International Journal of Education, Business and Economics Research (IJEBER)

ISSN: 2583-3006

Vol. 6, Issue 1, January-February 2026, pp 28-42

To cite this article: Florin Andreescu and Dorin Simionescu (2026). Hybrid Sensing and Cloud Deployed Agent Mesh for AI-Driven Traffic Optimization in Congested Cities. International Journal of Education, Business and Economics Research (IJEBER) 6 (1): 28-42

HYBRID SENSING AND CLOUD DEPLOYED AGENT MESH FOR AI-DRIVEN TRAFFIC OPTIMIZATION IN CONGESTED CITIES

[1]Florin Andreescu and [2]Dorin Simionescu

[1]Florin Andreescu, Bucharest University of Economic Studies, Bucharest, Romania

[2]Dorin Simionescu, Raven Automatizari Vamale SRL

https://doi.org/10.59822/IJEBER.2026.6103

ABSTRACT

Urban traffic congestion in crowded cities is increasingly shaped by short term demand fluctuations, spillback formation, and heterogeneous vehicle dynamics, which challenge fixed time signal plans and reactive controllers. This paper proposes an AI enabled intersection agent that combines (i) a deep learning (DL) predictor for short horizon traffic forecasting (15–60 minutes) and (ii) an intersection level machine learning (ML) controller that applies bounded adaptations of green times based on predicted demand and real-time sensing. To support both model training and closed loop evaluation without requiring immediate access to city scale labeled data, we introduce a two simulator workflow: a correlation aware city behavior generator that synthesizes realistic temporal patterns (e.g., day/night and weekday cycles, holidays, weather dependent modal shifts, and long term trends) to produce training data for the DL predictor, and a microscopic grid simulator that models heterogeneous vehicles (AUTO, VAN, BUS) and intersection geometry to quantify control impact under congestion. The evaluation uses a simplified Intersection Type-1 sensing layout for results reporting and discusses extensibility to Intersection Type-2 designs, including full control and simplified sensor configurations suitable for different deployment constraints. Performance is summarized through an Influence Coefficient (CI) that measures the control induced change in congestion factor between baseline and ML enabled runs. Simulator experiments indicate that the proposed agent can improve congestion outcomes while remaining implementable under cost constrained sensing, and we outline cloud/MLOps requirements for periodic retraining, governance, and safe deployment in real world urban infrastructures. This correlation aware synthetic data generator extends the dynamic historical data generation concept previously introduced for traffic management model training (Author, 2022).

KEYWORDS: Smart city; Traffic signal control; Microscopic simulation; Congestion factor; Multi agent systems; Synthetic data; Spatio-temporal prediction; Cloud architecture; IoT; MLOps.

Contributions

- A hybrid sensing + synthetic data generation pipeline to reduce dependence on dense sensor deployments.
- A decentralized intersection agent meshes with model selection and voting supervisor for robust cycle time adjustments.
- A practical evaluation workflow using a congestion sensitive influence coefficient (CI) for cross scenario comparability.
- A monitored-route case study demonstrating integration constraints and operational considerations.

1.0 INTRODUCTION

Traffic congestion in large cities affects travel time, emissions, and economic productivity. While adaptive control is widely studied, city deployments face fragmented data availability, legacy infrastructure, procurement constraints, and the need for predictable operational behavior. In this work we focus on crowded urban corridors and dense intersection networks where (a) sensing is heterogeneous, (b) traffic patterns shift rapidly (events, incidents, seasonal effects), and (c) operational teams require transparent, modular solutions rather than monolithic black box controllers. (Zhao et al., 2024).

We frame the problem as a closed loop optimization system that must (i) ingest mixed quality data, (ii) predict near-term spatio-temporal demand, (iii) select safe and effective signal parameters, and (iv) provide measurable benefits under simulation and limited field evidence. To support urgency driven publication and deployment timelines, we emphasize an approach that is implementable with conventional ML classifiers/regressors and integrates cleanly into cloud pipelines. (Zhao et al., 2024).

2.0 RELATED WORK

Urban traffic management research spans (i) **short-horizon traffic state prediction** and (ii) **adaptive signal control**. Forecasting work increasingly models road networks as graphs and learns spatio-temporal dependencies using deep architectures. Representative approaches include STGCN (Yu et al., 2018), diffusion based recurrent models such as DCRNN (Li et al., 2018), and adaptive graph methods such as Graph Wave Net (Wu et al., 2019). These models motivate our design choice of a dedicated DL predictor that provides rolling 15–60 min forecasts for each controlled intersection. This correlation aware synthetic data generator extends the dynamic historical data generation concept previously introduced for traffic management model training (Andreescu, 2022).

For signal timing optimization, classical coordinated control and actuated heuristics are complemented by learning based methods, especially deep reinforcement learning (DRL) in multi intersection settings. Recent surveys highlight DRL as one of the most active directions in traffic

signal control (Zhao et al., 2024). Within DRL, Press Light connects learning with max pressure inspired reward design and demonstrates strong performance on arterial networks (Wei et al., 2019). CoLight improves network level coordination by learning dynamic inter agent communication via graph attention (Wei et al., 2019). Our contribution differs by explicitly separating forecasting (DL) from control (ML) and by introducing a two simulator workflow for data generation and closed loop evaluation.

Simulation underpins evaluation in this domain. SUMO is widely used for microscopic traffic simulation (Alvarez Lopez et al., 2018), and City Flow was proposed as a scalable environment for city-wide traffic signal control studies (Zhang et al., 2019). In contrast to using a single simulator for both training and evaluation, we employ two generators with complementary purposes: a city scale behavioral generator to create structured spatio-temporal patterns for DL training and a microscopic grid simulator to quantify control impact via travel time based metrics.

3.0 PROBLEM SETTING AND DESIGN GOALS

G1. Operate with heterogeneous sensors (AI and non-AI cameras) and intermittent IoT feeds.
G2. Enable safe incremental deployment (intersection by intersection) while coordinating at network scale.
G3. Provide fast evaluation and monitoring using metrics interpretable by city stakeholders.
G4. Support emergency overrides (e.g., green wave for responders) without retraining.
G5. Fit within cloud native MLOps pipelines for retraining, validation, and versioned rollouts.

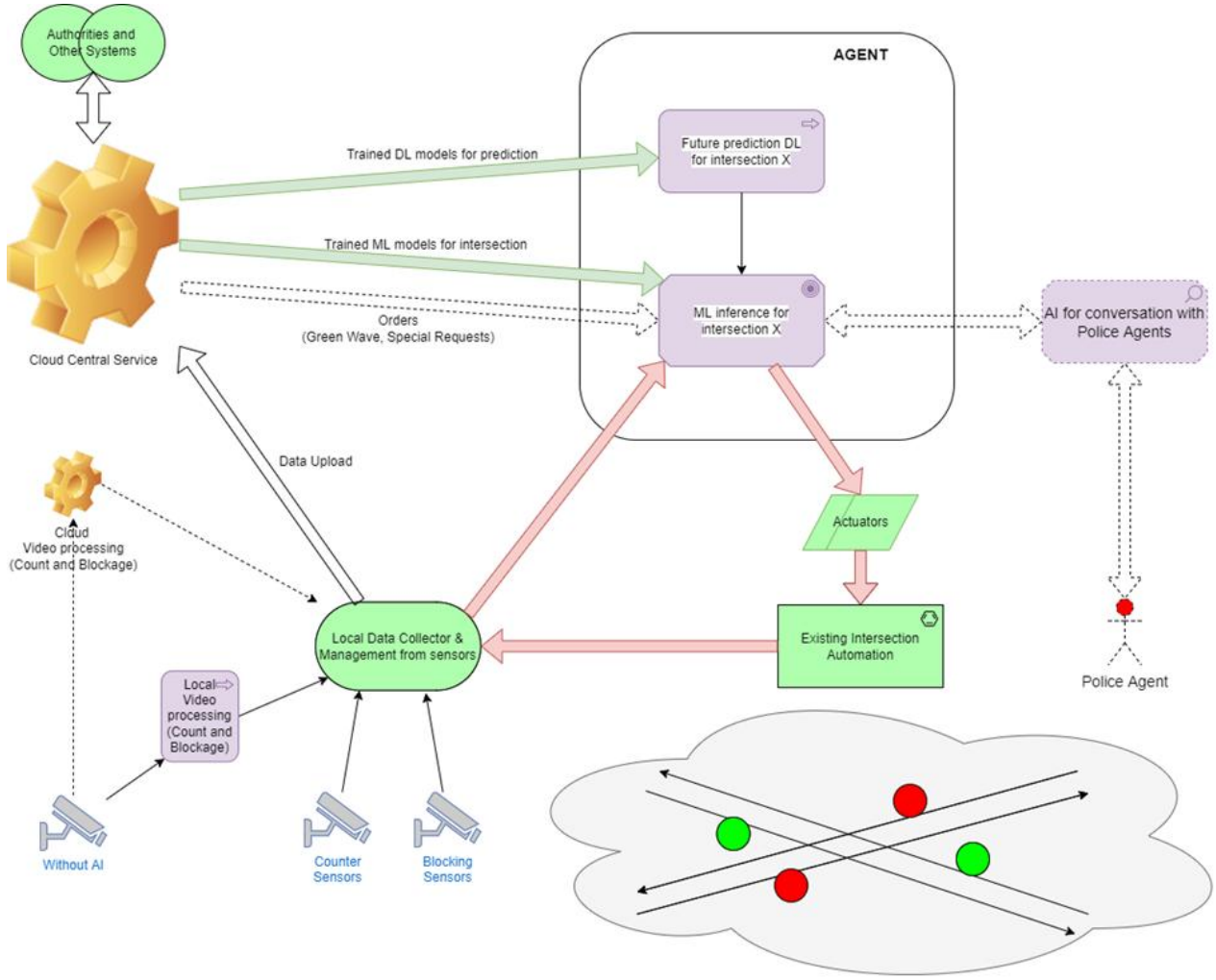
4.0 PROPOSED FRAMEWORK

4.1 Intersection Agent: ML Controller + DL Traffic Predictor

We model each signalized intersection as an autonomous agent that continuously estimates the near future load and adapts its signal timing within safety bounds. The agent consists of two coupled components:

As shown in the following figure, we summarize the component discussed in this subsection.

Figure 1. Intersection agent architecture (DL future-traffic prediction + ML inference/control), fed by local sensing and coordinated by a central service (including green-wave / special requests).



We use this figure to support the formal definitions and the experimental discussion that follows.

- A **deep learning (DL) predictor** that forecasts the traffic level for the next decision horizon (typically 15–60 minutes) using city scale context.
- A lightweight **machine learning (ML) controller** that converts the forecast and local measurements (queues, arrivals, phase utilization) into incremental green time adjustments for the current cycle plan.

This separation is intentional: the DL predictor focuses on learning broader spatio-temporal patterns, while the ML controller remains interpretable and easily constrained (e.g., minimum pedestrian green, maximum cycle length, emergency overrides).

4.2 Simulator/Generator A: City Scale Behavior Generator for DL Training

The first simulator is a city scale behavior generator whose primary goal is **data generation** for training and stress testing the DL predictor. It produces approximate intersection level traffic indicators (e.g., density/flow/queue proxies) for any location in the city over long time spans. The generator embeds a set of domain correlations ("generation principles") that emulate realistic non stationarity and multi-factor effects.

The initial objective is to control the presence/strength of these injected correlations and evaluate how accurately the DL predictor learns them, before integrating the predictor into the closed loop control system.

4.2.1 Injected Correlation Families

The generator injects controlled, configurable correlation families that emulate real-world demand shifts: (i) **diurnal effects** (hour-of-day) modulating density, braking intensity, and attractor/repulsor ("magnet") strengths by road/section class; (ii) **weekday effects** (e.g., amplified In/Out-point magnetism on Fridays and density on Sundays); (iii) **calendar events** (public holidays and holiday periods) shifting demand peaks toward the day(s) before and the final day of the period; (iv) **weather-driven mode choice**, where temperature/precipitation/wind speed update the transport-type distribution and wind direction adds cost for pedestrians/bicycles (triggering possible mode switches); and (v) a **long-term trend** where the private-auto share increases by ~1% per year (configurable). By adjusting these parameters, we stress-test the DL predictor's ability to learn known patterns before integrating it into the control loop.

4.3 Simulator/Generator B: Microscopic Grid Simulator for Control Evaluation

The second simulator is a microscopic evaluation environment used to quantify control benefits under controlled conditions. It represents the road network as a rectangular mesh of intersections and lanes, and supports randomized route choices (left/right/straight) to create diverse flows.

As shown in the following figure, we summarize the component discussed in this subsection.

Figure 2. Type-2 intersection geometry used in the microscopic simulator (schematic).

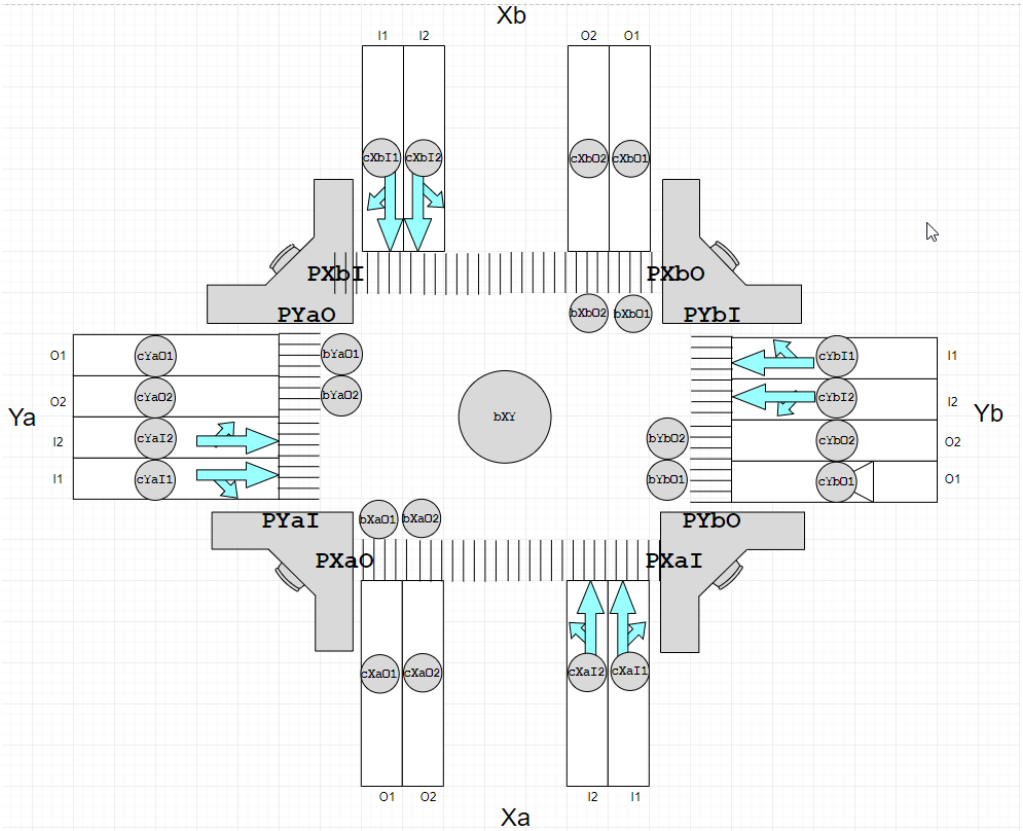
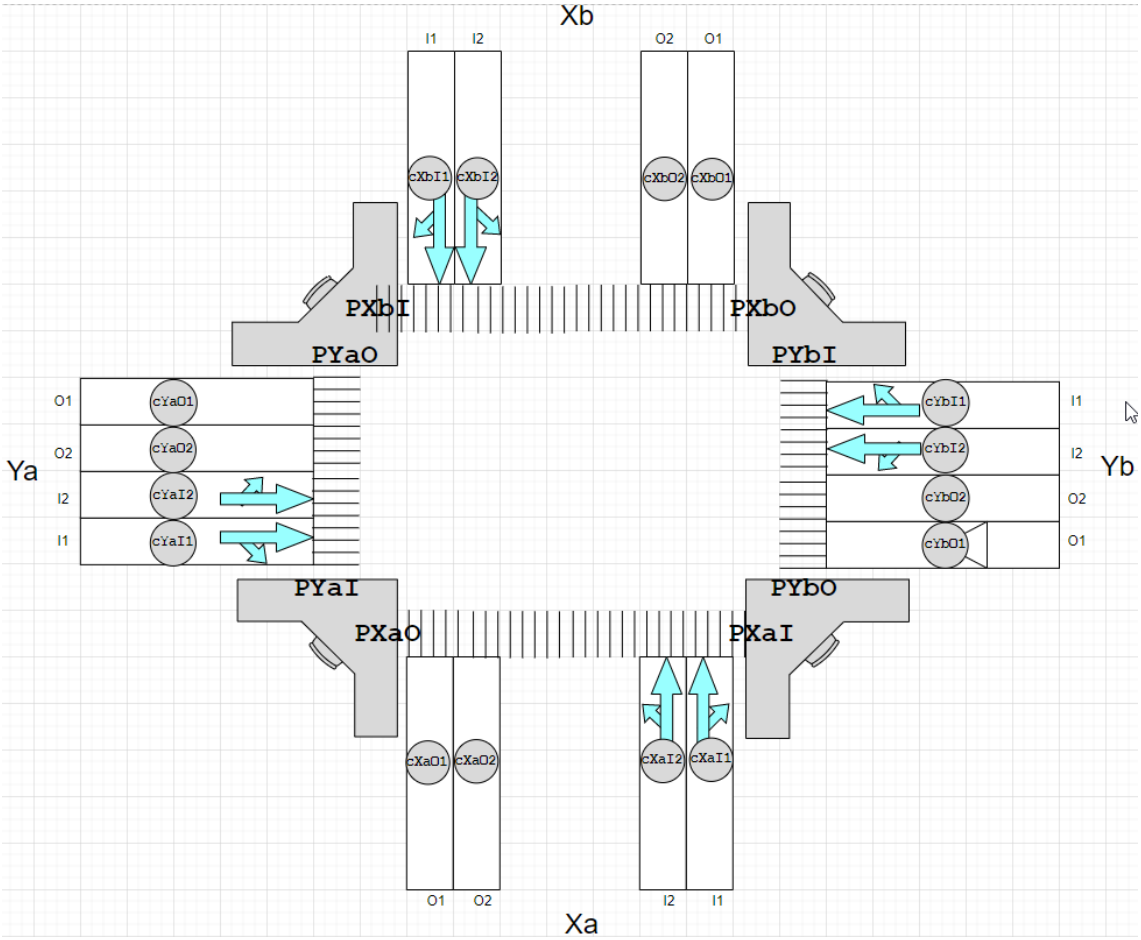


Figure 2 depicts the **full-control** Intersection Type-2 model. In this configuration, each sensor is an AI camera that estimates: (1) the **number of vehicles** visible in the scene along with their speeds, and (2) the **vehicle debt**—the number of vehicles crossing a reference line per unit time.

The full model includes sensors for inbound traffic (cxxxIx), outbound traffic (cxxxOx), blocking traffic inside the intersection (bxx), and pedestrian flows via PX and PY sensors. With full control, the ML controller is explicitly informed about internal blocking conditions and can adjust green allocations to reduce the risk of complete gridlock.

In addition to the full model, we also consider a **simplified** Intersection Type-2 configuration that includes only inbound and outbound sensors. This variant omits the internal blocking sensors, providing reduced control fidelity but lower deployment complexity.

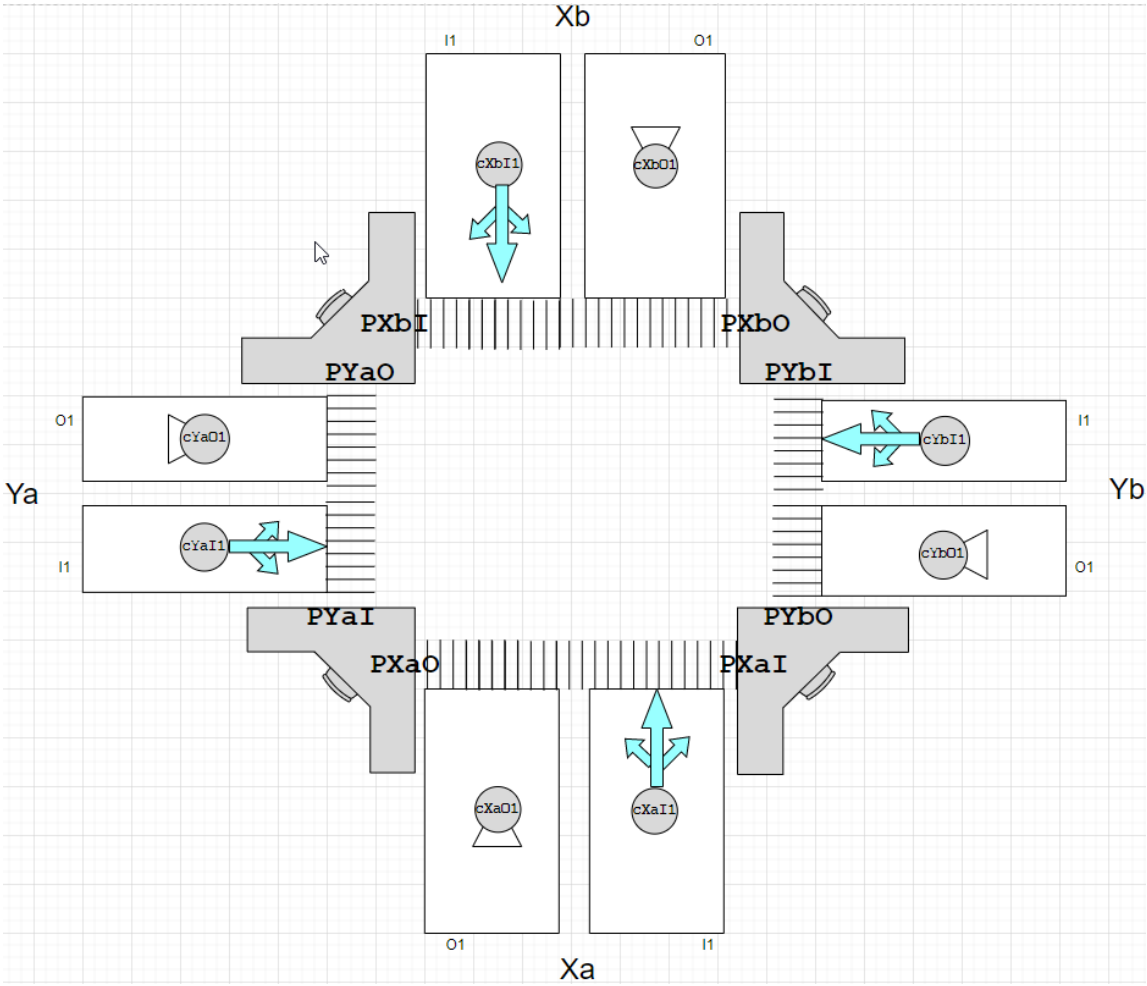
Figure 3. Simplified Intersection Type-2 sensor layout (inbound/outbound sensing only).



This simplified layout supports basic demand-adaptive signal timing while avoiding the additional cost and calibration required for blockage detection.

The results reported in this study were obtained using **Intersection Type-1** configurations. To ensure completeness, we provide the simplified Intersection Type-1 layout used in experiments.

Figure 4. Simplified Intersection Type-1 sensor layout used in experiments.



Intersection Type-1 preserves inbound/outbound sensing for demand estimation while minimizing on-site instrumentation.

4.3.1 Vehicle and Intersection Modeling

We simulate three vehicle classes—bus, van, and passenger car—each with distinct physical characteristics (maximum speed, acceleration, braking). The simulator instantiates a parameterized intersection geometry (Type-2 intersection template) and collects signal states, per phase statistics, and full vehicle trajectory histories.

We evaluated several Intersection Type-2 configurations to reflect different deployment constraints: (i) approaches with 1, 2, or 3 lanes per direction (per sense), and (ii) **full-control** sensing versus a **simplified** sensing layout with fewer sensors. This allows the controller to be assessed under both high instrumentation and cost constrained settings.

Vehicle Dynamic Behavior

- To reproduce the behavior of a vehicle, we use a simplified form of the Intelligent Driver Model (IDM) introduced by Treiber, Hennecke, and Helbing (2000).
- At each simulation iteration, the position and speed of each vehicle are updated using the following discrete time kinematic approximations:

$$v(t + dt) = v(t) + a(t) \cdot dt$$
$$x(t + dt) = x(t) + v(t) \cdot dt + a(t) \cdot dt^2 / 2$$

Here, x is longitudinal position, v is speed, a is acceleration, and dt is the simulation time step. Vehicles are sampled dynamically using type dependent weights (p_weight) and are parameterized by length (l), desired minimum gap to the leader (s0), driver reaction time (T), maximum speed (v_max), maximum acceleration (a_max), and maximum braking deceleration (b_max). For visualization, types are rendered in distinct colors (BUS: green, VAN: blue, AUTO: brown).

A typical microscopic snapshot illustrating queue build-up is shown in the following figure.

Figure 5. The formation of "waiting waves" at the entrance to the intersection.

This phenomenon emerges naturally from heterogeneous vehicle parameters and stochastic turning decisions, and it is a key mechanism behind congestion propagation in the grid.

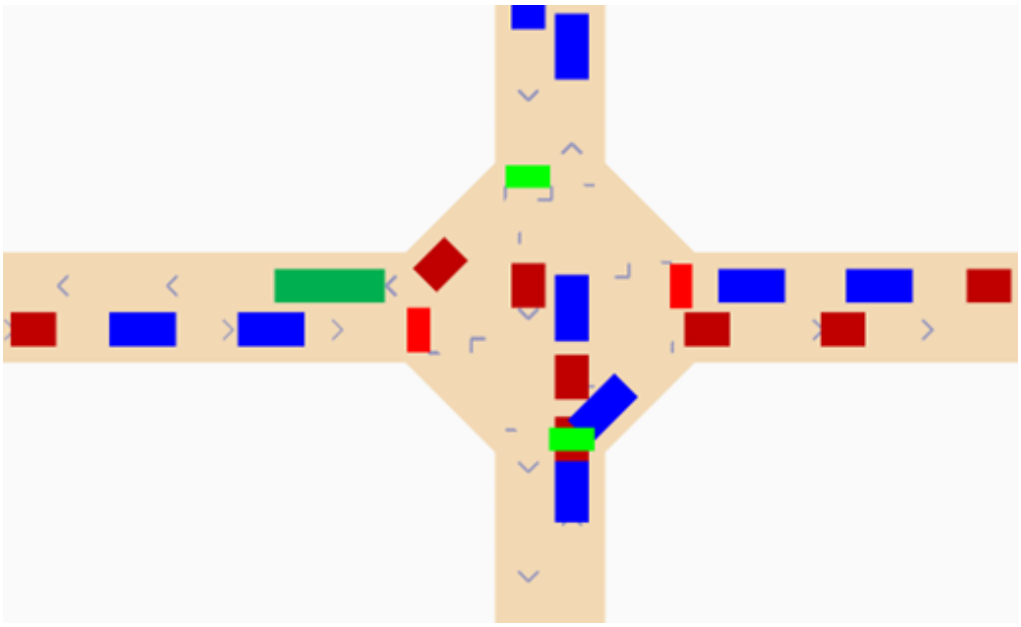
Table 1. Vehicle types and parameters used in the microscopic simulator

Type	Color	p_weight	l (m)	s0 (m)	T (s)	v_max (m/s)	a_max (m/s²)	b_max (m/s²)
AUTO	Brown	9	4	4	1	10.0	2.5	7.5
VAN	Blue	4	6	6	1	8.0	2.0	6.0
BUS	Green	3	10	10	1	6.0	1.5	4.5

Where p_weight is a weight for the vehicle generator.

As shown in the following figure, we summarize the component discussed in this subsection.

Figure 6. Qualitative snapshot of simulated vehicles approaching and traversing an intersection.



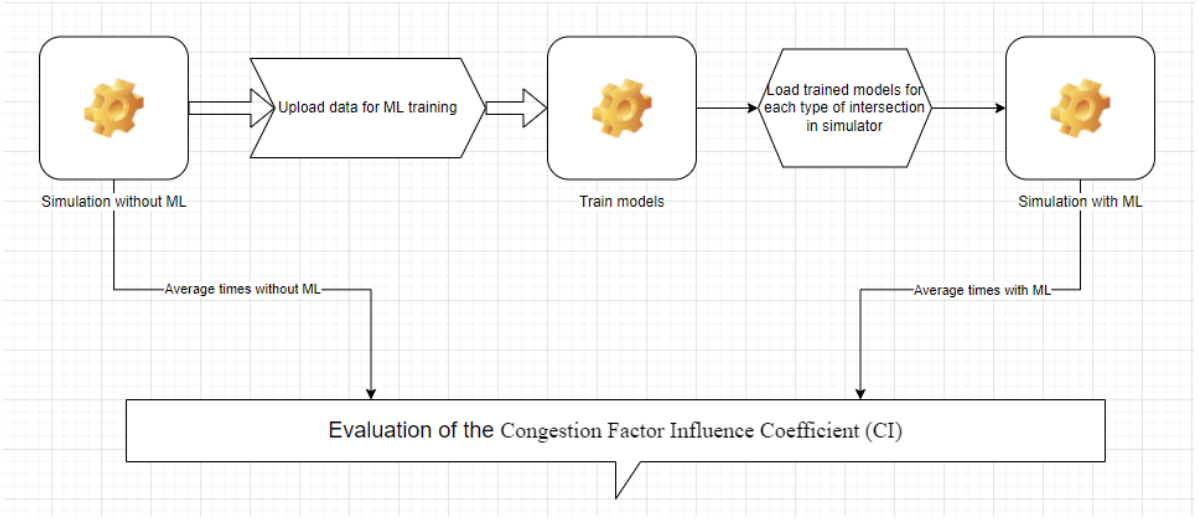
We use this figure to support the formal definitions and the experimental discussion that follows.

4.3.2 Closed Loop Test Protocol

We evaluate the intersection agent with a two run protocol:

As shown in the following figure, we summarize the component discussed in this subsection.

Figure 7. Two-run experimental protocol: baseline simulation → ML training → simulation with ML control → CI evaluation.



We use this figure to support the formal definitions and the experimental discussion that follows.

- **Run 1 (baseline):** execute the simulation without any trained intersection ML controller (fixed time or no adaptive signal plan). Collect per intersection feature/label pairs for training.
- **Training:** train the local ML controllers using the data collected in Run 1 (optionally augmented by city scale DL forecasts).

- **Run 2 (controlled):** re-run the simulation with each intersection connected to its trained ML controller, which adjusts green times within predefined limits.

The outcome comparison uses (i) the congestion factor computed from simulator statistics and (ii) an influence coefficient (CI) that aggregates improvements across the network and scenarios. This enables consistent comparisons across traffic loads.

4.4 Implementation Notes and Safety Constraints

To ensure operational feasibility, the ML controller applies bounded adjustments: minimum green per phase, maximum cycle length, and optional fairness constraints to prevent starvation. Emergency 'green wave' requests are handled as high priority overrides that temporarily supersede ML adjustments, while preserving audit logs for post incident analysis.

5.0 EXPERIMENTAL SETUP

We evaluate the proposed framework using a paired, closed loop protocol in Simulator/Generator B. For each scenario, we run (i) a **baseline simulation without ML control** to collect training data and (ii) a **simulation with ML control enabled** under the same configuration to quantify impact using the CI metric defined in Section 6.

5.1 Scenario Factors and Demand Regimes

Scenarios vary demand intensity and temporal variability to emulate urban traffic fluctuations. The boundary vehicle generators can be configured with periodic functions (e.g., sinusoid or square/rectangle waves) or step changes to stress the controller under both smooth and abrupt demand shifts. Generator A can additionally inject structured correlations (Section 4.2.1) to produce training data for the DL predictor.

5.2 Features and ML Controllers

At each intersection, we compute per cycle features from local counts and queue proxies, optionally augmented with predicted near future traffic from the DL predictor. We evaluate three feature variants (A/B/C) and train multiple supervised models: k-NN, SVM, Naive Bayes, Decision Tree, Random Forest, and a small DNN. During the control run, the trained model outputs a score that adjusts the next cycle's green allocation within predefined bounds.

5.3 Control Limits and Update Frequency

Green-time adjustments are applied at each phase transition (i.e., once per cycle per approach group). To prevent oscillations and maintain safety, changes are bounded (minimum/maximum green per direction) and scaled by a correction coefficient cycle_correction_coef (default 0.5 in our implementation).

Table 2. Experimental setup factors and levels used for evaluation.

Experimental factor	Levels used	Rationale
Demand variation function	Sinusoid; Rectangle; Steps	Tests controller robustness to smooth vs abrupt demand changes.

Injected correlation regime (Generator A)	Diurnal/weekday; Calendar events; Weather & mode choice; Long-term trend	Stress-tests the DL predictor under structured shifts before closed-loop control.
Sensor coverage	Counters only; Counters + blockage sensors; (optional) AI camera count/blockage	Evaluates performance under heterogeneous and partial sensing.
Control model/feature set	Feature variants 8A/8B/8C; multiple classifiers	Compares local controller sensitivity to inputs and model class.

6.0 METRICS AND INFLUENCE COEFFICIENT

We evaluate congestion using the Travel Time Ratio between an origins A and a destination B:

- Tt = travel time under traffic conditions (seconds).
- T0 = travel time in a free flow (empty city) condition (seconds).

The congestion factor is defined as:

$$FC = Tt / T0$$

For a simulation run, the General Congestion Factor is defined as:

$$GCF = med(Tt) / med(T0)$$

We run the simulator twice under identical demand and geometry:

- FCG = GCF for the baseline run (without ML).
- FCGML = GCF for the ML-controlled run.

The influence coefficient is the relative reduction of congestion due to ML control:

$$CI = (FCG - FCGML) / FCG$$

Since med(T0) and med(T0ML) are equal by construction, CI can be expressed using only route travel times:

$$CI = (med(Tt) - med(TtML)) / med(Tt)$$

These definitions are used consistently across all experimental scenarios and repeated runs.

7.0 RESULTS

We evaluate the closed loop intersection control using the two run protocol (Figure 9) and report the Congestion Factor Influence Coefficient (CI) defined in Section 6. Experiments vary both (i) the feature variants used for the intersection ML controller (A/B/C) and (ii) the shape of the traffic variation functions used to generate demand patterns (sinusoid, rectangle, steps).

As shown in the following figure, we summarize the component discussed in this subsection.

Intersection ML training setup and feature variants.

Figure 8. Classifier accuracy by feature variant (8A/8B/8C) under a fixed cycle configuration

Model	Rate	Cycles	8A_accuracy (%)	8B_accuracy (%)	8C_accuracy (%)
knn	8	20	49	50	50
svc	8	20	56	57	56
nb	8	20	32	36	41
dtr	8	20	58	64	51
rfc	8	20	59	64	57
kfold	8	20	32	32	35

For training the intersection ML models, we executed a simulation of 20 cycles, each cycle lasting 1000 seconds, with a reference vehicle generation rate of 8.

During feature engineering, we retained three feature variants:

Variant A: feature_names = ['cycle_time_WE','L','R','S','upstream','upstream_last_cycle', 'passed_in_last_cycle', 'total_pred']

Variant B: feature_names = ['road_id','L','R','S','upstream','upstream_last_cycle', 'passed_in_last_cycle', 'total_pred']

Variant C: feature_names = ['road_id','L','R','S','upstream','upstream_last_cycle', 'total_pred']

Based on the generated data and the preprocessing pipeline, we trained six ML model types: knn (K Neighbors Classifier), svm (Support Vector Classifier), nb (Gaussian Naive Bayes), dtr (Decision Tree Classifier), rfc (Random Forest Classifier), and kfold (DNN model).

The DNN (kfold) model uses the following structure:

```
model.add(Dense(model_size, input_dim=model_size, activation='relu'))
model.add(Dense(model_size * model_size, activation='relu'))
model.add(Dense(model_size, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

Where model_size is the number of features for variants (A, B, C).

The accuracy values obtained across the 18 training runs (percent) are reported in Figure 9.

We use this figure to support the formal definitions and the experimental discussion that follows. Across tested scenarios, feature variant **B** yields the most consistent gains, and several classifiers exhibit sensitivity to the demand variation regime. For example, SVC performs comparatively better under smooth traffic variation, while NB can behave better under abrupt changes—suggesting that no single model is uniformly optimal across all regimes. This motivates a future 'voting' or met selection supervisor that can switch the local controller model based on recent performance and forecast error.

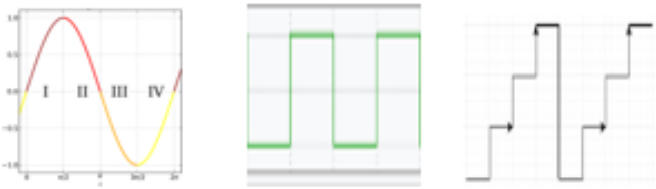
As shown in the following figure, we summarize the component discussed in this subsection.

Figure 9. Coefficient reduction (coef.red%) by model across traffic-variation generator functions (sinusoid, rectangle, steps).

Cycle length: 500 sec

Cycle number: 1

Gen Function:



1

model	coef.red%	coef.red%	coef.red%
knn	43.65	55.47	43.83
svm	47.13	57.79	49.65
nb	46.89	58.44	48.4
dtr	44.34	51.66	41.91
rrfc	-20.17	26.48	2.91

We use this figure to support the formal definitions and the experimental discussion that follows. Figure 9 summarizes classifier accuracy across the three feature variants (8A/8B/8C) under a fixed cycle configuration, highlighting stronger performance for the best feature set. Figure 9 reports the percentage reduction in the congestion related coefficient (coef.red%) across demand generation functions. Notably, the worst observed CI value in the reported experiments is **14.64%** (SVM), while the remaining configurations achieve higher reductions. The results suggest that Naïve Bayes (NB) is particularly effective in sparse-traffic scenarios, where the vehicle flow rate ("debit") varies sharply over short time intervals, yielding the highest observed congestion reductions. Some configurations (e.g., RFC) can yield poor or even negative effects under certain training conditions, indicating the need for robust validation and drift monitoring prior to deployment.

8.0 DISCUSSION AND PRACTICAL DEPLOYMENT

Architecture note (brief): In addition to the algorithmic workflow, we elaborated deployment architectures for **Azure** and **Google Cloud Platform (GCP)** to validate an end-to-end pipeline (ingestion, training, deployment, monitoring, rollback). We keep cloud/MLOps details concise because the scientific contribution is the intersection agent design and the two-simulator evaluation protocol.

Implementation note: The proposed framework is provider agnostic and can run on municipal servers, edge devices, or cloud infrastructure. Deployment is treated as an engineering constraint (reliable ingestion, monitoring, rollback), not as a scientific contribution.

Discuss how heterogeneous sensing affects model stability, how often retraining is needed, and how to monitor drift in calendar/weather effects. Explain the separation between DL forecasting and ML control as an engineering choice for transparency and incremental deployment.

9.0 CONCLUSION

The simulation results are encouraging and support the feasibility of the proposed approach for training and deploying intersection-specific prediction and control models across heterogeneous intersection types. The two generator workflow—a correlation aware city behavior generator for forecasting-data creation and a microscopic grid simulator for closed loop evaluation—provides a practical basis for iterating on model design before field deployment.

For real world adoption, a fully automated cloud training pipeline is required to periodically retrain models on large volumes of streaming data and to manage robust MLOps practices (versioning, validation gates, staged rollout, and monitoring). The design explicitly considers interactions between cloud components and terrestrial infrastructure, as well as the human factor: hybrid sensing scenarios are supported, combining AI-enabled cameras with legacy sensors and IoT integration. Real-time processing and analytics, together with visualization and command interfaces, are essential to enable authorities to intervene and influence system behavior when needed.

We also investigated how the two major cloud ecosystems (Azure and Google Cloud Platform) can support CI/CD, AI pipelines, and geographically distributed deployments across city infrastructure. Such systems demand strict governance over trained models and deployments spanning cloud, on premises, and edge components.

Suggestions for further research include:

- Integrating large language models (LLMs) and generative AI as supervisory layers that interact with human operators for scenario planning, explanation, and audit ability.
- Green-wave coordination for emergencies, authorities, and disaster response, including safe override policies and compliance constraints.

REFERENCES

Treiber, M., Hennecke, A., & Helbing, D. (2000). Congested traffic states in empirical observations and microscopic simulations. *Physical Review E, 62*(2), 1805–1824. https://doi.org/10.1103/PhysRevE.62.1805

Alvarez Lopez, P., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., & Wießner, E. (2018). Microscopic traffic simulation using SUMO. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE.

Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR 2018)*.

Wei, H., Chen, C., Zheng, G., Wu, K., Gayah, V., Xu, K., & Li, Z. (2019). PressLight: Learning max pressure control to coordinate traffic signals in arterial network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)* (pp. 1290–1298). https://doi.org/10.1145/3292500.3330949

Wei, H., Xu, N., Zhang, H., Zheng, G., Zang, X., Chen, C., Zhang, W., Zhu, Y., Xu, K., & Li, Z. (2019). CoLight: Learning network-level cooperation for traffic signal control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*. https://doi.org/10.1145/3357384.3357902

Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph WaveNet for deep spatial-temporal graph modeling. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI 2019)* (pp. 1907–1913). https://doi.org/10.24963/ijcai.2019/264

Yu, B., Yin, H., & Zhu, Z. (2018). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI 2018)* (pp. 3634–3640). https://doi.org/10.24963/ijcai.2018/505

Zhang, H., Feng, S., Liu, C., Ding, Y., Zhu, Y., Zhou, Z., Zhang, W., Yu, Y., Jin, H., & Li, Z. (2019). CityFlow: A multi-agent reinforcement learning environment for large scale city traffic scenario. *arXiv*. https://doi.org/10.48550/arXiv.1905.05217

Zhao, H., Dong, C., Cao, J., & Chen, Q. (2024). A survey on deep reinforcement learning approaches for traffic signal control. *Engineering Applications of Artificial Intelligence, 133*(Part A), 108100. https://doi.org/10.1016/j.engappai.2024.108100

Andreescu, F. (2022). A dynamic generator for machine learning training for traffic management systems. Informatica Economica, 26(4), 55–65. https://doi.org/10.24818/issn14531305/26.4.2022.05

Figure Table

Table 3. Figure list and titles (Figure Table).

Figure	Title
1	Intersection agent architecture (DL future-traffic prediction + ML inference/control), fed by local sensing and coordinated by a central service (including green-wave / special requests).
2	Type-2 intersection geometry used in the microscopic simulator (schematic).
3	Simplified Intersection Type-2 sensor layout (inbound/outbound sensing only).
4	Simplified Intersection Type-1 sensor layout used in experiments.
5	The formation of "waiting waves" at the entrance to the intersection.
6	Qualitative snapshot of simulated vehicles approaching and traversing an intersection.
7	Two-run experimental protocol: baseline simulation → ML training → simulation with ML control → CI evaluation.
8	Classifier accuracy by feature variant (8A/8B/8C) under a fixed cycle configuration.
9	Coefficient reduction (coef.red%) by model across traffic-variation generator functions (sinusoid, rectangle, steps).

