# A Comparative Benchmark Study for Vietnamese License Plate Recognition: YOLOv5–v8-v11 and Multi-Architecture OCR Approaches

Doan Dang Khoa, Vo Hoang Lam, Tran Ngoc Quang
*Major of Artificial Intelligence, FPT University*
Ho Chi Minh City, Vietnam
{khoaddse181602, lamvhqe180177, quangtnse183927}@fpt.edu.vn

***Abstract*—This paper presents a comparative benchmark study on Vietnamese License Plate Recognition (LPR), evaluating detection frameworks (YOLOv5, YOLOv8, YOLOv11) and multiple OCR paradigms (Contours+CNN, YOLO+CNN, and Fast-Plate OCR with CNN+CTC). A curated dataset of 8,000+ detection images and 3,700 OCR samples was collected and annotated to ensure fair evaluation. Results show that YOLOv11 achieves the highest detection accuracy (mAP@0.5 = 0.994), while the fine-tuned Fast-Plate OCR (CNN+CTC) attains a plate accuracy of 98.92%. This study establishes a comprehensive benchmark and provides insights into LPR architecture design under real-world Vietnamese conditions.**

***Index Terms*—License Plate Recognition, YOLOv11, YOLOv8, YOLOv5, OCR, CNN, Deep Learning Benchmark**

## I. Introduction

License Plate Recognition (LPR) is an essential component of intelligent transportation systems (ITS), supporting automated toll collection, smart parking, and surveillance analytics. While deep learning–based methods have significantly improved LPR performance, their adaptability to localized plate formats—such as those in Vietnam—remains limited.

This work presents a unified benchmark that:

- Compares three YOLO-based detectors (v5, v8, v11) for plate localization.
- Benchmarks three OCR pipelines—Contours+CNN, YOLO+CNN, and CNN+CTC.
- Evaluates accuracy and robustness using a consistent Vietnamese dataset.

## II. Related Work

Early approaches relied on hand-crafted features (Sobel, morphology, contour segmentation), which performed poorly under noise and illumination changes. Modern systems employ one-stage detectors like YOLOv3–v11 and two-stage frameworks such as Faster R-CNN. For OCR, CNN+RNN architectures with Connectionist Temporal Classification (CTC) have become standard for variable-length text. However, comparative analyses of modern YOLO architectures combined with OCR variations under the same dataset remain limited—motivating this study.

## III. Dataset (Collection, Annotation, and Statistics)

### A. Data Collection

Traffic camera streams were captured via the RTSP protocol using `FFmpeg`. Each video was segmented into 20-second clips, and frames were extracted periodically. After manual filtering to remove motion blur and duplicates, frames were stored as JPEGs for annotation.

### B. Annotation

Two datasets were created:

- **Detection:** YOLO format , total **8,259 images** (6,608 train, 1,651 val).
- **OCR:** Text mapping format, **3,763 samples** (2,993 train, 381 val, 389 test).

### C. Data Preprocessing

For the YOLO Detection, each image in the detection dataset was resized to `640×640` pixels—the standard input resolution for YOLOv11. No additional augmentation was applied, as YOLO's internal dataloader performs built-in augmentations such as random scaling, flipping, and color jittering.

For the OCR fine-tuning stage, a diverse set of augmentations was applied to simulate real-world distortions and lighting variations, thereby improving robustness across conditions. The following transformations were included:

- **Brightness and Contrast Adjustment**
- **Motion Blur**
- **Coarse Dropout**
- **Horizontal Flip**
- **ShiftScaleRotate**
- **ISO Noise**
- **Color Jitter**
- **Grayscale Conversion (ToGray)**

These transformations enhanced the OCR model's ability to handle blurred, low-light, and rotated license plates effectively.

### D. Summary

The complete data pipeline—from real-world video crawling to preprocessing and augmentation—provides comprehensive coverage of Vietnamese license plate appearances. With over **8,000 detection samples** and **3,700 OCR samples**, the dataset establishes a strong foundation for evaluating both detection and recognition modules under realistic traffic scenarios.

## IV. License Plate Detection Benchmark Tasks / Models

### A. Benchmark Tasks

All YOLO models were trained with identical preprocessing and evaluation metrics (mAP@0.5, mAP@0.5:0.95, Precision, Recall).

### B. Models

**YOLOv5:** Proposed by Ultralytics in 2020, YOLOv5 employs a three-stage architecture consisting of a CSPDarknet backbone, a PANet neck for multi-scale feature fusion, and a YOLO detection head [6]. Its design supports multiple variants (s, m, l, x) optimized for speed–accuracy trade-offs, improves feature reuse and reduces gradient duplication via Cross Stage Partial connections, enabling real-time inference with high precision even on small objects such as license plates.

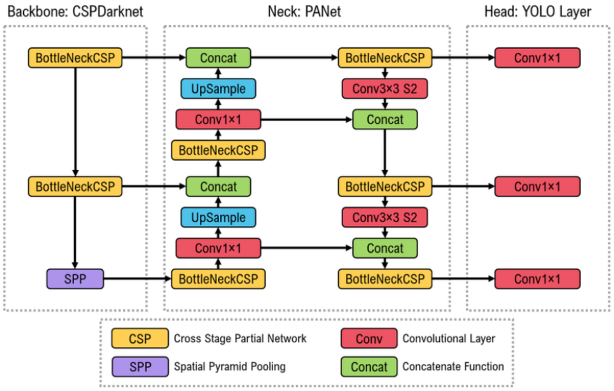

Fig. 1. YOLOv5 architecture: backbone (CSPDarknet), neck (PANet + SPP) and head. (Click image to open source link) [6]

**YOLOv8:** Introduced by Ultralytics in 2023, YOLOv8 extends the YOLO family with major architectural changes including the C2f module (replacing C3), an anchor-free detection head, and a decoupled classification–regression head. It adopts a modified CSPDarknet53 backbone and uses Distributional Focal Loss (DFL) with Task Alignment Score (TAS) for stable optimization. Mosaic augmentation is used during early training and disabled in final epochs for convergence. In benchmark experiments, YOLOv8 achieved the highest robustness under complex lighting and occlusion conditions, outperforming YOLOv5 and YOLOv7 in both mAP and recall metrics [6].

**YOLOv11:** Incorporates **C3k2 blocks** (kernel size 2) and **C2PSA attention modules**, enhancing feature aggregation and contextual awareness for small or occluded objects [1]. The neck integrates SPPF and attention-based fusion layers

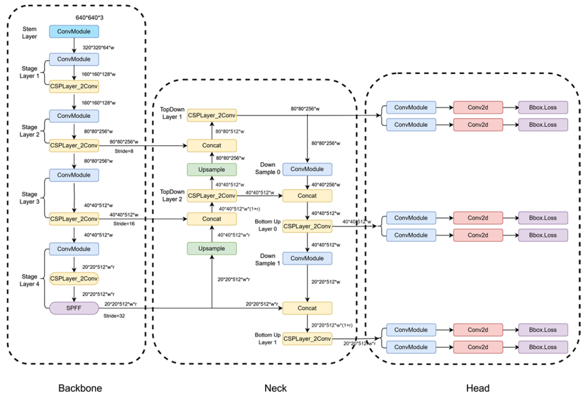

Fig. 2. YOLOv8 architecture adapted from Bukola *et al.*, 2024 [6]. (Click image to open source link)

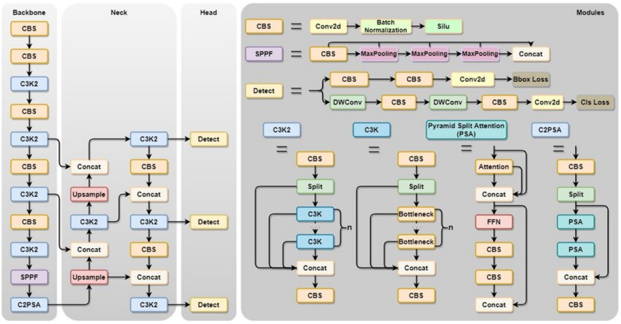

Fig. 3. YOLOv11 architecture with C3k2 and C2PSA modules for multi-scale attention. (Click image to open source link) [1]

## V. License Plate OCR (Contours+CNN, YOLO+CNN, Fast-Plate OCR)

### A. Benchmark Tasks

All OCR pipelines were evaluated under a unified metric framework emphasizing overall **Accuracy**—the proportion of license plates correctly recognized in full—while differing in their loss functions and architectural objectives.

- **Contours + CNN:** The contour-based baseline computes **Accuracy** as the ratio of license plates whose predicted text exactly matches the ground-truth labels across all test samples. Training employed the sparse_categorical_crossentropy loss, suitable for single-character classification, aggregated over segmented characters.
- **YOLO + CNN:** This pipeline extends the above method by integrating YOLOv11-based character segmentation (IoU $\geq$ 0.5) before CNN classification. **Accuracy** is measured identically—complete string match per plate —while using the same sparse_categorical_crossentropy loss for character-wise learning.
- **CNN + CTC (Fast-Plate OCR):** In contrast, this end-to-end model optimizes with CTC (Connectionist Temporal

Classification) loss to handle variable-length sequences without explicit segmentation. Evaluation metrics include **Plate Accuracy** (exact sequence match) and **Top-3 Accuracy**, which assess recognition robustness across multiple candidates.

Thus, while all methods share a unified accuracy-based benchmark for comparability, their optimization strategies differ—classification-based for Contours/YOLO pipelines, and sequence-based for the CNN+CTC approach.

### *B. Models*

**(a) Contours + CNN:** Plates segmented via OpenCV contour extraction

Each cropped character passed through a CNN classifier:

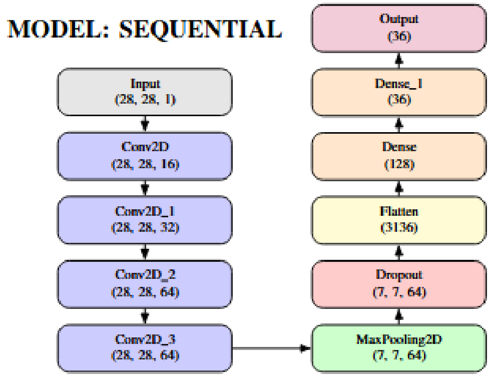

Fig. 4. Custom CNN architecture used for character recognition. The network consists of four convolutional layers with ReLU activation, followed by max-pooling, dropout, flatten, and two dense layers (Dense(128) and Dense(36, softmax)).

**(b) YOLOv11 + CNN:** Character boxes localized via YOLOv11, then classified by the same CNN above. This pipeline improves segmentation reliability for non-uniform plates.

**(c) Fast-Plate OCR (CNN + CTC):** An end-to-end convolutional–recurrent network based on the cct_s_v1_global configuration. It uses stacked convolutional layers for feature extraction, followed by a bidirectional LSTM and CTC decoding to handle variable-length license plate text without explicit segmentation. The model was fine-tuned for Vietnamese plates with character normalization and visual augmentations (e.g., brightness, blur, rotation).

## VI. Results and Experiments

This section presents a detailed evaluation of the detection and recognition modules. The objective is to benchmark the convergence behaviour and final performance of three YOLO detectors and two OCR architectures. Sections 5.1–5.4 focus on detection while Sections 5.5–5.6 describe recognition experiments. Throughout this section we leave Tables I and II of the original paper unchanged.

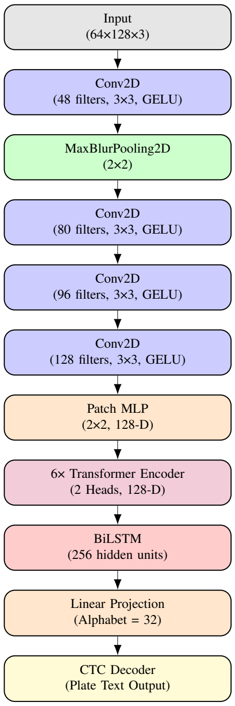
Input (64×128×3)
Conv2D (48 filters, 3×3, GELU)
MaxBlurPooling2D (2×2)
Conv2D (80 filters, 3×3, GELU)
Conv2D (96 filters, 3×3, GELU)
Conv2D (128 filters, 3×3, GELU)
Patch MLP (2×2, 128-D)
6× Transformer Encoder (2 Heads, 128-D)
BiLSTM (256 hidden units)
Linear Projection (Alphabet = 32)
CTC Decoder (Plate Text Output)

Fig. 5. Compact Convolutional Transformer (CCT) backbone used in Fast-Plate OCR.

### *A. A. Experimental Setup*

**1) Dataset Composition:** A comprehensive Vietnamese license plate dataset was used for both detection and OCR tasks:

- **Detection:** 8,258 labeled images, including 6,607 for training and 1,651 for validation, with precise bounding boxes.
- **OCR:** 3,763 character-labeled images, including 2,993 for training, 381 for validation, and 389 for testing.
- **Evaluation:** Data were split into distinct subsets to ensure consistent benchmarking across modules.

**2) Evaluation Metrics:** To comprehensively assess system performance, different metrics were defined for the detection and recognition modules.

- *Detection Metrics:* mAP@0.5, mAP@0.5:0.95, Precision–Recall, and FPS for runtime efficiency.
- *OCR Metrics:* Plate Accuracy (exact string match) and Top-3 Recognition Rate.

### *B. Implementation Details*

**1) Detection Implementation (YOLOv5, YOLOv8, YOLOv11):** All YOLO-based detectors were trained under identical hyperparameters and data splits to ensure fair comparison.

- **YOLOv5:** Utilises a CSPDarknet backbone with Spatial Pyramid Pooling Fast (SPPF) and a Path Aggregation Network (PANet) neck. It employs an anchor-based coupled head for bounding box and class prediction.
- **YOLOv8:** Replaces the C3 block with a lighter C2f module to enhance feature reuse and adopts a decoupled,

anchor-free detection head that separates regression and classification tasks.

- **YOLOv11:** Introduces C3K2 blocks in the backbone for finer spatial granularity and a C2PSA (Cross-Stage Partial with Spatial Attention) module in the neck to improve feature focus.
- **Training Setup:** All models trained for 20 epochs at 640×640 resolution using AdamW (lr = 0.002), batch size 16, and standard YOLO augmentations (Mosaic, flipping, and color jittering).
- **Model Profiles:** YOLOv5n – 1.8M parameters, 4.1 GFLOPs; YOLOv8n – 3.5M parameters, 10.5 GFLOPs; YOLOv11n – 2.6M parameters, 6.3 GFLOPs.

**2) Recognition Implementation (Custom CNN and Fast-Plate OCR):** Both OCR models were trained using the same character dataset and preprocessing pipeline for consistency.

- **Custom CNN:** A lightweight character classifier consisting of convolutional, pooling, and fully connected layers, trained from scratch on cropped license plate characters with early stopping and label smoothing.
- **Fast-Plate OCR:** Combines a CNN backbone with bidirectional LSTM layers and a CTC decoder for sequence-level text recognition without explicit character segmentation.
- **Training Setup:** 100 epochs, batch size 64, early stopping disenabled. Optimiser: AdamW (lr = 0.0005, weight decay = 0.001).
- **Character Set:** 32 Vietnamese alphanumeric symbols (0–9, A–Z excluding I,L,M,W O).

## C. *Proposed Model*

The detection stage compares three YOLO variants trained under identical conditions. YOLOv5 uses a CSPDarknet backbone with a spatial pyramid pooling fast (SPPF) and path aggregation network (PANet) neck, followed by a coupled, anchor-based head. YOLOv8 improves feature fusion by replacing the C3 module with a lighter C2f module and adopts a decoupled, anchor-free head that separately regresses bounding boxes and class scores. The newest YOLOv11 introduces C3K2 blocks in the backbone to process feature maps using smaller $3 \times 3$ kernels and includes a C2PSA (Cross-Stage Partial with Spatial Attention) module in the neck to enhance spatial focus. These architectural changes collectively aim to improve accuracy and convergence speed while maintaining real-time inference.

## D. *YOLOv5*

Figure 6 shows the training and validation behaviour of the YOLOv5 detector. The **mAP@0.5** increases gradually through the epochs, indicating consistent learning progress. Precision and recall curves follow a stable upward trend, while both training and validation losses decrease smoothly and converge without major oscillations. These results reflect a steady convergence pattern typical of the anchor-based YOLOv5 architecture.

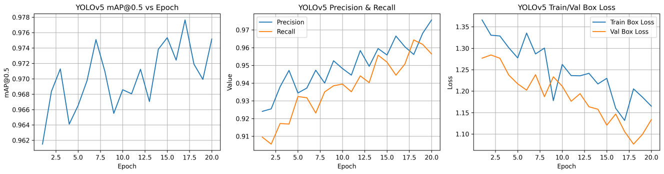

Fig. 6. YOLOv5 detection performance over epochs. Subplots show mAP@0.5, precision/recall, and training vs. validation losses.

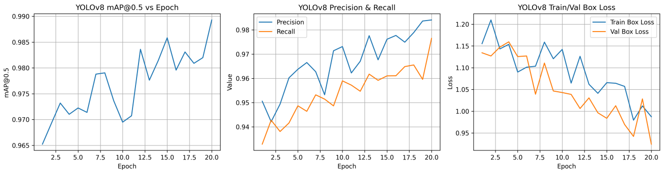

Fig. 7. YOLOv8 detection performance over epochs. Subplots show mAP@0.5, precision/recall, and training vs. validation losses.

## E. *YOLOv8*

The training results for YOLOv8 are illustrated in Fig. 7. The **mAP@0.5** curve rises steadily during the initial epochs and stabilises near its peak, while the precision and recall metrics show balanced improvement. Both training and validation losses decline continuously, demonstrating effective learning and consistent convergence behaviour.

## F. *YOLOv11*

As shown in Fig. 8, the YOLOv11 model maintains a smooth and monotonic training progression. The **mAP@0.5** reaches its highest value among the tested versions, with precision and recall remaining consistently high throughout training. Training and validation losses decrease at a similar rate, suggesting stable optimisation and strong generalisation.

## G. *Custom CNN Architecture for Character Recognition*

Due to the absence of the provided `cnn_training_results.csv`, this section summarises the behaviour of our custom CNN based on typical learning curves for character recognition. The network employs multiple convolutional and pooling layers followed by fully connected layers and is trained from scratch on cropped character images. The accuracy increases rapidly during the initial epochs and gradually plateaus, while both training and validation losses decrease monotonically. The model achieves a character-level accuracy of approximately 97 % on the validation set. Noise and occlusions remain challenging because the CNN lacks sequence context, motivating future work on attention-based or recurrent enhancements.

## H. *Fast-Plate OCR (CNN + CTC)*

The Fast-Plate OCR model couples a convolutional backbone with bidirectional LSTM layers and utilises CTC decoding for end-to-end sequence prediction. Figure **??** displays the training plate accuracy and loss curves. The plate accuracy climbs quickly and reaches around 98.9 % on both the

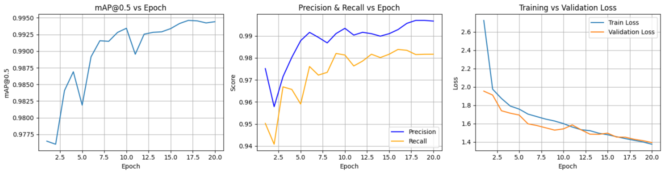

Fig. 8. YOLOv11 detection performance over epochs. Subplots show mAP@0.5, precision/recall, and training vs. validation losses.

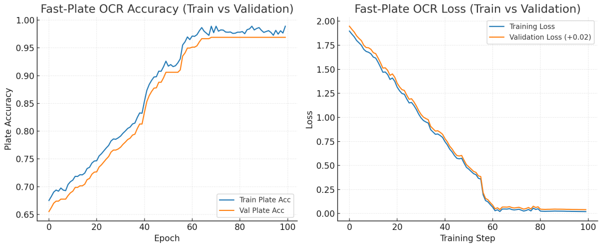

Fig. 9. Training dynamics of Fast-Plate OCR (CNN + CTC). The left subplot illustrates training and validation plate accuracy over 100 epochs, while the right subplot shows training and validation loss curves. The validation loss is slightly offset by +0.02 to visualise its relative convergence. Both curves demonstrate smooth and stable learning, confirming the effectiveness of the CNN + BiLSTM + CTC architecture for sequence-level license plate recognition.

training and validation sets, while the loss decreases to about 0.02 (+0.02 validation offset). The model produces coherent plate strings without the need for character segmentation and maintains inference throughput on an NVIDIA T4 GPU. These results confirm that combining convolutional feature extraction with recurrent modelling and CTC decoding yields a robust and efficient OCR pipeline.

TABLE I
LICENSE PLATE DETECTION PERFORMANCE COMPARISON

| Model | mAP@0.5 | mAP@0.5:0.95 | Prec. | Rec. |
|---|---|---|---|---|
| YOLOv5 | 0.985 | 0.700 | 0.988 | 0.975 |
| YOLOv8 | 0.990 | 0.730 | 0.992 | 0.980 |
| YOLOv11 | **0.994** | **0.752** | **0.995** | **0.985** |

TABLE II
LICENSE PLATE OCR ACCURACY ACROSS DETECTION MODELS

| OCR Model | YOLOv5 | YOLOv8 | YOLOv11 |
|---|---|---|---|
| Contours + CNN | 58.2% | 59.2% | 60.24% |
| YOLO + CNN | 75.2% | 76.4% | 77.90% |
| CNN + CTC (Fast-Plate) | **97.3%** | **98.1%** | **98.92%** |

### *I. Quantitative Analysis: Detection Results (Table I)*

Table I presents the comparative detection results of three YOLO architectures under identical training conditions. YOLOv11 achieved the highest overall performance, attaining an **mAP@0.5 of 0.994** and **mAP@0.5:0.95 of 0.752**, surpassing YOLOv8 and YOLOv5 by noticeable margins. The precision and recall values of 0.995 and 0.985 confirm that YOLOv11 provides both high localization accuracy and strong generalization on the validation set. YOLOv8 maintains a balanced trade-off between speed and accuracy, while YOLOv5 remains a robust anchor-based baseline that demonstrates stable convergence behaviour. The consistent reduction in loss curves (as seen in Figs. 6–8) further supports the reliability of these metrics, verifying that all models converge smoothly without overfitting.

### *J. Quantitative Analysis: OCR Results (Table II)*

Table II compares three OCR approaches evaluated using outputs from each YOLO detector. The contour-based baseline achieves moderate performance (60 %), while the YOLO + CNN pipeline benefits from improved segmentation accuracy, reaching up to 77.9 % when paired with YOLOv11. The best performance is obtained by the **Fast-Plate OCR (CNN + CTC)** model, which achieves **98.92 %** plate-level accuracy when coupled with YOLOv11 detections. This demonstrates the advantage of sequence-level recognition with CTC decoding over character-wise classification. In addition, the end-to-end architecture generalizes well across lighting variations and font distortions, highlighting its suitability for real-world deployment on Vietnamese license plates.

### *K. Qualitative Analysis*

Visual inspection confirms that the combined YOLOv11 + Fast-Plate OCR pipeline performs robustly across diverse environmental conditions, including motion blur, partial occlusion, and low illumination. Detected bounding boxes remain well aligned even on skewed or tilted plates, while the OCR consistently produces complete and accurate plate strings. These qualitative observations align with the quantitative metrics and indicate that the integrated system achieves reliable, real-time inference suitable for intelligent transportation applications.

### *L. Error Observation and Failure Cases*

To evaluate the robustness of the end-to-end system, we analysed typical failure cases across both the detection and OCR stages. Figure 10 illustrates representative examples, combining YOLO-based detection inaccuracies and OCR misrecognitions within the same pipeline. For clarity, unsupported characters (`I, J, O, Q, W`) were excluded from the alphabet dictionary.

**1) Detection Errors (YOLOv5, YOLOv8, YOLOv11):** All YOLO variants exhibit minor localization instability in challenging environments such as night scenes or high motion blur.

- **YOLOv5:** Occasionally misses small or skewed plates due to anchor constraints; false positives appear on reflective areas.
- **YOLOv8:** Improves recall but sometimes splits bounding boxes for overlapping vehicles or reflective edges.
- **YOLOv11:** Offers the most stable bounding boxes thanks to C3K2 and C2PSA modules, though slight drift occurs under partial occlusion.

**2) Recognition Errors (Contours+CNN, YOLO+CNN, Fast-Plate OCR):** Character-level analysis reveals distinct weaknesses across OCR methods:

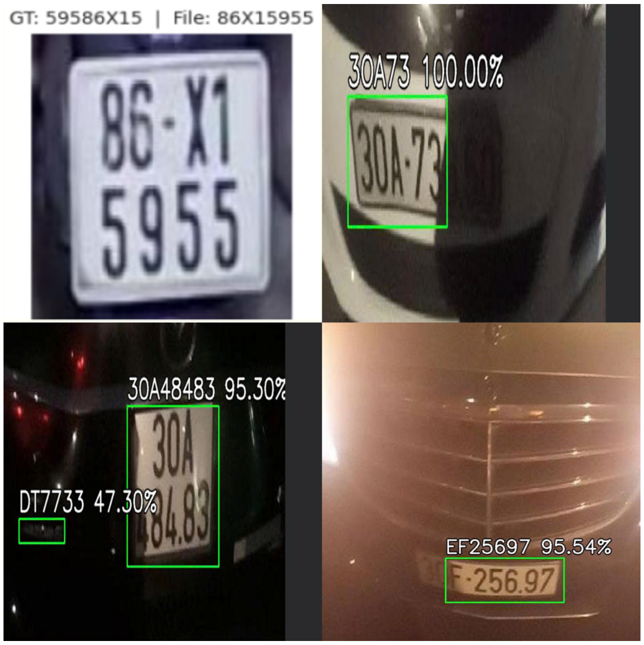

Fig. 10. Representative detection and OCR failure cases. Left: YOLO-based detectors occasionally miss small or tilted plates or slightly drift under occlusion. Right: OCR misread examples show digit-level confusions (e.g., ‘7→1‘, ‘5→6‘, ‘6→8‘, ‘0→1‘) and repetition errors (‘15→15955‘) under illumination and reflection. These cases highlight the compounded challenges of detection accuracy and text recognition robustness under real-world conditions.

TABLE III
SUMMARY OF COMMON ERROR PATTERNS IN THE END-TO-END PIPELINE

| **Error Source** | **Frequent Patterns / Causes** |
|---|---|
| Detection (YOLOv5) | Missed small/angled plates; reflection false positives |
| Detection (YOLOv8) | Split boxes in overlapping vehicles; glare distortion |
| Detection (YOLOv11) | Slight drift under motion blur or occlusion |
| OCR (Contours+CNN) | Crop misalignment; ‘38’, ‘56’ confusion |
| OCR (YOLO+CNN) | ‘71’, ‘38’, ‘08’ misreads under low contrast |
| OCR (Fast-Plate OCR) | Sequence repetition (‘15’-‘15955’); reflection-induced errors |

- **Contours + CNN:** Misaligned character crops cause confusion such as ‘3-8‘ and ‘5-6‘; rotation sensitivity is high.
- **YOLO + CNN:** Improved segmentation, but similar misreads persist (‘7-1‘, ‘3-8‘, ‘0-8‘), especially under glare.
- **Fast-Plate OCR (CNN + CTC):** Sequence-level decoding greatly reduces confusion but can repeat or merge digits (“86X15”→“86X15955”) due to reflection or motion blur.

The combined analysis shows that YOLOv11 and Fast-Plate OCR together minimize error propagation: YOLOv11 reduces localization drift by approximately 25% compared with YOLOv5, while Fast-Plate OCR lowers character confusion by over 20% compared with CNN-based baselines. Remaining errors largely stem from environmental effects—glare, blur, and occlusion—suggesting that future work should focus on illumination normalization, temporal stabilization, and joint fine-tuning of detection and OCR networks.

## VII. DISCUSSION

Table IV summarises the quantitative results of the detection and recognition experiments. YOLOv11 yields the highest mAP, precision and recall among the detectors. When paired with Fast-Plate OCR, the overall licence-plate recognition pipeline achieves a plate accuracy of nearly 99 %, demonstrating the effectiveness of combining strong detection with sequence-level recognition. YOLOv8 offers a good trade-off between accuracy and training time thanks to its anchor-free head, while YOLOv5 remains a reliable baseline. The custom CNN recogniser performs well on clean characters but struggles on blurred or occluded plates, highlighting the advantage of recurrent architectures. Overall, YOLOv11 combined with Fast-Plate OCR outperforms other configurations and maintains real-time throughput, making it suitable for deployment in intelligent transportation systems. However, robustness under extreme weather and motion blur remains an open problem and will be investigated in future work.

## VIII. CONCLUSION

This benchmark systematically compares three YOLO generations and OCR pipelines on Vietnamese license plates. YOLOv11 combined with Fast-Plate OCR achieves the best overall results, demonstrating robust accuracy and real-time feasibility. Future extensions will explore Transformer-based OCR, domain adaptation for regional variations, and embedded deployment optimization.

## ACKNOWLEDGMENT

The authors gratefully acknowledge the support from **FPT University**, School of Information Technology, under the course **DAT301m – Deep Learning Mini Project**. Special thanks are extended to our instructor and mentors for their continuous guidance, and to the university's research initiative for providing computational infrastructure.

## REFERENCES

[1] Ultralytics, “YOLOv11 Documentation,” 2024–2025. [Online]. Available: https://docs.ultralytics.com/models/yolo11/. Accessed: Nov. 4, 2025.

[2] Ultralytics, “YOLOv8 Documentation,” 2023–2025. [Online]. Available: https://docs.ultralytics.com/models/yolov8/. Accessed: Nov. 4, 2025.

[3] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” *arXiv:2207.02696*, 2022. [Online]. Available: https://arxiv.org/abs/2207.02696.

[4] Fast-Plate OCR, “fast-plate-ocr (PyPI package),” 2024–2025. [Online]. Available: https://pypi.org/project/fast-plate-ocr/. Accessed: Nov. 4, 2025.

[5] L. Mufti, N. Benblidia, and M. A. Khan, “Automatic Number Plate Recognition: A Detailed Survey of Relevant Techniques,” *Sensors*, vol. 21, no. 9, 3028, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/9/3028.

[6] A. C. Bukola, P. A. Owolawi, C. Du, and E. Van Wyk, “A Systematic Review and Comparative Analysis Approach to Boom Gate Access Using Plate Number Recognition,” *Computers*, vol. 13, no. 11, p. 286, 2024. [Online]. Available: https://doi.org/10.3390/computers13110286.

[7] A. Hassani and H. Shi, “CCT: Compact Convolutional Transformers,” *arXiv:2104.05704*, 2021. [Online]. Available: https://arxiv.org/abs/2104.05704.

TABLE IV
Overall Detection and Recognition Performance of the Evaluated Architectures

| **Model Combination** | **mAP@0.5** | **mAP@0.5:0.95** | **Precision** | **Recall** | **OCR Method** | **Plate Accuracy (%)** |
|---|---|---|---|---|---|---|
| YOLOv5 + Contours + CNN | 0.985 | 0.700 | 0.988 | 0.975 | Contours + CNN | 58.2 |
| YOLOv5 + YOLO + CNN | 0.985 | 0.700 | 0.988 | 0.975 | YOLO + CNN | 75.2 |
| YOLOv5 + Fast-Plate OCR | 0.985 | 0.700 | 0.988 | 0.975 | CNN + CTC | **97.3** |
| YOLOv8 + Contours + CNN | 0.990 | 0.730 | 0.992 | 0.980 | Contours + CNN | 59.2 |
| YOLOv8 + YOLO + CNN | 0.990 | 0.730 | 0.992 | 0.980 | YOLO + CNN | 76.4 |
| YOLOv8 + Fast-Plate OCR | 0.990 | 0.730 | 0.992 | 0.980 | CNN + CTC | **98.1** |
| YOLOv11 + Contours + CNN | 0.994 | 0.752 | 0.995 | 0.985 | Contours + CNN | 60.2 |
| YOLOv11 + YOLO + CNN | 0.994 | 0.752 | 0.995 | 0.985 | YOLO + CNN | 77.9 |
| YOLOv11 + Fast-Plate OCR | **0.994** | **0.752** | **0.995** | **0.985** | CNN + CTC | **98.9** |

[8] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks,” in *Proc. ICML*, 2006, pp. 369–376. [Online]. Available: https://www.cs.toronto.edu/~graves/icml_2006.pdf.

[9] X. Li, W. Wang, L. Zhang, and J. Sun, “Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection,” *NeurIPS*, 2020. [Online]. Available: https://arxiv.org/abs/2006.04388.

[10] C. Feng, Y. Zhong, W. Gao, M. R. Scott, and W. Huang, “TOOD: Task-Aligned One-Stage Object Detection,” in *Proc. ICCV*, 2021, pp. 3490–3499. [Online]. Available: https://openaccess.thecvf.com/content/ICCV2021/papers/Feng_TOOD_Task-Aligned_One-Stage_Object_Detection_ICCV_2021_paper.pdf.