ALPR

Automatic License Plate Recognition

Index

Team

Yash Kesarwani

Yolo v5

Shalini Punthambekar

OCR

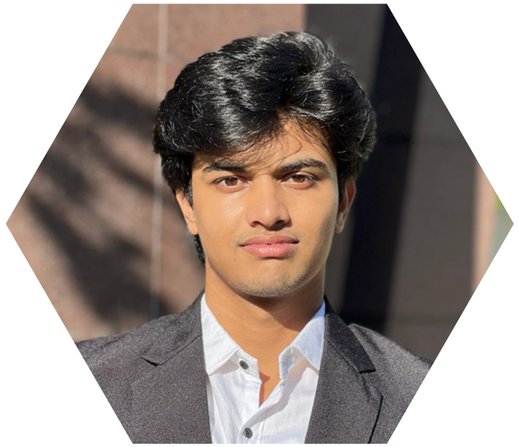
Pooja Sri

Data Collection & Documentation

Team

Shubh Patel

Documentation & Literature Review



Ronit Sahu

Data Preprocessing

Mohammed Mubarizuddin

Deployement & Front End

Introduction

- In populated countries, like India, the traffic on roads can become quite unmanageable which in turns lead to traffic rules violation
- In order to make handling traffic easier for officials, we want to perform Implementation of License Plate Detection to engage in practical scenarios.
- Practical Scenarios include automatic toll collection, traffic law enforcement, private spaces access control, and road traffic monitoring.

About the Dataset

The entire system can be described as 2 key models:

- YOLO model

This model was trained using a dataset comprising various images of cars and other vehicles from different angles

- OCR Model

This model was trained using around 36000 black & white images of 36 characters - 0 to 9 & A to Z. Each class of data contains approximately 1000 images each

We have collected the data from Github

Technology Used

TensorFlow

Streamlit

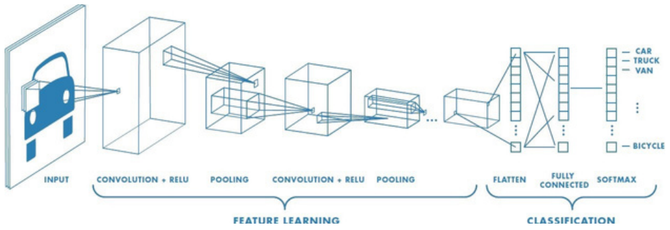
What did we build?

- We designed and developed a system that would be capable of vehicle detection and license plate recognition
- First, we perform object detection to differentiate the vehicle from its background, and then the plate
- Subsequently, we attempt to segment the image such that the image is broken into separate characters - for ease of identification
- We then pass the images to an Optical Character Recognition (OCR) model, giving us the plate number, and record it for further use

Concepts Used

Digital Image Processing

Artifical Neural Networks



Convolutional Neural Networks

OCR (Optical Character Recognition)

Diagrams & Visualization

The YOLO Model

- Input image is an image that contains a car
- The model detects the presence and position of the license plate
- Final output is the cropped image shown below

Diagrams & Visualization

Data Pre-processing

- The bounding box is done on the dilated image of the plate
- The bounding box acts as a tool to segment the digits in the plate
- The final output of this step is an array of segmented digits that will be further used to recognize the digits using OCR

Diagrams & Visualization

OCR Model

- The model makes use of neural network containing dense layers and convolutional layers
- The final output is a string containing the number plate

```
Plate-output: 2A05990
```

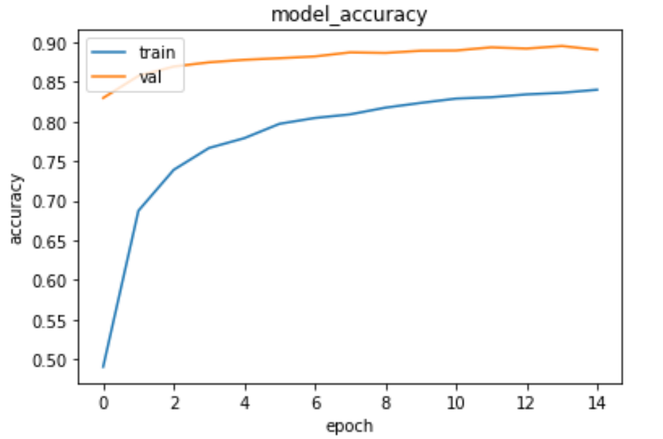
Diagrams & Visualization

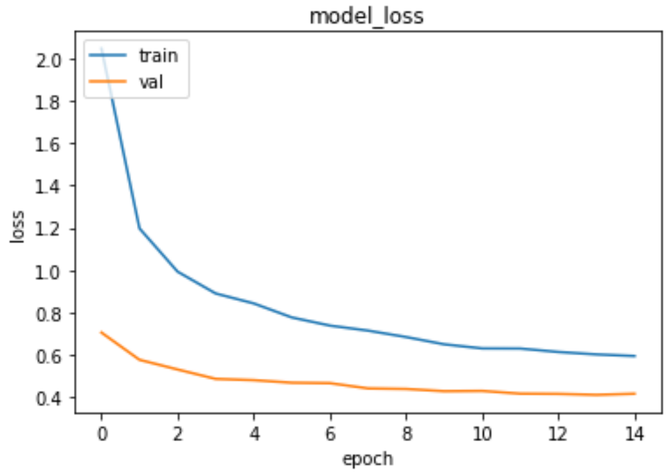
OCR Model - A Comparison

```
w0 = layers.Rescaling(1./255, input_shape=(img_height, img_width, 3))
m5_3 = keras.models.Sequential()
m5_3.add(w0)
m5_3.add(Flatten())
m5_3.add(Dense(100, activation=keras.layers.LeakyReLU(alpha=0.3)))
m5_3.add(Dropout(0.5))
m5_3.add(Dense(50, activation=keras.layers.LeakyReLU(alpha=0.3)))
m5_3.add(Dropout(0.3))
m5_3.add(Dense(36, activation="softmax"))
```

Hyperparameters:
Batch size = 250
Number of Epochs = 15

Train Accuracy = 0.8401
Validation Accuracy = 0.8906





Model5_2.h5

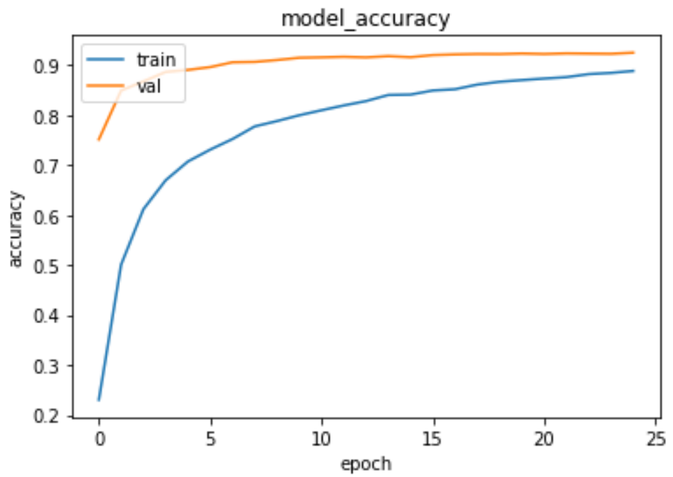
Diagrams & Visualization

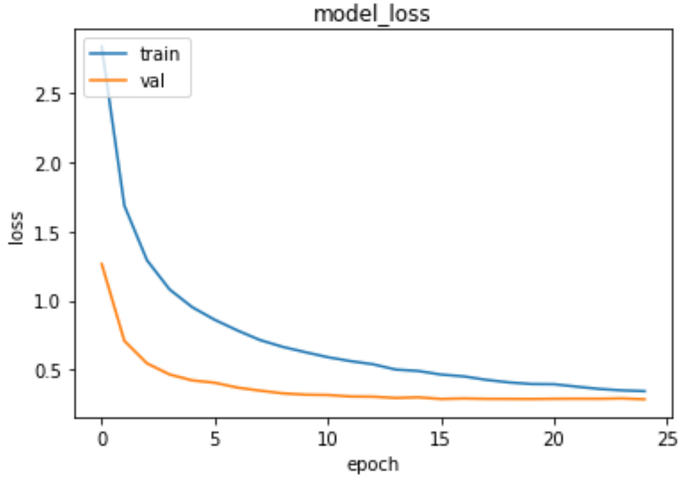
OCR Model - A Comparison

```
m5_4 = keras.models.Sequential([
    layers.Rescaling(1./255, input_shape=(img_height, img_width, 3)),
    layers.Conv2D(32, 3, padding='same', activation='relu'),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(64, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(36, activation="softmax")
])
```

Hyperparameters:
Batch size = 250
Number of Epochs = 25

Train Accuracy = 0.8886
Validation Accuracy = 0.9251



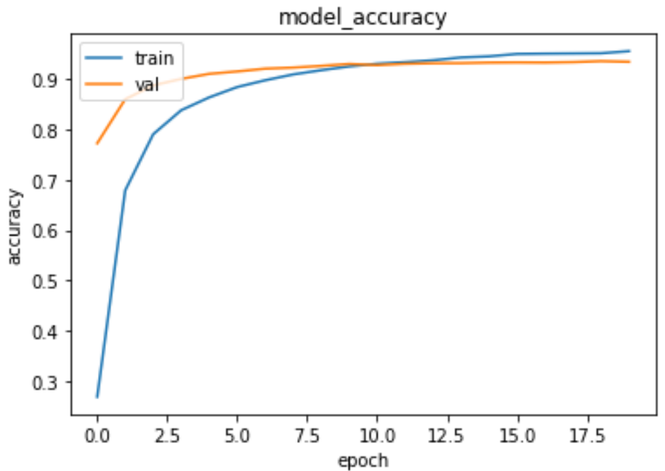


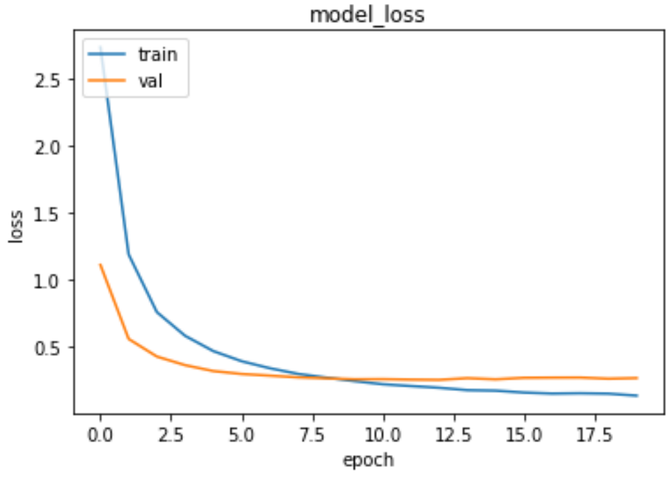
Model5_3.h5

Diagrams & Visualization

OCR Model - A Comparison

```
m5_5 = keras.models.Sequential([
    layers.Rescaling(1./255, input_shape=(img_height, img_width, 3)),
    layers.Conv2D(32, 3, padding='same', activation='relu'),
    layers.Flatten(),
    layers.Dense(128, activation='tanh'),
    layers.Dropout(0.5),
    layers.Dense(64, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(32, activation='relu'),
    layers.Dense(36, activation="softmax")
])
```



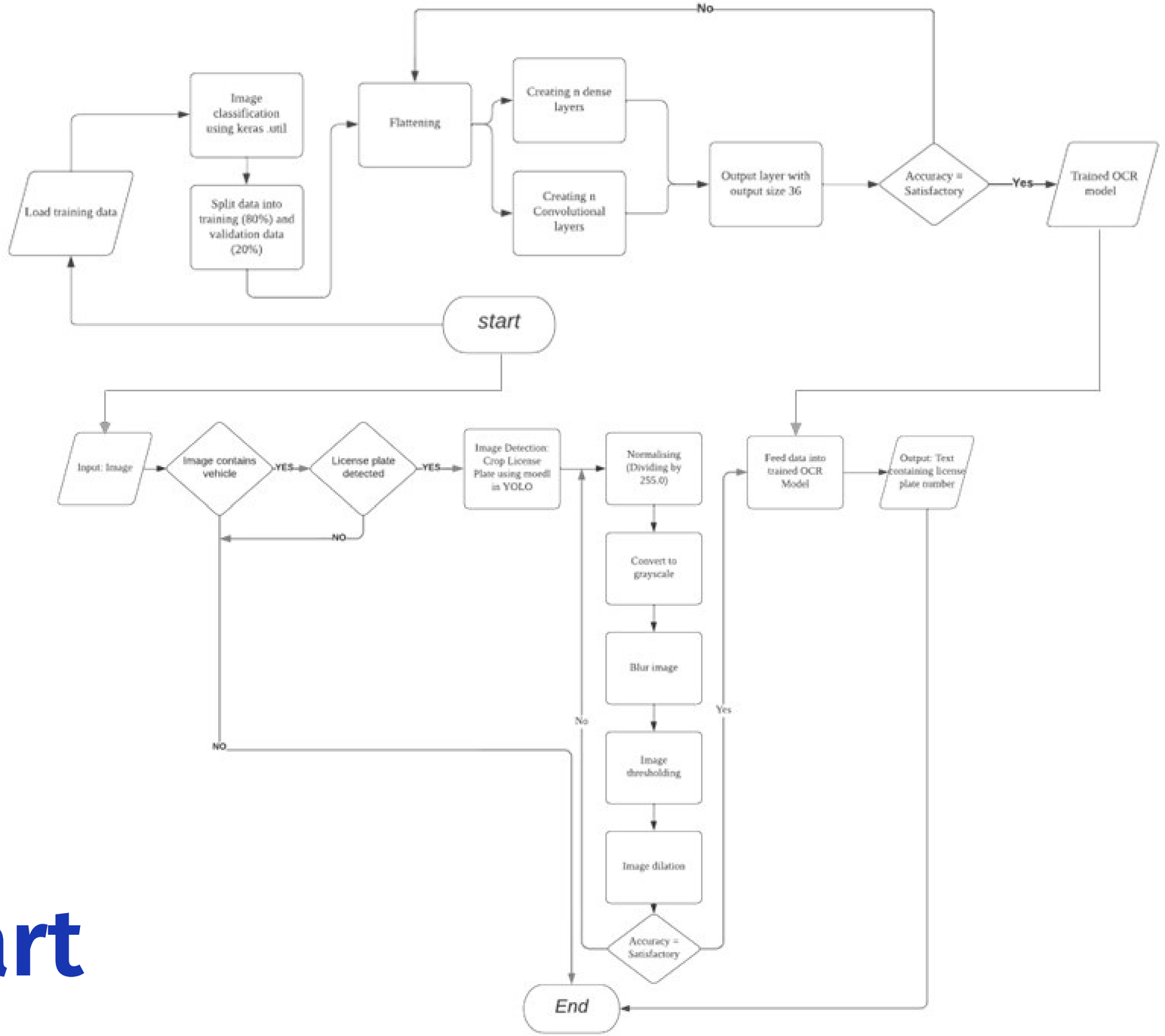


Hyperparameters:
Batch size = 250
Number of Epochs = 20

Train Accuracy = 0.9555
Validation Accuracy = 0.9343

Model5_4.h5

Flowchart


No
Image classification using keras .util
Flattening
Creating n dense layers
Creating n Convolutional layers
Output layer with output size 36
Accuracy = Satisfactory
Yes
Trained OCR model
Load training data
Split data into training (80%) and validation data (20%)
start
Input: Image
Image contains vehicle
YES
License plate detected
YES
NO
Image Detection: Crop License Plate using model in YOLO
Normalising (Dividing by 255.0)
Convert to grayscale
Blur image
Image thresholding
Image dilation
Accuracy = Satisfactory
No
Yes
NO
Feed data into trained OCR Model
Output: Text containing license plate number
End

Challenges Faced

- Obtaining the ideal dataset for training & testing
- Setting up & execution of models on local machines
- Bounding box of plates that have a shape other than a rectangle
- Linking the 2 models built
- Obtaining a preferred accuracy in predicition
- Deployment of the models

Our Learning

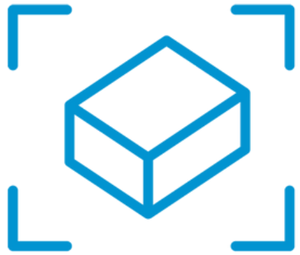


Image detection

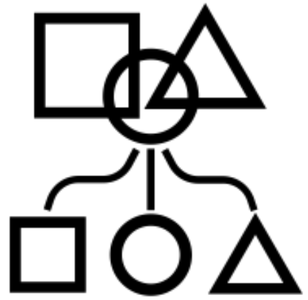
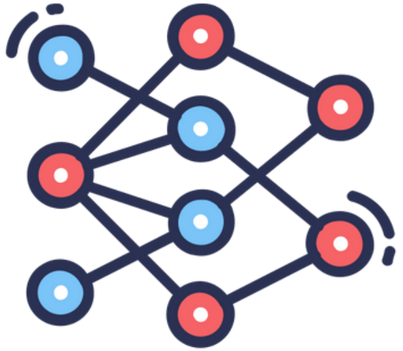


Image classification

Azure



Neural Networks

Business Model

Traffic Violation

ALPR can be used by police or law enforcement to identify traffic violations by any vehicle at a traffic light using a single camera.

Parking Management

ALPR it can be reduce the human interaction required for parking management by using the camera installed in the parking lots to recognize and register the plates at the entrance

Toll Payment

At highways, manual toll booths get hectic and can lead to huge traffic. Using ANPR, toll booths can recognize the license plates and receive payments automatically.

Deployment

Demo

Thank You