A Hybrid Approach to Automatic Number Plate Recognition

A MINOR PROJECT REPORT

Submitted by

AISHWARYA JAYARAMAN [RA1911026010102]

ANSHUMAN PAL [RA1911026010117]

Under the guidance of

Dr. Babu R

(Assistant Professor, Department of Computational Intelligence)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M.Nagar, Kattankulathur, Chengalpattu District - 603203

NOVEMBER 2022

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section3 of UGC Act,1956)

BONAFIDE CERTIFICATE

Certified that 18CSP107L minor project report titled "**A Hybrid Approach to Automatic Number Plate Recognition**" is the bonafide work of "**AISHWARYA JAYARAMAN [RA1911026010102] and ANSHUMAN PAL[RA1911026010117]"** who carried out the minor project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE
Dr. Babu R
Assistant Professor
Dept. of Computational Intelligence

SIGNATURE
Dr. R Annie Uthra
Professor & Head
Dept. of Computational Intelligence

SIGNATURE OF PANEL HEAD
Dr. Saad Yunus Sait
Research Associate Professor
Dept. of Computational Intelligence

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support. We are indebted to **Dr. T.V. Gopal**, Dean, College of Engineering & Technology, SRM Institute of Science and Technology, for all the help provided to carry out the Project Work. We extend our sincere thanks to **Dr. Revathi Venkataraman**, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her invaluable support. We wish to thank **Dr. R. Annie Uthra**, Professor & Head, Department of Computational Intelligence, SRM Institute of Science and Technology, for her valuable suggestions and encouragement throughout the period of the project work.

We are extremely grateful to our Panel Head, **Dr. Saad Yunus Sait**, Associate Professor, Department of Computational Intelligence, SRM Institute of Science and Technology, for his inputs during the project reviews. We register our immeasurable thanks to our Faculty Advisor, **Dr. R. Athilakshmi**, Assistant Professor, Department of Computational Intelligence, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide, **Dr. R. Babu**, Assistant Professor, Department of Computational Intelligence, SRM Institute of Science and Technology, for providing us an opportunity to pursue our project under his mentorship. He provided us the freedom and support to explore the research topics of our interest.

We sincerely thank staff members and students of the Computing Technologies Department, SRM Institute of Science and Technology, for their help during our project work. Finally, we would like to thank our parents, our family members and our friends for their unconditional love, constant support and encouragement.

Aishwarya Jayaraman

Anshuman Pal

ABSTRACT

India is one of the world's most densely inhabited countries having a lot of vehicles. Therefore, a proper traffic management system to correctly detect automobiles is necessary.

We will extract the vehicle registration number from the license plate of different Indian Automobiles using Deep learning techniques. We will develop a real time model that inputs automobile images and displays the registration number of the vehicle. Initially, we start with detecting the vehicle image from which the region of interest is extracted i.e. the License plate using hybrid method i.e. combining YOLO and contour method. In order to accomplish character segmentation, a bounding box is made around each character. Next, the registration number is extracted for character recognition using CNN. The registration number is then displayed as an output.

We will be creating a custom data set consisting of images of different automobiles in different conditions and orientations to train our model.

ANPR can be used to correctly monitor vehicles at college entrances and other locations with rigorous access controls, by police forces to identify criminal activity, and on highways as a method of identifying speeding through average speed calculation.

TABLE OF CONTENTS

Chapter No.			Title	Page No.
			Acknowledgement	**iii**
			Abstract	**iv**
			Table of Contents	**v**
			List of Figures	**vii**
			List of Tables	**viii**
			Abbreviations	**ix**
1			**Introduction**	**1**
	1.1		Automatic Number Plate Detection	1
	1.2		Motivation	1
	1.3		Problem Statement	2
	1.4		Innovative Idea for our Project	2
2			**Literature Survey**	**3**
	2.1		Automatic Number Plate Recognition Literature Survey	3
	2.2		Research Gap	5
3			**Requirement Gathering**	**6**
	3.1		Number Plate Detection Data	6
	3.2		Character Recognition Data	6
4			**Module Description**	7
	4.1		Number Plate Detection	7
		4.1.1	YOLO	7
		4.1.2	Contour Method	8
	4.2		Character Segmentation	8
	4.3		Character Recognition	9
5			**Coding and Testing**	**10**
	5.1		Code Snippets for Number Plate detection using YOLO	10
	5.2		Code Snippets for Number Plate detection using Contour	11
	5.3		Code Snippets for Character Segmentation	12
	5.4		Code Snippets for Character Recognition	12

	5.5	Result	13
6		**Experimental Analysis**	**14**
7		**Conclusion and Future Work**	**15**
		References	**16**

LIST OF FIGURES

Figure No.	Figure Name	Page No.
3.1	Dataset for YOLO	6
3.2	Dataset for CNN	6
4.1	System Architecture	7
4.2	License Plate Detection using YOLO	8
4.3	License Plate Detection using Contour Method	8
4.4	Character Segmentation using Image Processing Techniques	9
4.5	Predicted Characters using CNN	9
5.1	Parameters for YOLO prediction	10
5.2	Functions used in YOLO	10
5.3	Number Plate detection using YOLO	11
5.4	Image Processing Techniques used in Contour Method	11
5.5	Character Segmentation using Image Processing Techniques	12
5.6	Character Recognition using CNN trained model	12
5.7	Prediction using YOLO + CNN	13
5.8	Prediction using Contour method + CNN	13
5.9	Prediction using Hybrid Method	13
6.1	CNN training and validation set curve	14

LIST OF TABLES

Table No.	Table Name	Page No.
6.1	Training Curve of CNN model	14

ABBREVIATIONS

ANPR	Automatic Number Plate Recognition
YOLO	You Only Look Once
CNN	Convolutional Neural Network
OCR	Optical Character Recognition

CHAPTER 1
INTRODUCTION

Automatic Number Plate Recognition system detects the Region of Interest i.e. the license number plate and then extracts the vehicle registration number from it. It is helpful for upholding the law, determining whether a vehicle is registered, identifying speeders, managing parking spaces, preventing vehicle theft, and assisting authorities in identifying and tracking down moving automobiles.

1.1 Automatic Number Plate Recognition

The highly accurate Automatic Number Plate Recognition (ANPR) technology can read vehicle license plates without the need for human involvement.

YOLO, also known as You Only Look Once, is one of the most powerful real-time object detection algorithms. It detects and classifies the whole image at once using a single convolutional network that predicts the bounding box and class probabilities. For a particular object multiple bounding boxes are created with corresponding class probabilities IOU (Intersection over Union) for all the bounding boxes generated for a particular class are compared and the one with the maximum IOU is selected.

Contour method follows a set of procedures - resizing the image to 500px width, converting to grayscale, removing noise, binarizing, finding contours and drawing contour.

Convolutional neural networks are a popular deep neural network architecture which can be used for implementing Optical Character Recognition. It learns from the important 2D features present in the image for classification

1.2 Motivation

The motivation behind the project is to:

Uphold law and order: In India, excessive speeding is the primary cause of fatalities. ANPR can be used to track the average speed of the cars and spot any that go over the speed limit.

Parking management - Vehicles with registered license plates can enter parking lots without paying, while unregistered vehicles must pay at the time of check-in and check-out. Parking tickets can be paid directly from the user's account against the produced ticket number using the number plates of the automobile that are directly linked to the owner's mobile phone.

Preventing vehicle theft - ANPR may be used to track vehicles, which can help to prevent auto theft. Law enforcement will be able to determine the exact moment, place, and path taken by a stolen car.

1.3 Problem Statement

To develop a real-time model that captures the image of a vehicle provided to detect the number plater and extract the registration number from it.

1.4 Innovative idea for our project

The project is divided into two phases - Number Plate Detection and Character Recognition. We will be using the hybrid approach i.e. YOLO algorithm and contour method for Number plate detection and training a CNN model for Character recognition.

We will be creating our own custom dataset consisting of images of different automobiles in different conditions and orientations.

The remaining sections of this paper are systematized as follows – chapter 2 provides a detailed literature survey of existing methodologies for ANPR and their findings, chapter 3 mentions the research gaps found in the papers studied and chapter 4 details the requirement gathering. The project modules are discussed in chapter 5 followed by discussion of experimental analysis in chapter 6. The result and future work is given in the final chapter 7.

.

CHAPTER 2
LITERATURE SURVEY

In this digital era, research in automatic number plate detection is very important for its uses in several areas. For example, it can be used at places with strong access restrictions to monitor vehicles properly, by police forces to track down criminal behavior and to detect speeding through average speed calculation.

2.1 Automatic Number Plate Recognition Literature Survey

[1] used Image Processing Techniques like Color Image Processing, Contrast Extension, Adaptive Thresholding, Median Filtering, Segmenting Characters for plate detection & Template Matching using OCR for text extraction. For plates at a specific angle and plates near the edge of the photograph, it did not yield the required results. Character segmentation was successful in circumstances where 82.6% of the letters could be identified.

[2] used image processing techniques for plate detection and an algorithm for template matching to segment and identify the characters. If English characters cannot be accord with characters, Hindi characters from the established template are used instead. They found that taking a photograph from the side and with low brightness and contrast has an effect on the results. There were 97% and 98% average segmentation rates (SR) and recognition rates (RR), respectively.

[3] used Image Processing Techniques for plate detection and Template matching using OCR algorithm for text extraction. Number Plate Extraction success rate was found to be around 85% and Character Segmentation success rate 80%.

[4] used Faster-RCNN with ResNet-50 model and VGG16 separately for training the network. Character Segmentation (CS) was done using ResNet-50 as Base-CNN. Standard CNN model was used for Character Recognition. They discovered that the precision of license plate detection was 94.98%. At an average mean accuracy of 99.55 %, more than 99% of characters were successfully segmented, and the pipeline correctly identified 98.6% of the segmented characters.

[5] trained a Faster-RCNN model using TF Object Detection API. Contrast was maximized using Top and Black hat morphological transformations. Character segmentation and recognition was done using Tesseract OCR. The proposed work was sensitive to changes of light intensity in the surroundings of the number plates. The Respro optimizing algorithm performed best at a learning rate of 0. 002.

[6] compared two methodologies - YOLO v3 and Traditional Image Processing. The accuracy, recall, precision for traditional image processing technique was found to be 72%, 0.92, 0.92 whereas for YOLOv3 it was 90%, 0.92 and 0.92 respectively.

[7] carried out Plate Segmentation using image processing techniques, Plate Classification using SVM and Plate Recognition using OCR Segmentation and OCR Classification i.e. MLP (Multi Layered Perceptron). The suggested method successfully identified several vehicle licenses plates in a single video frame. Even license plates with intricate backgrounds can be successfully traced.

[8] used Image Processing Techniques for Plate detection and compared classification of characters from license plate using 1NN and ANN. Plate Recognition accuracy of 1NN was found to be 87.43% and that of ANN was 86.34%. As the "k" value is raised, fewer characters are accurately identified. For wrongly identified characters, the opposite is accurate.

[9] used Image Processing Techniques for Plate Extraction, Bounding Box for Character Segmentation and template matching for Character Recognition. Character Segmentation, Plate Extraction, and Character Recognition had an accuracy of 86.67% , 93.33%, and 93.33% respectively.

[10] used custom YOLO for Plate Detection and Character Recognition. The Number Plate Detection accuracy was found to be 100% and Number Plate Recognition was 91%

[11] used Wiener filtering, YOLOv3 for License Plate detection and OCR for image enhancement and recognition. Accuracy for number plate detection using YOLO model was found to be 96% and the accuracy for character recognition using OCR model was found to be 91.5%

[12] employed YOLO v3 for Number Plate Recognition and CRNN for Number Plate

Classification. The suggested technique recognized 88% of 3-letter plates and 99 % of 4-letter plates, yielding an 86% recognition rate.

After doing a detailed literature survey on Automatic Number Plate Recognition we found a few limitations in the above-mentioned Research papers. The detection was dependent on a few factors such as weather condition, orientation of the image, distance of the number plate in the image and font.

2.2 Research Gap

In most of the literature survey done, it can be identified that the detection is dependent on a few factors such as weather condition, orientation of the image, distance of the number plate in the image and font making recognition difficult and less accurate.

1. **Weather condition**s – According to the weather conditions the image might contain raindrops, fog, etc.

2. **Rotated images** - Images of license plates taken from some angle.

3. **Farther images** - Images taken from a distance having a smaller area of license plate.

4. **Misplaced Characters** - Characters having similarities or looking alike such as 8 and B, 5 and S, etc.

We will improve upon these limitations by creating a custom dataset to train the model in all aspects.

CHAPTER 3
REQUIREMENT GATHERING

3.1 Number Plate Data

We have gathered a dataset of 4954 images by combining datasets from different sources - dataset containing 603 images of vehicles of different Indian states annotated in PASCAL V2 format then converted to YOLO format as shown in figure 3.1 from Kaggle by Sai Sirisha N, Pranav Kant Gaur and Vikshipt, dataset containing 4191 images of vehicles of Brazil annotated in YOLO format and dataset containing 160 images captured from a video taken in the state of Odisha from Kaggle by Deepak

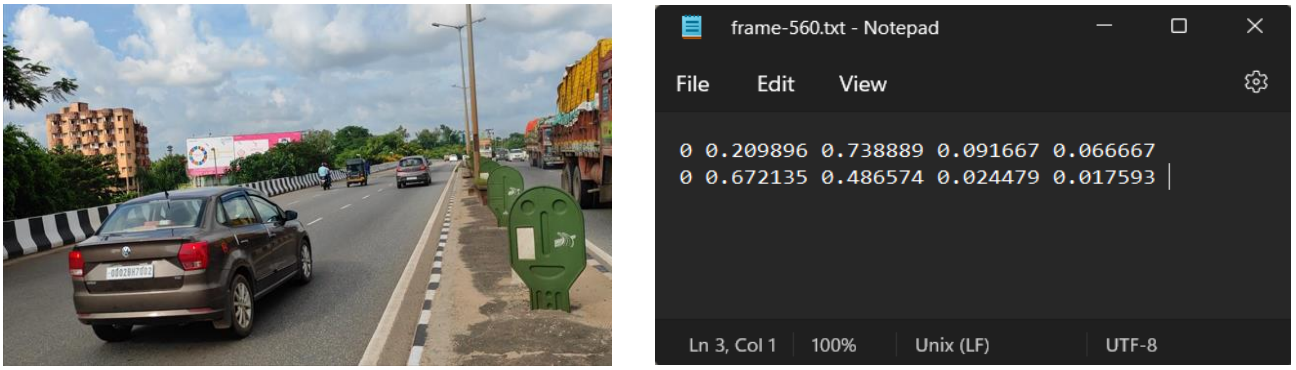


Figure 3. 1 Dataset for YOLO

3.2 Character Recognition Data

We gathered a dataset having a total of 862 images as shown in figure 3.2 for training and 216 images for validation of 36 classes (0-9 and A-Z) from Kaggle for character recognition.

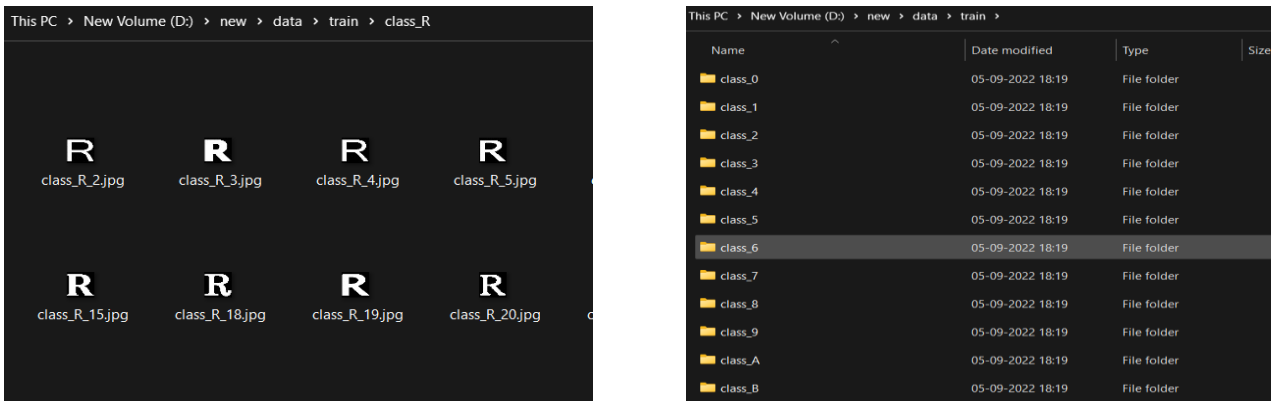


Figure 3. 2 Dataset for CNN

CHAPTER 4
MODULE DESCRIPTION

The Automatic Number Plate Detection System consists of 5 modules as shown in Figure 4.1. They are as follows:

1. **Image Acquisition:** Gathering the dataset required for training the YOLOv3 model and CNN model.
2. **Image Pre-processing:** Annotating the images in YOLO format for YOLO model training and creating classes for CNN model training.
3. **Number Plate detection:** Detecting number plate from vehicle images using the trained YOLO model and contour method (Image Processing techniques).
4. **Character Segmentation:** Segmenting characters from the detected number plate using image processing techniques
5. **Character Recognition:** Recognizing characters obtained from segmentation using trained CNN model.

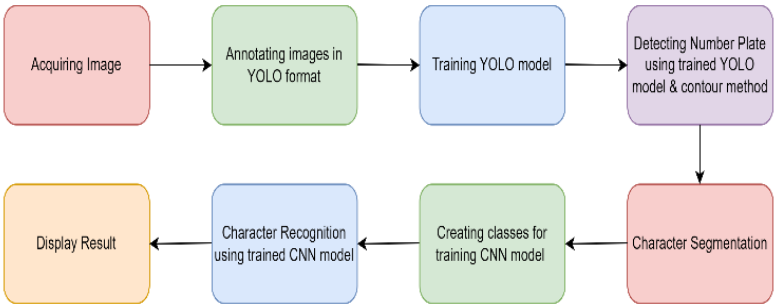


Figure 4. 1 System Architecture

4.1 Number Plate Detection

4.1.1 YOLO

An image is divided into feature vectors having the following properties - Pc(presence of object), Bx (abscissa of center of B.B), By (ordinate of center of B.B), Bw (width of the B.B), Bh (height of the B.B) and C1, C2 (class probabilities).

The image is divided into m x m grids with feature vectors of their own displaying the corresponding values only if the centroid of the object is present in the particular grid creating the final feature vector of m x m x w (w denoting size of feature vector of one grid)

dimension.

For a particular object multiple bounding boxes are created with corresponding class probabilities IOU (Intersection over Union) for all the bounding boxes generated for a particular class are compared and the one with the maximum IOU is selected.

The prediction of Number Plate by YOLO model is shown below in Figure 4.2

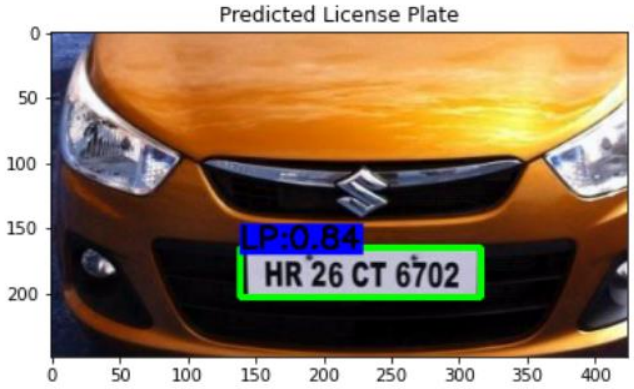


Figure 4. 2 License Plate detection using YOLO

4.1.2 Contour Method

Number Plate detection by contour method follows a set of procedures as depicted in Figure 4.3. These steps include - resizing the image to 500px width, converting to grayscale, removing noise, binarizing, finding contours and drawing contour if it is in quadrilateral form with area greater than 30.

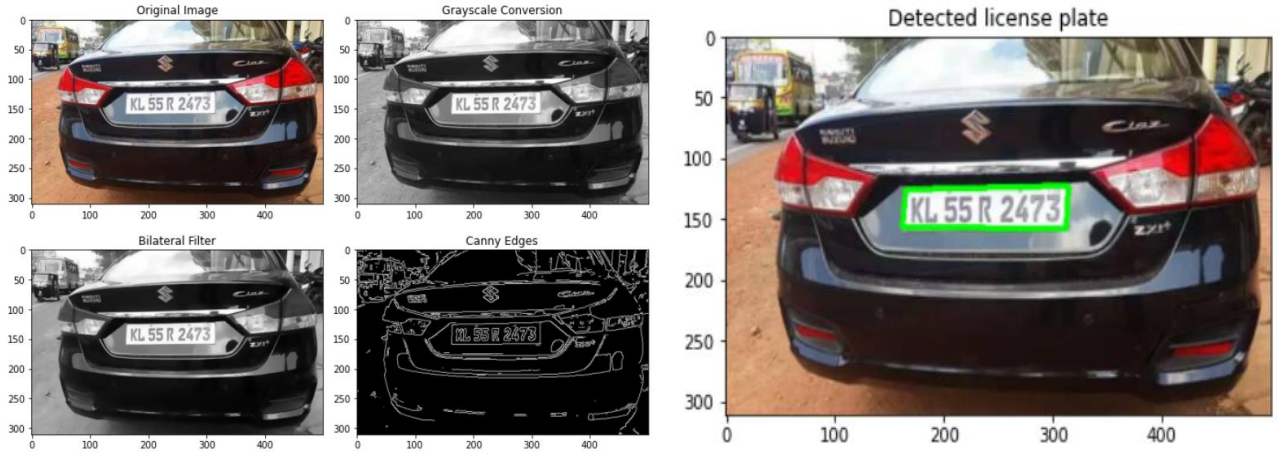
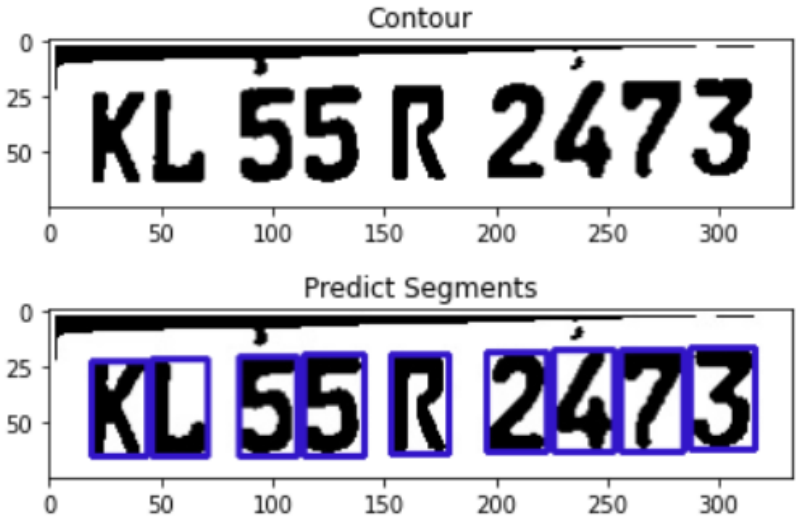


Figure 4. 3 License Plate Detection using Contour Method

4.2 Character Segmentation

The following image processing techniques are used for segmentation of characters from the detected license plate – Gray Scale Conversion, Thresholding for converting to binary image,

eroding for noise removal, Dilation to fill up absent pixels, creating a list of dimensions and comparing them to create the smallest box possible around each character. The segmentation of individual characters by contour method is depicted below in Figure 4.4



2

Figure 4. 4 Character Segmentation using Image Processing Techniques

4.3 Character Recognition

Character Recognition is carried out using a trained CNN model. The model as shown in Figure 4.5 by extracting features from a character, making it into a filter and sliding it over the entire image to create a feature map.

Two main parts of the CNN architecture -

- A convolution tool with several pairs of Convolution+ReLu and pooling layers that isolates and identifies the distinct features of the image for feature extraction.
- A fully connected layer that utilizes the results of the convolutional process and establishes the image class utilizing the features that were previously extracted.

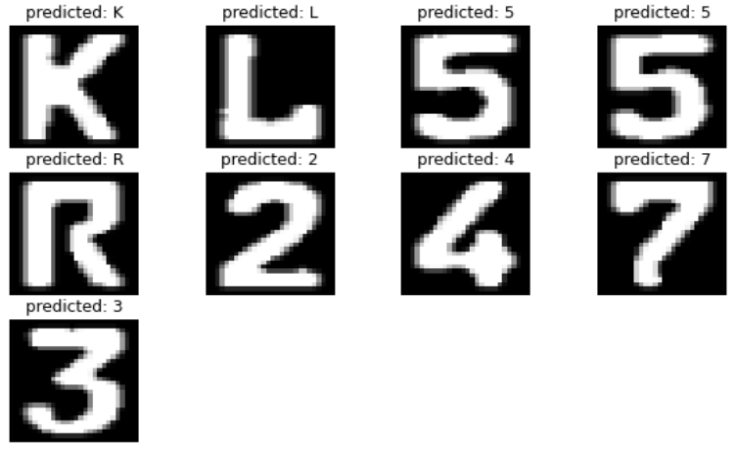


Figure 4. 5 Predicted Characters using CNN

CHAPTER 5

CODE AND TESTING

5.1 Code Snippets for Number Plate detection using YOLO

YOLO Plate Detection

Initialize the parameters

```
# Initialized the parameters

confThreshold = 0.5  #Confidence threshold
nmsThreshold = 0.4  #Non-maximum suppression threshold

inpWidth = 416     #Width of network's input image
inpHeight = 416     #Height of network's input image
```

```
# Loaded names of classes
classesFile = "/content/drive/MyDrive/yolo_utils/classes.names";
```

```
# Appended all different classes into the list 'classes'
classes = None
with open(classesFile, 'rt') as f:
    classes = f.read().rstrip('\n').split('\n')
```

```
# Gave the configuration and weight files for the model and loaded the network using them.
modelConfiguration = "/content/drive/MyDrive/yolo_utils/yolov3_testing.cfg";
modelWeights = "/content/drive/MyDrive/yolo_utils/new_yolov3_training_last.weights";
```

After importing the config and weights file, we set up the model using `cv2.dnn.readNetFromDarknet()` function.

```
net = cv2.dnn.readNetFromDarknet(modelConfiguration, modelWeights)
net.setPreferableBackend(cv2.dnn.DNN_BACKEND_OPENCV)
net.setPreferableTarget(cv2.dnn.DNN_TARGET_CPU)
```

Figure 5. 1 Parameters for YOLO prediction

```
# Drawing the predicted bounding box
def drawPred(classId, conf, left, top, right, bottom, frame):
    # Drawing a bounding box.
    cv2.rectangle(frame, (left, top), (right, bottom), (0, 255, 0), 3)

    label = '%.2f' % conf

    # Get the label for the class name and its confidence
    if classes:
        assert(classId < len(classes))
        label = '%s:%s' % (classes[classId], label)

    #Displaying the label at the top of the bounding box
    labelSize, baseLine = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, 0.5, 1)
    top = max(top, labelSize[1])
    cv2.rectangle(frame, (left, top - round(1.5*labelSize[1])), (left +round(1.5
                    *labelSize[0]), top + baseLine), (0, 0, 255), cv2.FILLED)

    cv2.putText(frame,label,(left,top),cv2.FONT_HERSHEY_SIMPLEX,0.75,(0,0,0),2)
```

```
# Removing the bounding boxes with low confidence using non-maxima suppression
def postprocess(frame, outs):
    frameHeight = frame.shape[0]
    frameWidth = frame.shape[1]


    classIds = []
    confidences = []
    boxes = []
    for out in outs:
        #print("out.shape : ", out.shape)
        for detection in out:
            #if detection[4]>0.001:
            scores = detection[5:]
            classId = np.argmax(scores)
            #if scores[classId]>confThreshold:
            confidence = scores[classId]
            '''if detection[4]>confThreshold:
                print(detection[4], " - ", scores[classId],"-th:" confThreshold)
                print(detection)'''
            if confidence > confThreshold:
                center_x = int(detection[0] * frameWidth)
                center_y = int(detection[1] * frameHeight)
                width = int(detection[2] * frameWidth)
                height = int(detection[3] * frameHeight)
                left = int(center_x - width / 2)
                top = int(center_y - height / 2)
                classIds.append(classId)
                confidences.append(float(confidence))
                boxes.append([left, top, width, height])
```

Figure 5. 2 Functions used in YOLO

```
while cv2.waitKey(1) < 0:

    hasFrame, frame = cap.read() #frame: an image object from cv2

    # Stop the program if reached end of video
    if not hasFrame:
        print("Done processing !!!")
        break

    # Create a 4D blob from a frame.
    blob=cv2.dnn.blobFromImage(frame,1/255,(inpWidth,inpHeight),[0,0,0],1,crop=False)

    # Sets the input to the network
    net.setInput(blob)

    # Runs the forward pass to get output of the output layers
    outs = net.forward(getOutputsNames(net))

    # Remove the bounding boxes with low confidence
    cropped = postprocess(frame, outs)

    plt.imshow(frame)
    plt.title("Predicted License Plate")
    plt.show()
    plt.imshow(cropped)
    plt.title("Cropped Image")
    plt.show()
```

Figure 5. 3 Number Plate detection using YOLO

5.2 Code Snippets for Number Plate detection using Contour Method

```
image = imutils.resize(image, width=500)
img=cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Display the original image
fig, ax = plt.subplots(2, 2, figsize=(10,7))
ax[0,0].imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
ax[0,0].set_title('Original Image')

# RGB to Gray scale conversion
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
ax[0,1].imshow(gray, cmap='gray')
ax[0,1].set_title('Grayscale Conversion')

# Noise removal with iterative bilateral filter
gray = cv2.bilateralFilter(gray, 11, 17, 17)
ax[1,0].imshow(gray, cmap='gray')
ax[1,0].set_title('Bilateral Filter')

# Find Edges of the grayscale image
edged = cv2.Canny(gray, 170, 200)
ax[1,1].imshow(edged, cmap='gray')
ax[1,1].set_title('Canny Edges')

fig.tight_layout()
plt.show()

# Find contours based on Edges
cnts = cv2.findContours(edged.copy(), cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)[0]
cnts=sorted(cnts, key = cv2.contourArea, reverse = True)[:30]
NumberPlateCnt = None #we currently have no Number plate contour

# loop over our contours to find the best possible approximate contour of number
count = 0
for c in cnts:
        peri = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.02 * peri, True)
        if len(approx) == 4:  # Select the contour with 4 corners
            NumberPlateCnt = approx #This is our approx Number Plate Contour
            x,y,w,h = cv2.boundingRect(c)
            ROI = img[y:y+h, x:x+w]
            break
```

Figure 5. 4 Image Processing Techniques used in Contour Method

5.3 Code Snippets for Character Segmentation

```python
# Find characters in the resulting images
def segment_characters(image) :

    # Preprocess cropped license plate image
    img_lp = cv2.resize(image, (333, 75))
    img_gray_lp = cv2.cvtColor(img_lp, cv2.COLOR_BGR2GRAY)
    _, img_binary_lp = cv2.threshold(img_gray_lp, 200, 255, cv2.THRESH_BINARY)
    img_binary_lp = cv2.dilate(img_binary_lp, (3,3))

    LP_WIDTH = img_binary_lp.shape[0]
    LP_HEIGHT = img_binary_lp.shape[1]

    # Make borders white
    img_binary_lp[0:3,:] = 255
    img_binary_lp[:,0:3] = 255
    img_binary_lp[72:75,:] = 255
    img_binary_lp[:,330:333] = 255

    # Estimations of character contours sizes of cropped license plates
    dimensions = [LP_WIDTH/6,
                       LP_WIDTH/2,
                       LP_HEIGHT/10,
                       2*LP_HEIGHT/3]
    plt.imshow(img_binary_lp, cmap='gray')
    plt.title('Contour')
    plt.show()
    cv2.imwrite('contour.jpg',img_binary_lp)

    # Get contours within cropped license plate
    char_list = find_contours(dimensions, img_binary_lp)

    return char_list
```

Figure 5. 5 Character Segmentation using Image Processing Techniques

5.4 Code Snippets for Character Recognition

```python
# Create a new model instance
model = Sequential()

model.add(Conv2D(16, (22,22), input_shape=(28, 28, 3), activation='relu', padding='same'))
model.add(Conv2D(32, (16,16), input_shape=(28, 28, 3), activation='relu', padding='same'))
model.add(Conv2D(64, (8,8), input_shape=(28, 28, 3), activation='relu', padding='same'))
model.add(Conv2D(64, (4,4), input_shape=(28, 28, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(4,4)))

model.add(Dropout(0.4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(36, activation='softmax'))

# Restore the weights
model.load_weights('/content/gdrive/MyDrive/cc3/my_checkpoint')
```

```
<tensorflow.python.training.tracking.util.CheckpointLoadStatus at 0x7f7011119510>
```

```python
# Predicting the output
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), interpolation=cv2.INTER_AREA)
        img = fix_dimension(img_)
        img = img.reshape(1,28,28,3) #preparing image for the model
        y_ = model.predict(img)[0]#predicting the class
        for j in range(36):
          if y_[j]>0.9:
            character = dic[j]
            output.append(character) #storing the result in a list
    plate_number = ''.join(output)
    return plate_number
print(show_results())
```

Figure 5. 6 Character Recognition using CNN Trained Model

5.5 Result

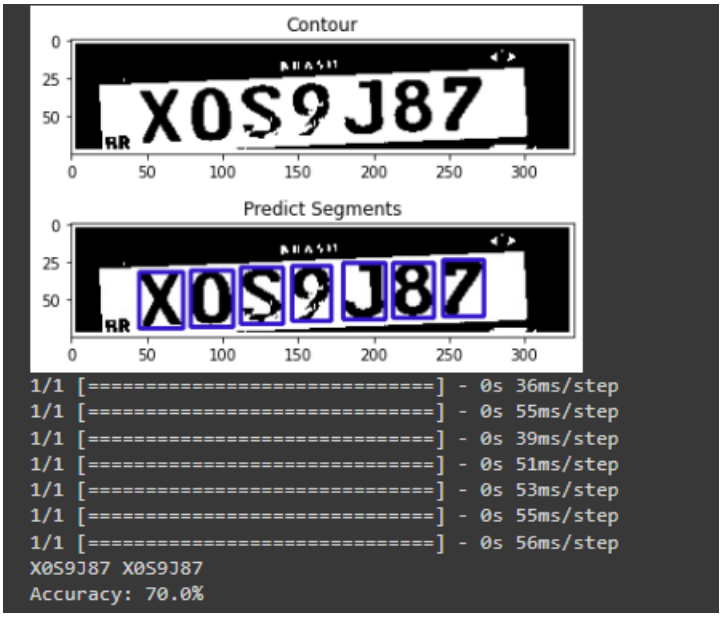


Figure 5. 7 Prediction using YOLO + CNN

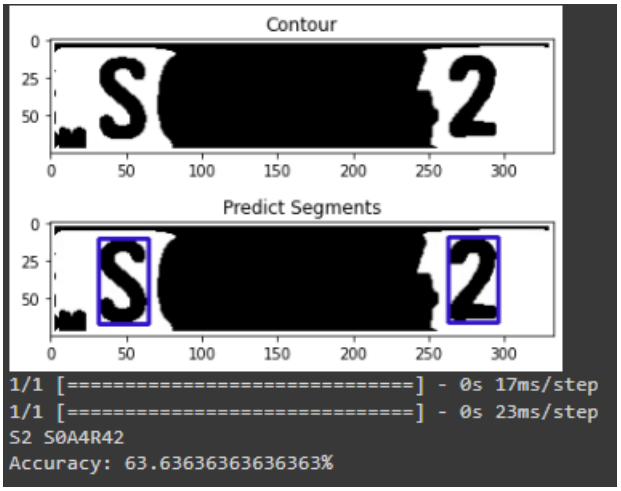


Figure 5. 8 Prediction using Contour method + CNN

Figure 5. 9 Prediction using Hybrid Method

CHAPTER 6

EXPERIMENTAL ANALYSIS

The YOLO and CNN models were trained using Alexey AB darknet and TensorFlow Framework respectively in NVIDIA-SMI GPU with CUDA integration.

The trained YOLO model gave an average loss of 0.166, average IOU of 0.886 in 1600 epochs.

The CNN model trained for 25 epochs gave results as in Table 1.

	Accuracy	Loss
Training	0.9826	0.0480
Validation	0.9630	0.0785

Table 6. 1 Training Curve of CNN model

The CNN training curve with accuracy and loss of training set and validation set is depicted in Figure 6.

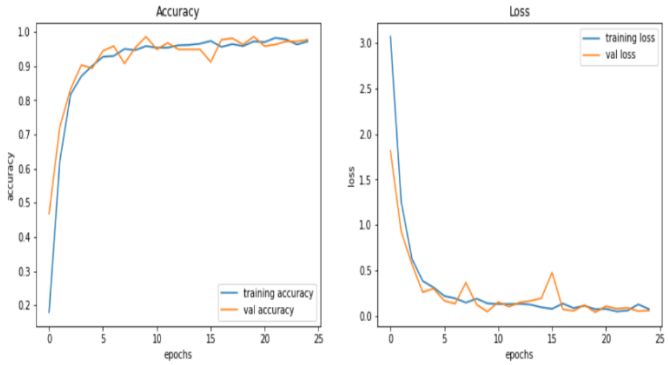


Figure 6. 1 CNN training and validation set curve

CHAPTER 7
CONCLUSION AND FUTURE WORK

The number plate detection using YOLO with CNN on our test dataset gave an accuracy of 70% and that of Contour Method with CNN gave an accuracy of 63.6%. The Hybrid model combining these two methods i.e. YOLO and Contour for Plate recognition and CNN for character recognition gave us an accuracy of 90.91%.

Images which are closer and at an angle are better detected by Contour method.

The YOLO approach was used to alleviate the problem of the contour method having trouble detecting images that are farther away or have number plate colors that are similar to that of the vehicle. Consequently, utilizing a hybrid technique produced greater outcomes than using each one separately.

Provided that the project is further worked upon, we can implement real-time number plate recognition that has a lot of use-cases. For example, it can be used as an added security feature. The awareness that a person's license plate is being recorded and examined is sufficient to deter criminal activity in advance.

It can be improved to become an automated service. If the project can achieve high accuracy readings, it can be used to monitor parking solutions in an effective and affordable manner. In the past, it took time to record number plates, and it took much longer to notify offenders of traffic regulations of their penalties.

However, with the help of this effort, number plates can be quickly identified and compared to the database. Like FASTAG, we can implement number plate recognition while cars are in motion that can be beneficial in numerous sectors of the industry.

REFERENCES

[1] Abhishek Kashyap, B. Suresh, Anukul Patil, Saksham Sharma, Ankit Jaiswal, Automatic Number Plate Recognition, 2018

[2] Arun Vaishnav, Manju Mandot, An Integrated Automatic Number Plate Recognition for Recognizing Multi Language Fonts, 2018

[3] B. Pechiammal & Dr. J. Arokia Renjith, An Efficient Approach for Automatic License Plate Recognition System, 2017

[4] Praveen Ravirathinam & Arihant Patawari, Automatic License Plate Recognition for Indian Roads Using Faster-RCNN, 2019

[5] Crystal Dias, Astha Jagetiya, Sandeep Chaurasia, Anonymous Vehicle Detection for Secure Campuses: A Framework for License Plate Recognition using Deep Learning, 2019

[6] Chinmaya Kumar Sahu, Sushree Barsa Pattnayak, Susantini Behera, Manas Ranjan Mohanty, A Comparative Analysis of Deep Learning Approach for Automatic Number Plate Recognition, 2020

[7] Aiswarya Menon, Bini Omman, Detection and Recognition of Multiple License Plate from Still Images, 2018

[8] Ana Riza F. Quiros, Rhen Anjerome Bedruz, Aaron Christian Uy, Alexander Abad, Argel Bandala, Elmer P. Dadios, Arvin Fernando, A kNN-based Approach for the Machine Vision of Character Recognition of License Plate Numbers, 2017

[9] Naren Babu R, Sowmya V, Soman K P, Indian Car Number Plate Recognition using Deep Learning, 2019

[10] Shashidhar R, Roopa M, A S Manjunath, Puneeth S B, Santhosh Kumar R, Vehicle Number Plate Detection and Recognition using YOLO- V3 and OCR Method, 2021

[11] Waqar Riaz, Abdullah Azeem, Gao Chenqiang, Zhou Yuxi, Saifullah, Waqas Khalid, YOLO Based Recognition Method for Automatic License Plate Recognition, 2020