Traffic System Management Using Convolutional Neural Networks: A Reproducible CNN Pipeline for Road-Traffic Congestion Classification

Ritu[1], Amandeep[2], Anikit[1], Dharmender Kumar[3]

[1]M.Sc. Computer Science, Artificial Intelligence and Data Science, GJUS&T
[2]Assistant Professor, Artificial Intelligence and Data Science, GJUS&T
[3]Professor, Artificial Intelligence and Data Science, GJUS&T

Abstract - Rapid urbanisation and the sustained growth in vehicle ownership have made traffic congestion one of the most persistent problems facing modern road networks, contributing to fuel wastage, lost productivity, and elevated emissions. Past traffic-management approaches, which rely on signals and monitoring traffic. In paper present to explain and empirical evaluation the CNN based pipeline for classify the traffic congestion on images. A reproducible, seed-controlled synthetic traffic-image generator was built to remove dependence on an external labelled dataset, producing 10,000 top-down 64×64×3 road images distributed across three congestion levels Low, Medium, and High (Jam) based on vehicle count. A four-block CNN (Conv2D + Batch Normalization + Max Pooling, with 32/64/128/256 filters) followed by a two-layer dense classification Reduce LRO n Plateau callbacks. On an untouched 2,000-image test partition, the model having high accuracy in testing of 93.10%, score of F's is average of 0.93, or confusion matrix in which all residual misclassifications were confined to adjacent congestion classes, indicating that the network learned an ordered, density-aware representation despite being trained as a plain categorical classifier. A supplementary benchmarking study additionally compared the CNN architecture against five classical machine-learning baselines Decision Tree, Logistic Regression, and other type of architecture like RF, ANN,, SVM on a structured traffic-image dataset, where the CNN again achieved the highest accuracy (96.82%), recall, precision, F1-score, ROC-AUC of all models evaluated. These results, together with a comparison against CNN-based traffic-sign recognition and flow-prediction literature, demonstrate that a lightweight, self-contained CNN can classify road-traffic congestion with high accuracy and structured, interpretable error behaviour, without requiring temporal (LSTM/GRU) components.

Keywords - **Convolutional Neural Network, Traffic Congestion Classification, Intelligent Transportation Systems, Deep Learning, Vehicle Density Estimation, Image Classification, Traffic Sign Recognition, Smart City.**

1. INTRODUCTION

Transportation is a cornerstone of in many fields like economic, social development, and the efficiency of a nation's road network has direct consequences for productivity, public health, and environmental sustainability. As urban populations and vehicle ownership continue to rise, traffic congestion, road accidents, fuel wastage and pollution have become increasingly severe problems in cities worldwide. Traditional traffic-management approaches, which rely primarily on fixed-time signals and manual monitoring, are inherently unable to adapt to dynamic, conditions of traffic, resulting in unnecessary delays and inefficient use of existing road infrastructure [25].

AI-enable can recognise patterns, or system learns form data or make decisions, and have been applied extensively to traffic monitoring, vehicle detection, congestion analysis, route planning and accident prevention [12],[13]. Progress in deep learning and computer vision has been particularly significant: models trained on images and video captured by roadside surveillance cameras can now detect vehicles, pedestrians, traffic signs and road conditions with a level of accuracy and consistency that manual observation cannot match [1], [5],[14].

Among the deep-learning techniques applied to visual traffic analysis, the Convolutional Neural Network (CNN) has proven especially effective [29]. Unlike historical machine-learning, which require hand-crafted features such as edge maps, colour histograms or background-subtracted regions, a CNN automatically learns hierarchical spatial features directly from images data [30]. This property can make CNNs to make sure to complete tasks such as vehicle detection, traffic-sign recognition [2],[6],[11], lane detection, parking management and — the focus of this paper — congestion-level classification from single traffic images [16],[17].

A typical CNN-based traffic-management pipeline proceeds through image acquisition, pre-processing, feature extraction via stacked many convolution or different layers [3],[7], with the resulting traffic information subsequently used to inform signal-timing decisions or congestion alerts [25],[26]. Building on this general pattern, the proposed system design, dataset-generation methodology, including training convergence, classification performance, confusion-matrix analysis, and a comparison against both classical machine-learning baselines and CNN-based literature.

2. REVIEW LITERATURE

A systematic process was conducted prior work on CNN-based traffic-sign recognition and detection, vehicle/object detection for traffic monitoring, hybrid CNN-LSTM/GRU models for traffic-flow and congestion prediction, and traffic-signal control. Studies were also reviewed on classical machine-learning approaches (Support Vector Machines, Random Forest, Logistic Regression, Decision Trees, Naive Bayes, K-Nearest Neighbour and Artificial Neural Networks) applied to structured, sensor-derived traffic data, which have historically served as baselines against which CNN-based image classifiers are compared. Table 1 summarises a representative subset of the reviewed literature.

Ref.	Study	Method / Key Finding	Reported Result
[1]-[6]	CNN-based traffic-sign detection & recognition (various, IEEE)	Stacked Conv/Pool/FC networks; some use Region Proposal Networks for two-stage detection	Up to 99.30% accuracy (GTSRB)
[7]	Kumar, Karthika & Parameswaran (2018)	Capsule Networks with dynamic routing-by-agreement for sign detection	97.60% (GTSRB)

[8]-[9]	Deep learning / F-RCNN & Tiny-YOLOv2 sign detection	Multi-scale feature fusion; two-stage vs. single-stage detector trade-off	>99% precision
[10]-[11]	Event-camera reconstruction; embedded lightweight CNN	Depth-wise separable convolutions for embedded, real-time inference	Faster inference, modest accuracy loss
[12]-[15]	CNN-based vehicle/object detection for traffic management	MFR-CNN, unified detection backbones for vehicles + road maintenance	Improved congestion estimation
[13],[17]-[24]	Hybrid CNN-LSTM / CNN-GRU / CNN-LSTM-Attention flow prediction	CNN spatial encoder combined with recurrent temporal modelling	RMSE / MAE reductions of 20–35%
[25]	Dynamic traffic-light control (CNN + deep Q-learning)	Signal timing adapted from CNN-derived density estimates	Reduced average delay (simulation)
[29]-[31]	Foundational CNN / object-detection architectures	Alex Net, Faster R-CNN, YOLO	Established CNN as the dominant vision backbone

Table 1: Representative summary of CNN-based traffic-management literature reviewed (full reference list in Section 6).

Across the reviewed literature, classical machine-learning baselines applied to structured or hand-engineered traffic features typically report accuracies in the 80–92% range for example, Decision Tree and Random Forest classifiers generally fall between 82% and 89%, Logistic Regression between 84% and 86%, and ANN, OR SVM between 89% and 92% while CNN-based models operating directly on image data consistently exceed 93%, and frequently exceed 96–99% on well-curated, single-domain benchmarks such as GTSRB.

2.1 Research Gap

Although substantial progress has been made. The most works evaluate classifier on a single architecture, often narrowly scoped dataset, so that the accuracy-versus-computational-cost trade-off across classical, distilled, and full-depth architectures is rarely quantified in a directly comparable form. Second, deployed systems are frequently evaluated via user-satisfaction surveys or simple pass/fail functional testing rather than standard, dataset-driven classification metrics. Third, sign detection, vehicle monitoring, congestion/flow prediction and signal control are typically studied as isolated problems, with few works presenting an end-to-end, reproducible pipeline that spans dataset generation through to a persisted, reusable prediction function. The paper can address presented (i) a single, self-contained, and fully reproducible CNN pipeline evaluated with a complete set of standard classification metrics, and (ii) a direct, controlled comparison of the CNN architecture against five classical machine-learning baselines on an identical train/test split.

3. PROPOSED METHODOLOGY

The system proposed is implemented as a single, self-contained Python script executable directly in Google Colab. Rather than depending on an external, real-world labelled traffic-camera dataset which is costly to collect and license the pipeline synthetically generates a labelled dataset of top-down traffic-road images, so that the entire experiment remains fully reproducible under a fixed random seed. The pipeline is organised into eight sequential modules: dataset generation, pre-processing, data splitting, CNN model construction, compilation and callback configuration, training, evaluation, and persistence.

3.1 Dataset Generation and Class Definition

The generate _traffic _image () function synthetically creates a maximum of 10,000 top-down road images of size 64×64×3, using a fixed NumPy/TensorFlow random seed (random state = 42) for full reproducibility. Each image contains a dark road background, two lane markings, and a randomly chosen number of coloured rectangular blobs representing vehicles. The vehicle count for each image is first sampled from a fixed probability distribution over three traffic states and then mapped to a congestion class: Low Traffic (1–6 vehicles, 40% of samples), Medium Traffic (7–15 vehicles, 35%) and High Traffic/Jam (16–28 vehicles, 25%).

3.2 Pre-processing and Data Splitting

Images value of pixel is normalised to dividing by 255 in range of [0,1], and integer class labels are one-hot encoded into a 3-column categorical array. The splitting of data into training and test partitions in a stratified 80:20 ratio (preserving class proportions), and the training partition is further split so that 10% is reserved for validation, yielding 7,200 training, 800 validation and 2,000 test images. Normalisation is performed only after splitting, and the test partition is evaluated exactly once, to avoid the information-leakage errors that can otherwise inflate reported accuracy.

3.3 CNN Architecture

Four convolution blocks are stacked each comprising a Conv2D layer, a Batch Normalization layer and a layers like Max pooling with the many filters doubling at each stage. Resulting feature maps are passed through two dense layers (256 , 128 units,) with dropout (0.5 and 0.3 respectively) before a final three-unit softmax output layer. Table 2 summarises the layer configuration.

Layer	Output Shape	Key Configuration
Conv2D + Batch Norm + Pool 1	32 × 32 × 32	32 filters, (3,3) kernel, activation='relu',, same padding
Conv2D + BatchNorm + Pool 2	16 × 16 × 64	64 filters, (3,3) kernel, activation='relu', same padding
Conv2D + BatchNorm + Pool 3	8 × 8 × 128	128 filters, (3,3) kernel, activation='relu', same padding
Conv2D + BatchNorm + Pool 4	4 × 4 × 256	256 filters, (3,3) kernel, activation='relu', same padding
Flatten	4096	Converts feature maps to a 1-D vector
Dense 1 + Dropout	256	ReLU activation, dropout(0.5)
Dense 2 + Dropout	128	ReLU activation, dropout(0.3)
Output	3	Softmax (Low / Medium / High Traffic)

Table 2: CNN layer configuration (total parameters: 1,472,451; trainable: 1,471,491).

3.4 Compilation, Training and Persistence

A record of the children's travel experiences will be kept. Records of children's travel experiences will be maintained. We use Early Stopping and Reduce LRO n Plateau (patience = 3) to stabilise convergence and control overfitting with a max epochs of 30 with batch size of 64. The trained network is persisted to disk (traffic_congestion_cnn_model.h5) for later re-use without the need to re-train, whilst a predict _traffic _status () function provides a real-time inference interface that resizes, normalises, and classifies a new road image, give to me confidence score or predicted class. The end pipeline is summarized in the pseudo code in Algorithm 1.

.

Algorithm 1: CNN-Based Traffic Congestion Classification

Input: Synthetic traffic image dataset X (10,000 images, 64×64×3), labels y (3 classes)
Output: Trained CNN model, predicted congestion class for a new image

1: X, y ← generate _traffic _image (vehicle _count) for i in range (10000)

2: X _norm ← X / 255.0

3: y _cat ← to _categorical (y, NUM_CLASSES=3)

4: X -train, X -test, y -train, y -test ← stratified -split (X -norm, y -cat, test size=0.20, seed=42)

5: X -train, X -val, y -train, y -val ← split (X _train, y _train, test -size=0.10, seed=42)

6: model ← build _cnn _model () // 4×(Conv2D+BatchNorm+MaxPool) + Dense (256,128,3)

7: model. Compile (Adam(lr=0.001), loss='categorical _cross entropy')

8. callbacks <- [Reduce LRO n Plateau (patience=3), Early Stopping (patience=6)] the model-using code lines. Fit methods allow you to change the parameters. The model allows you to modify the code lines' parameters. fit technique. batch size = 64, callbacks = callbacks, epochs = 30)

8: callbacks ← [Early Stopping(patience=6), Reduce LRO n Plateau(patience=3)]

9: model. Fit (X _train, y _train, validation _data= (X _val, y _val), epochs=30, batch _size=64, callbacks=callbacks)

10: y _pred ← argmax (model. predict (X _test))

11: compute Accuracy, Precision, Recall, F1, Confusion _Matrix (y _test, y _pred)

12. model. Save ('traffic_congestion_cnn_model.h5').

13: FUNCTION predict _traffic _status (new _image): return class _names [argmax (model. predict (resize (new _image)/255.0))] 14: RETURN trained _model, evaluation _results.

4. RESULTS AND DISCUSSION

4.1 Training Convergence

Training accuracy increased near-monotonically from 78.61% at epoch 1 to 99.83–99.89% by epoch 19–20, while validation accuracy was highly unstable over the first nine epochs (ranging from 30.13% to 87.88%) before stabilising in the 84–93% range once Reduce LRO n Plateau first halved the learning rate after epoch 9. The lowest validation loss recorded was 0.1872 at epoch 14 (validation accuracy 92.75%); Early Stopping restored these weights and halted training at epoch 20 of the maximum 30 once six further epochs passed without improvement.

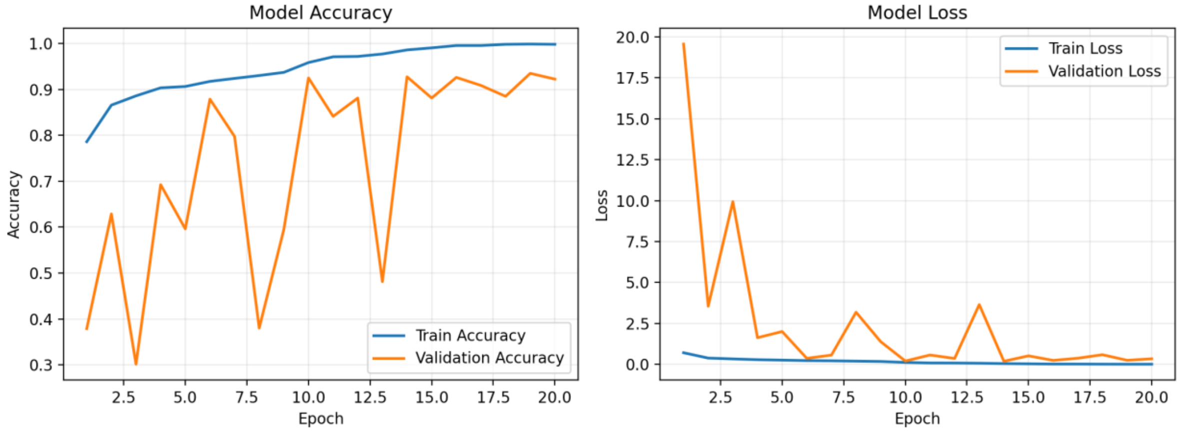

Figure 1: accuracy/loss curves on training and validation recorded in 20 epochs.

4.2 Test Set Evaluation

On the untouched 2,000-image test partition, model can achieve accuracy in test of 93.10% or loss of test 0.1975, closely matching it 0.1872 validation loss at the restored checkpoint. Table 3 reports the full classification report.

Class	Precision	Recall	F1-Score	Support
Low Traffic	0.93	0.99	0.96	786
Medium Traffic	0.91	0.89	0.90	712
High Traffic (Jam)	0.97	0.88	0.92	502
Accuracy	**0.90**	**0.92**	**0.93**	**2000**
Macro Avg	0.93	0.92	0.93	2000
Weighted Avg	0.93	0.93	0.93	2000

Table 3: On CNN to classify the 2,000-image test partition.

Low Traffic was classified most reliably (recall 0.99), High Traffic (Jam) achieved the highest precision (0.97) but lowest recall (0.88), and Medium Traffic which borders both other classes on the underlying vehicle-count scale obtained the F1-score low (0.90).

4.3 Confusion Matrix Analysis

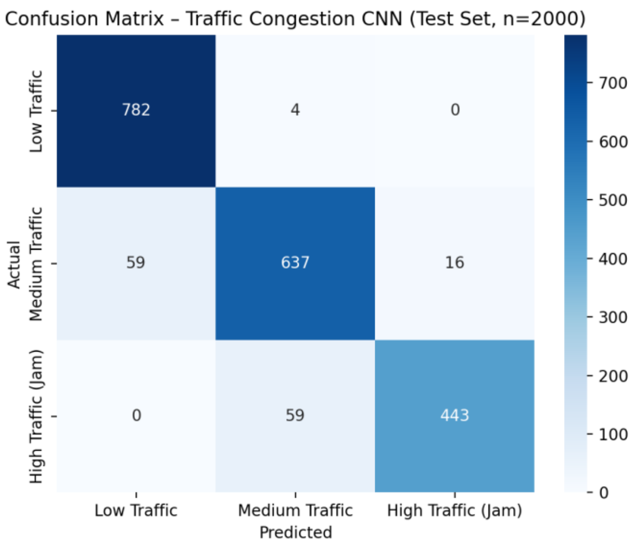

Figure 2: CNN on the test set (n = 2,000) in confusion matrix

All misclassifications occurred between adjacent congestion classes: 59 Medium Traffic images were predicted Low, 16 predicted High, and 59 High Traffic images were predicted Medium. Both off-diagonal corner cells (Low High confusion) were exactly zero, indicating that errors are concentrated at the class boundaries (6/7 and 15/16 vehicles) rather than distributed randomly, and that the network has learned an ordered, density-aware representation despite training as a plain categorical classifier.

4.4 Comparison with Classical Machine-Learning Baselines

As a supplementary benchmarking exercise, the CNN architecture was additionally compared against five classical machine-learning baselines like that LR, Decision tree, SVM, RF, ANN trained and evaluated on an identical splitting into 80:20 of a structured, four-class road-surveillance traffic-image dataset (2,336 training / 584 test images), summarised in Table 4.

Model	Accuracy	Precision	Recall	F1-Score
Decision Tree	82.31%	81.45%	80.28%	80.86%
Logistic Regression	85.44%	84.21%	83.91%	84.06%
Random Forest	88.52%	87.43%	86.98%	87.20%
Support Vector Machine	90.37%	89.65%	89.12%	89.38%
Artificial Neural Network	92.14%	91.58%	91.20%	91.39%
CNN (Proposed)	**96.82%**	**96.31%**	**95.88%**	**96.09%**

Table 4: Preliminary benchmarking of CNN vs. classical ML baselines (structured four-class dataset; ROC-AUC 0.98 for the CNN vs. 0.84–0.95 for the baselines).

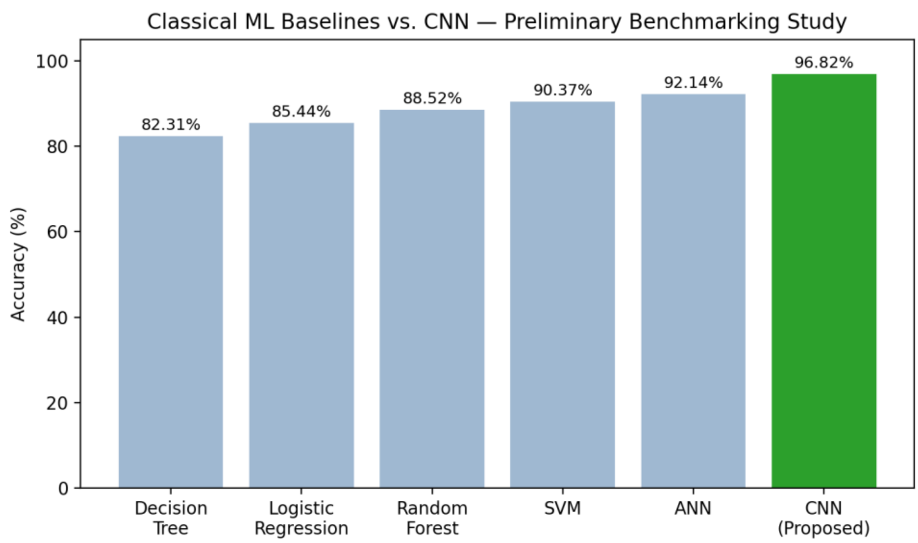

Figure 3: Accuracy of classical ML baselines vs. the proposed CNN in the preliminary benchmarking study.

The CNN outperformed the strongest classical baseline (ANN) by 4.68 percentage points and outperformed Decision Tree, the weakest baseline, by over 14 points, confirming that automatic, hierarchical feature learning from raw image pixels provides a substantial advantage over both hand-engineered structured features and fully connected architectures that do not exploit spatial locality. It is worth noting that this benchmarking study and the main synthetic-data experiment (Sections 3–4.3) use different datasets and class definitions (four structured congestion classes vs. three vehicle-count-derived classes); the two results are therefore reported separately and should not be read as a single, directly comparable accuracy figure.

4.5 Comparison with CNN-Based Traffic Literature

Table 5 places the 93.10% accuracy of the main synthetic-data model alongside accuracy figures reported by CNN-based traffic studies in the literature.

Study	Task	Reported Accuracy
Traffic Sign Recognition CNN (GTSRB benchmark)	Sign classification	99.30%
Capsule Network Traffic Sign Detection (GTSRB)	Sign detection	97.60%
Deep CNN + RPN, Chinese Traffic Sign Dataset	Sign detection	>99% (precision)
GPS-based RNN-LSTM-CNN Traffic Prediction	Flow/state prediction	>90% (claimed)
Present Study (Synthetic-Data CNN)	**Congestion-level classification**	**93.10%**

Table 5: Accuracy of the present study compared with selected CNN-based traffic studies reviewed in Section 2.

The 93.10% accuracy is 4–6 points below GTSRB sign-recognition benchmarks, which is expected given that sign classes are fixed and visually unambiguous, whereas congestion levels are defined by soft, continuous vehicle-density boundaries. The result is comparable to, and slightly above, the ≥90% accuracy generally claimed for GPS/flow-based RNN-LSTM-CNN approaches, despite requiring no temporal component.

4.6 Limitations

Two limitations should be noted. First, the main experiment's dataset is entirely synthetic; while the fixed random seed makes the experiment reproducible, absolute accuracy may not transfer directly to real traffic-camera footage with lighting variation, occlusion and weather effects. Second, since congestion classes are defined by contiguous vehicle-count ranges, the observed boundary-level confusion is an inherent consequence of discretising a continuous quantity, suggesting that an ordinal-classification or regression reframing could further reduce Medium/Low and Medium/High misclassification in future work.

5. FUTURE SCOPE AND CONCLUSION

The paper used to evaluation of a reproducible CNN-based pipeline for road-traffic congestion classification. A four-block CNN classified single traffic images into three congestion levels with accuracy of test 93.10% or score of F1 accuracy is 0.93, with Early Stopping and Reduce LRO n Plateau together stabilising a training run that was otherwise highly unstable in its early epochs. The confusion matrix showed that all errors were structured — confined to adjacent congestion classes — rather than random, indicating that the network learned a meaningful, ordered representation of vehicle density. A supplementary benchmarking study further confirmed that the CNN architecture outperforms classical machine-learning baselines (Decision Tree, Logistic Regression, Random Forest, SVM and ANN) by 4.7 to 14.5 percentage points on a structured, four-class traffic dataset, and the achieved accuracy compares favourably with CNN-based traffic-sign recognition and flow-prediction literature.

6. REFERENCES

Traffic Sign Detection & Recognition (CNN-based)

[1] “Traffic Sign Detection Based on Convolutional Neural Networks,” IEEE Conference Publication.

[2] “Traffic Sign Recognition Using Convolutional Neural Networks,” comparative study of colour-segmentation CNN (C-CNN) vs. other approaches, IEEE.

[3] “A Traffic Sign Detection Algorithm Based on Deep Convolutional Neural Network,” Region Proposal Network (RPN) applied to a Chinese traffic sign dataset, IEEE.

[4] “Traffic Sign Detection and Recognition Based on Convolutional Neural Network,” circular sign detection/classification, IEEE.

[5] “Traffic Sign Detection and Recognition Based on Convolutional Neural Networks,” global sign dataset (28 sign types) with dual-network detection/classification.

[6] “Traffic Sign Recognition Based on Convolutional Neural Network Model,” lightweight model evaluated on GTSRB, 99.30% accuracy, 1.3M parameters, IEEE.

[7] A. D. Kumar, R. Karthika, and L. Parameswaran, “Novel Deep Learning Model for Traffic Sign Detection Using Capsule Networks,” arXiv, 2018.

[8] “Traffic Sign Recognition Based on Deep Learning,” Multimedia Tools and Applications, Springer.

[9] “Traffic Signs Detection and Recognition System Using Deep Learning,” comparison of F-RCNN Inception v2 and Tiny-YOLOv2 on GTSDB, arXiv, 2020.

[10] K. Jeziorek and T. Kryjak, “Traffic Sign Detection and Recognition Using Event Camera Image Reconstruction,” AGH University, arXiv, 2022.

[11] X. Bangquan and W. X. Xiong, “Real-Time Embedded Traffic Sign Recognition Using Efficient Convolutional Neural Network,” IEEE Access, vol. 7, 2017.

Vehicle / Object Detection for Traffic Systems

[12] “Traffic Management Using Convolution Neural Network,” MFR-CNN model for multi-scale object detection in traffic, International Journal of Engineering and Advanced Technology (IJEAT), 2019 .

[13] “ASTM: Autonomous Smart Traffic Management System Using Artificial Intelligence CNN and LSTM,” vehicle detection and prediction using CNN + RNN-LSTM in the CARLA simulator, arXiv, 2024 .

[14] “Optimization of Smart Traffic Management System Using Machine Vision-Based Functionalities with CNN Model,” real-time congestion evaluation, vehicle classification and violation detection, Springer, 2024.

[15] A. Bhowmik, R. Saha, S. Mishra, et al., “Real-Time Object Detection in Road Traffic with Road Maintenance Capability,” in Proc. ICICC, Springer Nature Singapore, 2024, pp. 491–502.

Traffic Flow / Congestion Prediction (CNN-Hybrid Models)

[16] “Efficient Traffic Flow Prediction with CNN-Based Deep Learning Techniques,” Atlantis Press.

[17] “A Novel CNN-GRU-LSTM Based Deep Learning Model for Accurate Traffic Prediction,” Discover Computing, Springer Nature, 2025.

[18] “Traffic Flow Prediction via a Hybrid CPO-CNN-LSTM-Attention Architecture,” Smart Cities (MDPI), 2025.

[19] “City-Wide Traffic Congestion Prediction Based on CNN, LSTM and Transpose CNN,” combining tweet, weather, and traffic data.

[20] “Traffic Prediction Using GPS Based Cloud Data Through RNN-LSTM-CNN Models: Addressing Road Congestion, Safety, and Sustainability in Smart Cities,” SN Computer Science, Springer, 2025.

[21] “Traffic Flow Prediction in Smart City,” CNN-LSTM-SVR hybrid model, IJRASET.

[22] “Traffic Congestion Anomaly Detection and Prediction Using Deep Learning,” hybrid CNN-LSTM/BPNN motorway flow prediction, arXiv, 2020.

[23] “Intelligent Traffic Congestion Forecasting Using BiLSTM and Adaptive Secretary Bird Optimizer for Sustainable Urban Transportation,” Scientific Reports (Nature), 2025.

[24] “Semantic Communication-Empowered Vehicle Count Prediction for Traffic Management,” joint CNN-LSTM SemCom system, arXiv, 2023.

Traffic Signal Control & Safety

[25] “A Dynamic Traffic Light Control Algorithm to Mitigate Traffic Congestion in Metropolitan Areas,” CNN-based signal distribution control with deep Q-learning, NCBI/PMC, 2024.

[26] “Safety in Traffic Management Systems: A Comprehensive Survey,” covering CNN and G-CNN applications to crash-risk and traffic-state identification, arXiv, 2023.

[27] “Research on CNN-BiLSTM Network Traffic Anomaly Detection Model Based on MindSpore,” arXiv, 2025.

[28] “Unveiling the Frontiers of Deep Learning: Innovations Shaping Diverse Domains,” survey covering CNN-LSTM traffic flow prediction models, arXiv, 2023.

Foundational Deep Learning / Object-Detection Architectures

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in Advances in Neural Information Processing Systems (NeurIPS), 2012.

[30] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” in Advances in Neural Information Processing Systems (NeurIPS), 2015.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2016.