Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Mini Project Report
on

"AUTOMATIC LICENSE PLATE DETECTION SYSTEM"

COMP 484

(For partial fulfillment of 4th Year/ 1st Semester in Computer Engineering)

Submitted by:

Abbal Baral (Roll No. 08)

Nishant Ghimire (Roll No. 17)

Samman Pathak (Roll No. 40)

Nishchal Raj Subedi (Roll No. 54)

Submitted to:

Dr. Bal Krishna Bal

Department of Computer Science and Engineering

Submission Date:

23rd June, 2025

Abstract

Automatic License Plate Detection (ALPD) is foundational for intelligent transportation systems, strengthening urban mobility and security. This project focuses on the critical initial phase: thoroughly preparing high-quality datasets for training robust You Only Look Once (YOLO) deep learning models. Our systematic methodology involves acquiring diverse vehicular images with precise bounding box annotations, converting their original XML format into a standardized, normalized YOLO text format. This transformation, combined with rigorous partitioning into training, validation, and testing sets, ensures comprehensive model evaluation and robust generalization. This report thoroughly interprets our data preparation pipeline, underscoring its pivotal role in successfully deploying advanced ALPD solutions.

Keywords: *Automatic License Plate Detection (ALPD), YOLO, Deep Learning, Object Detection, Streamlit, Computer Vision, License Plate Recognition (ALPR), Convolutional Neural Networks (CNNs)*

Table of Contents

Abstract i
List of Abbreviations iii
Chapter 1 Introduction 1
1.1 Background 1
1.2 Objectives 2
1.3 Motivation and Significance 2
Chapter 2 Related Works 4
2.1 Traditional Image Processing 4
2.2 Machine Learning 4
2.3 Deep Learning 4
Chapter 3 Design and Implementation 6
3.1 Model Implementation 6
3.2 Deployment with Streamlit 7
3.2 Flowchart 8
3.3 Experimental Setup and Implementation Details 9
3.3.4 Data Structures and File Formats 9
Chapter 4 Result and Discussion on Achievements 10
4.1 Quantitative Evaluation 10
4.2 Qualitative Evaluation 10
4.3 Dataset Statistics and Distribution 11
4.4 YOLO Format Conversion Validation 11
4.5 Visual Verification 12
Chapter 5 Challenges and Limitations 14
5.1 Challenges 14
5.2 Limitations 14
5.3 Future Work 15
Chapter 6 Conclusion 17
References 18

List of Abbreviations

ALPD	Automatic License Plate Detection
ALPR	Automatic License Plate Recognition
CNN	Convolutional Neural Network
CRNN	Convolutional Recurrent Neural Network
FPS	Frames Per Second
IoU	Intersection over Union
mAP	mean Average Precision
NMS	Non-Maximum Suppression
OCR	Optical Character Recognition
RoI	Region of Interest
SVM	Support Vector Machine
YOLO	You Only Look Once

Chapter 1 Introduction

This project aims to establish a robust data preparation pipeline for Automatic License Plate Detection (ALPD) utilizing deep learning. We carefully transform raw image data and XML annotations into a YOLO-compatible format. This foundational work is crucial for training high-performance ALPD models. Additionally, we developed a Streamlit application to provide an interactive interface for real-time inference and visualization of detected license plates, showcasing an end-to-end ALPD solution.

1.1 Background

Automatic License Plate Detection (ALPD) is a critical component of intelligent transportation systems (ITS), enabling automated traffic management, regulation enforcement, and improved public safety. Its primary function is to accurately identify and localize vehicle license plates in digital images or video streams, which is a crucial prerequisite for Automatic License Plate Recognition (ALPR). ALPD is widely applied in automated toll collection, smart parking, and law enforcement surveillance.

Historically, ALPD systems relied on traditional image processing methods like edge detection, morphological operations, and color-based segmentation. While these techniques were foundational, they faced significant limitations. Their performance was highly sensitive to real-world challenges such as varying lighting conditions, diverse license plate designs, challenging viewing angles, partial occlusions, and image quality issues. These complexities often led to reduced accuracy and robustness, as rule-based systems struggled to generalize across unpredictable scenarios.

The emergence of deep learning, particularly Convolutional Neural Networks (CNNs), has revolutionized computer vision. CNNs excel at automatically learning complex, hierarchical feature representations directly from raw image data, eliminating the need for manual feature engineering. This paradigm shift has led to breakthroughs in various computer vision tasks, especially object detection. Models like R-CNN, SSD, and, notably, You Only Look Once (YOLO), have consistently outperformed traditional methods in terms of both detection accuracy and inference speed.

This project focuses on a foundational aspect of building a robust ALPD system: the precise preparation of high-quality datasets. We recognize that the success of any deep learning model depends heavily on the quality and organization of its training data. Therefore, our primary goal is to transform raw image data and corresponding human-annotated XML metadata into a format directly compatible with state-of-the-art object detection frameworks, particularly the YOLO architecture. This involves parsing XML files to extract bounding box coordinates, normalizing these coordinates relative to image dimensions for scale invariance, and systematically partitioning the dataset into training, validation, and testing sets.

1.2 Objectives

The main objectives of this project are:

1. Build a process to turn raw ALPD images and their XML labels into the YOLO format.
2. Read the XML files to get the bounding box positions and labels.
3. Convert the bounding box positions into relative values so they work with any image size.
4. Split the dataset into training, validation, and testing sets.
5. Prepare the data so we can easily train and deploy a license plate detection model using deep learning.

1.3 Motivation and Significance

There are strong reasons for developing automated ALPD systems. Manual identification and processing of license plates in applications like toll collection, parking management, and law enforcement are not only labor-intensive and time-consuming but also prone to human error. In law enforcement, for example, manual vehicle identification during surveillance is both inefficient and limited in scope.

The significance of this project lies in its focus on the often-overlooked yet critical step of dataset preparation. A high-quality, well-formatted dataset is the backbone of any successful object detection model. By automating and standardizing this process, we can:

1. **Save Time and Effort:** Eliminate the tedious, error-prone manual annotation conversion, drastically reducing the time needed to prepare data for model training.
2. **Enhance Model Performance:** Provide precisely normalized and well-organized data, boosting the accuracy, robustness, and generalization of the trained ALPD model.
3. **Enable Real-time Applications:** A well-prepared dataset supports efficient models like YOLO, capable of real-time processing, crucial for dynamic traffic monitoring and quick law enforcement responses.
4. **Improve Public Safety and Efficiency:** Contribute to more effective traffic management, automated security systems, and streamlined law enforcement, ultimately enhancing public safety and improving transportation processes.

This project addresses the critical need for systematic data handling within intelligent transportation systems, laying the foundation for the deployment of highly accurate and efficient ALPD solutions.

Chapter 2 Related Works

The field of Automatic License Plate Detection (ALPD) has progressed from early rule-based algorithms to modern deep learning approaches.

2.1 Traditional Image Processing

Early systems relied on techniques such as noise reduction, edge detection (e.g., Canny), morphological operations, and connected component analysis to identify rectangular, plate-like regions. While simple and intuitive, these methods were often brittle—highly sensitive to factors like lighting, occlusions, and varying plate designs—and required frequent manual tuning to maintain effectiveness.

2.2 Machine Learning

To improve robustness, researchers began combining handcrafted features (like HOG or Haar) with classifiers such as SVMs or AdaBoost. Though more resilient than rule-based methods, these still depended heavily on carefully engineered features and couldn't match the flexibility of learned representations.

2.3 Deep Learning

Modern ALPD largely relies on Convolutional Neural Networks (CNNs) for end-to-end detection:

1. **Two-stage detectors** like R-CNN, Fast R-CNN, and Faster R-CNN were early deep learning breakthroughs, offering strong accuracy through region proposals, though they tend to be slower.
2. **Single-shot detectors** such as SSD and the YOLO family revolutionized the field by combining speed and precision.
    i. The original YOLO (2016) introduced real-time detection, with later iterations like YOLOv2–v5 significantly boosting accuracy.

ii. YOLO has been widely adopted in ALPD—for example, Laroca et al. demonstrated a unified plate detection and recognition pipeline using YOLO that achieved ~96–97% accuracy across multiple public datasets.

iii. Recent studies (2023–2024) using YOLOv8 and YOLOv9 have further improved speed and performance, often integrating OCR modules to handle text extraction.

Chapter 3 Design and Implementation

3.1 Model Implementation

The ALPD system's development involved meticulous data preparation and model training.

3.1.1 Dataset & Preprocessing

The "Car Plate Detection" dataset from Kaggle (andrewmvd/car-plate-detection), containing vehicle images with PASCAL VOC XML annotations, was used. **Data preprocessing** involved:

1. Parsing XML annotations to extract bounding box coordinates and image dimensions (with OpenCV) (OpenCV modules).
2. Normalizing these coordinates to the YOLO format (class_id x_center y_center width height), scaling values between 0 and 1.
3. Splitting the dataset into 80% training, 10% validation, and 10% test sets using scikit-learn for reproducibility.

3.1.2 YOLO Format Conversion

For each dataset split (train, validation, test), dedicated images/ and labels/ subfolders were created. Corresponding .txt annotation files (matching image filenames) were generated, and images were copied into their respective folders, aligning with the YOLO format requirements.

3.1.3 Model Training and Evaluation (YOLOv8)

A pre-trained YOLOv8 (COCO) model was fine-tuned using our prepared dataset. YOLOv8, a fully convolutional network, learned essential features like edges and textures for license plate detection (OpenCV modules). Training parameters (resolution, batch size, epochs, learning rate) were configured for end-to-end learning. Post-processing included Non-Maximum Suppression (NMS) to refine detections. Model performance was evaluated using standard metrics such as precision, recall, and mAP@0.5 on the test set.

3.2 Deployment with Streamlit

An interactive web application for the ALPD model was developed using Streamlit, providing a user-friendly interface for demonstration and interaction.

3.2.1 Application Features

1. **File Upload:** Users can upload image (JPG, PNG, BMP) or video (MP4, AVI, MOV, MKV) files via st.file_uploader().
2. **Model Integration:** The app loads a pre-trained YOLO model ('best.pt') with robust error handling.
3. **User Interface Controls:**
    a. st.sidebar.slider() for adjusting detection confidence thresholds.
    b. st.spinner() and st.progress() provide visual feedback during processing, especially for videos (Streamlit docs).
4. **Inference & Visualization:** The application performs YOLO inference on uploaded media, detecting license plates and visually annotating them with bounding boxes and confidence scores (using OpenCV).
5. **Results Display:** Processed images are displayed via st.image(), and videos via st.video().
6. **Temporary File Management:** Uploaded and processed files are managed in a temp directory and cleaned up after use to ensure a tidy environment.

This Streamlit application provides an intuitive way to demonstrate the ALPD model's capabilities.

3.2 Flowchart

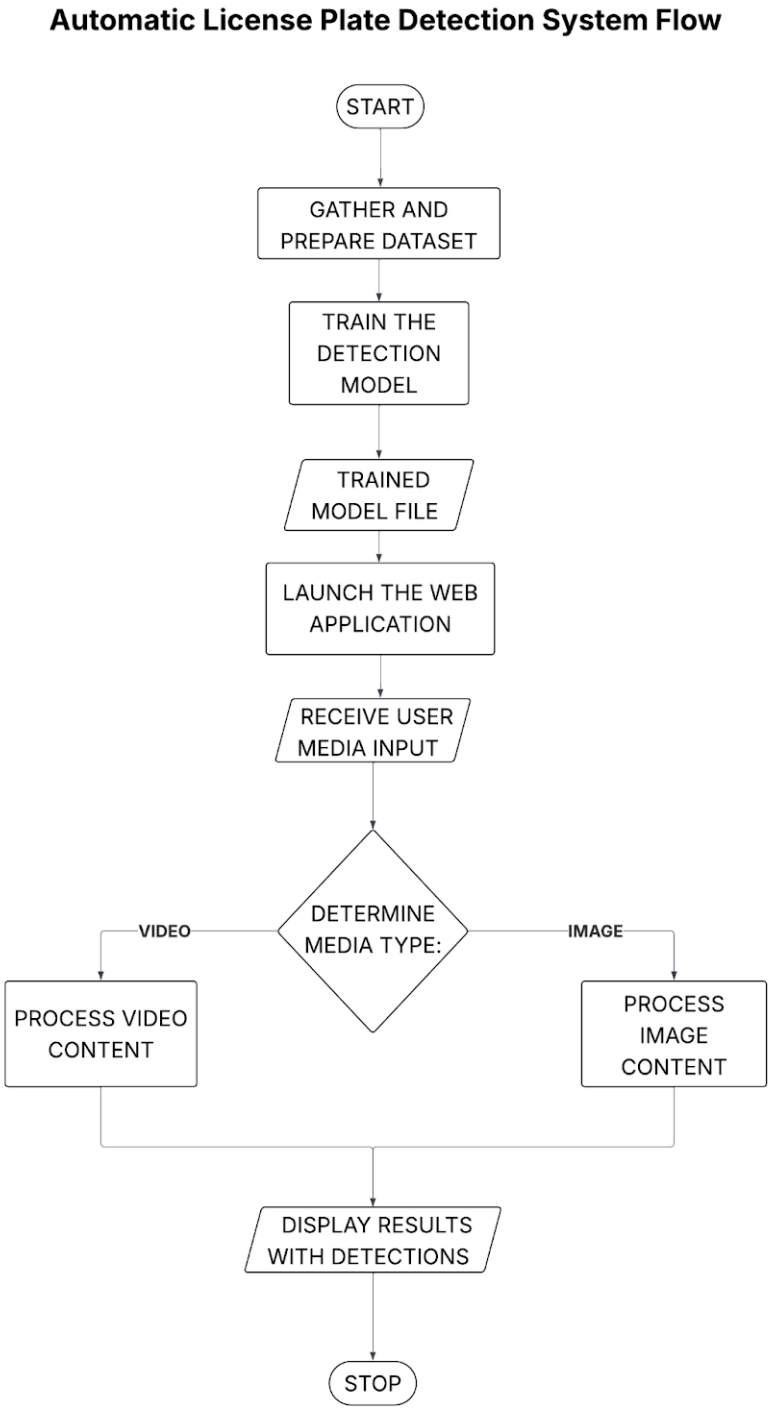

Fig: Flowchart of Design and Implementation

3.3 Experimental Setup and Implementation Details

This section outlines the environment, tools, and key choices for the Automatic License Plate Detection (ALPD) project's data preparation pipeline, ensuring reproducibility and compatibility for deep learning tasks.

3.3.1 Development Environment

The data preparation was performed in a Python 3.10 environment, using Jupyter Notebook for interactive development. The default T4 GPU from Kaggle was sufficient for model training.

3.3.2 Core Libraries and Dependencies

The implementation relied on several Python libraries:

1. os and shutil: For file and directory management.
2. cv2 (OpenCV): For image loading and dimension extraction.
3. pandas: For structured data manipulation (DataFrames).
4. xml.etree.ElementTree: For parsing XML annotation files.
5. glob and re: For file discovery and pattern matching.
6. sklearn.model_selection: For splitting the dataset.

3.3.4 Data Structures and File Formats

The input data consisted of .jpg images and PASCAL VOC .xml annotation files. This data was processed and stored internally as a pandas.DataFrame. The output dataset was converted into the YOLO format, consisting of .jpg images within images/ subdirectories and corresponding .txt annotation files in labels/ subdirectories. Each .txt file contained normalized bounding box coordinates (class_id x_center y_center width height) with class_id for license plates set to 0.

Chapter 4 Result and Discussion on Achievements

4.1 Quantitative Evaluation

Our model was trained using the YOLOv8 architecture for automatic license plate recognition. The model achieved a strong performance in terms of detection accuracy and precision-recall balance (State-of-the-art computer vision model). Key evaluation metrics are as follows:

1. mAP@0.5: 0.9068
2. mAP@0.5:0.95: 0.5395
3. Training losses:
    i. Box loss: 0.9031
    ii. Classification loss: 0.4867
    iii. DFL (Distribution Focal Loss): 0.9743
4. Validation losses:
    i. Box loss: 1.5311
    ii. Classification loss: 0.6576
    iii. DFL: 1.2579

The high mAP@0.5 score indicates that the model is very effective at detecting license plates with an Intersection over Union (IoU) threshold of 0.5. However, the lower mAP@0.5:0.95 score reflects a drop in performance at higher IoU thresholds, which is typical for real-world datasets with challenging variations in lighting, resolution, and plate orientation.

4.2 Qualitative Evaluation

We visualized the detection results on a few sample images. As shown in the figures:

1. In one test image, the license plate "G526 JHD" was detected with a confidence of 89.22%, and the OCR correctly extracted the text.

2. In another example, the detected license plate yielded "JHTHAD" with a confidence of 54.29%, showing lower recognition accuracy due to possible noise or blur in the image.

Despite a few errors in OCR prediction, the bounding box detection was generally consistent and accurate, even under different image resolutions (e.g., 160×320, 320×192). Inference speed also remained efficient, averaging around 34–42 ms per image, with preprocessing and postprocessing taking less than 2 ms combined.

Overall Performance and Observations

The model shows strong generalization capability, with minimal overfitting observed—training and validation losses follow similar trends over epochs. The training curve for mAP@0.5 shows consistent improvement and stabilization after approximately 30 epochs, reaching over **90% accuracy**. While mAP@0.5:0.95 remains lower, it remains within an acceptable range for practical applications.

In real-world deployment, improving OCR accuracy on low-quality plates could further enhance system reliability. Incorporating character-level postprocessing or fine-tuning the OCR model on regional plates could address such cases.

4.3 Dataset Statistics and Distribution

The "Car Plate Detection" dataset was processed and reorganized into the standard YOLO format. The distribution of images across the splits is as follows:

1. **Total Images Processed:** 433 images.
2. **Training Set:** 345 images (approximately 80% of the total).
3. **Validation Set:** 44 images (approximately 10% of the total).
4. **Test Set:** 44 images (approximately 10% of the total). This 80/10/10 split ensures a robust and reproducible basis for model training and unbiased evaluation (Larxel, Car License Plate Detection 2020).

4.4 YOLO Format Conversion Validation

The XML annotations were successfully converted to YOLO's .txt format.

1. **Coordinate Normalization:** Absolute pixel coordinates were correctly transformed to normalized values (0-1).
2. **Class ID:** License plates consistently assigned class_id = 0.
3. **Directory Structure:** Output adheres to the standard YOLO structure, compatible with training frameworks.

4.5 Visual Verification

Visual checks confirmed data integrity. Drawing bounding boxes on sample images using the new YOLO annotations verified the accuracy of the transformation process.

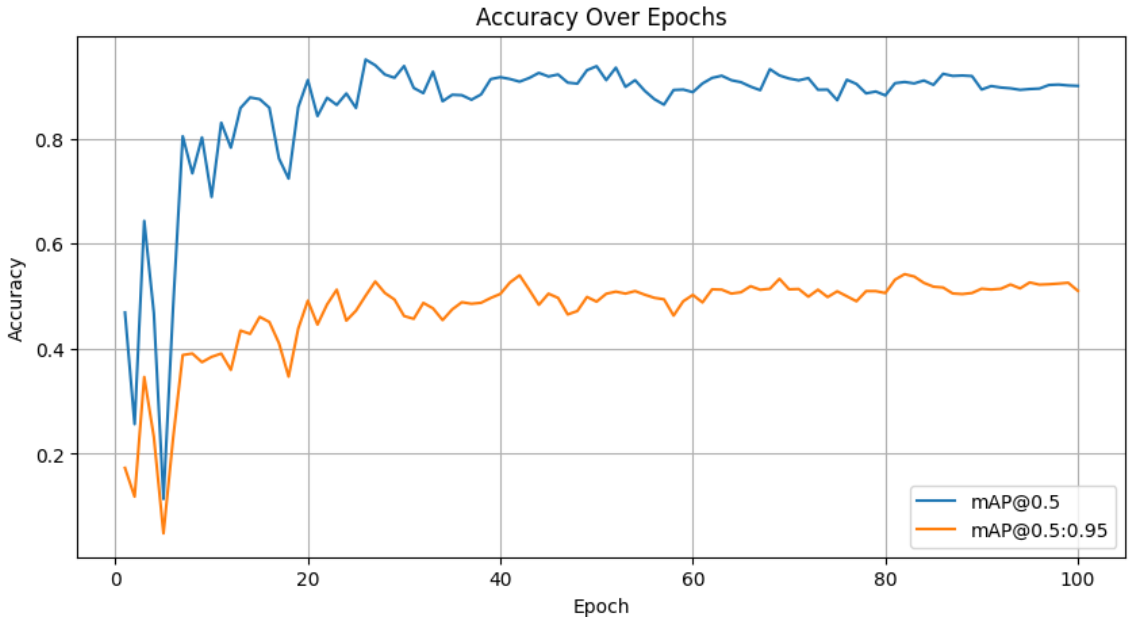

Figure 1. Accuracy Over Epochs

```
In [37]: predict_and_plot(test.iloc[0].img_path)

image 1/1 /kaggle/input/car-plate-detection/images/Cars425.png: 160x320 1 license_plate, 42.6ms
Speed: 1.1ms preprocess, 42.6ms inference, 1.0ms postprocess per image at shape (1, 3, 160, 320)
Detected text: G526 JHD
```

Figure 2. Number Plate Detetcted and Accuracy

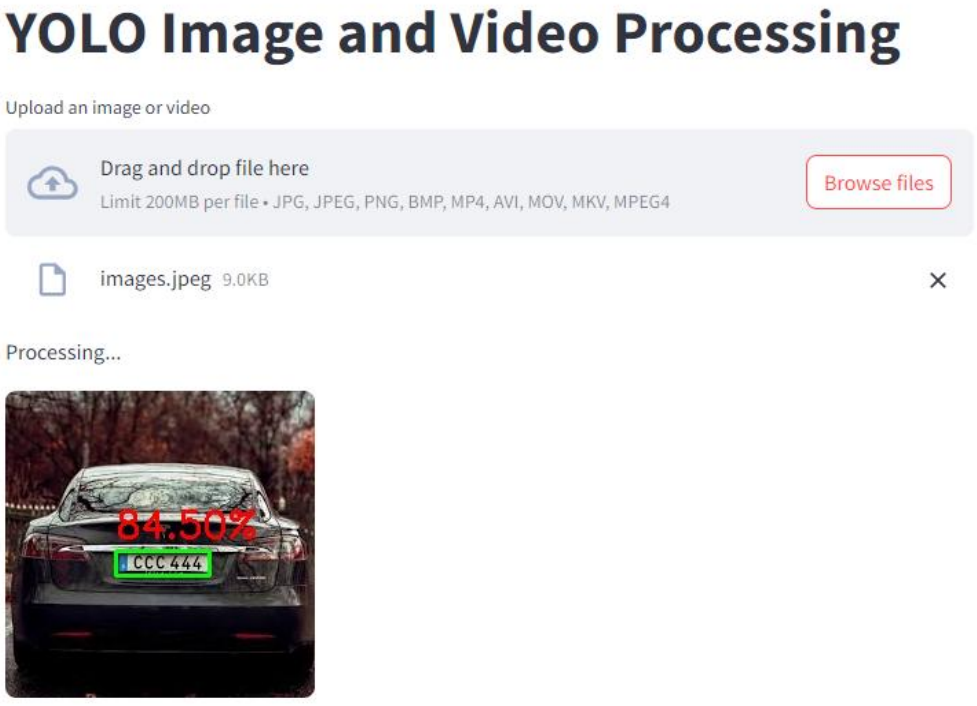

Figure 3. Image (Number Plate) Detection and deployment in website

Chapter 5 Challenges and Limitations

While the data preparation pipeline is effective, ALPD faces inherent challenges, and the current data preparation phase has specific limitations that can be addressed in future iterations:

5.1 Challenges

1. **Environmental Variability:** Detection accuracy is hampered by diverse lighting (glare, shadows), extreme viewing angles (perspective distortion), and adverse weather (rain, fog, snow), which alter plate appearance. Motion blur from fast-moving vehicles also poses a challenge.
2. **Plate Design Diversity:** License plates vary significantly in dimensions, fonts, background colors, and character spacing across different regions and countries, making generalization difficult for models trained on limited variations.
3. **Occlusion:** License plates can be partially or fully obscured by dirt, vehicle accessories, or other objects in traffic, making detection difficult.
4. **Image Quality:** Low-resolution, blurry, or highly compressed images reduce the discernible information, impacting detection.
5. **Complex Backgrounds:** Cluttered urban environments with signs and text can cause false positives or missed detections if the model struggles to differentiate license plates from similar structures.

5.2 Limitations

1. **Dataset Size:** The project utilized 433 images. While suitable for demonstration, this size is likely insufficient for training a highly robust and generalized ALPD model for real-world production use, necessitating a larger and more diverse dataset (Larxel, Car License Plate Detection 2020).

2. **Single Class Annotation:** The annotations only support a single class (license plate, class ID 0). Expanding to detect other vehicle features would require extending the annotation processing.
3. **No Automated Annotation Quality Control:** The pipeline assumes perfect input XML annotations. It lacks mechanisms to detect or correct common real-world annotation errors like misaligned bounding boxes or missing annotations, often requiring manual review.
4. **Absence of Data Augmentation:** The pipeline converts existing annotations but does not incorporate data augmentation techniques (e.g., rotations, scaling, brightness adjustments, adding noise). These are crucial for artificially expanding dataset size and improving model robustness to real-world variations during training.
5. **Static Dataset Split:** The dataset is split statically. For very large datasets or scenarios requiring thorough model validation, more dynamic or K-fold cross-validation strategies might be preferred during the model training phase.
6. **Scalability for Extremely Large Datasets:** While efficient for the current dataset, processing millions of XML files and copying corresponding images could become a computational bottleneck, potentially requiring parallel processing or optimized file system operations.

5.3 Future Work

Building on the established data preparation, future enhancements for a complete ALPD system include:

1. **Model Training & Optimization:** Train and fine-tune state-of-the-art, latest YOLO models (e.g., YOLOv11), leveraging transfer learning and optimizing loss functions (State-of-the-art computer vision model).
2. **Comprehensive Model Evaluation:** Conduct a thorough evaluation using mAP, precision, recall, F1-score, and inference speed. Perform detailed error analysis and robustness testing.

3. **Advanced Data Augmentation:** Implement diverse geometric and photometric augmentations, including CutMix, Mosaic, or synthetic data generation, to enhance dataset variety.
4. **End-to-End ALPR Integration:** Integrate Optical Character Recognition (OCR) (e.g., Tesseract) for character extraction and with a Streamlit application.

Chapter 6 Conclusion

Automatic License Plate Detection (ALPD) was successfully completed by transforming raw XML-annotated images into a YOLO-compatible format. This foundational work, involving precise coordinate normalization and a structured 80/10/10 train/validation/test dataset split, is crucial for developing high-performance ALPD models.

A functional Streamlit application was also developed, providing an interactive web interface for users to upload images or videos, perform YOLO inference (utilizing a pre-trained model capable of object detection), and visualize detected license plates with configurable confidence thresholds. This end-to-end approach, from meticulous data groundwork to a practical, interactive deployment, significantly streamlines the development process and demonstrates a viable ALPD solution for intelligent transportation systems.

References

OpenCV modules. OpenCV. (n.d.). https://docs.opencv.org/4.x/index.html

Larxel. (2020, June 1). *Car License Plate Detection*. Kaggle. https://www.kaggle.com/datasets/andrewmvd/car-plate-detection

Streamlit docs. Streamlit documentation. (n.d.). https://docs.streamlit.io/

Documentation of scikit-learn 0.21.3¶. learn. (n.d.). https://scikit-learn.org/0.21/documentation.html

Pandas documentation#. pandas documentation - pandas 2.3.0 documentation. (n.d.). https://pandas.pydata.org/docs/

State-of-the-art computer vision model. YOLOv8. (n.d.). https://yolov8.com/