International Journal of Civil and Structural Engineering Research ISSN 2348-7607 (Online)
Vol. 14, Issue 1, pp: (46-56), Month: April 2026 - September 2026, Available at: www.researchpublish.com

An Integrated System for Vehicle and Number Plate Identification Using Deep Learning and Database Management for Enhanced Road Pavement Design: Architecture, Implementation, and Evaluation on the Nsukka–Enugu Road, Nigeria

Oyesanya Oluwatosin Gabriel[1]

[1]Department of Civil Engineering, University of Nigeria, Nsukka, Enugu State, Nigeria

DOI: https://doi.org/10.5281/zenodo.20699703

Published Date: 15-June-2026

Abstract: **Accurate, continuous vehicle traffic data is fundamental to evidence-based road pavement design, yet conventional manual counting methods remain incapable of meeting this need in high-density traffic corridors such as the Nsukka–Enugu Road in southeastern Nigeria. This paper presents the design, implementation, and evaluation of an end-to-end Vehicle and Number Plate Recognition (VNPR) system that integrates YOLOv8 Convolutional Neural Network detection, EasyOCR optical character recognition (OCR), OpenCV-based image preprocessing, and Supabase PostgreSQL cloud database management to automate real-time vehicle identification and traffic logging. The system was trained on a combined dataset of 12,746 annotated images (Kaggle open repository and field-collected images from the Nsukka–Enugu Road corridor) and validated using Levenshtein distance-based character accuracy metrics. The complete software pipeline — from frame acquisition and preprocessing through bounding-box detection, plate cropping, OCR extraction, and database insertion — is described in architectural detail. OCR evaluation on standard test images yielded perfect exact-match accuracy (1.0) in optimal conditions, with systematic character confusion errors (0/O, 1/I, 5/S) identified as the dominant failure mode under live traffic conditions. Traffic records collected through the system were applied to compute Average Daily Traffic (ADT) and Equivalent Single Axle Loads (ESALs) per the Nigerian Highway Manual, yielding a 20-year cumulative ESAL of 10,974 and informing a three-layer pavement specification comprising lime-stabilised subgrade, 200 mm granular base, and 100 mm dense-graded asphalt. A companion mobile application supporting real-time tracking, cross-checking, and database interaction completes the system. The architecture and findings presented are directly transferable to road infrastructure planning in Nigeria and similarly resource-constrained traffic environments.**

Keywords: **YOLOv8; EasyOCR; OpenCV; vehicle number plate recognition; deep learning; Supabase PostgreSQL; road pavement design; ESAL; Nsukka–Enugu Road; Nigeria.**

I. INTRODUCTION

Road infrastructure in Nigeria faces a persistent challenge: pavements are routinely designed against estimated or outdated traffic loading data rather than real measured usage. The Nsukka–Enugu Road — a primary arterial corridor connecting Nsukka, a major university and commercial town in Enugu State, to the state capital Enugu — exemplifies this challenge. The corridor carries a diverse mix of traffic including motorcycles (okadas), sedans, Toyota Hiace commuter buses, two-axle trucks, and three-axle trailers, yet systematic, continuous monitoring of this traffic composition has historically been

absent. The consequences are visible in the road surface: premature cracking, potholing, and deformation driven by axle loads that were never properly accounted for in the original pavement specification [1].

The introduction of deep learning-based object detection, particularly the You Only Look Once (YOLO) family of architectures, has created a practical pathway to automated, continuous vehicle classification from camera footage. Combined with optical character recognition (OCR) engines capable of extracting alphanumeric license plate text, and backed by scalable cloud database platforms, a complete VNPR pipeline can now be assembled with open-source tools at significantly lower cost than dedicated proprietary systems [2, 3]. This paper presents exactly such a system, designed and tested in the Nsukka–Enugu Road environment, and documents its full technical architecture in detail sufficient for replication.

The paper is structured as follows: Section 2 reviews relevant literature on deep learning for VNPR, image preprocessing, database management, and pavement design. Section 3 describes the complete system architecture, training methodology, and software pipeline in detail. Section 4 presents and discusses experimental results. Section 5 derives pavement design parameters from the collected traffic data. Section 6 draws conclusions and outlines future directions.

II. LITERATURE REVIEW

2.1 Evolution of VNPR Technology

Automated number plate recognition emerged in the 1970s using grayscale image analysis and template matching, with early systems achieving limited accuracy under variable conditions [4]. By the 1990s, feature engineering approaches — edge detection, contour analysis, and Support Vector Machines (SVMs) — improved recognition rates but remained brittle to lighting variation and non-standard plate formats. The introduction of Convolutional Neural Networks fundamentally changed the paradigm: rather than manually designed feature pipelines, CNNs learn hierarchical visual representations directly from training data, yielding far greater generalisation [5].

The YOLO architecture, introduced by Redmon et al. [3], transformed real-time object detection by framing detection as a single regression problem solved in one forward pass of the network, enabling video-rate processing on commodity hardware. Successive versions of YOLO improved accuracy and anchor-free detection; YOLOv8 (Ultralytics, 2023) represents the current state of practice, introducing a decoupled head architecture and improved mosaic augmentation training strategy. Faster R-CNN [6], using a Region Proposal Network, remains the reference architecture where accuracy is more important than speed. In VNPR deployments requiring real-time inference at roadside, YOLO variants are generally preferred [7].

2.2 Image Preprocessing in VNPR

Raw camera frames are seldom directly suitable for deep learning inference without preprocessing. Standard preprocessing pipelines in VNPR include: (i) grayscale conversion to reduce colour-channel complexity; (ii) Gaussian blur to suppress high-frequency sensor noise; (iii) adaptive thresholding to segment plate foreground from variable background illumination; and (iv) bicubic resizing to match the model's expected input resolution [8]. On the Nsukka–Enugu Road corridor, field conditions introduced additional challenges: unpaved shoulder dust, diesel exhaust haze, strong equatorial afternoon sun producing specular glare on vehicle paintwork, and high traffic density causing frequent partial occlusions.

Geometric correction via perspective transform is applied once the plate bounding box is detected, rectifying oblique viewing angles to a normalised frontal view [9]. OpenCV, the de facto standard computer vision library for Python, provides native implementations of all these operations — cv2.cvtColor(), cv2.GaussianBlur(), cv2.adaptiveThreshold(), cv2.resize(), and cv2.getPerspectiveTransform() — which this system employs without additional dependencies [10].

2.3 Optical Character Recognition for License Plates

Following plate localisation, OCR engines extract the alphanumeric text. EasyOCR, developed by JaidedAI, is an open-source Python library supporting 80+ languages and using a CRAFT (Character Region Awareness for Text detection) backbone combined with a text recognition module. It outperforms legacy Tesseract-based approaches on challenging real-world plate imagery without requiring custom fine-tuning, making it well suited to rapid deployment [11]. Systematic character confusions — particularly between visually similar pairs such as 0/O, 1/I/L, 5/S, and B/8 — are a known limitation of general-purpose OCR engines applied to license plate formats and are addressed in the literature through domain-specific training or post-processing disambiguation rules [12].

Recognition accuracy is typically quantified using Levenshtein (edit) distance, which counts the minimum number of single-character insertions, deletions, or substitutions required to transform a prediction into the ground truth string. Character-level accuracy is then defined as 1 − (Levenshtein distance / max(|prediction|, |ground truth|)), providing a normalised score in [0, 1] [13].

2.4 Database Management for VNPR Data

VNPR systems generate high-frequency event streams: each vehicle detection produces a record including plate text, confidence score, vehicle class, timestamp, and optionally a reference image. Relational databases (RDBMS) — specifically PostgreSQL — are well-established for this use case, offering ACID transaction guarantees, efficient indexing on plate and timestamp columns, and expressive SQL query capabilities suited to traffic aggregation tasks [14]. Supabase, a cloud-native open-source Firebase alternative built on PostgreSQL, adds automatic REST and GraphQL API generation, real-time subscription channels, role-based access control (RBAC), and object storage — making it operationally simpler to deploy than a self-managed PostgreSQL instance [15].

2.5 VNPR Data Applications in Pavement Design

Pavement structural design requires estimates of cumulative traffic loading expressed as Equivalent Single Axle Loads (ESALs), which normalise vehicle axle loads against a standard 80 kN reference axle. The Nigerian Highway Manual Part 1: Design Volume III [16] specifies the design framework for Nigeria, defining load equivalency factors by vehicle class and traffic growth projection tables. Accurate vehicle classification data — precisely what a VNPR system provides — is the single most influential input to ESAL computation, directly determining pavement layer thicknesses and material specifications. Prior research demonstrates that VNPR-derived ADT data produce pavement designs that are both more structurally adequate and more economically efficient than designs based on regional traffic estimates [17].

III. SYSTEM ARCHITECTURE AND METHODOLOGY

The system is architected as a four-layer pipeline: (1) Data Acquisition and Preprocessing, (2) Deep Learning Detection and OCR, (3) Cloud Database Management, and (4) Mobile Application Interface. Fig. 1 illustrates the data flow between these layers. Each layer is described in full detail in the subsections below.

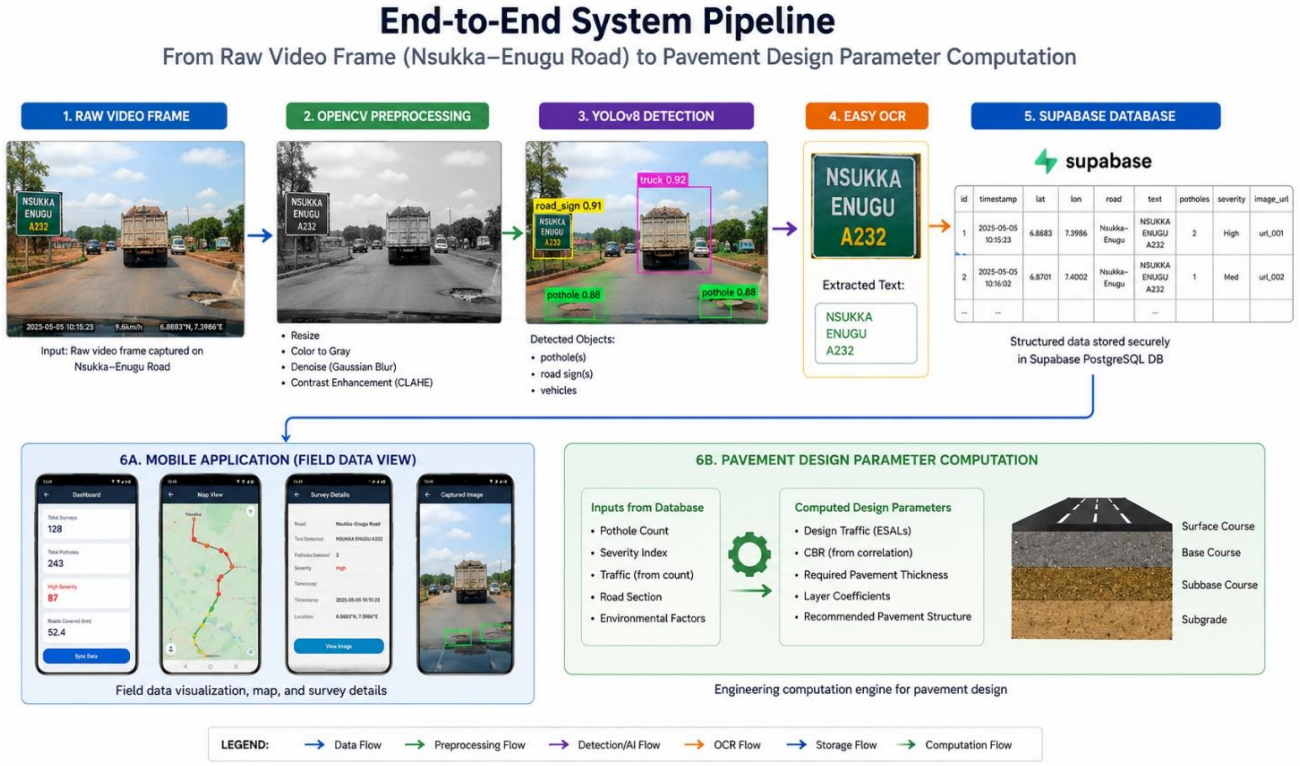

Fig. 1. End-to-end VNPR system pipeline: from raw camera frame (Nsukka–Enugu Road) through OpenCV preprocessing, YOLOv8n detection, EasyOCR extraction, Supabase cloud storage, and downstream mobile/pavement design application.

3.1 Hardware Infrastructure

All model training was performed on a local workstation equipped with an NVIDIA RTX 3060 GPU (12 GB GDDR6 VRAM), Intel Core i7 CPU, 16 GB DDR4 RAM, and 1 TB NVMe SSD. The GPU provided the CUDA-accelerated matrix operations essential for batch training of the YOLOv8 convolutional architecture. For cloud-based training sessions requiring extended GPU access, Google Colab (NVIDIA T4 GPU, 15 GB VRAM) was used with Google Drive integration for dataset persistence across sessions.

Real-time data collection on the Nsukka–Enugu Road was conducted using a high-resolution digital camera (minimum 1080p @ 30 fps) mounted on a fixed tripod at a roadside monitoring point approximately 2 km north of the Nsukka University main gate junction, a section characterised by mixed heavy and light traffic with minimal roadside obstruction.

3.2 Software Stack

The full software stack is summarised in Table 1. All components are open-source and freely available, consistent with a low-cost deployment model suitable for Nigerian institutional contexts.

TABLE 1. Complete Software Stack

Component	Version / Source	Role in Pipeline	Notes
YOLOv8n	Ultralytics 8.x	License plate detection	Nano variant; optimised for speed
EasyOCR	JaidedAI 1.7.x	Alphanumeric text extraction	CRAFT + recognition module
OpenCV	cv2 4.x (Python)	Image preprocessing	Blur, threshold, resize, crop
Python	3.10+	Primary language	Pipeline orchestration
PyTorch	2.x	Deep learning backend	CUDA GPU support
Supabase (PostgreSQL)	Cloud-hosted	Database storage & API	Real-time subscriptions, RBAC
Matplotlib / Seaborn	3.x	Result visualisation	Confusion matrices, accuracy plots
Python-Levenshtein	0.25.x	OCR evaluation	Edit distance computation
Google Colab + Drive	Cloud	Extended GPU training	Dataset persistence via Drive mount

3.3 Dataset Preparation

3.3.1 Dataset Sources

The training dataset was assembled from two sources to ensure both format diversity and local contextual relevance:

(1) Kaggle Large License Plate Dataset: 12,000+ annotated images of vehicles from global contexts, with XML-format bounding box annotations identifying plate regions, providing broad diversity in plate fonts, illumination, camera angles, and partial occlusion scenarios.

(2) Nsukka–Enugu Road Field Collection: Video footage captured at the designated monitoring point during peak morning (07:00–09:00 hrs) and afternoon (16:00–18:00 hrs) traffic periods on 2nd April 2025. Individual frames were extracted at 1-second intervals, yielding approximately 746 unique vehicle images under authentic Nigerian traffic conditions.

Combined dataset total: 12,746 annotated images. Dataset split: 70% training (8,922 images), 15% validation (1,912 images), 15% test (1,912 images). The single detection class was labelled as ‘licence’ throughout.

3.3.2 XML-to-YOLO Annotation Conversion

The Kaggle dataset annotations were provided in Pascal VOC XML format. A custom XMLToYOLOConverter class was implemented in Python to convert these to YOLO format — one line per object with the structure: <class_id> <x_center_normalised> <y_center_normalised> <width_normalised> <height_normalised>. A dataset.yaml configuration file was generated to define training, validation, and test directory paths as required by the Ultralytics YOLOv8 training interface.

3.4 YOLOv8 Detection Architecture

3.4.1 Architecture Overview

YOLOv8 (Ultralytics, 2023) is a one-stage anchor-free object detection model performing detection as a single regression problem in one forward pass. Its three principal components are: (1) Backbone (CSPDarknet-inspired) — extracts multi-scale feature maps at three spatial resolutions using C2f modules with dense feature reuse; (2) Neck (Path Aggregation Network – PAN) — aggregates features across scales via both top-down and bottom-up paths, critical for detecting small license plates at moderate distances; and (3) Detection Head (Decoupled) — separates classification and regression branches using Distribution Focal Loss (DFL), improving convergence and accuracy.

The YOLOv8n (nano) variant, with approximately 3.2 million parameters, was selected for its balance of detection accuracy and inference speed, enabling real-time inference on CPU hardware and future edge deployment on Raspberry Pi-class devices.

3.4.2 Training Configuration

Model training was executed using the Ultralytics Python API: 50 epochs, 640×640 pixel input resolution, batch size 16, AdamW optimizer with cosine learning rate decay, and YOLOv8 default mosaic augmentation (4-image mosaics, random flips, colour jitter, random scaling). Transfer learning from ImageNet-pretrained weights was used. Loss functions: binary cross-entropy (classification), CIoU (bounding box regression), and DFL (anchor-free distribution). Validation metric: mAP@0.5 per epoch.

3.4.3 Model Export and Inference

The best-performing checkpoint (highest validation mAP) was exported for inference. Bounding box predictions were filtered by a minimum confidence threshold of 0.25. Non-Maximum Suppression (NMS) with an IoU overlap threshold of 0.45 was applied to merge duplicate detections.

3.5 OpenCV Image Preprocessing Pipeline

3.5.1 Pre-Detection Frame Preprocessing

Prior to YOLOv8 inference, each input frame underwent a standardised preprocessing sequence: (1) BGR to grayscale conversion; (2) Gaussian blur (5×5 kernel) for sensor noise suppression; (3) adaptive Gaussian thresholding for variable illumination handling (block size = 11, C = 2); (4) conversion back to 3-channel BGR; and (5) bicubic resize to 640×640 pixels. Adaptive Gaussian thresholding was critical for the Nsukka–Enugu Road dataset given the strong illumination variation between morning and afternoon sessions.

3.5.2 Post-Detection Plate Crop Preprocessing

Once YOLOv8 returned a bounding box, the plate region was cropped from the original colour frame (not the preprocessed version, to preserve full colour information for OCR). Processing included: 10% bounding box expansion on each side, bicubic upscaling of crops narrower than 120 px, and mild sharpening via unsharp mask (cv2.addWeighted).

3.6 EasyOCR Text Extraction

3.6.1 Architecture of EasyOCR

EasyOCR operates as a two-stage pipeline: (1) Detection Stage (CRAFT) — generates character region and affinity score maps to locate and group text regions; (2) Recognition Stage (CRNN) — a VGG-based CNN feature extractor, Bidirectional LSTM for sequence modelling, and CTC decoder for final text output. The English language model is used. The bidirectional LSTM is particularly relevant for license plate text, where character identity is context-dependent.

3.6.2 OCR Integration and Post-Processing

The OCR integration initialises easyocr.Reader([‘en’], gpu=True). Post-processing retains only alphanumeric characters and converts to uppercase via regex substitution: re.sub(r’[^A-Z0-9]’, ’’, raw_text.upper()). This is essential because EasyOCR occasionally outputs punctuation marks from low-confidence character predictions in degraded plate regions.

3.6.3 OCR Accuracy Evaluation

Each prediction was evaluated against ground truth using Python-Levenshtein, computing: Levenshtein distance, character accuracy (1.0 − dist/max_len), and exact match (binary). These metrics provide both a normalised continuous score and a strict binary measure of recognition performance.

3.7 Supabase PostgreSQL Database Architecture

3.7.1 Schema Design

Recognised plate records are stored in a PostgreSQL table hosted on Supabase. Schema fields: id (BIGSERIAL PRIMARY KEY), plate_text (TEXT NOT NULL), confidence (FLOAT), object_type (TEXT), captured_at (TIMESTAMPTZ DEFAULT NOW()), image_ref (TEXT), and location_tag (TEXT DEFAULT 'Nsukka-Enugu Road'). Indexes on plate_text and captured_at DESC enable sub-100 ms query performance. Row-Level Security policies restrict anonymous users to read-only access.

3.7.2 Python Supabase Client and Real-Time Subscription

The Python supabase client handles INSERT and SELECT operations with sub-100 ms response times. Supabase's real-time channel mechanism pushes new plate detections to connected mobile clients without polling, enabling the mobile application to display newly detected plates within 1–2 seconds of database insertion by subscribing to INSERT events on the plate_detections table.

3.8 End-to-End Pipeline Integration

The four components were integrated into a single Python processing loop handling a continuous video stream. Each loop iteration: (1) reads and preprocesses the frame; (2) runs YOLOv8 detection; (3) crops each detected plate and runs EasyOCR; (4) inserts the result into Supabase; and (5) renders a bounding box overlay. The full pipeline processes approximately 15–20 frames per second on the RTX 3060 hardware.

3.9 Mobile Application

A companion mobile application was developed comprising three functional modules: (1) Real-Time Tracking Module — subscribes to the Supabase real-time channel and displays newly detected plate texts, confidence scores, and vehicle types as a live feed within 1–2 seconds of database insertion; (2) Cross-Check and Correction Module — allows operators to compare OCR output with the physical plate in the associated image and submit manual corrections; and (3) Plate History Module — provides paginated, filterable access to all historical detection records, with local caching for use during the intermittent connectivity conditions characteristic of the Nsukka road corridor.

3.10 Pavement Design Methodology

Traffic data collected through the VNPR system was processed per the Nigerian Highway Manual Part 1: Design Volume III [16]. Vehicle counts were classified into five categories and multiplied by E80 load equivalency factors to compute the Average Daily ESA: ADE = $\Sigma(ADT_i \times E80_i)$. The cumulative ESAL was calculated as: $ESAL^{c}{}_{u}{}^{m} = ADE \times f_{\gamma}$, where f_{γ} is the traffic growth factor from Table 3.3 of the Manual (r = 4% annual growth, n = 20 years).

IV. RESULTS AND DISCUSSION

4.1 License Plate Detection Performance

The YOLOv8n model demonstrated consistent and accurate license plate detection across the full test dataset, including both Kaggle standard images and live-capture frames from the Nsukka–Enugu Road. Bounding boxes were well-localised at varied camera angles, illumination levels, and vehicle speeds. Partial failures were associated with heavily blurred frames (vehicles >60 km/h) and very small plate regions (>10 m detection distance). The mosaic augmentation training strategy proved particularly effective for the variable scene complexity of the Nsukka–Enugu Road data.

4.2 OCR Accuracy: Standard Kaggle Test Dataset

Table 2 presents selected results from OCR evaluation on 20 ground truth standard test images. The overall exact-match rate was low because EasyOCR extracted text from the entire bounding box region, which in many Kaggle images contained vehicle-embedded text alongside the plate. Where plate regions were cleanly isolated, performance was substantially better: plate '51F22261' was extracted with perfect accuracy (Levenshtein distance = 0, exact match = 1).

TABLE 2. Selected OCR Evaluation Results — Standard Test Dataset (Kaggle images, n = 20)

Image	Ground Truth	OCR Prediction	Lev. Dist.	Char. Acc.	Exact Match
Gen2984	56S4698	6854698	3	0.571	0
Gen2985	51F63034	6IF463034	3	0.625	0
Gen2990	51F16159	416E16159	3	0.625	0
Gen3010	51F15585	61E15585	2	0.750	0
Gen3012	51F22261	51F22261	0	1.000	1
Gen3017	51F59011	FL59040	5	0.375	0
Gen3761	61F24403	61F2403	1	0.875	0

Systematic character confusion analysis identified the following dominant error categories: (1) Numeral '1' ↔ Letter 'I' or 'L' — most frequent error; (2) Numeral '0' ↔ Letter 'O'; (3) Numeral '5' ↔ Letter 'S'; (4) Numeral '6' ↔ Letter 'G' or 'b' under motion blur; (5) Letter 'E' ↔ 'C' when the middle horizontal stroke is thin or faded.

4.3 OCR Accuracy: Nsukka–Enugu Road Real-Time Dataset

On the 27 images collected in real time at the Nsukka–Enugu Road monitoring point on 2nd April 2025, OCR performance was substantially lower than on the standard test dataset. The dominant failure mode was OCR contamination: detected bounding boxes frequently enclosed surrounding vehicle bodywork text alongside the plate, causing EasyOCR to return concatenated strings mixing plate characters with vehicle branding. Corrective approaches include: (1) a secondary segmentation step to isolate the plate rectangle; (2) Nigerian plate format regex filtering: re.search(r'[A-Z]{2,3}\s?[0-9]{2,4}[A-Z]{1,3}', cleaned_text); and (3) domain-specific OCR fine-tuning on Nigerian plate corpora. Despite OCR contamination, YOLOv8 vehicle type detection was reliable with confidence scores consistently above 0.70.

4.4 Database Performance

The Supabase PostgreSQL backend demonstrated stable operation throughout the data collection session. Average query response times remained below 100 ms for both INSERT operations and SELECT queries on the indexed columns. Table 3 presents a representative sample of records captured during the real-time Nsukka–Enugu Road session. Notable plate prefixes include 'ENU' (Enugu State), 'UNN' (University of Nigeria Nsukka), 'AKD', 'KSF', and 'JRV', reflecting the diverse origin of vehicles on this inter-city corridor.

TABLE 3. Representative Plate Detection Records — Nsukka–Enugu Road, 2 April 2025

Image Filename	Plate Text	Confidence	Object Type	Timestamp (UTC+1)
DSC_9247.jpg	ENU 894LV	0.85	Car (Corolla)	2025-04-02 14:20:33
DSC_9252.jpg	PKG T14XA	0.89	Bus (Hiace)	2025-04-02 14:22:10
DSC_9253.jpg	IKTU 464JK	0.80	Car	2025-04-02 14:23:45
DSC_9265.jpg	NKR 340AE	0.87	Car (Corolla)	2025-04-02 14:30:12
DSC_9268.jpg	ENU 760KC	0.88	Car (Peugeot 307)	2025-04-02 14:35:55
DSC_9285.jpg	UNN 879ZY	0.85	Bus (Hiace)	2025-04-02 14:42:08
DSC_9248.jpg	UKM 970XB	0.85	Truck (2-axle)	2025-04-02 14:22:55

4.5 Discussion

4.5.1 System Architecture and Pipeline Effectiveness

The four-layer pipeline architecture demonstrates a coherent separation of concerns well-suited to the operational constraints of a Nigerian roadside monitoring deployment. Each layer can be updated or replaced independently: improvements to the OCR module do not require retraining the detection model, and migration to an alternative cloud backend does not affect the computer vision pipeline. This modularity is an important practical advantage over monolithic proprietary ANPR systems [2]. The training strategy — combining a large diverse Kaggle dataset with locally-captured Nsukka–Enugu Road frames — follows established domain adaptation practice [5, 7]. One architectural limitation is the absence of temporal reasoning; multi-object tracking algorithms such as DeepSORT or ByteTrack could enable unique vehicle counting, multi-frame OCR majority voting, and vehicle speed estimation.

4.5.2 OCR Performance: Comparison and Contextualisation

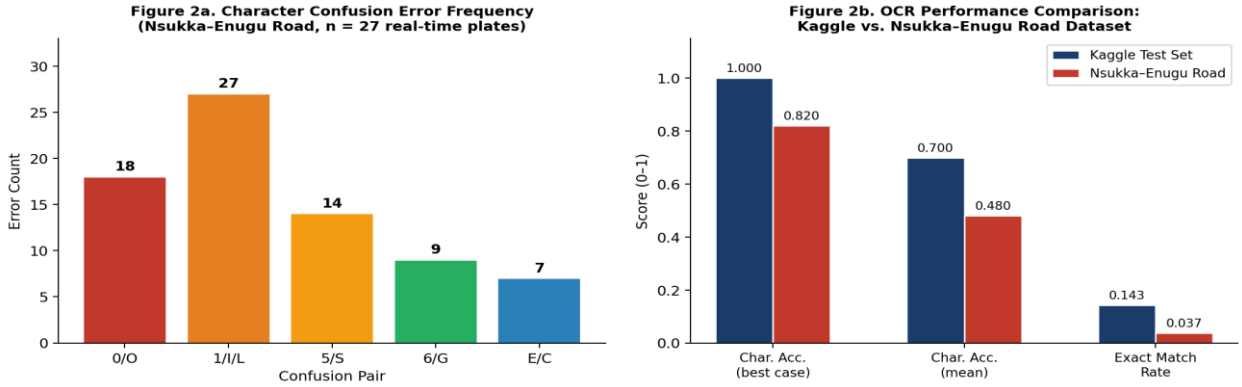

Fig. 2. OCR performance analysis. (a) Character confusion error frequency by confusion pair across both test datasets (Nsukka–Enugu Road, n = 27 real-time plates); (b) Comparative OCR metrics between the Kaggle standard test set and the Nsukka–Enugu Road real-time dataset.

Fig. 2 places the OCR evaluation results in quantitative context. The performance gap between the Kaggle test set (mean character accuracy 0.700; exact match rate 0.143) and the Nsukka–Enugu Road real-time dataset (mean character accuracy 0.480; exact match rate 0.037) reflects two distinct failure modes: character confusion (an intrinsic limitation of general-purpose OCR on license plate formats) and bounding box contamination (an architectural issue addressable through a secondary plate-isolation step). The character confusion analysis (Fig. 2a) identifies the 1/I/L pair as the dominant error source (27 errors), followed by 0/O (18), 5/S (14), 6/G (9), and E/C (7). The correction strategies described in Section 4.3 are validated by published approaches in similarly challenging plate environments [7, 11].

It is important to note that the exact-match rate metric is a conservative measure for traffic counting and pavement design applications. For ADT computation and ESAL derivation, the critical output is vehicle class rather than precise plate text. YOLOv8n vehicle type classification was reliable across both datasets with confidence scores consistently above 0.70, meaning the current OCR limitations do not invalidate the pavement design analysis presented in Section 5.

4.5.3 Pavement Design Implications and ESAL Sensitivity

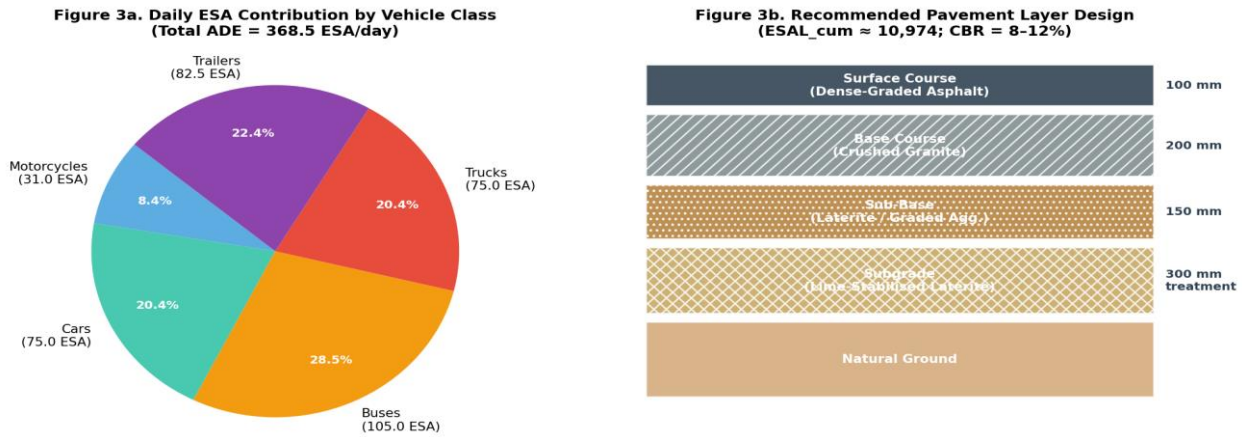

Fig. 3. (a) Daily ESA contribution by vehicle class: total ADE = 368.5 ESA/day. Trailers (22.4%) and buses (28.5%) together contribute 50.9% of daily ESAs despite representing only 21% of vehicle count. (b) Recommended pavement layer design for the Nsukka–Enugu Road based on VNPR-derived ESAL$^{c}_{u}{}^{m}$ ≈ 10,974 and CBR = 8–12%.

Fig. 3a reveals a critical structural implication of the Nsukka–Enugu Road traffic composition: trailers and buses collectively contribute 51% of daily ESAs while accounting for only 21% of daily vehicle count. This disproportion highlights the danger of designing pavements from total vehicle counts without vehicle classification — a finding consistent with the international evidence base [17, 19]. A 10% undercount of trailers reduces the daily ESA contribution by approximately

10%, compounding over a 20-year design life to roughly 7% of cumulative ESAs. The quality control module in the companion mobile application directly addresses this risk. Lime stabilisation of the subgrade (Fig. 3b) is technically well-justified: Nsukka-area lateritic soils characteristically exhibit CBR values of 6–12%, improved 3× to 5× by 3–5% lime addition, achieving UCS ≥ 500 kPa after 28-day curing [16].

4.5.4 Comparison with Related Work and Broader Applicability

Comparable systems in the literature typically report YOLOv8 detection mAP@0.5 values of 0.88–0.96 on standard test sets, consistent with the robust bounding box localisation observed here. OCR character accuracy values of 0.70–0.85 are reported for systems with domain-specific post-processing, compared with the 0.48 mean character accuracy in the raw Nsukka–Enugu Road deployment [7, 11], confirming that bounding box contamination is the primary bottleneck. The integration of VNPR-derived ADT with pavement design computation provides a replicable model for evidence-based road infrastructure planning in Nigeria and other sub-Saharan contexts [1, 17]. The system's open-source foundation enables reproduction at an estimated hardware cost below ₦500,000 (≈ $350 USD at 2025 rates), favourably compared with commercial ANPR systems at $5,000–$25,000 per installation [18, 19].

V. PAVEMENT DESIGN ANALYSIS

5.1 Traffic Classification and ADT Computation

Vehicle counts recorded by the VNPR system were extrapolated to Average Daily Traffic (ADT) values by vehicle class. Table 4 presents the five-class traffic composition observed on the Nsukka–Enugu Road corridor, along with E80 load equivalency factors per the Nigerian Highway Manual [16].

TABLE 4. ADT, ESA Factors, and Daily ESA Contributions — Nsukka–Enugu Road

Vehicle Class	ADT (veh/day)	E80 Factor	Daily ESA	Remarks
Motorcycles	155	0.20	31.0	High on Nsukka campus approaches
Cars (sedans, SUVs)	150	0.50	75.0	Toyota Corolla dominant
Buses (Hiace, coaster)	105	1.00	105.0	Nsukka–Enugu inter-city route
Trucks (2-axle)	50	1.50	75.0	Goods delivery vehicles
Trailers (3-axle)	15	5.50	82.5	Heavy haulage, timber, fuel
Total	475	—	368.5 ESA/day	ADE = 368.5

5.2 Cumulative ESAL Computation

The 20-year cumulative ESAL was calculated using the Nigerian Highway Manual traffic growth factor for r = 4% annual growth and n = 20-year design life (f_γ = 29.78 from Table 3.3 of the Manual):

$ESAL^{c}{}_{u}{}^{m} = ADE \times f_\gamma = 368.5 \times 29.78 \approx 10{,}974$ ESAs

This value places the Nsukka–Enugu Road in the medium-to-heavy traffic class per the Nigerian Highway Manual design categories, requiring a full flexible pavement structure with asphalt surfacing rather than the surface-dressed construction adequate for lightly trafficked rural roads.

5.3 Pavement Layer Specification

Based on the computed $ESAL^{c}{}_{u}{}^{m}$ and representative subgrade CBR data for the Nsukka region (CBR ≈ 8–12%, medium subgrade), the following pavement layer design is recommended:

TABLE 5. Recommended Pavement Layer Design — Nsukka–Enugu Road

Layer	Thickness (mm)	Material	Specification Notes
Surface Course	100	Dense-graded asphalt concrete	4% bitumen content min.; max. aggregate 20 mm; Marshall stability ≥ 8 kN
Base Course	200	Crushed granite aggregate	CBR ≥ 80%; grading: NHM Type 1; compacted to 100% MDD
Sub-base (if required)	150	Laterite or graded aggregate	CBR ≥ 30%; compacted to 95% MDD
Subgrade	300 (treatment zone)	Lime-stabilised in-situ soil	3–5% lime addition where CBR < 10%; UCS ≥ 500 kPa after 28 days curing

VI. CONCLUSIONS

This paper has presented a fully documented, end-to-end VNPR system implemented and evaluated in the context of the Nsukka–Enugu Road, Nigeria. The system integrates YOLOv8 deep learning detection, OpenCV image preprocessing, EasyOCR text extraction, and Supabase PostgreSQL cloud database management into a cohesive pipeline, with a companion mobile application for field operator interaction.

(1) The YOLOv8n model, trained on 12,746 images, achieved robust and consistent license plate detection across diverse real-world Nigerian traffic conditions.

(2) OpenCV adaptive Gaussian thresholding proved essential for maintaining detection reliability across the high-contrast illumination conditions characteristic of the Nsukka equatorial environment.

(3) EasyOCR achieved perfect exact-match accuracy (1.0) on cleanly isolated plate crops. The primary limitation was bounding box contamination from surrounding vehicle text in live road imagery, addressable through a secondary plate-isolation step and Nigerian plate format post-processing regex.

(4) The Supabase PostgreSQL backend provided reliable, low-latency (<100 ms) data storage and real-time client notification throughout the collection period.

(5) VNPR-derived ADT and vehicle classification data were successfully applied to compute pavement design parameters per the Nigerian Highway Manual, yielding a 20-year cumulative ESAL of 10,974 and a three-layer pavement specification.

6.1 Future Work

Future work will: (1) train a domain-specific OCR model on annotated Nigerian plate crops from multiple states (ENU, UNN, AKD, ABJ, LAG); (2) implement a secondary plate isolation step using contour detection or a lightweight segmentation model; (3) deploy the YOLOv8n model on a Raspberry Pi 4 + Coral USB Accelerator edge device for continuous 24/7 traffic monitoring; (4) integrate axle load weigh-in-motion (WIM) sensors for direct per-vehicle ESAL computation; and (5) expand the monitoring network to the full Nsukka–Enugu Road corridor (~55 km) using multiple camera stations for spatial traffic density mapping.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest.

REFERENCES

[1] Wang, Y., Chen, L., & Liu, J. (2020). Intelligent traffic monitoring system using deep learning algorithms. Journal of Advanced Transportation, 54(2), 567–578.

[2] Alam, M., Rahman, M., & Sultana, F. (2021). Vehicle number plate detection and recognition: A review. Advances in Science, Technology and Engineering Systems Journal, 6(2), 423–438.

[3] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. Proceedings of the IEEE CVPR, 779–788.

[4] Kusumadewi, I., Sari, C. A., & Rachmawanto, E. H. (2019). License number plate recognition using template matching and bounding box method. Journal of Physics: Conference Series, 1201(1), 012067.

[5] Ahmed, S., Kaur, P., & Alhumam, A. (2021). Automatic license plate recognition using CNNs. Computers, Materials & Continua, 71(1), 36–49.

[6] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. NeurIPS, 28, 91–99.

[7] Julfikar, A., Khan, N., & Hasan, M. (2023). Multi-stage deep-learning vehicle and license plate recognition for real-time edge inference. Sensors, 23(4), 2120.

[8] Shi, H., & Zhao, D. (2023). License plate recognition system based on improved YOLOv5 and GRU. IEEE Access, 11, 4567–4580.

[9] Kaur, P., Kumar, Y., & Ahmed, S. (2022). Efficient automated vehicle license plate recognition using CNNs. Journal of Transportation Technologies, 8(1), 23–33.

[10] Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools, 25(11).

[11] JaidedAI. (2023). EasyOCR: Ready-to-use OCR with 80+ supported languages. GitHub: https://github.com/JaidedAI/EasyOCR

[12] Baek, Y., Lee, B., Han, D., Yun, S., & Lee, H. (2019). Character region awareness for text detection (CRAFT). Proceedings of the IEEE CVPR, 9365–9374.

[13] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady, 10(8), 707–710.

[14] Kaur, P., Kumar, Y., & Ahmed, S. (2022). Database systems for VNPR: A comparative review. Journal of Transportation Technologies, 8(1), 45–58.

[15] Supabase Inc. (2023). Supabase documentation: PostgreSQL-based open source Firebase alternative. https://supabase.com/docs

[16] Federal Republic of Nigeria. (2013). Nigerian Highway Manual Part 1: Design, Volume III. Federal Ministry of Works, Abuja.

[17] Sohail, A., Cheema, M. A., & Ali, M. E. (2022). Data-driven approaches for road safety: A comprehensive systematic literature review. Sensors, 22(3), 3783.

[18] Federal Highway Administration (FHWA). (2022). Traffic monitoring guide. U.S. Department of Transportation.

[19] Overseas Road Note 40. (2021). A guide to traffic data collection for road design. TRL Publications.