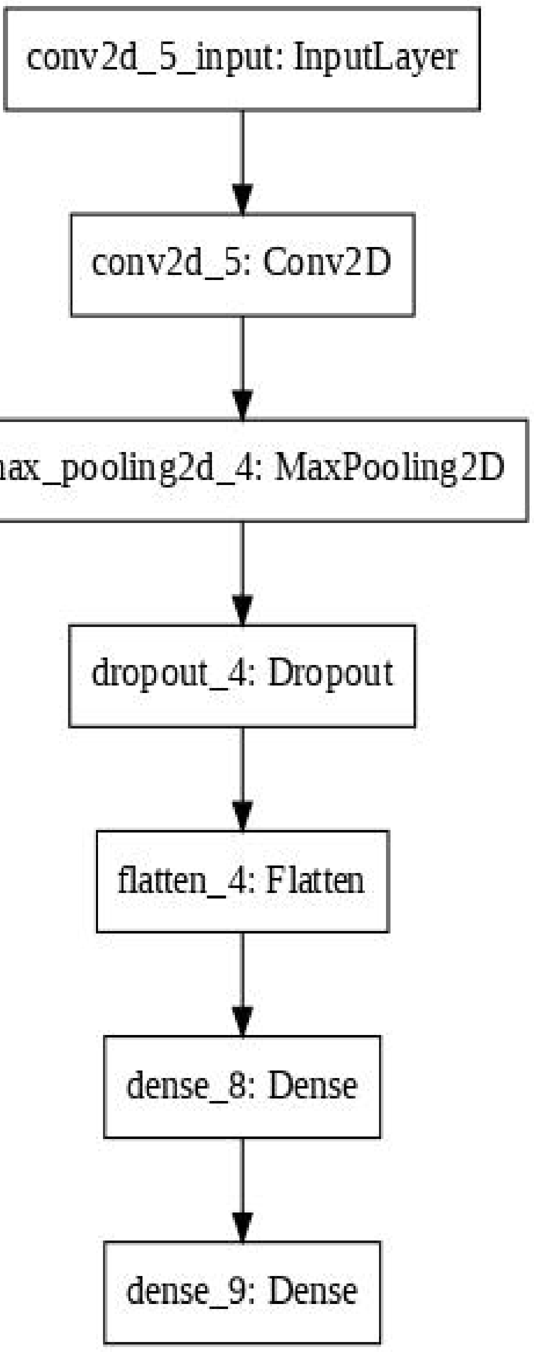


Fig 6: Steps in CNN

The first step will be the cleaning of the data. Then data is ready to use. Now we will create a convolution neural network. After training, this CNN model will be so efficient that it can recognize the characters.

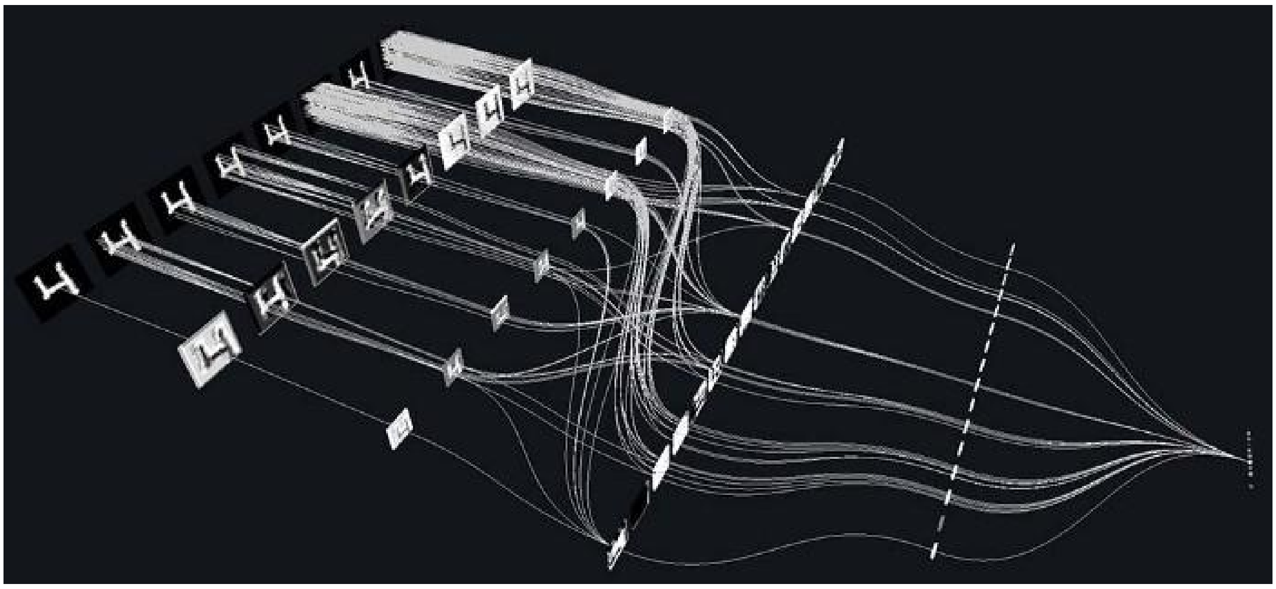
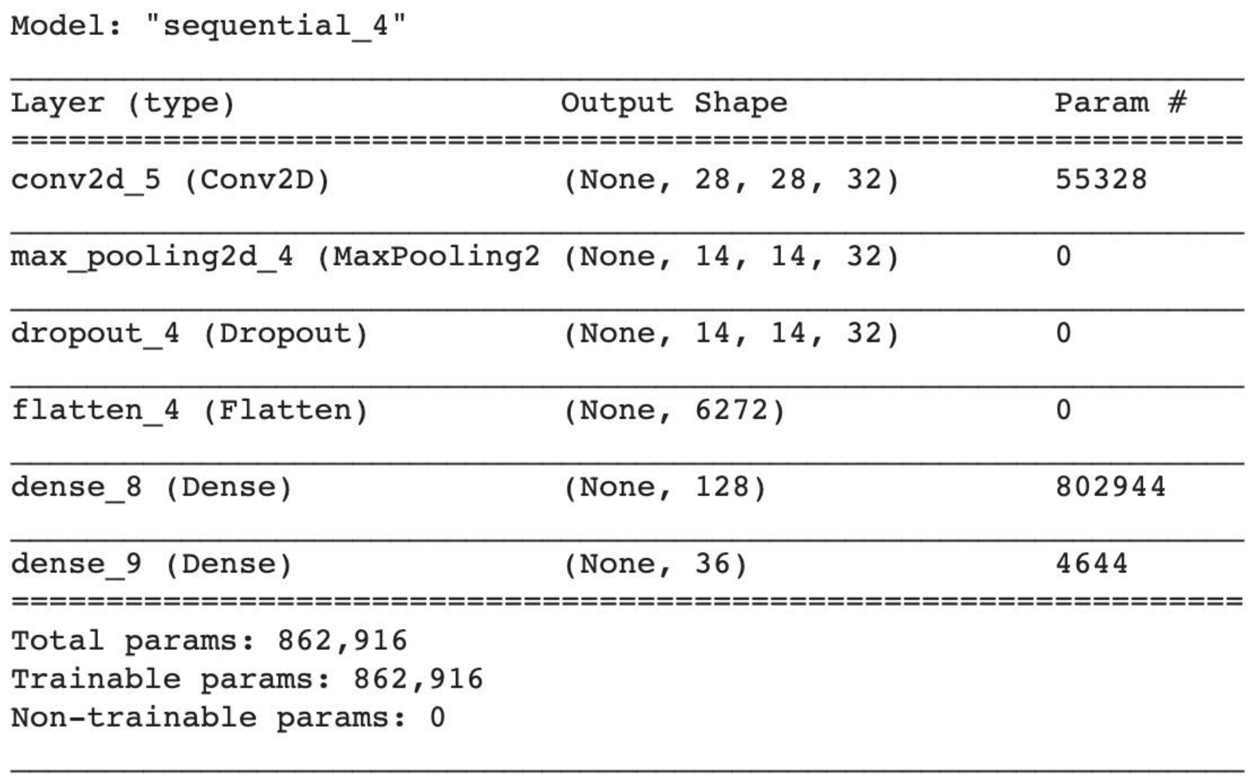


Fig 7: Simple model of neural network.

For the modeling of this system, we are using Convolutional Neural Network with 3 layers as shown in figure bellow:



```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 28, 28, 32)        55328
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
dropout_4 (Dropout)          (None, 14, 14, 32)        0
_________________________________________________________________
flatten_4 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_8 (Dense)              (None, 128)               802944
_________________________________________________________________
dense_9 (Dense)              (None, 36)                4644
=================================================================
Total params: 862,916
Trainable params: 862,916
Non-trainable params: 0
_________________________________________________________________
```

Fig 8: Sequential model

We will start and use the sequential object to keep this model simple. The first and the main layer of this CNN (Convolutional neural network) will be the convolution layer having 32 output filters and also having a convolution window of size (5,5). This model has ‘ Relu ‘ as activation function as shown below:

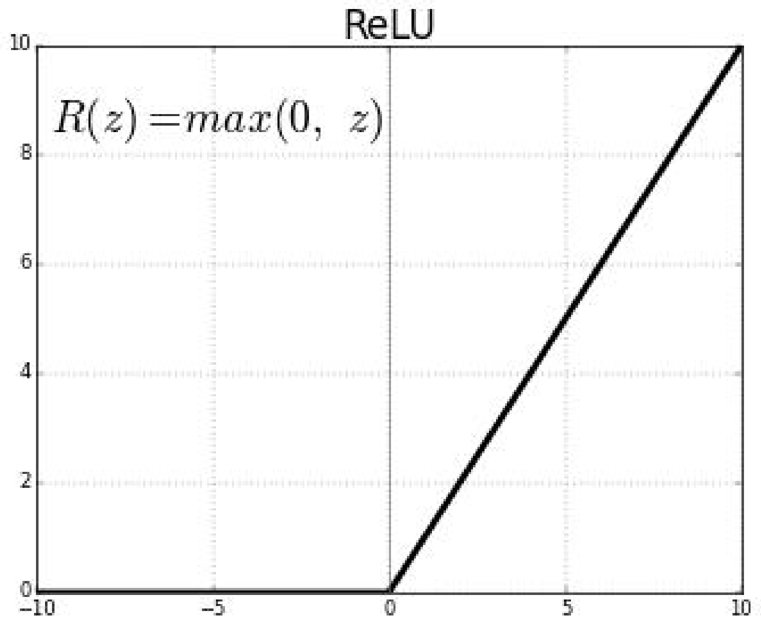


Fig 9: Relu Activation function

If we are using ANN Artificial neural network instead of CNN than we will use ‘sigmoid’ as the activation function in place of ‘relu’. One of the advantages of Sigmoid is that it will not be blowing up activation. Where on the other hand relu did not vanish gradient. Relu is also more computationally efficient in comparison with the sigmoid. Relu does not perform expensive operations because it just needs to pic max (0,x).

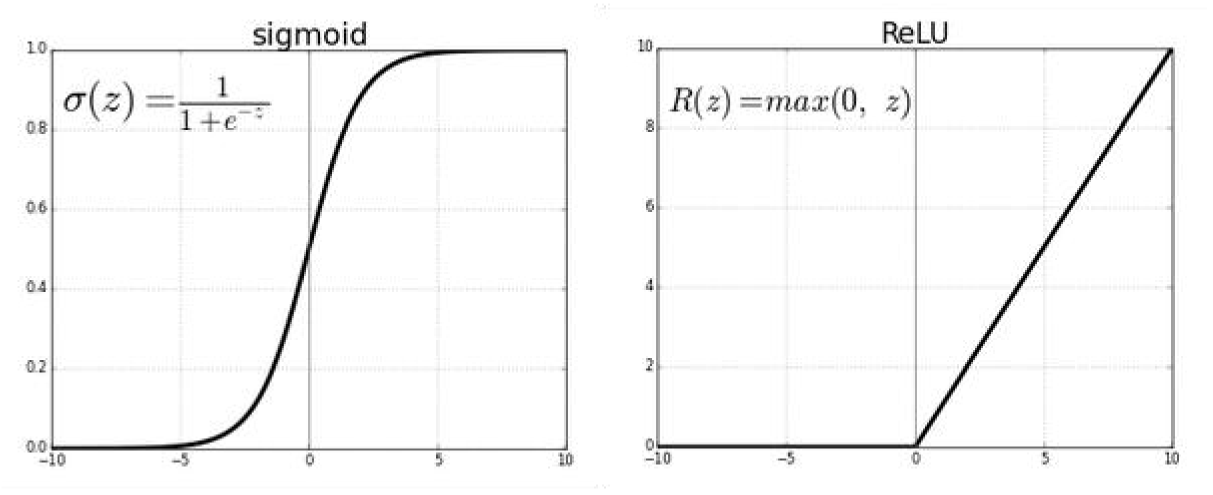


Fig 10: Sigmoid vs Relu Activation Function

Where

$$RELU(x) = \begin{cases} 0 \ if \ x < 0 \\ x \ if \ x >= 0 \end{cases}$$

There is one more activation function named as Tanh. Tanh is also more computationally expensive than Relu like sigmoid. It only involves simpler mathematical operations during computation. Tanh is best for negative images or negative inputs. For classification between 2 classes we normally use Tanh. Whereas we use Relu function which is more suitable for CNN neural network. We apply a relu activation function on our images because it can increase the non-linearity in those images. As, when we normally look on images we can find that image has contained a lot of non-linear features.

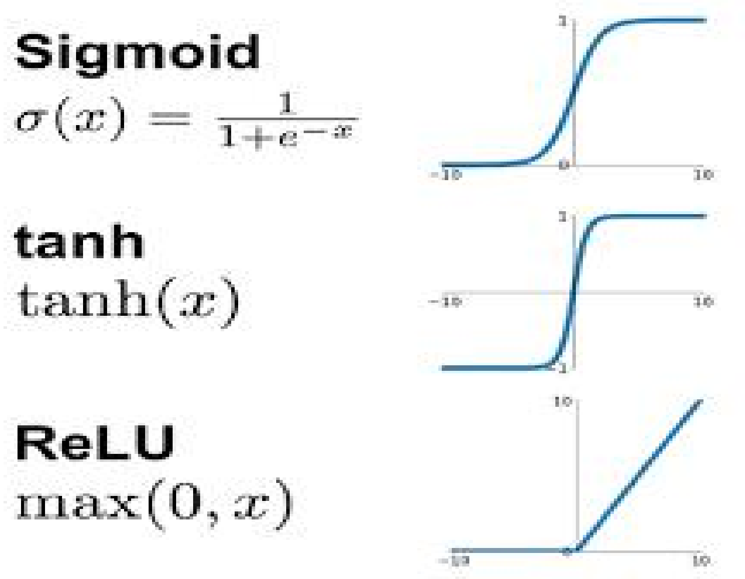


Fig 11: Sigmoid vs Relu vs Tanh Activation function.

In the next step, we will use a new layer name as a max-pooling layer having a window size of (2,2) as shown in the figure. Basically, Max-pooling is a process which is a sample based discretization process. In this max-pooling, we will do down sample an input image, hidden layer out the matrix and try to reduce its dimensionality. This will allow the assumptions about features to be made which contained in the sub-regions binned.

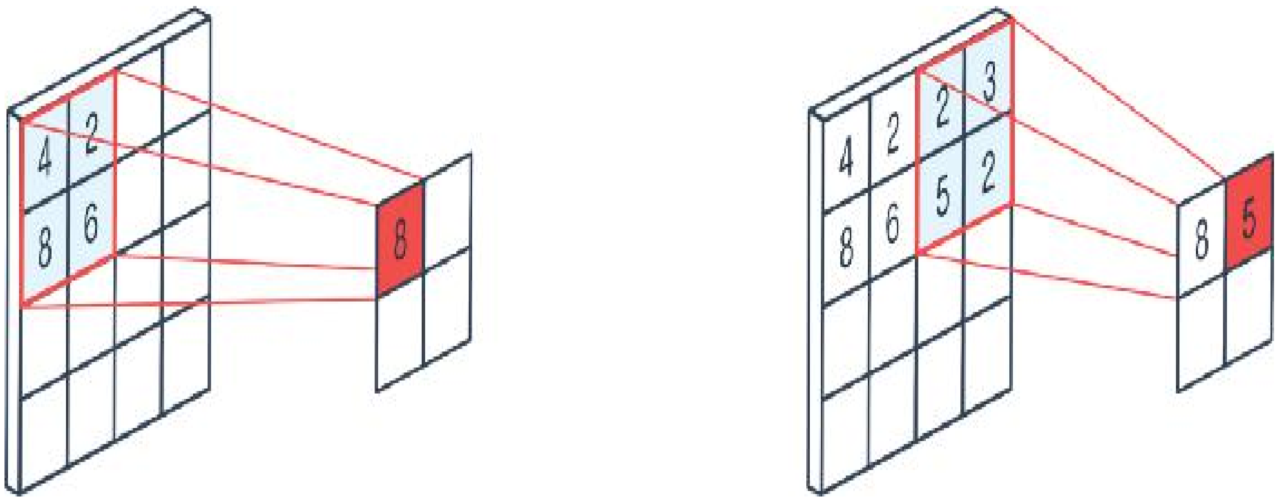


Fig 12: Max- Pooling Layer

To take care of overfitting, in this process we will add some droupout rate. Droupout is initialized as regularization hyperparameter which will prevent our neural network from overfitting. Some of the neurons are ignored. So dropout will select the ignored neurons randomly. These neurons are ignored during training randomly. For this model we are using a dropout rate of 0.4 which means 60 percent of the node which are retained.
Now we will use the flatten layer because this is the time when we need to flatten the node data. Data from the previous layer will be taken by this flatten layer and this represents it in one dimension.

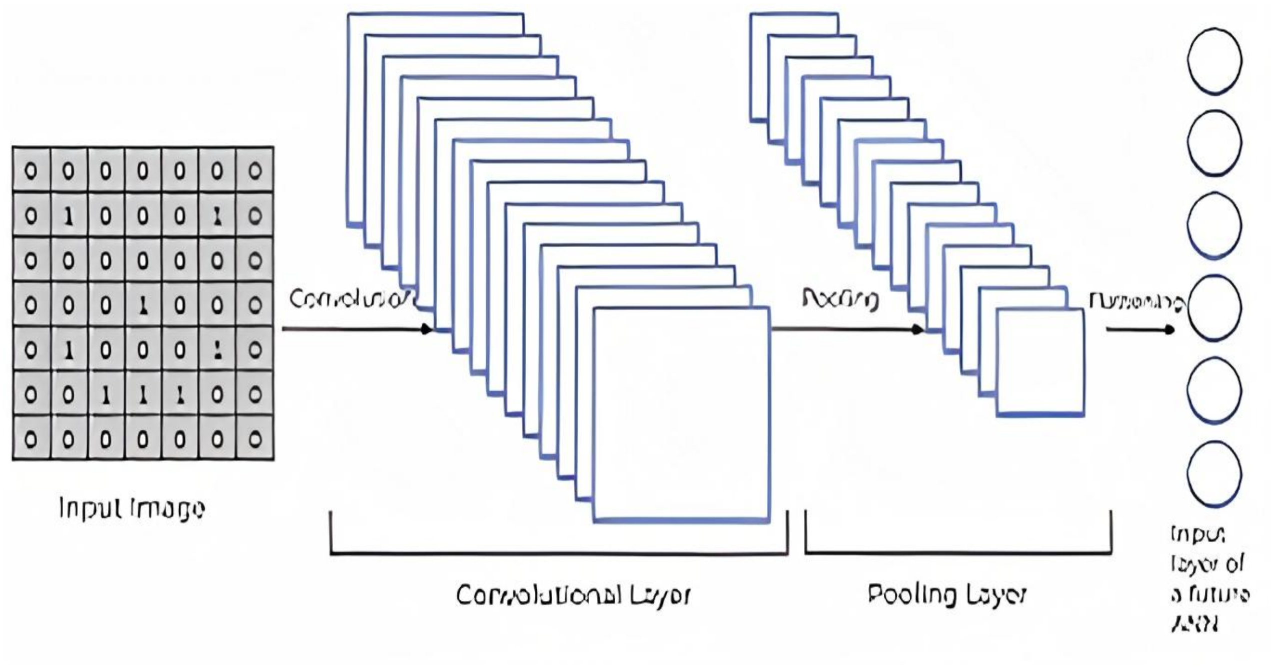


Fig 13: CNN Layers

In the final step of creating this model, we will add 2 dense layers. The first dense layer has the dimensionality of output space equal to 128 with ‘ relu ‘ as the activation function. In the second layer which is our final layer with 36 outputs for categorizing the 26 alphabets from A to Z and 10 digits from 0 to 9 and having ‘ softmax ‘ as the activation function.

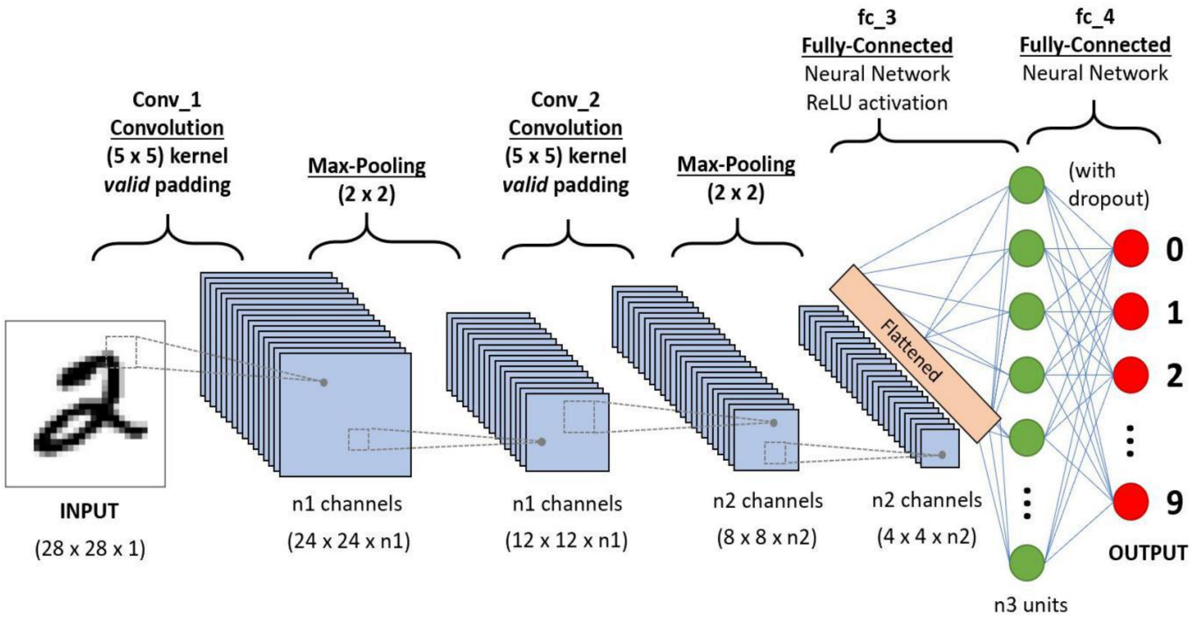


Fig 14: ANPR CNN model

We have created the model. Now we have to train the model. For training the model we need a dataset. To recognize the alphabets and digits present in the number of plates, we will use the images of all the Alphabets from A to Z and digits from 0 to 9 to train this model.
So our dataset has contained the images of all the Alphabets from A to Z and digits from 0 to 9. All the images in the dataset are of size 28 x 28. We have also balanced the data so we don’t have to do any type of data turning. We have split our dataset into train and test images with the ratio of 80:20.
We are using ImageDataGenerator class. This class is available in Keras. We are using image augmentation techn iques like height sift and width shift to generate some more data.
Now we will train our model. We will use loss function = ‘categorical_crossentropy’, optimization function = ‘Adam’ and error matrix = ‘Accuracy’.

CHAPTER-4
PERFORMANCE ANALYIS

4.1 **Performance Evaluation**

Our dataset contains images of digits from 0 to 9 and images of alphabets from A to Z.in both upper and lower case. The size of all the images is 28 X 28. For training the model, we have created a zip file and we have split our dataset into train and test images with the ratio of 80:20 as shown in figure below:

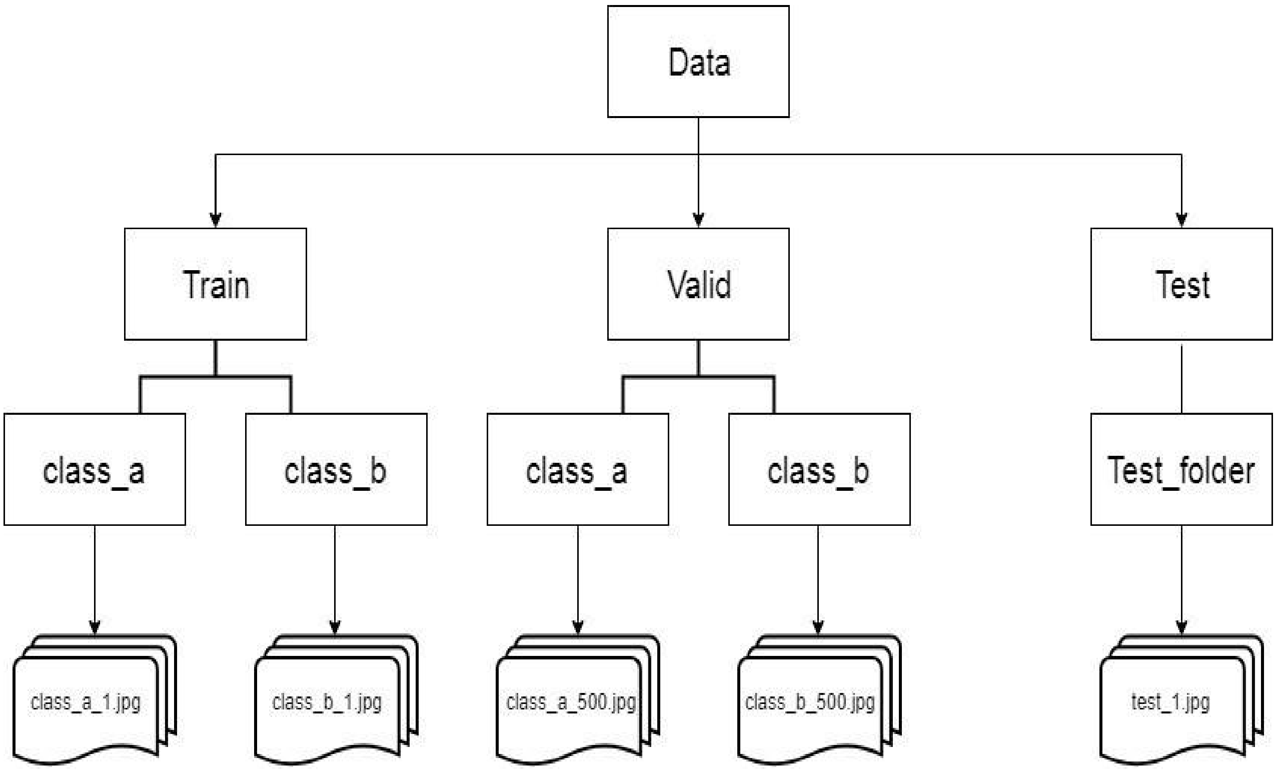
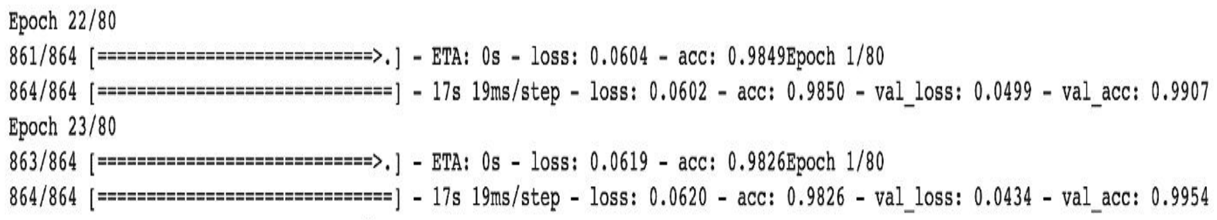


Fig 15: Flow diagram of Dataset

Our Model has achieved an accuracy of 99.54% after training for 23 epochs. As shown in figures below:



```
Epoch 22/80
861/864 [============================>.] - ETA: 0s - loss: 0.0604 - acc: 0.9849Epoch 1/80
864/864 [==============================] - 17s 19ms/step - loss: 0.0602 - acc: 0.9850 - val_loss: 0.0499 - val_acc: 0.9907
Epoch 23/80
863/864 [============================>.] - ETA: 0s - loss: 0.0619 - acc: 0.9826Epoch 1/80
864/864 [==============================] - 17s 19ms/step - loss: 0.0620 - acc: 0.9826 - val_loss: 0.0434 - val_acc: 0.9954
```

Fig 16: Screenshot of training.

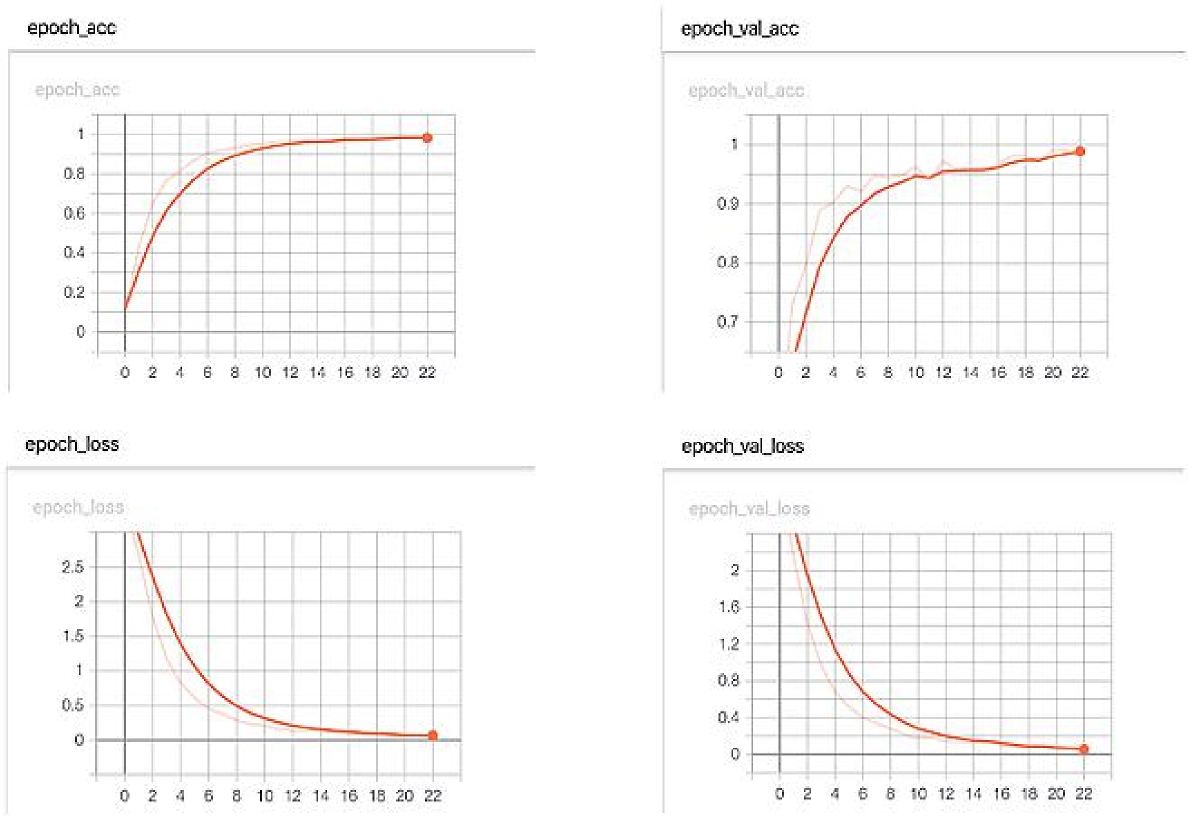
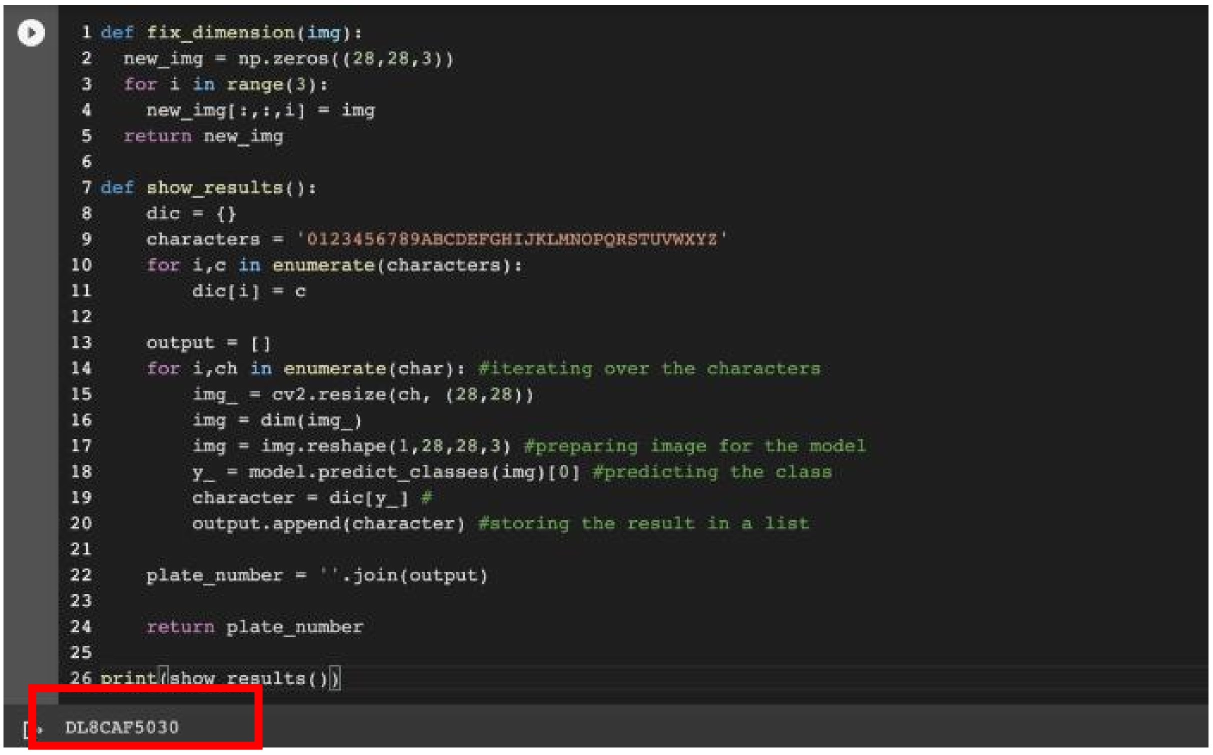


Fig 17: Graphical representation of loss and accuracy.

Now our system is created and trained properly with efficiency more the 99%. So we test this model on the input image.



```
def fix_dimension(img):
  new_img = np.zeros((28,28,3))
  for i in range(3):
    new_img[:,:,i] = img
  return new_img

def show_results():
    dic = {}
    characters = '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ'
    for i,c in enumerate(characters):
        dic[i] = c

    output = []
    for i,ch in enumerate(char): #iterating over the characters
        img_ = cv2.resize(ch, (28,28))
        img = dim(img_)
        img = img.reshape(1,28,28,3) #preparing image for the model
        y_ = model.predict_classes(img)[0] #predicting the class
        character = dic[y_] #
        output.append(character) #storing the result in a list

    plate_number = ''.join(output)

    return plate_number

print(show_results())
```

```
DL8CAF5030
```

Fig 18: Screenshot of text recognit

4.2 **Discussion**

We can see from the from the fig 3 to the fig 11. We have done the pre processing of the images. In which figure 3 is of resizing, fig 5 is of noise removal, fig 6 is the image of blured image, fig 9 is of segmented image and fig 11 is the marked image.

From fig 12 to fig 14, we have done plate localization. In fig 13 we have cut the image and focus on the number plate. We have plotted a rectangle around number plate. But this rectangle has some extra areas rather than number plate. Then we have extracted the number plate and removed the unwanted area.

From the fig 15 to fig 18, we have done the character segmentation. On fig 14 we have applied sobel edge dectector. After apply edge detector we get fig 15. Then we have drawn contours and we get fig 16. In fig 17 we got binary image of fig fig 16. At last of this we get fig 18.

Next stage is character recognition. In this stage we have from fig 19 to fig 25. Fig 19 shows the steps of the CNN neural network. Fig 22 to fig 24 shows the different activation functions. fig 25 shows the max-pooling layer of CNN neural network.

Fig 26 to fig 32 shows the flow diagram of dataset and performance of the CNN model after training. Fig 30 shows the graphical representation of loss and accuracy during testing. Fig 31 shows final output in form of strings. It shows the text that we have recognized from the number plate. So our system is efficiently recognizing the number plates.

CHAPTER - 5

CONCLUSIONS

5.1. Conclusions

In this project, we have created two models to recognize the Characters from the number plate. First one is by using ‘Tesseract’ which is the OCR engine. We use tesseract in python by importing a py-tesseract library. The second one is ‘CNN’ character recognition model whose training accuracy is more than 99%.

During the test the average accuracy was 99.54% and average loss during training the model was 4.34%.

We have also compared both of the models during testing and we have found that the tesseract OCR engine is not efficient where on other hand the CNN model is predicting the alphabets and numerical numbers perfectly and efficiently.

In future we are ready to modify and bring changes in our approach and this project so we can increase the efficiency of our project.

5.2. Future Scope

In future we can create more efficient model the character recognition of number plates. We can use this system in the following areas in future:

- Parking robotization and stopping security:
  a. Ticketless stopping expense the executives.
  b. Parking access robotization.
  c. Vehicle area direction
  d. Car robbery counteraction.
  e. Partially or completely robotized instalment measure.

- We can utilize ANPR in Intelligent Transportation System prepared which can give:
  a. A framework which can consequently gather expressway cost.
  b. Analysis of hefty traffic in urban communities.
  c. Keep a record of substantial vehicles like Trucks.
  d. For the counteraction of vehicles from criminals.
  e. To watch that Public is keeping the laws or not.
  f. To discover who is disrupting norms.

g. To control monitor vehicles entering from outskirts, and so forth.

- Other potential applications include:
  a. We can likewise make an information base of vehicle developments.
  b. We can utilize ANPR at checkpoints for observing the security of streets.
  c. Vehicle reconnaissance at service stations, inns, and eateries and so forth
  d. Keeping the record of date and season of vehicles.
  e. State fringe control is one of the main utilizations of ANPR.

REFFERENCE

[1] Circuit Digest: https://circuitdigest.com/tutorial/vehicle-number-plate-detection-using-matlab-and-image-processing

[2] MathWorks:https://www.mathworks.com/matlabcentral/fileexchange/40426-vehicle-number-plate-recognition

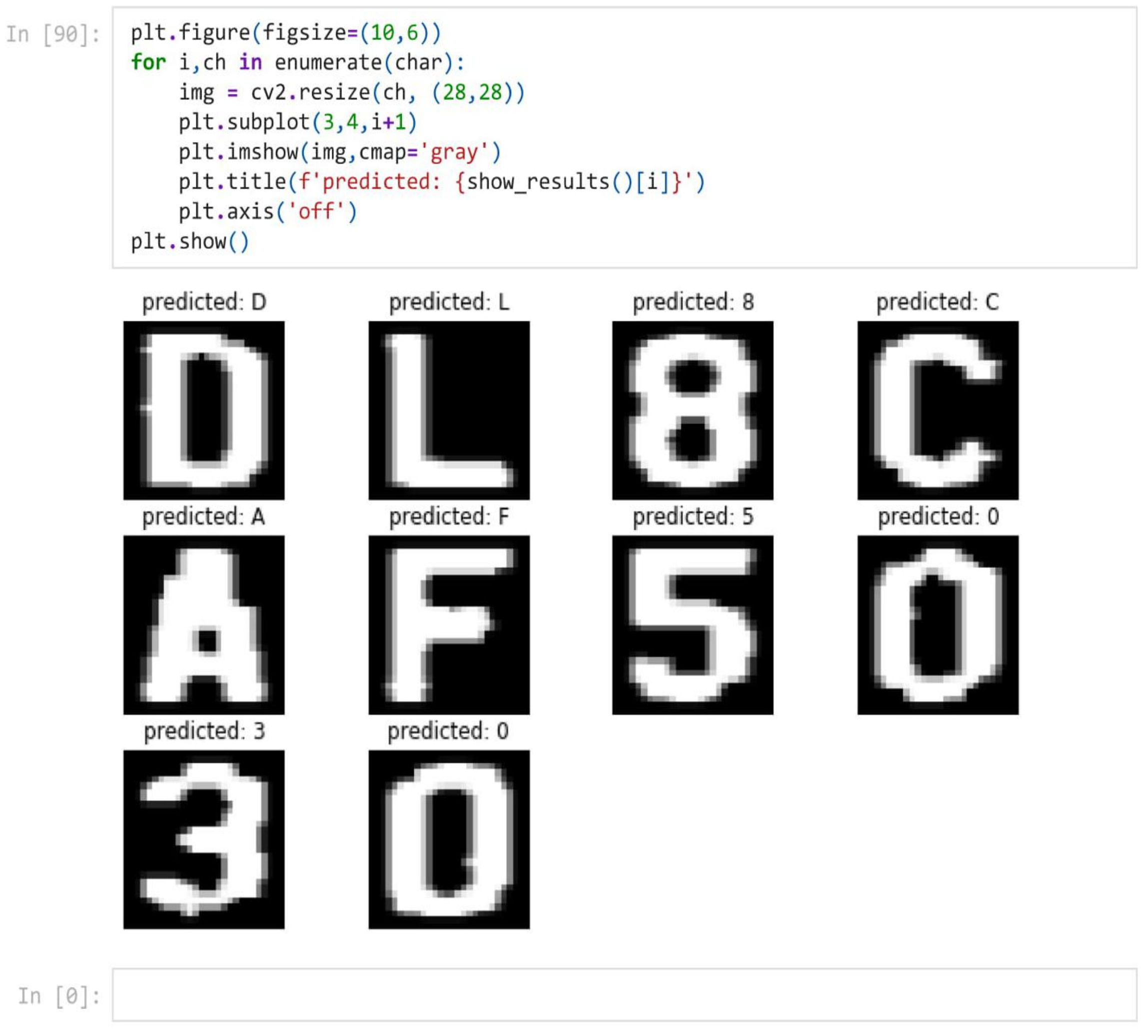
[3] Wikipedia:https://en.wikipedia.org/wiki/Automatic_number-plate_recognition, Birmohan, Manpreet Kaur, Dalwinder Singh, and Gurwinder Singh. "Automatic number plate recognition system by character position method." *Int J Comput Vision Robot 6*, no. 1/2 (2016): 94-112.

[4] Kasaei, S. Hamidreza, S. Mohammadreza Kasaei, and S. Alireza Kasaei. "New Morphology-Based Method for RobustIranian Car Plate Detection and Recognition." *International Journal of Computer Theory and Engineering 2, no. 2 (2010): 264.*

APPENDEX

```
import cv2

import numpy as np
import imutils
import easyocr

store_imag = cv2. imread('im1.jpg')
grayScale_con =   cv2. cvtColor   (img, cv2.COLOR_BGR2GRAY)

bfilter_obtained = cv2. bilateralFilter(gray, 12, 16, 16)      # Noise reduction
edged_detect = cv2. Cannys(bfilter, 40, 300)    # Edge detection

mainPoint   =   cv2.findContours(edged.copy(),   cv2.   RER_TREES,   cv2.
CHAIN_APROX_SIMPLES)
contours_obtained = imutils. grab_contours(mainpoints)
contours_modifie   =   sorted   (contour   ,   crux=cv2.   contourAreaObtained,
revrse=True)

locate = None
for contour in contours:
   aprox = cv2. aproxPoly(contour, \20,True)
    if len(approx) == 4:
        locate = aprox
        break

        maskUp = np.zeros(grayScale.shape, np.uint8)
        imageNew = cv2. drawContours(masks , [locate], 0, 256, -1)
        imageNew = cv2. bitwise_and(img,img, mask=mask)
```

```
(m, n) = np. where(maskUp == 256)
(m1, n1) = (np. min(m), np. min(n))
(m2, n2) = (np. max(m), np. max(n))
imageCropped = gray[m1:m2 + 1, n1:n2 + 1]
read = easyocr .Read(['en'])
resultExtrat  =  readr.readTextData(imageCroped)
text = resultExtract[0][-1]
font = cv2.FONT_HERSHEY_SIMPLEX
res = cv2.putText(img, text=text, org=(approx[0][0][0], approx[1][0][1] + 60), fontFace=font, fontScale=1,
                color= (0, 256, 0), thick=3, line=cv2.LINE_B)
res_new = cv2.rectangle(img, tuple(aprox [1][0]), tuple(aprox [2][0]), (0, 256, 0), 2)
plt.
```



```
In [90]: plt.figure(figsize=(10,6))
         for i,ch in enumerate(char):
             img = cv2.resize(ch, (28,28))
             plt.subplot(3,4,i+1)
             plt.imshow(img,cmap='gray')
             plt.title(f'predicted: {show_results()[i]}')
             plt.axis('off')
         plt.show()
```

```
In [0]:
```

Fig 19:Output consol image