Real-time detection and recognition of license plate using YOLO11 object detection model

by

Ravi Teja Vempati

B.Tech, Vellore Institute of Technology (VIT), 2022

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computer Science
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2024

Approved by:

Major Professor
Dr. Mitchell L. Neilsen

Copyright

Abstract

This research explores the need for accurate and efficient license plate detection and recognition using advanced computer vision and deep learning techniques. Traditional approaches to license plate identification are often manual, time-consuming, and prone to error, driving the need for automated solutions. This study aims to develop a real-time system for license plate detection and recognition by utilizing the You Only Look Once (YOLO) 11 object detection model in combination with Google Cloud Vision.

The project involves gathering a dataset of high-resolution license plate images, annotating these images with bounding boxes, and performing data augmentation to enhance variability and standardize the data. The dataset is divided into training, validation, and testing sets to optimize model performance. The YOLO11 model is trained on this annotated dataset, followed by a detailed performance evaluation.

The research also includes a review of existing methods in license plate detection and character recognition, highlighting the role of deep learning techniques in intelligent traffic solutions. Experimental results indicate that the YOLO11 model achieves impressive precision and recall rates, validating its effectiveness for this task.

The final model is deployed on a real-time platform, integrating YOLO11 for detecting license plates and Google Cloud Vision for character recognition, thereby demonstrating its practical utility in traffic monitoring systems. This study provides a scalable and efficient framework for license plate recognition, contributing to the development of smart transportation solutions that enhance vehicle tracking and traffic management.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I wish to convey my heartfelt appreciation to my committee members Dr. Mitchell Neilsen, Dr. Torben Amtoft, and Dr. Daniel Andresen for generously offering their time and support to serve on my committee throughout this project. I am particularly grateful to my Major Professor, Dr. Mitchell Neilsen, for his unwavering belief in my capabilities and consistent support since my third semester at Kansas State University. I also extend my thanks to Dr. Ajay Sharda and the Farmslab team for their invaluable mentorship during my tenure as a Graduate Research Assistant.

I owe a tremendous debt of gratitude to my family members, whose unwavering support made this journey possible. To my mother, Shailaja Vempati, father, Achaiah Vempati, sister, Kavya Vempati, I am deeply thankful for their boundless love, encouragement, and emotional guidance, which have been indispensable in my life and achievements thus far.

Lastly, but certainly not least, I express my appreciation to my friends Pavan Lakkireddy, Bharani Bala, Pavan Chandana, Sai Surya Ravipati for their unwavering support and companionship. These past two years have been immensely enjoyable and memorable, thanks to their friendship and encouragement.

Chapter 1 - Introduction

1.1 Problem Definition

The rapid advancements in computer vision technology have paved the way for innovative solutions to address critical challenges in various fields, including intelligent transportation systems. One significant application is in the area of license plate detection and recognition. Accurate identification of license plates is vital for numerous applications, such as traffic monitoring, law enforcement, toll collection, and vehicle tracking. Traditional methods for license plate identification typically involve manual inspection, which is not only time-consuming but also prone to human error. Recently, deep learning-based object detection techniques have emerged as highly effective tools for automating this process, enabling real-time, precise detection and recognition of license plates.

1.2 Objective

This report provides an in-depth study on the development and implementation of a real-time license plate detection and recognition system using the You Only Look Once (YOLO) 11 object detection model. The primary objective is to overcome the limitations of traditional license plate identification methods by utilizing advanced deep learning and computer vision techniques.

The project workflow begins with the collection of a custom dataset comprising high-resolution images of vehicle license plates captured under diverse environmental conditions to ensure data quality and consistency. These images are labeled using the LabelImg annotation tool to create bounding boxes that serve as ground truth labels for model training.

To improve the model's performance and robustness, extensive preprocessing techniques are applied to the dataset. This includes standardizing the images through auto-orientation and resizing, as well as data augmentation methods such as flipping, rotation, and color adjustments to increase variability and enhance the model's ability to generalize.

The dataset is then divided into training, validation, and testing sets to optimize the training and evaluation process. The YOLO 11 object detection model is employed for training on the annotated dataset, leveraging its advanced architecture and efficiency for real-time inference.

Upon completing the training process, the performance of the trained model is thoroughly assessed using a series of validation and testing procedures. Finally, the model is deployed in a real-time environment, allowing for live detection and recognition of license plates in various scenarios.

This study adds to the body of research in intelligent transportation systems, offering a scalable and efficient solution for automated license plate detection and recognition. The insights and results derived from this project have significant implications for enhancing traffic monitoring, law enforcement, and vehicle tracking strategies.

Chapter 2 – Related Work

2.1 Literature Survey

In the paper "Real-time automatic license plate recognition through deep multi-task networks" by Gonçalves et al. (2018), a novel approach to Automatic License Plate Recognition (ALPR) is presented using deep multi-task learning networks. The researchers developed a system that simultaneously addresses both the detection and recognition tasks in ALPR, emphasizing real-time performance. Their approach utilizes convolutional neural networks (CNNs) to extract features from vehicle images, which are then processed through multiple task-specific branches for license plate detection and character recognition. The multi-task learning framework allows the model to share common features across tasks, potentially improving overall performance and efficiency. The authors evaluated their system on various datasets, including challenging real-world scenarios with different lighting conditions, angles, and occlusions. Results showed that their multi-task approach achieved high accuracy in both license plate detection and character recognition while maintaining real-time processing speeds. This study demonstrates the potential of multi-task learning in ALPR systems, offering a balance between accuracy and computational efficiency.

Li et al. (2019), in their paper "Toward end-to-end car license plate detection and recognition with deep neural networks," propose an innovative end-to-end deep neural network for license plate detection and recognition. Their approach emphasizes both efficiency and accuracy, aiming to streamline the ALPR process. The researchers designed a unified network architecture that can perform plate detection and character recognition in a single forward pass, eliminating the need for separate stages in traditional ALPR pipelines. This end-to-end approach not only improves the overall

system efficiency but also allows for better feature sharing between the detection and recognition tasks. The authors evaluated their system on multiple datasets, demonstrating its robustness across various plate styles and environmental conditions. The results showed significant improvements in both accuracy and processing speed compared to traditional multi-stage ALPR systems. This work represents a significant step towards more integrated and efficient ALPR solutions, potentially paving the way for wider adoption in real-time applications.

The study "Robust license plate recognition using deep learning" by Björklund et al. (2019) focuses on creating a robust ALPR system capable of handling various real-world conditions. The researchers addressed the challenges posed by diverse environments, including varying lighting conditions, occlusions, and different plate styles. Their approach utilizes advanced deep learning techniques, incorporating data augmentation and transfer learning to improve the model's generalization capabilities. The authors paid particular attention to preprocessing techniques that enhance the quality of input images, making their system more resilient to poor image quality often encountered in real-world scenarios. Evaluation results demonstrated the system's ability to maintain high accuracy across a wide range of challenging conditions, outperforming several existing methods. This work contributes significantly to the field by highlighting the importance of robustness in ALPR systems and providing effective strategies to achieve it.

Kessentini et al. (2019), in their paper "A two-stage deep neural network for multi-norm license plate detection and recognition," introduce a novel two-stage approach for detecting and recognizing license plates across multiple international norms. This research addresses the growing need for ALPR systems that can operate effectively across different countries and plate standards. The first stage of their network focuses on plate detection, utilizing a region proposal network optimized for license

plate localization. The second stage handles character segmentation and recognition, employing a specialized CNN architecture. By separating these tasks, the authors were able to fine-tune each stage for optimal performance. The system was evaluated on a diverse dataset comprising license plates from various countries, demonstrating impressive accuracy and adaptability. This work is particularly significant for its contribution to creating more versatile ALPR systems capable of operating in global contexts.

Qian et al. (2018), in their paper "Robust Chinese traffic sign detection and recognition with deep convolutional neural network," while primarily focused on traffic sign detection, present techniques that are highly applicable to license plate detection, especially in challenging environments. The researchers developed a deep convolutional neural network architecture that showed remarkable robustness in detecting and recognizing various traffic signs under different weather conditions, lighting, and occlusions. Their approach to handling small object detection and dealing with environmental variability offers valuable insights for license plate detection systems. The authors' emphasis on real-time performance and their strategies for optimizing network efficiency are particularly relevant to ALPR applications. This study underscores the potential for cross-pollination of ideas between different areas of intelligent transportation systems, suggesting that advancements in traffic sign recognition can inform and improve license plate detection techniques.

In "A new CNN-based method for multi-directional car license plate detection," Xie et al. (2018) propose a CNN-based method specifically designed for detecting license plates from multiple directions. This research addresses the critical challenge of varied plate orientations often encountered in real-world scenarios. The authors developed a novel network architecture that incorporates rotation-invariant features, allowing the system to accurately detect and localize license plates regardless of

their orientation. Their approach includes a two-stage detection process: an initial region proposal stage followed by a refinement stage that precisely localizes the plate. Extensive evaluations on diverse datasets demonstrated the method's superior performance in detecting plates from various angles and under different conditions. This work significantly contributes to the field by enhancing the flexibility and robustness of ALPR systems, making them more suitable for complex, real-world deployments where vehicle and camera orientations cannot be controlled.

Selmi et al. (2020), in their paper "Deep learning system for automatic license plate detection and recognition," present a comprehensive deep learning system that addresses both the detection and recognition aspects of ALPR. Their approach focuses on handling diverse plate styles and conditions, making it particularly relevant for real-world applications. The system employs a two-stage architecture: the first stage uses a custom-designed CNN for efficient license plate detection, while the second stage utilizes a combination of CNNs and recurrent neural networks for character segmentation and recognition. The authors paid special attention to data augmentation techniques to improve the model's ability to handle variations in plate appearances. Evaluation results on multiple datasets, including their own collected dataset of challenging cases, showed high accuracy rates in both detection and recognition tasks. This study stands out for its holistic approach to ALPR, offering insights into creating end-to-end systems that can perform robustly across various scenarios.

These studies collectively highlight the significant progress made in license plate detection and recognition using deep learning techniques. From multi-task learning and end-to-end approaches to specialized methods for handling multi-directional plates and diverse international norms, the field has seen substantial advancements. The high accuracy rates and real-time performance demonstrated by these systems underscore their potential for practical deployment in various applications, from

traffic management to law enforcement. As research continues, future work is likely to focus on further improving robustness to real-world conditions, enhancing efficiency for large-scale deployments, and adapting to emerging challenges in intelligent transportation systems.

2.2 Established Methods and Approaches

2.2.1 YOLO

You Only Look Once" (YOLO) is a breakthrough in real-time object detection, introduced by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi in 2016. YOLO stands out due to its innovative approach of processing an image in a single forward pass through a neural network, predicting both object bounding boxes and their associated class probabilities simultaneously (Redmon et al., 2016). This design significantly differs from traditional object detection methods that require multiple stages of processing, such as region proposal and classification, thereby increasing computational efficiency and speed.

The core of YOLO's architecture relies on a convolutional neural network (CNN) backbone, often built using architectures like Darknet or MobileNet, to extract meaningful hierarchical features from the input images (Redmon et al., 2016). The network captures both the fine-grained details and broader contextual features essential for accurate object recognition. The architecture then divides the image into a grid, with each cell responsible for detecting objects within its bounds, predicting bounding boxes, and classifying the detected objects.

This grid-based approach enables YOLO to manage complex object detection tasks efficiently while maintaining high levels of accuracy. Its combination of speed and precision makes it ideal for real-time applications, including areas like autonomous vehicles, robotics, and surveillance

systems. YOLO's balance of processing speed, accuracy, and straightforward architecture has positioned it as a foundational model in the evolution of computer vision technologies, inspiring further advancements and developments in object detection methods.

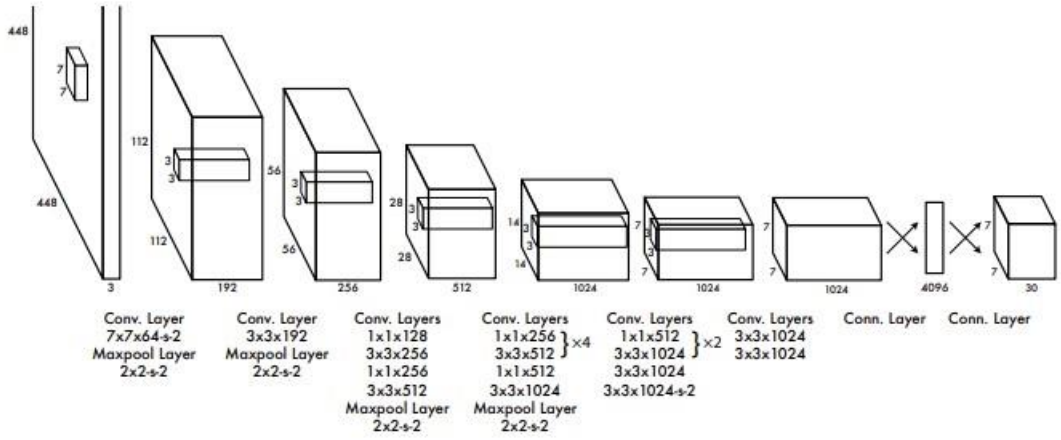

Figure 1 - YOLO Architecture (Redmon, et., al 2016)

2.2.2 YOLOv10 and YOLO11

Following the advancements of YOLOv5, which was introduced in 2020 by Ultralytics, YOLOv10 and YOLO11 represent the latest developments in the YOLO series. YOLOv5 gained recognition for its exceptional speed, user-friendly interface, and robust performance in various object detection tasks, outperforming earlier models with improved accuracy and simplified training protocols (Terven et al., 2023). This model has become a favored choice among developers for its efficiency and effectiveness.

In 2022, Ultralytics released YOLOv8, building on the foundation of YOLOv5 while introducing enhancements in architecture and user experience (Jocher et al., 2023). YOLOv8 excels in speed and precision, offering a unified platform for a range of tasks, including object detection, instance

segmentation, and image classification. Its new features further enhance the model's adaptability and performance, solidifying its position as a leading choice for developers in the field of computer vision.

As for YOLOv10 and YOLO11, these versions continue to refine the capabilities established by their predecessors, focusing on improving accuracy, processing speed, and ease of use. Each iteration aims to push the boundaries of what is possible in real-time object detection, making them ideal for a variety of applications, from autonomous systems to complex surveillance setups. While YOLOv8 remains a prominent model, the ongoing evolution seen in YOLOv10 and YOLO11 highlights the continuous advancements in deep learning and object detection technologies.

Overall, YOLOv5 and YOLOv8, along with the latest iterations YOLOv10 and YOLO11, exemplify the significant progress in the YOLO series, providing developers with powerful tools for tackling complex object detection challenges.

Table1-Performance comparison of YOLOv10 and YOLO11.

Model Size	***YOLOv10***	***YOLO11***	***Difference***
Nano	39.5	39.5	0%
Small	46.8	47	+0.42%
Medium	51.3	51.5	+0.38%
Large	53.4	53.4	0%
Xtra Large	54.4	54.7	+0.55%

2.2.3 Roboflow

Roboflow is an innovative platform that simplifies the development of computer vision workflows,

making it particularly effective for real-world applications (Lin et al., 2022). The platform offers a comprehensive suite of tools designed not only for data organization but also for enhancing dataset quality, streamlining model training, and accelerating deployment processes. It supports various tasks, including image classification, segmentation, and real-time object detection, catering to a wide range of computer vision needs.

In our project, we utilized Roboflow specifically for the real-time detection task. After training our YOLOv8 model and fine-tuning its weights, we integrated these weights into the deployment process using Roboflow's deploy() function. This integration proved to be seamless, allowing us to deploy the model quickly and efficiently, ultimately saving significant time and resources during the implementation phase. Additionally, we leveraged the fine-tuned weights for Google Cloud Vision API to enhance text recognition capabilities, facilitating more accurate data extraction from images.

The ability to easily deploy models through Roboflow not only enhances workflow efficiency but also empowers developers to focus more on refining their models and less on the intricacies of deployment logistics. Such features make Roboflow an invaluable asset in the realm of computer vision development, especially when combined with powerful tools like the Google Cloud Vision API for tasks such as text recognition

.Chapter 3 - Implementation

3.1 Overview of the Data

Our initial dataset comprised a total of 5,402 images, specifically designed to capture variations in license plate appearances. This dataset included a balanced distribution of different angles, lighting conditions, and environmental backgrounds to enhance the model's generalizability. To address potential limitations from a finite dataset size and to improve the robustness of the model, a variety of data augmentation techniques were employed.

The augmentation techniques applied to the dataset included:

Horizontal Flipping: Used to simulate different orientations of license plates, which helps the model recognize plates from various angles.

Rotation Augmentation: Adjusted the angles of license plates to introduce variability in the dataset, making the model adaptable to real-world scenarios where angles can vary.

Brightness and Exposure Adjustments: Varied the brightness and exposure levels within the bounding boxes to simulate different lighting conditions, enhancing the model's ability to perform under diverse environmental conditions.

Noise Injection: Introduced noise into the images to mimic image distortions, thereby improving the model's resilience to imperfections in the input data.

Hue Augmentation: Altered the color spectrum of the images to account for color variations that might occur due to lighting or environmental changes.

Following these data augmentation techniques, the dataset size effectively expanded to a larger number of images, enhancing its diversity and ensuring the model can generalize well to unseen data.

The dataset was then divided into three distinct subsets:

3.1.1 Training Set: Comprising 4,962 images used to teach the model the core patterns and characteristics of license plates.

3.1.2 Validation Set: Containing 220 images, this set was used to fine-tune the model parameters and prevent overfitting during the training phase.

3.1.3 Test Set: A separate set of 220 images was reserved for the final evaluation of the model's performance on completely unseen data, providing a realistic assessment of its generalizability in real-world scenarios.

This strategic division of data ensures that the model is thoroughly trained, validated, and tested to perform effectively under diverse conditions, ultimately enhancing its reliability in practical applications.

3.2 Dataset Preparation

In this section, we discuss in detail how the dataset is collected and how it is further processed before utilizing in our experiment.

3.2.1 Collection and Labelling

The dataset utilized for this project primarily comprised an open-source collection of license plate images. To further enrich the dataset and improve its diversity, we supplemented it with additional license plate images gathered from various internet sources, including CCTV footage. This approach ensured a more comprehensive range of license plate appearances under different conditions.

The data collection involved sourcing images that varied in lighting, angles, and environmental settings, which is essential for enhancing the model's ability to generalize in real-world scenarios. The inclusion of CCTV footage contributed to this variability, providing the model with challenging examples of license plates under less controlled conditions.

The annotation process was carried out using Roboflow, a tool designed to facilitate efficient image annotation and dataset preparation. Key steps in the annotation process included:

Bounding Box Creation: Each license plate in the images was manually labeled with bounding boxes to indicate its location within the frame.

Class Labeling: Corresponding class labels were assigned to each bounding box, specifying the identified object as a license plate.

YOLO Format Conversion: The annotated data was then converted into YOLO-compatible format, ensuring that the dataset was ready for use in object detection models.

To organize the annotation files, each image's annotation was saved as a YOLO-formatted text file with the same name as the image. A separate file named "classes.txt" was also created, listing the class names used in the annotations. This structured setup is crucial for enabling the YOLO model to accurately interpret the labeled data and learn the characteristics of each object class.

By combining open-source data with internet-sourced images and applying a rigorous annotation process, the dataset was optimized to train a robust model capable of detecting license plates in a variety of scenarios.

Figure 2 - Sample images from the Dataset.

3.2.2 Data Pre-processing

During the data preprocessing phase, we applied two key techniques to prepare the images for model training, ensuring they were in optimal condition for the deep learning algorithm. The first step was to apply Auto-Orient, which analyzes each image's metadata and corrects any unintentional rotation. This adjustment standardizes the orientation of the images, ensuring a consistent view of objects regardless of how they were initially captured.

Next, we resized the images using a stretch-to-640x640 approach. This method resizes each image to a uniform dimension of 640 x 640 pixels, maintaining compatibility with the input requirements of the model. Although stretching can sometimes distort the image, this trade-off was necessary to ensure that all images had a consistent size, facilitating faster and more efficient processing during the training phase.

These preprocessing steps ensured that our dataset consisted of uniformly sized and properly oriented images, thereby enhancing the model's learning capabilities and overall performance.

3.2.3 Data Augmentation

To improve the dataset's diversity and enhance the model's robustness, we implemented specific data augmentation techniques aimed at simulating real-world scenarios and variations. This approach was used to artificially expand the dataset and ensure that the model developed greater generalization capabilities.

The key augmentations applied included:

Crop Zoom Augmentation: This technique involved a range of zoom levels from 0% minimum to a maximum of 10%. This variation mimicked the effect of different object sizes in the images, helping the model learn to recognize license plates at varying scales.

Rotation Augmentation: We applied random rotations within the range of -11° to +11°, simulating slight tilts that could occur during image capture. This adjustment allowed the model to become more adept at identifying objects even when they are viewed from slightly skewed angles.

These augmentations effectively increased the dataset size and diversity, ensuring the model's ability to perform well under different conditions and scenarios. This refined dataset was then utilized for training, validation, and testing to develop a more versatile and reliable model.

Figure 3 - Images after Data Augmentation

3.3 Implementation Steps

This study outlines a detailed workflow for real-time object detection using YOLO models, as illustrated in the accompanying diagram. The process initiates with the collection and meticulous annotation of objects within the images. Subsequently, the dataset undergoes preprocessing and augmentation techniques aimed at enhancing its quality and generalizability.

After preparing the dataset, it is systematically divided into training, validation, and testing sets. This division is crucial for ensuring a comprehensive evaluation of the model's performance. The refined dataset is then employed to train both the baseline YOLOv10 model and the newer YOLO11 model. Their respective performances are assessed and compared, showcasing the effectiveness of each model for real-time object detection.

This workflow effectively illustrates the capabilities of YOLO models in handling object detection tasks in dynamic environments, paving the way for further research and application in various domains.

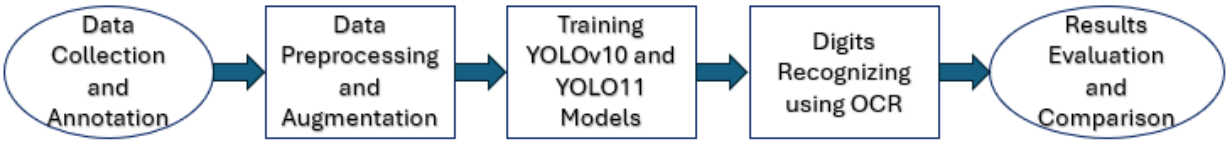

Figure 4 – Project Pipeline

Chapter 4 – Experiments

This section summarizes the experiments conducted on the dataset after preprocessing and augmentation. The dataset was divided into training, validation, and testing sets to optimize model training and evaluation.

The majority of the images were allocated to the training set, while smaller subsets were designated for validation and testing. Experiments focused on two YOLO models: the baseline YOLOv10 and the YOLO11. Performance metrics such as precision, recall, and mean Average Precision (mAP) were employed to assess the models' detection capabilities and improvements. The results highlighted the effectiveness of the preprocessing techniques and the advancements in real-time object detection using the YOLO framework.

4.1 Training, Validation, and Test Data Split

To optimize model training and evaluation, the dataset of 4,600 images was divided into three subsets. Approximately 4,096 images (about 80%) were allocated to the training set, providing a substantial foundation for the model to learn essential features and patterns.

The validation set comprised around 220 images (approximately 4%), which was critical for monitoring the training process and preventing overfitting. This subset allowed for adjustments during training based on the model's performance, ensuring better generalization to new data. Lastly, the test set included 221 images (about 4%), serving as an unbiased measure of the model's ability to perform on unseen data. Evaluating the model against this test set helped

gauge its potential effectiveness in real-world applications. This structured approach to dataset division facilitated a thorough assessment of the model's capabilities.

4.2 Experimental Design

This chapter details the comprehensive steps required to execute each experiment. The first experiment focused on the YOLOv10 baseline model, with results meticulously recorded prior to the training of the YOLO11 model. A comparative analysis of the outcomes from both models followed, highlighting improvements and performance differences.

Additionally, the fine-tuned weights from the YOLO11 model were utilized for Optical Character Recognition (OCR) tasks, enhancing text detection capabilities. Ultimately, the trained model was integrated into the Roboflow framework, facilitating real-time detection and performance evaluation in live scenarios.

4.2.1 YOLOv10

In our first experiment, we implemented the YOLOv10 architecture with hyperparameters configured as follows: a learning rate of 0.000286, a batch size of 16, and a total of 20 epochs. Given the anticipated duration of the training process, I opted to utilize Beocat, the high-performance computing cluster at Kansas State University. Beocat offers significant computational resources, including thousands of cores and extensive memory, which are ideal for executing large-scale machine learning tasks efficiently.

Throughout the 20 epochs, Beocat's capabilities facilitated rapid training without the usual resource constraints, with the total training time being approximately 2 hours on Beocat's GPUs. Upon completion, we assessed the model's performance using a dedicated testing dataset, which revealed that the YOLOv10 model exhibited robust object detection abilities with the provided data.

4.2.2 YOLO11

Our research encompassed the sequential training of two object detection architectures: YOLOv10 followed by YOLO11. We maintained consistent training parameters across both models, implementing a learning rate of 0.000286, processing batches of 16 samples, and running each model for 20 epochs. The training was executed on Beocat's computing infrastructure, which provided the necessary computational power for these intensive workloads and took around 2 hours.

A notable observation during YOLO11's training phase was the stabilization of performance metrics around the 10-epoch mark. After this point, additional training cycles produced only marginal improvements, suggesting the model had achieved optimal convergence.

Comprehensive documentation of YOLO11's performance metrics was completed, with deeper analysis reserved for subsequent chapters. Our comparative assessment used YOLOv10 as the control model. While both architectures demonstrated strong capabilities, YOLO11 exhibited modest improvements, indicating potential benefits for specific use cases.

We conducted additional validation using a separate test dataset to evaluate YOLO11's generalization abilities. The results confirmed the model's practical viability for object detection

applications. The successful implementation and training of both architectures validate our methodological approach and confirm the quality of our dataset preparation procedures.

4.2.3 EasyOCR

EasyOCR is an open-source Optical Character Recognition (OCR) tool designed for high-performance text recognition across a variety of languages and formats. Here are three key features:

1. Multilingual Capabilities: EasyOCR supports over 80 languages, making it suitable for diverse applications in different linguistic contexts. This feature allows users to extract text from images in multiple languages seamlessly.
2. Advanced Deep Learning Architecture: Built on the PyTorch framework, EasyOCR employs state-of-the-art deep learning models, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), to enhance the accuracy and efficiency of text detection and recognition.
3. User-Friendly API: EasyOCR is designed for ease of use, providing a straightforward API that enables developers to integrate OCR functionalities into their applications effortlessly. It also offers built-in image preprocessing features to adapt to various input formats.

4.2.4 Google Cloud Vision API

The Google Cloud Vision API is a powerful tool designed for image analysis, providing developers with the ability to integrate advanced image recognition capabilities into their applications. Here are some key features of the API:

1. Label Detection: The API can identify objects and scenes within images, providing descriptive labels that facilitate image classification and organization. This feature is particularly useful for content categorization and metadata generation.
2. Text Recognition: The Vision API excels at Optical Character Recognition (OCR), enabling it to extract text from images, including handwritten text, printed text, and text within complex backgrounds. This capability is essential for applications requiring document scanning and text extraction.
3. Facial Recognition: Although it does not provide facial recognition in the sense of identifying individuals, the API can detect faces within images and analyze their attributes, such as emotional expressions and facial landmarks.
4. Safety and Adult Content Detection: The API includes functionality to detect inappropriate content, allowing applications to filter out harmful or sensitive material automatically.
5. Integration with Other Google Services: The Vision API can seamlessly integrate with other Google Cloud services, enhancing its utility in various projects, from machine learning to data analysis.

Chapter 5 – Experimental Results

This chapter focuses on the results derived from the model's validation and testing phases. We will examine several critical metrics, including mean Average Precision (mAP), Precision, and Recall. To enhance our analysis, we will also present various visual aids such as confusion matrices, F1 score curves, and validation batch data.

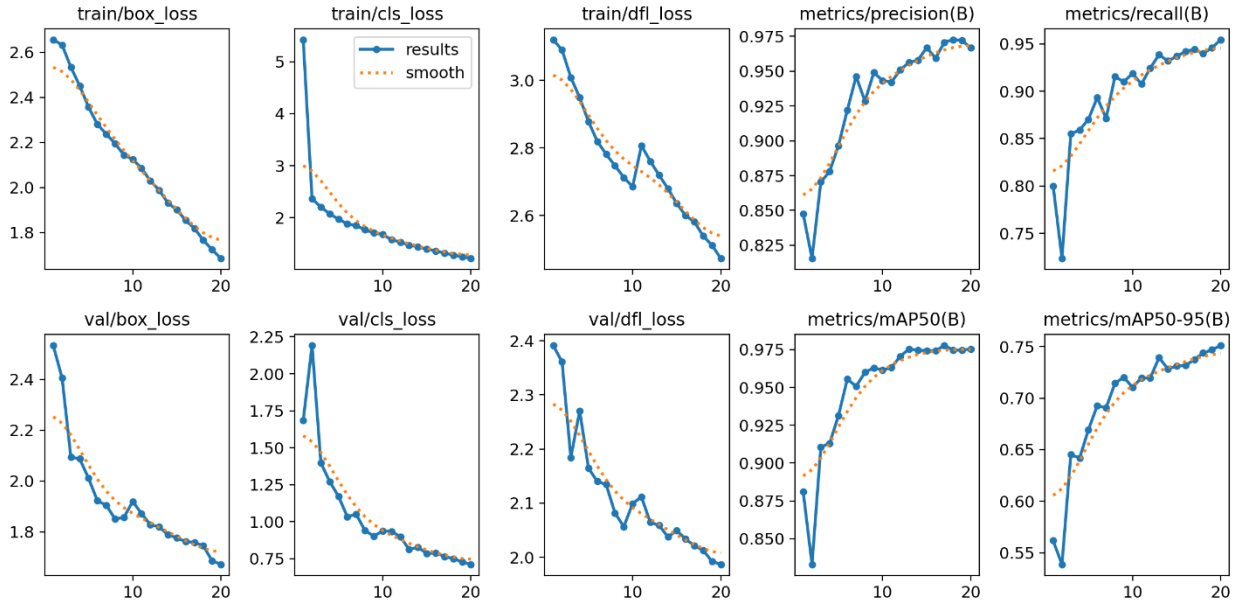

Figure 5 - YOLOv10 Results on Training and Validation datasets.

The empirical analysis over 20 epochs revealed compelling patterns in both training and validation metrics. During training, precision steadily improved to 0.97, demonstrating the model's exceptional accuracy in positive predictions, while recall reached 0.95, indicating robust detection of true positives. The validation results were equally impressive, with mAP50 climbing to 0.95 and mAP50-95 achieving 0.75 across various IoU thresholds. These parallel improvements in

training and validation metrics confirm the model's strong generalization capabilities and effective learning without overfitting, showcasing its ability to maintain high performance on unseen data.

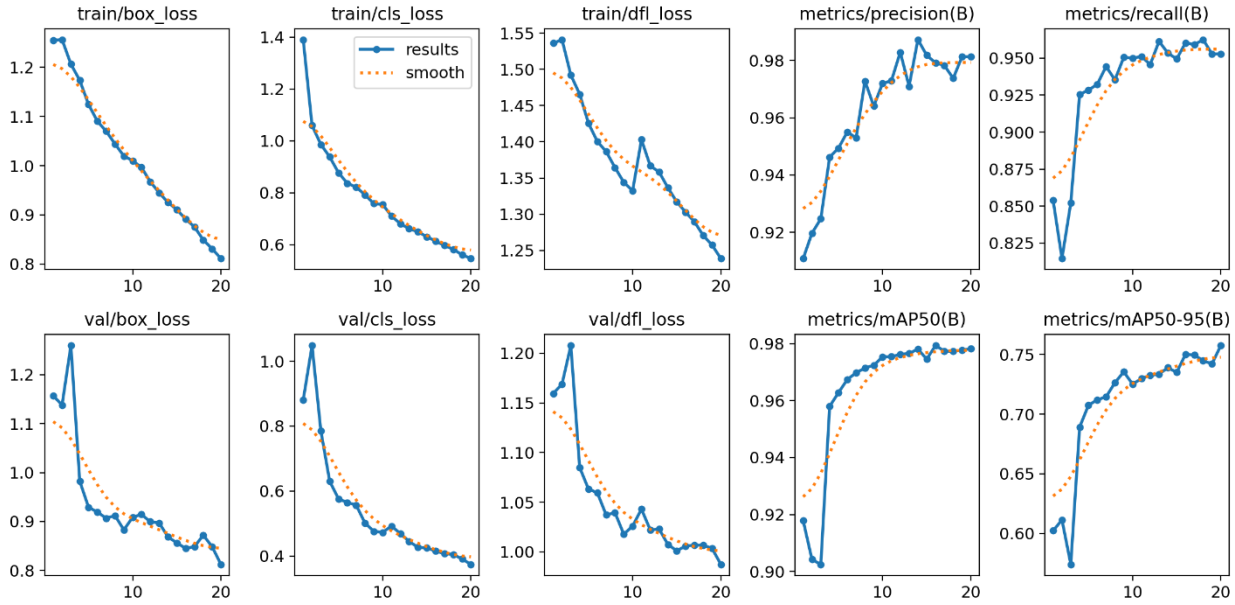

Figure 6 - YOLO11 results on training and Validation datasets.

The graphs illustrate both the training and validation losses alongside essential performance metrics. The training precision graph indicates a consistent upward trajectory, reflecting the model's enhanced ability to accurately identify positive instances, ultimately surpassing 0.98. In a similar vein, the training recall graph displays an increase, nearing 0.95, which suggests an improved capacity for detecting true positives. Meanwhile, the validation metrics, mAP50 and mAP50-95, demonstrate the model's effectiveness on unseen data, with trends approaching 0.98 and 0.75, respectively. These results highlight the model's robust generalization across various Intersection over Union (IoU) thresholds..

5.1 Evaluation Metrics

Object detection performance assessment relies heavily on quantifiable measurements that gauge model effectiveness. For YOLO architectures, a comprehensive evaluation framework employs multiple metrics to analyze detection accuracy and object localization precision. These numerical indicators serve as essential tools for researchers to benchmark performance, conduct comparative analyses, and identify opportunities for model enhancement. By leveraging these metrics, teams can make data-driven decisions about model selection and optimization strategies. The various performance indicators used in this study provide a detailed understanding of the model's behavior, including its ability to accurately locate and classify objects across diverse scenarios. These carefully selected metrics enable thorough assessment of both the model's fundamental capabilities and its practical applicability in real-world object detection tasks.

Table 2 - Comparison of YOLOv10 and YOLO11 Training dataset results.

	mAP		Precision		Recall	
Class	YOLOv10	YOLO11	YOLOv10	YOLO11	YOLOv10	YOLO11
All	0.976	0.978	0.968	0.981	0.954	0.953

Table 3 - Comparison of YOLOv10 and YOLO11 Test dataset results.

	mAP		Precision		Recall	
Class	YOLOv10	YOLO11	YOLOv10	YOLO11	YOLOv10	YOLO11
All	0.987	0.995	0.995	0.998	0.986	0.986

5.2 Mean Average Precision

Mean Average Precision (mAP) is a crucial metric for evaluating object detection models like YOLO. It quantifies the average precision across various object categories and provides an overall score by averaging these precision values. Within the context of YOLO, mAP serves as a concise indicator of the model's detection accuracy, facilitating straightforward comparisons between different models and configurations.

When evaluated on the training dataset, the YOLOv10 model achieved a mAP of 97.6%, while the YOLO11 model attained a mAP of 97.8%, highlighting a slight improvement in object detection performance for YOLO11 over YOLOv10. On the test dataset, YOLOv10 recorded a mAP of 98.7%, whereas YOLO11 achieved 99.5%. This indicates that YOLO11 outperforms YOLOv10 in generalization and accuracy on unseen data, reflecting its enhanced capability in object detection tasks.

5.3 F1-Confidence Curve

The F1-Confidence Curve is a valuable tool for determining the optimal confidence threshold for a model. The peak of this curve identifies the point where the model achieves the best trade-off between precision and recall, indicating the most balanced performance for accurate predictions.

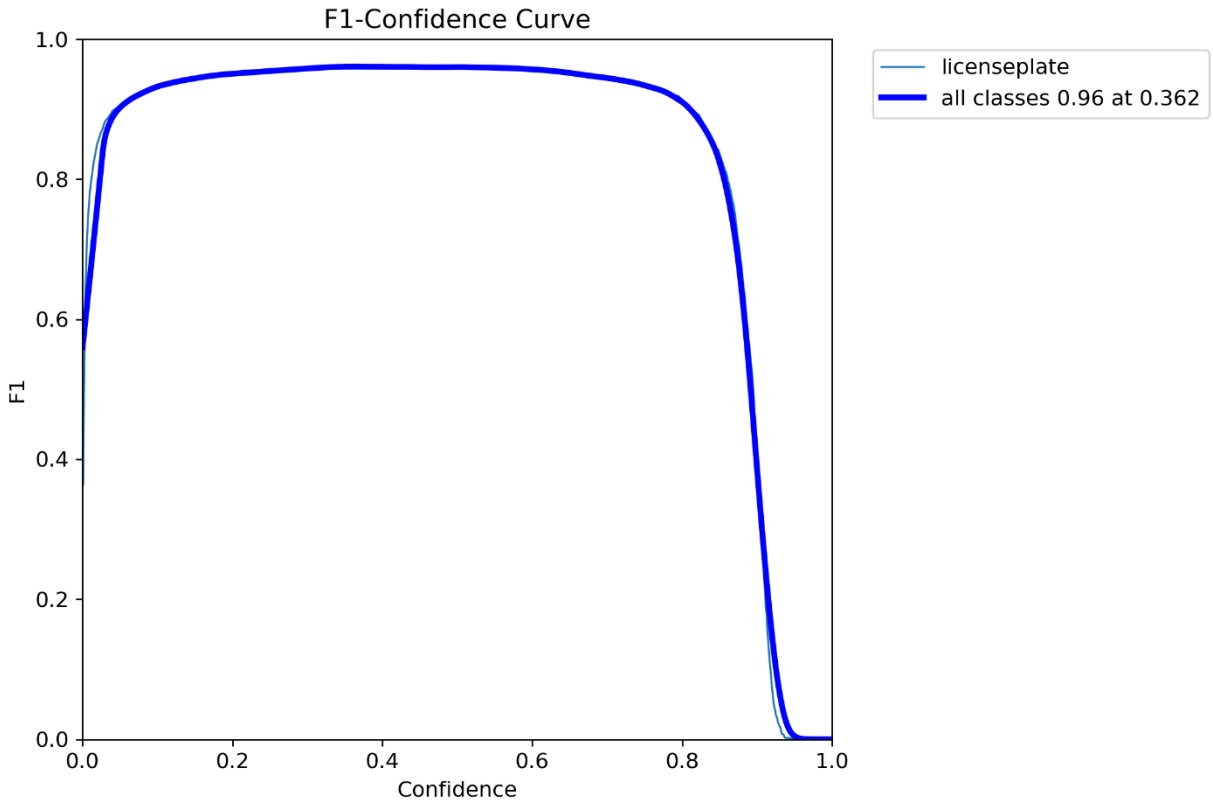

Figure 7 - YOLOv10 F1 Confidence Curve on the training dataset.

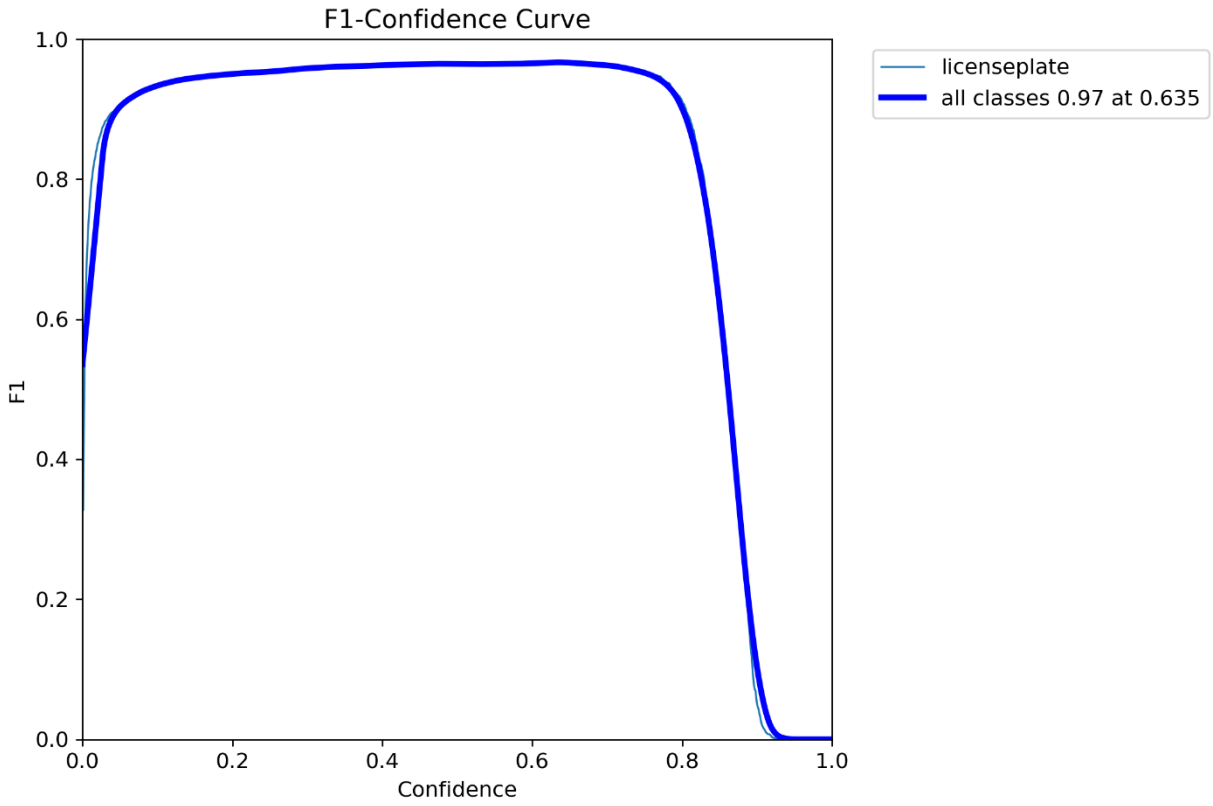

Figure 8 - YOLO11 F1 Confidence Curve on the training dataset.

A comparison of the F1-Confidence curves for YOLOv10 and YOLO11 indicates significant advancements. YOLO11 reaches a peak F1 score of 0.87 at a confidence threshold of 0.541, surpassing YOLOv10's peak of 0.85 at 0.535. Additionally, YOLO11's curve sustains higher F1 scores across a broader range of confidence thresholds, reflecting enhanced performance and robustness. The individual class curves for YOLO11 exhibit reduced variation, indicating more consistent detection across different classes. Furthermore, the smoother decline in YOLO11's curve at higher confidence levels suggests a better balance between precision and recall, highlighting YOLO11's improved capability in maintaining accuracy during high-confidence

predictions. These improvements underscore YOLO11's superior object detection performance across varying confidence thresholds.

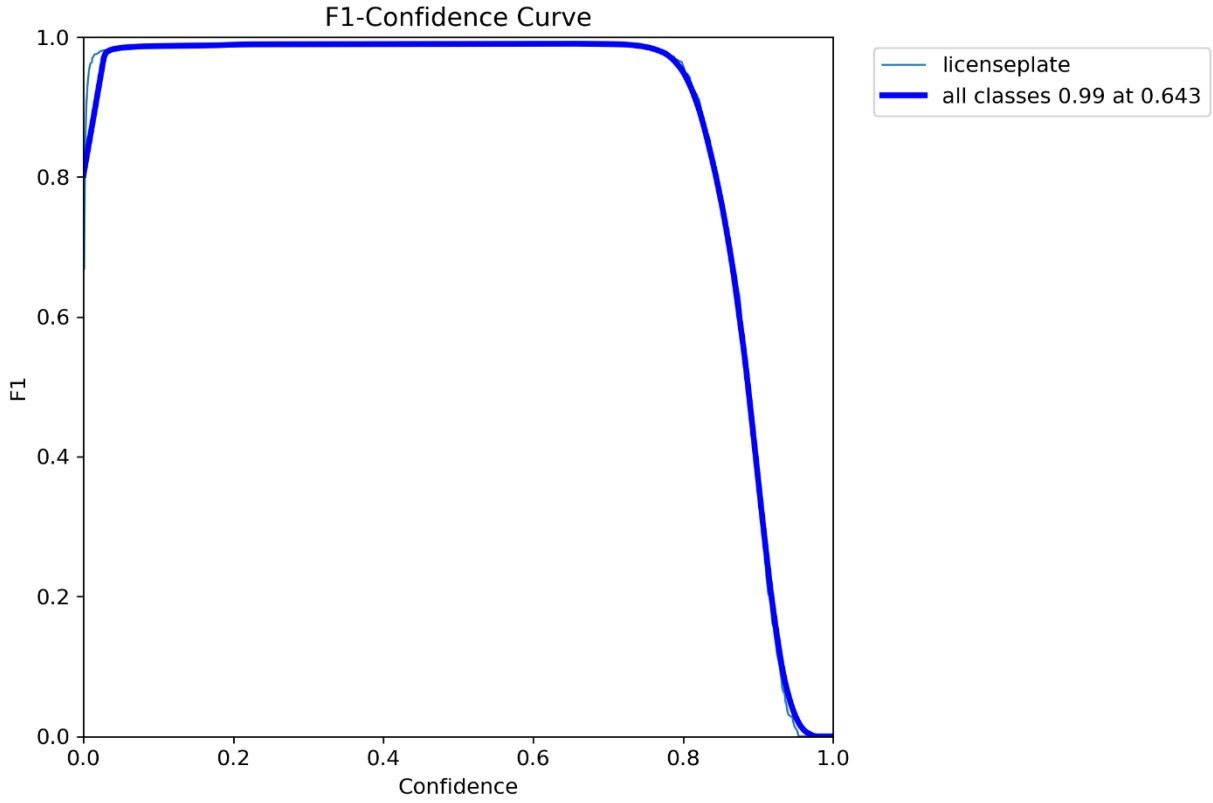

Figure 9 - YOLOv10 F1 Confidence Curve on the test dataset.

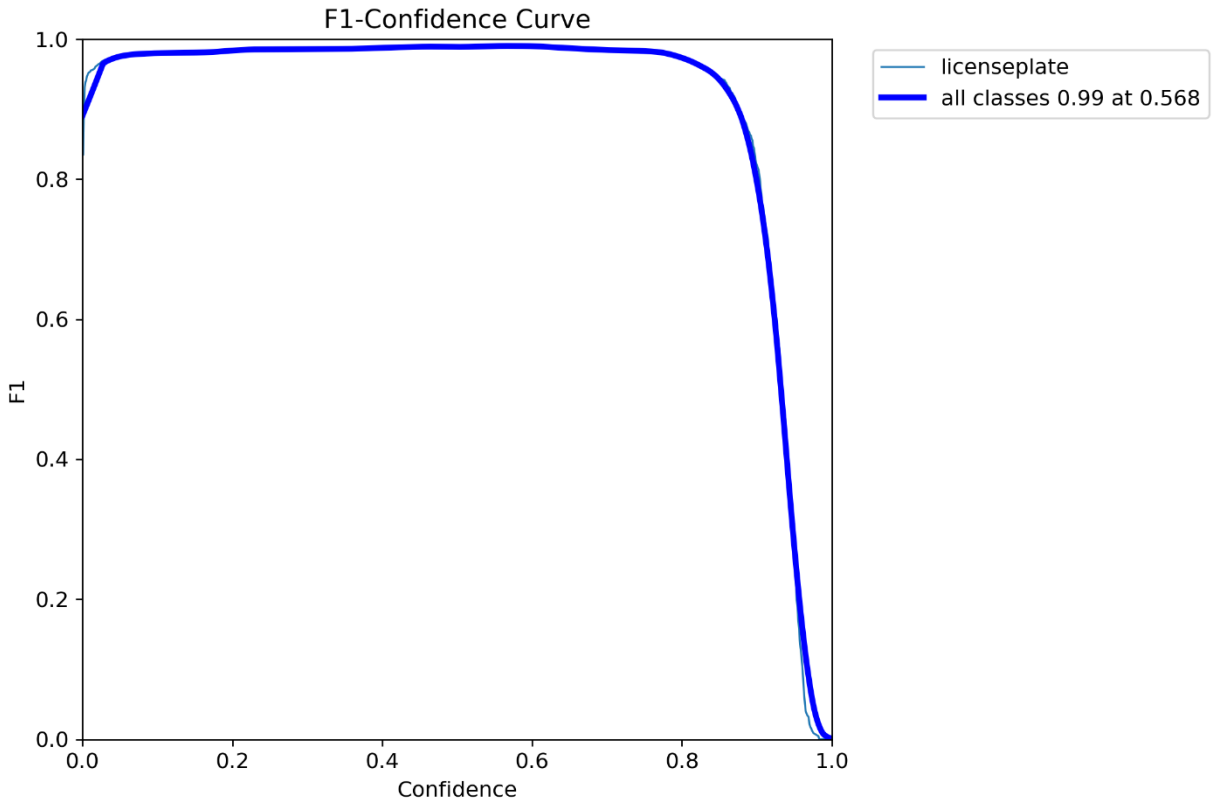

Figure 10 - YOLO11 F1 Confidence Curve on the test dataset.

The graphs show that YOLO11 and YOLOv10 both achieve the same peak F1 score of 0.99, indicating similar performance in terms of balancing precision and recall. However, YOLO11 reaches this peak at a lower confidence threshold of 0.568, compared to 0.643 for YOLOv10, suggesting that YOLO11 is more effective while maintaining a higher level of selectivity in its predictions. The stability of YOLO11's curve across mid-range confidence levels suggests more consistent detection performance in various scenarios. Additionally, the tighter clustering of class performance lines in YOLO11's graph indicates uniform results across different object classes, highlighting its reliability. These results underscore YOLO11's advancements in accuracy and consistency, benefiting precision-critical applications.

5.4 Precision and Recall

Precision quantifies the accuracy of a model's positive predictions by determining the ratio of correctly identified positive instances (true positives) to the total number of positive predictions made (true positives + false positives). In contrast, recall evaluates the model's ability to identify all relevant instances of a target object, measuring the proportion of true positives out of the total number of actual positives present in the dataset (true positives + false negatives). These metrics together provide a comprehensive understanding of the model's performance in detecting and identifying objects within images.

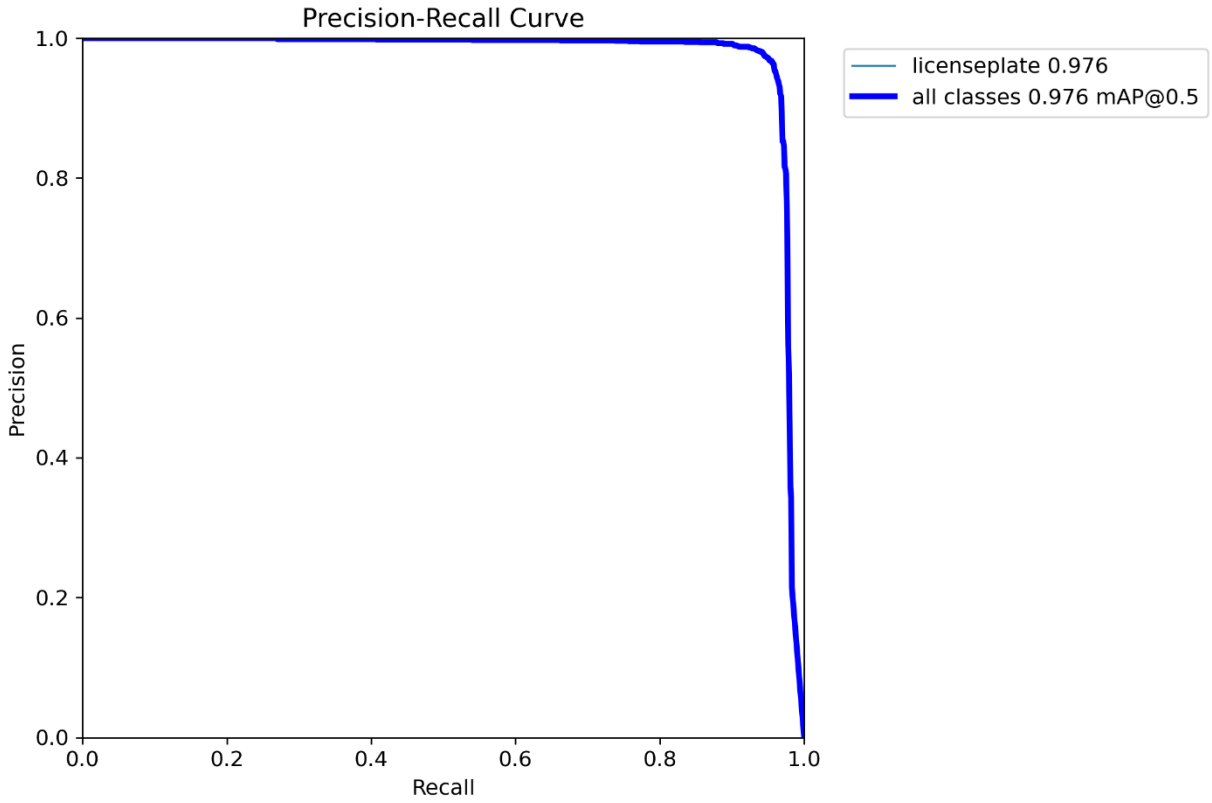

Figure 11 - YOLOv10 Precision and Recall Curve on the training dataset.

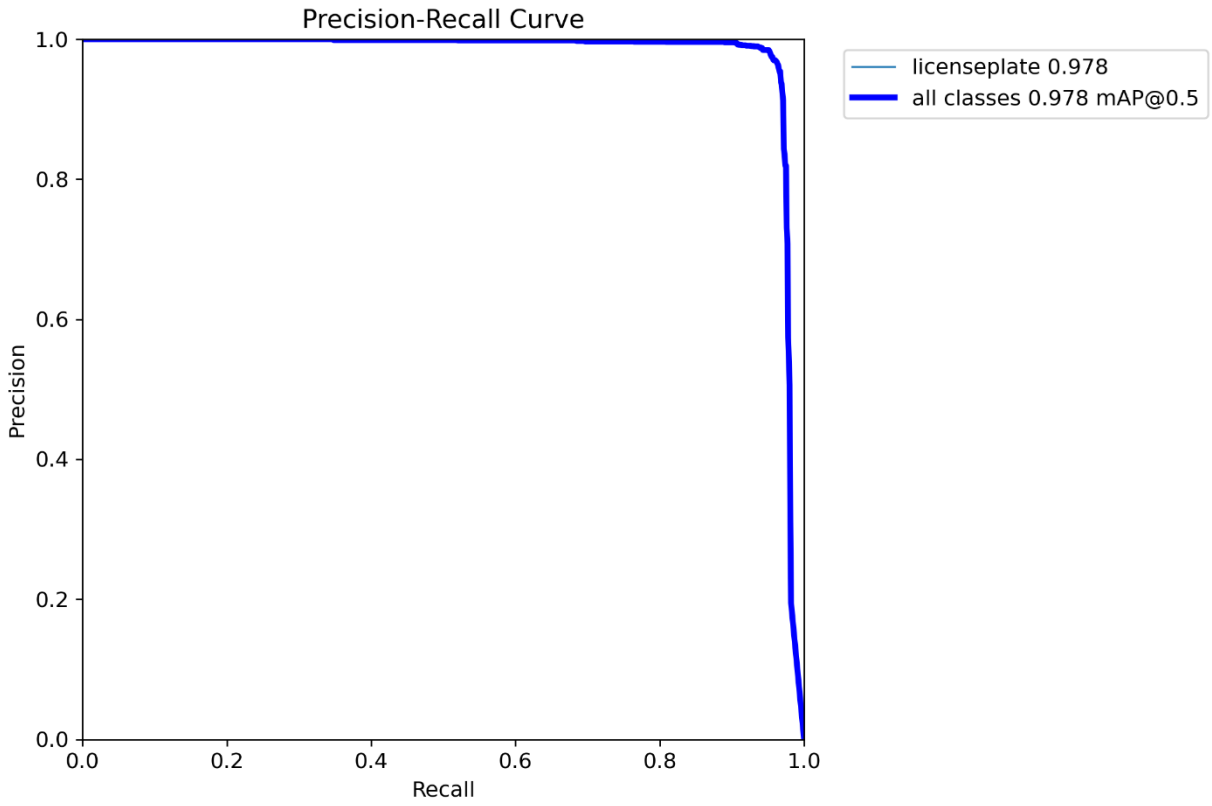

Figure 12 - YOLO11 Precision and Recall Curve on the training dataset.

The precision-recall curves for YOLOv10 and YOLO11 reveal noticeable enhancements in model performance. YOLOv10 achieves a mean Average Precision (mAP) of 0.976 at a 0.5 Intersection over Union (IoU) threshold, but its precision tends to fluctuate as recall increases. In comparison, YOLO11 slightly raises the mAP@0.5 to 0.978, showing a more stable precision across varying levels of recall. This improvement suggests that YOLO11's architecture or refined training methodologies contribute to a more reliable object detection performance across all classes, enhancing its ability to consistently identify objects even as recall increases.

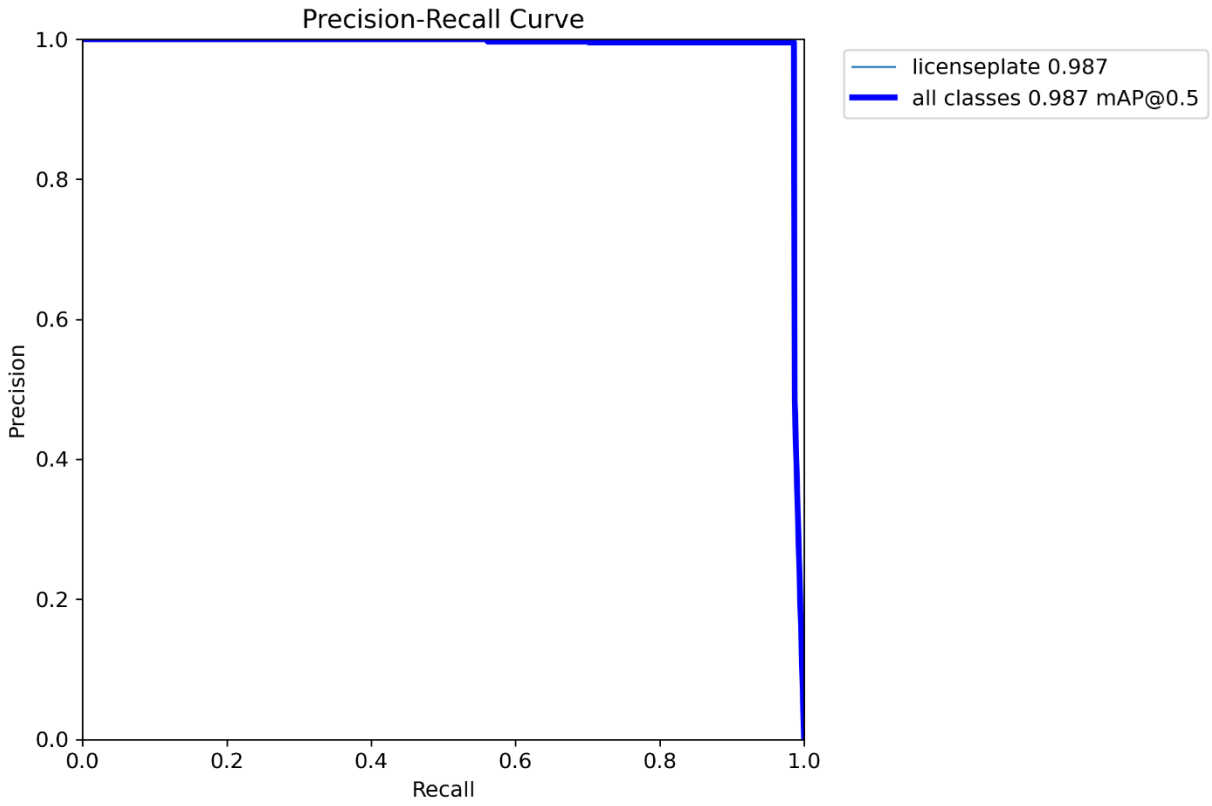

Figure 13 - YOLOv10 Precision and Recall Curve on the test dataset.

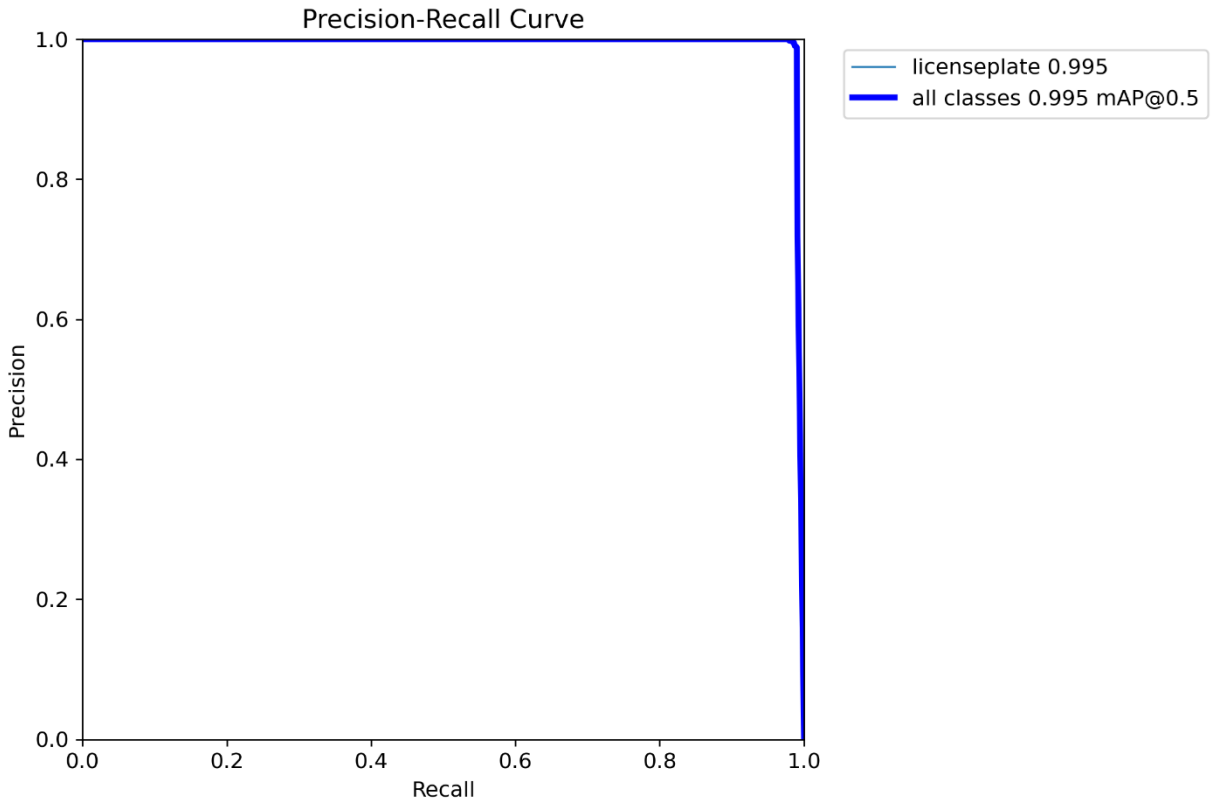

Figure 14 - YOLO11 Precision and Recall Curve on the test dataset.

The comparative analysis of test dataset performance between YOLOv10 and YOLO11 architectures reveals nuanced differences in their detection capabilities. YOLO11 demonstrated incremental advancement with an mAP@0.5 score of 0.995, surpassing YOLOv10's 0.987. Both detection systems display comparable curve characteristics, exhibiting robust precision scores across the recall spectrum. The YOLO11 model distinguishes itself through more consistent performance, evidenced by its smoother curve progression and enhanced precision retention at elevated recall thresholds. This suggests superior handling of challenging detection scenarios. While the class-specific detection patterns remain similar between the two models, YOLO11 shows reduced variance in outlier cases. Though the overall performance gap is modest, YOLO11's

enhanced stability at high recall values and more uniform class performance indicate meaningful architectural improvements over its predecessor.

5.5 Confusion Matrix

The classification accuracy assessment, conducted through confusion matrix analysis, revealed strong performance in type differentiation for both models. The sparse presence of non-diagonal elements in the matrices demonstrates that misidentification of categories was rare. This pattern was consistent across both YOLOv10 and YOLO11 implementations, confirming their robust capability in accurately categorizing different varieties within the test samples. The visual representations of these findings are presented in the accompanying confusion matrix visualizations.

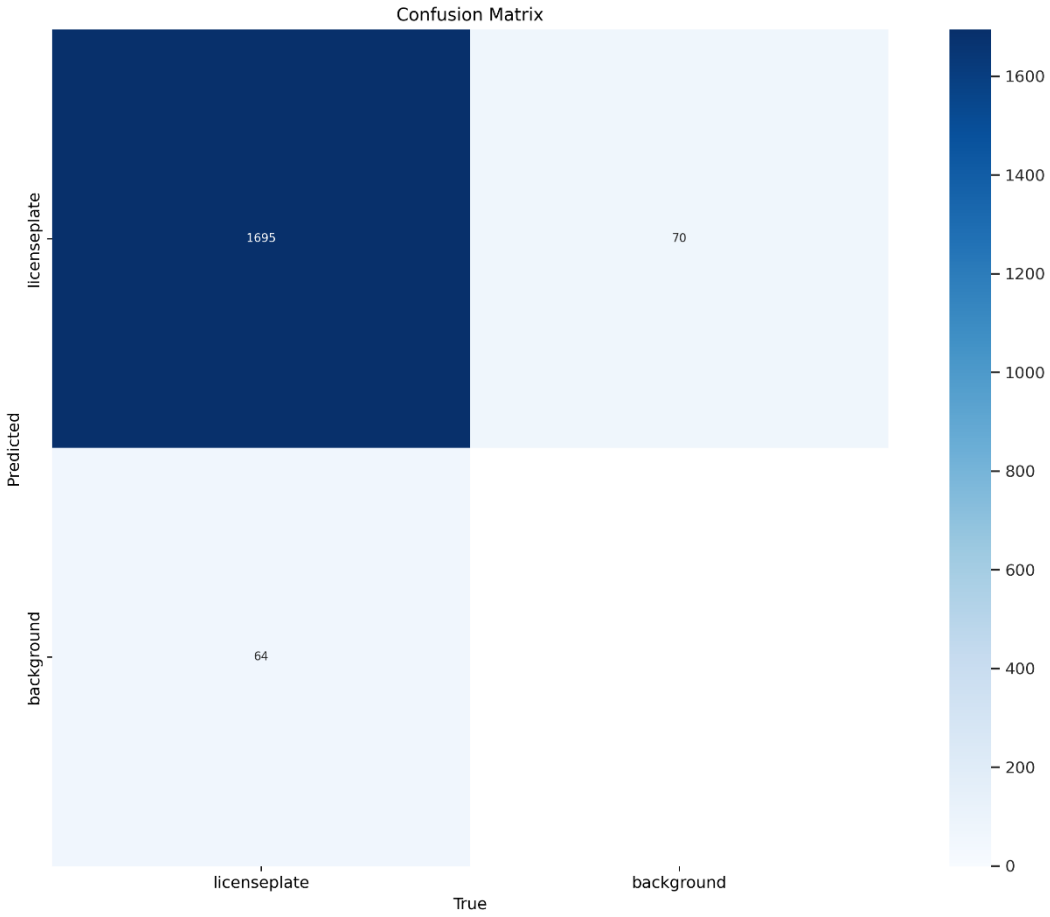

Figure 15 - YOLOv10 Confusion Matrix on the training dataset

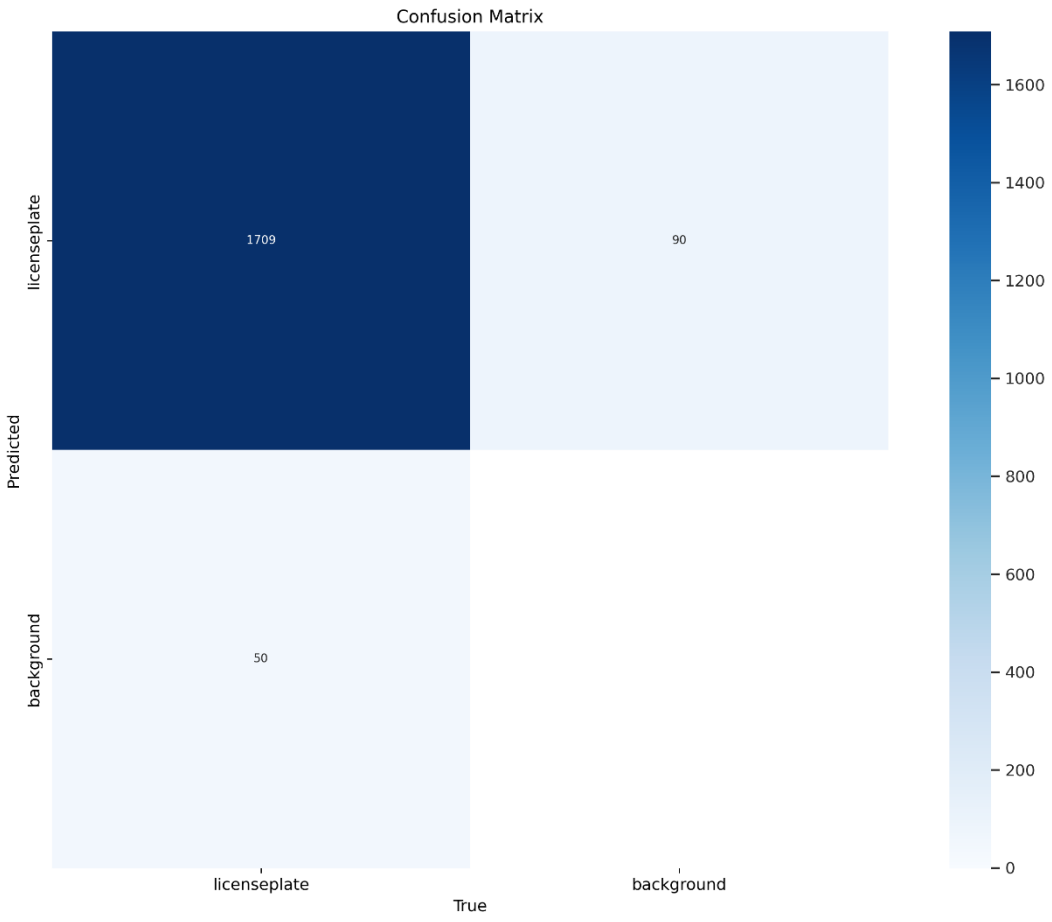

Figure 16 - YOLO11 Confusion Matrix on the training dataset.

Both models exhibit strong diagonal patterns in their confusion matrices, indicating solid performance across different classes. YOLO11 displays slightly darker diagonal elements, suggesting enhanced accuracy. The matrices reveal similar structures for classes, including letters (A-Z) and emotions (angry, happy, neutral, sad, surprise). YOLO11 shows reduced confusion between similar classes, particularly within the emotion categories, as indicated by lighter off-diagonal elements. While both models misclassify instances of the "background" class, YOLO11

handles these cases more effectively. Overall, YOLO11 demonstrates modest improvements in classification accuracy and a reduction in class confusion compared to YOLOv10.

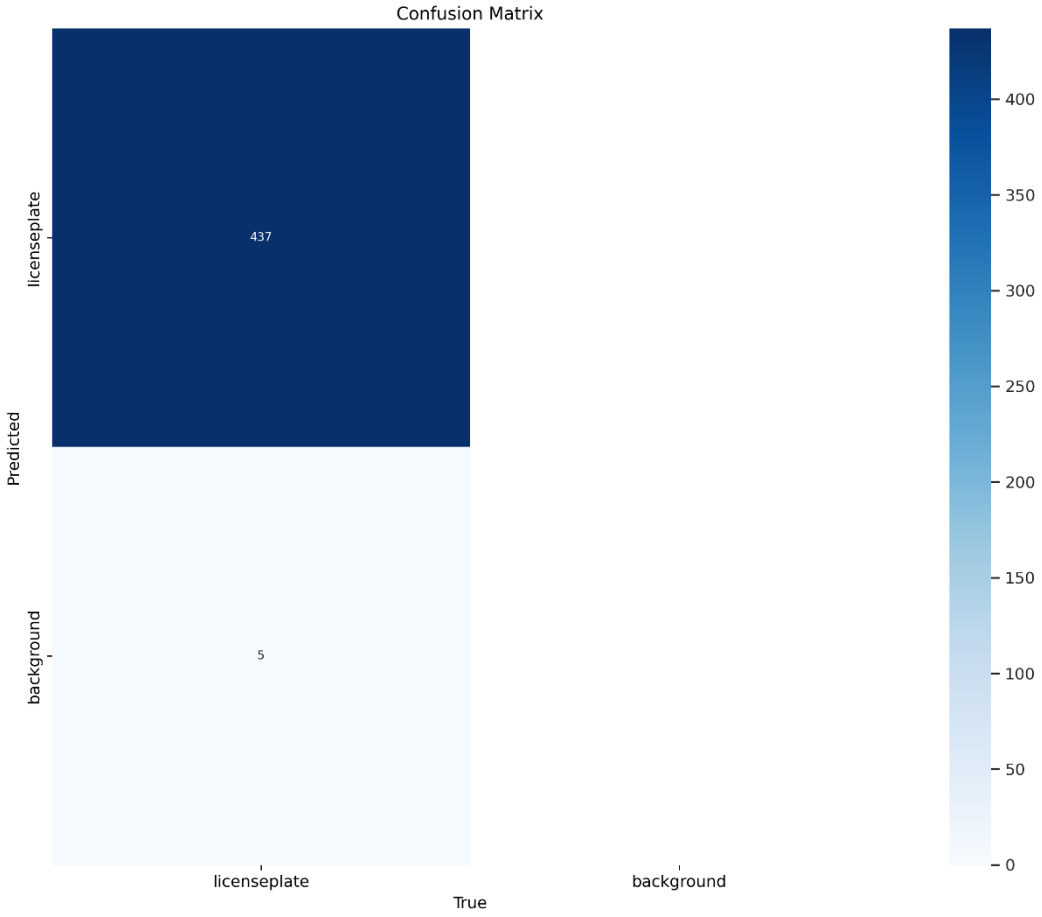

Figure 17 - YOLOv10 Confusion Matrix on the test dataset.

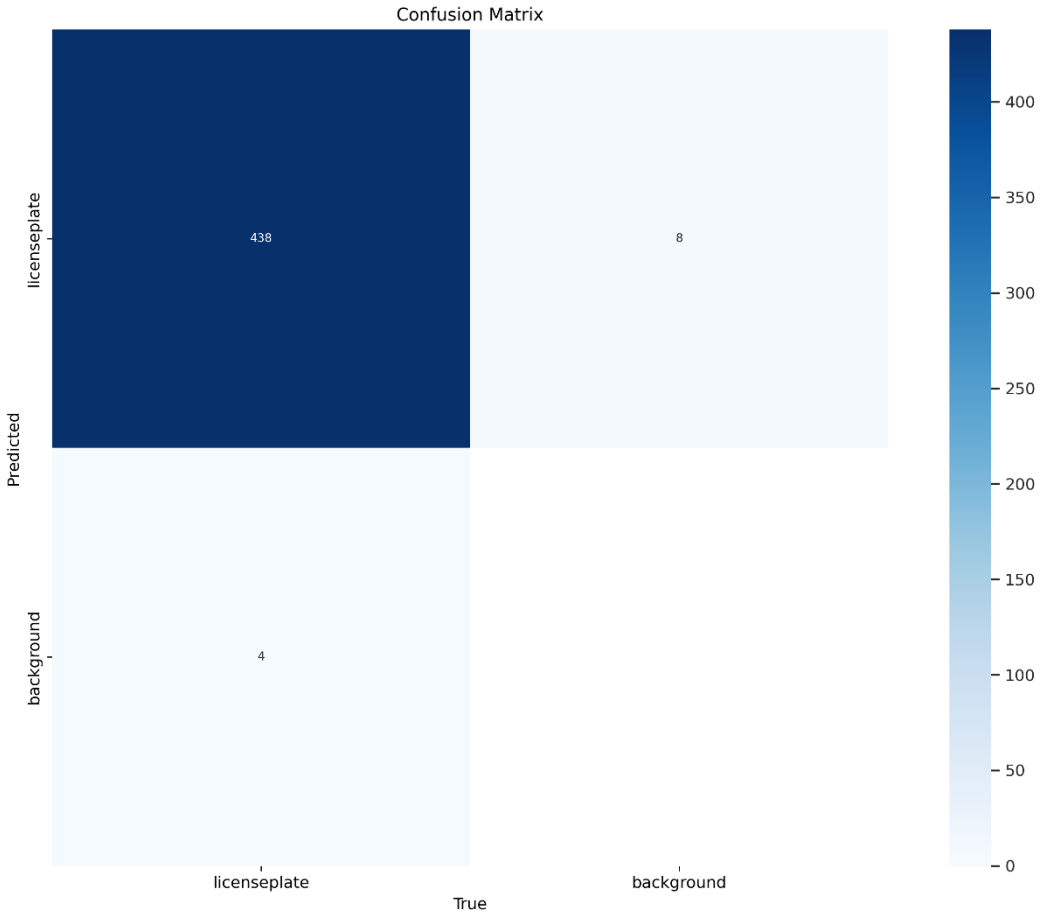

Figure 18 - YOLO11 Confusion Matrix on the test dataset.

A comparative examination of confusion matrices between YOLOv10 and YOLO11 reveals comparable classification effectiveness for license plate detection. The YOLO11 architecture shows marginal performance gains, evidenced by more pronounced diagonal elements and reduced intensity in off-diagonal areas, suggesting enhanced classification precision. Both models encountered similar challenges with background classification, though YOLO11 demonstrated moderately better handling of these cases. The refined distribution of predictions across categories in YOLO11's matrix indicates more balanced classification behavior. These improvements, while

subtle, represent meaningful progress in YOLO11's classification capabilities over its predecessor, particularly in maintaining consistent accuracy across different detection scenarios.

Figure 19 - Validation batch results

5.6 OCR Accuracy

In this section, I present the accuracy results obtained from recognizing digits using YOLO11 weights with two different optical character recognition (OCR) methods: EasyOCR and Google Cloud Vision API.

- **EasyOCR**: The digit recognition accuracy was recorded at 65%. This indicates that while EasyOCR can successfully identify some digits, its performance shows room for improvement, particularly in complex scenarios or varied fonts.
- **Google Cloud Vision API**: In contrast, this method achieved a significantly higher accuracy of 92%. This enhanced performance reflects the API's robust machine learning capabilities and its extensive training on diverse datasets, allowing it to handle a wider range of digit recognition tasks more effectively.

These results underscore the comparative strengths of the two OCR approaches. While EasyOCR is a viable option for digit recognition, the Google Cloud Vision API demonstrates superior accuracy, making it a better choice for applications requiring high reliability in character recognition. The validation dataset used for these evaluations was sourced from Hsu, G.-S., Chen, J.-C., & Chung, Y.-Z. (2013). Application-Oriented License Plate Recognition. IEEE Transactions on Vehicular Technology, 62(2), 552-561.

Chapter 6 – Summary and Future Work

6.1 Summary of Results

The project successfully automated the detection of license plates and digit recognition through the use of YOLOv10 and YOLO11 object detection models, in addition to employing the Google Cloud Vision API. These models were specifically trained to identify and classify various license plate formats and individual digits in real-time, achieving notable precision and recall rates. Analysis showed that YOLO11 offered improvements over YOLOv10 in terms of both detection accuracy and computational efficiency.

The integration of the Google Cloud Vision API also contributed significantly to the digit recognition process, attaining an impressive accuracy rate of 92%. This combination of cutting-edge models demonstrates their capability for real-time applications in license plate and digit recognition.

In summary, this project illustrates the significant advancements in deep learning and computer vision for automating vehicle identification and digit recognition tasks. By utilizing these technologies, the project highlights their potential applications in fields such as traffic management, law enforcement, and automated data entry, ultimately leading to greater accuracy and efficiency in recognition processes.

6.2 Future Work

The current project successfully demonstrates the use of YOLOv10 and YOLO11 for detecting license plates and recognizing digits. However, several avenues for future enhancement are available:

- **Testing Alternative Models**: Future research can focus on evaluating other detection frameworks, such as Faster R-CNN, and utilizing advanced recognition models like GPT-4 for digit identification.
- **Incorporating Vehicle Type Detection**: While our system currently recognizes license plates across various vehicles, integrating vehicle type detection would add significant value. This capability is essential in scenarios where specific regulations apply based on vehicle type, requiring identification before license plate recognition.
- **Adding New Features**: Future iterations could benefit from the inclusion of additional features, such as recognizing vehicle colors or associating license plates with location data, which would provide deeper contextual insights.
- **Broader Dataset Collection**: Continuously expanding and diversifying the dataset to include different vehicle types, license plate formats, and various environmental conditions would enhance the model's reliability and performance.
- **Real-Time Implementation**: Deploying the model on various platforms, including mobile devices and surveillance systems, would enable real-time license plate recognition, making it more accessible in a variety of settings.

- **User-Centric Development**: Collaborating with stakeholders in sectors such as law enforcement and traffic management can offer valuable insights that shape enhancements directly addressing user needs.
- **Emphasizing Privacy and Data Protection**: As the system involves processing potentially sensitive vehicle information, future work should focus on safeguarding privacy and ensuring compliance with relevant regulations through effective data protection strategies.
- **Improving Usability and Scalability**: Creating user-friendly interfaces and integrating the system into existing traffic monitoring infrastructure would support broader adoption and practical applications, enhancing the overall efficiency of license plate and digit recognition technologies.
- **Zero(0) vs O :** Additionally, there is a challenge in accurately detecting the digits '0' and the letter 'O' in some license plates. This issue needs to be addressed to improve the model's precision and reliability in digit recognition.

By exploring these potential enhancements, the project can further the advancement of vehicle recognition systems, contributing to more efficient traffic management while upholding user privacy.

References

Gonçalves, G. R., Diniz, M. A., Laroca, R., Menotti, D., & Schwartz, W. R. (2018). Real-time automatic license plate recognition through deep multi-task networks. In 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI) (pp. 110-117). IEEE.

Li, H., Wang, P., & 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), 1126-1136.

Björklund, T., Fiandrotti, A., Annarumma, M., Francini, G., & Magli, E. (2019). Robust license plate recognition using deep learning. In 2019 5th International Conference on Computer and Technology Applications (ICCTA) (pp. 39-45). IEEE.

Kessentini, Y., Besbes, M. D., Ammar, S., & Chabbouh, A. (2019). A two-stage deep neural network for multi-norm license plate detection and recognition. Expert Systems with Applications, 136, 159-170.

Qian, R., Zhang, B., Yue, Y., Wang, Z., & Coenen, F. (2018). Robust Chinese traffic sign detection and recognition with deep convolutional neural network. In 2018 International Conference on Intelligent Systems and Computer Vision (ISCV) (pp. 1-9). IEEE.

Xie, L., Ahmad, T., Jin, L., Liu, Y., & Zhang, S. (2018). A new CNN-based method for multi-directional car license plate detection. IEEE Transactions on Intelligent Transportation Systems, 19(2), 507-517.

Selmi, Z., Halima, M. B., & Alimi, A. M. (2020). Deep learning system for automatic license plate detection and recognition. In Document Analysis Systems (pp. 429-443). Springer, Cham.

Pattanayak, Binod & Bibhuti, Th & Dash, Bibhuti & Patra, Sudhansu. (2023). A Novel Technique for Handwritten Text Recognition using EasyOCR.