LPR systems comparative analysis

September 23, 2025

Contents

1 Introduction

Automatic License Plate Recognition (ALPR) has become an essential component of modern intelligent transport systems, parking automation, and law enforcement. However, performance can degrade significantly under adverse environmental conditions such as poor illumination, heavy rain or fog, motion blur, or extreme viewing angles.

This research systematically evaluates state-of-the-art OCR and ALPR frameworks to determine their robustness, efficiency, and deployability in such scenarios. It also proposes practical integration strategies for real-time server-side workflows that must balance accuracy, latency, and scalability.

2 Frameworks Considered

Four representative frameworks were reviewed to capture the spectrum of current ALPR approaches—from traditional OCR engines to deep learning–based detection-recognition pipelines.

Tesseract OCR (Apache-2.0): A mature, CPU-efficient OCR engine suitable for clean, high-contrast plate crops. It struggles with noise, low light, or motion blur.

EasyOCR (Apache-2.0): A deep learning–based OCR library supporting multiple languages and fonts. It achieves higher robustness than Tesseract and benefits from GPU acceleration.

OpenALPR (AGPL-3.0 / Rekor commercial): An end-to-end ALPR system offering detection, segmentation, and recognition. The open-source version is functional but outdated and performs poorly under complex lighting or weather.

YOLO-based Detection + OCR: Combines a YOLO object detector for license plate localization with an OCR engine such as EasyOCR or PaddleOCR for recognition. This hybrid approach achieves state-of-the-art robustness but requires GPU resources for real-time use.

3 Evaluation Criteria

Each framework is evaluated using quantitative and qualitative metrics that reflect real-world deployment constraints:

- **Detection/Recognition Accuracy (%):** Mean character and sequence accuracy on mixed-condition datasets (e.g., CCPD-Weather, UFPR-ALPR). Robust systems should exceed 90%.
- **Latency and Throughput:** Average per-frame inference time (ms) and effective FPS on CPU/GPU. Real-time threshold: $\leq$60 ms per frame ($\geq$16 FPS).

- **Robustness Index (0–1):** Weighted accuracy under low light, rain/fog, blur, and perspective distortion.
- **Integration Complexity:** Effort required for containerization and API integration (Low/Medium/High).
- **Resource Efficiency:** CPU/GPU utilization at target FPS.
- **Licensing and Deployability:** Implications of Apache-2.0 vs. AGPL-3.0 for proprietary deployment.

4 Condition-Specific Observations

A targeted review of published benchmarks and experiments reveals clear behavioral trends across conditions:

Low-Light/Nighttime: YOLO-based models paired with contrast enhancement (CLAHE or Retinex) maintain accuracy above 90%, while Tesseract often falls below 50%. EasyOCR remains moderately stable when preprocessing is applied.

Adverse Weather (Rain/Snow/Fog): YOLO with weather-augmented training outperforms classical methods, whereas OpenALPR's accuracy drops more than 30% in heavy fog.

Motion Blur/High Speed: YOLO reliably detects plates, but OCR accuracy decreases unless motion deblurring or temporal voting is used. EasyOCR with frame-level aggregation recovers about 10% accuracy.

Angles/Perspective: YOLO-OBB (Oriented Bounding Box) models combined with OpenCV homography correction substantially improve recognition under skewed views, reducing errors by up to 40%.

5 Comparative Evaluation

What the table compares? It ummarizes how five OCR/ALPR stacks perform when conditions are *not ideal*—including night/low light, rain/snow/fog, motion blur, and skewed viewing angles. Each row represents a practical stack that could be deployed in production, while each column corresponds to a tangible criterion relevant to server-side systems, such as accuracy, computational speed, integration effort, and licensing.

How to read the columns.

- **Accuracy (Adverse)** — average character or sequence accuracy on mixed-condition datasets (e.g., low light, weather, blur, angle). "High" typically means above 90%

sequence accuracy with adequate preprocessing or augmentation; "Medium" indicates 80–85%; "Low" refers to less than 65%.

- **CPU/GPU Speed** — qualitative measure of inference throughput for 720p crops. *Very High* corresponds to real-time performance on a single modern GPU; *High* approaches real-time; *Medium/Low* indicate below real-time or CPU-only batch processing.
- **Ease of Use** — reflects integration effort when deploying as a Dockerized API (model export, dependencies, maintenance). *High* = straightforward integration; *Medium* = requires moderate engineering effort.
- **License** — summarizes open-source licensing implications for production use.
- **Overall Notes** — key deployment takeaways, caveats, or trade-offs based on reported behavior.

Context. All compared systems assume a two-stage pipeline: license plate detection followed by text recognition. YOLO-based detectors handle challenging environments more reliably, while OCR accuracy is primarily influenced by preprocessing (denoising, deblurring, rectification) and, where video data is available, temporal voting across frames.

Stack	Accuracy (Adverse)	CPU Speed	GPU Speed	Ease of Use	License	Overall Notes
YOLO + OCR (EasyOCR/PaddleOCR)	High (>90%)	Low	Very High	Medium	AGPL[a]	Most balanced; robust across low light, weather, blur, and angled plates.
YOLO OBB + OCR	High (>92%)	Low	High	Medium	AGPL[a]	Excels under perspective distortion; oriented boxes improve geometric accuracy.
EasyOCR + OpenCV	Medium (80–85%)	Medium	High	High	Apache-2.0[b]	Excellent for prototyping; depends heavily on preprocessing quality.
Tesseract + OpenCV	Low (<60%)	Very High	–	High	Apache-2.0[b]	Only viable for clean, well-lit, frontal crops; fails under low light or blur.
OpenALPR (open-source)	Low (<65%)	High	Medium	High	AGPL[a]	Outdated baseline; unsuitable for

Comparative Insights.

- **YOLO-based systems** outperform traditional OCRs in every adverse category, maintaining above 90% recognition even in rain or low light.
- **Classical OCR engines** (Tesseract, OpenALPR) remain fast but degrade sharply when blur or glare is introduced.
- **EasyOCR + OpenCV** is a pragmatic middle ground for CPU-based deployments.
- **Licensing** significantly affects adoption: AGPL restricts proprietary reuse, while Apache-2.0 frameworks are commercially flexible.

Comparative Conclusion. In summary, the **YOLO + EasyOCR + OpenCV** pipeline delivers the best balance of detection robustness, recognition accuracy, and integration simplicity. Although it requires GPU acceleration for optimal throughput, it offers a sustainable foundation for scalable, real-time ALPR systems in both on-premise and cloud environments.

6 Integration into a Server-Side Workflow

For practical deployment, YOLO performs plate detection, OpenCV handles preprocessing, and EasyOCR executes character recognition. These components operate within a **FastAPI** microservice, containerized with Docker for portability.

- **GPU path:** TensorRT optimization for sub-100 ms inference.
- **CPU path:** ONNX Runtime for edge devices or cost-sensitive setups.
- **Storage:** PostgreSQL for metadata; object storage for images.
- **Monitoring:** Prometheus metrics visualized via Grafana dashboards.

This architecture sustains real-time throughput on NVIDIA L4/A10G GPUs and near-real-time on optimized CPUs.

7 Benchmarking Protocol

Consistent evaluation requires standardized datasets and metrics:

- *Datasets:* CCPD (Weather/Blur/Tilt), UFPR-ALPR (Day/Night), RodoSol-ALPR (Mixed), and LP-Blur (Motion).
- *Metrics:* Detection mAP/Recall, character and sequence accuracy, end-to-end frame accuracy, FPS throughput, and latency distributions.

8 Cloud and Server Recommendations

For prototyping, consumer GPUs (RTX 4060/4070) are sufficient. For scalable production deployment:

- **Google Cloud G2 (L4):** Best price-to-performance for inference workloads.
- **AWS EC2 G6 (L4) / G5 (A10G):** Mature ecosystem and deployment tooling.
- **Azure NVads A10 v5:** Cost-effective entry with fractional GPU instances.

9 Conclusion and Logical Comparison

This research compared five OCR/ALPR frameworks across multiple adverse conditions. Traditional OCR engines like Tesseract and OpenALPR demonstrate acceptable speed but fail under realistic environmental noise. Deep learning–based systems—especially those combining **YOLO detection with EasyOCR recognition and OpenCV preprocessing**—provide the strongest balance of accuracy, robustness, and integration flexibility.

The proposed pipeline is therefore recommended for modern ALPR deployments requiring:

- Reliable performance across lighting, weather, and motion variations.
- Maintainable containerized APIs (FastAPI + Docker).
- Scalable GPU/CPU inference with TensorRT or ONNX Runtime.

This configuration ensures adaptability from research prototypes to production-scale deployments, representing a robust, future-ready solution for real-world license plate recognition.