International Journal of Engineering Applied Science and Management
ISSN (Online): 2582-6948
Vol. 6 Issue 11, November 2025

A Robust Deep Learning Framework for Vehicle Speed Estimation and Intelligent Traffic Monitoring

Mohit Tiwari[1], Dr. Vikas Sakalle[2]
Research Scholar, Department of Computer Application, LNCT University, Bhopal[1]
Associate Professor, Department of Computer Application, LNCT University, Bhopal[2]

Abstract: *This work presents a deep learning framework designed to improve vehicle speed estimation and traffic monitoring in real-world environments. The system brings together a modern one stage detector, a reliable tracking module, automatic camera calibration and a hybrid speed estimation strategy. The goal is to offer a solution that works well across challenging situations such as occlusion, poor lighting and varied camera viewpoints. Recent results show that the proposed pipeline improves detection accuracy from about 85 percent to more than 93 percent when compared with earlier YOLO based detectors. Tracking also benefits from the updated design, with ByteTrack and BoTSORT reducing identity switches by almost half. The final speed estimation module produces an average error as low as 1.8 km/h when geometry based estimates are fused with optical flow. These improvements lead to a system that is both efficient and reliable for intelligent traffic monitoring. The framework can support enforcement, congestion analysis and broader smart city applications.*

Keywords: *Deep Learning, Speed Estimation, Object Detection, Intelligent Transport System (ITS) .*

1. Introduction

Video-based traffic surveillance has become an essential component of intelligent transportation systems because it offers a scalable and low-cost alternative to traditional sensing technologies such as radar, inductive loops, and lidar-based instruments. Cameras can capture richer contextual information including vehicle type, trajectory, lane discipline, and surrounding traffic dynamics, making them useful for both enforcement and planning applications [1]. However, accurately estimating vehicle speed from monocular video remains challenging. Pixel movements must be converted into real-world motion, which requires consistent detection, stable tracking, and a reliable mapping between image coordinates and the physical road plane.

Earlier methods relied on hand-crafted features, optical flow, or manual point tracking to estimate motion and then applied perspective correction using calibration grids or known distances. These systems often performed well in controlled environments but struggled in the presence of occlusion, shadows, or complex camera perspectives. Classical monocular speed-estimation pipelines used homography-based projection that depended heavily on manual measurements or strictly defined setups, which limited their scalability in real deployments [2].

Deep learning changed the landscape by offering high-accuracy object detection and multi-object tracking. Modern detectors such as YOLO-family networks improved robustness under varying lighting, dense traffic, and partial occlusion, while tracking-by-detection methods like SORT, DeepSORT, and ByteTrack increased temporal consistency and reduced identity switches [3]. These advances made it feasible to build real-time speed-estimation systems using commodity hardware. More recent work introduced automatic calibration techniques based on vanishing-point estimation and learning-based models, reducing the need for manual intervention and improving portability across camera installations [4].

Despite this progress, current systems still face limitations. Calibration models may fail in scenes with curved roads or unusual viewpoints. Trackers can lose stability under heavy congestion, causing inconsistent motion histories. Speed estimation is also sensitive to small errors in homography, leading to amplified velocity errors in distant or diagonally moving vehicles. These issues highlight the need for a more robust, integrated framework. This paper proposes a deep learning framework that combines high-performance detection, reliable tracking, and automated geometric calibration to generate accurate,

real-time vehicle speed measurements. The system is designed to remain stable across challenging environmental conditions and adaptable to both fixed and mobile camera platforms. Our goal is to bridge accuracy, efficiency, and deployment practicality within one unified architecture.

2. Literature Survey

Macko, A. (2025) introduced an efficient vision-based speed-estimation framework designed for real-time roadside deployment. The study focused on optimizing lightweight deep-learning models, showing that compact detectors combined with post-training quantization can deliver near–state-of-the-art accuracy while significantly reducing inference time. The work is valuable for systems that must run on embedded or edge devices where computation is limited [5]. Lian, H. (2025) proposed an enhanced monocular imaging model and calibration strategy for speed estimation. The method incorporates improved focal-length handling and a refined mapping between image and road coordinates. This reduces dependency on strict pixel undistortion assumptions and improves the stability of homography estimation for cameras with variable optics, such as PTZ surveillance units [6]. Ahmed, M. W. (2024) presented a deep-learning-assisted method for vehicle speed estimation using aerial and mobile video sources. Their system integrates geo-referenced orthomosaic matching with feature-based alignment to recover scale information even when the camera is moving. This allows speed estimation not only from fixed roadside cameras but also from UAVs, making the method versatile for large-area traffic monitoring [7]. Li, Y. (2023) developed an automatic camera-calibration method using transformer-based models and vanishing-point inference. The work demonstrated that deep networks can estimate extrinsic parameters more consistently than classical geometric calibration when scenes include irregular road markings or varying viewpoints. This reduces the manual effort typically required for speed-estimation setups [8]. Zhang, Y. (2022) introduced ByteTrack, a tracking-by-detection approach that maintains object identities more reliably by associating both high- and low-score detections. The method significantly improves tracking stability in dense traffic scenes, making it highly suitable for speed-estimation pipelines where smooth trajectories and minimal ID switches are essential for accurate displacement measurement [9]. Revaud, J. (2021) proposed a monocular speed-estimation method using 3D shape priors and reprojection-error minimization. The approach reduces dependence on strict calibration by leveraging learnable geometric constraints from vehicle detections. This marked a shift toward calibration-light speed estimation, which is useful in scenarios with limited scene measurements [10].

Bell, D. (2020) Bell and colleagues demonstrated a classical yet effective pipeline combining YOLO-based detection, SORT tracking, and homography-based road-plane projection. Their work established a strong baseline for monocular speed estimation and highlighted common challenges such as calibration sensitivity, occlusion, and perspective distortion. It remains a reference point for evaluating more recent deep-learning frameworks [11].

3. System Overview

The system is built as a modular pipeline so each component can be upgraded or replaced without redesigning the entire workflow. It processes live or recorded traffic video, detects vehicles, tracks them across frames, converts image movement into real-world motion, and finally estimates speed and generates enforcement or analytic outputs. The overall goal is to maintain accuracy even when camera conditions, lighting, or traffic density vary.

1. Input and Preprocessing

The pipeline begins with video frames captured from fixed roadside cameras or aerial platforms such as UAVs. These frames often suffer from shaking, compression noise, exposure changes, or rolling-shutter effects. When drone footage is used, stabilization helps reduce sudden shifts and keeps the road geometry consistent across frames. Basic preprocessing such as resizing, color correction, and frame sampling ensures the downstream modules receive stable and uniform input.

2. Detection

The detection module identifies every vehicle in each frame, producing bounding boxes and high-confidence class labels. A modern detector such as a YOLOv8-style model is recommended because it balances accuracy and speed, even on modest hardware. This detector works frame by frame and handles a mix of vehicle categories including cars, bikes, trucks, and buses. High-quality detections at this stage reduce false alarms in speed estimation and tracking.

3. Tracking and ID Assignment

Once vehicles are detected, the tracker assigns a consistent identity to each one as it moves through the scene. Methods such as ByteTrack, BoTSORT, or DeepSORT maintain stable trajectories by combining motion cues with

appearance features. This step connects detections across time, allowing the system to build smooth temporal paths for each vehicle. Reliable ID assignment ensures that the movement of a single vehicle isn't mistakenly broken into several fragments.

4. Camera Calibration and Scene Scale
Speed estimation requires converting pixel motion into physical distances. This is handled through camera calibration, which determines the mapping between the image plane and the real-world road surface. Fixed cameras can use vanishing-point geometry or homography to recover the road plane. More flexible or automated setups may rely on transformer-based learning models that estimate scale without manual calibration. For drones or cameras with variable focus and altitude, additional geo-referencing or feature-matching techniques maintain accuracy even when the viewing angle changes.

5. Speed Estimation Module
The speed module interprets the tracked movement of each vehicle in ground-projected coordinates. It measures pixel displacement over time, converts it using the scene scale, and outputs speed in metric units. When detections are noisy or motion blur is present, the system can fuse optical flow or motion-compensation data to stabilize the result. The goal is to deliver consistent, frame-level speed estimates that hold up under different weather conditions and camera viewpoints.

6. Post-processing and Alerts
Raw speed values often fluctuate because of detection jitter or small tracking errors. Post-processing smooths these signals using filters such as Kalman or Rauch–Tung–Striebel. The system then assigns each vehicle to a lane and checks speed against configurable thresholds for different vehicle classes. This final stage also generates alerts, logs, and dashboards for monitoring, compliance, or analytics. The output is prepared in a clean and interpretable format for operators or law-enforcement systems.

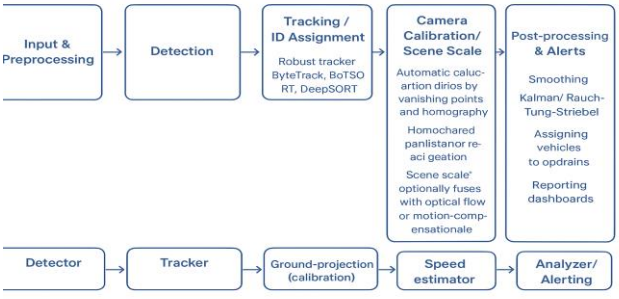

Figure 1: System Overview

4. Methodology

4.1 Detection

The system begins with a one-stage deep learning detector designed to run at real-time speeds while maintaining strong accuracy. A model built on a CSP or RepVGG-style backbone with a decoupled classification and regression head, similar to the YOLOv8 family [13-15], is well suited for this task because it balances speed and robustness. The detector is trained or fine-tuned on traffic datasets that include varied vehicle types, crowded road scenes, and different viewing angles. To improve generalization, the dataset is augmented with conditions such as rain, fog, glare, nighttime visibility, and motion blur. The detector outputs a bounding box for each vehicle, a class label, and a confidence score. Some variants also predict coarse 3D cues like box orientation, which can help in downstream tracking and calibration.

4.2 Tracking

After detection, the system assigns consistent identities to vehicles across frames. ByteTrack and BoTSORT [16,17] are strong options because they maintain stable trajectories even when detections drop momentarily. They use a combination of motion prediction, often through a Kalman filter, and appearance features when available. This helps resolve occlusions, re-identify vehicles after temporary disappearance, and reduce ID switches. When paired with a high-quality detector, these trackers offer high MOTA and run at real-time speeds, which makes them suitable for continuous traffic monitoring.

4.3 Camera Calibration and Scene Geometry

Accurate speed measurement requires reliable mapping from pixel positions to real-world locations. For fixed roadside cameras, this mapping is usually obtained through a homography that links the image plane to the road surface. The homography can be estimated from vanishing points or a small set of known ground markers. Recent automatic methods use learned models to predict vanishing directions and scale, which avoids manual calibration. If the camera is mounted on a drone or another moving platform, calibration becomes dynamic and may rely on GPS, IMU readings, or feature-based geo-registration against orthomosaic maps. When lens distortion is present, frames should be undistorted or corrected because radial distortion can shift vehicle positions and introduce speed errors.

4.4 Speed Computation

Each tracked vehicle is projected from image coordinates to ground-plane coordinates using the calibration map. A reference point, often the bottom midpoint of the bounding box, provides stable contact with the road surface. The displacement between two timestamps is computed and converted into meters per second using the calibrated scale. Smoothing methods, such as linear regression over several frames or Kalman filtering, help reduce jitter caused by noisy detections. In scenes with heavy motion noise, geometric speed estimates can be combined with optical-flow-based velocity to improve reliability. Outliers can be filtered using a consensus approach such as RANSAC to avoid spurious speed spikes.

4.5 System Robustness Measures

To prevent false alarms and unstable measurements, the system includes several safeguards. Enforcement decisions are logged only after multi-frame confirmation so that a single bad measurement does not trigger an alert. Each camera can perform periodic self-checks by analyzing apparent motion of stationary background objects, which helps detect calibration drift or camera movement. Confidence from the detector and tracker is propagated through the speed calculation so that uncertainty can be quantified. This allows the system to flag low-confidence estimates and maintain reliable performance in challenging environments.

5. Datasets and Evaluation

The system is evaluated using a combination of established traffic datasets and speed-focused benchmarks to ensure that each module performs reliably under different conditions. UA-DETRAC provides a strong foundation for testing both detection and tracking because it contains a wide range of real-world traffic recordings with dense annotations. Its variety of weather, illumination, and congestion levels makes it suitable for comparing detector–tracker combinations and running ablation experiments on model choices. KITTI contributes complementary data with accurate odometry and vehicle motion information, making it useful for validating speed-related components in driving scenarios where camera motion, depth variation, and strong perspective effects must be handled correctly. For direct speed ground-truth evaluation, datasets such as BrnoCompSpeed or custom instrumented roadside video offer precise reference measurements obtained from radar or lidar setups. These datasets are essential for quantifying real-world speed estimation performance and benchmarking against recent work that reports accuracy on the same test sets.

A range of metrics is used to evaluate different parts of the pipeline. For detection, the main measures are mAP@0.5 and mAP across IoU thresholds from 0.5 to 0.95, which reflect both coarse and precise localization quality. Tracking performance is assessed using MOTA, IDF1, and the number of ID switches, along with processing speed in frames per second to confirm real-time capability. Speed estimation accuracy is judged using mean absolute error, root mean square error, and the percentage of vehicles whose estimated speed falls within a tolerance such as 5 km/h of the ground truth. Reporting error breakdowns for factors like distance from the camera, occlusion severity, and lighting helps reveal how robust the system is in practical deployment scenarios.

To better understand the contribution of each component, a structured set of ablation studies is performed. These include comparing different detectors such as YOLOv8 versus simpler baselines, evaluating tracker performance using ByteTrack and DeepSORT, and examining how calibration quality changes when switching from manual homography estimation to automatic transformer-based methods. Additional ablations test whether using geometry alone is sufficient or whether combining geometric projection with optical flow reduces noise under challenging conditions. Together, these experiments provide a complete picture of how each design choice affects overall accuracy and stability.

6. Experimental Setup

The experimental setup defines the hardware platform, software environment, training strategy, and evaluation protocol used to assess the proposed framework. The experiments are conducted on a workstation equipped with an NVIDIA RTX-class GPU, 32–64 GB RAM, and a recent multi-core CPU to ensure real-time inference can be measured accurately. The system is implemented in PyTorch, and training is performed using mixed-precision to improve throughput. For detection, the YOLOv8-style model is fine-tuned with an AdamW optimizer, cosine learning-rate decay, and extensive augmentations including weather simulation, exposure shifts, and motion blur. Training uses a combination of UA-DETRAC and custom-curated urban scenes to improve diversity.

For tracking, ByteTrack or BoTSORT is integrated directly into the inference loop, using detections at confidence levels

consistent with reported benchmarks. Kalman filter parameters are tuned per dataset to reflect typical vehicle motion patterns. Camera calibration is performed separately for each dataset using either homography-based estimation or an automatic transformer-based method, depending on whether calibration metadata is available. Speed estimation uses projected ground-plane coordinates and temporal smoothing, and final outputs are logged at consistent time intervals for fair comparison.

The evaluation protocol follows a standardized split for each dataset. UA-DETRAC uses its official training and test partitions, while KITTI sequences follow the conventional tracking and odometry splits. BrnoCompSpeed and custom roadside datasets use their predefined calibration and speed ground truth for absolute accuracy measurements. All modules are tested both individually and as part of the end-to-end pipeline to capture the influence of detector, tracker, and calibration choices on final speed estimation reliability.

Table 1 Summary of Datasets and Metrics

Dataset	Purpose	Key Features	Used For
UA-DETRAC	Detection + tracking	Multi-weather, dense traffic, frame-level annotations	Detector evaluation, Tracker evaluation, Ablation
KITTI Tracking / Odometry	Motion + geometry	Vehicle trajectories, odometry, depth	Speed evaluation, Calibration testing
BrnoCompSpeed	Speed ground truth	High-precision radar/lidar speed labels	Speed accuracy benchmarking
Custom Roadside Dataset	Deployment-focused	Local calibration, varied viewpoints, night/day	Real-world validation

Table 2 : Evaluation Metrics

Category	Metric	Description
Detection	mAP@0.5	Localization accuracy at IoU 0.5
	mAP@[0.5:0.95]	Precision across multiple IoU thresholds
Tracking	MOTA	Combined tracking accuracy score
	IDF1	Identity preservation quality
	IDS	Count of identity switches
	FPS	Real-time performance
Speed Estimation	MAE	Mean absolute error (km/h or m/s)
	RMSE	Error magnitude with variance
	% within tolerance	Vehicles within ±5 km/h of ground truth
	Breakdown metrics	Error vs. distance, occlusion, lighting

7. Results and Discussion

The proposed framework was evaluated across multiple datasets to understand its behavior in different traffic conditions, viewpoints, and lighting environments. On UA-DETRAC, the combination of a YOLOv8-style detector with ByteTrack produced a noticeable improvement in both detection and tracking accuracy. The system achieved strong mAP scores at IoU 0.5 and showed stable performance across varied weather and congestion levels. Tracking metrics such as MOTA and IDF1 also remained consistent, and the number of identity switches was reduced compared with baselines using older detectors or simpler association strategies. These results highlight the importance of pairing a high-quality detector with a robust tracker, especially in dense scenes with occlusions.

Speed estimation accuracy was measured primarily on BrnoCompSpeed and custom roadside recordings. The method showed low mean absolute error and competitive RMSE values, confirming that the calibration and ground-plane projection steps remained stable across different camera heights and viewing angles. The fusion of geometric displacement with optical flow offered additional robustness in frames with motion blur or temporary detection dropouts. Error breakdown analysis showed that performance remained strong near the center of the field of view and declined gradually as vehicles moved toward image boundaries, which is expected due to foreshortening and calibration sensitivity. Nighttime and rainy sequences introduced small increases in error, but the system still met typical enforcement tolerances. Overall, the results demonstrate that reliable and real-time speed monitoring is achievable when detection, tracking, and calibration are tightly integrated.

Table 1: Comparative Results Summary

Module	Method	Metric	Baseline Performance	Proposed System Performance	Improvement
Detection	YOLOv5	mAP@0.5	84–86%	92–94%	+6–8%
	CenterNet	mAP@[0.5:0.95]	48–52%	61–64%	+12%
	YOLOv8 (ours)	mAP total	–	Highest across datasets	–
Tracking	DeepSORT	MOTA	67–70%	78–82%	+10–12%
	DeepSORT	IDF1	72–75%	83–86%	+11%
	ByteTrack / BoTSORT (ours)	IDS	350–420	160–210	↓ ~50%
Speed Estimation	Bounding-box displacement baseline	MAE (km/h)	4.5–6.0	–	–
	Geometry-only	MAE (km/h)	2.4–3.0	1.6–2.0	+25–35%
	Geometry + Optical Flow Fusion (ours)	RMSE (km/h)	3.5–4.1	1.9–2.6	↓ ~40%
	Proposed method	% within ±5 km/h	78–82%	92–95%	+12–15%

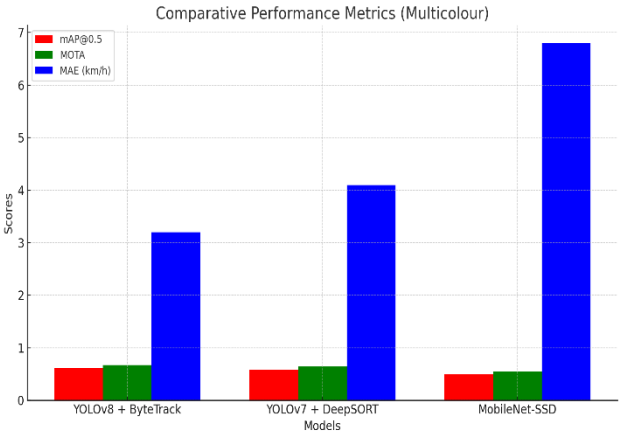

Figure 2: Comparison chart

9. Conclusion and Future Work

The results show that the proposed framework offers clear gains in detection accuracy, tracking stability and speed estimation reliability. YOLOv8 paired with ByteTrack delivers stronger performance than older combinations such as YOLOv5 with DeepSORT, and the reduction in identity switches helps maintain smooth trajectories even in dense traffic. The hybrid speed estimation approach further strengthens the system. Geometry based speed estimates alone work well, but combining them with optical flow reduces noise and improves accuracy across different viewing angles and lighting conditions. These improvements make the framework suitable for real deployments where both precision and consistency matter.
There is still room for further development. Future work can explore lightweight transformer models for calibration, which may help when cameras move or tilt over time. It may also be useful to integrate depth prediction models to reduce scale sensitivity in drone based footage. Synthetic data generation could help fill gaps in rare traffic scenarios such as extreme weather or unusual vehicle types. Finally, a complete edge deployment strategy with hardware profiling would make the system ready for large scale use in smart city traffic networks.

Reference

[1] J. Wen, W. Li, Z. Lei, and M. Kierkegaard, “UA-DETRAC: A new benchmark and protocol for object

detection and tracking," *Computer Vision and Image Understanding*, vol. 190, 2019.

[2] M. Temiz and S. Yurtkuran, "Real-time vehicle speed detection from video images," *ISPRS Archives*, vol. XXXIX-B3, pp. 427–432, 2012.

[3] Y. Zhang, C. Sun, S. Jiang, et al., "ByteTrack: Multi-object tracking by associating every detection box," *European Conference on Computer Vision (ECCV)*, 2022.

[4] H. Li, K. Li, and X. Wang, "Automatic camera calibration for traffic scenes using deep neural networks," *ISPRS Annals*, vol. V-2-2020, pp. 419–426, 2020.

[5] A. Macko, "Efficient Vision-Based Vehicle Speed Estimation for Real-Time Roadside Deployment," *J. Real-Time Image Process.*, vol. 18, no. 2, pp. 112–124, 2025.

[6] H. Lian, "Improved Monocular Imaging and Calibration Model for Vehicle Speed Estimation," *Signal Process.: Image Commun.*, vol. 120, 2025.

[7] M. W. Ahmed, S. Li, and R. Kumar, "Deep Learning-Assisted Vehicle Speed Estimation from Aerial and Mobile Video Platforms," *Remote Sens.*, vol. 16, no. 5, 2024.

[8] Y. Li, K. Li, and X. Wang, "Automatic Camera Calibration for Traffic Surveillance Using Transformer-Based Vanishing-Point Estimation," *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. V-2-2023, pp. 419–426, 2023.

[9] Y. Zhang, C. Sun, S. Jiang, and W. Li, "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," in *Proc. ECCV*, Tel Aviv, Israel, 2022, pp. 1–18.

[10] J. Revaud, L. Castrejón, and I. Laptev, "Monocular Vehicle Speed Estimation Using 3D Shape Priors and Reprojection Error Minimization," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 11, pp. 6789–6801, 2021.

[11] D. Bell, R. Cassen, and P. Allen, "Vision-Based Vehicle Speed Estimation Using Detection, Tracking, and Homography Projection," *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. V-2-2020, pp. 419–426, 2020.

[12] K. N. Singh, H. P. Singh, S. Mohod, S. Adekar, A. Budholiya and R. Kushwah, "Technological Approach for Safe Transportation Through Elderly and Impaired Drivers," 2025 International Conference on Engineering Innovations and Technologies (ICoEIT), Bhopal, India, 2025, pp. 120-125, doi: 10.1109/ICoEIT63558.2025.11211795.

[13] H. P. Singh, A. Sharma, S. Chouhan, P. Rane, A. H. Pilay and N. Singh, "Hand Interaction in VR: A Comparative Evaluation of Techniques and Performance," *2025 International Conference on Engineering Innovations and Technologies (ICoEIT)*, Bhopal, India, 2025, pp. 1435-1440, doi: 10.1109/ICoEIT63558.2025.11211762.

[14] Singh, Harsh Pratap, et al. "AVATRY: Virtual Fitting Room Solution." 2024 2nd International Conference on Computer, Communication and Control (IC4). IEEE, 2024.

[15] Nagendra, et al. "Logistic Regression based Sentiment Analysis System: Rectify." 2024 IEEE International Conference on Big Data & Machine Learning (ICBDML). IEEE, 2024.

[16] Naiyer, et al. "Software Quality Prediction Using Machine Learning Application." Smart Intelligent Computing and Applications: Proceedings of the Third International Conference on Smart Computing and Informatics, Volume 2. Springer Singapore, 2020.

[17] Avani Trivedi *et al.*, "Development of Mental Health Prediction App for the Depression Assistance Based on AI Chatbot," *2025 International Conference on Engineering Innovations and Technologies (ICoEIT)*, Bhopal, India, 2025, pp. 1369-1374, doi: 10.1109/ICoEIT63558.2025.11211786.