A MAJOR PROJECT FINAL REPORT ON

Devanagari License Plate Detection and Recognition for Smart Parking

Submitted by:

Aayushma Paudel [20070546]

Anisha Silwal [20070547]

Garima Paudel [20070551]

Nisha Pokharel [20070557]

Submitted in Partial Fulfillment of the Requirements for

Bachelor of Engineering in Computer Engineering

under **United Technical College**

Submitted to:

Department of Computer Engineering

Supervisor: Er. Sahit Baral

UNITED TECHNICAL COLLEGE

(Affiliated to Pokhara University)

Bharatpur-11, Bhojad, Chitwan

[Subject Code: CMP-490]

July, 2024

Project Work for the Degree of Bachelor's in Computer Engineering

Devanagari License Plate Detection and Recognition for Smart Parking

Supervised by Er. Sahit Baral

A project work submitted in partial fulfillment of the requirements for the degree of Bachelor's in Computer Engineering

Aayushma Paudel	20070546
Anisha Silwal	20070547
Garima Paudel	20070551
Nisha Pokharel	20070557

United Technical College
Faculty of Science and Technology
Pokhara University, Nepal

July, 2024

Dedication

This project is dedicated with profound gratitude to our exceptional teachers, whose unwavering guidance and mentorship have been instrumental in our successful completion of this endeavor. Their wisdom and encouragement have been a guiding light throughout this journey, inspiring us to reach for excellence.

We also extend our heartfelt appreciation to our parents, whose unwavering financial and moral support have sustained us throughout the development of this project. Their belief in our abilities, their sacrifices, and the valuable life lessons they've imparted have shown us that even the most daunting tasks can be conquered, one step at a time.

To our dedicated educators and loving parents, your unwavering faith in us has been the cornerstone of our achievements, and for that, we are eternally grateful.

Declaration

We hereby declare that this study entitled **“Devanagari License Plate Detection and Recognition for Smart Parking”** is based on our original work. Related works on the topic by other researchers have been duly acknowledged. We owe all the liabilities relating to the accuracy and authenticity of the data and any other information included hereunder.

Name of the Students:
Aayushma Paudel
Anisha Silwal
Garima Paudel
Nisha Pokharel

Date: July 19, 2024

Recommendation

This is to certify that this project work entitled **"Devanagari License Plate Detection and Recognition for Smart Parking",** prepared and submitted by **Aayushma Paudel, Anisha Silwal, Garima Paudel, Nisha Pokharel** in partial fulfillment of the requirements of the degree of Bachelor of Computer Engineering awarded by Pokhara University, has been completed under our supervision. We recommend the same for acceptance by Pokhara University.

Signature:

Name of the Supervisor: Er. Sahit Baral

Organization: United Technical College

Date: July 19, 2024

Certificate

This project entitled **“Devanagari License Plate Detection and Recognition for Smart Parking** prepared and submitted by **Aayushma Paudel, Anisha Silwal, Garima Paudel, Nisha Pokharel** has been examined by us and is accepted for the award of the degree of Bachelor in Computer Engineering by Pokhara University.

Name of External: Er. Raj Sapkota Designation: Engineer Organization: Forbes College	Signature:	Date:
Name of Supervisor: Er. Sahit Baral Designation : Head of Department Organization : United Technical College	Signature:	Date:
Name of Principal: Prof. Keshab Datt Awasthi, Ph.D. Organization: United Technical College	Signature:	Date:

Acknowledgements

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this project and who influenced our thinking, behavior and acts during the time of project.

We would like to express our profound gratitude to United Technical College for providing a suitable environment and its contributions to the completion of our project titled “Devanagari License Plate Detection and Recognition for Smart Parking”.

We express our sincere gratitude to Prof. Keshab Datt Awasthi, principal of our college, United Technical College for the support, cooperation and motivation provided during the project.

We would like to express our special thanks to our supervisor Er. Sahit Baral for his time and efforts he provided throughout the project. Your useful advice and suggestions were really helpful to us during the project’s completion. In this aspect, we are eternally grateful to you.

We would like to acknowledge that this project was completed entirely by our group and not by someone else.

Thanks for all your encouragement!

Name of the Students:

Aayushma Paudel

Anisha Silwal

Garima Paudel

Nisha Pokharel

Date: July 19, 2024

Abstract

The “Devanagari License Plate Detection and Recognition for Smart Parking” aims to manage parking spaces through advanced image processing and deep learning techniques. This project involves capturing 942 vehicle images under diverse conditions, followed by preprocessing, cropping, and annotating these images with Roboflow to train a YOLOv9 model. The dataset was split into training (70%), validation (20%), and testing (10%) sets, with preprocessing including resizing to 640x640 pixels and applying various augmentations to enhance model generalization. The YOLOv9 model, pretrained on the COCO dataset, was fine-tuned specifically for license plate localization, and EasyOCR was employed for character recognition. Over 50 epochs, the model showed significant improvements in training metrics, with box loss decreasing from 4.8124 to 1.2199, classification loss from 4.7545 to 0.58378, and distribution focal loss from 5.4523 to 1.6743. Validation metrics also improved, demonstrating good generalization. Key performance metrics include a precision increase from 0.26895 to 0.97836 and a recall rise from 0.38095 to 0.98413, with mAP at IoU 0.5 improving from 0.19138 to 0.97672. The trained model was integrated into a web-based platform using HTML, CSS, and Django. This platform allows users to upload images, detect license plates, extract characters, and store this information in a database. Additionally, users can view real-time parking space availability through the web application. The system is deployed locally and currently serves as a prototype that only works with images.

Keywords: *ANPR, Precision, Recall, EasyOCR, Preprocessing, Augmentation, YOLOv9 algorithm*

Table of Contents

List of Figures

List of Equations

Acronyms and Abbreviation

AI	:	Artificial Intelligence
AHE	:	Adaptive Histogram Equalization
ALPR	:	Automatic License Plate Recognition
ANPR	:	Automatic Number Plate Recognition
BBA	:	Boundary Box Analysis
CCA	:	Connected Component Analysis
CSS	:	Cascading Style Sheets
DWT	:	Discrete Wavelet Transform
IOU	:	Intersection Over Union
maP	:	Mean Average Precision
MSE	:	Mean Square Error
NMS	:	Non-Max Suppression
OCR	:	Optical Character Recognition
PSNR	:	Peak Signal to Noise Ratio
R-CNN	:	Region Based Convolutional Neural Network
SSD	:	Single Shot Multibox Detector
SVM	:	Support Vector Machine
YOLO	:	You Only Look Once

Chapter 1: Introduction

1.1. Background

As cities of Nepal are getting crowded and space is becoming limited, managing parking efficiently becomes important. One smart solution is using Automatic Number Plate Detection and extraction of Devanagari characters of detected license plates for record keeping in parking systems. This technology aims to automate the process of entering and leaving parking areas, making it easier for everyone. This makes sure that parking is well recorded and well organized.

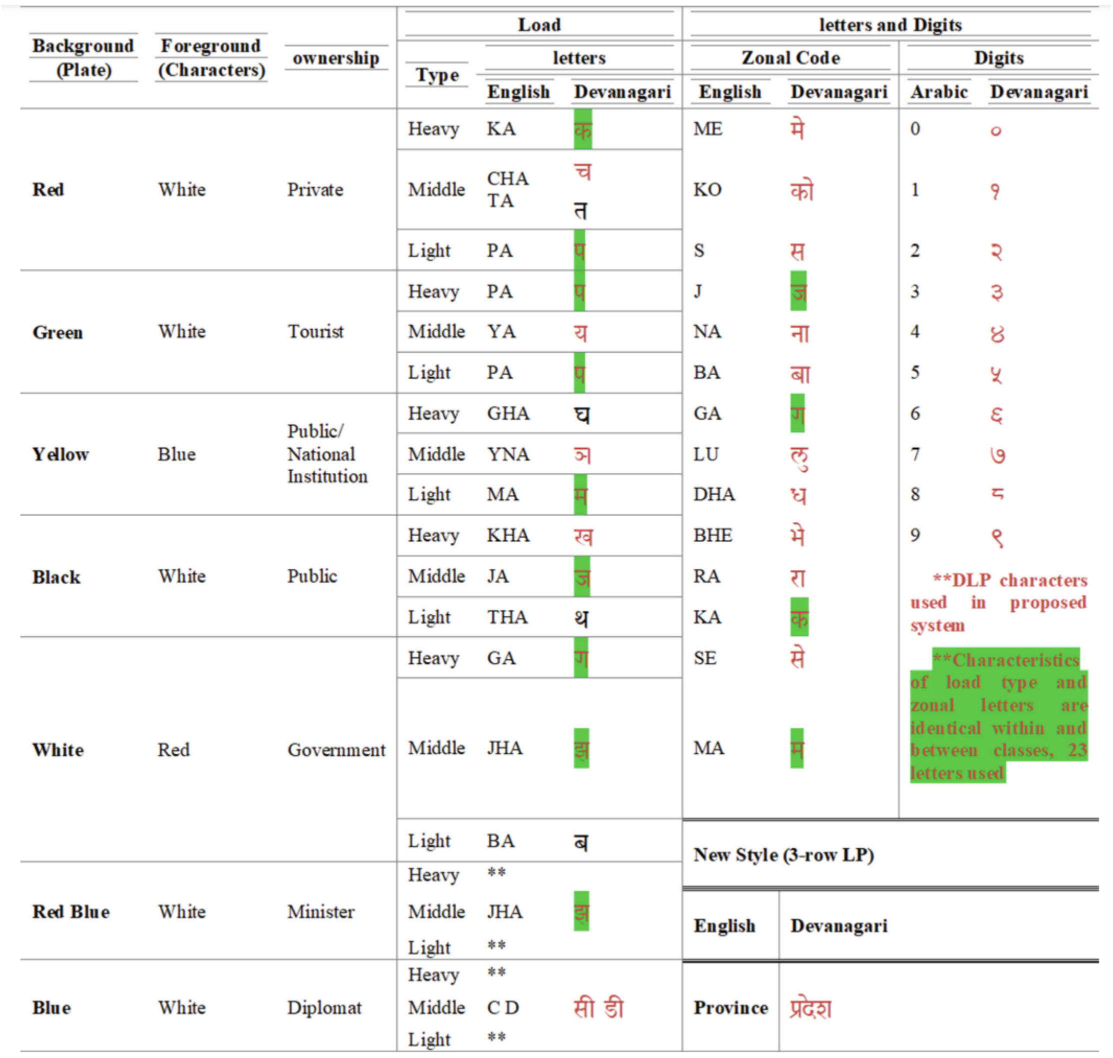
Automatic Number Plate Recognition (ANPR) system is a crucial component of smart cities as it includes number plate detection and Optical Character Recognition (OCR) technology to read detected vehicle number plates. The detection task can be done using several alternatives like Faster R-CNN, SSD (Single Shot Multibox Detector), RetinaNet etc. YOLO model has been selected for the project because unlike some algorithms requiring multiple passes, YOLO conducts detection in a single forward pass, making it computationally efficient and suitable for real-time applications like video analysis. ANPR enables traffic control and law enforcement through an automated, fast, reliable, and robust vehicle plate recognition system [1]. License plate readers employ Optical Character Recognition (OCR) technology to quickly and accurately interpret vehicle information. To automatically identify and monitor vehicles as they enter and depart, ANPR systems are often employed in parking garages, toll booths, and traffic control systems. An automated vehicle entry system using ANPR typically consists of a camera, a processor, and a database. The camera is mounted at the facility's entrance and captures images of the license plates of incoming vehicles. The license plate’s characters are read using OCR [2].

Identifying vehicles within a country relies on the recognition of a distinctive alphanumeric number displayed on their license plates, and localizing these plate regions in images presents a challenging task due to variations in color, texture, size, shape, and position of these regions [3].

Monitoring of vehicle traffic and management of parking lots in crowded and congested locations like shopping malls, business parks, residential complexes etc has emerged as a major area of research in Intelligent Transportation System (ITS). Often parking lot operators do not enter complete details or sometimes enter incorrect details into the system, especially during peak hours, which may later cause problems for vehicle owners while exiting the lot and is also a major security issue [4].

1.1.1. Nepalese License Plate

The Nepalese government has categorized license plates (LPs) into seven classes based on vehicle ownership, denoted by color combinations in the LP's background and foreground. LPs come in three structures: 1-row, 2-row, and 3-row, each with specific character arrangements. These plates include fixed elements like Province, province identifier (PN), plate status (PS), lot number (L), load type (LD), and vehicle identity (X). Various LP structures exist, such as 3-row Red (private) LP and zonal format LPs. Inconsistencies arise in non-standardized LPs, with variations in fonts, character sizes, and spacing [5].



			Load			letters and Digits			
			Type	letters		Zonal Code		Digits	
Background (Plate)	**Foreground (Characters)**	**ownership**		**English**	**Devanagari**	**English**	**Devanagari**	**Arabic**	**Devanagari**
Red	White	Private	Heavy	KA	क	ME	मे	0	०
			Middle	CHA TA	च त	KO	को	1	१
			Light	PA	प	S	स	2	२
Green	White	Tourist	Heavy	PA	प	J	ज	3	३
			Middle	YA	य	NA	ना	4	४
			Light	PA	प	BA	बा	5	५
Yellow	Blue	Public/ National Institution	Heavy	GHA	घ	GA	ग	6	६
			Middle	YNA	ञ	LU	लु	7	७
			Light	MA	म	DHA	ध	8	८
Black	White	Public	Heavy	KHA	ख	BHE	भे	9	९
			Middle	JA	ज	RA	रा	****DLP characters used in proposed system**	
			Light	THA	थ	KA	क		
White	Red	Government	Heavy	GA	ग	SE	से	**Characteristics of load type and zonal letters are identical within and between classes, 23 letters used	
			Middle	JHA	झ	MA	म		
			Light	BA	ब	**New Style (3-row LP)**			
Red Blue	White	Minister	Heavy	**					
			Middle	JHA	झ	**English**	**Devanagari**		
			Light	**					
Blue	White	Diplomat	Heavy	**					
			Middle	C D	सी डी	**Province**	प्रदेश		
			Light	**					

Figure 1.1: License Plate Number System In Nepal [5]

1.2. Problem statement

The current state of parking lots in Nepal faces several challenges that hinder efficient and organized operations. One of the primary issues is the reliance on manual record-keeping, which not only consumes valuable human resources but is also prone to errors and inefficiencies. The absence of a robust automated parking system utilizing Automatic Number Plate Recognition (ANPR) technology worsens the problem. The lack of automation not only results in a difficult process for both parking lot management and drivers but also contributes to the absence of a reliable record-keeping system for vehicles entering and leaving the premises.

Furthermore, there is a notable absence of a centralized platform for parking lot managements to access real-time information about vehicles within their facilities. Additionally, drivers lack a user-friendly platform to ascertain the availability of parking spaces in real-time, leading to frustration and inefficiencies in their search for suitable parking spots.

In the context of Nepal, where automation systems for parking management are yet to be implemented, the need for a comprehensive solution is urgent. Moreover, during our research, we identified a unique challenge specific to Nepal – the difficulty in accurately identifying and processing Nepali letters using existing ANPR systems. This adds an additional layer of complexity to the adoption of automated parking systems in the country. Considering these challenges, our proposed project aims to develop an advanced solution for smart way of parking using ANPR that not only addresses the current limitations in record-keeping and automation but also incorporates solutions for the unique linguistic challenges posed by Nepali letters.

1.3. Research Questions

1. How can YOLOv9 be used to detect license plates?
2. How can Devanagari characters be extracted from the detected license plates?
3. How can we develop a system for proper recording of vehicles entering and exiting a parking lot?

1.4. Objectives

The project aims to develop an affordable, scalable, and accurate ANPR-based system, contributing to improved security and streamlined operations. The objectives of the project are listed below:

1. To implement a system (YOLOv9 Deep Learning Model) for license plate detection.
2. To use Optical Character Recognition for extracting Devanagari characters from the detected license plates.
3. To establish a robust record-keeping mechanism that captures and maintains license plate information of vehicles along with entry and exit time records for effective monitoring.

1.5. Application

The "Devanagari License Plate Detection and Recognition for Smart Parking" project has various applications across different sectors. Some potential applications include:

1. Urban Parking Management: The system can be used for streamlining parking in busy urban areas to reduce congestion and optimize space utilization and for enhancing the efficiency of municipal parking systems for better city planning.
2. Business Parks and Commercial Areas: The proposed system can be used for managing employee and visitor parking in business parks and commercial complexes and for improving the overall traffic flow and parking experience for customers and clients.
3. Event and Venue Management: The system can also be used for facilitating efficient parking solutions for events, concerts, and sports venues and ensuring

a smooth flow of traffic and for minimizing parking-related challenges during large gatherings.

4. Commercial Parking Lots: The system can be used for optimizing parking operations in commercial parking lots, including pay-and-park facilities and for enhancing customer experience through quick and convenient parking solutions.
5. Security and Surveillance Integration: The system can also be used by integrating ANPR technology with security systems for enhanced surveillance and access control and improving overall security measures by identifying and monitoring vehicles entering specific areas.

These applications demonstrate the versatility and wide-reaching impact of the project, offering solutions to various parking challenges across different sectors and environments.

1.6. Scope and Limitations

The project has a wide scope and presents a holistic approach to streamline vehicle monitoring in parking centers by leveraging the YOLOv9 Deep Learning Model for real-time license plate detection. Using YOLOv9 for plate localization and EasyOCR for character extraction, the project holds significant scope in enhancing vehicle identification systems in regions where Devanagari script is prevalent, such as Nepal. This advanced system can be integrated into traffic management and law enforcement agencies to automate the process of vehicle monitoring, thereby improving efficiency and accuracy in tracking and identifying vehicles. Additionally, it can be employed in automated toll collection, parking management systems, and security checkpoints to streamline operations and reduce manual intervention. The implementation of state-of-the-art technologies like YOLOv9 ensures high precision in plate detection even in challenging conditions, while EasyOCR provides robust character recognition, making the system versatile and reliable for real-world applications. This project paves the way for further advancements in multilingual OCR applications and smart city infrastructure development.

Despite its promising scope, the Parking Management System does have certain limitations. The accuracy of license plate detection may be compromised under extreme environmental conditions, such as severe weather or low lighting, which could impact the system's overall reliability. Additionally, there may be initial implementation costs associated with hardware, software, and training. Dependency on hardware reliability poses another limitation, as the system's performance depends on the proper functioning and maintenance of cameras and servers. Addressing data privacy concerns remains crucial, and despite advanced security measures, continuous adaptation to evolving regulations and user expectations is essential. Furthermore, challenges may arise in user training and adoption, requiring careful consideration and proactive measures to ensure a smooth transition for parking administrators and users. Integrating the system with existing infrastructure or legacy systems may also present obstacles, necessitating strategic planning and potential modifications for seamless integration. Understanding these limitations is integral to managing expectations and addressing challenges during the deployment and operation of the system.

Chapter 2: Literature Review

The Literature Review portion of the proposal is based on the observations of many research and journals on similar titles.

According to the paper by Budi Setiyono et.al [6], different processes were performed to detect the number plate correctly such as character recognition was done to get text character data from license plate. YOLOv3 (You Only Look Once), and Darknet-53 were used as feature extractors. The data used were number plate images derived from the extraction and cropping of motorized vehicle videos taken using cellphones and cameras. Two models were tested: one without preprocessing and one with additional data preprocessing to enhance image quality. The results indicate that the model with preprocessing achieves higher accuracy, with 88% in number plate recognition and 98.2% in character recognition. In contrast, the model without preprocessing achieves a lower accuracy, with 80% in number plate recognition and 97.1% in character recognition.

In their work described in reference [7], M. M. Abdellatif et.al used 200 images to identify Egyptian car plates. The model successfully identified Arabic license plates with 93% accuracy. A prototype was implemented using ESP32 Cameras and Raspberry-Pi to test the system's performance.

According to a paper by Mohamed S. Farag et.al [8], image processing was used for smart parking entrance control. Car plate recognition involved preprocessing, license plate detection, character extraction, and recognition. Techniques such as image enhancement, noise reduction, color filtering, DWT (Discrete Wavelet Transform) for feature extraction, and SVM (Support Vector Machine) as a classifier contribute to achieving high detection (97.8%), segmentation (98%), and recognition (97%) rates, making it an effective method for smart parking.

A paper by Hanae Moussaoui et.al [9], introduces a novel Arabic and Latin license plate detection and recognition method. In the first stage of the proposed method and after gathering images to build a new dataset, YOLOv7 was used to detect and localize the license plate in the image. The dataset was labeled manually before feeding it to the detection system. Secondly, some of the machine learning algorithms were used to enhance the detected license plate. Thresholding and the kernel algorithms were used to remove the additional vertical lines in the plate. Afterward, Arabic OCR (Optical Character Recognition) and Easy OCR techniques were used to recognize Arabic and Latin letters on the license plate. The proposed detection algorithm achieved a precision of 97%, a recall of 98% and an F1 score of 98% in license plate detection. For image segmentation, while using Arabic OCR and Easy OCR to segment and extract characters from the detected license plate, an accuracy of 99 % was achieved.

The article in reference [10], focuses on object detection techniques, the YOLO v8 architecture, and prior studies on license plate detection. The authors highlight the unique challenges of recognizing Bangla license plates and discuss relevant literature in the field. They emphasize the significance of YOLO v8 as a state-of-the-art object detection framework known for real-time capabilities. The article addresses the methodology, including data collection, preprocessing, and model training, while discussing the challenges faced, such as data labeling and handling varying plate formats. The YOLO v8 model demonstrated outstanding performance in recognizing Bangla license plates, achieving a mean Average Precision of 98.4% on the test set and a precision of 98.1%. The analysis further revealed the model's robustness in handling diverse license plate variations, emphasizing its potential for real-world applications and ability to generalize well across challenging scenarios. Overall, the YOLO v8 architecture proves highly effective for Bangla license plate recognition.

In their paper presented in reference [11], V. Gnanaprakash, N. Kanthimathi, and N. Saranya propose an Automatic Number Plate Recognition (ANPR) system using deep learning, specifically utilizing the YOLO algorithm for vehicle and license plate detection. The research addresses the challenges of real-time processing of surveillance camera footage for efficient vehicle tracking. A four-step approach, involving video-to-image conversion, car detection, license plate localization, and character recognition is introduced. The deep learning model employs the Image AI library and is trained on a dataset of Tamil Nadu license plate images. The system achieves high accuracy, with 97% for car detection, 98% for number plate localization, and 90% for character recognition. The authors highlight the significance of ANPR in traffic management, parking systems, and security applications, emphasizing the need for real-time, accurate, and automated tracking of vehicles. The proposed system demonstrates advancements over existing methods, leveraging YOLO and Image AI for enhanced performance in dynamic environments. The literature review discusses prior studies on ANPR, including approaches using machine learning, deep learning, and OCR techniques, providing a comprehensive overview of the current state of the art.

In the paper by R. Laroca et *al.* [12], the proposed ALPR system tackles the complexities of detecting LPs in traffic images. Instead of directly working on image frames, it first locates vehicles and then identifies their License Plate (LPs) within the vehicle patches. This two-stage process helps overcome challenges such as confusion with non-LP textual elements. A novel aspect is the introduction of a layout classification stage after LP detection, using a unified network for robust recognition across different LP layouts. The system leverages YOLO-inspired models for vehicles and LP detection, while LP recognition employs CR-NET. The approach proves adaptable to new LP layouts through retraining, offering a promising solution for real-world applications with diverse LP configurations.

The literature by S.Kaur [13] on ANPR systems underscores the advancements in image acquisition and processing. Researchers tackle challenges presented by diverse image categories, including light, dark, low contrast, blurred, and noisy images, with a focus on weather-induced variations. Pre-processing techniques, such as iterative bilateral filters and Adaptive Histogram Equalization (AHE), enhance image quality by improving contrast and reducing noise. Morphological operations, image binarization, and Sobel edge detection contribute to refining candidate plate areas. The literature highlights challenges in character segmentation and recognition, addressed through methods like Connected Component Analysis (CCA) and Boundary Box Analysis (BBA). The proposed ANPR approach proves effective in handling diverse image conditions, exhibiting improvements in metrics like Peak Signal to Noise Ratio (PSNR), Mean Square Error (MSE), and overall success rate when compared to existing methods.

In conclusion, several articles highlight the diverse applications and methodologies employed in automatic license plate recognition (ALPR) using various technologies. YOLO-based models, such as YOLOv3, YOLOv7, YOLO v8, and YOLO-inspired architectures, consistently emerge as powerful tools for efficient vehicle and license plate detection. These models exhibit real-time capabilities and robust performance across different scenarios, contributing to high accuracy rates in number plate recognition and character extraction. The importance of preprocessing techniques, machine learning algorithms, and deep learning approaches is huge in enhancing ALPR systems' overall effectiveness. The results underscore YOLO's versatility and effectiveness in handling challenges related to license plate detection, localization, and character recognition, making it a preferred choice for researchers and developers working on ALPR applications.

Chapter 3: Methodology

3.1. Automatic Number Plate Recognition (ANPR) System

Automatic Number Plate Recognition (ANPR) system is a system that reads and process images that consist of vehicle number plate as input and recognizes the number plate as output automatically [14]. Figure below shows the overview of the ANPR system.

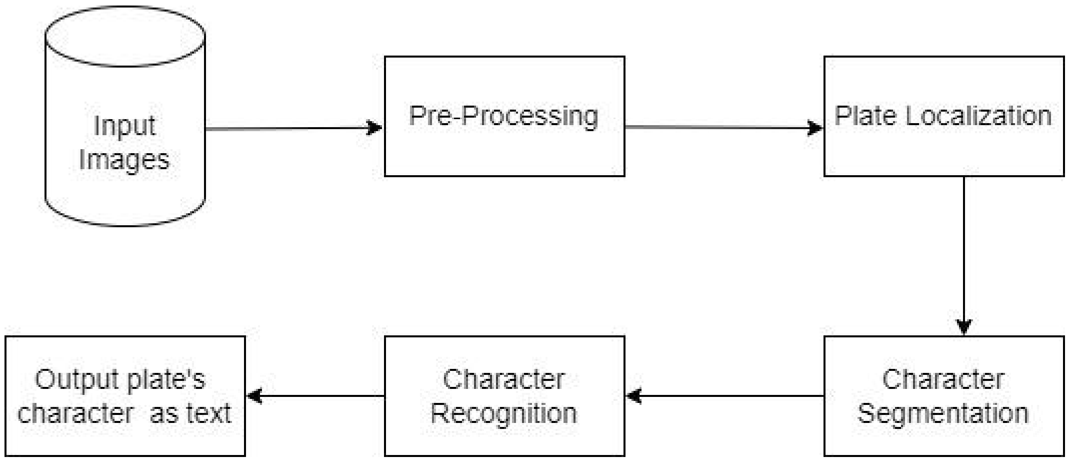


Figure 3.1: Automatic Number Plate Recognition System [15]

The license plate recognition process begins with the input images as the initial step. These images may come from various sources such as cameras or snapshots. The second step involves preparing the image for analysis or human perception through a pre-processing stage. This step ensures that the image is of good quality and suitable for computational processing or visual understanding. Following this, the plate localization stage comes into play, which focuses on identifying and pinpointing the location of the license plate within the image. Once the plate is located, the character segmentation stage takes over, separating each individual letter or number on the plate. This segmentation is crucial for making it easier to identify and analyze the characters. The final phase involves character recognition, where the system identifies and interprets the segmented characters, providing a comprehensive understanding of the license plate information. In essence, the entire process involves preparing the image, finding the license plate, breaking down its characters, and ultimately recognizing and understanding each character for various applications such as automated vehicle identification or security systems [15].

The overall steps involved in Automatic Number Plate Recognition (ANPR) are explained below:

3.1.1. Dataset Collection

In the first step i.e. Data Collection, large amounts of data were gathered by capturing images of vehicles, ensuring clear visibility of number plates. 942 images were collected to form a dataset covering vehicles in different scenarios, ensuring the dataset encompasses various lighting conditions, vehicle types, and plate variations to ensure the model's robustness. The type of data that has been used is a primary source of data or custom dataset [10].

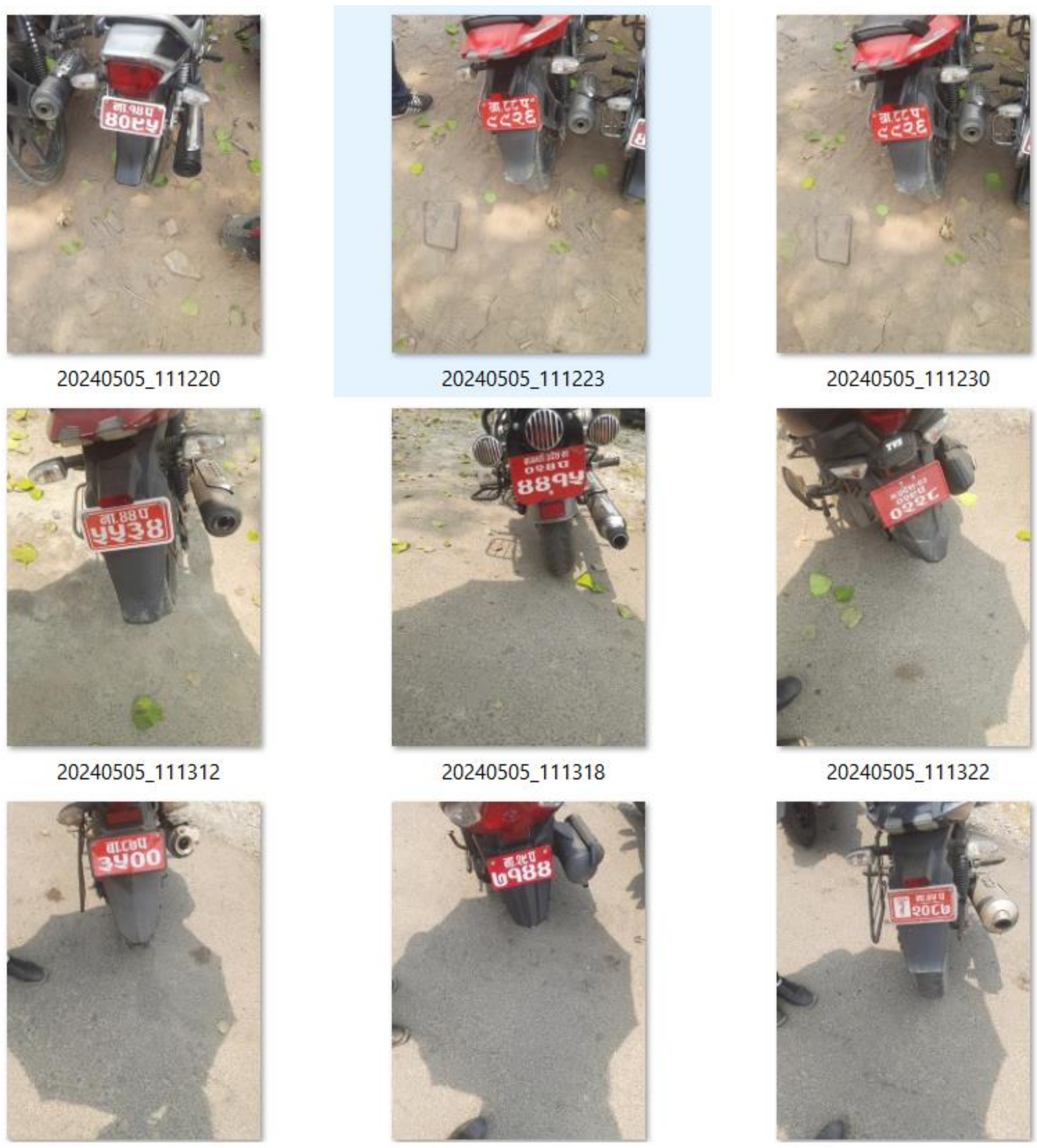


Figure 2.2: Collected Data

3.1.2. Dataset Preparation and Annotation

In this phase, all the collected 942 images were first cropped to ensure that the license plate is clearly visible. This was followed by the dataset annotation step where the images were annotated by marking bounding boxes around the license plates using Roboflow. Roboflow, as a platform providing tools and services for computer vision and image processing tasks, has been utilized to efficiently manage and preprocess the image datasets needed for the project. This annotation step formed the foundation for training the YOLOv9 model to accurately recognize and localize license plates [10].

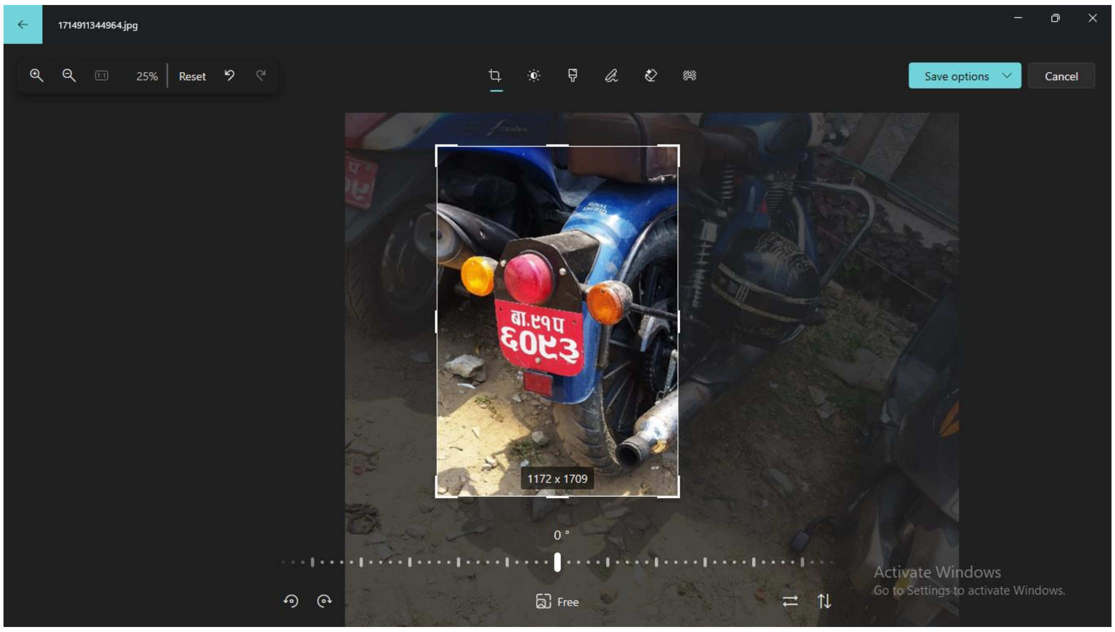


Figure 3.3: Image Cropping

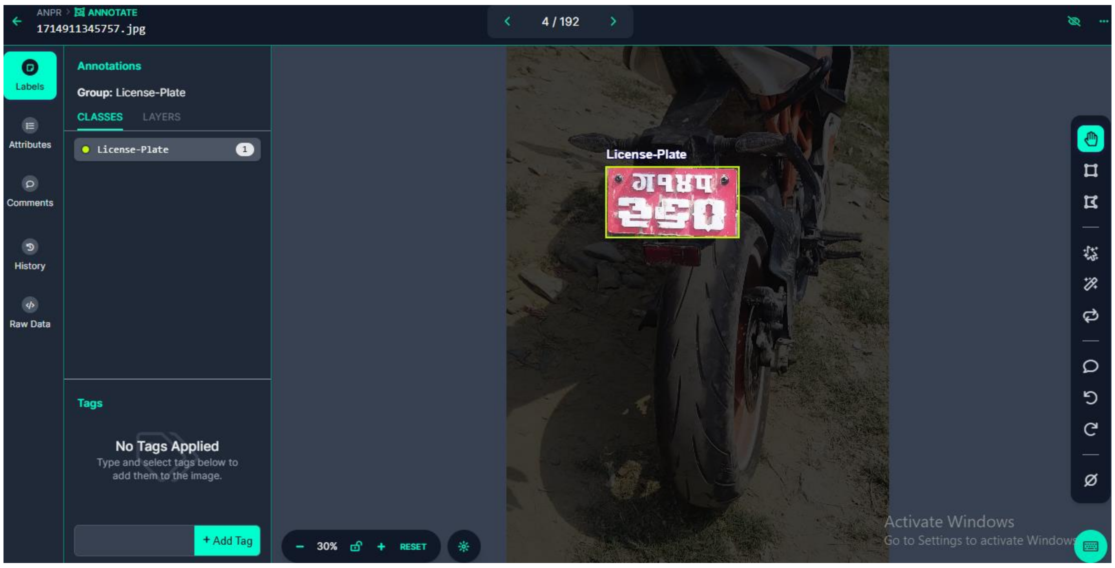


Figure 3.4: Image Annotating

3.1.3. Dataset Splitting

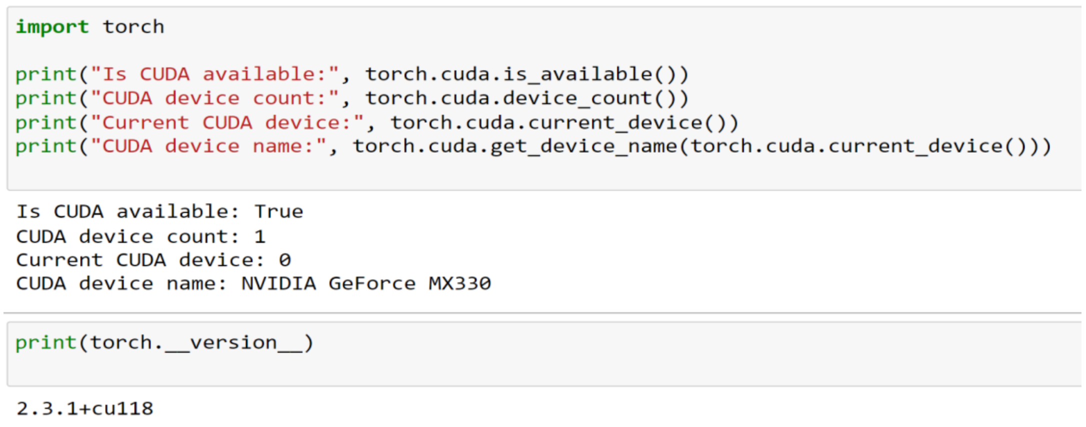
In the Dataset Splitting stage, the dataset was divided into training, validation and testing sets, comprising 70%, 20%, and 10% of the dataset, respectively. This step divided datasets as: 661 images in training set, 188 images in validation set, 94 images in test set.

3.1.4. Preprocessing and Augmentation

In this Preprocessing phase, the Auto-Orient option was enabled to ensure that all images have a consistent orientation. Also, the images were resized i.e. stretched to 640*640. Similarly, several augmentation tasks were also done. Each training example was augmented to produce three variations. The augmentations included random rotations between -15° and +15°, applying grayscale to 15% of the images, adjusting brightness between -20% and +20%, adding blur up to 0.7px, and introducing noise to up to 1.49% of the pixels. This step multiplied the number of images in the training dataset by 3 and the training set reached a total of 1983 images. This way, the dataset of total 2265 images was prepared after this step. This diverse augmentation strategy helped in creating a robust model by exposing it to various transformations of the training data [16].

3.1.5. Training Setup and Configuration

For training purposes, the need for high computation was noticed. So, NVIDIA drivers, CUDA toolkit, and cuDNN were installed. The environment variables were set. This was done with the help of support matrix for CUDA and cuDNN.



```
import torch

print("Is CUDA available:", torch.cuda.is_available())
print("CUDA device count:", torch.cuda.device_count())
print("Current CUDA device:", torch.cuda.current_device())
print("CUDA device name:", torch.cuda.get_device_name(torch.cuda.current_device()))
```

```
Is CUDA available: True
CUDA device count: 1
Current CUDA device: 0
CUDA device name: NVIDIA GeForce MX330
```

```
print(torch.__version__)
```

```
2.3.1+cu118
```

Figure 3.5: Training Setup and Configuration

3.1.6. Model Training

We trained the YOLOv9 model, pre-trained on the COCO dataset, using a batch size of 16, 8 data loading workers, and an image size of 512x512 pixels, on a GPU (device 0) for 50 epochs. The training utilized specific data, model, and hyperparameter configurations provided in YAML files, with the data path specified for the annotated dataset and pre-trained weights initialized from a given checkpoint. This setup ensured efficient data handling and computation, enabling the model to learn effectively and adapt its pre-trained knowledge to the new dataset.

About YOLO Algorithm

YOLO algorithm aims to predict a class of an object and the bounding box that defines the object location on the input image. The main idea behind YOLO is to divide the input image into a grid and make predictions for each grid cell. Unlike traditional object detection methods, YOLO performs object detection in a single forward pass through the neural network.

It recognizes each bounding box using four numbers:

i. Center of the bounding box (b_x, b_y)

ii. Width of the box (b_w)

iii. Height of the box (b_h)

YOLO predicts the corresponding number for the predicted class '*c*' and the probability of the prediction (P_c). YOLO is designed to be fast and efficient, allowing for real-time object detection.

Several new versions of the same model have been proposed since the initial release of YOLO in 2015, each building on and improving its predecessor. Here's a timeline showcasing YOLO's development in recent years [17].

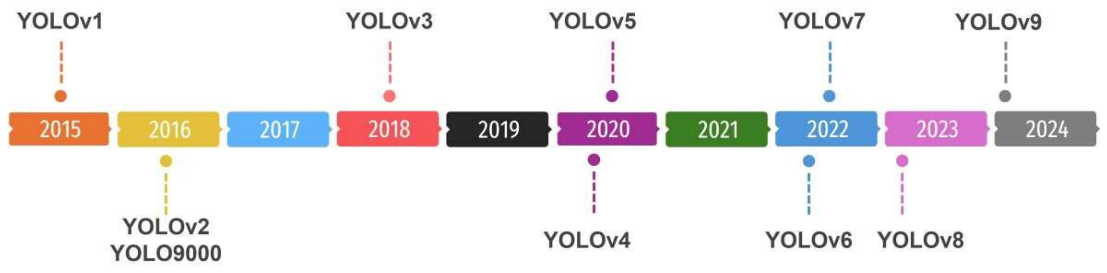


Figure 3.6: Evolution of YOLO Algorithm [17]

Yolo Architecture

The YOLO algorithm takes an image as input and then uses a simple deep convolutional neural network to detect objects in the image. The architecture of the CNN model that forms the backbone of YOLO is shown below.

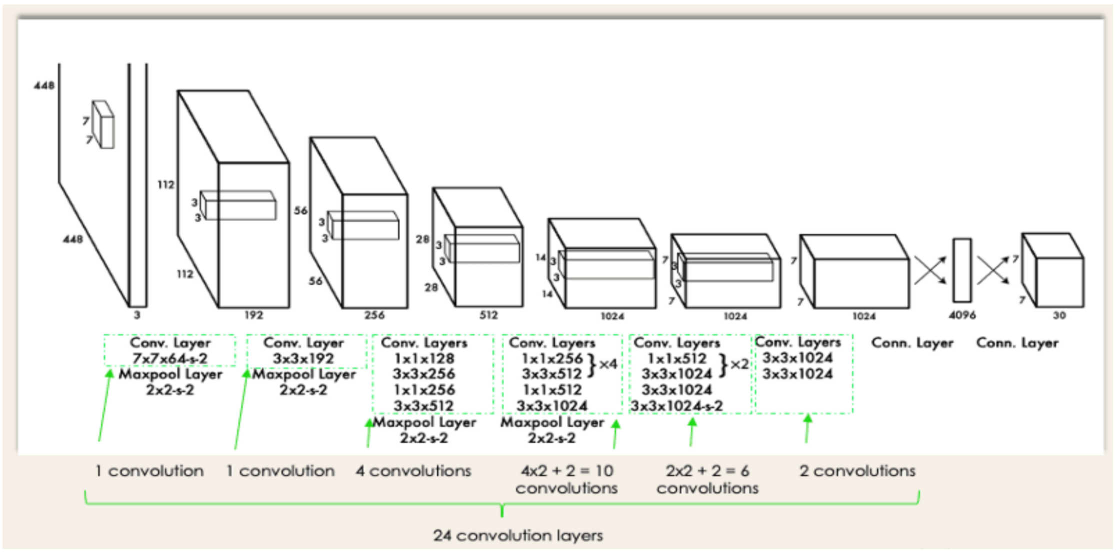


Figure 3.7: Architecture of YOLO Algorithm [18]

The first 20 convolution layers of the model are pre-trained using ImageNet by plugging in a temporary average pooling and fully connected layer. Then, this pre-trained model is converted to perform detection since previous research showcased that adding convolution and connected layers to a pre-trained network improves performance. YOLO’s final fully connected layer predicts both class probabilities and bounding box coordinates [18].

YOLO divides an input image into an S × S grid. If an object’s center falls within a grid cell, that cell is responsible for detecting the object. Each grid cell predicts ‘B’ bounding boxes and confidence scores, reflecting the model's confidence in the box containing an object and its accuracy. The bounding box attributes are predicted using a single regression module, forming a vector Y = [pc, bx, by, bh, bw, c1, c2], where pc is the probability of an object being in the grid, bx and by are the center coordinates, bh and bw are the height and width, and c1, c2 are class probabilities.

To address multiple bounding boxes for a single object, YOLO uses the Intersection over Union (IoU) metric and Non-Max Suppression (NMS) to retain only the most relevant boxes, reducing noise [19].

The formula of IoU is:

$$IoU = \frac{Area\ of\ the\ intersection\ between\ B1\ and\ B2}{Area\ of\ the\ union\ between\ B1\ and\ B2} \quad \ldots\ldots\ldots\ldots\ldots\ldots \text{Equation(3.1)}$$

where B_1 and B_2 are two bounding boxes.

About Yolov9

On February 21st, 2024, Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao released the paper "YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information," introducing the YOLOv9 model architecture. YOLOv9 outperforms earlier YOLO models (YOLOv8, YOLOv7, YOLOv5) in mean Average Precision (mAP) on the MS COCO dataset. Its key contributions include superior performance and efficiency, utilizing 41% fewer parameters and 21% less computational power, thanks to Programmable Gradient Information (PGIs) and reversible functions. YOLOv9 is highly accurate and supports object detection and semantic segmentation. The source code is available for public use, allowing for training custom YOLOv9 models. [20].

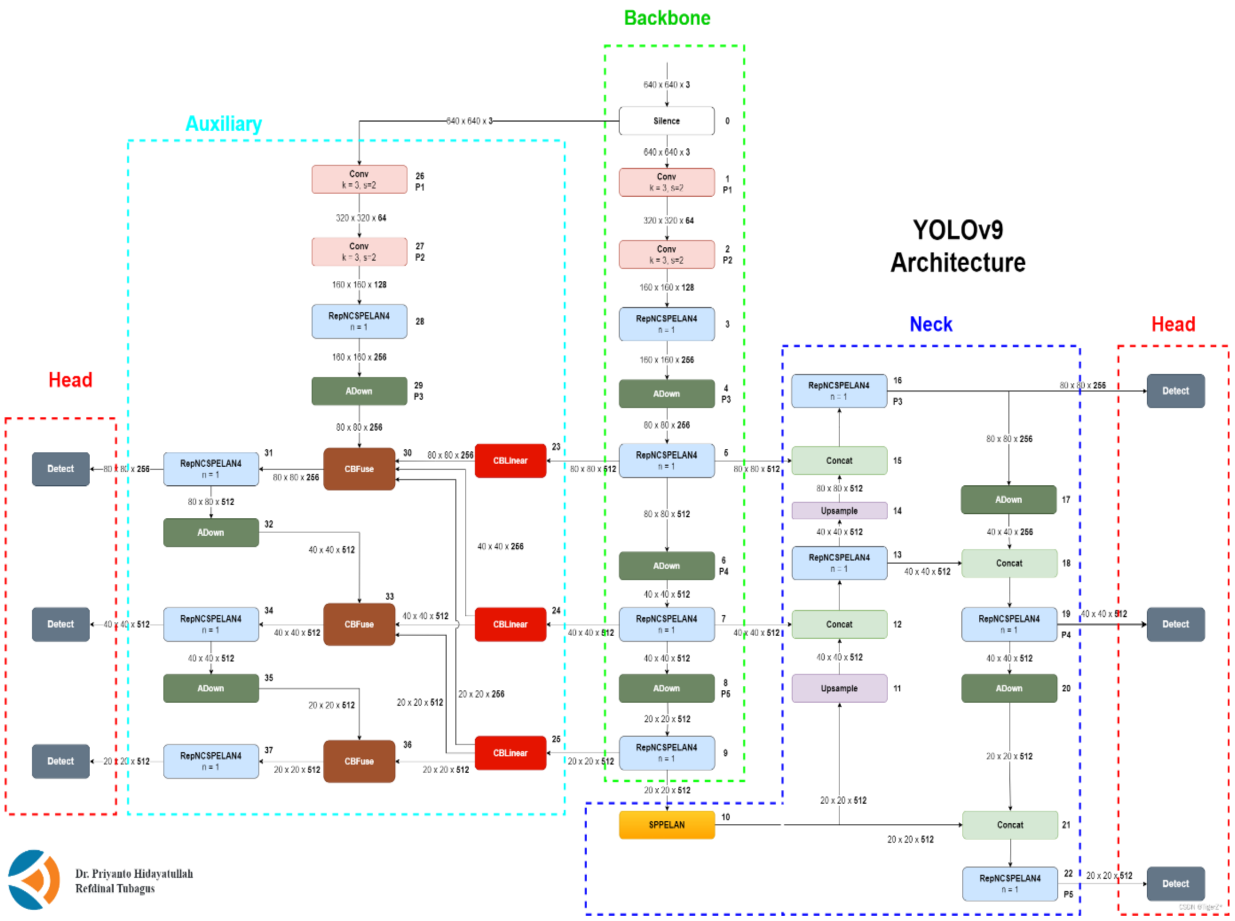


Figure 3.8: YOLOv9 Architecture [20]

The model needs to go through following steps to be ready for the detection purpose:

3.1.7. Model Evaluation

After training the YOLOv9 model for license plate recognition, various measures were employed to evaluate its performance comprehensively. These included training metrics and validation metrics to assess learning progress. Additionally, a confusion matrix was used to visualize the performance across different classes. Precision-Recall (PR) curves, Precision-Confidence (P) curves, and Recall-Confidence (R) curves were generated to understand the trade-offs between precision, recall, and confidence thresholds. Visualization of labeled data and pair plots provided insights into the data distribution and model predictions.

Model Performance Analysis:

The model's performance was evaluated over 50 epochs using a series of training metrics and validation metrics. The analysis of these metrics provides insights into the model's learning process and its ability to generalize to unseen data.

The key findings are summarized below:

Training Metrics

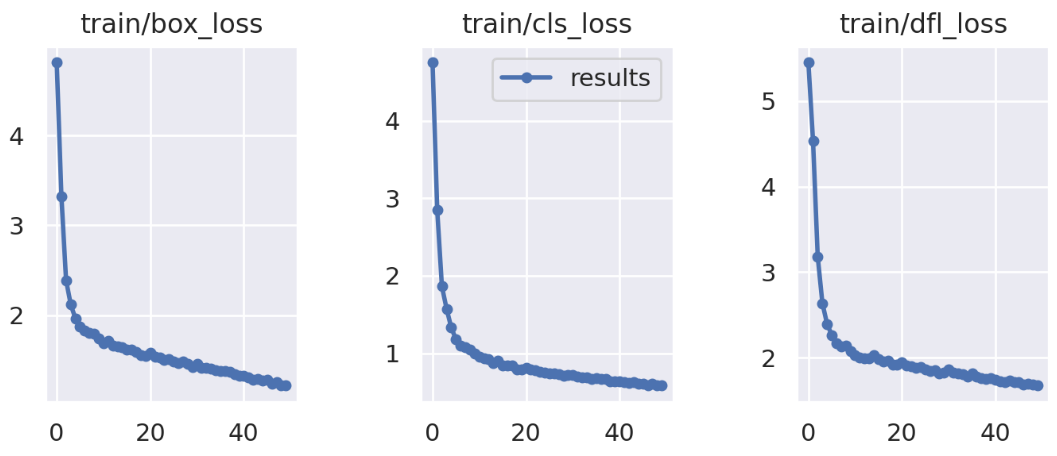


Figure 3.9: Model Performance Analysis

1. train/box_loss:
    1. Epoch 0: 4.8124
    2. Epoch 49: 1.2199
    3. Observation: The box loss decreases steadily from 4.8124 to 1.2199, indicating consistent improvement in the model's ability to predict bounding boxes accurately.
2. train/cls_loss:
    - Epoch 0: 4.7545
    - Epoch 49: 0.58378
    - Observation: The classification loss drops significantly from 4.7545 to 0.58378, suggesting the model is becoming better at classifying detected objects correctly.
3. train/dfl_loss:
    - Epoch 0: 5.4523

- Epoch 49: 1.6743
- Observation: The distribution focal loss decreases from 5.4523 to 1.6743, indicating overall improvement in the model's performance.

Validation Metrics

4. val/box_loss:
    - Epoch 1: 1.234
    - Epoch 49: 1.1346
    - Observation: The validation box loss shows a slight decrease from 1.234 to 1.1346, indicating some improvement in the model's ability to predict bounding boxes accurately during validation.

5. val/cls_loss:
    - Epoch 1: 1.1735
    - Epoch 49: 0.42023
    - Observation: The validation classification loss drops significantly from 1.1735 to 0.42023, suggesting the model is becoming better at classifying detected objects correctly during validation.

6. val/dfl_loss:
    - Epoch 1: 1.3926
    - Epoch 49: 1.3761
    - Observation: The validation distribution focal loss slightly decreases from 1.3926 to 1.3761, indicating a marginal improvement in the model's overall performance during validation.

Precision and Recall

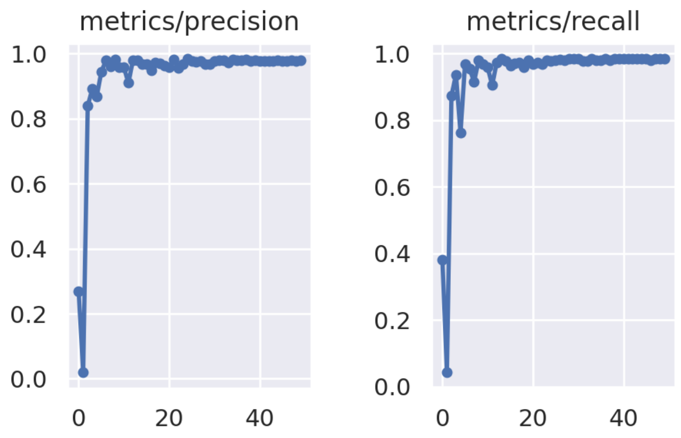


Figure 30: Precision And Recall

1. metrics/precision:
   - Epoch 0: 0.26895
   - Epoch 49: 0.97836
   - Observation: Precision starts at 0.26895 and increases to 0.97836, indicating the model maintains a high true positive rate and low false positive rate.
2. metrics/recall:
   - Epoch 0: 0.38095
   - Epoch 49: 0.98413
   - Observation: Recall starts at 0.38095 and increases rapidly to 0.98413, indicating the model's improved ability to detect all relevant objects.

Mean Average Precision (mAP)

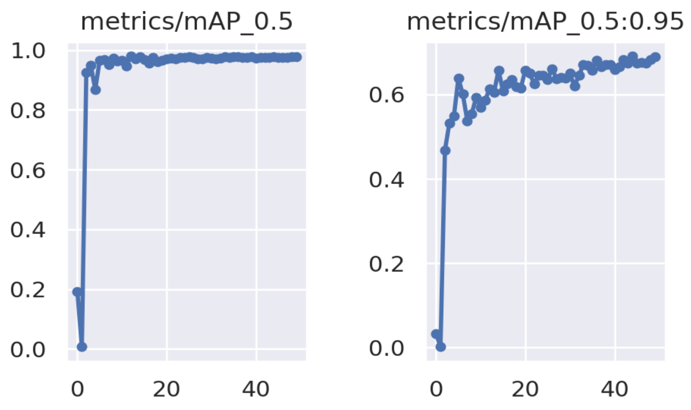


Figure 3.11: Mean Average Precision

1. metrics/mAP_0.5:
    - Epoch 0: 0.19138
    - Epoch 49: 0.97672
    - Observation: mAP at IoU threshold 0.5 increases from 0.19138 to 0.97672, indicating strong performance in terms of precision and recall at this threshold.
2. metrics/mAP_0.5:0.95:
    - Epoch 0: 0.032446
    - Epoch 49: 0.69018
    - Observation: mAP over a range of IoU thresholds (0.5 to 0.95) starts at 0.032446 and increases to 0.69018, indicating the model's improving performance over stricter IoU thresholds.

Learning Rate

The learning rates x/lr0x/lr0x/lr0, x/lr1x/lr1x/lr1, and x/lr2x/lr2x/lr2 started at approximately 0.070242, 0.0033065, and 0.0033065, respectively, at epoch 0. By epoch 49, the learning rate decreased to 0.000496. This progressive decrease in learning rates is characteristic of a learning rate schedule designed to allow for finer adjustments and improved convergence as training advances.

Confusion Matrix

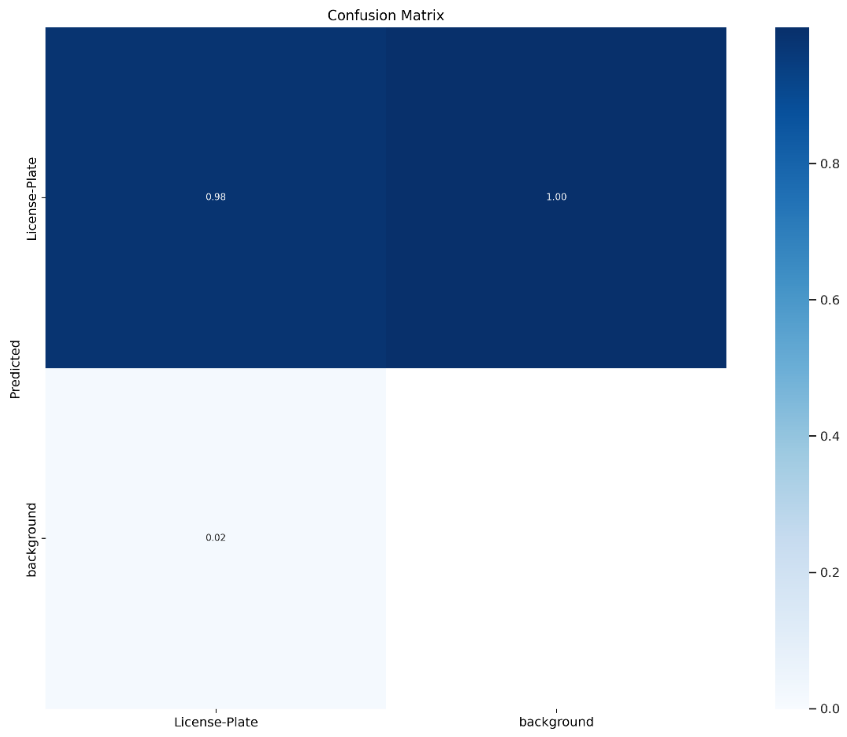


Figure 3.12: Confusion Matrix

Confusion Matrix Summary:

- True Positives (License-Plate predicted as License-Plate): 0.98
- False Negatives (License-Plate predicted as background): 0.02
- False Positives (background predicted as License-Plate): 1.00
- True Negatives (background predicted as background): 0.00

Interpretation:

- True Positives (TP): The model correctly identified license plates 98% of the time.
- False Negatives (FN): 2% of actual license plates were incorrectly classified as background.
- False Positives (FP): The model has a high rate (100%) of classifying background as license plates, which suggests a significant issue.
- True Negatives (TN): There were no instances where the background was correctly identified as background.

The model demonstrates a robust learning process and effective performance improvement across all evaluated metrics.

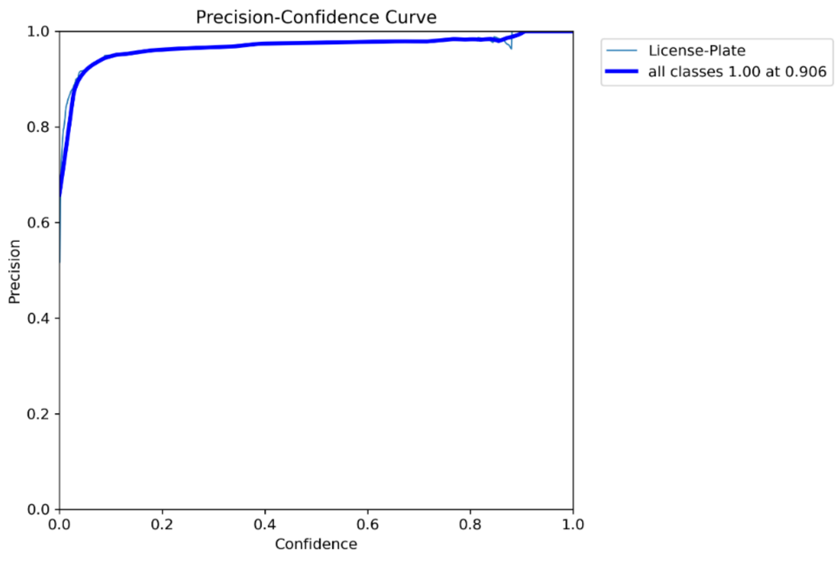


Figure 3.13: Precision-Confidence Curve

The Precision-Confidence Curve shows that precision starts around 0.6 at low confidence thresholds and rapidly improves to near 1.0 as confidence increases, stabilizing above 0.9 from a threshold of 0.5 onward. This indicates the model's high accuracy in identifying license plates, especially at higher confidence levels, making it highly reliable for applications requiring precise predictions.

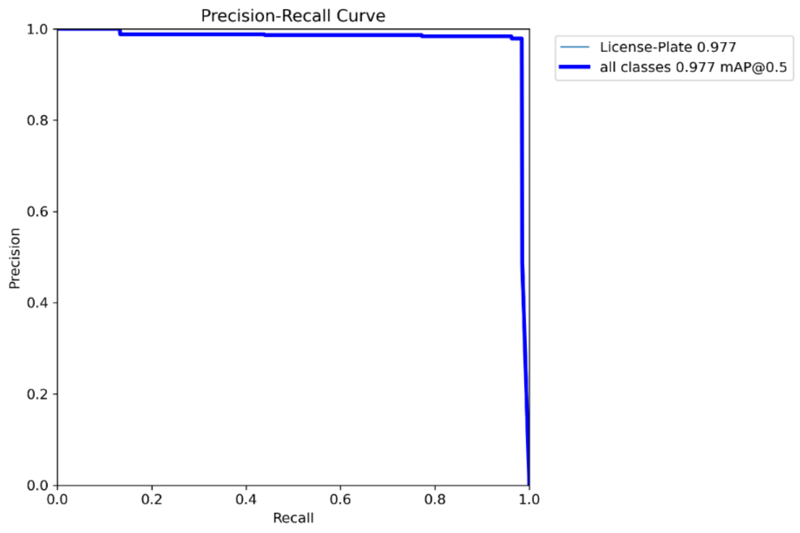


Figure 3.14: Precision-Recall Curve

The Precision-Recall (PR) curve shows excellent model performance, with a near-rectangular shape indicating high precision across almost all recall values. The mean Average Precision (mAP) at 0.5 is 0.977, demonstrating exceptional accuracy in license plate detection. This high precision is consistent for the "License-Plate" class, confirming the model's effectiveness in balancing precision and recall.

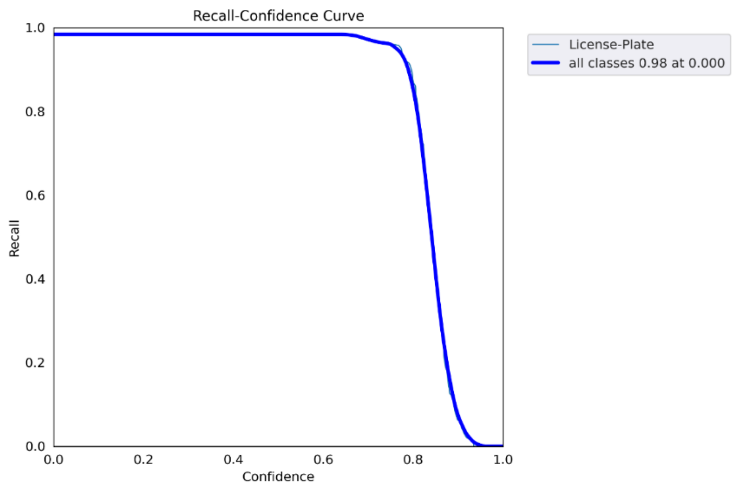


Figure 3.15: Recall-Confidence Curve

The Recall-Confidence curve shows that the model maintains a high recall close to 1.0 for confidence levels up to around 0.8, after which recall sharply declines. The overall recall for all classes is 0.98 at a confidence threshold of 0. This indicates the model is highly effective at detecting license plates at lower confidence levels, with recall decreasing as confidence thresholds increase.

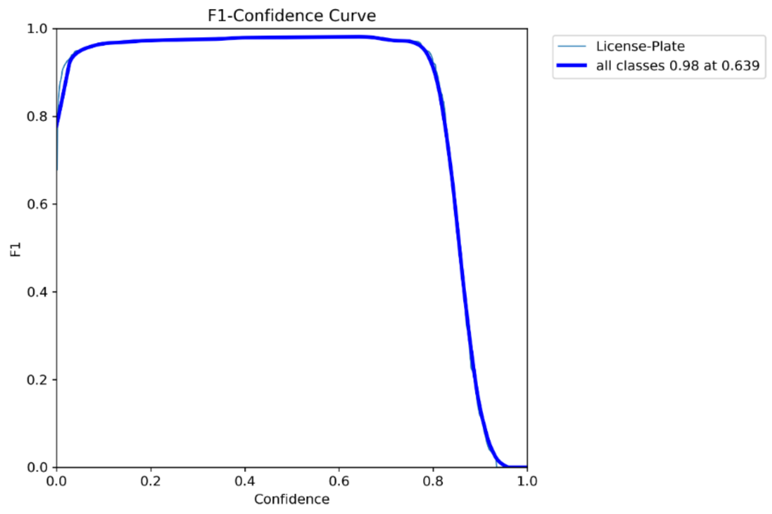


Figure 3.16: F1 Confidence Score

The F1-Confidence curve shows that the model achieves a high F1 score close to 1.0 for confidence levels up to about 0.8, after which the F1 score drops sharply. This indicates that at higher confidence thresholds, the model becomes more conservative in its predictions, missing more actual positives, thereby reducing the overall F1 score. The overall F1 score for all classes is 0.98 at a confidence threshold of 0.639, indicating strong performance across various confidence levels.

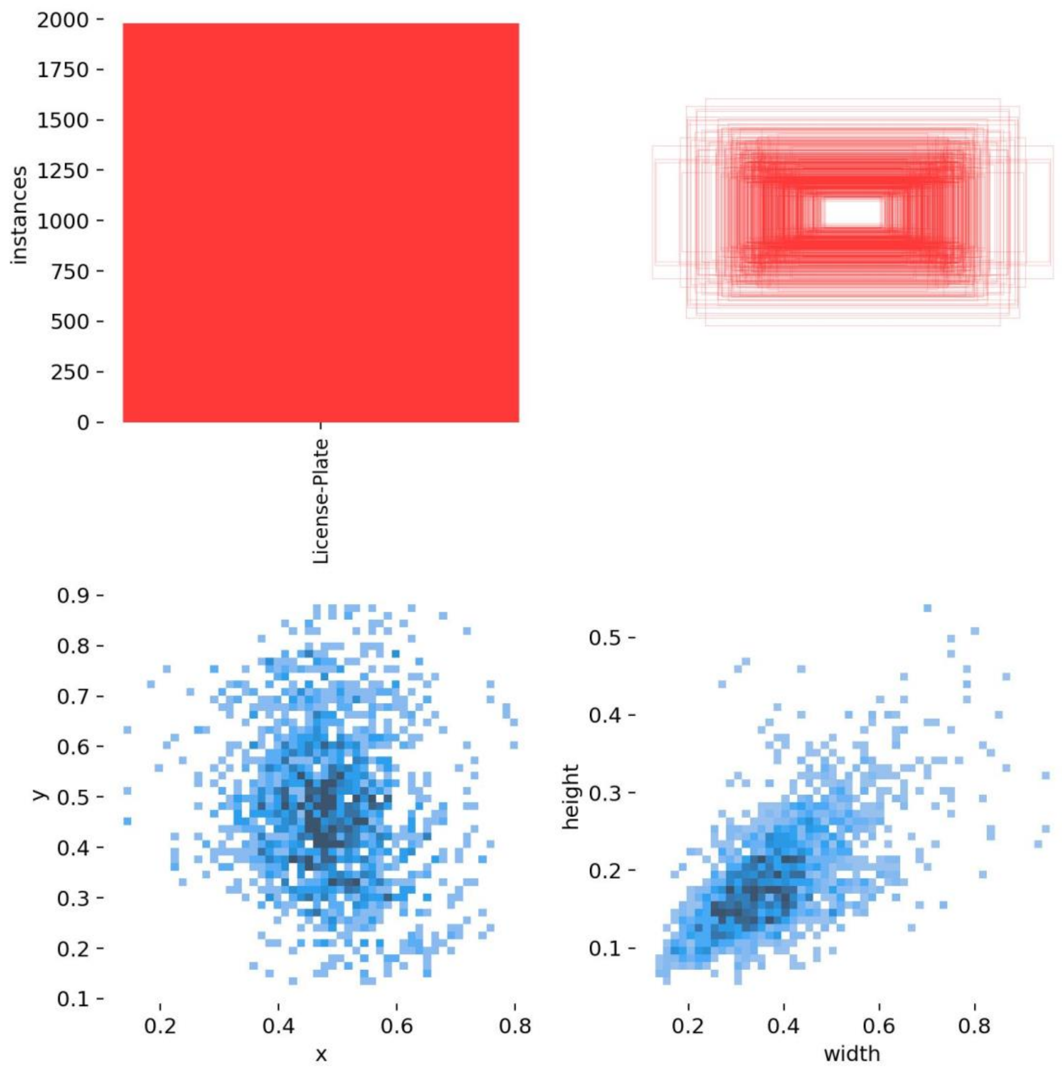


Figure 3.17: Visualization of labeled data

The figure contains 4 parts:

1. Top Left Plot (Red Square): This plot indicates a count of instances for a single class labeled "License-Plate," showing 2000 instances, represented by the large solid red square.
2. Top Right Plot (Red Rectangles): This plot appears to show the distribution of bounding boxes for detected license plates. The overlapping red rectangles suggest the locations and sizes of detected plates, with more overlap in the center indicating a higher density of detections.
3. Bottom Left Plot (Blue Scatter Plot): This scatter plot shows the distribution of x and y coordinates of the bounding boxes' centers. The concentration around the center indicates that most bounding boxes are located near the image center.
4. Bottom Right Plot (Blue Scatter Plot): This scatter plot depicts the width and height distribution of the bounding boxes. The clustering around a specific region suggests common dimensions for detected license plates, with width mostly around 0.3-0.5 and height around 0.1-0.3.

This image below is a pair plot, which is often used to visualize the relationships between multiple variables (x, y, height, width).

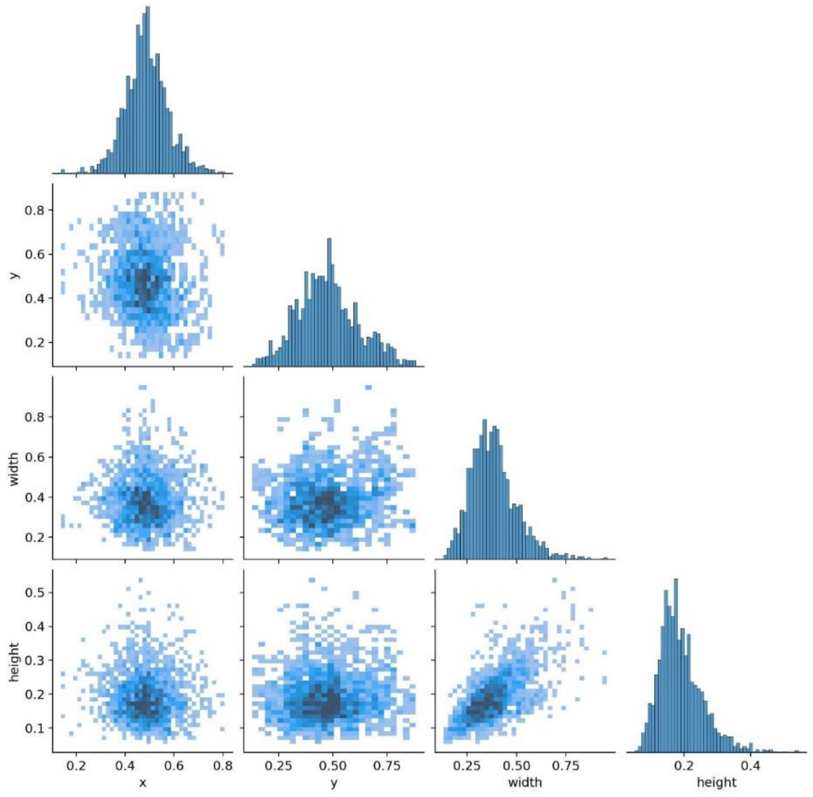


Figure 3.18: Pair Plot

Training Batches

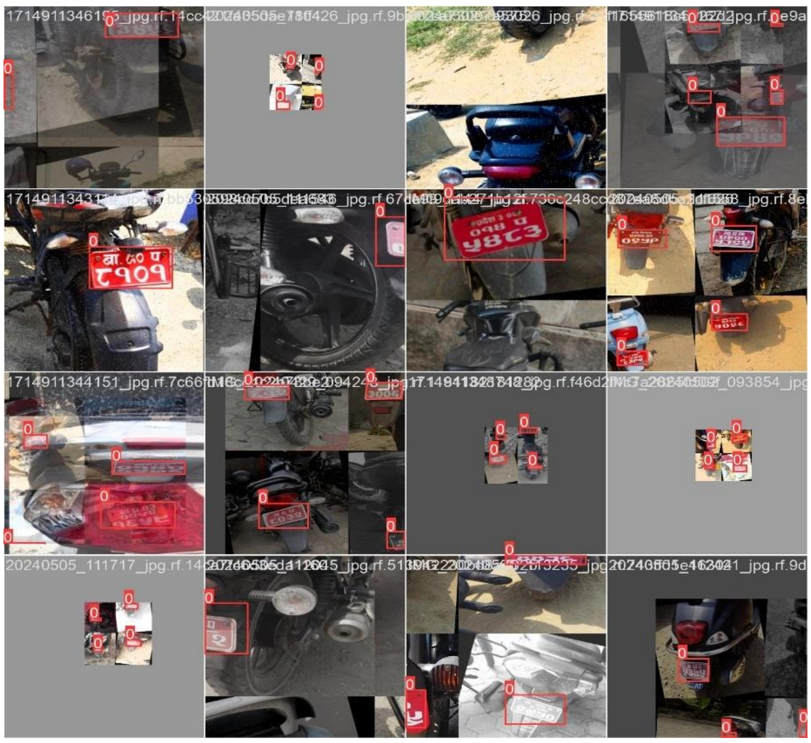


Figure 3.19: Training Batches

Validation batch labels:

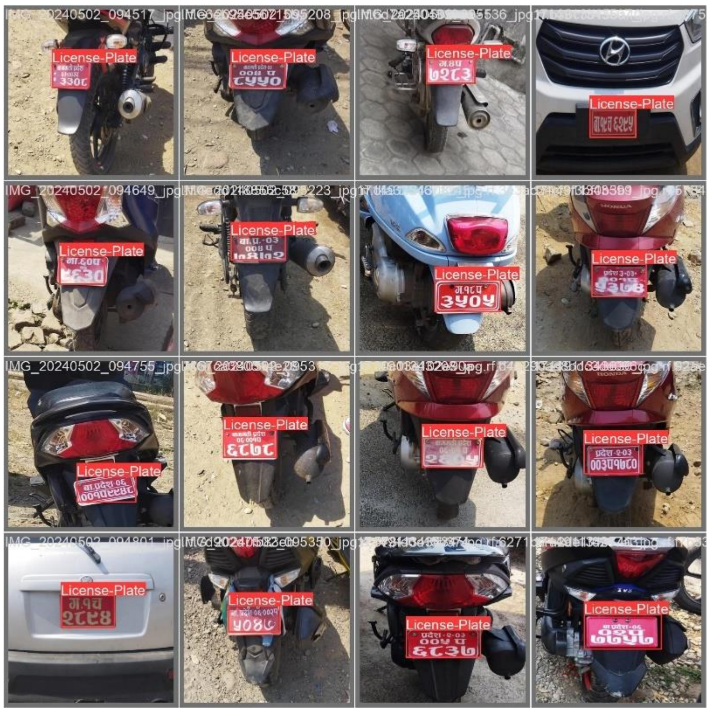


Figure 3.20: Validation Batch Label

Validation batch predict

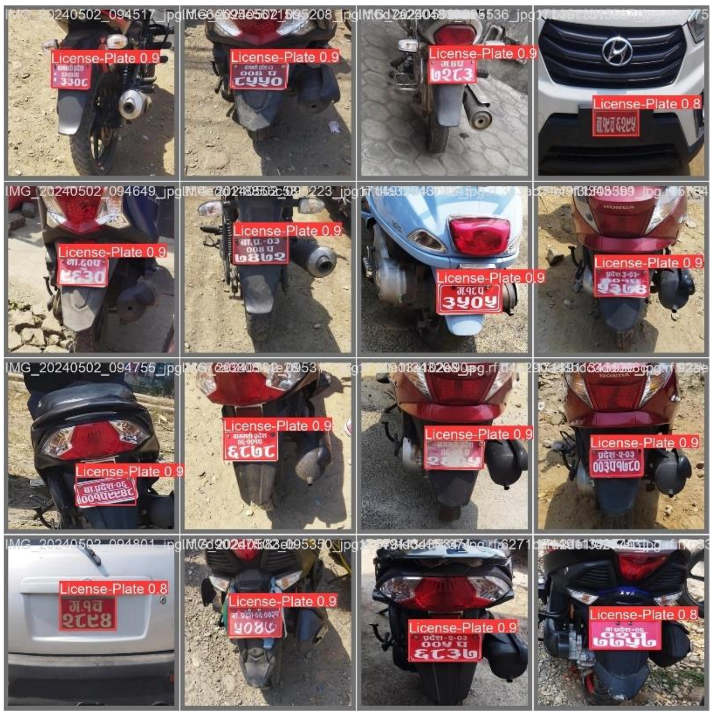


Figure 3.21: Validation Batch Predict

3.1.8. Optical Character Recognition (OCR)

OCR (Optical Character Recognition) is crucial for reading characters from license plates. In this project, EasyOCR was used due to its support for multiple languages, including Nepali, ensuring accurate recognition of Nepali license plates. EasyOCR employs deep learning to detect and recognize text from images [21].

After the ANPR system locates the license plate, EasyOCR extracts the alphanumeric characters from the detected plate, handling variations in font, size, and quality effectively.

3.2. Parking Management Software Development

The web-based platform was developed by using frontend and backend technologies which are described below:

3.2.1. Html

HTML, which stands for Hypertext Markup Language, is the standard markup language used to create and design documents on the World Wide Web. It forms the backbone of web content by providing a structured way to describe the elements on a web page, such as text, images, links, forms, and multimedia [22].

3.2.2. CSS

CSS, which stands for Cascading Style Sheets, is a style sheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG or XHTML). So, CSS allows us to control the layout, formatting, and appearance of web pages [23].

3.2.3. Django

Django is a high-level web framework designed for rapid web development, prioritizing clean and efficient code. It is a Python based web-platform. Django helps in building applications quickly. It uses the Model-View-Template (MVT) framework: the model handles the database, the view manages the logic, and the template deals with user interactions. [24].

Using HTML, CSS, and Django, a web-based platform was developed for the project. This platform integrates a pre-trained YOLOv9 model with a Nepali dataset and employs EasyOCR for character extraction from license plates. HTML and CSS are utilized for structuring and styling the web content, while Django provides the backend framework for efficient development and application management.

3.3. Application Interface and User Experience

3.3.1. Home Page

The central hub of the application where users can upload images for parking space detection and view the results.

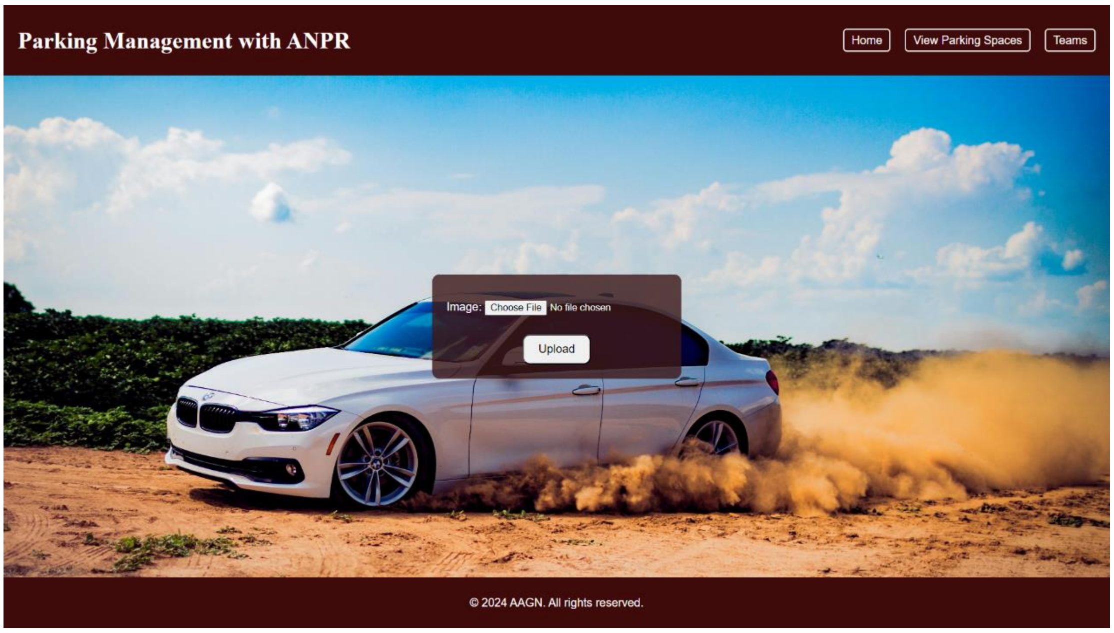
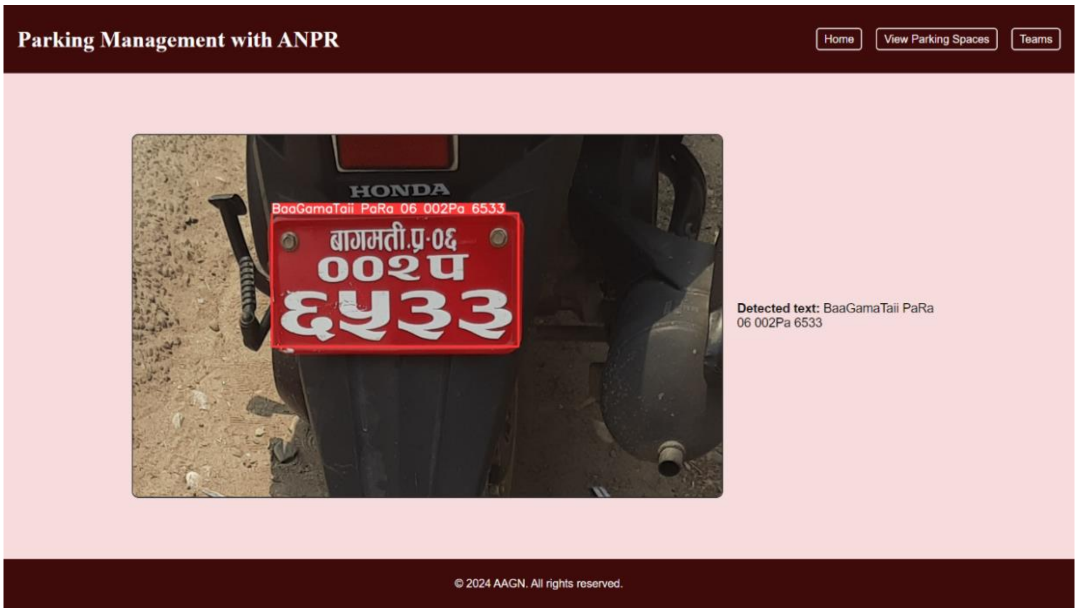


Figure 3.22: Home Page of User Interface

When a user uploads an image through the home page, the license plate is detected, and OCR recognizes the characters from the image, which are then displayed on the page.



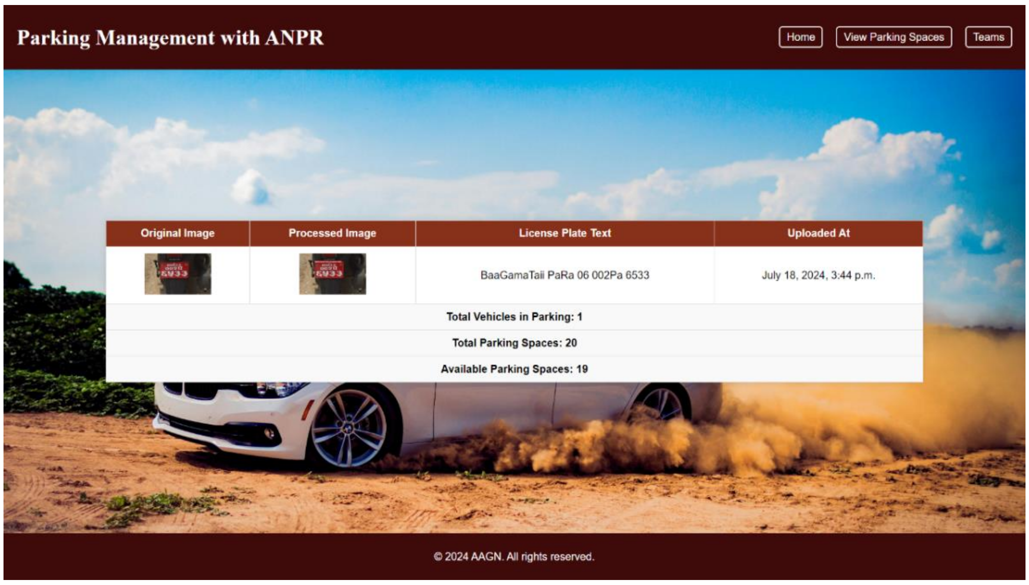
Detected text: BaaGamaTaii PaRa 06 002Pa 6533

© 2024 AAGN. All rights reserved.

Figure 3.23: Plate Detection

3.3.2. View Parking Spaces Page:

The "Viewing Parking Space" page displays the original image, the detected license plate image, the recognized license plate text, the upload date and time, the total number of vehicles, the total parking spaces, and the available parking spaces. This information helps users determine the availability of parking spaces in advance, allowing them to plan their parking accordingly.



Original Image	Processed Image	License Plate Text	Uploaded At
		BaaGamaTaii PaRa 06 002Pa 6533	July 18, 2024, 3:44 p.m.

Total Vehicles in Parking: 1

Total Parking Spaces: 20

Available Parking Spaces: 19

© 2024 AAGN. All rights reserved.

Figure 3.24: View Parking Spaces Page

3.4. Database Integration

A database was integrated to store information about recognized license plates, license plate number, parking availability, and other relevant data.

3.5. Testing and Deployment

The entire system was thoroughly tested to ensure seamless integration and functionality. Once testing was successful, the YOLOv9 model for license plate detection and EasyOCR for character extraction was deployed in a local machine.

Chapter 4: Result and Discussion

The Automatic Number Plate Recognition (ANPR) system involves capturing 942 vehicle images under diverse conditions, followed by cropping and annotating these images with Roboflow to train the YOLOv9 model. The dataset was divided into training (70%), validation (20%), and testing (10%) sets, with preprocessing including resizing to 640x640 pixels and applying various augmentations to enhance model generalization. The YOLOv9 model, pre-trained on COCO, was fine-tuned for plate localization, and EasyOCR was used for character recognition.

The YOLOv9 model for license plate recognition was evaluated over 50 epochs, showing significant improvements in performance. Training box loss decreased from 4.8124 to 1.2199, training classification loss dropped from 4.7545 to 0.58378, and training distribution focal loss improved from 5.4523 to 1.6743. Validation box loss decreased from 1.234 to 1.1346, validation classification loss dropped from 1.1735 to 0.42023, and validation distribution focal loss improved from 1.3926 to 1.3761. Precision increased from 0.26895 to 0.97836, while recall rose from 0.38095 to 0.98413. Mean Average Precision (mAP) at IoU 0.5 improved from 0.19138 to 0.97672, and mAP over IoU thresholds 0.5 to 0.95 increased from 0.032446 to 0.69018. The learning rate decreased from 0.070242 to 0.000496, indicating effective convergence. Despite a high true positive rate of 98%, the model had a significant false positive rate, with background frequently misclassified as license plates. Precision-Confidence and Precision-Recall curves showed high precision and recall, while the F1-Confidence curve indicated strong performance at lower confidence thresholds. Visualization of labeled data revealed common bounding box dimensions, with most boxes clustered near the image center. Overall, while the model demonstrated robust performance, addressing the high false positive rate is crucial for future improvements.

After obtaining weight it is integrated in web bases platform. A web-based platform was created with HTML, CSS, and Django, allowing users to upload images for parking space detection and viewing real-time parking information. A database was integrated to store and manage recognized license plates and parking availability. After thorough testing, the system was deployed locally for practical use.

Chapter 5: Conclusion and Future Enhancement

The project successfully developed an Automatic Number Plate Recognition (ANPR) system integrated with a smart parking management application. Utilizing the YOLOv9 model, the system efficiently recognized license plates, while EasyOCR provided accurate character recognition for Nepali language plates. The web application offered real-time parking space information, enhancing user convenience. With a robust database for managing license plate data, the system was thoroughly tested and successfully deployed, presenting a practical solution for automated vehicle identification and parking management.

Future enhancements for the project include:

1. Improve OCR Accuracy: Integrate advanced OCR models and preprocessing techniques to enhance character recognition.
2. Expand Training: Increase training epochs to improve model performance.
3. Real-Time Processing: Implement support for real-time video and webcam integration.
4. Cloud Deployment: Deploy the system on cloud platforms to ensure better scalability and accessibility.
5. Confusion Matrix Refinement: Address the high false positive rate and work on improving classification accuracy.

References

[1] R. Tater, P. Nagrath, J. Mishra, V. H. de Albuquerque, and J. W. Menezes, “Smart parking system using Yolov3 Deep Learning Model,” *Proceedings of Data Analytics and Management*, pp. 387–398, 2023. doi:10.1007/978-981-99-6550-2_30

[2] S. Gopikrishnan, A. K. Madduru, K. Karamsetty, and D. R. Ravuri, “Smart parking system with automated vehicle log using Haar Cascade Classifier ANPR,” *IFIP Advances in Information and Communication Technology*, pp. 266–286, 2023. doi:10.1007/978-3-031-38296-3_21

[3] A. Menon and B. Omman, “Detection and recognition of multiple license plate from still images,” *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, 2018. doi:10.1109/iccsdet.2018.8821138

[4] N. Darapaneni *et al.*, “Computer Vision based license plate detection for Automated Vehicle Parking Management System,” *2020 11th IEEE Annual Ubiquitous Computing, Electronics &amp; Mobile Communication Conference (UEMCON)*, 2020. doi:10.1109/uemcon51285.2020.9298091

[5] P. R. Dawadi, B. K. Bal, and M. Pokharel, “Devanagari license plate detection, classification and recognition,” *Proceedings of the First International Conference on Advances in Computer Vision and Artificial Intelligence Technologies (ACVAIT 2022)*, pp. 304–318, 2023. doi:10.2991/978-94-6463-196-8_24

[6] B. Setiyono, D. A. Amini, and D. R. Sulistyaningrum, “Number Plate recognition on vehicle using YOLO - Darknet,” *Journal of Physics: Conference Series*, vol. 1821, no. 1, p. 012049, 2021. doi:10.1088/1742-6596/1821/1/012049

[7] M. M. Abdellatif, N. H. Elshabasy, A. E. Elashmawy, and M. AbdelRaheem, “A low cost IOT-based Arabic license plate recognition model for Smart Parking Systems,” *Ain Shams Engineering Journal*, vol. 14, no. 6, p. 102178, 2023. doi:10.1016/j.asej.2023.102178

[8] M. S. Farag, M. M. Mohie El Din, and H. A. Elshenbary, “Parking entrance control using license plate detection and recognition,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 15, no. 1, p. 476, 2019. doi:10.11591/ijeecs.v15.i1.pp476-483

[9] H. Moussaoui, N. E. AKKAD, and M. BENSLIMANE, *Arabic and Latin license plate detection and recognition based on Yolov7 and image processing methods*, 2023. doi:10.21203/rs.3.rs-3195386/v1

[10] S. H. Fahim, A. A. Shafi Rasel, A. R. Sarker, T. Chowdhury, and E. L. Rimban, *Utilizing yolo V8 for automated license plate detection of vehicles in Bangladesh: An approach for vehicle identification*, 2024. doi:10.33774/coe-2024-gxw4j

[11] V. Gnanaprakash, N. Kanthimathi, and N. Saranya, “Automatic number plate recognition using Deep Learning,” *IOP Conference Series: Materials Science and*

Engineering, vol. 1084, no. 1, p. 012027, 2021. doi:10.1088/1757-899x/1084/1/012027

[12] R. Laroca *et al.*, “An efficient and layout-Independent Automatic License Plate Recognition System based on the Yolo Detector,” *IET Intelligent Transport Systems*, vol. 15, no. 4, pp. 483–503, 2021. doi:10.1049/itr2.12030

[13] S. Kaur, “An automatic number plate recognition system under image processing,” *International Journal of Intelligent Systems and Applications*, vol. 8, no. 3, pp. 14–25, 2016. doi:10.5815/ijisa.2016.03.02

[14] N. A. Jalil, A. S. Basari, S. Salam, N. K. Ibrahim, and M. A. Norasikin, “The utilization of template matching method for license plate recognition: A case study in Malaysia,” *Lecture Notes in Electrical Engineering*, pp. 1081–1090, 2014. doi:10.1007/978-3-319-07674-4_100

[15] W. W. Keong and V. Iranmanesh, “Malaysian automatic number plate recognition system using Pearson correlation,” *2016 IEEE Symposium on Computer Applications &amp; Industrial Electronics (ISCAIE)*, 2016. doi:10.1109/iscaie.2016.7575034

[16] D. Neupane *et al.*, “Shine: A Deep Learning-based accessible parking management system,” *Expert Systems with Applications*, vol. 238, p. 122205, 2024. doi:10.1016/j.eswa.2023.122205

[17] W. by: E. Zvornicanin, “What is Yolo Algorithm?,” Baeldung on Computer Science, https://www.baeldung.com/cs/yolo-algorithm (accessed Jan. 22, 2024).

[18] “Yolo algorithm for object detection explained [+examples],” YOLO Algorithm for Object Detection Explained [+Examples], https://www.v7labs.com/blog/yolo-object-detection (accessed Jan. 22, 2024).

[19] Z. Keita, “Yolo Object Detection explained: A beginner’s guide,” DataCamp, https://www.datacamp.com/blog/yolo-object-detection-explained (accessed Jan. 22, 2024).

[20] “Yolov9 object detection model: What is, how to use,” Roboflow, https://roboflow.com/model/yolov9 (accessed Jul. 19, 2024).

[21] M. Simon, B. Shibwabo, and K. Mutua, “An automatic number plate recognition system for car park management,” *International Journal of Computer Applications*, vol. 175, no. 7, pp. 36–42, 2017. doi:10.5120/ijca2017915608

[22] “What is HTML (hypertext markup language)?,” Computer Hope, https://www.computerhope.com/jargon/h/html.htm (accessed Jan. 19, 2024).

[23] CSS tutorial, https://www.w3schools.com/css/ (accessed Jan. 19, 2024).

[24] B. Nithya Ramesh, A. R. Amballi, and V. Mahanta, “DJANGO THE PYTHON WEB FRAMEWORK,” International Journal of Computer Science and Information Technology Research, vol. 6, no. 2, pp. 59-63, Apr. - Jun. 2018. Available:https://www.researchpublish.com/upload/book/DJANGO%20THE%20PYTHON%20WEB-5585.pdf. [Accessed: Jul. 19, 2024].

Appendix

The code and resources for the project are available at the following GitHub repository:
GitHub Repository link: https://github.com/NiShApOkHaReL/myproject
Collected Dataset link: https://www.kaggle.com/datasets/nishapokharel/nepali-vehicles-number-plate-dataset