A Comparative Study of YOLOv8 and YOLOv12 Models for License Plate detection using the CCPD Dataset

Banasmita Jena, Aryan Shaiju, Adhithi M, Shashi Kumar
School of Computer Science and Engineering, RV University, Bangalore, India
USN: 1RVU23CSE020, 1RVU23CSE084, 1RVU32CSE104, 1RVU23CSE425

***Abstract*— License plate detection and recognition plays a crucial role in intelligent transport systems, surveillance, and automated vehicle detection. As the need for real time, precise detection increases, deep learning models have become vital in overcoming these challenges. In this project we performed an extensive study on License plate detection using Chinese City Parking Dataset (CCPD). Our approach combines the use of object detection models- YOLOv8 and YOLOv12 with optical character recognition using the EasyOCR python package. We compared the model on the basis of detection precision, accuracy, inference time and OCR Accuracy within the CCPD dataset. Our findings provide valuable insights for choosing an adequate model for real-time deployment AI - based traffic control.**

***Index Terms*—License plate detection, YOLOv8, YOLOv12, EasyOCR, CCPD dataset, Intelligent transportation systems, Deep learning, Object detection, Optical character recognition.**

I. Introduction

With the rapid development of intelligent transport systems and smart cities, automated license plate recognition (ALPR) has become an essential component. Most ALPR systems have two main components: License plate detection and character recognition. Achieving high accuracy in both stages is still a challenge due to various factors such as distortion and varying formats of license plate across different regions.

Recents developments in the field of deep learning, especially object detection and optical character recognition (OCR) Have increased the performance of ALPR systems considerably. Object detection models in the YOLO family are increasingly popular due to their ability to detect real time as well as offer high accuracy. For this project we explore how two YOLO models- YOLOv8 and YOLOv12 perform in the detection of license plates. Each of these models are integrated with EasyOCR, a light and efficient OCR engine for end-to-end license plate recognition.

We utilized the Chinese City Parking Dataset CCPD), a popular benchmark for ALPR tasks that contain license plate images taken under various conditions like different angle, lighting and distances. Our aim is to assess and compare the performance of the two models with respect to detection accuracy, precision, OCR success rate and robustness. This study aims to provide a more practical insight on choosing a model for real world deployment in automated traffic systems.

II. Related Work

Previous work in license plate recognition (LPR) has advanced from traditional computer vision methods to more sophisticated deep learning-based techniques. Initial techniques utilized edge detection, morphological transformations, and old machine learning classifiers like SVMs and KNNs, but such methods were not robust under different lighting conditions, angles and plate types.

With the advent of deep learning, models such as the YOLO family have exhibited great advancements in real time object detection. Recent versions of the YOLO model are faster and more accurate while still being capable of deployment into dynamic settings. There are a few studies that provide a systematic comparison between various YOLO versions coupled with OCR pipelines, which provide opportunities for further investigation in the optimization of end-to-end LPR systems.

III. Methodology

A. Dataset

We used the CCPD (Chinese City Parking Dataset), primarily tailored for license plate detection for this project. The dataset contains several sub-datasets including CCPD-Base, CCPD-DB, CCPD-Blur, CCPD-FN, CCPD-Rotate, CCPD-Tilt, and CCPD-Challenge, which are created to evaluate the robustness of the model under diverse circumstances, such as blur, rotation, tilt, and low visibility.

We transformed the native annotation format into a format compatible with YOLOv8, i.e., normalized bounding box coordinates in the form of <class_id> <x_center> <y_center> <width> <height>. This format was required to train our model with the YOLOv8 architecture, which is tailored for high-speed and high-accuracy detection operations.

B. YOLO Architectures

We evaluate YOLOv8, YOLOv12 architectures for License plate detection:

1. YOLOv8:

YOLOv8 was designed by Ultralytics, has a compact architecture with three primary components: the backbone, neck and head. The backbone uses a CSPDarknet based structure that enhances feature representation using effective convolution and residual connections. The neck uses a Path Aggregation Network (PANet) for feature fusion at multiple scales for the model to effectively handle objects of different sizes. The head takes on anchor free strategy, predicting object centers and bounding boxes directly without using fixed anchors. This decoupled head structure separates the classification and localization tasks, enhancing accuracy and training stability. In general, YOLOv8 strikes a good balance between speed and accuracy and is a good starting point for real-time applications such as license plate detection.

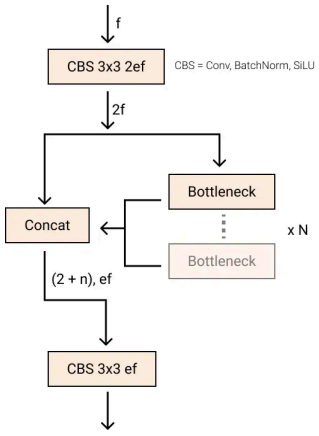


Fig. 1. Visual representation of YOLOv8 C2f model

2. YOLOv12:

Yolov12 is a powerful and smart object detection model it is used to identify license plates on vehicles.It helps in improving older versions with help of CNN and transformers.CNN helps in finding patterns in small areas and transformers gives clear understanding of the image.The combination helps YOLOv12 detect license plates even if images is not that clear or is blurry or taken from various angles.It uses various methods to mix features and focus on important parts of the image.YOLOv12 does not rely on fixed anchor boxes due to which it becomes faster and easier to train.It separates various tasks like finding ,locating and classifying objects due to which its accuracy is improved.
With help of various smart training methods like image mixing .YOLOv12 gives highly accurate results which can be used for traffic monitoring and vehicle tracking when we combine it with OCR tools like EasyOCR

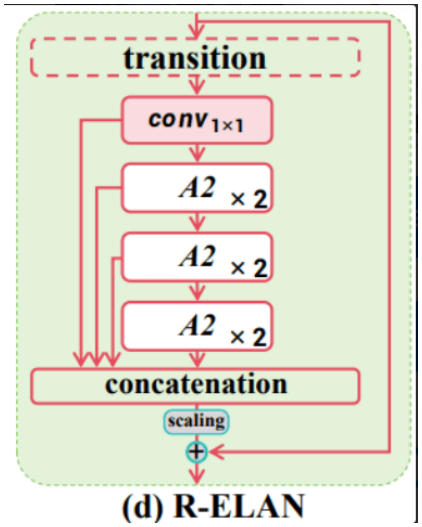


Fig. 2. Visual representation of R-ELAN used in YOLOv12

3. EasyOCR:

EasyOCR is an open-source optical character recognition(OCR) system created with PyTourch that is able to extract text from images easily and efficiently. It can support more than 80 languages and utilizes deep learning based detection and recognition. The architecture has mainly three stages: text detection, text recognition and optional text grouping. For detection, EasyOCR employs the CRAFT(Character Region Awareness for Text Detection) algorithm, which finds the exact location of text regions by detecting individual characters and linking them together. For recognition it uses CRNN (Convolutional Recurrent Neural Network), which integrates CNN for feature extraction, RNN for sequence modeling and a CTC (Connectionist Temporal Classification) decoder for output. This blend enables EasyOCR to process both printed and handwritten text with high accuracy, which makes it ideal for such as license plate recognition when used with object detection models such as YOLO.

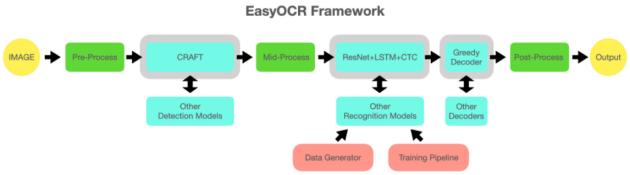


Fig. 3. Visual representation of EasyOCR Framework

C. Training

Training of the CCPD dataset was done using two object detection models, YOLOv8n and YOLOv12s, to test and compare their performance, computational requirements and architecture.

The Ultralytics lightweight model, YOLOv8 was trained with a batch size of 32 with an input image of size 640×640 and 20 epochs. Stochastic gradient descent (SGD) was used for adaptive learning rate and momentum scheduling during the training process as an optimizer. Data augmentation methods like horizontal flipping, blurring, and modifications of brightness were used to improve generalization. The improvement of the training speed and reducing the GPU memory usage was further done by the use of Automatic Mixed Precision (AMP).

The newer YOLOv12s, which has a more complex and deeper architecture, was trained with a batch size of 32 using the same input image size of 640×640 and 15 epochs. The same automated optimization approach, data augmentation method and AMP were used throughout YOLOv12 training.

IV. Results

A. Evaluation

In evaluating the performance of each model for license plate detection and recognition, we employ a comprehensive set of metrics to assess both the accuracy and efficiency of the overall system.

1. Accuracy:

This metric measures the model's performance in accurately identifying the license plate characters. It is determined as the percentage of correctly predicted characters over the total number of characters in the test set. The higher the recognition accuracy, the better the performance in correctly reading the license plates.

The accuracy formula is given by:

$$Accuracy = \frac{TP+TN}{TP + TN + FP + FN}$$

- **TP** = True Positives (correctly predicted positive instances)
- **TN** = True Negatives (correctly predicted negative instances)
- **FP** = False Positives (incorrectly predicted positive instances)
- **FN** = False Negatives (incorrectly predicted negative instances)

2. Precision:

Precision measures the ratio of correct positive prediction to all positive predictions made by the model. The higher the precision, the fewer the false positives, the more reliable the model is in detection tasks.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **TP** = True Positives (correctly detected license plates/characters)
- **FP** = False Positives (incorrect detections labeled as license plates/characters)

3. Recall:

Recall measures the model's ability to identify all the correct instances, here, all true license plate characters. The higher the recall, the fewer true instances the model misses (i.e. fewer false negatives).

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **TP** = True Positives (correctly detected license plates/characters)
- **FN** = False Negatives (incorrectly predicted negative instances)

4. F1- Score:

F1-Score is the harmonic mean between precision and recall, yielding a single measure that balances both. It is particularly helpful when there is an unbalanced class distribution, or when false positives and false negatives are both critical.

$$\text{F1 Score} = \frac{2*Precision*Recall}{Precision + Recall}$$

5. Stochastic Gradient Descent:

Stochastic Gradient Descent (SDG) is a form of the classic gradient descent algorithm, in which the weights of the model are updated after every mini-batch or single training sample.

SGD Weight Update Rule:

$$W_{t+1} = W_t - \eta * \nabla L\ (x_i, y_i, W_t)$$

- x_i, y_i: Represents a single training sample and its corresponding label.
- $\nabla L\ (x_i, y_i, W_t)$: Denotes the gradient of the loss function with respect to the model weights, evaluated at the i^{th} training sample.
- η: Learning rate, which controls the step size during the update

6. Computational Cost (GFLOPs):

This gives an indication of the amount of computational work needed per inference. YOLOv8n runs at around 8.2 GFLOPs, which is lower computational work and thus appropriate for light applications. YOLOv12s, on the other hand, needs around 19.6 GFLOPs, which indicates its more advanced architecture and higher resource usage.

$$GFLOPs = \sum_{\text{all layers}} [2 * H_{out} * W_{out} * C_{in} * C_{out} * K^2] \div 10^9$$

- H_{out} , W_{out}: Output height and width
- C_{in}: Input channels
- C_{out}: Output channels
- K: Kernel size

7. Trainable Parameters:

The total number of parameters directly affects the model's learning capacity. A model which has more trainable parameters captures more complex patterns.

$$\text{Total Parameters} = \sum_{\text{all layers}} [(C_{in} * C_{out} * K_2) + C_{out}]$$

- C_{in}: Input channels
- C_{out}: Output channels
- K: Kernel size

8. Model Complexity:

The structure of architectures like YOLO is characterized by model complexity, which is crucial where precision and speed are important. It is measured by the total number of trainable parameters. In our project, this is influenced by the models like YOLOv8n and YOLOv12s. The factors which contribute to the overall complexity of the model are number of layers, kernel sizes and type of connections between the layers. The presence of higher parameter count in YOLOv12s helps the model to capture more detailed features from input images but this in turn increases the computational resource.

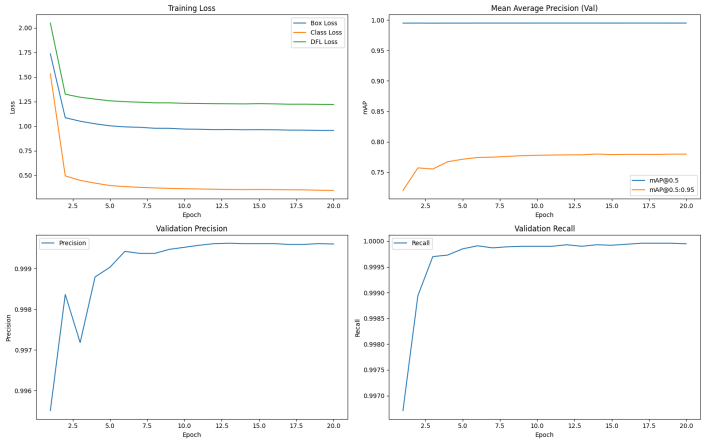
B. Results Analysis

Table: Performance comparisons of YOLOv8n and YOLOv12s

Model	Parameters (≈)	GFLOPs	Layers
YOLOv8n	3,011,043	8.2	129
YOLOv12s	9,096,851	19.6	497

Model	Accuracy	Precision
YOLOv8n	99.89%	99.89
YOLOv12s	99.95%	99.95%

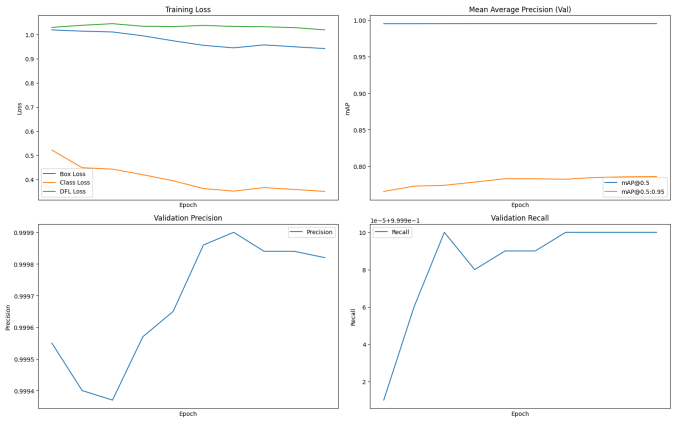
A. YOLOv8n:



Training curves of the YOLO model when used in license plate detection shows effective learning with a steady reduction in training losses such as box loss, class loss and DFL (Distribution Focal Loss). The trend here is that the model is learning the task of identifying and locating license plates from the input images continuously. Also, the fact that there is no randomness or sudden dive suggests that the learning process itself is stable and that the model is not just memorizing during training. Loss values converging at later stages ensure that the model has become optimal in the training process.

On the validation side, the model performs very well with high precision and recall values close to 1.0, and have mAP (mean Average Precision) also show stable performance. The almost identical training and validation signify that the model generalizes very well and not overfitting. Overfitting usually leads to high training performance with bad validation results, whereas in this case both remain high and consistent.

B. YOLOv12s:



The consistent decreasing training losses and the stable high performance across validation metrics shows that the YOLOv12 training results have no signs of overfitting. The validation mAP@0.5 is ~0.99 throughout all 15 epochs and mAP@0.5:0.95 steadily increases which indicates that the model is improving at localizing license plates with stricter IoU thresholds. Validation precision and recall both are remaining high (~0.999), showing the model's accuracy in detecting true license plates with minimal false positives or negatives.

IV. Conclusion

In this study, we tested and compared YOLOv8n, YOLOv12s and EasyOCR for license plate detection and recognition.YOLOv8n, with its light architecture, provides faster inference with fewer computations, hence being ideal for real-time execution on edge devices. However, due to its compact architecture and fewer parameters it might underperform in complex detection tasks.

Contrariwise, YOLOv12s with its deeper model and greater quantity of trainable parameters has much superior accuracy and reliability. It can better capture subtle features in images, which works extremely well to precisely identify license plates even when there is some adverse condition such as motion blur or insufficient light. Although computationally more intensive, the added performance makes it worthwhile.

EasyOCR complements these models by efficiently recognizing the text from the identified license plates. While it might not be as precise in detection as YOLO models, its power is in precise character recognition after a plate has been localized. The combination of YOLOv12s for detection and EasyOCR for text recognition creates a very efficient pipeline for license plate detection systems, providing a good trade-off between accuracy, speed and reliability.

V. Acknowledgements

We would like to express our sincere gratitude to our esteemed professor Dr. Shabbeer Basha S H from the School of Computer Science and Engineering at RV University.

VI. References

[1] Shenghu Pan, Jian Liu and Dekun Chen, “Research on License Plate Detection and Recognition System based on YOLOv7 and LPRNet,” *January* 2023. [Online]. Available: Link

[2] Zahra Ebrahimi Vargoorani and Ching Yee Suen, “Licence Plate Detection and Character Recognition Using Deep Learning and Font Evaluation”, 17 December 2024, [Online], Available: Link

[3] Rayson Laroca, Luiz A. Zanlorensi, Gabriel R. Gonçalves, Eduardo Todt, William Robson Schwartz, David Menotti, “An Efficient and Layout-Independent Automatic License Plate REcognition System Based on the YOLO detector”, 4 September 2019, [Online], Available: Link