Optimizing ANPR Performance Through Unique Dataset Selection & Model Testing.

RASHED HAIDER

Table of Contents

Acknowledgement:

This thesis is dedicated in loving memory of my father, who departed from this world in April 2023 at the age of 85. My journey to become a software engineer was fuelled by a desire to make him proud, a sentiment I deeply wish I could have shared with him in person. Though he is not here to witness this accomplishment, I believe he would have been a proud reader of this work. His unwavering faith in my abilities and aspirations continues to inspire and guide me.

Abstract

AI revolution significantly shaped by advancements in computing infrastructure, notably through the abundance of cloud computing and the exponential increase in data availability. Deep Learning: subset of machine learning become more mainstream only since 2012. This initiative is to set the stepping stone towards computer vision challenge, development of an Automatic Number Plate Recognition (ANPR) software. Spanning from data acquisition to model training and eventually deployment in a local environment. This study demonstrates the software development lifecycle, adhering to resonant design principles. The ANPR model achieved a creditable precision rate of 87%, showcasing its efficacy. Focused on object detection and number plate recognition, the local deployment was meticulously constructed. Despite initial aspirations to incorporate real-time object tracking, this feature was ultimately omitted due to the computational demands of Graphical Processor Unit (GPU) resources. This project stands as a microcosm of an integrated system, harmoniously blending various technologies, platforms, and pre-trained models. Furthermore, it includes a comprehensive review of contemporary studies, further enriching the academic value of this work. The web application, designed with future scalability in mind, lays the groundwork for the development of a robust and comprehensive ANPR system.

1.0 Introduction:

1.1 Background and Inspiration:

In the dawning ages of human history, knowledge was engraved onto cave walls and moulded from clay tablets, primitive yet profound marks of mankind's earliest attempts at recording their understanding of the world. As civilizations advanced, so did the mediums of expression. Handwriting on parchment transformed individual insights into shared wisdom, further democratized by Gutenberg's printing press. Yet, the real seismic shift in knowledge dissemination came with the digital revolution, as computers and the internet turned information into an ever-flowing global stream. Internet has bridged the communication and data flow in the last two decades that in effect create a vast ocean of bi product called raw data in numerous formats. The amount of data surges extensively due to massive digitization as well as availability of the smart phone across the globe.

1.2 Data driven programming:

The overflow of digital data presented a new challenge: How to untangle and make sense of it all? Under the weight of vastness data traditional analytical methods crumbled. Thus, appeared machine learning as the compass to navigate this data-driven era. Deep learning, a subset of machine learning, harnesses the power of algorithms to interpret patterns in data, paving the way for advancements in various fields, including computer vision. Fast forward, year 2022 AI has taken the world to storm by introducing generative AI powered by large language model to the mass public. Application of deep learning model seemingly become the ultimate choice for its high accuracy and performance. (Kim, Jeon and Hyung Il Koo, 2017)

1.3 Computer vs biological vision:

Computer vision is the enterprise of building machines that can see; a computer vision is designed to surpass the capability of biological vision and extract information about the world. This field is a captivating intersection of disciplines, melding core subjects like mathematics and optics. And, when the situation demands, it draws insights from diverse areas such as neuroscience, psychology, art history, and even biology.

Vision is arguably the most potent sense, allowing both humans and animals to engage with their surroundings without direct physical touch. While the intrinsic connection between vision and the brain remains an enigma yet it effortlessly providing us with a rich, qualitative experience. Transferring this faculty of vision to machines is an intricate challenge. At its core, a machine interprets everything in binary, converting data into sequences of 0s and 1s. A computer typically "sees" through webcams or other image/video inputs. When a camera captures an image or records footage, it essentially converts the light from a 3D scene into a 2D representation. The role of computer vision is almost paradoxical — it seeks to reverse this process, translating 2D images that will comprehend of the 3D world. The biological eye is inherently biased; unable to analyse objects with meticulous precision or absolute accuracy. Such limitations underscore the importance of machine vision. In today's fast-paced world, it's inconceivable to rely on humans for tasks like monitoring every vehicle on a highway manually.

This thesis explores into computer vision, more specifically, the application of Automatic Number Plate Recognition (ANPR). The study will incorporate convolutions of optimizing deep learning models for ANPR using cutting-edge technology, and evaluating how these advancements can shape the future landscape of digital information processing for automation.

2. Literature Review:

2.1 Introduction

Literature review considered as the cornerstone of academic exploration. Standing in end of year 2023, our interaction with AI has taken centre stage. The visual perception of infants is initially limited, extending merely up to 12 inches, primarily to facilitate the recognition of their parents. However, as they mature, their visual range expands, eventually encompassing distant objects. This developmental trajectory mirrors the progression from a constrained to an extensive field of vision. Taking a step back into its early days would give an idea that it was just thin end of wadge. Back in the days AI operated much like a tool, driven by traditional programming. The main goal was to have AI not just mimic the human repetitive task but to do them faster and more accurately with precision. While we can program basic intelligence into code, it doesn't capture the full range of human smarts. Simple tasks, like sorting numbers or playing a basic game, has been taught to a machine in 90's and machine perform flawlessly. Complex tasks, like understanding speech or recognizing images, can't be easily coded. Let's take an example, teaching a machine to recognize a dog in one setting might work, but change the setting of the image, and it gets confused. This is where ML models paly its role, it can be trained to understand the refined differences between categories. Key point to remember that ML algorithms aren't the magic pill. The real acumen still lies with humans. Its human who gathers the data, decide what's is significant, and choose the best ML approach. Once set up, the ML algorithm take control by learning from the data and making connections. But its role is crucial, figuring out the patterns that connect inputs to outputs. (Ahmed Fawzy Gad, 2018).

2.2 Foundational Concepts:

This section underline on taking a step back and grasp a comprehensive overview of each concept, blending concise definitions with more detailed explanations to offer readers both breadth and depth of understanding even from a layman perspective.

2.2.1 Computer Vision:

In short understanding machines that can 'See'. Computer vision is a multidisciplinary domain within the broader field of artificial intelligence, dedicated to empowering machines with the

capability to interpret and make decisions based on visual data. Rooted by fundamentals of mathematics, computer science, and optics, computer vision seeks to solve the complexities of sight and perception. From facial recognition and medical imaging to autonomous vehicles and augmented reality computer vision is forefront of the AI revolution.

2.2.2 Deep Learning Model:

Deep learning, a subset of machine learning, employs neural networks with multiple layers often called hidden layers (thus "deep") to analyse various features of data. Inspired by the human brain's neural networks, DL has transformed the approach machines understand and make sense of vast and complex datasets and learn from it.

Training:

Training implies to the iterative process where a model learns and adapts when its exposed to a dataset. Each iteration is termed as epochs whereby model learns from its training dataset. During training, a model adjusts its internal parameters to minimize errors. This part is vital as it governs the model's power to make accurate predictions.

Testing:

Testing refers to valuating the Learned Knowledge. Testing is the phase where the trained model is evaluated using a separate set of unseen data. Testing confirms model's learning is not just confined to the training data but is effective and accurate when presented with new, unseen data.

Validation:

Ensuring Generalization in Deep Learning. Validation involves assessing a model's performance using a dataset that it hasn't encountered during the training phase. Role of validation is to defend against overfitting, to ensure model's learning is robust and applicable beyond the training dataset. It provides insights into the model's ability to generalize its learning to diverse data scenarios. (Opencv.org, 2023)

2.2.3 Convolutional Neural Networks (CNN):

At the core of image recognition and computer vision lies the dominance of Convolutional Neural Networks (CNNs) architecture. These networks, drive the advancements in deep learning algorithms. CNNs are structured with layers of nodes, surrounding an input layer,

multiple hidden layers, and an output layer. Each node in these layers is interconnected, possessing specific weights and thresholds as well as pre-set biased. When a node's output surpasses its threshold, it activates, transmitting data to the subsequent layer. This complicated design empowers computers to extract meaningful insights from visual inputs like digital images and videos or even real time media, based on that take it can take informed actions. This intricate network facilitates the system's ability to differentiate and recognize various visual elements, making CNNs forefront architecture in deep learning and computer vision. (Ibm.com, 2023)

2.2.4 ANPR:

Automating Vehicle License Plate Recognition, or Automatic Number Plate Recognition, is a dedicated application within computer vision designed to automatically detect track and read vehicle license plates. ANPR systems are pivotal in modern traffic management, enforcement, security surveillance, toll collection and many more use cases commercially. By automating the process of license plate reading, these systems improve efficiency and accuracy.

2.2.4 OpenCV:

It’s a open source initiatives welcome everyone to contribute. Initiated within Intel's corridors in 1999 by Gary Bradsky, OpenCV unveiled its first iteration in 2000. Intel's Russian software division dedicated to OpenCV made a notable milestone was in 2005 when Stanley, a vehicle equipped with OpenCV, clinched the title at the 2005 DARPA Grand Challenge. Today, OpenCV stands as a repository of numerous algorithms in Computer Vision and Machine Learning, continually evolving and growing. (Columbia.edu, 2023)

Adding another triumph, OpenCV-Python emerged as a library offering Python bindings tailored to solve computer vision challenges. A remarkable feature of OpenCV-Python is its integration with Numpy, a library popular for its optimized numerical operations, reminiscent of MATLAB's syntax. This seamless synergy ensures that all OpenCV array structures undergo conversion to and from Numpy arrays. Such interoperability also paves the way for effortless integration with other Numpy-reliant libraries, including SciPy and Matplotlib. (Dr. Dobb’s, 2013)

2.2.4 YOLOv8:

YOLOv8, an acronym for 'You Only Look Once,' denotes the eighth iteration of a contemporary object detection model. YOLOv8 gained enormous popularity and stands out for its ability to detect objects in real-time, making it invaluable in applications that require instant results, such as real-time surveillance or autonomous driving. Yolov8.com. (2023).

2.2.5 Bounding Boxes:

A bounding box is really framing objects in vision. Its a rectangular overlay that outlines the location of an object within an image. Essentially, a bounding box works as visual markers, aiding in object detection by highlighting regions of interest (ROI). They are foundational in many computer vision tasks, providing a spatial context for detected objects.

2.2.6 Object Interactions in Visual Data:

Object detection and tracking involve identifying specific objects within visual data and monitoring their movements over time. These processes are integral to numerous applications, from video surveillance and sports analytics to wildlife monitoring and traffic management.

Detection:

Identifying Objects in Frames. Object detection is the computational task of identifying specific objects within visual data. Detection algorithms scrutinize through visual data, pinpointing areas of interest, and classifying objects based on learned patterns. In short they termed as ROI (region of interest).

Tracking:

As the name suggest tracking is object trajectories. Object tracking follows the movement and trajectory of these objects over consecutive frames in a video sequence. Once objects are detected, tracking algorithms monitor their spatial progression, providing insights into object behaviours and interactions.

2.2.7 Optical Character Recognition:

Easy OCR is an optical character recognition tool capable of extracting textual content from a given images. OCR technologies, like Easy OCR, bridge the gap between visual data and textual interpretation. They play a vital role in digitizing printed materials, automating data entry, and enhancing accessibility. (Shi, Bai and Yao, 2015)

2.2.8 Datasets in Machine Learning:

Datasets are curated collections of data used to train, validate, and test in deep learning models. The quality and diversity of datasets directly influence the efficacy of the models. They provide the foundational knowledge upon which models base their predictions and decisions.

Roboflow:

Roboflow is a state-of-the-art platform that restructures preparing and annotating image datasets ready for ML. The quality of datasets is predominant in model development. Dataset is the deciding factor of the performance of the model. Roboflow aids researchers and developers by offering tools that simplify the often-tedious process of dataset curation, ensuring that data is structured and annotated accurately for optimal model training.

(Roboflow.com. 2023).

Kaggle:

The Treasure Trove of Data Science, Kaggle is a renowned platform for data science competitions and offers ocean of datasets, appropriate to computer vision tasks. As a hub for data science enthusiasts worldwide, Kaggle not only fosters a community of knowledge-sharing but also provides access to diverse datasets. These datasets, often contributed by global users, serve as invaluable resources for researchers and developers looking to benchmark and innovate in various AI domains. (Kaggle.com. 2023).

2.2.9 Python & Collaboratory:

Python language is considered as Lingua Franca of Deep Learning Implementation in the world of programming. Its versatility, bolstered by a huge libraries and frameworks, has made it the de facto choice for researchers and data scientist. Coupled with platforms like Google Collaboratory (Google.com, 2019) cloud-based Python notebook environment—it empowers

users with GPU-accelerated compute resources, crucial for the demanding nature of deep learning tasks like ANPR. (Python.org, 2023)

2.3 Related Studies:

The world has seen resurgence in the interest surrounding neural networks mimicking how human brain works, primarily driven by the impressive achievements of deep neural network models, especially Deep Convolutional Neural Networks (DCNN). Shi et al. (2015) has the study paper introduce a unique neural network model tailored for the recognition of sequence-like entities within images termed as the Convolutional Recurrent Neural Network (CRNN), this model marries the strengths of DCNN and RNN, boasting a recognition accuracy of 95.9%.

Diaz et al. (2017) study went on Automatic Number Plate Recognition (ANPR), specifically on UK number plates from high-resolution imagery. Leveraging state-of-the-art Computer Vision techniques, their research provides a comparative analysis of ANPR approaches, highlighting algorithms like Support Vector Machines (SVM) and Artificial Neural Networks (ANN). Wide-ranging dataset of vehicle images reinforced their study, with the resulting system achieving a striking accuracy rate exceeding 90%. Diaz et al. has also touched upon the standardized EU format for number plates in accordance with the Vienna Convention on Road Traffic.

Khan et al in their study "A Survey of the Recent Architectures of Deep Convolutional Neural Networks"; handle the complexities of Deep Convolutional Neural Networks (CNNs). The authors introduce a systematic classification of recent CNN architectural advancements into seven distinct categories, including spatial exploitation and channel boosting. This comprehensive survey not just finish by offering insights into the foundational components of CNNs but went one step ahead by addressing the prevailing challenges and applications in the domain.

The research "Intelligent System for Vehicles Number Plate Detection and Recognition Using Convolutional Neural Networks" in intelligent traffic monitoring systems, with a focus on the utilization of Convolutional Neural Networks (CNNs) for vehicle number plate detection and recognition, on plates written in the Bengali language. The system, developed through this

research, is branched into two primary portions: number plate detection and recognition. Initially, the vehicle's image is captured, and the number plate region is segmented from the image frame. Subsequently, a super-resolution method is applied, utilizing the convolutional layer of CNN to enhance the pixel quality of the input image. The recognition part extracts feature and classifies them using CNN technique, showcasing the novelty in developing an intelligent system that recognizes number plates even with lower resolution.

Deep Learning Model for Automatic Number/License Plate Detection and Recognition System in Campus Gates by Mustafa et al. (2023). This paper concentrates into the use of deep learning techniques, including OpenCV, YOLO, PaddleOCR, and Tesseract OCR, to develop ANPR systems. The study evaluates the effectiveness of these techniques in various environmental conditions.

Ahire, et al. (2023) discusses license plate identification on their paper Image Enhancement and Automated Number Plate Recognition by using the Grab Cut algorithm and combines the Bernsen algorithm with the Wiener filter for image noise reduction.

2.4 Comparative Analysis

Understanding the numerous approaches based on the above studies it is evident that model accuracy depends on various factors. The core performance heavily relies on the selection of algorithm and pre-process the data set in an iterative process until the confidence score gets higher. Benchmark in this context is accuracy, precision, recall. ANPR performance benchmarks can be evaluate by its extraction rate, segmentation rate, recognition rate and overall time it takes to present the data in the chosen deployment interface. (An, Wang and Chen, 2022)

One study combined different technologies, including OpenCV, YOLO, PaddleOCR, and Tesseract OCR, to develop ANPR systems and evaluate their performance in different environmental conditions. Another innovative approach focused on license plate recognition using Grab Cut algorithm with Bernsen algorithm and Wiener filter to improve recognition accuracy. Other notable studies proposed robust deep learning-based models that used

character segmentation using Convolutional Neural Networks (CNN) to improve license plate recognition accuracy. In addition, studies have investigated the application of long short-term memory (LSTM) in various machine learning applications, emphasizing its combination with other deep learning methods. Also, the field has developed ALP recognition systems, hybrid models combining CNN and RNN, and methods using visual attention, which aim to improve the accuracy and efficiency of license plate recognition under different conditions.

Below the comparison can be encapsulates in a mind map diagram.

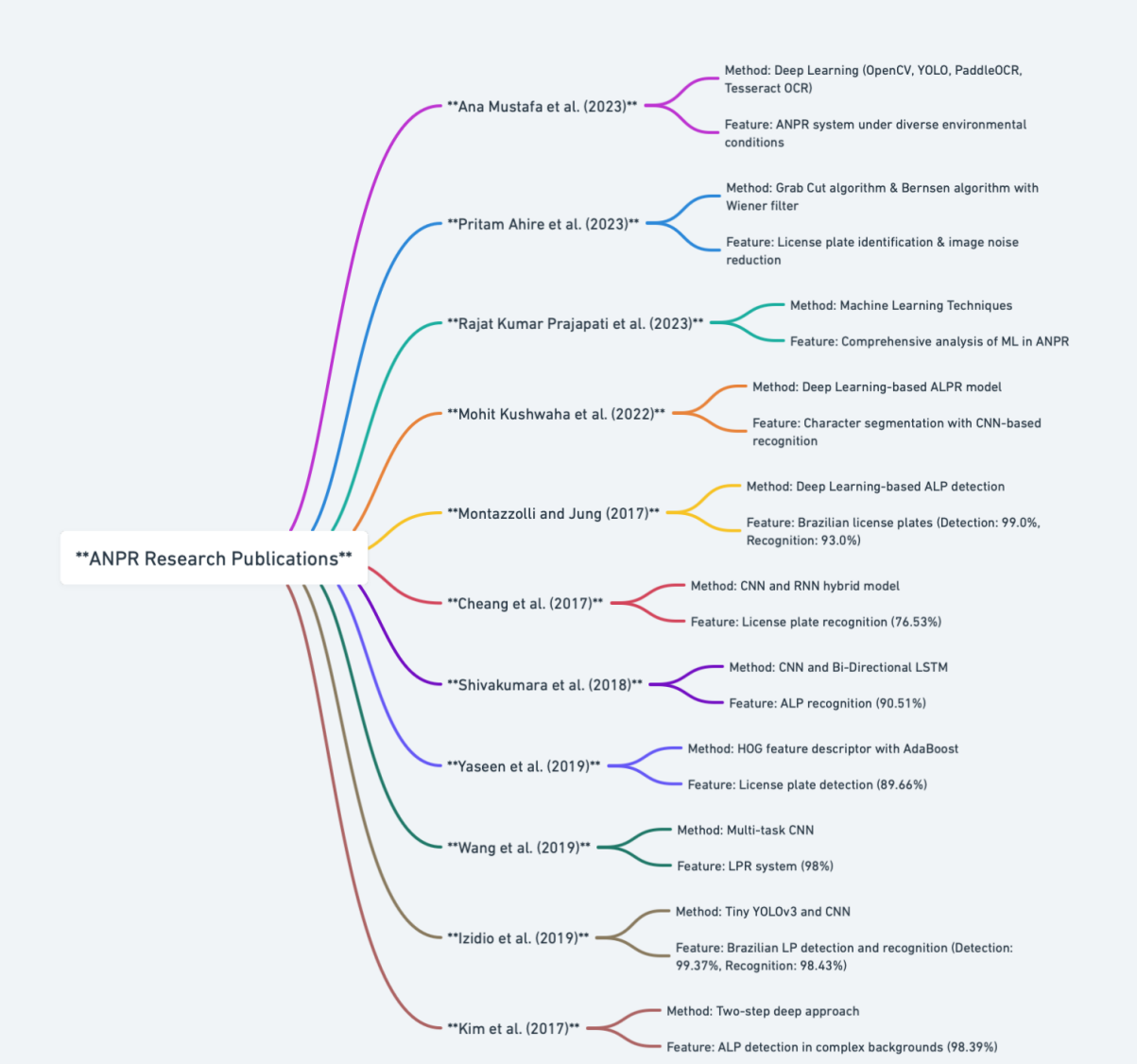

Figure 1 Visualizing ANPR Research in Mind Map.

2.6 Conclusion

To boil down the literature review, it is evident that ANPR has underpin transformative advancements, particularly when strengthened by deep learning and computer vision techniques. Through the lens of seminal works by Shi et al. (2015) and Diaz et al. (2017), the literature underscores the predominant significance of dataset selection in achieving high recognition precision. A meticulously curated dataset not only serves as the bedrock for training a robust model but aid to achieve expected benchmark for evaluating their performance. The art of model tuning is equally significant, which fine-tunes the neural architectures to confirm optimal results. The interplay between dataset selection and model tuning appears to be the key player in the quest to optimize ANPR systems side by side getting to know the trajectory of future innovations in this area. This literature review, therefore, setting the tone for a focused exploration into their role in ANPR performance.

3.0 Scope:

3.1 Research Scope:

In alignment with the research objectives, this review will:

- Enquire into the cutting-edge world of deep learning, with a spotlight on its relevance to ANPR systems.
- Elucidate the difficulties of selecting and refining the optimal dataset for the proposed ANPR model.
- Outline the blueprint for crafting ANPR in terms of software development life cycle (SDLC); a system that can demonstrate on deep learning and identifying vehicle plates.
- Evaluate the model's accuracy, benchmarking it using key metrics and comparing it to existing industry standards.
- Highlight potential avenues for enhancement and shed light on promising horizons in the realm of ANPR research.

3.2 Research Limitation:

Deep learning model are primary rely upon enormous computational power to execute from coding to deployment. Acquiring the optimal dataset is inheritably exposed the challenge; conversely affect the model performance several pivotal facets emerge:

1. The undertaking at hand is an individual work, spanning a 30-week undergraduate project course. It's crucial to acknowledge that in real-world scenarios, the construction of an ANPR system is not individual but collaborative effort, often requiring a dedicated team and an extensive timeline, extending over months if not years.

2. The deployment of deep learning takes significant computational power, particularly a GPU processor. For testing and demonstration purposes, this research will leverage the free GPU provided by Google via Google Collab. While the free version is not free from errors and does not allow running multiple sessions, it provides a viable platform for initial exploration and implementation. (Nvidia.com, 2012)

3. Having no prior engagement in deep learning project, the challenge was twofold: grasping foundational and intricate concepts while parallelly experimenting hands-on, drawing from a surplus of online resources. This was especially relatable given the absence of a dedicated deep learning module within the degree curriculum.

4. This effort is intended not as an endpoint but as a foundational step. The aspiration is for this work to serve as a stepping stone, driving the researcher further along the path of deep learning study.

5. Experimenting through various tools and platforms was not without its moments of inactivity. Deciding the most apt tools for the task at hand often led to periods of deliberation and procrastination.

6. While expertise in the domain was admittedly limited, the guiding light throughout this journey was the sagacious guidance of the supervisor, PhD in the field of machine learning. It's owing to this mentorship that the research could navigate challenges, culminating in a substantive and insightful thesis write-up.

3.3 Dataset:

In this study dataset has initially been downloaded form Kaggle pre-processed in roboflow to train the model in a trial-and-error basis to get the optimal result. This study straight forward in terms of data collection. This study only represents opensource secondary data. Data collection and use of information is also with in line with data protection act 2018 (Legislation.gov.uk, 2018). The study is only dealing with vehicle number plate without looking up the any other details of the drivers or the cars. For testing the model various picture on social media marketplace has been chosen whereby users deliberately share the number plate in public domain.

4.0 Methodology:

4.1 Rationale of ANPR research:

With the surge in vehicular traffic, old-fashioned monitoring methods, like manual oversight and traffic police surveillance, are becoming increasingly inadequate. The escalating reliance on vehicles has amplified the probabilities of traffic violations, resulting unanticipated mishaps and escalating road traffic-related offenses. Addressing these challenges necessitates the deployment of a sophisticated traffic monitoring mechanism ANPR has emerged as a prominent and impactful area within computer vision (Hasan Erdinç Koçer and Kerim Kürşat Çevik, 2011). Such an advanced system can significantly enhance traffic management by adeptly identifying vehicle number plates. This study endeavours to design a system adept at detecting and decrypting vehicle number plates, exploiting the power of convolutional neural networks (CNN), a deep learning model.

4.2 Deep Learning in ANPR

Contemporary expansion in 2023 of Ultra Low Emission Zone (ULEZ) at London UK administer by ANPR camera. This is a prime example of enterprise grade modern ANPR systems, noticeably upgradation of software and hardware infrastructures. Compare to that; this study is miniature version of ANPR investigate how the region of interest can be detected and make readable through Optical Character from static image. The base architecture is CNN from YOLO model. Convolutional operations are foundational to convolutional neural networks, commonly used in image processing tasks. In convolution, an input (like an image) is combined with a kernel (a smaller, learned matrix) to produce a feature map. This operation can be visualized as sliding the kernel over the input and computing the sum of element-wise products at each position. The process can be applied to multi-dimensional data, such as 2-D images. (Li, Wang and Shen, 2019)

The convolution operation is commutative, meaning the order of input and kernel doesn't matter. In machine learning, the dataset often comes in the form of multidimensional arrays, both for the input and the kernel. These arrays, commonly referred to as tensors, contain elements that are adjusted and fine-tuned by the learning algorithm. Given that each element in both the input and the kernel is stored distinctly, it's a common practice to assume that values not explicitly stored are zero. This simplifies the process, allowing us to represent what would be an infinite summation with a more manageable summation over the stored elements. However, many machine learning libraries implement a related function called cross-correlation, which doesn't flip the kernel. This operation can be represented as matrix multiplication, specifically with a Toeplitz matrix in one dimension. In the context of neural networks, the specific values in the kernel are learned during training to optimize a given task.

4.3 Dataset Selection and Pre-processing:

4.3.1 importance of Dataset Selection:

A comprehensive and diverse dataset is crucial for training a robust ANPR model. The dataset's quality directly impacts the model's accuracy and its ability to generalize across different scenarios.

4.3.2 Challenges in Dataset Procurement:

Obtaining a balanced and diverse dataset for ANPR can be challenging due to the variety of license plates, lighting conditions, and potential obstructions. Ensuring that the dataset incorporates all these differences is essential for a reliable ANPR system. A larger dataset means more time required to process and train. Time of inertia seems more at this stage faced by the runtime error when the data may look very attractive but not ready for the use case perhaps. File not found error and data out of index error has been observed on some optimistic large dataset. (Larxel, 2020)

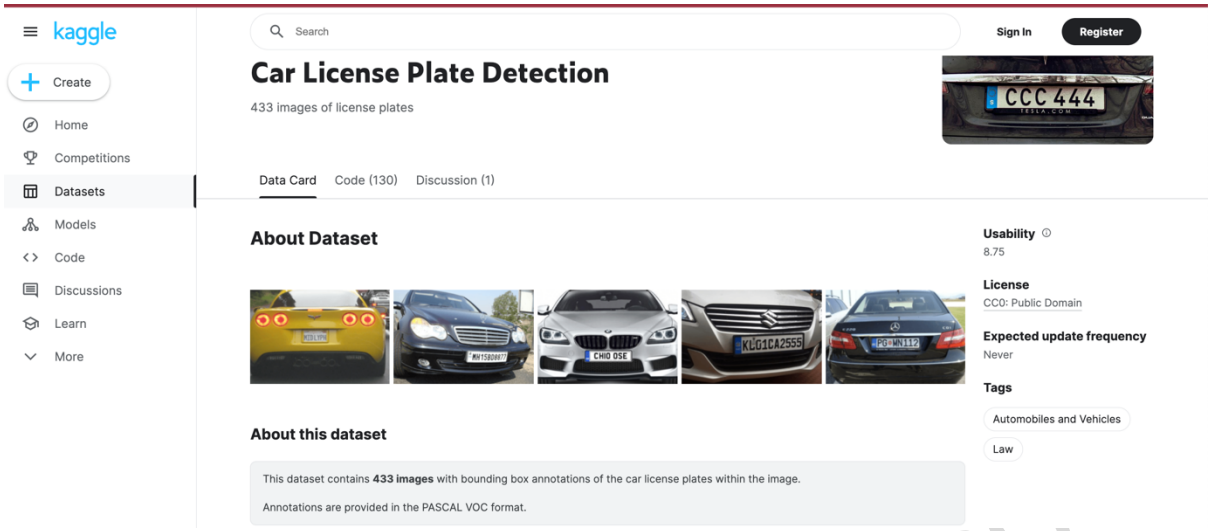

Figure 2 Figure Original dataset with 433 images from Kaggle.

4.3.3 Pre-processing Techniques:

Datasets often require pre-processing to enhance their utility. For ANPR, this might include image normalization, grayscale conversion, and noise reduction using the OpenCV library. In the next method web-based platform roboflow is used to make them ready in the right format. These steps ensure that the model is trained on images, leading to better performance. Images has carefully been selected and only 300 has been taken to finally use as dataset. (Bolton, 2023)

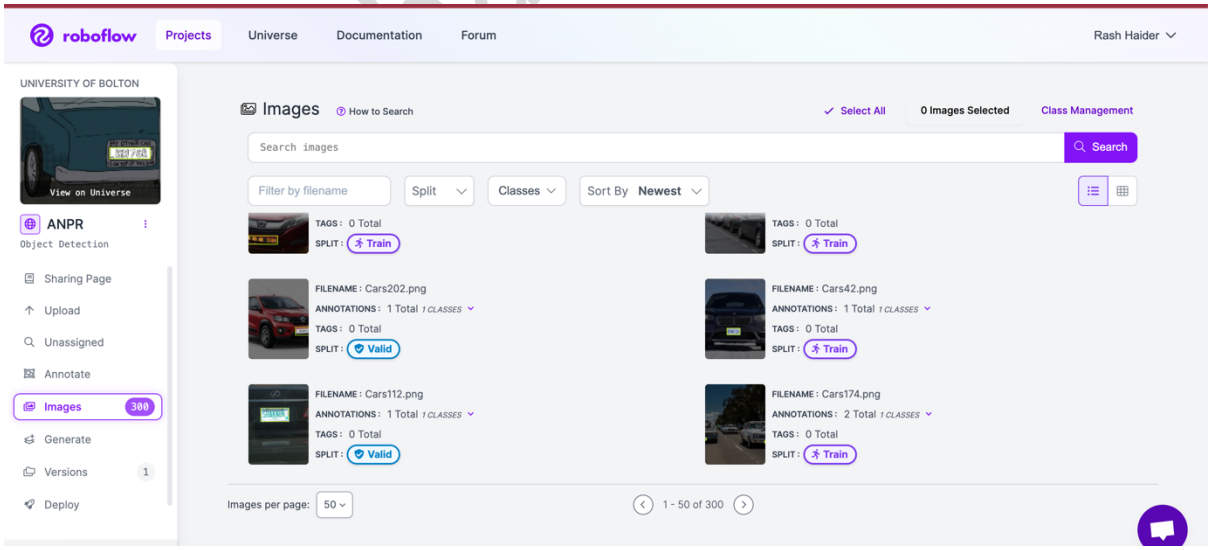

Figure 3 Figure Processed dataset with 300 images from roboflow

Bounding box are adjusted by going through each image. Beauty of roboflow lies as this allow the dataset to be downloaded to the notebook through roboflow api that makes more secure and effortless ensures no data loss.

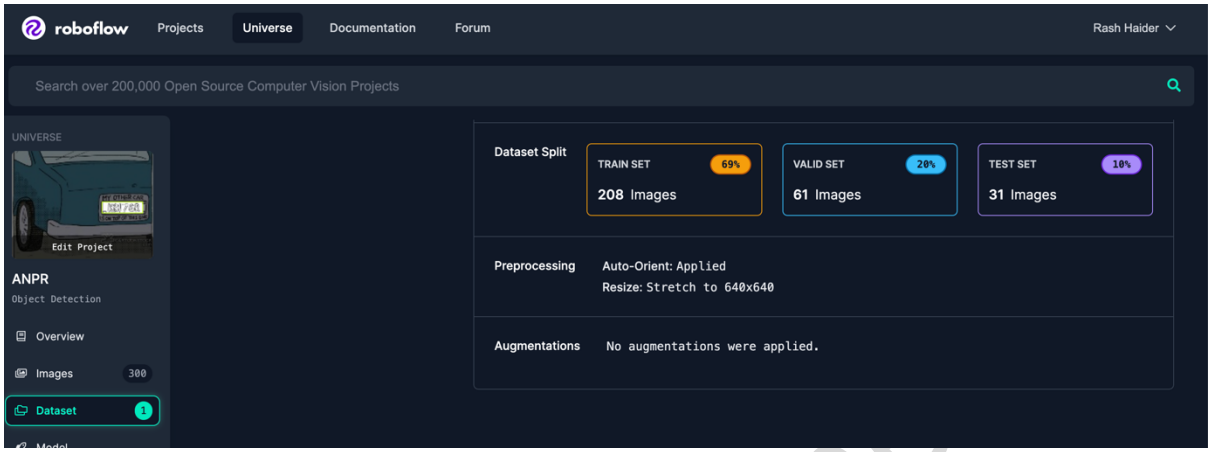

Figure 4 Dataset split for train, test and validation.

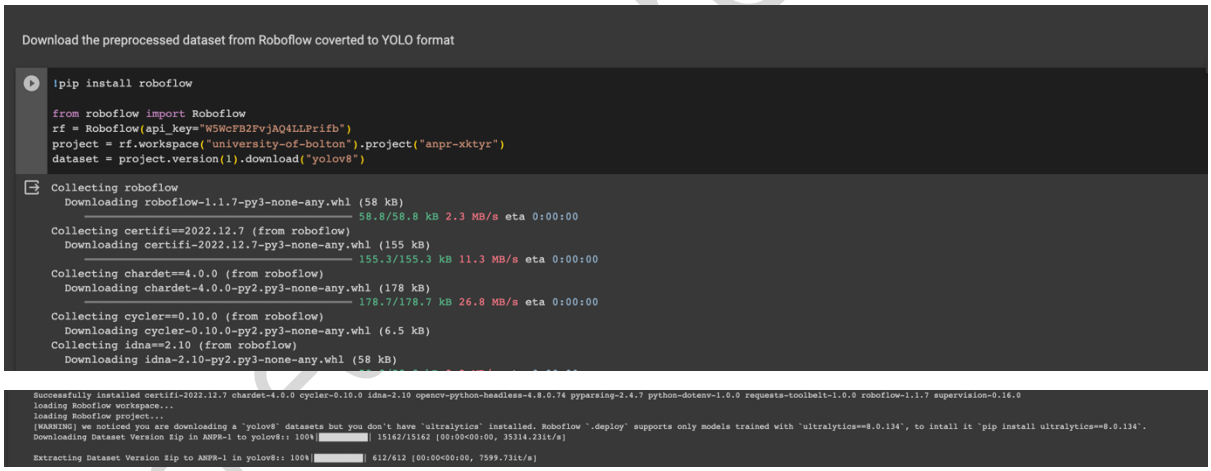

Figure 5 Download dataset in Google Colaboratory notebook.

4.4 Algorithm & Libraries used:

The ANPR system developed in this project leverages a combination of algorithms and libraries tailored for image recognition. Specifically, the YOLO algorithm is used for object detection, while the EasyOCR tool facilitates optical character recognition. The OpenCV library

provides essential functions for image processing, ensuring the quality and clarity of the images being analysed.

```
from flask import Flask, render_template, request
from werkzeug.utils import secure_filename
import os
import numpy as np
import cv2
import easyocr
from ultralytics import YOLO
import csv
from datetime import datetime
```

Figure 6 Import libraries list in a nutshell.

4.5 Frameworks and architecture used:

Default Baseline model is YOLO. (ultralytics, 2023)

The ANPR system has been locally deployed for demonstration purposes using the Flask framework in Visual Studio Code. Flask, a lightweight web application framework, allows for the creation of a user-friendly interface where users can upload images for license plate recognition. The backend architecture integrates the YOLO algorithm, OpenCV, and EasyOCR to process the uploaded images, detect license plates, and recognize the characters. The results, along with timestamps, are then saved in a CSV file for record-keeping.

4.6 Application of Kanban and Scrum:

In the development of the ANPR system, the agile methodologies of Kanban and Scrum were seamlessly integrated to optimize workflow and task management. The Kanban board visualized the project's progression, with tasks represented as cards moving through stages like "To Do," "In Progress," and "Done." This visualization ensured continuous tracking and quick identification of bottlenecks. On the other hand, Scrum divided the 30-week project into four distinct sprints, each with its objectives, from model creation to deployment. Regular sprint reviews ensured alignment with project goals, and any deviations were promptly addressed.

The solo nature of the project posed challenges in task distribution, making prioritization vital. Here, the Kanban board was instrumental, offering clarity on pending tasks. The iterative approach of Scrum, coupled with Kanban's visual tracking, provided a structured yet flexible framework. This agile combination proved invaluable in navigating the intricacies of developing the ANPR system, underscoring the importance of adaptability in software development.

Kanban Board							
Backlog			**Doing**	**Tesing**	**Status**	**Size**	**Sprint**	**Priority**
Dataset Acquisition			Done	Done	Done	12	1	High
Dataset Pre Processing			Done	Done	Done	12	1	High
Initial Model Creation			Done	Done	Done	20	1	High
Model Fine-tuning			Done	Done	Done	8	1	High
ROI Detection			Done	Done	Done	10	2	High
OCR Implementation			Done	Done	Done	8	2	High
License Plate Annotation			Done	Done	Done	12	2	High
HTML Interface Design			Done	Done	Done	12	3	High
Local Deployment			Done	Done	Done	20	3	High
Webcam Integration			Dismiss	Dismiss	Dismiss	8	3	Low
Code Refactoring			Done	Done	Done	4	4	Low
Error Handling			Done	Done	Done	6	4	Medium
CSV Data Storage Mechanism			Done	Done	Done	6	4	Low
Sprint 1	Sprint 2	Sprint 3	Sprint 4		**Total Hours**	**138**		

Figure 7 Final version of Kanban Board

(Cohn and Beck, 2011)

5.0 Requirement Analysis:

5.1 Introduction

The Automatic Number Plate Recognition (ANPR) system is designed to detect and interpret vehicle number plates from images. This demonstration version is tailored for local hosting and focuses on only image-based detection, excluding video or real-time tracking functionalities. The system is trained on the image dimension 650 by 450 pixel and can only able to detect images closer to the pixel. The system enable user to upload image file format like jpeg, png.

5.2 Purpose

The purpose of this section is to present a complete software requirement analysis for the ANPR system, detailing its functionalities, user interactions as well as development iterations.

5.3 Scope

The ANPR system is premeditated for demonstration purposes created. The project is an individual effort, spanning a 30-week undergraduate course duration. It will be hosted locally and is designed to detect number plates from images only image pixel 400 X 400. Real-time detection object tracking functionalities are not included in this version.

5.4 User Stories:

Only one user is considered in this scenario for simplicity. To zoom down the requirement from user perspective as a user, I want to:

User Story ANPR WebApp

Upload an image containing a vehicle number plate.

Receive the detected number plate text as output.

View the region of interest and the bounding box around the detected number plate.

Access the system through a user-friendly web interface.

Receive error messages for any issues during the detection process.

Save the detected number plate information in a CSV format for record-keeping.

Figure 8 User Stories

5.5 Workflow Iterations:

Demonstration of 4 sprint followed to shadow the iterative and test-driven development ideology and design patterns discussed in the next chapter. Kanban board incorporation streamline the workflow and subdivide the task in a manageable chunk. (Rubin Kenneth 2013, p16-18)

5.5.1 Sprint 1:

Model Development and Initial Training done in the first sprint with primary focus lay the foundation on developing the ANPR model using various open-source datasets. Understanding data preparation into its right format.

Training was executed in Google Colab, leveraging its enhanced GPU computational capabilities.

5.5.2 Sprint 2:

Fine-Tuning and OCR Integration: The model underwent fine-tuning, and Optical Character Recognition (OCR) was integrated to interpret characters on detected number plates. To remove the small text of the number plate OCR threshold was set to 0.2 meaning that only 80% of the occupied text will be considered remaining small text will be filtered out.

5.5.3 Sprint 3:

Deployment and Real-time Detection: The model was deployed locally using Flask with an HTML user interface. An attempt was made to integrate a webcam for real-time detection, revealing latency issues due to limited GPU resources.

5.5.4 Sprint 4:

Refactoring, Error Handling, and Data Storage has taken at this stage. Code refactoring is a decent practice to undertake for at the end to check how well the code is structured and find any room of improvement. Error handling mechanisms were integrated upon considering that user might forget to upload any images yet press the submit button or other case uploading an unsupported file format. To add reporting feature recognition results are saved in CSV format for further analysis.

5.6 Functional Requirements

Image Upload: The system shall allow users to upload images. Supported formats: JPEG, PNG.

Number Plate Detection: The system shall detect number plates from the uploaded images. Detected plates shall be highlighted using bounding boxes.

Character Interpretation: The system shall interpret characters from the detected number plates using OCR.

Data Storage: The system shall provide an option to save detection results in a CSV format.

5.7 Non-Functional Requirements:

Non-functional requirements lay out the performance and security and safety of the user in a given system.

Performance: The system shall process image uploads without delay. The output should be visible momentarily.

Usability: The user interface shall be intuitive and user-friendly.

Error Handling: The system shall handle errors gracefully, providing informative feedback to the user.

5.8 Conclusion

The ANPR system, developed over four iterative sprints to shadow the agile methodology, the potential of individual initiative in deep learning and computer vision. While designed for demonstration purposes, it serves as a foundation for future enhancements and real-world applications. The system is designed for demonstration purposes and is hosted locally. Real-time detection or tracking is not supported in this version. The system's performance is constrained by the absence of dedicated GPU resources.

6.0 Software System Design:

6.1 Design pattern principle:

Software design is the next step towards the development whereby the project blueprint carefully drawn to ensure modular, scalable, and robust code structures that are easier to apprehend and modify can be enrich over time. Based on the workflow, the following five significant software design principles and patterns are evident:

Separation of Concerns (SoC): The process distinctly divides tasks into specific phases, such as pre-processing with Roboflow, model training in one notebook, and deployment in Visual Studio Code. This division ensures clarity and ease of management for each segment.

Model-View-Controller (MVC): In the deployment phase, using app.py and HTML templates essentially embodies the MVC pattern. The trained ANPR system acts as the model, the HTML templates represent the view, and the app.py serves as the controller, orchestrating the interaction between the model and view.

Prototype Pattern: The model underwent iterative refinement, starting from a foundational version and subsequently incorporating enhancements like OCR, mirroring the prototype pattern's approach of creating and refining a solution.

Factory Pattern: The setup phase, where decisions like selecting YOLOv8 or determining the number of epochs were made, aligns with the factory pattern, centralizing the creation of models based on specific parameters.

Decorator Pattern: Additional functionalities, such as OCR and bounding box detection, can be perceived as "decorations" to the base ANPR model, aligning with the decorator pattern's concept of dynamically adding new features to objects.

6.2 Architectural & High-level Design:

Even though this is academic project; the attention to detail should not be undermined by the core principle of design practice relatable with software development life cycle. Underpinning the workflow and choosing right method and application syncing by the programming language alongside base architecture algorithm set the tone to get started. This section will encompass few other principle and flow. (Cohen Mike, 2009 p4-5)

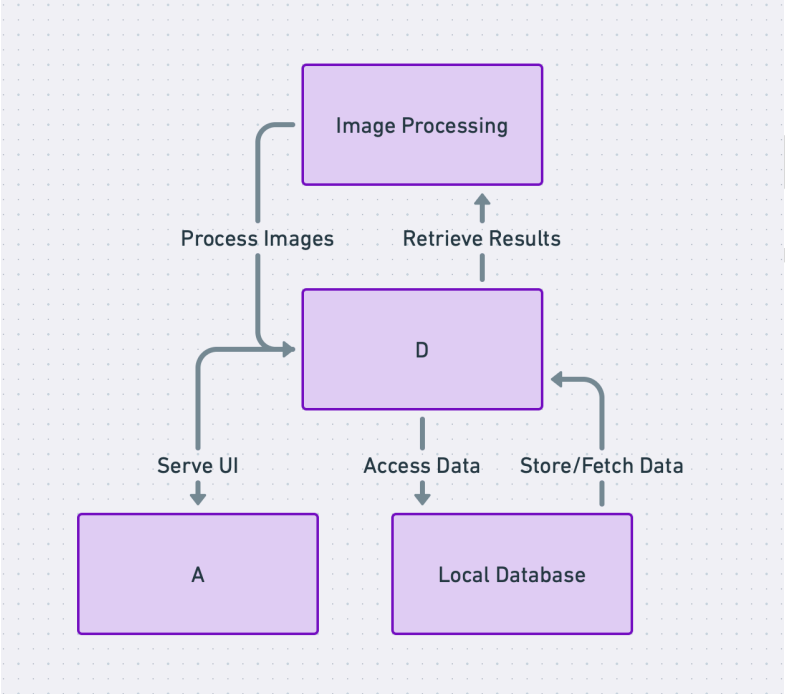

Figure 9 ANPR Web App Component Diagram

6.2.1 Use Case Diagram:

The interactions between the User, Web App, Local Server, and Flask Backend are depicted in the diagram. It explains the main use cases as well as the system's flow of processes.

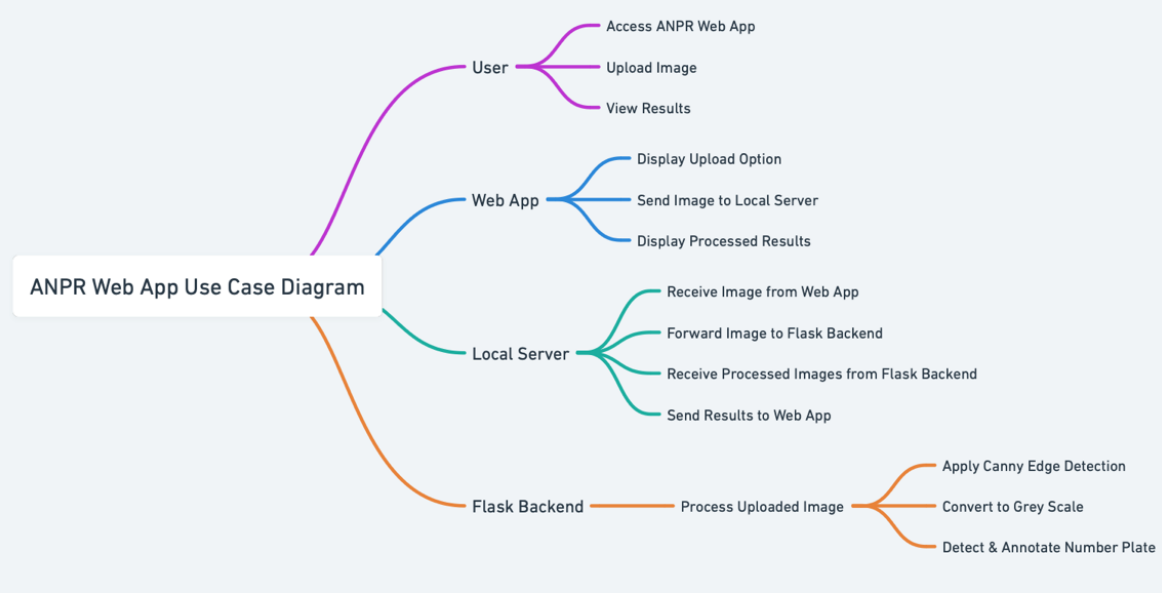

Figure 10 ANPR Use Case Diagram

6.2.1 Dataflow Diagram:

The data flow between the User, Web App, Local Server, and Flask Backend is represented in the diagram. It displays the user how the image is uploaded, processed, and returned with the results.

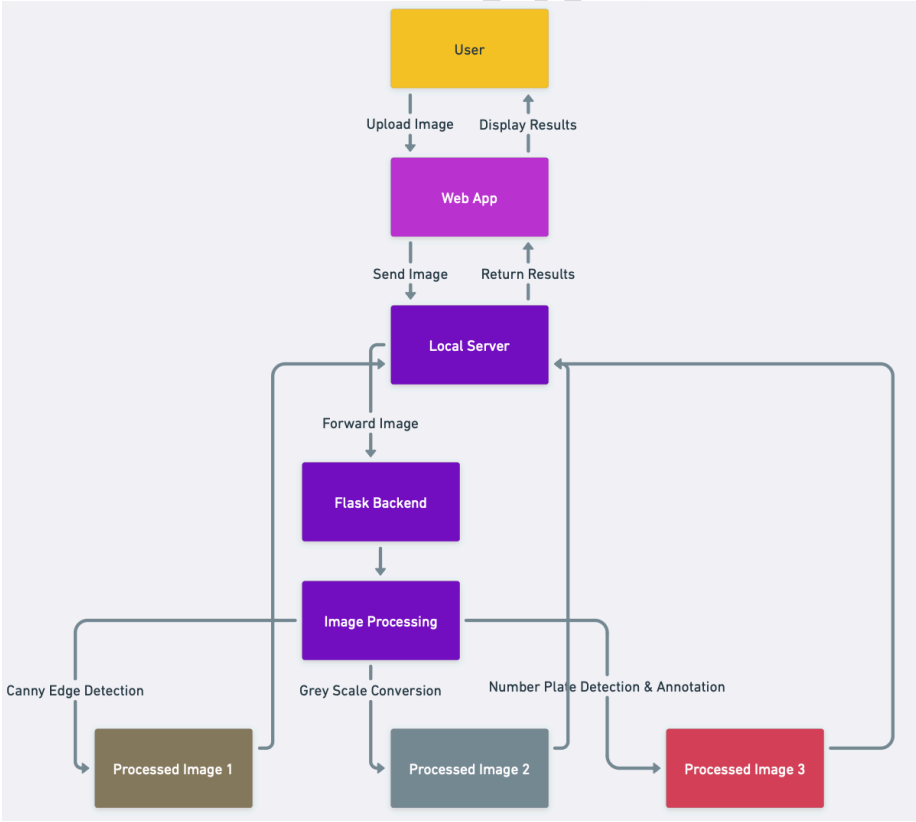

Figure 11 Data Flow Diagram

6.2.3 Workflow Diagram

Its crucial to comprehend the workflow that in essence illustrate the sequential process from start to finish of the development software project.

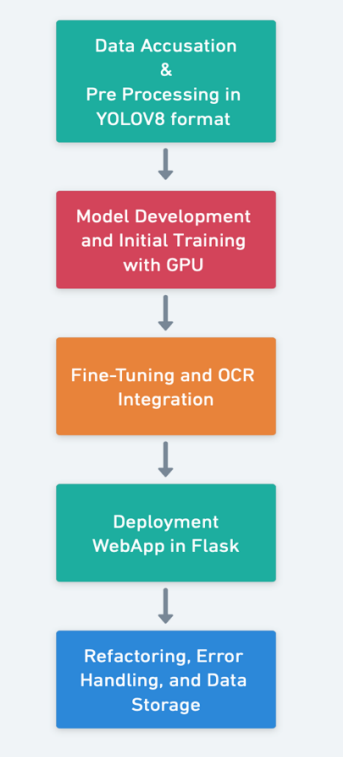

Figure 12 Workflow Diagram

7.0 Implementation:

This chapter provides a comprehensive overview of the implementation phase of the ANPR system. It sheds light on the decisions made, the challenges encountered, and the solutions adopted, a holistic understanding of the process. The implementation phase is pivotal in the development of any system, and the Automatic Number Plate Recognition (ANPR) system is no exception. This chapter delves into the intricacies of the implementation process, detailing the decisions made, challenges faced, and solutions adopted to ensure the system's successful realization.

7.1 Goals of Implementation:

The primary objectives of this phase were multifaceted. They encompassed the development of a functional ANPR system, the seamless integration of its various components, and ensuring its smooth operation in diverse conditions.

7.2 Tool and Platform Selection:

Any software implementation the coherence between tools and platforms is utmost. The selection process was guided by the need for compatibility, ensuring that each tool could seamlessly integrate and operate in seamlessly with others. The journey to identify the right tools was not without its complexities. The landscape of available tools is vast, and finding those that would work harmoniously posed a significant challenge.

7.3 Dataset Implementation:

Any deep learning model is only as good as the dataset it's trained on. The selection of an appropriate dataset was crucial to ensure the model's accuracy and reliability. Initial optimism led to the selection of a large dataset. However, this choice presented numerous challenges during the training iteration stage, from handling the diversity of license plates to managing varying lighting conditions. To prepare the dataset for training, several pre-processing techniques were employed. The Roboflow web service proved invaluable in this regard, aiding in data augmentation, cleaning, and preparation.

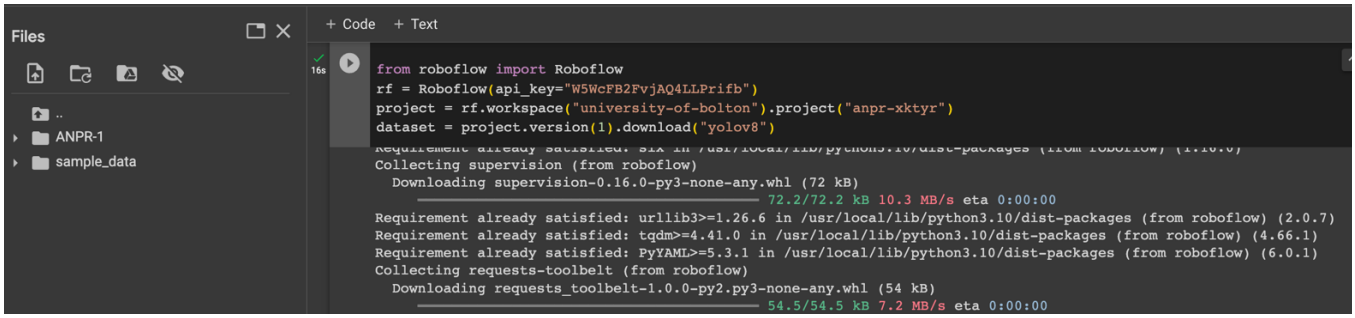

Figure 13 Downloading dataset direct to notebook via API

Downloading dataset direct to notebook creates the dataset with it necessary folders for the model

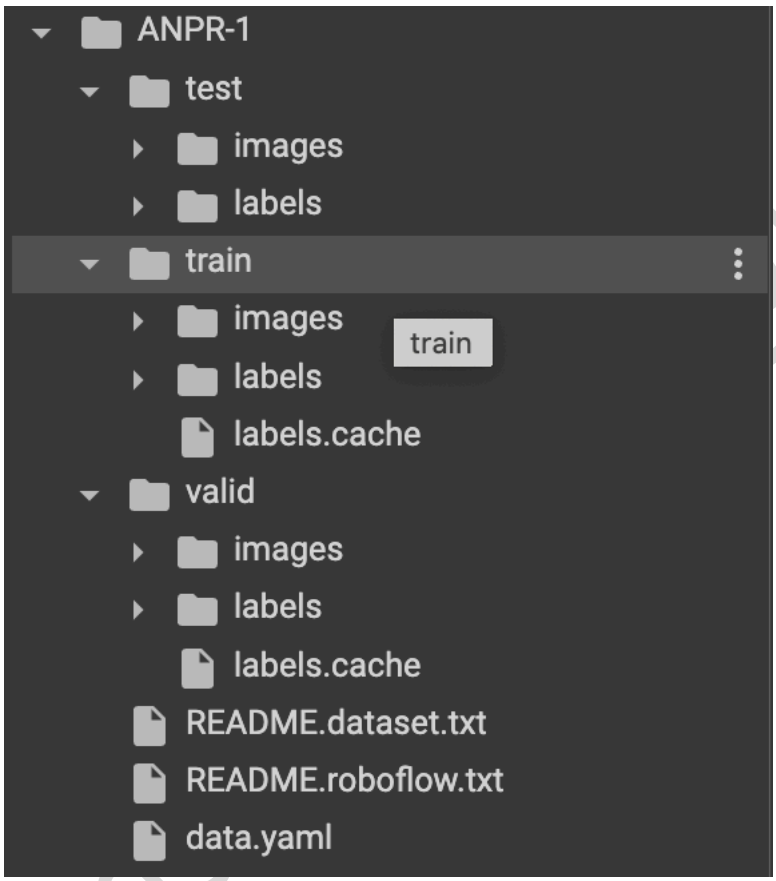

Figure 14 Dataset File Explorer view

At this stage the data.yaml file need to modified according to the path of images of test train and validation. The role yaml file is to point out the directories of the dataset.

*data.yaml

```yaml
names:
- licence
nc: 1
roboflow:
  license: Public Domain
  project: anpr-xktyr
  url: https://universe.roboflow.com/university-of-bolton/anpr-xktyr/dataset/1
  version: 1
  workspace: university-of-bolton
test: /content/ANPR-1/test/images
train: /content/ANPR-1/train/images
val: /content/ANPR-1/valid/images

```

Figure 15 Data.yaml file attributes

7.4 Frameworks and Architecture Implemented:

Libraries like OpenCV and easyOCR played a pivotal role in the system's implementation. They provided the necessary functions and methods to process images, detect number plates, and recognize characters. The decision to use the YOLOv8 medium model as the base was influenced by the limited number of epochs available for training. However, for more extensive training, the larger L model remains a viable option. The ANPR system's architecture is a harmonious blend of various components. From image processing to character recognition, each component was meticulously integrated to ensure seamless operation. (Ultralytics.com, 2023)

7.5 Model Training and Validation:

Training Process:

Training the ANPR model was a meticulous process. From initializing weights to optimizing the learning rate, each step was carefully executed to ensure the model's efficacy. Below screenshot showing the resources used to train the model.

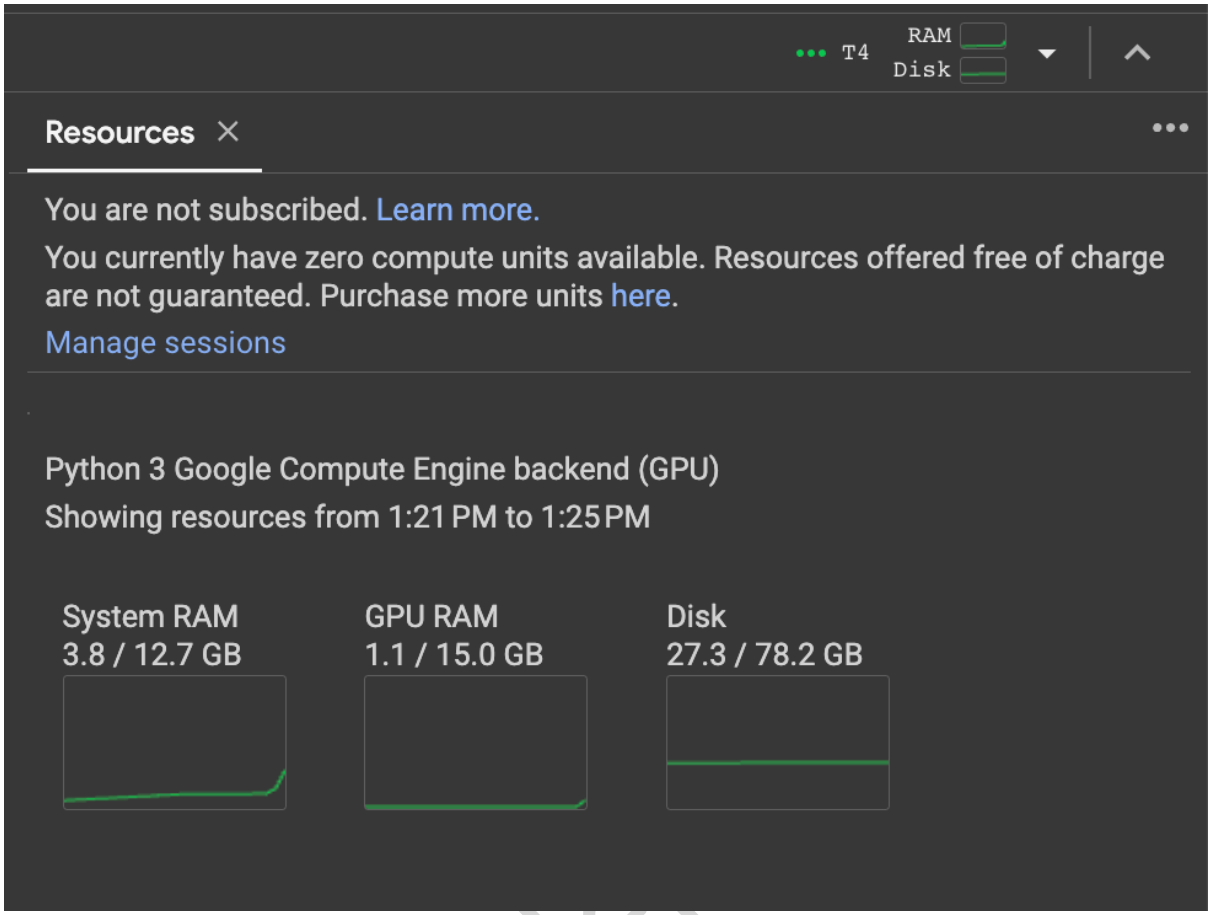

Figure 16 GPU for Training the Model

The model is imported and connect with the data.yaml file.

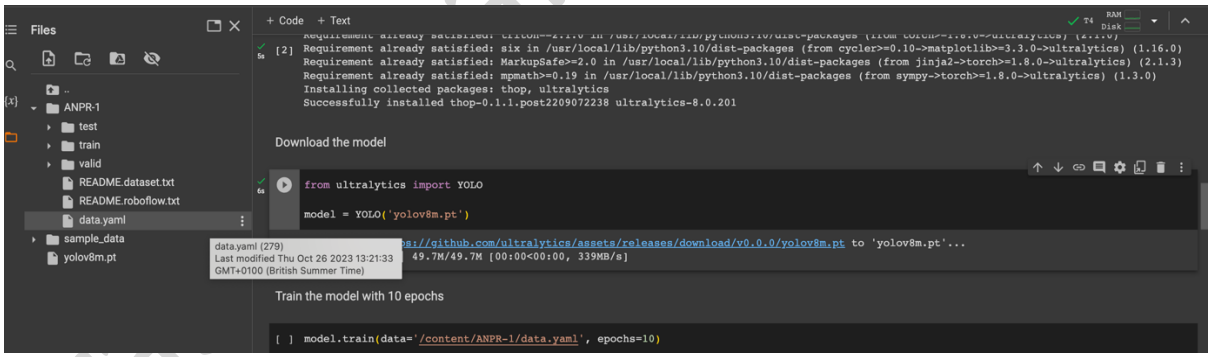

Figure 17 Download YOLOV8 m model in the notebook.

Using the YOLO version 8 medium model since number of epochs is only 10. The training has started and the logs are visible. (GitHub, 2023)

Train the model with 10 epochs

```
model.train(data='/content/ANPR-1/data.yaml', epochs=10)

Ultralytics YOLOv8.0.200 🚀 Python-3.10.12 torch-2.1.0+cu118 CUDA:0 (Tesla T4, 15102MiB)
engine/trainer: task=detect, mode=train, model=yolov8m.pt, data=/content/ANPR-1/data.yaml, epochs=10, patience=50, batch=16, imgsz=640, save=True, save_period=-1, cache=False,
Downloading https://ultralytics.com/assets/Arial.ttf to '/root/.config/Ultralytics/Arial.ttf'...
100%|██████████| 755k/755k [00:00<00:00, 28.5MB/s]
Overriding model.yaml nc=80 with nc=1

                   from  n    params  module                                       arguments
  0                  -1  1      1392  ultralytics.nn.modules.conv.Conv             [3, 48, 3, 2]
  1                  -1  1     41664  ultralytics.nn.modules.conv.Conv             [48, 96, 3, 2]
  2                  -1  2    111360  ultralytics.nn.modules.block.C2f             [96, 96, 2, True]
  3                  -1  1    166272  ultralytics.nn.modules.conv.Conv             [96, 192, 3, 2]
  4                  -1  4    813312  ultralytics.nn.modules.block.C2f             [192, 192, 4, True]
  5                  -1  1    664320  ultralytics.nn.modules.conv.Conv             [192, 384, 3, 2]
  6                  -1  4   3248640  ultralytics.nn.modules.block.C2f             [384, 384, 4, True]
  7                  -1  1   1991808  ultralytics.nn.modules.conv.Conv             [384, 576, 3, 2]
  8                  -1  2   3985920  ultralytics.nn.modules.block.C2f             [576, 576, 2, True]
  9                  -1  1    831168  ultralytics.nn.modules.block.SPPF            [576, 576, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  2   1993728  ultralytics.nn.modules.block.C2f             [960, 384, 2]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  2    517632  ultralytics.nn.modules.block.C2f             [576, 192, 2]
 16                  -1  1    332160  ultralytics.nn.modules.conv.Conv             [192, 192, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  2   1846272  ultralytics.nn.modules.block.C2f             [576, 384, 2]
 19                  -1  1   1327872  ultralytics.nn.modules.conv.Conv             [384, 384, 3, 2]
 20             [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  2   4207104  ultralytics.nn.modules.block.C2f             [960, 576, 2]
 22        [15, 18, 21]  1   3776275  ultralytics.nn.modules.head.Detect           [1, [192, 384, 576]]
```

Figure 18 Train the data with base model

Module with extension .conv.Conv is stands for convolutional neural network. YOLO is essentially CNN rather than ANN or RNN.

```
Model summary: 295 layers, 25856899 parameters, 25856883 gradients, 79.1 GFLOPs

Transferred 469/475 items from pretrained weights
TensorBoard: Start with 'tensorboard --logdir runs/detect/train3', view at http://localhost:6006/
Freezing layer 'model.22.dfl.conv.weight'
AMP: running Automatic Mixed Precision (AMP) checks with YOLOv8n...
Downloading https://github.com/ultralytics/assets/releases/download/v0.0.0/yolov8n.pt to 'yolov8n.pt'...
100%|██████████| 6.23M/6.23M [00:00<00:00, 114MB/s]
AMP: checks passed ✅
train: Scanning /content/ANPR-1/train/labels... 208 images, 0 backgrounds, 0 corrupt: 100%|██████████| 208/208 [00:00<00:00, 1877.04it/s]
train: New cache created: /content/ANPR-1/train/labels.cache
albumentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01), CLAHE(p=0.01, clip_limit=(1, 4.0), tile_grid_size=(8, 8))
val: Scanning /content/ANPR-1/valid/labels... 61 images, 0 backgrounds, 0 corrupt: 100%|██████████| 61/61 [00:00<00:00, 751.43it/s]
val: New cache created: /content/ANPR-1/valid/labels.cache
Plotting labels to runs/detect/train3/labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum' automatically...
optimizer: AdamW(lr=0.002, momentum=0.9) with parameter groups 77 weight(decay=0.0), 84 weight(decay=0.0005), 83 bias(decay=0.0)
Image sizes 640 train, 640 val
Using 2 dataloader workers
Logging results to runs/detect/train3
Starting training for 10 epochs...
Closing dataloader mosaic
albumentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01), CLAHE(p=0.01, clip_limit=(1, 4.0), tile_grid_size=(8, 8))

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/10      6.85G      1.694      4.502      1.649         17        640: 100%|██████████| 13/13 [00:09<00:00,  1.38it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 2/2 [00:01<00:00,  1.03it/s]
                   all         61         68      0.149      0.132     0.0638     0.0323
```

Figure 19 Model Summary page

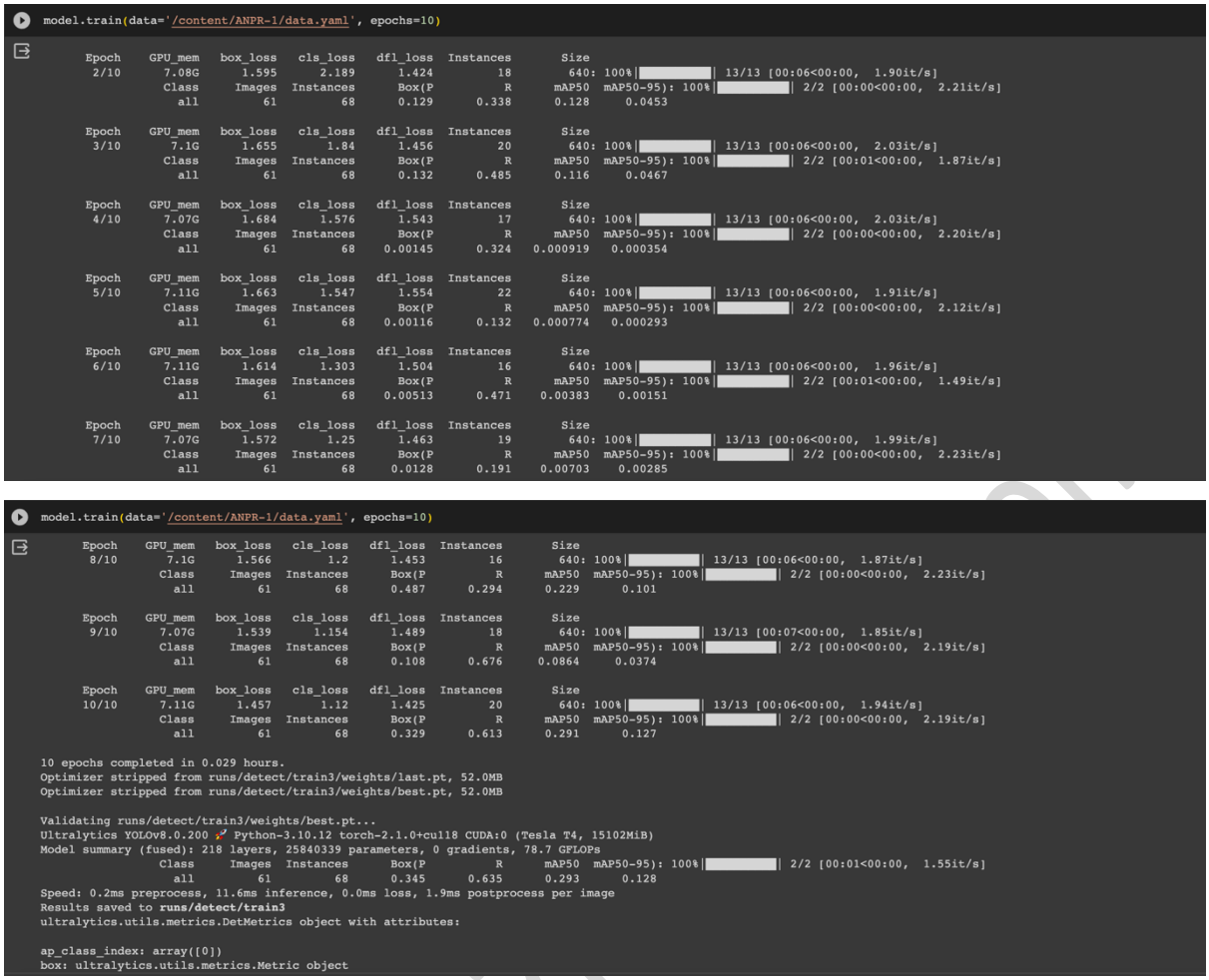

Figure 20 Training the Dataset in 10 Epochs

Each epochs meaning that training is happening between loss and predicted value. The output shows the training process of a YOLOv8 model using the Ultralytics framework. The model is run on a Tesla T4 GPU with 15,102MiB of memory. The training task is license plate detection with a dataset located at /content/ANPR-1/data.yaml and is trained for 10 epochs. The model architecture is displayed, detailing layers and parameters, with a total of 25,856,899 parameters. As training progresses, loss metrics like box loss, class loss, and dfl loss are shown per epoch, as well as performance metrics like precision, recall, and mAP (mean average precision) for both 50% IoU (Intersection over Union) and 50%-95% IoU ranges. After training, the best model is validated, achieving a mAP of 0.293 for 50% IoU and 0.128 for 50%-95% IoU, indicating the model's accuracy in predicting license plates. The inference speed metrics are provided, taking approximately 10.6ms for model inference per image.

Validation and Prediction:

Post-training, the model underwent rigorous validation using a separate dataset. This phase also saw the model being fine-tuned for optimal performance, with particular attention given to the bounding boxes' generalization for image processing. At this stage under detect folder under runs been created and present us best.pt file that is the weighted train file of the model that fits to our dataset.

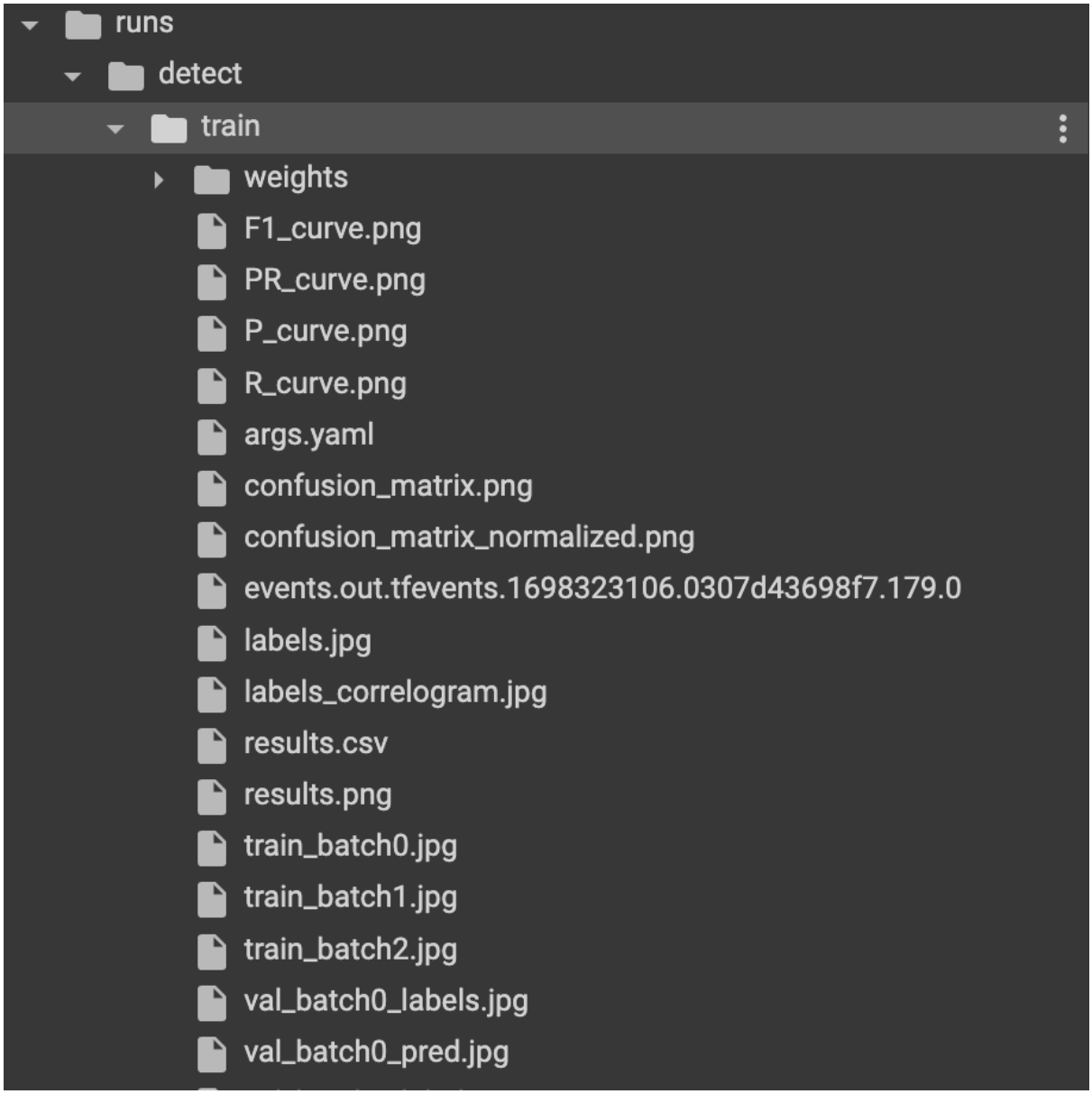

Figure 21 File Explorer overview of the training weighted.

The pre-trained model transfers the weights to the dataset that means it customise to the use case. Details of the prediction using the weighted training gives a prediction of the model.

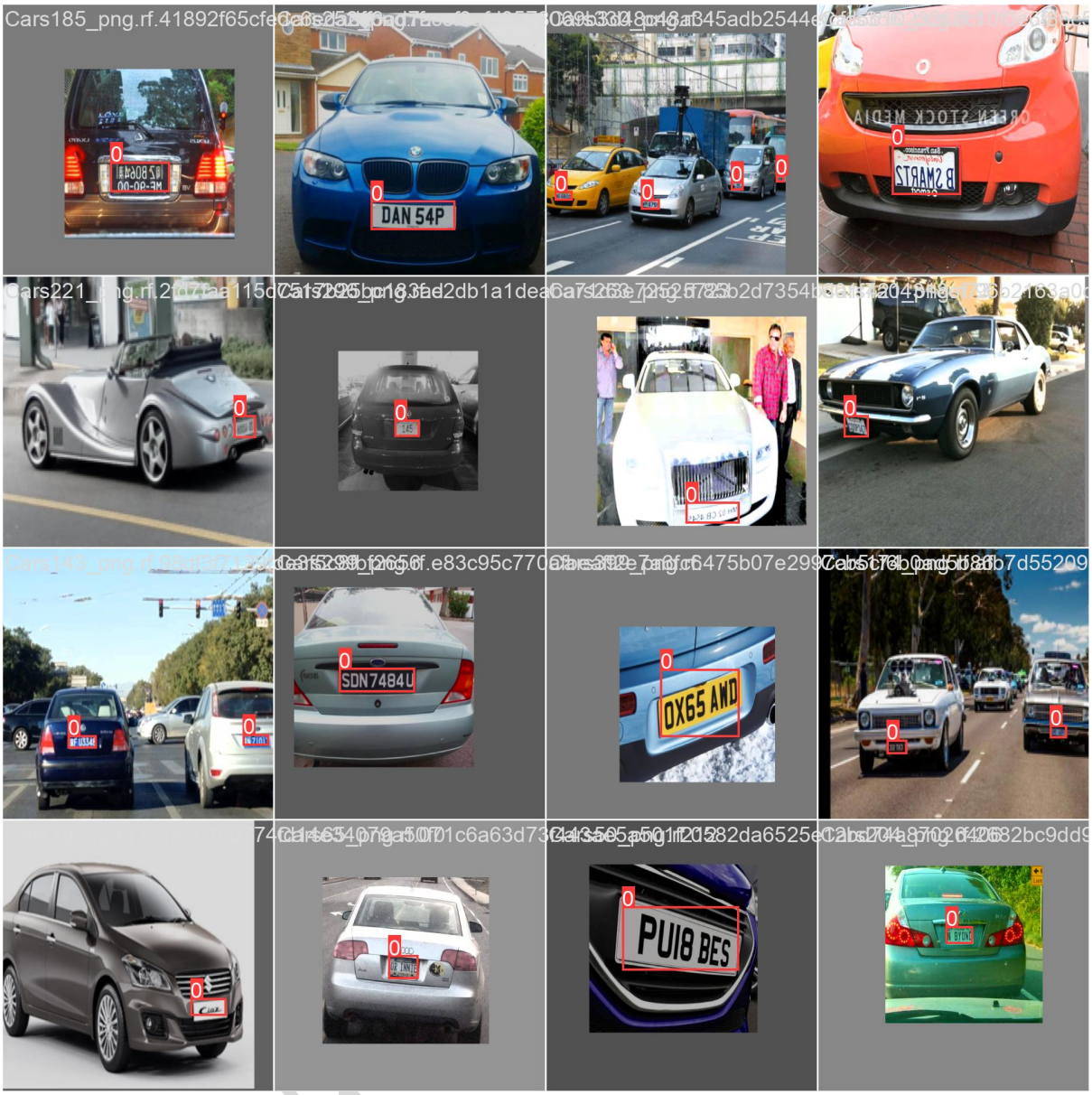

Figure 22 Training Batch 1

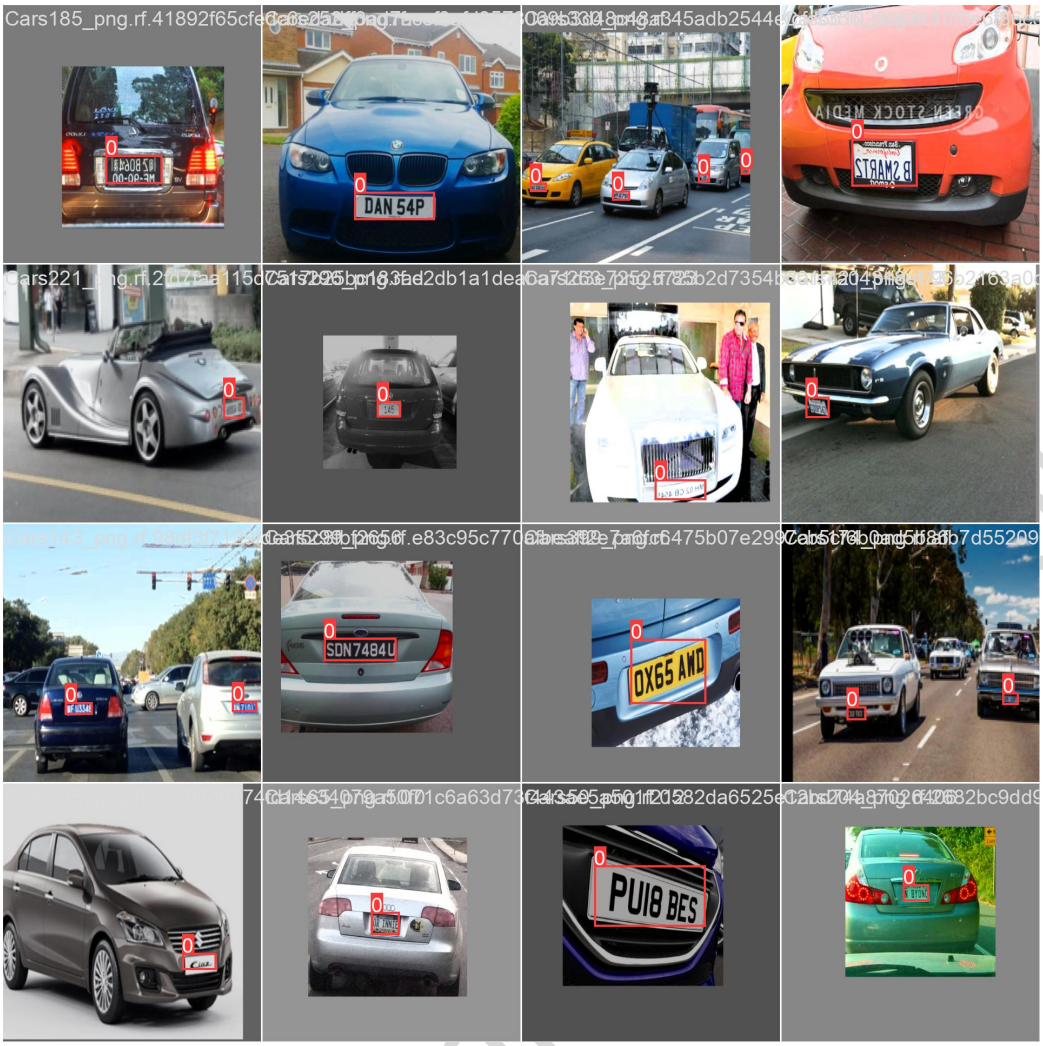

Figure 23 Training Batch

Figure 24 Validation Batch

```
model = YOLO("/content/runs/detect/train/weights/best.pt")

from PIL import Image
import cv2
img = cv2.imread("/content/RashedTest4.jpeg")
prediction = model.predict(img)[0]
print(prediction)

prediction = prediction.plot(line_width=1)
prediction = prediction[:, :, ::-1]
prediction = Image.fromarray(prediction)
prediction.save("output_image.png")
```

```
0: 640x544 1 licence, 75.9ms
Speed: 3.0ms preprocess, 75.9ms inference, 2.0ms postprocess per image at shape (1, 3, 640, 544)
ultralytics.engine.results.Results object with attributes:

boxes: ultralytics.engine.results.Boxes object
keypoints: None
masks: None
names: {0: 'licence'}
orig_img: array([[[206, 190, 174],
        [206, 190, 174],
        [206, 190, 174],
        ...,
        [221, 211, 204],
        [221, 211, 204],
        [221, 211, 204]],

       [[206, 190, 174],
        [206, 190, 174],
        [206, 190, 174],
        ...,
        [221, 211, 204],
        [221, 211, 204],
        [221, 211, 204]],

       [[205, 189, 173],
        [205, 189, 173],
        [205, 189, 173],
        ...,
        [221, 211, 204],
        [221, 211, 204],
        [221, 211, 204]],

       ...,
```

```
       ...,

       [[ 51,  49,  48],
        [ 52,  50,  49],
        [ 53,  51,  50],
        ...,
        [ 75,  75,  75],
        [ 78,  78,  78],
        [ 68,  68,  68]],

       [[ 37,  35,  34],
        [ 42,  40,  39],
        [ 47,  45,  44],
        ...,
        [ 72,  72,  72],
        [ 81,  81,  81],
        [ 79,  79,  79]],

       [[ 23,  21,  20],
        [ 32,  30,  29],
        [ 42,  40,  39],
        ...,
        [ 66,  66,  66],
        [ 78,  78,  78],
        [ 91,  91,  91]]], dtype=uint8)
orig_shape: (1280, 1039)
path: 'image0.jpg'
probs: None
save_dir: None
speed: {'preprocess': 3.0460357666015625, 'inference': 75.87575912475586, 'postprocess': 1.979827880859375}
```

Figure 25 Model Prediction

Even though the sample image is double the pixel size of the training set still it accurately make the prediction.

To visualise our result, we can use matplotlib to display the predict image. The bounding box clearly shows the right area of the images.

```
import matplotlib.pyplot as plt

# Display the prediction image
plt.figure(figsize=(12,8))
plt.imshow(prediction)
plt.axis('off')  # Turn off axis numbers and ticks
plt.show()
```

Figure 26 Visualisation of bounding box detection

7.6 Implementation of Optical Character Recognition:

Next step is going to be new notebook on Google Colaboratory again starting with the necessary imports. (Baek et al., 2019)

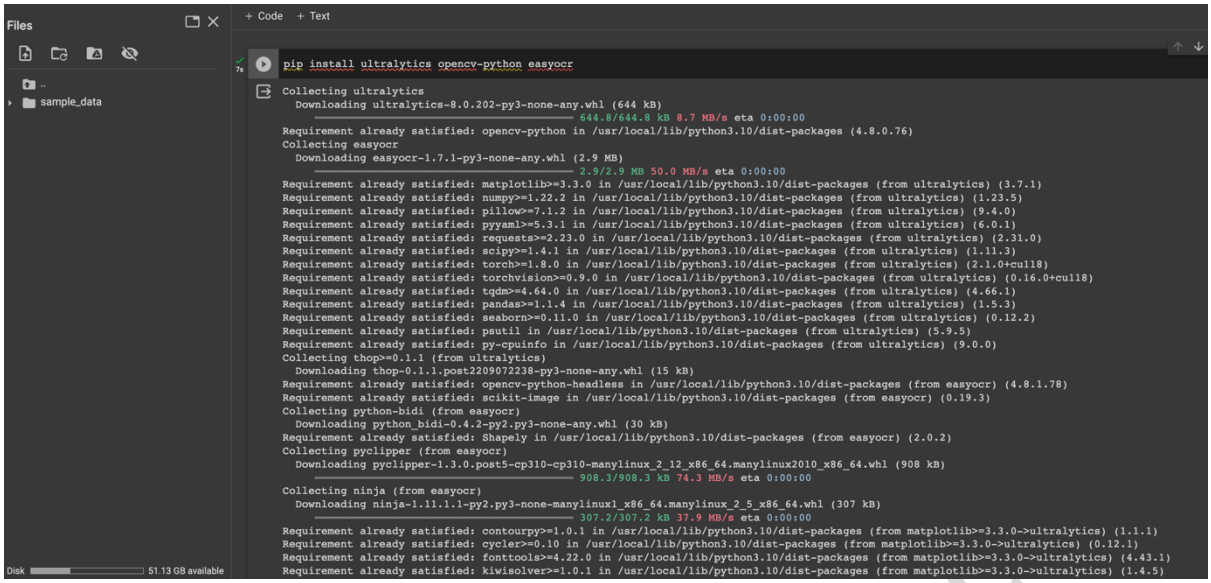

Figure 27 installation for OCR notebook

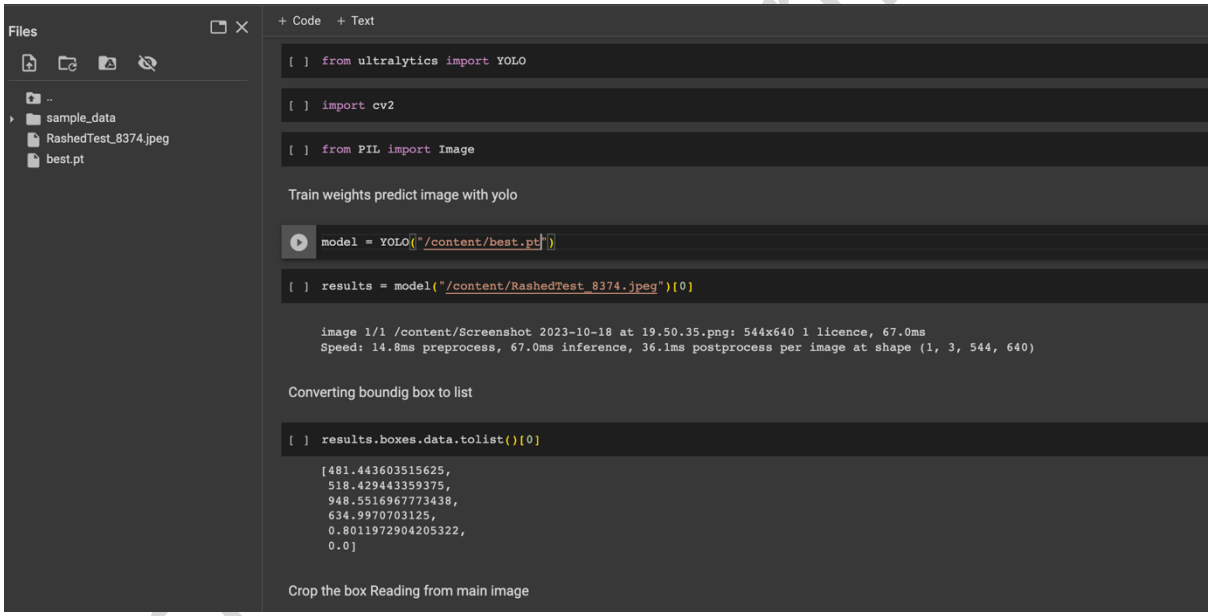

Figure 28 Import Necessary Libraries Train Weights by Predicting Image

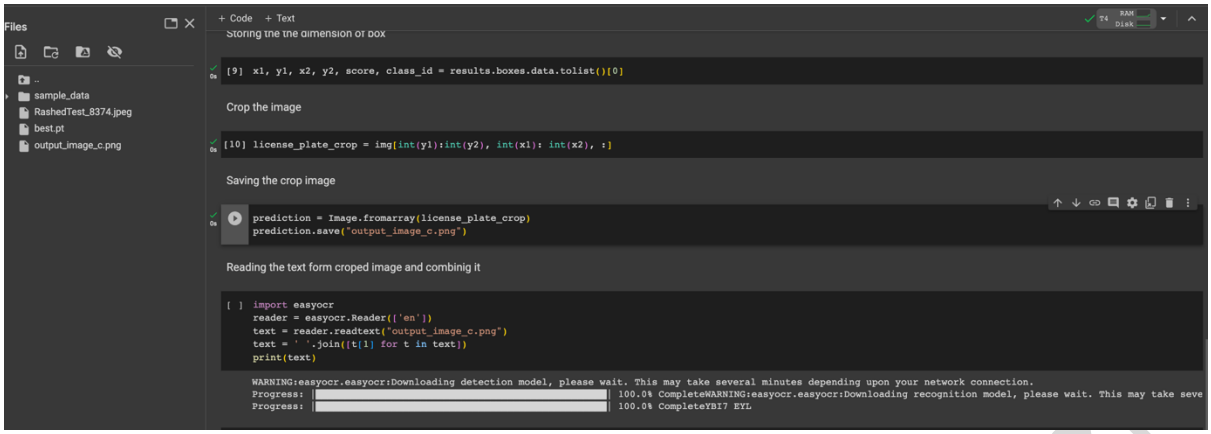

Figure 29 Optimizing Coordinates of Bounding Box

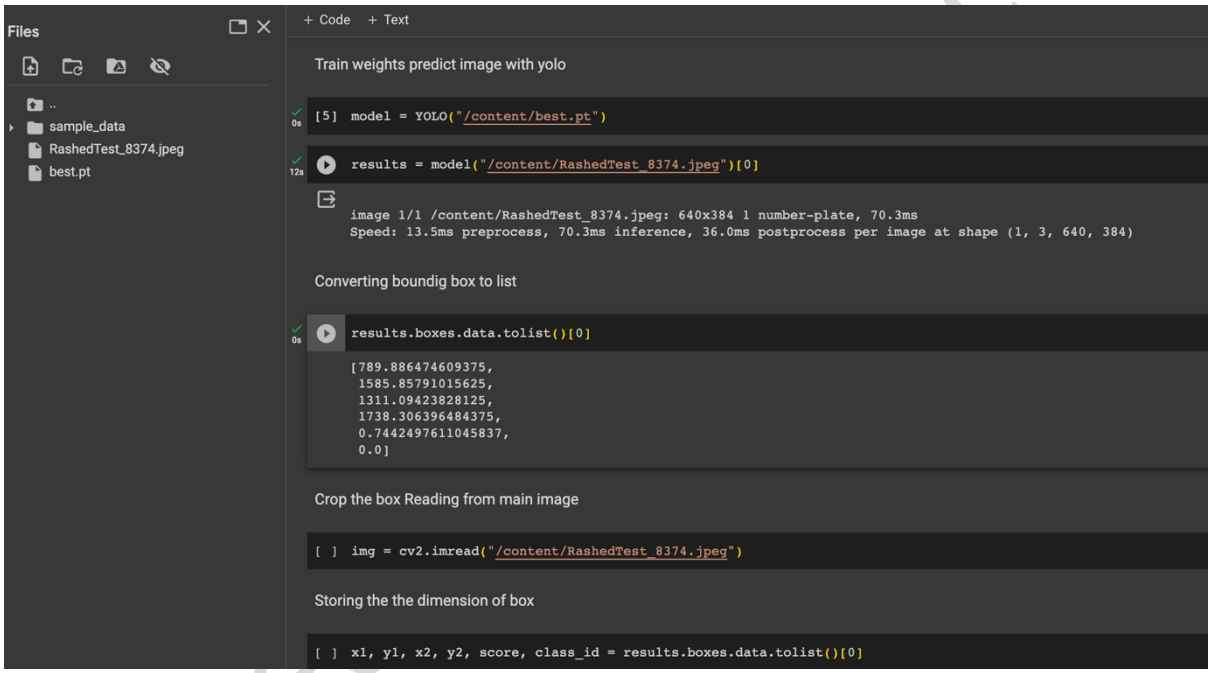

Figure 30 Sorting Bounding Box and Cropping the Image

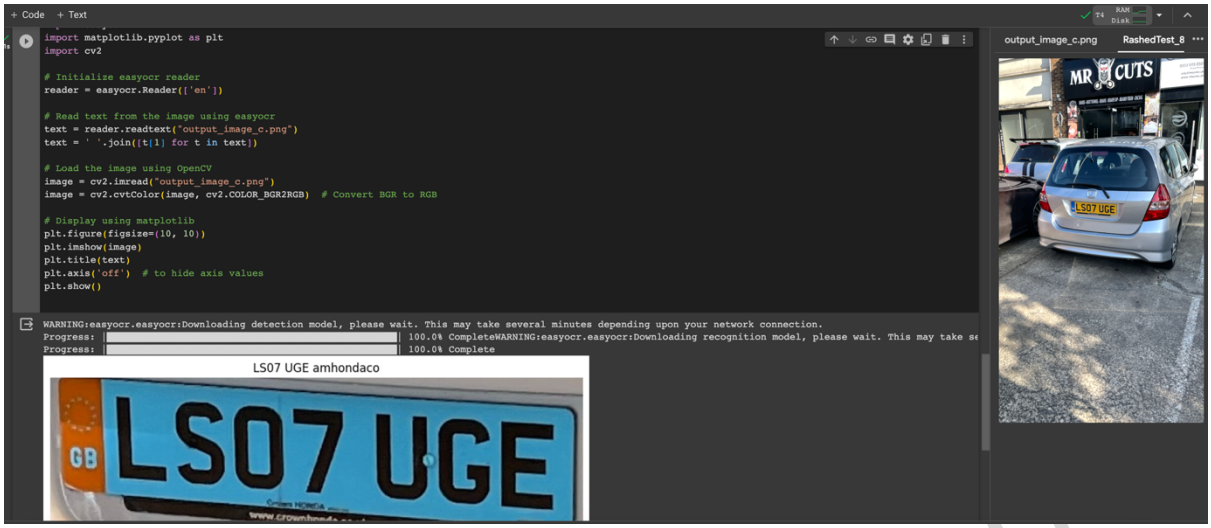

Figure 31 Visualise the Detected Number Plate

7.7 Deployment Preparations:

Before deployment, the model was optimized for real-world scenarios. This involved pruning, quantization, and converting the model to a format amenable to web deployment. The trained model was then integrated with a web application, ensuring that users could easily interact with the ANPR system and obtain results.

7.8 Real-time Detection Attempts:

The initial phase of implementation was marked by a high degree of optimism. There was a strong ambition to incorporate real-time detection and video processing into the ANPR system. However, real-time detection presented its own set of challenges. The lack of freely available GPU resources meant that real-time video processing became computationally intensive, leading to the eventual decision to dismiss this feature. The implementation phase was not devoid of challenges. From algorithmic complexities to hardware limitations, considering each hurdle required after careful consideration the feature has been postpends.

7.9 Model Tuning and Optimization:

The output displays the training progress and performance metrics of a model over 10 epochs. Let's break down the relevant information for model tuning and optimization:

Losses: Box_loss represents the error between the predicted bounding boxes and the ground truth boxes. A decrease in this value indicates the model is getting better at predicting accurate bounding boxes.

Cls_loss: This is the classification loss, indicating the model's error in predicting the correct class for an object. A decrease in this value is good, showing improved classification accuracy.

Dfl_loss: This might represent another specific loss related to the model, possibly a deformation loss or another custom loss. A decrease in this value suggests the model is improving in that specific area.

7.9.2 Trends & Observations:

Losses: Over the epochs, we observe that all three losses (box, cls, dfl) generally decrease, which is a positive sign indicating the model is learning.

Metrics: The mAP50 shows variability across epochs. It starts at 0.0638 in the first epoch, decreases in the fourth, and then starts to rise again, reaching 0.291 by the 10th epoch. This suggests that the model's detection capability is improving, but there may be epochs where it doesn't perform optimally.

7.9.3 Implications for Model Tuning & Optimization:

Learning Rate: If the losses plateau or increase significantly, it might be worth adjusting the learning rate or using learning rate annealing techniques.

Data Augmentation: The variability in mAP might indicate that the model could benefit from a more diverse training dataset. Data augmentation techniques can help the model generalize better.

Regularization: If there's a large gap between training and validation performance (not shown here but typically evaluated), it might indicate overfitting. In such cases, adding regularization techniques like dropout, L2 regularization, or augmenting the dataset can help.

Model Architecture: If performance isn't satisfactory after many epochs, it might be worth considering a change in the model's architecture or trying a different pretrained model.

Early Stopping: Given the variability in mAP, an early stopping mechanism could be beneficial. This would stop the training if the model's performance doesn't improve after a certain number of epochs, preventing potential overfitting and saving computational resources.

Overall, continuous monitoring of these metrics and losses during training provides crucial feedback for tuning and optimizing the model for better performance.

7.10 Performance Metrics:

The training process conducted using YOLOv8 model using the Ultralytics framework. The model is run on a Tesla T4 GPU with 15,102MiB of memory. The training task is license plate detection with a dataset located at /content/ANPR-1/data.yaml and is trained for 10 epochs. The model architecture is displayed, detailing layers and parameters, with a total of 25,856,899 parameters. As training progresses, loss metrics like box loss, class loss, and dfl loss are shown per epoch, as well as performance metrics like precision, recall, and mAP (mean average precision) for both 50% IoU (Intersection over Union) and 50%-95% IoU ranges. After training, the best model is validated, achieving a mAP of 0.293 for 50% IoU and 0.128 for 50%-95% IoU, indicating the model's accuracy in predicting license plates. The inference speed metrics are provided, taking approximately 10.6ms for model inference per image.

Despite the challenges, solutions were always at hand. Through strategic decision-making, continuous refinement, and the adoption of best practices, the ANPR system was successfully implemented, setting a benchmark for future developments in the field.

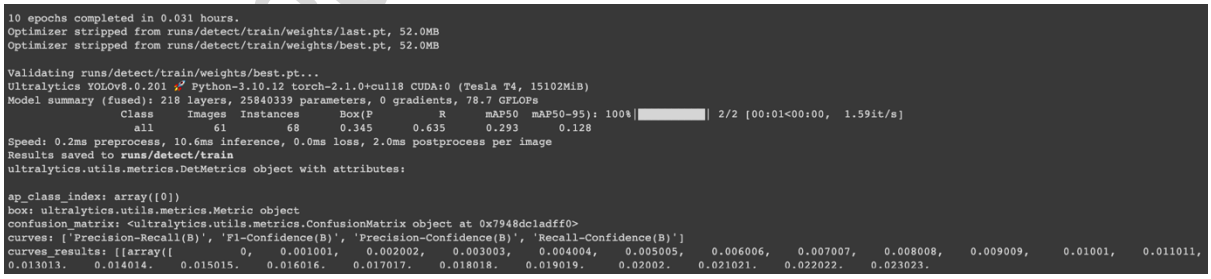

Figure 32 Performance Metrics overview

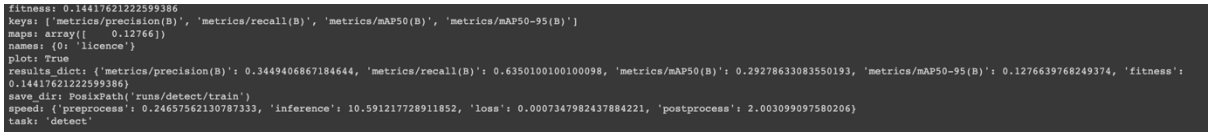

Figure 33 Performance Metrics Result

7.11 Evaluate model's performance:

mAP (mean Average Precision):
For 50% IoU (Intersection over Union), the mAP is 0.293 (or 29.3%).
For IoU ranging from 50% to 95%, the mAP is 0.128 (or 12.8%).

Precision: The model's precision is 0.345 (or 34.5%), which indicates the proportion of predicted license plates that were correct.

Recall: The recall is 0.635 (or 63.5%), representing the proportion of actual license plates that were correctly detected by the model.

mAP: Generally, a higher mAP indicates better model performance. An mAP of 29.3% at 50% IoU suggests that there's significant room for improvement. The mAP of 12.8% for IoU from 50%-95% further emphasizes this.

Precision and Recall: A precision of 34.5% suggests that a notable portion of the detections made by the model might be false positives. However, a recall of 63.5% indicates that the model is relatively better at capturing most license plates in the images, but it still misses out on over a third of them.

Inference Speed: The inference speed is approximately 10.6ms per image, which is relatively fast and indicates the model's efficiency in real-time or near-real-time applications. In summary, while the model is efficient in terms of inference speed, its accuracy (as indicated by mAP, precision, and recall) is modest and suggests there's ample room for improvement. Depending on the application's requirements, further training, data augmentation, fine-tuning, or even a change in model architecture might be necessary to achieve desired performance levels.

8.0 Deployment:

8.1 Deployment Environment and Tools:

The deployment environment and tools play a crucial role in the successful demonstration and functioning of the ANPR system. The ANPR system was developed and tested using a combination of tools and platforms. Flask, a micro web framework written in Python, was employed to create the web application. Google Colab, a cloud-based platform, provided a free GPU for testing enabling the training of deep learning models. Visual Studio Code, a versatile code editor, facilitated the coding, debugging, and local hosting of the application. (Palletsprojects.com, 2023)

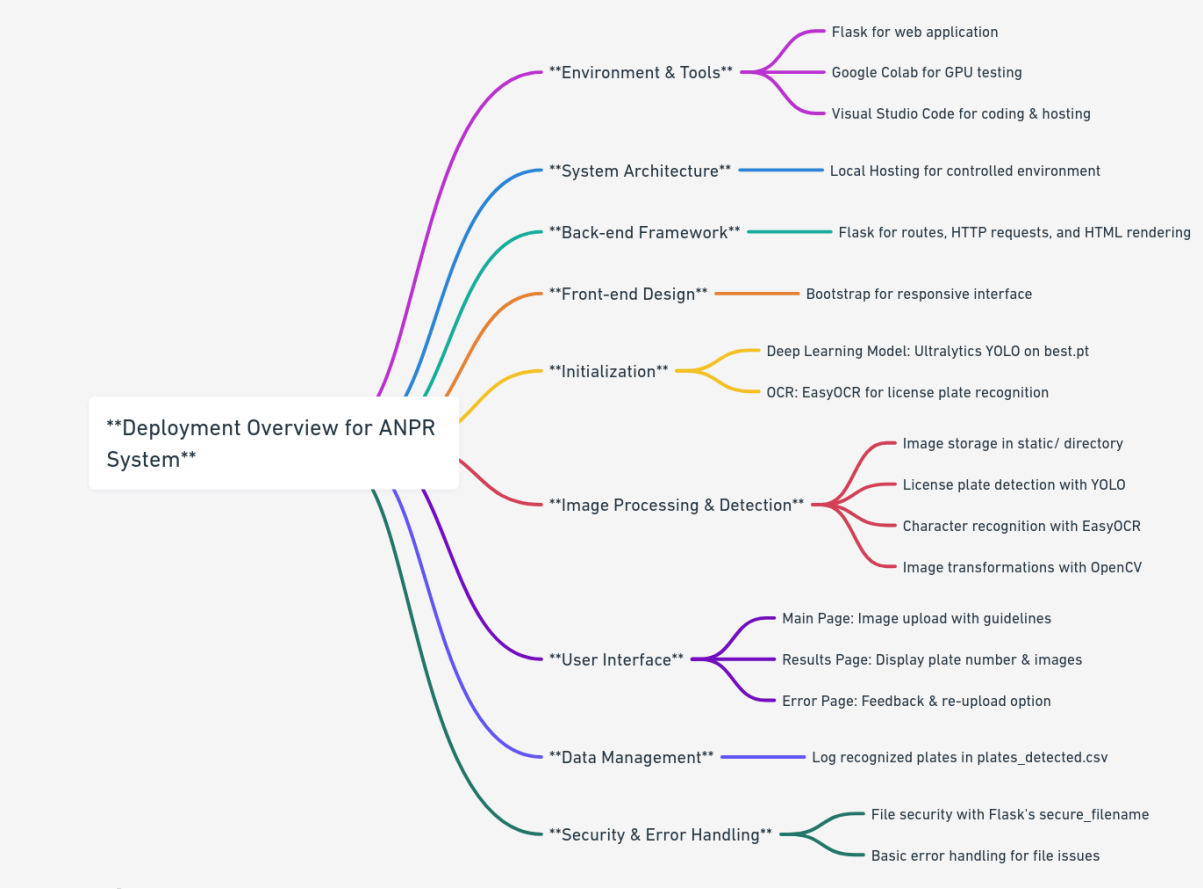

Figure 34 Deployment Mind map

8.2 System Architecture

Given that the project was an individual task for a 30-week undergraduate course, the decision to opt for local hosting was strategic. It allowed for a controlled environment, ensuring that the demonstration was seamless, without the complexities and potential challenges of cloud

deployment. This chapter elucidates the deployment process of the ANPR system on a local server. (Microsoft, 2021)

8.2.1 Back-end Framework:

Flask, a Python-based micro web framework, serves as the backbone of our deployment. It provides the tools to define routes, handle HTTP requests, and render HTML templates.

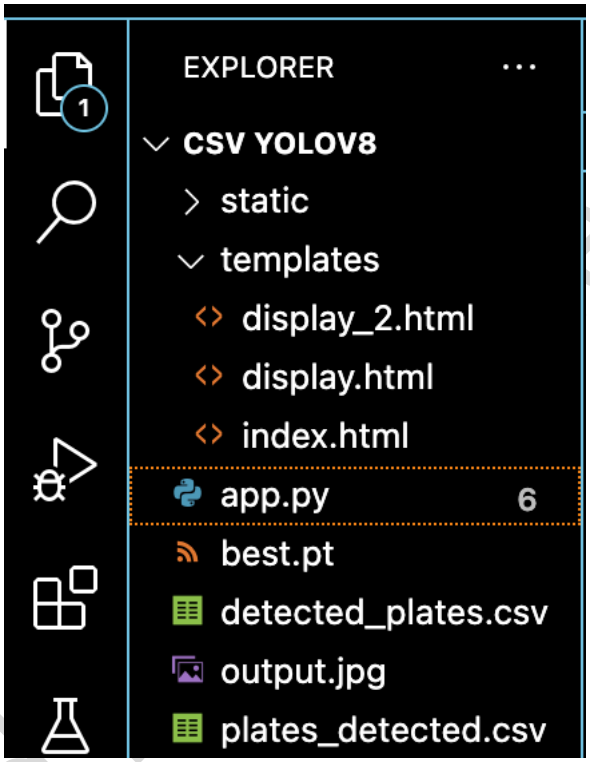

Figure 35 Project folder explorer at a glance

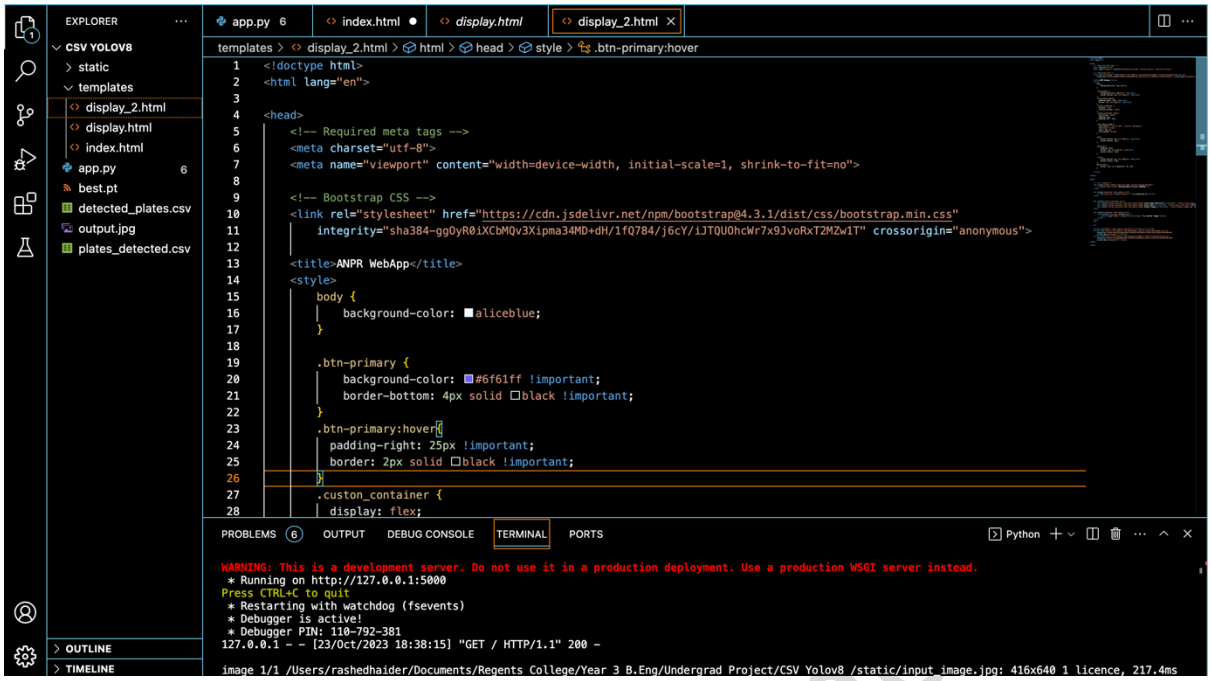

Figure 36 Backend overview

8.2.2 Front-end Design:

Bootstrap, a popular front-end framework, ensures the web application is responsive and visually appealing across browser.

8.3Components of the Deployment

8.3.1 Initialization

Deep Learning Model: The system relies on the ultralytics YOLO model, pretrained on the file best.pt, for detecting license plates in images.

Optical Character Recognition (OCR): Once the license plate region is identified, EasyOCR is employed to read and recognize the characters on the plate. It's initialized for English language recognition.

8.3.2 Image Processing and Plate Detection

The YOLO model identifies potential license plate regions within the image.

Detected regions are cropped and processed using EasyOCR to extract plate numbers.

Additional image transformations, such as grayscale conversion and canny edge detection, are performed using OpenCV (cv2). These transformed images offer users a visual insight into the intermediate steps of the process. (Rong et al., 2014)

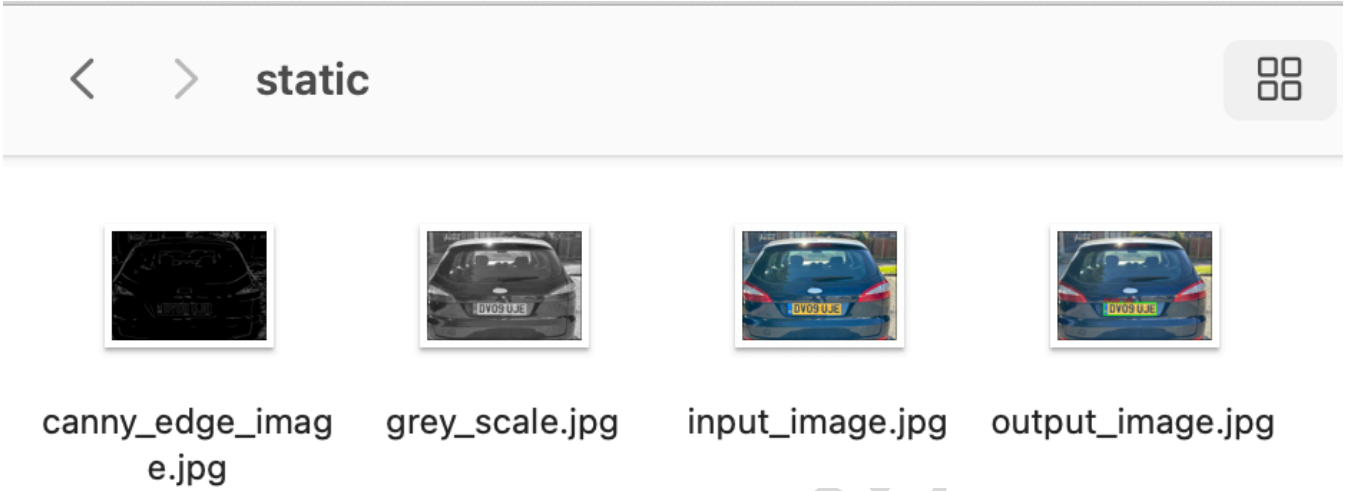

Figure 37 Uploaded images are stored in the static/ directory.

8.3.3. User Interface

Main Page (index.html): Users are presented with an interface to upload their vehicle images. Guidelines about the ideal image size and instructions on using the app are also provided.

Results Page (display_2.html): Upon successful plate detection, users are navigated to a results page displaying the recognized plate number and various processed versions of the uploaded image. They can also opt to upload a new image for recognition.

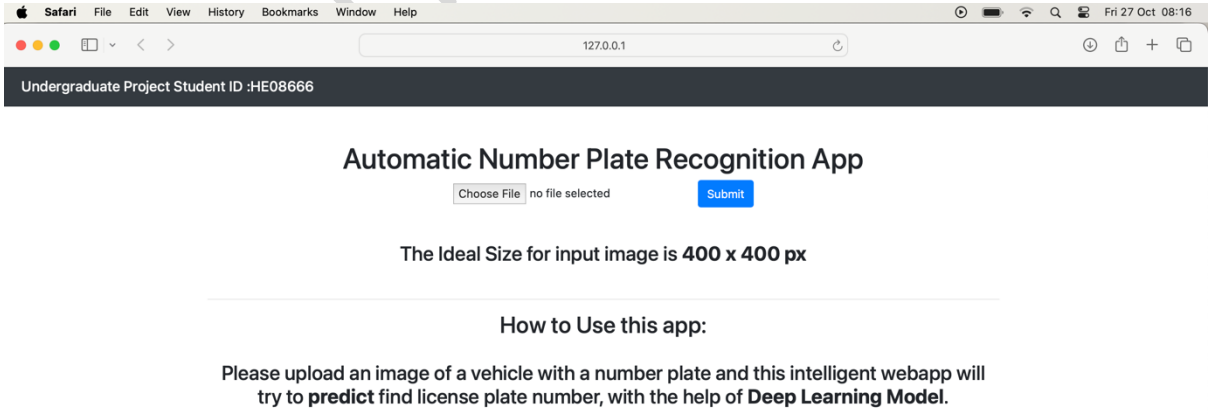

Figure 38 ANPR WebApp Home Page UI

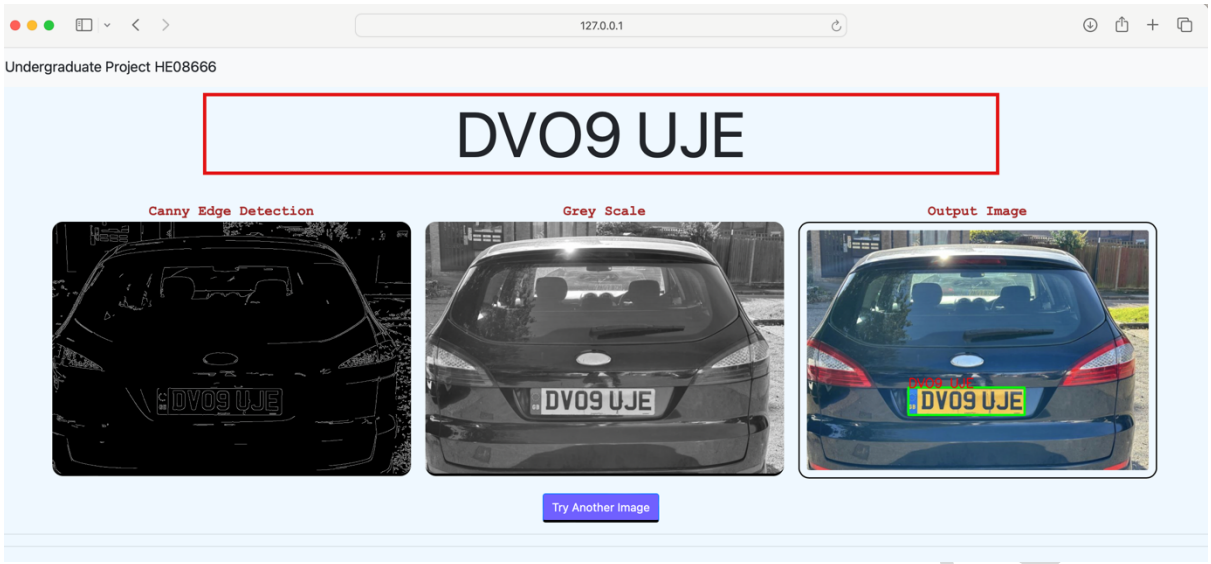

Figure 39 Output Page with accurate detection and reading

Error Page (display.html): If any issues arise during the image upload or processing, users are

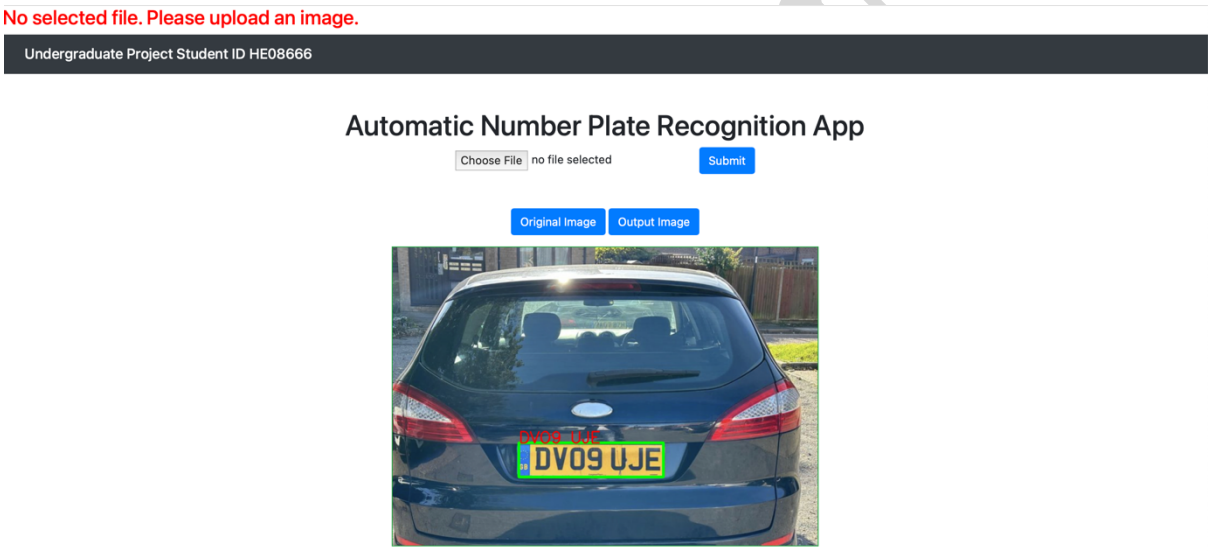

Figure 40 No file selected error page

directed to an error page, which provides feedback and the option to re-upload an image.

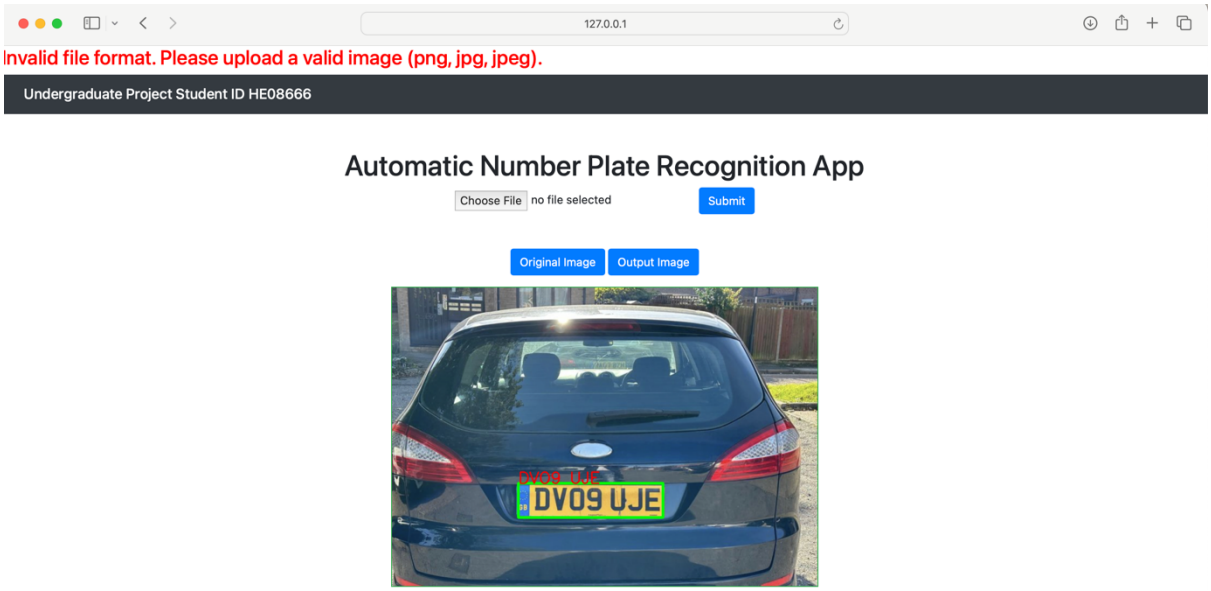

Figure 41 Invalid file error message

8.3.4 Data Management:

The recognized plate numbers, along with their respective timestamps and original file names, are recorded in a CSV file named plates_detected.csv. This approach facilitates easy tracking and logging of processed plates.

```
# Write detected plate to CSV
timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
with open('plates_detected.csv', mode='a', newline='') as file:
    writer = csv.writer(file)
    writer.writerow([plate_num, timestamp, original_filename])

return render_template("display_2.html", plate_num=plate_num)
```

Figure 42 CSV implementation code

plates_detected.csv Open with Numbers

OZOOON6	2023-10-23 13:59:23	Screenshot 2023-10-23 at 13.59.11.png
G8060 @	2023-10-23 14:04:20	Screenshot 2023-10-23 at 14.04.00.png
6Y68 0ND	2023-10-23 14:16:07	RashedTest_1394.jpeg
LR33 TEE	2023-10-23 14:17:32	example_input_1.jpg
RA69 LTU	2023-10-23 14:17:53	RashedTest3.png
AEI7 XPV	2023-10-23 14:18:15	Screenshot 2023-10-18 at 21.57.51.png
CK22 LYO	2023-10-23 14:18:36	Test3.png
M003 MLB	2023-10-23 14:19:04	Screenshot 2023-10-18 at 19.47.06.png
LS07 UGE	2023-10-23 14:19:23	RashedTest_8374.jpeg
FY6B	2023-10-23 14:21:05	output_image copy.png
OZOOON6	2023-10-23 14:21:24	Screenshot 2023-10-23 at 13.59.11.png
6Y68 0ND	2023-10-23 14:23:11	RashedTest_1394.jpeg
MH 20EE 7598	2023-10-23 14:23:39	Cars111.png
LS07 UGE	2023-10-23 14:44:18	RashedTest_8374.jpeg
JA6Z UAR	2023-10-23 18:38:27	Cars118.png
OPEC LOL	2023-10-23 18:45:40	Cars129.png
YP57 CZH	2023-10-25 16:03:49	RashedTest2.png
YP57 CZH	2023-10-25 16:05:35	RashedTest2.png
6Y68 0ND	2023-10-25 16:06:34	RashedTest_1394.jpeg
KY6O AKF	2023-10-25 16:07:11	RashedTest4.jpeg
LOR S16K	2023-10-25 16:07:55	example_input_5.png
AP68 LXG	2023-10-27 08:17:32	Screenshot 2023-10-18 at 19.41.57.png
B2228HM	2023-10-27 08:20:08	Cars120.png
MK52 POP	2023-10-27 08:20:58	Screenshot 2023-10-18 at 19.45.55.png
MK52 POP	2023-10-27 08:21:23	Screenshot 2023-10-18 at 19.45.55.png
MK52 POP	2023-10-27 08:21:38	Screenshot 2023-10-18 at 19.45.55.png
MHO14V8866=	2023-10-27 08:22:16	Cars108.png
MK52 POP	2023-10-27 08:22:53	Screenshot 2023-10-18 at 19.45.55.png
DVO9 UJE	2023-10-27 08:24:19	Screenshot 2023-10-18 at 19.44.33.png

Figure 43 CSV output result

8.3.5 Security and Error Handling:

File security is a paramount concern when accepting user uploads. The secure_filename function from Flask's werkzeug utility ensures that uploaded files have safe names, thereby

mitigating potential security risks. Basic error handling mechanisms are in place to manage issues such as invalid file formats or missing files. For a more robust system, comprehensive error handling should be integrated to manage potential challenges during model inference or OCR processing.

```
def uploader():
    if request.method == 'POST':
        if 'file1' not in request.files:
            return render_template("display.html", error="No file uploaded. Please upload an image.")

        f = request.files['file1']
        original_filename = f.filename
        if original_filename == '':
            return render_template("display.html", error="No selected file. Please upload an image.")

        allowed_extensions = {'png', 'jpg', 'jpeg', 'gif'}
        if not ('.' in original_filename and original_filename.rsplit('.', 1)[1].lower() in allowed_extensions):
            return render_template("display.html", error="Invalid file format. Please upload a valid image (png, jpg, jpeg).

        f.filename = "input_image.jpg"
        f.save(os.path.join(app.config['UPLOAD_FOLDER'], secure_filename(f.filename)))
        image_path = "static/input_image.jpg"
        img = cv2.imread(image_path)
        results = model(image_path)[0]

        if len(results.boxes.data.tolist()) == 0:
            return render_template("display.html", error="Unable to detect a number plate in the uploaded image. Check the C

        x1, y1, x2, y2, score, class_id = results.boxes.data.tolist()[0]
        license_plate_crop = img[int(y1):int(y2), int(x1): int(x2), :]
        plate_num = recognize_plate_easyocr(img=license_plate_crop, reader=EASY_OCR, region_threshold=OCR_TH)
```

Figure 44 Error Handling Codes

8.4 Recommendations for Production Deployment

Server Configuration: While Flask's development server suffices for local deployment, a production setting would benefit from a robust server such as Gunicorn or uWSGI, ideally behind a reverse proxy like Nginx. As the application scales, a structured database system would offer more efficient and scalable storage compared to a CSV file. Additional security measures, such as input validation and protection against potential threats like SQL injections or Cross-site Scripting (XSS), should be implemented.

9.0 Performance Testing:

Performance testing is a critical phase in the development of any system, including the ANPR system. Test driven era make sure the test is done at every possible stage. It ensures that the system meets the desired standards and functions optimally under various conditions. The primary objective of performance testing for the ANPR system was to validate its efficiency, detection, and responsiveness. Success was defined based on specific criteria and benchmarks. These included achieving a certain accuracy rate in license plate recognition, maintaining a consistent response time, and ensuring the system's stability under different loads.

9.1 Dataset and Testing Environment:

Dataset Description:

The dataset used for testing was carefully selected to represent a wide range of scenarios. It comprised images of license plates from various regions, captured under different lighting conditions and angles total images has been curtailed to 300.

Testing Environment:

The testing environment was meticulously set up to replicate real-world conditions. This included specific hardware configurations, such as CPU and GPU specifications, and software setups, ensuring that the tests were both rigorous and relevant.

9.2 Training Model Testing Matrices:

Several testing methodologies were employed to gauge the ANPR system's performance comprehensively. Stress testing evaluated the system's resilience under extreme conditions, , and latency testing measured the system's response times.

Precision-Recall Curve (PR-curve):

The PR curve is a graphical representation of a classifier's performance across different threshold values. It's particularly useful for imbalanced datasets.

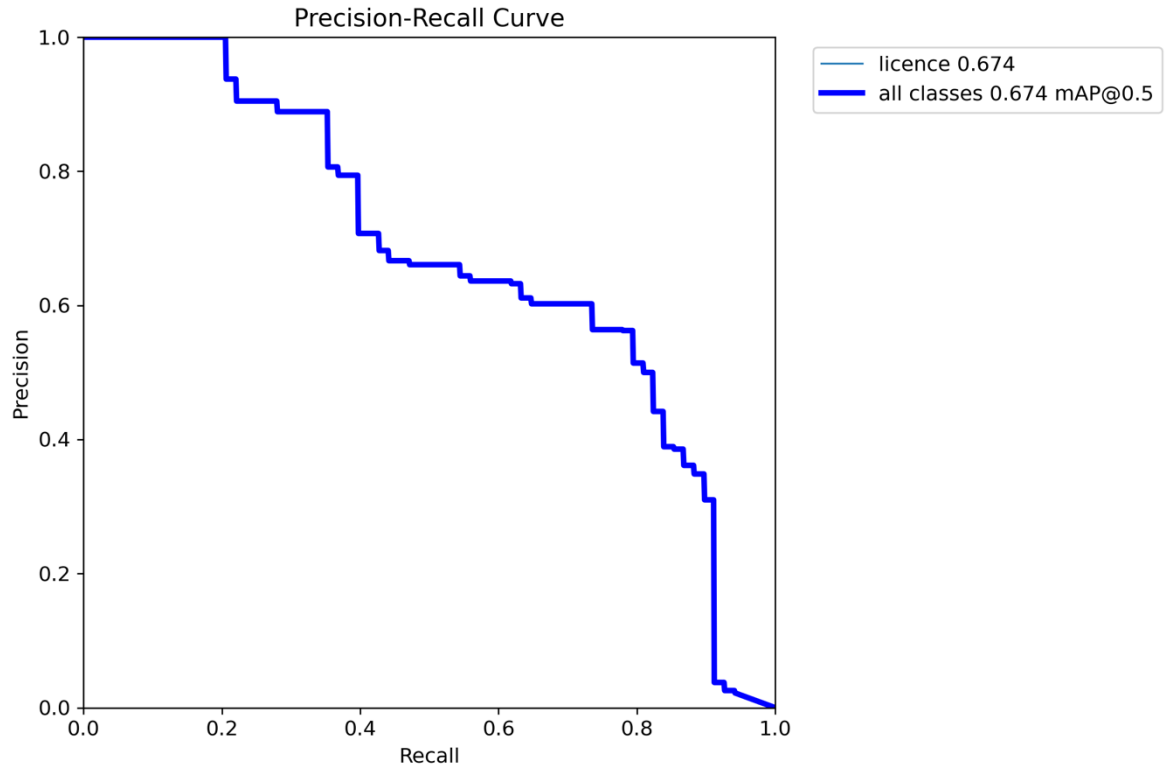

Figure 45 PR Curve

In the context of ANPR, the balance between precision and recall is crucial. The ideal model should have both high precision (to avoid false positives) and high recall (to capture all characters). Analysing the curve can help in selecting the right threshold for the model to operate at its best in real-world scenarios.

Precision Curve (P-curve):

The precision curve showcases the ability of the classifier to not label a negative sample as positive. In the ANPR context, it represents the model's ability to correctly identify license plate characters without falsely recognizing non-characters or other noise.

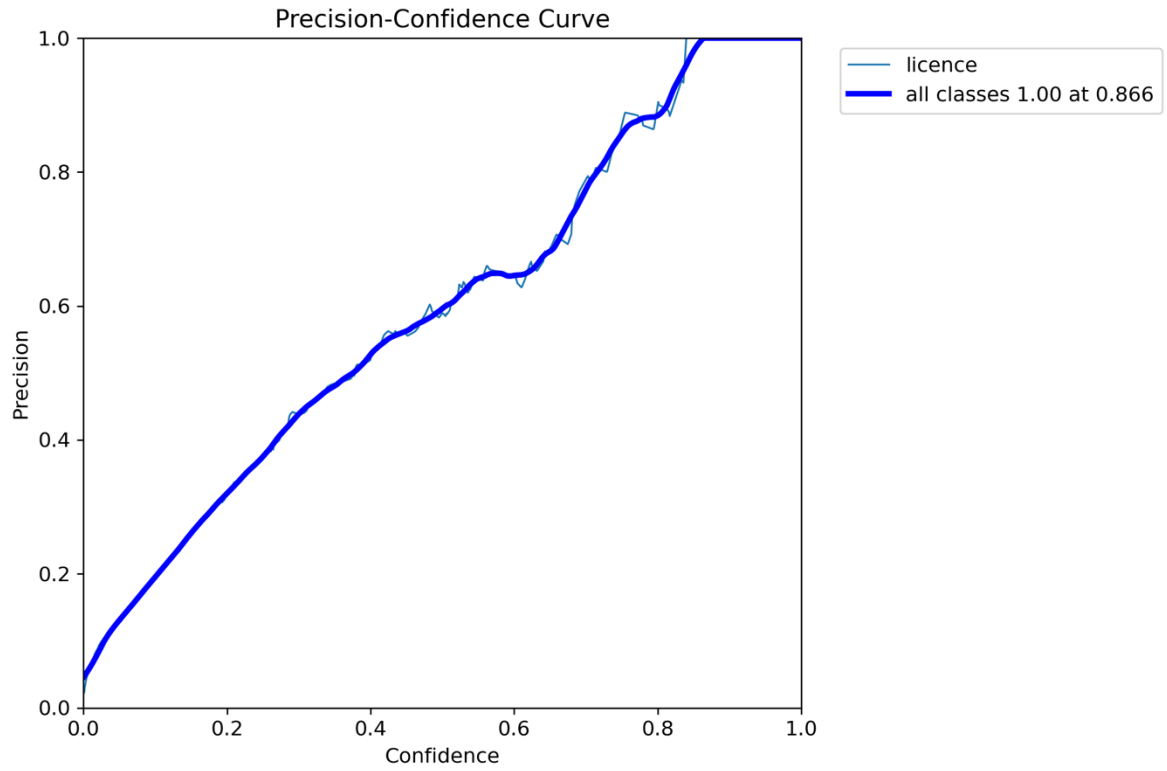

Figure 46 P Curve

Peaks and valleys in the curve can provide insights into thresholds where the model's precision fluctuates. For an ANPR system, maintaining high precision is critical to ensure accurate reads of license plates.

F1 Curve:

The F1 score is the harmonic mean of precision and recall, providing a balance between the two. A model with a high F1 score is considered to have a good balance between precision and recall.

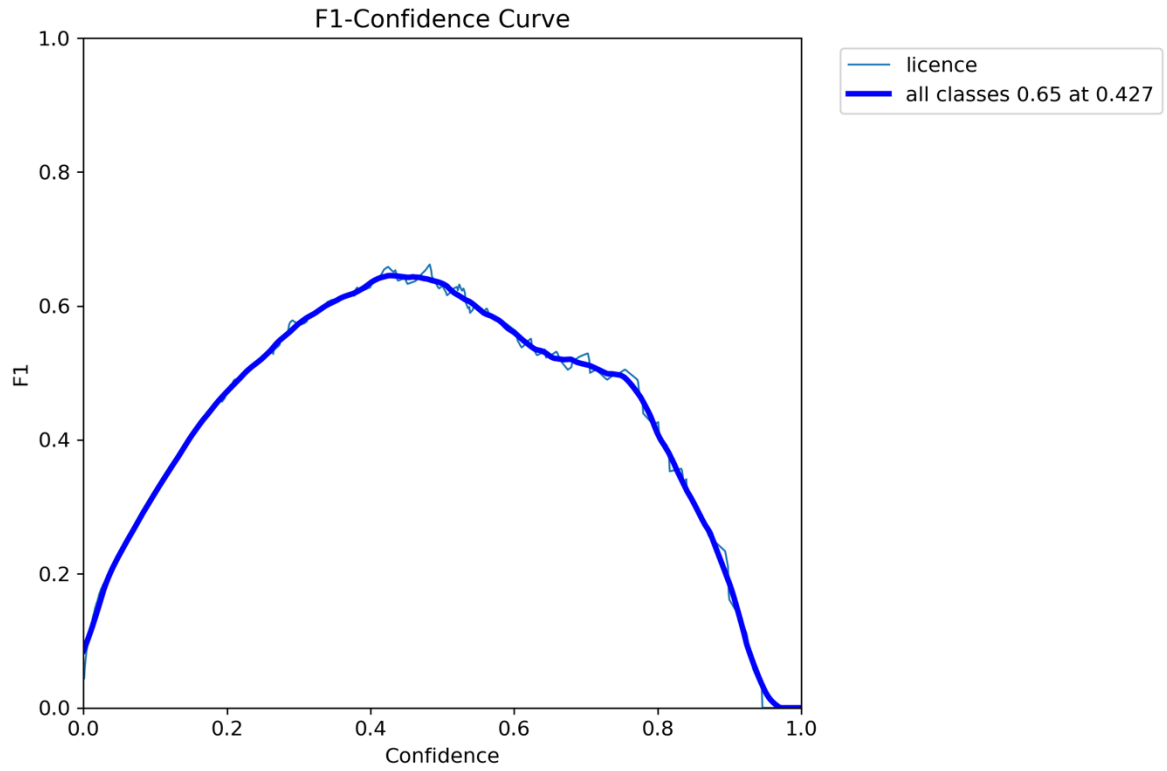

Figure 47 F1 Curve

Observing the trend of the F1 curve can provide insights into how well the model is balancing false positives and false negatives. Any sharp drops or spikes can be areas of concern and warrant further investigation.

Confusion Matrix:

This matrix provides the raw counts of correct and incorrect predictions for each class. It gives a clearer picture of the magnitude of errors and can be more impactful when discussing performance in terms of absolute numbers.

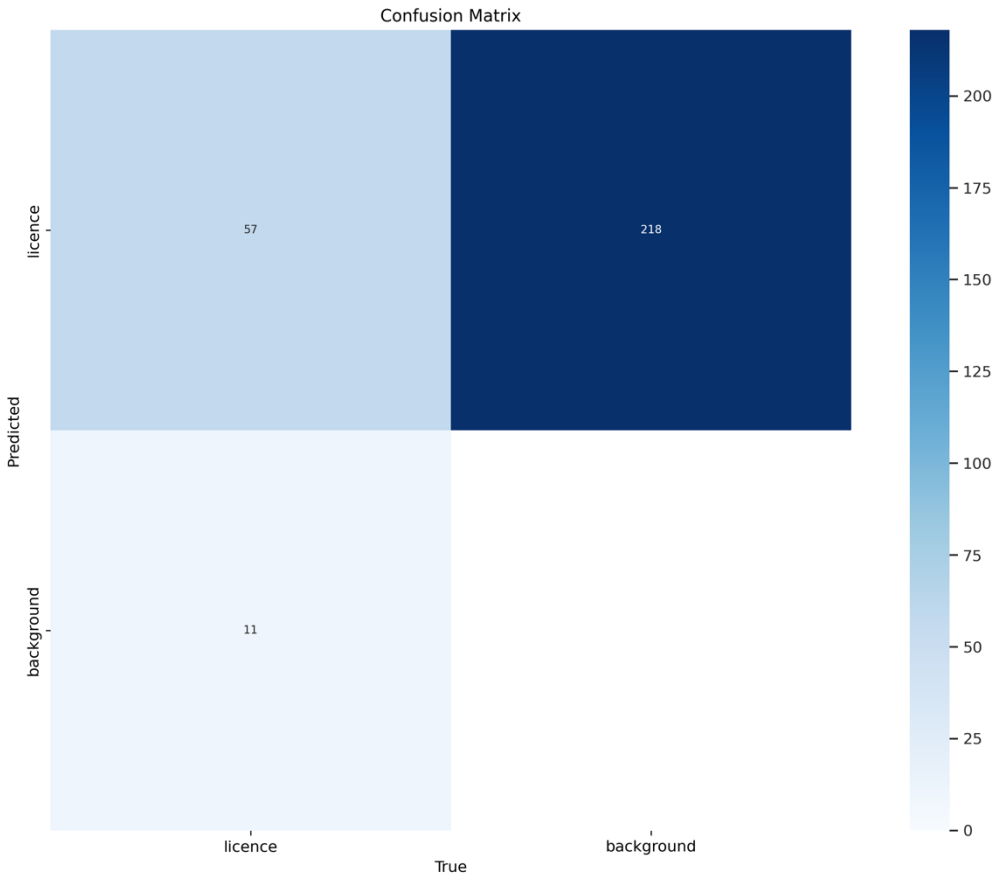

Figure 48 Confusion Matrix

As with the normalized confusion matrix, the focus should be on any large off-diagonal values which indicate a high number of misclassifications.

Comparing the two confusion matrices (normalized and standard) can give insights into the relative and absolute performance of the model.

Confusion Matrix (Normalized)

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized, often with respect to each class.

The matrix provides insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

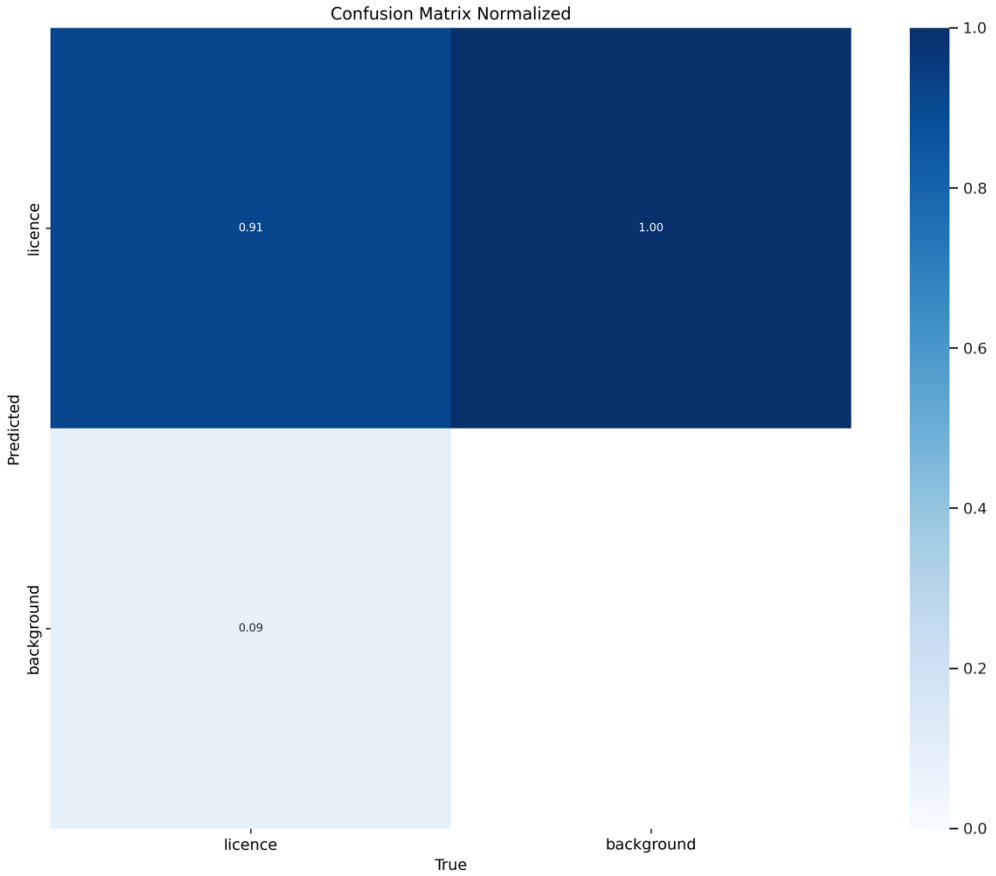

Figure 49 Confusion Matrix Normalized

In the context of the ANPR model, this matrix would show how often each label was predicted correctly versus incorrectly. The diagonal elements represent the number of points for which the predicted label is equal to the true label, while off-diagonal elements are those that are mislabelled by the classifier.

Critical Analysis:

The darker squares on the diagonal indicate that the model has a relatively high accuracy for those particular classes. However, any off-diagonal coloration would indicate misclassifications.

A deep analysis would involve discussing which classes have the most misclassifications and hypothesizing why this might be the case. For instance, are there similarities between certain license plates that lead to confusion?

Labels Distribution

This visualization illustrates the frequency of each label (number or character) in the dataset.

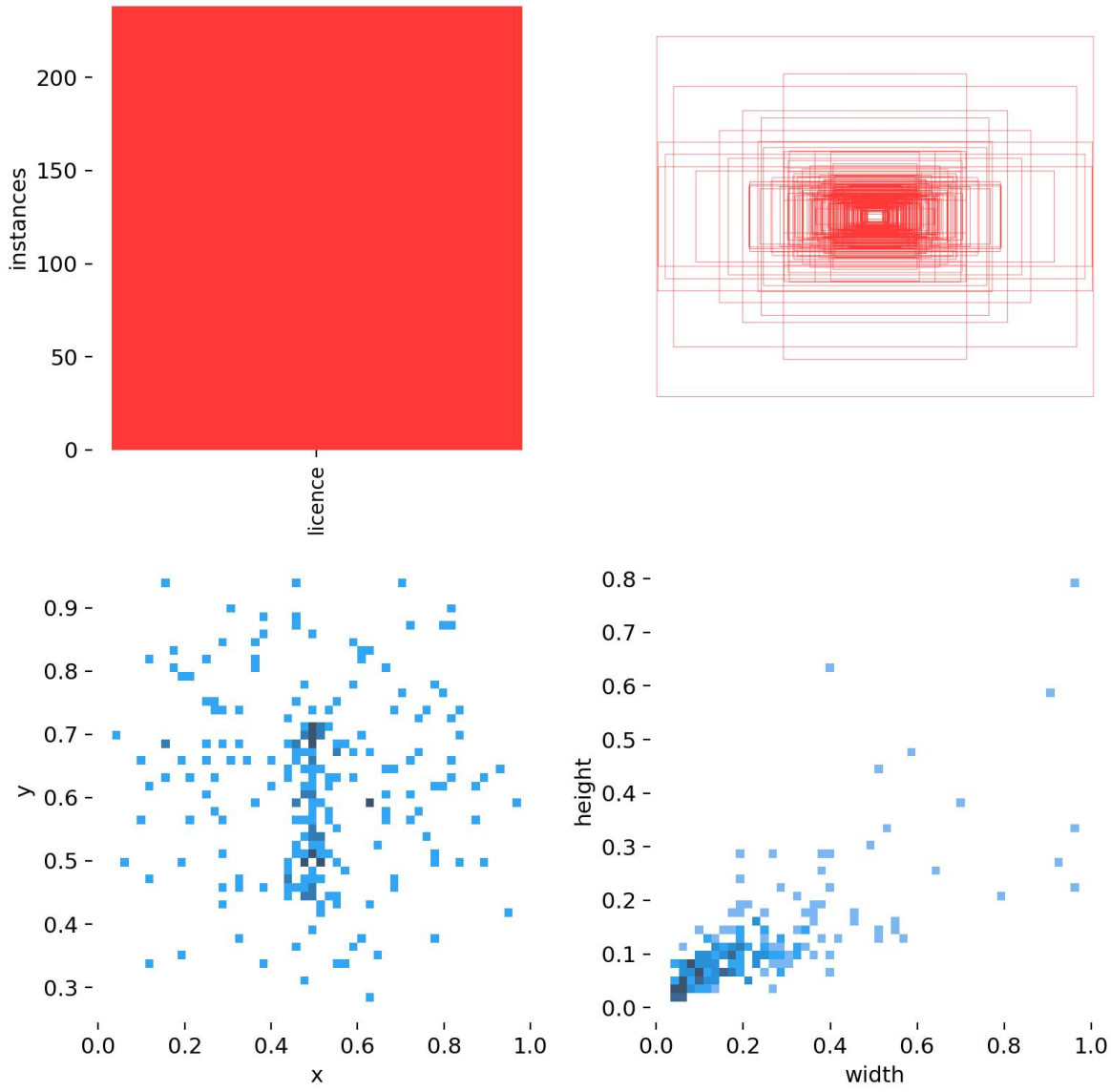

Figure 50 Labels Distribution

Uneven distribution might indicate potential biases in the model. For instance, if certain characters appear much more frequently than others, the model might be over-optimized for those and underperform on rarer characters. Understanding this distribution is crucial for

ensuring the model performs well across a variety of license plates, not just the most common ones.

Labels Correlograms:

Correlograms or heatmaps of labels typically showcase the co-occurrence or relationships between labels. In the context of ANPR, this might showcase how often certain characters or numbers appear together.

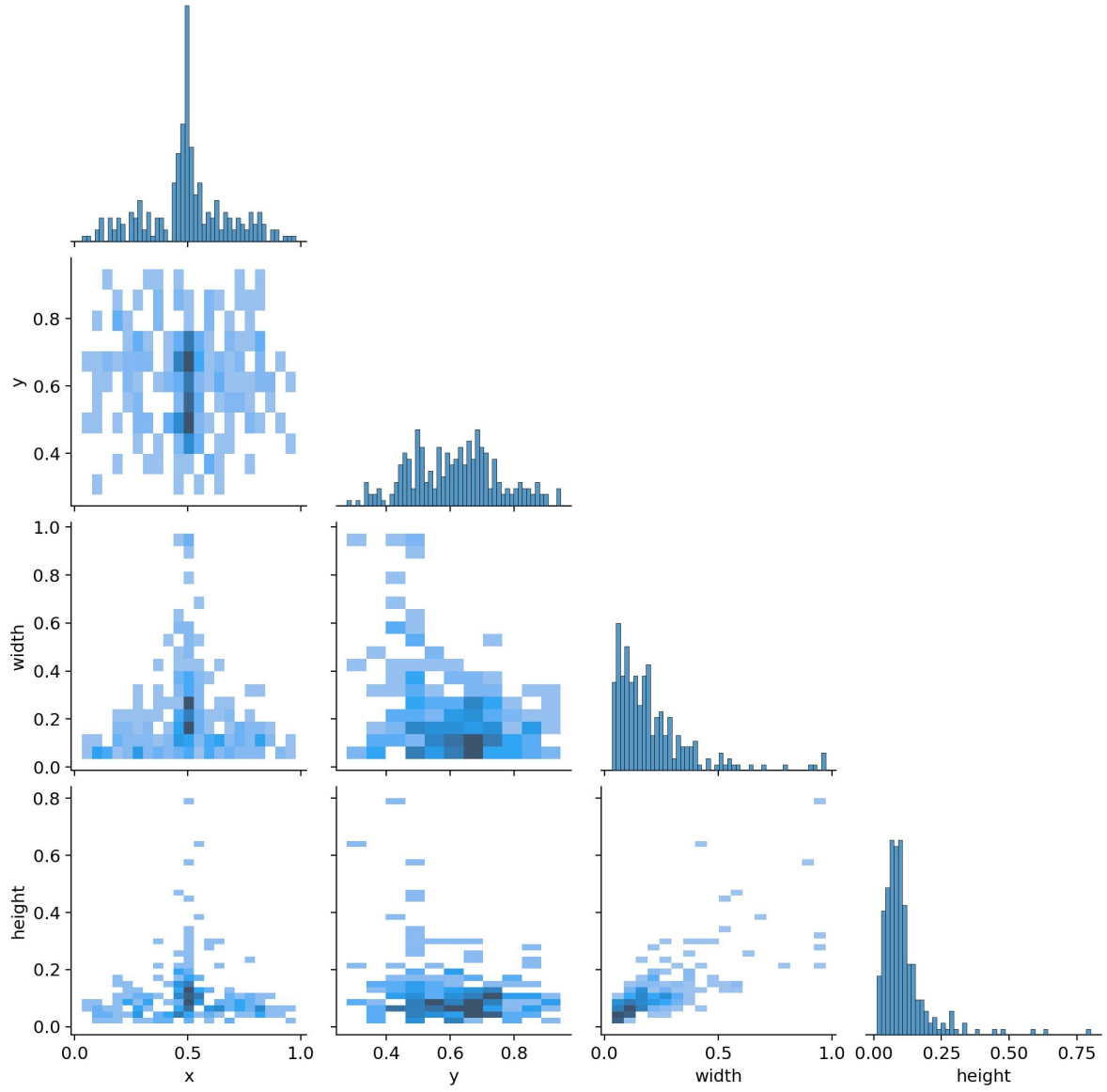

Figure 51 Labels Correlogram

For an ANPR system, certain patterns might emerge based on regional license plate standards or common number/letter combinations. Recognizing these patterns can aid in optimizing the model further. If there are unexpected correlations, it could indicate an issue with data collection or labelling.

9.3 Testing UI and Backend:

both User Interface (UI) and backend testing are indispensable. UI testing ensures that the visual elements, such as buttons, input fields, and display panels, function seamlessly and provide an intuitive user experience. It checks the application's responsiveness, layout consistency across different devices and screen sizes, and the efficacy of error messages or prompts. On the other hand, backend testing concentrates into the application's core, assessing component algorithms, the efficiency of data processing, and the robustness of database operations. It validates if the system can handle varying loads, manages data securely, and integrates well with other systems or APIs. Together, UI and backend testing ensure that both the surface and the underlying mechanisms of the ANPR application are reliable, efficient, and user-friendly, guaranteeing an optimal end-to-end user experience.

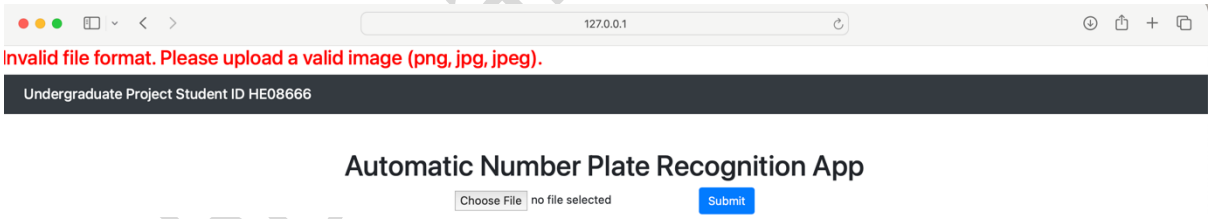

Figure 52 Invalid File Format Testing

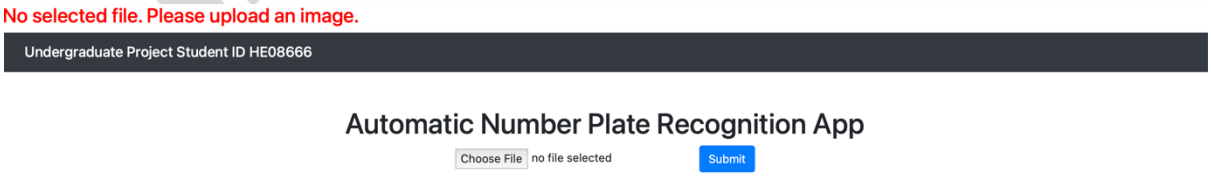

Figure 53 No File Found Test

On certain testing occasion results were rather ominous the example below unable to detect the full number plate as the model generalise the bounding box size.

Figure 54 Numberplate out of bounding box

In the following screenshot we observed that recognition were nowhere what expected.

Figure 55 Wrong Detection of Numberplate

9.4 Results and Analysis:

Presentation of Results:

The results of the performance tests were systematically documented, presenting metrics. These metrics provided a quantitative measure of the system's performance in testing chapter

Analysis:

An in-depth analysis of the results was conducted in a tabular format to understand the system's strengths and weaknesses. This analysis highlighted areas where the ANPR system excelled and pinpointed areas that required further refinement or optimization.

Epoch	GPU Memory	Box Loss	Class Loss	DFL Loss	Instances	Image Size	Precision (Box(P))	Recall (R)	mAP @ 50% IoU	mAP @ 50-95% IoU
1/10	6.85G	1.694	4.502	1.649	17	640	0.149	0.132	0.0638	0.0323
2/10	7.08G	1.595	2.189	1.424	18	640	0.129	0.338	0.128	0.0453
3/10	7.1G	1.655	1.84	1.456	20	640	0.132	0.485	0.116	0.0467
4/10	7.07G	1.684	1.576	1.543	17	640	0.00145	0.324	0.000919	0.000354
5/10	7.11G	1.663	1.547	1.554	22	640	0.00116	0.132	0.000774	0.000293
6/10	7.11G	1.614	1.303	1.504	16	640	0.00513	0.471	0.00383	0.00151
7/10	7.07G	1.572	1.25	1.463	19	640	0.0128	0.191	0.00703	0.00285
8/10	7.1G	1.566	1.2	1.453	16	640	0.487	0.294	0.229	0.101
9/10	7.07G	1.539	1.154	1.489	18	640	0.108	0.676	0.0864	0.0374
10/10	7.11G	1.457	1.12	1.425	20	640	0.329	0.613	0.291	0.127

9.4 Challenges and Limitations:

Challenges Encountered:

During performance testing, several challenges were encountered. These ranged from computational limitations with larger datasets, which affected real-time processing speeds, to biases in the dataset, which might have influenced the detection rates. Small text which is not part of the numberplate was also another challenge that needed to be resolved in the testing phase. This issue addressed in the deployment stage by limiting the threshold in OCR result.

Figure 56 System Reading Other Text in Numberplate

RESTART RUNTIME

```
loading Roboflow workspace...
loading Roboflow project...
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-3-c6fd90955d90> in <cell line: 6>()
      4 rf = Roboflow(api_key="W5WcFB2FvjAQ4LLPrifb")
      5 project = rf.workspace("university-of-bolton").project("anpr-xktyr")
----> 6 dataset = project.version(1).download("yolov8")

                                 2 frames
/usr/local/lib/python3.10/dist-packages/roboflow/util/versions.py in
get_wrong_dependencies_versions(dependencies_versions)
     29     for dependency, order, version in dependencies_versions:
     30         module = import_module(dependency)
---> 31         module_version = module.__version__
     32         if order not in order_funcs:
     33             raise ValueError(

AttributeError: module 'ultralytics' has no attribute '__version__'
```

Figure 57 Error During Training

Limitations Impacting Results:

Certain constraints might have influenced the testing results. For instance, the absence of a dedicated GPU could have impacted real-time detection speeds. Additionally, the dataset's diversity, or lack thereof, could have skewed the rates in favour of certain license plate designs over others.

10.0 Conclusion and Recommendation:

The local deployment of the ANPR system offers a tangible demonstration of the capabilities of Deep Learning in real-world applications. Through Flask, users can seamlessly interact with the model and gain insights into the license plate recognition process. Future enhancements should focus on optimizing the system for production environments, ensuring scalability, security, and efficiency.

10.1 Challenges and Future Directions

While ANPR is quite precise at recognising standard licence plates in optimal conditions, it needs help with distorted or dirty plates, leading to errors in data capture. Deep learning requires huge computational power and real time deployment performance depends upon how robust the server is. Leveraging unlabelled data or self-generated labels to improve ANPR models can reduce the need for extensive labelled datasets. Integration of ANPR with other smart city systems like traffic management and parking solutions. There's potential in researching models optimized for these integrated applications.

10.1.2 Integration with Neural Architectures Search (NAS):

NAS automates the design of machine learning models. Scholars are searching for its potential to tailor models particularly for ANPR.

10.1.3 Transformer-based Models:

Transformers have primarily been used for natural language processing; they are increasingly being applied to vision tasks. Their application in ANPR can potentially enhance accuracy. Deep learning models can use contextual information (e.g., vehicle type, time of day, location) to aid in number plate recognition potentially reduce false positives and increase overall accuracy.

10.1.4 Robustness to Adversarial Attacks:

As ANPR systems are deployed in security-critical environments, they need to be robust against adversarial attacks where malicious agents try to deceive the system. Research on

designing security aware deep learning models that can detect and counteract adversarial attacks on ANPR systems is gaining momentum.

10.1.5 Real-time Processing and Edge Deployment:

Deploying ANPR on edge devices (like surveillance cameras) requires models to be lightweight yet effective. Model quantization, pruning, and distillation are techniques under examination to achieve this.

10.1.6 Multilingual and Multi-format Number Plate Recognition:

As vehicular movement becomes more globalized, ANPR systems that can recognize number plates from various countries and in different scripts (e.g., Arabic, Bengali) are essential since vehicles crossing the border is a common scenario.

10.1.7 Incorporation of Temporal Information:

Using recurrent neural networks or Long Short-Term Memory (LSTM) cells to process video data and integrate temporal information can improve ANPR accuracy, especially in blurry or low-resolution captures.

10.1.8 Enhanced Training Data Diversity:

ANPR models can occasionally struggle with plates that are dirty, damaged, or obscured. Generating synthetic data using GANs (Generative Adversarial Networks) or augmenting training data to include these challenging scenarios can make models more robust.

10.1.9 Fusion of Different Data Sources:

Combining traditional camera-based data with other sources, such as infrared or LIDAR, might improve the robustness and precision of ANPR systems, especially in challenging conditions like low light or fog.

10.1.10 Privacy Concerns and Solutions:

As ANPR systems become widespread, there are growing concerns about privacy. Research into methods that can obscure non-essential data or that use differential privacy techniques to train models without compromising individual data can address these red flags.

10.2 Personal Reflection:

Developing the ANPR system was a challenging yet rewarding work. As I navigated the complexities of the AI revolution, I quickly realized the importance of tool and platform selection. Dataset selection posed initial challenges, teaching me that in AI, precision often outweighs scale. The intricacies of image processing, especially with the YOLOv8 medium model and EasyOCR integration, highlighted the meticulous nature of this field.

Adaptability became the focal point. Initial ambitions, like real-time object tracking, had to be set aside due to resource constraints, emphasizing the need for agility in approach. The deployment phase, from integrating the model using Flask to designing a user-friendly interface, was a testament to the balance between technical rigor and user experience.

The project also carried personal significance. The passing of my father, Mr. Shamsul Haider, in 2023 was a poignant backdrop. His belief in my potential became a silent driving force behind each milestone.

In essence, this journey was not just about creating an ANPR system but about personal growth, understanding AI complexities, and honouring a personal legacy. It has set a foundation for my continued exploration in the ever-evolving tech landscape.

Reference:

1. Ahmed Fawzy Gad (2018). Practical computer vision applications using deep learning with CNNs : with detailed examples in Python using TensorFlow and Kivy. Berkeley, California: Apress.
2. Ammar, A., Anis Koubâa, Wadii Boulila, Bilel Benjdira and Yasser Alhabashi (2023). A Multi-Stage Deep-Learning-Based Vehicle and License Plate Recognition System with Real-Time Edge Inference. Sensors, [online] 23(4), pp.2120–2120. doi:https://doi.org/10.3390/s23042120.
3. An, Q., Wang, H. and Chen, X. (2022). *EPSDNet: Efficient Campus Parking Space Detection via Convolutional Neural Networks and Vehicle Image...* [online] ResearchGate. Available at: https://www.researchgate.net/publication/366336527_EPSDNet_Efficient_Campus_Parking_Space_Detection_via_Convolutional_Neural_Networks_and_Vehicle_Image_Recognition_for_Intelligent_Human-Computer_Interactions [Accessed 27 Oct. 2023].
4. Baek, Y., Lee, B., Han, D., Yun, S. and Lee, H. (2019). *Character Region Awareness for Text Detection*. [online] arXiv.org. Available at: https://arxiv.org/abs/1904.01941 [Accessed 27 Oct. 2023].
5. Bolton (2023). ANPR Object Detection Dataset (v1, 2023-10-10 4:29pm) by University of Bolton. [online] Roboflow. Available at: https://universe.roboflow.com/university-of-bolton/anpr-xktyr/dataset/1 [Accessed 27 Oct. 2023].
6. Carlson, J., Bryan, T. and Dell, M. (2023). Efficient OCR for Building a Diverse Digital History. [online] arXiv.org. Available at: https://arxiv.org/abs/2304.02737 [Accessed 27 Oct. 2023].
7. Cohn, M. and Beck, K. (2011). User stories applied : for agile software development. Boston Etc.: Addison-Wesley, , Cop.
8. Columbia.edu. (2023). CAVE. [online] Available at: https://cave.cs.columbia.edu/reference?bid=302 [Accessed 27 Oct. 2023].

9. CS230: Deep Learning. (n.d.). Available at: https://cs230.stanford.edu/files/cs230exam_win19_soln.pdf. [Accessed 27 Oct. 2023].
10. Deeplearningbook.org. (2013). Available at: https://www.deeplearningbook.org/contents/convnets.html [Accessed 25 Oct. 2023].
11. Diaz, C., Sanchez, G., Avalos, J.-G., Sanchez, G., Sanchez, J.-C. and Perez, H. (2017). Spike-based compact digital neuromorphic architecture for efficient implementation of high order FIR filters. Neurocomputing, 251, pp.90–98. doi:https://doi.org/10.1016/j.neucom.2017.04.012.
12. Diogo, Ferreira, A., Medeiros, H.R. and Barros, E. (2019). An embedded automatic license plate recognition system using deep learning. Design Automation for Embedded Systems, [online] 24(1), pp.23–43. doi:https://doi.org/10.1007/s10617-019-09230-5.
13. Dr. Dobb's. (2013). *The OpenCV Library*. [online] Available at: https://www.drdobbs.com/open-source/the-opencv-library/184404319 [Accessed 27 Oct. 2023].
14. Essential Scrum A Practical Guide to the Most Popular Agile Process. (n.d.). Available: https://ptgmedia.pearsoncmg.com/images/9780137043293/samplepages/0137043295.pdf. [Accessed 24 Oct. 2023].
15. Ghida Yousif Abbass and Ali Fadhil Marhoon (2022). Car license plate segmentation and recognition system based on deep learning. [online] ResearchGate. Available at: https://www.researchgate.net/publication/362397875_Car_license_plate_segmentation_and_recognition_system_based_on_deep_learning [Accessed 27 Oct. 2023].
16. Google.com. (2019). *Google Colaboratory*. [online] Available at: https://colab.research.google.com/notebooks/basic_features_overview.ipynb [Accessed 24 Oct. 2023].
17. Hasan Erdinç Koçer and Kerim Kürşat Çevik (2011). Artificial neural networks based vehicle license plate recognition. Procedia Computer Science, [online] 3, pp.1033–1037. doi:https://doi.org/10.1016/j.procs.2010.12.169.

18. Hashmi, S., Kumar, K., Khandelwal, S. and Mittal, S. (2019). *Real Time License Plate Recognition from Video Streams using Deep Learning*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/330052181_Real_Time_License_Plate_Recognition_from_Video_Streams_using_Deep_Learning [Accessed 27 Oct. 2023].

19. Ibm.com. (2023). What are Convolutional Neural Networks? | IBM. [online] Available at: https://www.ibm.com/topics/convolutional-neural-networks [Accessed 23 Oct. 2023].

20. Kaggle.com. (2023). *Find Open Datasets and Machine Learning Projects | Kaggle*. [online] Available at: https://www.kaggle.com/datasets [Accessed 13 Oct. 2023].

21. Khan, A., Sohail, A., Umme Zahoora and Aqsa Saeed Qureshi (2020). A survey of the recent architectures of deep convolutional neural networks. Artificial Intelligence Review, [online] 53(8), pp.5455–5516. doi:https://doi.org/10.1007/s10462-020-09825-6.

22. Kim, S.G., Jeon, H.-S. and Hyung Il Koo (2017). Deep-learning-based license plate detection method using vehicle region extraction. *Electronics Letters*, [online] 53(15), pp.1034–1036. doi:https://doi.org/10.1049/el.2017.1373.

23. Larxel (2020). Car License Plate Detection. [online] Kaggle.com. Available at: https://www.kaggle.com/datasets/andrewmvd/car-plate-detection/data [Accessed 27 Oct. 2023].

24. Li, H., Wang, P. and Shen, C. (2019). Toward End-to-End Car License Plate Detection and Recognition With Deep Neural Networks. IEEE Transactions on Intelligent Transportation Systems, 20(3), pp.1126–1136. doi:https://doi.org/10.1109/tits.2018.2847291.

25. Liu, W.-C. and Lin, C.-H. (2017). A hierarchical license plate recognition system using supervised K-means and Support Vector Machine. [online] doi:https://doi.org/10.1109/icasi.2017.7988244.

26. Madhusri Maity, Banerjee, S. and Sheli Sinha Chaudhuri (2021). Faster R-CNN and YOLO based Vehicle detection: A Survey. [online] doi:https://doi.org/10.1109/iccmc51019.2021.9418274.

27. Microsoft (2021). Visual Studio Code. [online] Visualstudio.com. Available at: https://code.visualstudio.com/docs/python/python-tutorial [Accessed 27 Oct. 2023].
28. Mohit Kumar Kushwaha and G. Suseela (2022). Car Number Plate Detection using Deep Learning. [online] doi:https://doi.org/10.1109/icdsaai55433.2022.10028818.
29. Mustafa, T. and Murat Karabatak (2023). Deep Learning Model for Automatic Number/License Plate Detection and Recognition System in Campus Gates. [online] doi:https://doi.org/10.1109/isdfs58141.2023.10131758.
30. Namysl, M. and Konya, I. (2019). Efficient, Lexicon-Free OCR using Deep Learning. [online] arXiv.org. Available at: https://arxiv.org/abs/1906.01969 [Accessed 27 Oct. 2023].
31. Nvidia.com. (2012). *Realtime Computer Vision with OpenCV | Research*. [online] Available at: https://research.nvidia.com/publication/2012-06_realtime-computer-vision-opencv [Accessed 27 Oct. 2023].
32. Omar, N., Salim and Abdulkadir Sengur (2019). An Efficient Model for Automatic Number Plate Detection using HOG Feature from New North Iraq Vehicle... [online] ResearchGate. Available at: https://www.researchgate.net/publication/338800196_An_Efficient_Model_for_Automatic_Number_Plate_Detection_using_HOG_Feature_from_New_North_Iraq_Vehicle_Images_Dataset [Accessed 27 Oct. 2023].
33. Opencv.org. (2023). OpenCV: Introduction. [online] Available at: https://docs.opencv.org/4.x/d1/dfb/intro.html [Accessed 24 Oct. 2023].
34. Palaiahnakote Shivakumara, Tang, D., Asadzadehkaljahi, M., Tong Lü, Pal, U. and Mohammad Hossein Anisi (2018). CNN-RNN based method for license plate recognition. CAAI Transactions on Intelligence Technology, [online] 3(3), pp.169–175. doi:https://doi.org/10.1049/trit.2018.1015.
35. Palletsprojects.com. (2023). *Application Setup — Flask Documentation (3.0.x)*. [online] Available at: https://flask.palletsprojects.com/en/3.0.x/tutorial/factory/ [Accessed 27 Oct. 2023].

36. Prajapati, R., Bhardwaj, Y., Jain, R. and Kamal Kant Hiran (2023). A Review Paper on Automatic Number Plate Recognition using Machine Learning : An In-Depth Analysis of Machine Learning Techniques in Automatic Number Plate Recognition: Opportunities and Limitations. [online] doi:https://doi.org/10.1109/cictn57981.2023.10141318.
37. Pritam Ahire, Kadam, S. and Ajay Jagtap (2023). *Image Enhancement and Automated Number Plate Recognition*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/370407572_Image_Enhancement_and_Automated_Number_Plate_Recognition [Accessed 27 Oct. 2023].
38. Python Developer's Guide. (2023). Python Developer's Guide. [online] Available at: https://devguide.python.org/ [Accessed 24 Oct. 2023].
39. Qian, R., Lai, X. and Li, X. (2022). 3D Object Detection for Autonomous Driving: A Survey. *Pattern Recognition*, p.108796. doi:https://doi.org/10.1016/j.patcog.2022.108796.
40. Roboflow Universe: Open Source Computer Vision Community (2022). *Roboflow Universe: Open Source Computer Vision Community*. [online] Roboflow. Available at: https://universe.roboflow.com/ [Accessed 27 Oct. 2023].
41. Rohith Polishetty, Mehdi Roopaei and Rad, P. (2016). A Next-Generation Secure Cloud-Based Deep Learning License Plate Recognition for Smart Cities. [online] ResearchGate. Available at: https://www.researchgate.net/publication/313450857_A_Next-Generation_Secure_Cloud-Based_Deep_Learning_License_Plate_Recognition_for_Smart_Cities [Accessed 11 Oct. 2023].
42. Rong, W., Li, Z., Zhang, W. and Sun, L. (n.d.). *An Improved Canny Edge Detection Algorithm*. [online] Available at: https://www.ire.pw.edu.pl/~arturp/Dydaktyka/PPO/pomoce/06885761.pdf.
43. S. Fakhar A. G, Mohd Saad Hamid, ahmad fauzan Kadmin and Aidil, M. (2019). *Development of portable automatic number plate recognition (ANPR) system*

on Raspberry Pi. [online] ResearchGate. Available at: https://www.researchgate.net/publication/333538178_Development_of_portable_automatic_number_plate_recognition_ANPR_system_on_Raspberry_Pi [Accessed 27 Oct. 2023].

44. Sarker, Md.M.K., Weihua, C. and Song, M.K. (2015). Detection and Recognition of Illegally Parked Vehicles Based on an Adaptive Gaussian Mixture Model and a Seed Fill Algorithm. *Journal of information and communication convergence engineering*, [online] 13(3), pp.197–204. doi:https://doi.org/10.6109/jicce.2015.13.3.197.

45. Sérgio Montazzolli and Claudio Rosito Jung (2017). Real-Time Brazilian License Plate Detection and Recognition Using Deep Convolutional Neural Networks. [online] ResearchGate. Available at: https://www.researchgate.net/publication/320677458_Real-Time_Brazilian_License_Plate_Detection_and_Recognition_Using_Deep_Convolutional_Neural_Networks [Accessed 27 Oct. 2023].

46. Sharma, N., Baral, S., May Phu Paing and Rathachai Chawuthai (2023). Parking Time Violation Tracking Using YOLOv8 and Tracking Algorithms. *Sensors*, [online] 23(13), pp.5843–5843. doi:https://doi.org/10.3390/s23135843.

47. Shi, B., Bai, X. and Yao, C. (2015). An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. [online] arXiv.org. Available at: https://arxiv.org/abs/1507.05717 [Accessed 27 Oct. 2023].

48. Studocu. (2022). *Sensors-21-03028*. [online] Available at: https://www.studocu.com/in/document/banaras-hindu-university/environment-management/sensors-21-03028/22904266 [Accessed 27 Oct. 2023].

49. ultralytics (2023). GitHub - ultralytics/ultralytics: NEW - YOLOv8 in PyTorch > ONNX > OpenVINO > CoreML > TFLite. [online] GitHub. Available at: https://github.com/ultralytics/ultralytics [Accessed 26 Oct. 2023].

50. Ultralytics.com. (2023). *YOLO VISION 2023 | The New Frontier of Vision AI*. [online] Available at: https://www.ultralytics.com/yolo-vision [Accessed 27 Oct. 2023].

51. V Gnanaprakash, N Kanthimathi and N. Saranya (2021). Automatic number plate recognition using deep learning. *IOP conference series*, [online] 1084(1), pp.012027–012027. doi:https://doi.org/10.1088/1757-899x/1084/1/012027.

52. Wang, W. and Tu, J. (2020). Research on License Plate Recognition Algorithms Based on Deep Learning in Complex Environment. *IEEE Access*, [online] 8, pp.91661–91675. doi:https://doi.org/10.1109/access.2020.2994287.

53. Yolov8.com. (2023). *YOLOv8: A New State-of-the-Art Computer Vision Model*. [online] Available at: https://yolov8.com/ [Accessed 13 Oct. 2023].

Appendix

Form RE1

RESEARCH ETHICS CHECKLIST
August 2021

Note: *undergraduate and taught postgraduate students must use this form where human participants, human tissues or data, potentially sensitive material or a potential reputational risk forms part of their project. Research students, staff and external researchers must use the online EFIT system.*

This checklist should be completed for every research project. It is used to identify whether a full application for ethics approval needs to be submitted.

Before completing this form, please refer to the University **'Code of Practice on Ethical Standards for Research Involving Human Participants'** and the **'Scope of the Code of Practice'** document. The student's supervisor is responsible for exercising appropriate professional judgment in this review.

This checklist must be completed before potential participants are approached to take part in any research.

Section 1: Applicant Details

1.	Name of Researcher (applicant):	Rashed Haider
2.	Status (please click to select):	Undergraduate
3.	Email Address:	Rashed.he08666@rct.uk.net
4a.	Contact Address:	51A Millais Road Leytonstone London E114HB
4b.	Telephone Number:	07701082720
5.	Project Title:	Optimizing ANPR Performance through Unique Dataset Selection and Model Testing
6.	Course title/module name/number School/Centre:	Course title: Undergraduate Project Module code: SEC6201
7.	Supervisor's or module leader's name:	Dr. Aimee Mirto
8.	Supervisor's Email address:	aimee.mirto@rcl.ac.uk
9.	Supervisor's Telephone number:	n/a
Comments from Researcher, and/or from Supervisor: **There are no ethical violations in this study.**		

Declaration by Researcher *(Please check the appropriate boxes)*

☒	I have read the University's Code of Practice
☒	The topic merits further research
☒	I have the skills to carry out the research
☒	The participant information sheet, if needed, is appropriate
☒	The procedures for recruitment and obtaining informed consent, if needed, are appropriate
☒	The research is exempt from further ethics review according to current University guidelines

☒	Where relevant, I have read the ethical guidelines of the regulatory body that is relevant to my discipline and verify that the research adheres to these guidelines

Section 2: Research Checklist (*Please answer each question by selecting the appropriate response*)

		YES/NO
1.	Will the study involve participants who are particularly vulnerable or who may be unable to give informed consent (e.g. children, people with learning disabilities, emotional difficulties, problems with understanding and/or communication, your own students)?	NO
2.	Will the study require the co-operation of a gatekeeper for initial access to the groups or individuals to be recruited (e.g. students at school, members of self-help group, residents of nursing home)?	NO
3.	Will deception be necessary, i.e. will participants take part without knowing the true purpose of the study or without their knowledge/consent at the time (e.g. covert observation of people in non-public places)?	NO
4.	Will the study involve discussion of topics which the participants (or readers of the research) may find sensitive or disturbing (e.g. sexual activity, drug use, controversial/extreme texts)?	NO
5.	Will drugs, placebos or other substances (e.g. food substances, alcohol, nicotine, vitamins) be administered to or ingested by participants or will the study involve invasive, intrusive or potentially harmful procedures of any kind?	NO
6.	Will human blood or tissue samples be obtained for use in the research?	NO
7.	Will pain or more than mild discomfort be likely to result from the study?	NO
8.	Could the study induce psychological stress or anxiety or cause harm or negative consequences beyond the risks encountered in normal life?	NO
9.	Will the study involve prolonged or repetitive testing?	NO
10.	Will financial inducements (other than reasonable expenses and compensation for time) be offered to participants?	NO
11.	Will participants' right to withdraw from the study at any time be withheld or not made explicit?	NO
12.	Will participants' anonymity be compromised or their right to anonymity be withheld or information they give be identifiable as theirs?	NO
13.	Might permission for the study need to be sought from the researcher's or from participants' employer?	NO
14.	Will the study involve recruitment of patients or staff through the NHS?	NO
15.	Does the research have any potential implications for the reputation of the University?	NO
16.	Does the research involve socially or politically sensitive (actual of potential) topics?	NO
17.	Will the research have the potential to uncover or highlight illegal or potentially harmful activities?"	NO

If ALL items in the Declaration are checked and all items in the Section 2 checklist have been answered NO; **send the completed and signed Form RE1 to your School/Centre Research Ethics Officer (REO)** for information. You should receive a signed copy in return from your REO. You may proceed with the research but should follow any subsequent guidance or requests from the School/Centre Research Ethics Officer or your supervisor/module leader where appropriate.

Undergraduate and taught postgraduate students should retain a copy of this form and submit the REO signed version with their research report or dissertation.

If Question 6 in the Section 2 checklist has been answered YES; the University of Bolton does not hold a Human Tissue Authority (HTA) licence. Therefore, no researcher at University premises can store human tissue (which may be body parts, organs, tissue, cells, bodily waste products, including blood, serum, plasma, etc.), unless an exemption applies. Refer to the document Registration and Storage of Human Tissue for guidance and refer to your Research Coordinator, REO or Designated HTA Officer. **You cannot proceed with your research at this stage.**

If Question 14 in the Section 2 checklist has been answered YES; you will have to submit an application to the appropriate external NHS ethics committee for approval. After you have received approval from the NHS you should then submit an RE2(U) form together with a copy of the NHS approval to the School/Centre Research Ethics Officer who will arrange for UREC to consider your request. **You cannot proceed with your research at this stage.**

If Question 15 and/or Q17 in the Section 2 checklist has been answered YES; you will need to complete an RE2(U) form and send it together with this RE1 form to the School/Centre Research Ethics Officer who will arrange for UREC to consider your request. **You cannot proceed with your research at this stage.**

If ANY of the items in the Declaration are not ticked AND / OR if you have answered YES to questions in Section 2 (apart from Q6, Q14, Q15, or Q17); you will need to describe more fully in Section 3 of this form how you plan to deal with the ethical issues raised by your research. **This does not mean that you cannot do the research, only that your proposal will need to be approved by the School/Centre Research Ethics Officer or DREC/UREC**. You will be guided on completion of form RE2(D) or RE2(U).

Section 3: Addressing Ethical Problems

If you have answered YES to any of questions in Section 2 (apart from Q6, Q14, Q15, or Q17) please complete below and submit the form to your School/Centre Research Ethics Officer.

Project Title

Principal Investigator/Student

Supervisor

Summary of issues and action to be taken to address the ethics problem(s)

Please note that it is your responsibility to follow the University's' Code of Practice on Ethical Standards' and 'Scope of the Code of Practice' alongside any relevant academic or professional guidelines in the conduct of your study. **This includes providing appropriate information sheets and consent forms and ensuring confidentiality in the storage and use of data**.

You may only conduct your research in line with the ethical approval received. Any significant change to the design or conduct of the research should be notified to the School/Centre Research Ethics Officer using form RE7 and may require a new application for ethics approval. You must stop your research until this variation is approved.

Signed: Rashed Haider Principal Investigator/Student

Approved: Dr. A Mirto Supervisor/Module Leader (for UGT & PGT)

Date: October 23,2023

For use by School/Centre Research Ethics Officer (REO):

	Tick all that apply
No ethical problems are raised by this proposed study	
Appropriate action taken to maintain ethical standards	
The research protocol should be revised to eliminate the ethical concerns or reduce them to an acceptable level, using the attached suggestions	
Please submit School/Centre Application for Ethics Approval (Form RE2(D))	
Please submit University Application for Ethics Approval (Form RE2(U))	

Note to REO: ensure this form contains all appropriate signatures and is completed fully. Retain a copy on your file and send a copy to the applicant and their supervisors.

REO name: _____

Signed: _____

Date: _____