

Algerian People's Democratic Republic
Ministry of Higher Education and Scientific Research
University of Ain Temouchent - Belhadj Bouchaib
Faculty of Science and Technology



Faculty of Science and Technology
Department of Mathematics and Computer Science
Master Thesis

For the Master's Degree in Computer Science
Option: CYSIA

Thesis title:

**Deep RL with Optimization-Based Refinement for Mobility
Prediction in Smart Cities Using MEC and V2X Communication**

Presented by:
M. Yassine BENKRAMA

Defended on: June 22, 2026

In front of the jury composed of:

President: Dr. D. BOUHALOUAN

U.A.T.B.B

Examiner: Dr. K. CHAOUI

U.A.T.B.B

Supervisor: Dr. Ali BENZERBADJ

U.A.T.B.B



“And say: ‘Work, for Allah will see your deeds, and so will His Messenger and the believers’.”

(Surah At-Tawbah, Verse 105)

“They say dreams are meant for the night, But I chose to turn them into my reality.”

At every stage, I came to understand that success is not given; it is built and earned. Education has taught me to aim high, to remain strong in the face of challenges, and to always believe that knowledge is a light granted by Allah to those who persevere with sincerity. May this project stand as a humble proof and a reminder that, as long as determination guides our steps, no summit is beyond reach.

DEDICATION





I dedicate this dissertation to...

To my mother, Mokrane Kheira

The official source of courage, whose prayers (duas) have surely supported this project more than any algorithm ever could!

To my father, Benkrama Abdelhafid

The quiet king of “take care of yourself,” the one who does not speak much, yet whose every glance says: *“I believe in you... but make sure to check the cables as well.”*

To my beloved sister, Fatima whom I adore

An expert in spontaneous motivation, always there to say: *“You are stressing too much. Come on, let’s eat first.”*

To my adorable sister, Khedidja, the apple of my eye

My unofficial assistant, full-time emotional counselor, and specialist in saying: *“You will make it, I can feel it!”*

To my lifelong friends

In memory of our sincere and deep friendship, and of the wonderful moments we have shared together.

To myself

A brave warrior of sleepless nights, tamer of reports, coffee enthusiast, who survived Word, crashing simulations, and the terrifying question: *“Have you finished your final project yet?”*

Yassine

Thanks



*First and foremost, I would like to express my deepest gratitude to **Allah Almighty** for granting me the strength, patience, and perseverance necessary to complete this work successfully.*

*I would like to extend my sincere and profound thanks to my **supervisor Dr. ALI BENZERBADJ**, for his valuable guidance, continuous support, constructive advice, and availability throughout the preparation of this dissertation. His expertise and encouragement greatly contributed to the successful completion of this work.*

*I would also like to express my heartfelt appreciation to **Dr. ABDEL RAZAK HAMADOU ADAMOU** for his assistance, insightful remarks, continuous follow-up, and valuable support, which significantly contributed to the improvement of this dissertation.*

*My special thanks also go to my **professor Dr. PETER SARCEVIC in Serbia**, whose guidance, support, and academic assistance were of great value throughout this experience. His kindness, encouragement, and expertise greatly enriched both my academic and personal journey.*

*I would like to express my sincere gratitude to the administrative staff of the (UAT) for their support, availability, and administrative assistance throughout my academic journey. A special and profound thank you goes to the Erasmus Coordinator, **Dr. Driss NEHARI**. His guidance, dedication, and tireless efforts were instrumental in making this invaluable international mobility experience possible.*

*I would also like to extend my appreciation to the administrative and academic staff of **Subotica Tech** for their warm welcome, support, and valuable assistance during my academic stay in Serbia.*

*My sincere thanks to the Jury President **Dr. D. BOUHALOUAN**, and the Examiner **Dr. K. CHAOUI**, for their time, expertise, and constructive evaluation of this dissertation.*

Finally, I would like to thank everyone who contributed, directly or indirectly, to the completion of this dissertation and supported me throughout this important stage of my academic journey.

Thank You all.

List of figures

List of table

General Introduction01

Chapter I : Background and state of the Art

I.1 Introduction.....05

I.2 Urban traffic as a vital issue05

I.3 Intelligent Transport System (ITS): A turning point of the history of road transportation management.....06

I.4 Mobility Prediction, the Foundation of Intelligent Transport System07

 I.4.1 Deep Reinforcement Learning: A Paradigm Shift.....08

 I.4.2 The Hybrid Frontier: Combining Learning and Optimization.....08

I.5 V2X Communication.....09

 I.5.1 Overview and Architecture09

 I.5.2 Vehicle-to-Vehicle (V2V) Communication.....10

 I.5.3 V2I Communication10

 I.5.4 Vehicle-to-Network (V2N) Communication.....11

 I.5.5 Vehicle-to-Pedestrian (V2P) Communication.....11

I.6 Multi-Access Edge Computing (MEC).....12

 I.6.1 Definition and Core Architecture12

 I.6.2 MEC in vehicular networks: reducing latency13

 I.6.3 Task Offloading and Resource Allocation13

 I.6.4 MEC as an enabler for real-time mobility intelligence14

I.7 Mobility and Traffic Prediction.....14

 I.7.1 Definitions and Range of Applicability.....14

 I.7.2 Historical View of the Prediction Systems15

 I.7.3 Challenges in dynamic urban environment15

I.8 Deep Reinforcement Learning.....16

 I.8.1 Reinforcement Learning Basics17

 I.8.2 Deep Reinforcement Learning18

I.8.2.1 Deep Q-Network (DQN).....	19
I.8.2.2 Proximal Policy Optimization (PPO)	20
I.9 State of the Art	22
I.9.1 Classical Approaches.....	22
I.9.1.1 ARIMA-based Methods	22
I.9.1.2 Kalman Filtering.....	23
I.9.2 Machine Learning and Deep Learning Approaches	24
I.9.2.1 Recurrent Neural Networks	25
I.9.2.2 Graph Neural Networks and Attention Mechanisms	25
I.9.3 Reinforcement Learning Approaches	26
I.9.3.1 Trajectory Prediction	26
I.9.3.2 Hierarchical Hybrid DL-RL Framework	27
I.9.4 Critical analysis and Problem statement.....	28
I.9.4.1 Research Objective	29
I.9.4.2 Key Contributions	29
I.9.4.3 Hypothesis.....	29
I.10 Conclusion	29

Chapter II: System Model and Problem Formulation

II.1 Introduction	32
II.2 System Architecture.....	32
II.2.1 Smart City System Overview	32
II.2.1.2 Smart City Road Network	33
II.2.1.3 Roadside Units (RSUs) - Sensor Fusion Nodes.....	33
II.2.1.4 5G Next-Generation Base Stations (gNB).....	34
II.2.1.5 MEC Server - Edge AI Intelligence Hub.....	34
II.3 Data and Assumptions	35
II.3.1 Data Source and Road Network.....	35
II.3.2 Dataset Generation and Temporal Structure.....	36
II.3.3 RSU State Features — Collected Data Variables	37

II.3.3.1 Rush Hour Demand Model.....	37
II.3.4 Directional Traffic Data.....	40
II.3.5 Observation Window and Input Tensor.....	41
II.3.6 Traffic Dynamics Model — Spillover and Non-Stationarity	42
II.3.7 Network and System Assumptions	42
II.4 Problem Definition.....	43
II.4.1 Nature of the Prediction Task	43
II.4.2 Formal Problem Statement	43
II.5 Mathematical Formulation	44
II.5.1 Problem Type	44
II.5.2 Objective Functions	44
II.5.3 Constraints	45
II.5.4 Complete Problem Statement	46
II.6 Conclusion.....	46

Chapter III: Proposed Methodology

III.1 Introduction.....	48
III.2 Overview of the Hybrid Approach	48
III.3 Proposed System Architecture.....	49
III.3.1 Environment	50
III.3.2 Agent.....	51
III.3.3 Metaheuristic Optimization.....	54
III.3.4 PPO Training Loop — PPO Stats / Updates	54
III.4 Reinforcement Learning Formulation	54
III.4.1 State Space	54
III.4.2 Action Space.....	54
III.4.3 Simulation Environment	55
III.4.3.1 The Gymnasium Framework.....	55
III.4.3.2 Observation Space	56
III.4.3.3 Reward Function.....	56

III.4.3.4 Episode Management and Monitoring	56
III.5 PPO Algorithm.....	57
III.5.1 Actor-Critic Architecture	57
III.5.2 Training Procedure and Hyper parameters.....	57
III.6 Optimization-Based Refinement Layer	57
III.6.1 Role of the Optimization Module	58
III.6.2 Optimization Problem Formulation	58
III.6.3 Genetic Algorithm (GA) Optimizer.....	59
III.6.4 Particle Swarm Optimization (PSO) Optimizer	59
III.7 Integration with MEC and V2X	60
III.8 Algorithm Pseudo-code	60
III.9 Conclusion	61

Chapter IV: Experimental Results and Performance Evaluation

IV.1 Introduction.....	64
IV.2 Experimental Setup	64
IV.2.1 Dataset and Evaluation Protocol	64
IV.2.2 Software and Hardware Environment.....	65
IV.3 Baseline Methods.....	65
IV.3.1 LSTM Supervised Classifier	65
IV.3.2 ARIMA	66
IV.3.3 Kalman Filter.....	67
IV.3.4 Random Classifier	67
IV.4 Evaluation Metrics	68
IV.5 Training Process and Convergence	68
IV.5.1 Stage 1: PPO Offline Training	68
IV.5.2 Stage 2: Online Metaheuristic Optimization.....	70
IV.6 Static Evaluation Results	70
IV.6.1 Overall Accuracy and Macro-F1	70

IV.6.2 Per-Direction Accuracy	71
IV.7 Non-Stationary Evaluation	72
IV.7.1 Rolling Accuracy under Regime Shifts	72
IV.7.2 Per-Segment Accuracy Analysis	73
IV.8 Computational Performance Analysis.....	74
IV.8.1 Inference Latency and Throughput.....	75
IV.8.2 Training Time Comparison	75
IV.9 Comparative Analysis	76
IV.9.1 Accuracy–Latency Trade-off	76
IV.9.2 Suitability for Real-World V2X Deployment	76
IV.9.3 Impact of Class-Weight Reward Shaping	76
IV.10 Closed-Loop Deployment Validation via SUMO Simulation	77
IV.10.1 Simulation Environment	77
IV.10.2 Hybrid Max-Pressure + DRL Direction Boost Controller	78
IV.10.3 Results.....	79
IV.11 Discussion: Results Analysis and Limitations	80
IV.11.1 Interpretation of Static and Non-Stationary Results	80
IV.11.2 SUMO Deployment Results Analysis	81
IV.11.3 Limitations	81
IV.12 Conclusion	82
Conclusion and perspective.....	84
Bibliography	88
Abstract.....	94

Chapter I : Background and state of the Art		
Figure	Title	Page
01	V2X connectivity modes composing Internet-of-Vehicle (IoV) paradigm [31].	10
02	Edge computing architecture [36]	13
03	Framework of reinforcement learning	17
04	Deep reinforcement learning agent of traffic [54].	19
05	Proximal Policy Optimization (PPO) architecture [59]	21

Chapter II : System Model and Problem Formulation		
Figure	Title	Page
01	System Architecture	33
02	Coverage Map of the Top 20 Roadside Units (RSUs) and GNB/MEC Infrastructure in Nördlingen, Germany.	36
03	Temporal Evolution of Traffic Demand with Morning and Evening Peak Hours.	38
04	Macroscopic Fundamental Diagram Illustrating the Relationship Between Traffic Density and Vehicle Speed.	39
05	Composition of the Roadside Unit (RSU) State Vector X.	40
06	Composition of the Directional Density Metric D.	41

Chapter III : Proposed Methodology		
Figure	Title	Page
01	Complete architecture of the proposed hybrid PPO + GA/PSO traffic trend prediction system.	51
02	Neural Network of Temporal Convolutional Network	52

03	Neural Network of Global Spatial Integration Network	53
04	Neural Network of Actor Network.	54
05	Neural Network of Critic Network	54
06	Gymnasium standard agent–environment [73]	57

Chapter IV: Experimental Results and Performance Evaluation		
Figure	Title	Page
01	LSTM Architecture.	67
02	Stage 1: PPO Training Reward. Smoothed 10-episode cumulative reward (blue) rises from ~75,000 to ~130,000 over 35 training episodes. Raw episode rewards (thin trace) show high variance, with a transient dip around episodes 10–14 followed by recovery.	70
03	Stage 1 — PPO Training Accuracy. Smoothed direction accuracy (green) starts at ~60% and rises to ~79% by episode 25, plateauing thereafter. The dashed line marks the 25% random baseline, which the policy exceeds by a large margin from the outset.	70

04	Static Test — All Methods Accuracy & Macro-F1. Solid and hatched bars show overall accuracy and Macro-F1 respectively. The dashed line marks the 25% random baseline. PPO+GA leads at 78.0%, followed by LSTM (76.6%) and PPO+PSO (74.4%). ARIMA's near-zero Macro-F1 (9.55%) despite random-level accuracy (23.6%) reveals systematic per-class failure.	72
05	Static Test — Per-Direction Accuracy. Each group of bars shows classification accuracy for one compass direction. ARIMA (purple) achieves ~100% on North but near 0% elsewhere, revealing degenerate constant-class prediction. PPO+GA (green) maintains the most balanced profile across all four directions.	73
06	Non-Stationary — Overall Accuracy (all 4 regimes combined). PPO+GA (46.8%) substantially outperform the frozen LSTM (31.2%). PPO+PSO (16.1%) falls below the random baseline (dashed line), indicating that PSO bias updates degrade rather than improve performance under abrupt regime shifts	74
07	Non-Stationary — Per-Segment Accuracy. Labels show the PPO+GA advantage over the frozen LSTM for each segment. The adaptation advantage grows from +1.6 pp in Segment 1 (partial distribution overlap) to +36.2 pp in Segment 0 and +18.4 pp in Segment 3.	75
08	Full Comparison Accuracy, Macro-F1, Latency, and Throughput across all evaluated methods.	76

09	SUMO side-by-side simulation windows. Left: Baseline — fixed-time signal control, no MEC overlays (T = 645 s). Right: MEC-Enabled — 20 RSU coverage circles (colored discs) and gNB boundary are rendered over the same Belgrade network (T = 322 s). Each circle represents the 250 m V2R communication range of one RSU.	79
10	MEC Adaptive Control vs Fixed-Time Baseline — full simulation comparison. Top row: Average Speed, Average Waiting Time, Active Vehicles time series. Bottom row: Halted Vehicles time series, Key Metrics bar chart, and Improvement Summary. The +33.3% throughput gain is the primary operational result.	81

Chapter III : Proposed Methodology		
Table	Title	Page
01	PPO training hyper parameters.	59
02	Metaheuristic optimizer hyper parameters.	61

Chapter IV: Experimental Results and Performance Evaluation		
Table	Title	Page
01	Full static evaluation — accuracy, Macro-F1, mean inference latency, and throughput for all methods.	71
02	SUMO simulation results — Fixed-Time baseline vs MEC Adaptive control (Belgrade network).	80

General Introduction

Mobility represents one of the core building blocks for 21st century smart city paradigms. The increasing urban population density makes anticipating and effectively responding in real time to the traffic conditions of road networks a fundamental necessity in terms of safety, efficiency of resource usage, and environmental sustainability. The two research threads that come together to address this problem are those concerned with Vehicle-to-Everything (V2X) communication (V2V, V2I, V2N, V2P), a mechanism to exchange contextual data in real time, and Multi-access Edge Computing (MEC), which brings computation and storage resources closer to end-users. In combination, these two research streams provide an unprecedented paradigm for enabling highly data intensive mobility predictions at the network edge.

Accurate mobility prediction is of utmost importance: in fact, the total annual cost attributed to global traffic congestion is reported to reach 750 billion dollar. Given the milliseconds latency requirement for safety critical V2X applications (e.g. Collision avoidance and platooning) central cloud processing is inappropriate. As vehicle connectivity increases, the models must maintain predictive accuracy under extreme data volume loads without an impractical increase in infrastructure costs. The formulation of an accurate, computationally efficient and adaptive mobility prediction scheme for MEC V2X systems thus appears critical.

However, current prediction techniques are largely inadequate for the requirements of V2X-MEC deployment: The computational light parametric methods (e.g. ARIMA, Kalman filters) do not even work under dynamic conditions, whereas deep learning models (e.g. LSTM, GNNs), are static offline models that work with IID assumptions: a concept drastically challenged by the inevitable distribution shifts resulting from accident or unexpected weather conditions, and which results in significant accuracy degradation. While Deep Reinforcement Learning approaches (e.g. DQN, PPO) show potential, they are more control oriented and do not account well for constrained computing resources. Thus a missing part of the literature is an online learning adaptive model suitable for V2X-MEC systems in terms of both latency and computational load constraints.

In this thesis, we present a novel framework that combines Deep Reinforcement Learning with an Optimization-based intelligent method for mobility prediction on V2X and MEC infrastructure. Three primary objectives guide our contributions:

- 1) Learning an adaptive mobility prediction policy through the use of a PPO-based agent, without reliance on stationary traffic models.
- 2) Developing a Genetic Algorithm (GA) based optimization routine to continuously refine mobility predictions.
- 3) Providing a formal design and formal mathematical treatment of the approach as a mathematically well defined constrained optimization problem by means of modeling a realistic network setup.

The contributions of this thesis are four-fold:

First, a novel two stage framework where RL agent learns the adaptive prediction policy and an metaheuristic algorithm improves the prediction during operation. Second, a mathematical formulation of the joint mobility prediction. Third, the integration of this hybrid scheme within architecture for a smart city network for ultra low latency edge-based operation. Finally, a rigorous evaluation of the prediction accuracy, computational latency, and robustness against dynamic traffic patterns with respect to state of the art classical ML, DL, and RL models.

The structure of the remainder of this thesis is as follows:

Chapter I reviews the prior art and describes the theoretical basics of V2X and MEC systems. It highlights the research gap that has motivated the present work. In Chapter II we present the system model and we mathematically formalize the multi-objective prediction problem. The main contribution of this thesis is presented in Chapter III: the hybrid learning approach, which describes the PPO formulation and the online adjustment component. The implementation and experimental results, including comparison against benchmark ML, DL, and RL models, on non stationary traffic conditions, are illustrated in Chapter IV. Finally, the General Conclusion is presented.

Chapter I
***Background and state of
the Art***

I.1 Introduction

In the last two decades, smart cities moved from an idealistic vision into a global policy concern. They are revolutionizing how cities are built, managed and inhabited. A smart city can be defined as a city that uses digital technology such as the IoT, AI, Cloud Computing, Big Data and Wireless communication networks in order to improve the provision of public services, the quality of life of their residents, and the sustainability of urban development over the long-term [1]. As opposed to traditional cities, driven by fixed infrastructure and responsive administration, smart cities are based on the constant loop of data collection, real-time processing, and intelligent response; which results into a reactive system adapting itself to citizen requirements.

The growing importance of smart cities is directly correlated to another global trend: continuous and rapid urbanization. The Union Nation (UN) forecasts that urban population will continue to grow. This change in population adds immense strain to public infrastructure, including road systems, public transit, energy supplies, water, and public services that were created with much lower numbers of people in mind. In this context smart cities offer a technological solution based upon the argument that city infrastructures are already in place and have reached saturation; thus, the efficient use of these existing infrastructure by implementing smart technologies which automate and improve manually based and inefficient processes will provide solutions for many of the stresses associated with urban growth [2]. Most prominent trend of smart city development now is smart mobility, which ranges from intelligent traffic management to zero emission transport systems, automated cars and car sharing schemes [3], but all of them share the same core belief: data makes for better decisions and better decisions make for more livable cities.

I.2 Urban traffic as a vital issue

The most complicated as well as the most significant one among the many facets of smart city management is urban traffic. Traffic congestion does not merely mean to put up with the commuting time for people; instead, it is a systemic disorder, bearing wide-ranging economic, environmental and social implications that pile up every single day with millions of people and thousands of road segments. It is estimated by World Bank that traffic congestion leads to as much as 3% Gross Domestic Product (GDP) loss annually by reducing productivity and increasing fuel consumption [4]. To give you an idea, according to the INRIX 2023 Global Traffic Scorecard, the average American spent 42 hours to combat traffic in the past year (which adds up to a whole working week), leading to a national loss of 70.4 billion US dollars. Among them, New York City solely experienced a total of 9.1 billion US dollars loss of time [5].

The environmental implications of traffic issues are also grave. Road traffic is the main culprit for urban green gas emission and particulate pollution. The World Health Organization (WHO) reveals that traffic caused 4.2 million premature deaths across the world every year [4]. Apart from air quality problems, it results in noise pollution, decreasing urban biodiversity, and accelerating the urban heat island effect. Lower-income areas near major roads or highways often bear the brunt of traffic issues, meaning that solving traffic

management is a matter of social justice besides that of urban efficiency. It has been found that enduring and repeated exposure to traffic congestion has been correlated with negative cognitive functions of commuters, thus exacerbating their levels of stress and decreasing the quality of life, which in turn influence worker productivity and health care system [1].

In essence, the problems contributing to the traffic congestion are structural and deep-rooted. Cities around the world were established on the idea that private cars should be the primary mode of transportation and this result in urban morphology that is unsustainable and inherently inefficient to handle the amount of movement required by urban societies, such as sprawl morphology, segregated land-use functions, and road networks focused on private cars [6]. With the increase in vehicle ownership rates which are closely linked with economic development in most Global South cities, the flaws in urban morphology become more acute. Simultaneously, conventional tools for traffic management, including fixed-time traffic signals, static route suggestions, and retroactive incident detection methods, do not possess enough flexibility and accuracy to cope with the enormous amount of variables that come into play within urban traffic flow. As a consequence, there appears a substantial disparity between traffic demand and the system's response [3].

I.3 Intelligent Transport System (ITS): A turning point of the history of road transportation management

The rise of Intelligent Transport System (ITS) marks a turning point in the long history of road transportation management. An ITS is generally defined as a system in which a diverse and interconnected group of technologies – such as sensing technologies, communication technologies, computing technologies, and control technologies are used in traffic infrastructure so that the system can automatically adapt and optimize itself. Basically, an ITS utilizes information from massive amounts of sensors such as: IoT sensors attached on the road surface or vehicles, smart phones equipped with GPS functions, traffic cameras located on roads, traffic monitoring equipment on roads and in weather stations, and ride hailing applications. This mass information then flows to either one centralized or multi-distributed data centers where algorithms including machine learning and AI, transform raw data into useful information and generate actions that can predict traffic jams, optimize traffic lights timing, dynamically reroute vehicles, coordinate different types of public transport fleets etc. [7].

By integrating machine learning in ITS, transportation planning in urban area is provided with more power on analytical and planning abilities. Previous methods relied on a stable statistical and rule-based modeling to predict traffic flow, and while this was suitable for a predictable traffic environment, it lacked the ability to effectively learn complex nonlinearities in the dynamic urban environment. Deep learning is effective in finding spatiotemporal patterns within highly heterogeneous and large-scale traffic sensor data to accurately predict future traffic flow. The application of gradient boosting algorithm in modeling the traffic data obtained from road sensors helped decrease congestion by 20% while a traffic control model with multi-agent reinforcement learning for coordination between Autonomous Vehicles (AV)

is said to improve the efficiency of traffic and traveling time by 25% and 30% respectively [8].

I.4 Mobility Prediction, the Foundation of Smart Transportation

The discipline at the heart of any modern intelligent transport system is the task of mobility prediction: using computational models to anticipate the future movements of people and vehicles throughout a city's network of roads. This task is not a simple technical problem, but rather, the foundational technology required to implement predictive, rather than responsive, traffic control systems. If a city can reliably predict where congestion will arise – where intersections will become gridlocked, roads will become saturated, transit lines will be overwhelmed – it can deploy resources, and alert commuters, to head off problems before they materialize by rerouting traffic, altering traffic signals, adding buses to a route, or informing travelers about better alternative journeys [9].

Mobility prediction has become dramatically more capable as data infrastructure and machine learning have evolved. While early traffic prediction systems were typically based on average travel times or simple time series methods that can only capture routine daily or weekly cycles, they are vulnerable to disruptive unexpected events-accidents, bad weather, special events or shifts in public travel preferences-that do not fit the expected patterns. Modern systems rely on an array of different data sources including real-time traffic sensor data, historical traffic volumes, weather information, social network data and event schedules, and are built using various advanced model architectures, such as Graph Neural Networks (GNNs), Long Short Term Memory (LSTM) networks, transformer based models and ensemble hybrid approaches that exploit multiple predictive strategies, to provide robust, fine-grained predictions [10]. GNNs are particularly adept at representing the complex interaction between locations within the road network and traffic behavior at individual points.

The benefits of mobility prediction can span every aspect of the urban transport system, from the strategic to the individual user level. Operationally, predictions can be used for dynamic traffic signal control, dynamic road pricing, incident management, and other systems for immediate traffic management. For long-term planning, prediction can aid in infrastructure development and optimization, transit scheduling, and policy decisions like congestion pricing or road space reallocation. For individuals, precise predictions power the Smartphone apps used to navigate roads and transit, coordinate car and taxi rides, and select the most appropriate way to travel from A to B [11]. Mobility prediction is not a mere research field; it is the essential underpinning infrastructure of the modern smart city, just as roads and streetlights are.

However, mobility prediction is far from solved. The accuracy of prediction systems deteriorates rapidly in unprecedented scenarios; such as accidents or pandemics, and we need to understand their reliability in the contexts in which such information is most vital [2]. The quality of input data is adversely affected by missing road network data, and incomplete road network sensor coverage, with privacy concerns limiting data availability. Furthermore, running accurate, city-wide traffic models in real time, particularly in a high-resolution

fashion, places significant demands on computing resources that not all cities may be able to sustain. Such challenges form the bedrock of this research thesis.

I.4.1) Deep Reinforcement Learning: A Paradigm Shift

Reinforcement Learning (RL) approaches traffic prediction and management as a sequence of decision making problems where an agent is trained to select an optimal policy by trial and error from an environment [15]. This approach differs from supervised learning methods which use a predefined set of labeled data; RL agents are trained by a reward signal associated with decision making, making them ideal for the non-stationary nature of traffic.

The intersection of deep neural networks and RL (Deep Reinforcement Learning, DRL) has yielded state-of-the-art results in a multitude of fields from game theory (Alpha Go defeating world champions [13]) to robotics (learning to pick-and-place [14]) and autonomous vehicles (navigating complex city environments [15]). Some of the strengths of applying DRL to traffic systems are:

- End-to-end learning (from sensory inputs, e.g. Densities, vehicle counts, queue length, to decisions, e.g. Signal timings, etc.) without manual feature engineering [19];
- Adaptive learning (online update policies based on observations and rewards, allowing for rapid adaptation to evolving traffic dynamics [20]); and
- Cost-sensitive learning (design appropriate reward functions to embed the traffic system's inherent objectives, e.g. Giving high priority to emergency vehicles, weighting upstream queues much higher than downstream queues, etc., which are difficult to encode within a loss function of supervised learning [21]).

Among a variety of DRL approaches, Proximal Policy Optimization (PPO) is often selected as the state-of-the-art algorithm due to its robustness to sample sizes and stability of updates [19]. PPO uses a clipped objective function to prevent too drastic updates which would prevent the agent's learning, making it appropriate for safety-critical applications where the cascading effects of poor decision making might be drastic.

I.4.2) The Hybrid Frontier: Combining Learning and Optimization

A critical issue for DRL agents is the slow adaptation to sudden shifts in data distribution. When traffic flows change drastically due to sudden road closures, for example, a policy trained from prior data might take thousands of training interactions (hours or days) to re-learn the appropriate behavior in the new traffic distribution, causing the traffic system's performance to degrade considerably during that period [23].

The research community has proposed combinations of learning based approaches with classical optimization algorithms as a means to leverage individual strengths and overcome respective weaknesses.

- Learning for Representation: Deep learning models have proven their ability in extracting meaningful features from rich data sets, along with modeling complex nonlinearities within a given traffic system.

- Optimization for Adaptation: Light metaheuristic algorithms (e.g. PSO, GA) are able to quickly update the decision boundaries of the system without requiring a costly gradient-based retraining procedure [24].

Such hybridized frameworks have recently been tested in analogous settings. Hybrid neural network forecast coupled with an online optimization component are found to be superior to standalone approaches in energy management when operating under distribution shift by 15-30% [25] and DRL models supplemented by an evolutionary strategy based adaptation show superiority over simple DRL agents in finance in terms of risk-adjusted returns under market regime shifts [26].

I.5 V2X Communication

I.5.1 Overview and Architecture

Vehicle-to-Everything (V2X) communications serve as one of the essential building blocks of modern ITS, in which vehicles are able to communicate with one another as well as the whole range of urban entities such as road side infrastructures, pedestrians, network servers and cloud networks, as illustrated in Figure I.1. In the ITS of modern society, vehicles are deployed with smart sensors, and interconnected into an Internet of Vehicles (IoV) system, so as to provide advanced vehicular applications based on multi-sensor data acquisition and processing for traffic efficiency and safety [26]. They can transform individual vehicles from stand-alone devices into smart nodes in the society with global awareness, and thus achieve inter-vehicles cooperation by sharing sensed information that can never be acquired from individual sensor alone.

The V2X communication employs two key wireless access technologies today. V2X communications are supported by two crucial radio access technologies: Dedicated Short Range Communication (DSRC) and the Cellular Vehicle-to-Everything (C-V2X), along with their evolutions IEEE 802.11bd and 5G NR V2X respectively [27]. Based on IEEE 802.11p, DSRC offers the possibility to transmit wirelessly between vehicles as well as vehicle and infrastructure over short distance of 300 m-1 km with a communication frequency at 5.9 GHz. However, C-V2X relies on large coverage and high capacity cellular networks so as to satisfy broad range communication and cloud connectivity requirements in smart transportation, which has been standardized by 3rd Generation Partnership Project (3GPP). C-V2X facilitates ubiquitous V2X communications, including V2V, V2I, V2N and V2P. Two types of communications exist from 3GPP Releases 14 to 16, which are Direct Communication using PC5 interface and Network Communication using Uu interface [28].

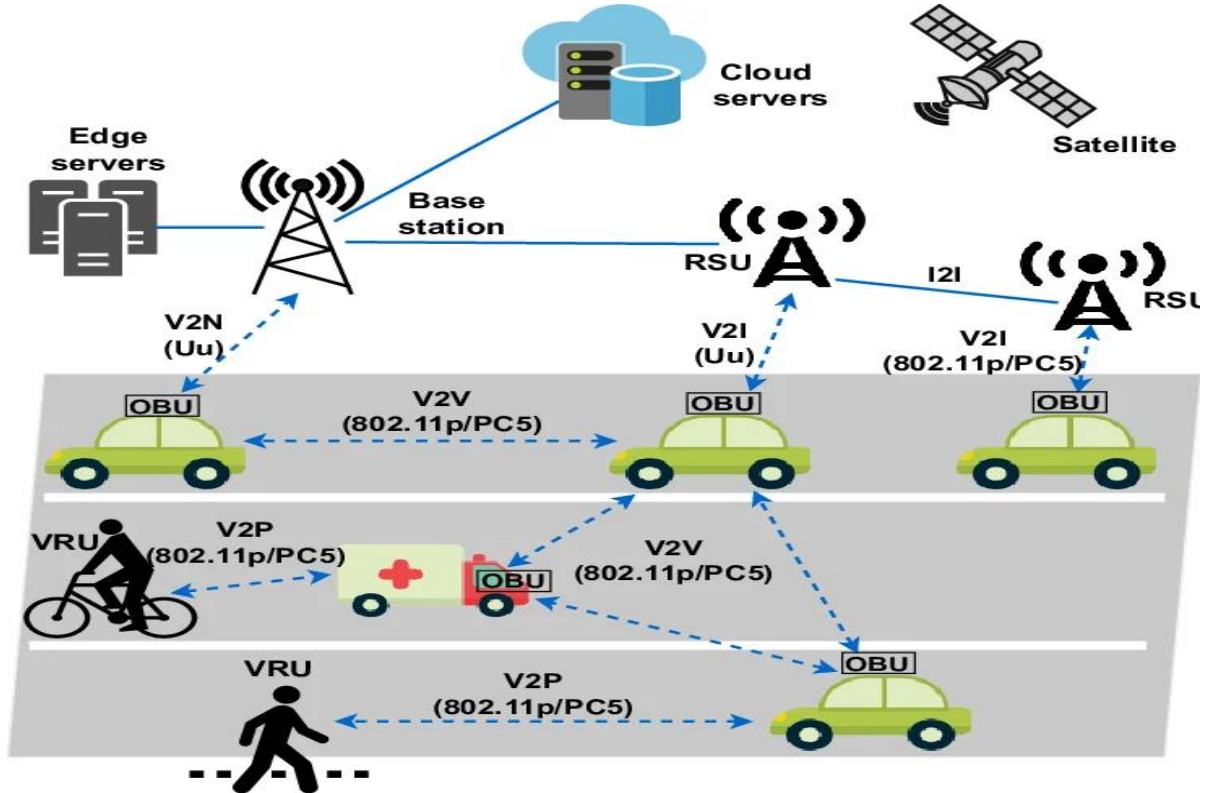


Figure I.1: V2X connectivity modes composing IoV paradigm [31].

I.5.2 Vehicle-to-Vehicle (V2V) Communication

V2V communication is the most basic form of V2X communication and allows direct communication between two or more vehicles, without using any intermediary infrastructure or network server. V2V links can be best utilized for safety-related purposes by vehicles broadcasting their speed, location, heading and brake status to all surrounding vehicles, with minimal latency. V2X offers benefits of reduction of road accidents by 80%, growing both the auto and telecom sectors for a smarter and safer ground traffic system, and V2V is the foundation for avoidance of collisions and cooperative maneuvers [29]. In practice V2V messages (e.g., Basic Safety Message (BSM) or Cooperative Awareness Message (CAM)) are broadcast at frequencies between 1 and 10 Hz. This provides a continuous image of the immediate surroundings that is sometimes beyond the capabilities of onboard sensors such as cameras and LiDAR, for example when it is dark or visibility is poor, or if the field of view of sensors is blocked [29].

I.5.3 V2I Communication

V2I is the V2X system between vehicles and infrastructure such as traffic light control devices, roadside units (RSUs), automatic toll system, variable message signs, roadside units (RSUs), environment sensors etc. The two way relationship established a completely new communication between vehicle and road, where road, formerly static infrastructure, now became an active part of transportation control. The V2I communications cover the car communication to road infrastructure via roadside units which work alone or relay nodes to broadcast safety and traffic information, such as green and red light state and Vehicle-to-

Home (V2H), Vehicle-to-Grid (V2G) (smart electricity distribution). Urban V2I applications, such as green wave passage that cars receive information of signal phase to adjust speed to go across intersection in green light, and dynamic speed advisory for reduction of acceleration and braking, can decrease fuel consumption and make traffic smoother [26].

The V2X system is based on V2I for traffic management, V2N to access Internet, V2V for collision avoidance, and V2P to provide security alarm to pedestrian and cyclists. C-V2X supports both the communication in terms of vehicle-to-vehicle, vehicle-to-infrastructure and vehicle-to-pedestrian, through a direct mode at PC5 Interface or based on network communication [32]. V2I is located at the crossroads of the traffic engineering and telecommunications industries, and requires not only robust wireless connectivity but also standardized communication formats, cybersecurity procedures and synchronized implementations at the municipality and highway level, the issues under study at the international bodies ETSI, IEEE and 3GPP.

I.5.4 Vehicle-to-Network (V2N) Communication

Vehicle-to-Network (V2N) communication links the vehicle with the larger cellular network, and via it, with various cloud based services, traffic management centers, map data providers, navigation applications, remote fleet managers, etc. Unlike V2V and V2I, which rely on direct wireless links operating at a shorter range, V2N utilizes the long range of 4G LTE and 5G cellular networks to enhance the vehicle's data coverage beyond its surrounding by far, thereby extending the vehicle's information horizon to a larger scope. With V2N, a vehicle is able to be fed with city-wide traffic information which aggregates data from numerous sensors and vehicles, receiving up-to-the-minute route guidance suggestions, real-time alert of traffic incidents from all across the city network, Over-The-Air (OTA) software updates, and proactive predictions on road congestion which is made through central intelligent systems using the data from many sources at a time.

V2N communication is particularly interesting as the type that links the vehicle with massive mobility prediction and optimization systems. For the purpose of infotainment services, traffic management, and large-scale cloud-based optimizations, a Cellular V2X standard has been proposed as part of 3GPP Release 14 where a Uu-based interface was established between the vehicle and the network [28]. With the upcoming 5G NR V2X technology, V2N communications capabilities are expected to improve greatly, where the end-to-end latency would decrease to less than 10 ms, and the data throughput will increase to support high-definition map streaming and large-scale sensor sharing [32].

I.5.5 Vehicle-to-Pedestrian (V2P) Communication

V2P communication addresses a key and the most vulnerable part of the urban traffic safety domain: the interaction between powered two-wheelers/four-wheelers and the road users that lack physical protection: pedestrian, cyclist and impaired mobility person. Via providing smart phones, smart wearable devices, and etc., the pedestrians could join the same information context as vehicles and infrastructures, in this way vehicle can predict and sense vulnerable road users before they enter the driver's eyes [32]. V2P communication is applied

to warn the pedestrians and cyclists and it is a necessary part of V2X which includes V2I for traffic management, V2N for internet access, V2V for collision avoidance.

The societal implication of V2P is quite significant: pedestrians and cyclist contributed to about 26% of total road traffic deaths around the world, and the percentage is growing rapidly with the increase of the densely-populated urban city. The information shared between Connected Autonomous Vehicles and pedestrians via V2P in cooperative perception of V2X enable vehicles to know about road conditions, road hazards and traffic signal status, expanding the perceiving range over on-board sensors [29]. However, mass deployment of V2P faces a lot of obstacles and difficult problem of pedestrian-linked side in accessible, low cost and privacy.

I.6 Multi-Access Edge Computing (MEC)

I.6.1 Definition and Core Architecture

Multi-Access Edge Computing (MEC) is a network architecture paradigm which pushes computation and data storage from remote centralized cloud data centers towards the edges of the communication network: specifically, onto servers at base stations, roadside units and other local access points near the mobile users, as depicted in the multi-tiered architecture of Figure I.2. This change in architecture is motivated by a core drawback of cloud based computing: transmitting data back to a remote server for computation involves a round trip latency that is unacceptable for real-time applications required in safety-critical situations such as in an autonomous driving car, real-time traffic light control, or collision avoidance. The aim of MEC is to achieve this reduction in latency by pushing computation capacity closer to users, providing a basis for the support of real-time applications and services; it is particularly promising for vehicular networks and smart cities providing opportunities for better wireless resource management while doing the computation closer to where data is generated [33]. MEC supports real-time responses, low latency and high bandwidth, access to real-time network information through services and application and are suitable for vehicular network environments where data comes from and changes quickly [34]. Instead of sending all data to distant clouds, cars and roadside sensors could offload computationally intensive computations, such as object recognition and traffic estimation inference, path planning etc., to a neighboring MEC servers to perform and then sending the result back to the car much faster. The characteristics of edge computing, including low latency, location awareness, wide-spread geographical distribution, mobility support, heterogeneity, and scalability makes edge computing suitable for the number of sensitive IoT services and applications including connected vehicles and smart city [35].

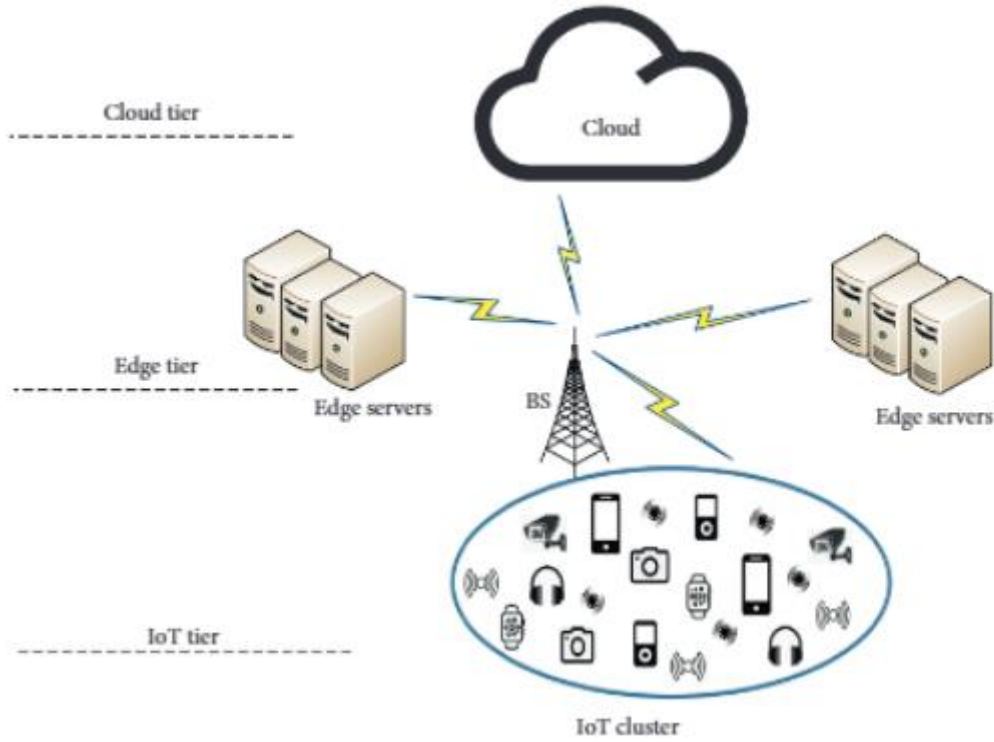


Figure I.2: Edge computing architecture [36].

I.6.2 MEC in vehicular networks: reducing latency

The application of MEC in vehicular networks, named Vehicular Edge Computing (VEC), becomes a particularly impactful development. Modern vehicles that communicate (connected cars or autonomous vehicles) generate a huge volume of sensor data (from cameras, LiDARS, Radars, GPS, V2X etc.) which is far too large to be transmitted to and computed in a remote cloud in order to meet the low latency requirements of real-time safety applications. VEC pushes computational capacity closer to vehicles thus lowering processing delays crucial to safety-critical application such as autonomous driving and smart traffic management where management of high and dynamic task load arising from vehicles data streams through enhanced task offloading and scheduling to minimize communication and computation latencies becomes an important challenge [37]. MEC stores and computes services/applications at the network edge closer to the mobile users to support services by using mobile backhaul and network core more efficiently and thus reduce latency. Combining 5G with MEC will achieve ultra reliable low latency data transmission enabling numerous latency sensitive automobile use cases and support the achievement of autonomous driving [38]. The interplay between high-speed 5G radio access network and edge computation capacity in MEC enables a whole network infrastructure that is capable to support end-to-end latency below 10 milliseconds, the difference between a responsive and truly real-time intelligent systems.

I.6.3 Task Offloading and Resource Allocation

The biggest operation challenge in deploying MEC applications is how to offload computational tasks either locally onto the vehicles, to nearby MEC servers or to the cloud

while taking account of current network states, server loading, task priorities and vehicle movement. These decisions will lead to a three-tier strategy (device-edge-cloud) to decide whether to process, offload or let the computation continue on the original place and the decision must be dynamically made to minimize latencies [39]. Effectively offloading tasks means deciding when and where to offload, while allocating resources properly to satisfy communication and computation constraints. Mobility of vehicles imposes further challenges, since a vehicle moving at a high speed can easily leave an MEC server coverage area while a task was assigned on that server, hence a vehicle is out of the area served by the MEC server. Traditional servers are challenged by scalability, latencies and utilization issue under a massive number of wireless devices requiring real-time processing, therefore the need of distributed and cooperative edge computing architectures emerged [40]. Many sophisticated task offloading mechanisms have been designed using Lyapunov optimization, deep reinforcement learning, and game-theoretic approaches to optimize quality of service and meet delay requirements for time critical tasks, especially under dynamic and urban vehicle environments.

I.6.4 MEC as an enabler for real-time mobility intelligence

The impact of MEC on the real-time intelligence at the city level required by smart transportation is not limited to reducing latencies for single vehicle applications. By means of placing MEC nodes around a city which will handle computation for local areas, the distributed MEC system will be able to achieve the same level of city-wide real-time traffic analysis as would be performed by centralized systems but on a higher speed. The adoption of AI/ML algorithms, especially deep learning and reinforcement learning approaches, helps the MEC to better adapt to changing environment, hence overcome the problem such as dynamic task offloading and resource allocation and real-time network function optimization [33]. The development of deep learning models will benefit significantly from moving AI towards the edge of the network as in MEC it enables computation much closer to the data sources thus minimizing the round-trip latency of cloud based deep learning inference. By this, predictions could be made quickly and react within each traffic signal phase time and provide computation resource that individual vehicles can't do efficiently or fast enough for time critical tasks, such as anomaly detection, object recognition and trajectory prediction [40]. This conjunction between MEC and deep learning mobility prediction makes MEC a powerful technology for intelligent mobility prediction research and provides a technical motivation for this thesis.

I.7 Mobility and Traffic Prediction

I.7.1 Definitions and Range of Applicability

The prediction of human and vehicular movement in a transportation network is often referred to as mobility prediction, i.e. predicting what will be the location of traffic, how concentrated it is, how quickly it is flowing, and how these will change within ranges of seconds or even hours. It is a wide and multidimensional problem and encompasses several sub-problems including short-term traffic flow forecast of traffic for each road segment/intersection, vehicle/pedestrian trajectory forecast, demand estimation of origin-destination pairs and network traffic congestion propagation. Increasing traffic congestion and

thus reduced travel times, environmental pollution has been triggered with accelerating urbanization which has led to mounting issues and in such cases deep learning approaches were developed to perform short-term traffic flow forecasting in support of intelligent transportation systems [41].

The primary role of mobility prediction in smart cities lies in the aspect that it helps a transportation network to be managed in a proactive way instead of a reactive one. A system that can only predict traffic congestion when it has occurred is of no use than predicting damage control. It can be predicted before it happens to prevent any kind of accident. Machine learning techniques combined with live traffic data is widely used to analyze the predicted congestion and journeys and to enable a decision making by authorities based on this data [42]. So the scope of application is wide, and includes traffic signal optimization, scheduling of public transport, and rapid response for emergency vehicles, flexible tolling, self-driven vehicle path control and a comprehensive multi-modal journey recommendation system.

I.7.2 Historical View of the Prediction Systems

Classical statistical models such as ARIMA and seasonal ARIMA were predominantly used for prediction of traffic in the initial years of traffic flow analysis which worked well for periodic behavior and stations time series. However they failed to predict non-stationary, interruptions and spatio-temporal characteristics of the traffic. With the increasing volume of traffic data such as loop detectors, GPS probes and connected vehicles, machine learning approaches such as Support Vector Regression, random forests were developed. However they consider individual road segments as independent.

The latest advancements in traffic prediction rely heavily on deep learning approaches which can effectively model complex spatiotemporal dependency and non-linearity as compared to statistical or classical machine learning approaches. Such approaches are generally categorized as GNNs, attention/memory-enhanced models, Generative Adversarial Network (GAN), transformers and Large Language Model (LLM), physics-guided models and various multi-information fusion based hybrid models [43]. LSTM and Gated Recurrent Unit (GRU) which form recurrent architectures have been very useful for modeling long-term dependencies in traffic time series while Graph Neural Networks are very effective in modeling spatio-temporal dependencies as they represent traffic network as a graph, nodes representing measurement sites and edges representing road connections. However deep learning models still need to perform much better at modeling long term temporal dependencies and heterogeneous dynamic spatio-temporal dependencies. Thus research has shifted towards the transformer-based heterogeneous spatio-temporal GNNs [44].

I.7.3 Challenges in dynamic urban environment

Even though significant advancement has been made in traffic prediction in the recent decade, yet urban mobility prediction remains a complex task because the traffic condition in urban environment is a non-stationary dynamic function influenced by various factors of natural occurring phenomena as, accidents, road constructions, traffic congestions, sporting events and variations of climate. This unpredictable behavior is not something that historical data can predict.

- **Data Quality and Completion:** One of the most prevalent and difficult issues for mobility prediction are the quality and completion of input data. Data quality assurance is reliant on maintaining data integrity and effective data cleaning protocols, whereas large volumes of data must be processed with scalability as the city grows and the sensor network expands. Instantaneous processing necessitates high-performance computing infrastructure and highly efficient algorithms, while storage and retrieval of enormous amounts of data become a critical bottleneck [43]. Failures in hardware and communications, as well as calibration errors in urban sensor networks result in lost information and erroneous records, which negatively impact prediction performance if not effectively addressed through data imputation and outlier detection [45].
- **Spatio-temporal Complexity:** Spatio-temporal dependencies in urban traffic are extremely complex. Congestion at a specific intersection leads to a ripple effect across the entire network which is heavily dependent on traffic signal timings, turning proportions and the routing behavior of numerous road users. Traffic flow prediction deep learning models achieve state-of-the-art performance in traffic participant interactions, using learned representations and hierarchical fusion to improve performance in dynamic settings, and generate multiple predictions in order to model various scenarios [46].
- **Heterogeneity in Data:** One major challenge is the presence of a variety of data sources with varied spatial and temporal characteristics such as loop detectors, GPS, connected cars, weather reports, social media messages and event schedules. Multimodal data processing becomes computationally intensive and acts as bottleneck for utilizing it fully [47]. Effective data fusion techniques along with hybrid deep learning models can fuse diverse sources of information.
- **Privacy and Data Access:** Human movement trajectories derived from individual GPS tracking reveal a large amount of sensitive information such as location of residence and work, medical information, religious preferences and social relationships. It is very hard to gain direct access to individual-level digital data, especially for a very large dataset, therefore making prediction of human mobility flows rather important [48].
- **Rare Events and Distribution Shift:** Trained traffic prediction models are very effective at modeling routine traffic scenarios but fail miserably when unexpected events such as weather extremities, accidents, civil emergencies, and significant sport events occur - precisely the critical situations requiring prediction. Some of the critical tasks are heterogeneity in data, computational performance, accuracy in dynamic conditions and handling data that is not encountered in model training.

I.8 Deep Reinforcement Learning

I.8.1 Reinforcement Learning Basics

Reinforcement Learning (RL) is a field of machine learning that is concerned with how an autonomous agent can learn to make sequential decisions in an environment in a way that maximizes cumulative return. An RL agent learns by interacting with the environment,

receiving a reward signal for actions, and refining its behavior to receive the greatest return in the long run [49]. Instead of using a data set of correct action-response mappings to learn as in supervised learning, RL does not know what the best behavior is and can only learn from its own experience in the environment guided solely by a scalar reward signal it receives from the environment each time it takes an action [49]. The trial-and-error method that characterizes RL makes it very well-suited to the kind of control and decision-making problems where the ideal strategy is unknown, and thus an agent must learn this optimal strategy through exploration of the problem environment, which is conceptualized in Figure I.3.

The core theoretical framework that defines RL is the Markov Decision Process (MDP) that is used to model sequential decision-making processes under uncertainty. An MDP is formally characterized by the tuple (S, A, P, R) , where each symbol serves a precise purpose in the model. The most general model of an MDP is known as reinforcement learning, a model that seeks to find an agent's policy to perform a sequence of action in an unknown environment and maximize cumulative rewards. A Markov decision process formalizes the interaction of an agent with its environment by stating a problem's states, actions, rewards, and transitions [50].

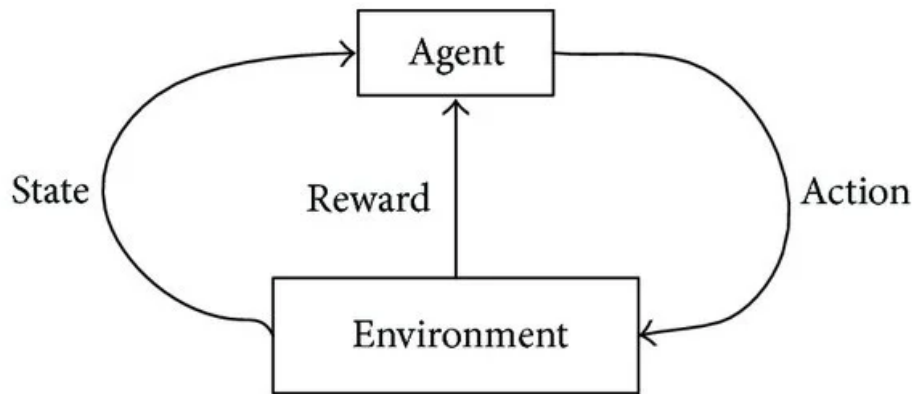


Figure I.3: Framework of reinforcement learning

- **States (S):** States S describe each state in which the environment can be at a particular time step. The information conveyed by states about the environment is completely captured, such as for a traffic management environment it would be the queues length of each approach, the present signal phases, the average speeds of the cars and time of day. States are also described by the Markov Property, stating that state fully determines the subsequent state transition regardless of its history, as future outcomes depend only on the present state and chosen actions. Formally, an MDP defines an agent interacting with an environment where it takes an action after observing states in discrete-time finite MDP [52].
- **Actions (A):** The action space A is the set of commands available to an agent for controlling its behavior at each time step. They can be discrete actions such as selecting the phase of a traffic light or they could be continuous such as selecting the green signal duration in seconds. The policy of an agent, the behavior to be learned by

the agent, is given by function from states to actions (or distributions over actions) while the agent has also a state-action-value function that is updated to predict how much future reward will result from taking a specific action in a specific state [53]. Policy is a measure of performance defined as the expected cumulative discounted return achievable from a given state, to maximize it is the objective of the learning algorithm.

- **Reward (R):** Reward signals are the key way to specify what the objective is to an RL agent; after every action, the environment provides a reward signal, typically a scalar value, to give an indication of the desirability of the action taken. In a traffic system, a reward could be defined as the negative waiting time, number of cars that exit the intersection per hour, or a combination of these parameters to minimize vehicle delay and reduce fuel consumption [52].

A discount factor in $[0,1]$ will also assign greater weight to current actions compared to far futures; values near to 1 indicate that agent will consider far future reward, and value near to 0 indicates that agent will tend to receive the maximum rewards possible at the current step. Reward signals determine the nature of a Markov decision process, and reward-centric concepts can provide insight into reinforcement learning topics such as sample complexity and exploration efficiency [52]. A key goal of reinforcement learning is to find an optimal policy that maximizes the expectation of the cumulative discounted reward (the return) starting from a particular state. Reinforcement learning algorithms may either focus on approximating a value function $V(s)$ or $Q(s, a)$, or they may focus on directly learning the policy $\pi(s)$ itself. Between exploring the unknown actions in the environment to find out which the best actions (exploitation) are, and choosing the best known actions (exploitation) lies the exploration-exploitation dilemma that all practical RL agents have to cope with [49].

I.8.2 Deep Reinforcement Learning

The methods that have been successfully applied to control tasks are based on tabulated representations of the state-action value function or the agent's policy. Unfortunately, this tabular representation is only suitable for problems where the number of states is relatively small; otherwise computation time would be very long. Therefore, most of real-life decision making problems (such as control of autonomous driving systems, robotic systems, traffic lights etc.) need to be able to handle a large number of state features. Deep Reinforcement Learning (DRL) offers a scalable approach, conceptually depicted in Figure I.4, that addresses this limitation by integrating deep neural networks as function approximations to approximate these value functions or policies. It uses deep neural networks which are powerful enough to generalize across large numbers of states [53].

In a DRL system, deep neural networks are trained as function approximations where they map states to actions (in case of a policy network) or states to values (in case of a value network), and then back propagated for further learning. By taking as inputs not processed states (image pixels, raw sensors information etc.) the deep neural network is able to capture complex relationships and features in the data to perform the task. Despite its impressive recent successes, the generalization of DRL to real world tasks which feature a number of

complicated challenges like unreliable and high dimensional states, real time control needs and variable environment dynamics is still an active area of research [53].

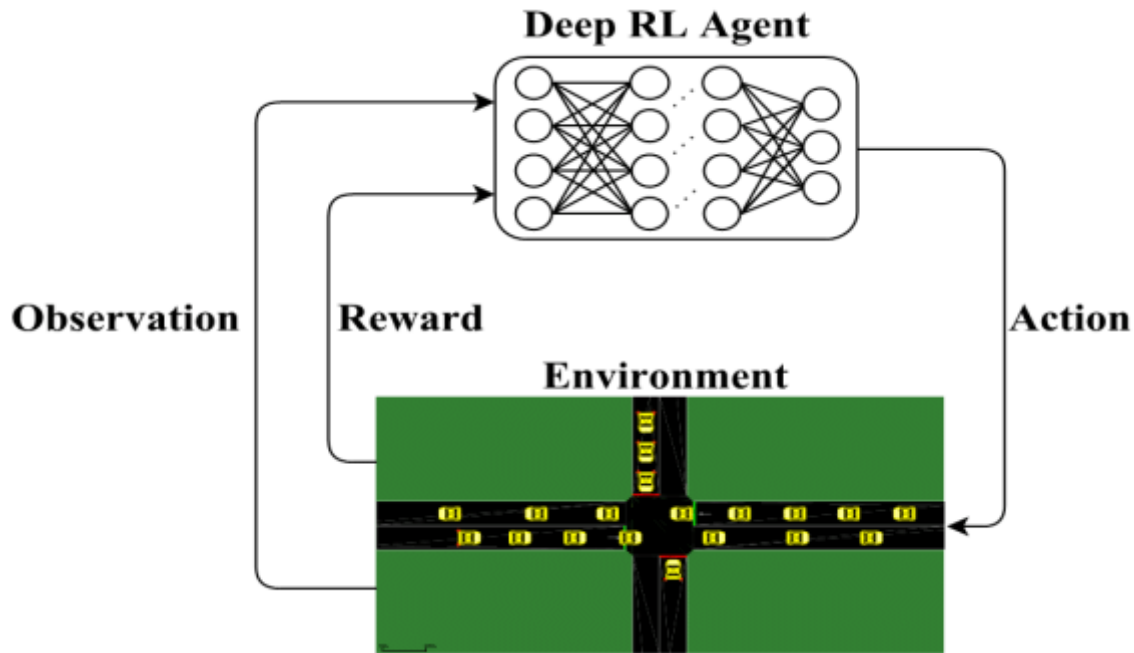


Figure I.4: Deep reinforcement learning agent of traffic [54].

I.8.2.1 Deep Q-Network (DQN)

The first breakthrough in deep reinforcement learning which has led to DRL becoming a real and promising approach was the creation of the Deep Q-Network (DQN) algorithm, in 2015 by Mnih et al. DQN represented for the first time that it was possible to master complex tasks from low dimensional inputs and only from the overall game score reward signals by playing atari video games at a super-human level. To achieve this level of play, DQN utilized two crucial mechanisms in training: experience replay which stores transitions into memory and sampling them randomly for training updates, to break the temporal correlations between data samples, and the target network which serves as a more stable target in the Q-learning update process [55]. Both techniques were the keys that proved that Q-learning can be stable even when non-linear function approximations, like neural networks, are involved, which previously had not been proven possible [56].

Since its introduction, DRL has rapidly developed with numerous families of algorithms evolving, for example value-based learning algorithms (like Double DQN, Dueling DQN or prioritized experience replay) or policy gradient methods that optimize the policy directly with respect to expected return; furthermore, actor-critic methods that combined policy-gradient and value-based approaches were developed, to allow adaptation of the policy based

The DRL field diversified and branched off into several classes of algorithms immediately following the introduction of DQN. Value based approaches were developed such as Double DQN, Dueling DQN and Prioritized Experience Replay; to over-estimate the Q-value, isolate state-value and advantage estimation, and learn on the most valuable transitions respectively. Policy Gradient algorithms learn by directly estimating the gradient of the expected return,

they don't need to calculate an explicit value function, and they are well suited to environments with continuous action spaces. Actor-Critic approaches combine these two concepts where an Actor Network learns a parameterized policy, and a Critic Network estimates a parameterized value function, to provide low-variance gradients that accelerate learning. Algorithms that attempt to learn a parameterized value function (Critic) and a parameterized policy (Actor) are called actor-critic algorithms, where a classic example is Asynchronous Advantage Actor-Critic (A3C) [57]. There are further advanced areas, such as Model-based DRL and Multi-agent DRL, where models are learned in the environment to plan actions, and multiple agents attempt to learn concurrently in an environment respectively, which could be very applicable to coordinating vehicles and traffic lights in Smart Cities.

I.8.2.2 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a policy gradient method, operating within the architecture as depicted in Figure I.5, proposed by Schulman et al. In 2017, this has become one of the most popular and practically usable algorithms in the field of DRL. The motivation of PPO comes from a fundamental stability problem in vanilla policy gradient: using standard gradient ascent, the large update steps in policy parameters might move the new policy far away from the old policy resulting in a performance drop to an unprecedented level and making recovery impossible. PPO offers a principled way of limiting the update size such that each step of optimization improves the policy performance without introducing instabilities [57].

PPO is designed to be a simplified version of TRPO (Trust Region Policy Optimization), which solves the problem of update instability by requiring a hard constraint on the KL-divergence between new and old policy at each update step. While TRPO is theoretically well-founded, it requires computing second order derivatives and solving a constrained optimization problem at every update, which is computationally very expensive and can be hard to be scaled up to large networks and complicated tasks. PPO significantly reduces this complexity by adopting a clipping mechanism on the probability likelihood ratio to avoid imposing the hard constraint completely, allowing it to use a first-order optimizer like gradient descent to optimize the objective while being simpler to implement and tune [58].

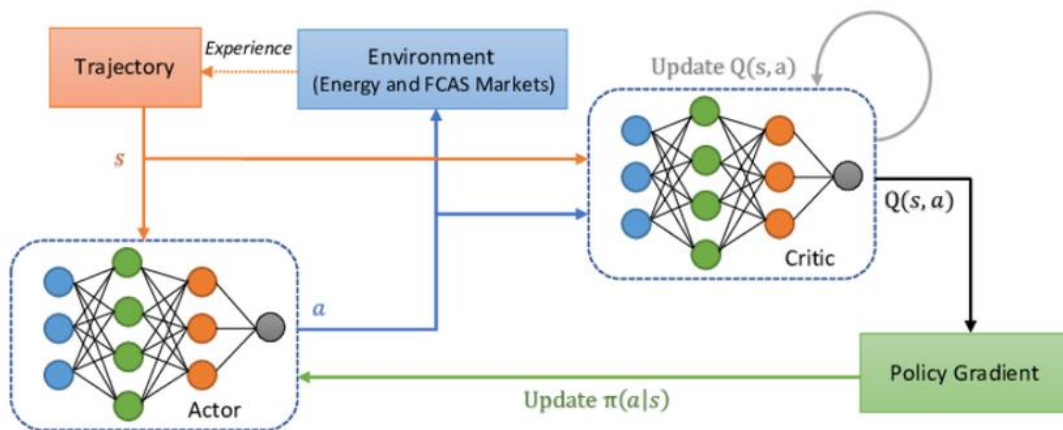


Figure I.5: Proximal Policy Optimization (PPO) architecture [59]

The central mechanism of PPO is the clipped surrogate objective function. At each training step, PPO computes the ratio $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$, which measures how much the new policy π_θ differs from the old policy $\pi_{\theta_{old}}$ in its probability of selecting each action. The clipped objective clips this ratio to the range $[1-\epsilon, 1+\epsilon]$ where ϵ is a hyper parameter typically set to 0.2 and takes the minimum of the clipped and unclipped objectives. By clipping the ratio, it is ensured that the policy does not deviate too much from the older one during each update step [19]. This clipping serves a dual purpose: it prevents destructive large updates while also providing an incentive for the policy to stay close to its previous version, as advantages arising from out-of-range policy deviations are discarded and contribute zero gradients. PPO has two variants. These are PPO-Penalty, which includes KL-divergence as a penalty term in the objective and adjusts the coefficient automatically throughout the training, and PPO-Clip, which just use the clipping mechanism without any KL-divergence term. It's PPO-Clip that most popularly used currently [60].

PPO belongs to the actor-critic family of algorithms. The actor network outputs the probability distribution over actions given the current state, while the critic network estimates the state value function $V(s)$, which is used to compute the advantage estimate $A_t = R_t - V(s, t)$ a measure of how much better the chosen action was than the average action in that state. This advantage estimate guides the policy update: actions with positive advantage are made more likely, and actions with negative advantage are made less likely, with the magnitude of the update controlled by the clipping mechanism. The advantage is computed using Generalized Advantage Estimation (GAE), which balances the bias-variance trade-off in advantage estimation through an exponential weighting of multi-step returns [61].

PPO holds several practical advantages that have made it the algorithm of choice across a wide range of DRL applications. On-policy- it learns from data generated by the current policy, guaranteeing that training values match policy that we are trying to optimize. The algorithm can learn in both discrete and continuous action spaces, which allows it to be applied in a wide range of traffic management problems, from discrete phase choices to continuous green-time decisions. It is comparatively stable and sample-efficient, requiring less hyperparameter tuning than many competing algorithms, with an ϵ value of 0.2 performing well across most cases [9]. Furthermore, PPO can be parallelized across multiple environment instances, dramatically accelerating data collection and training speed in large-scale simulations.

In the domain of smart city traffic management, PPO has demonstrated strong empirical performance across multiple application contexts. A study integrating STGCN-LSTM traffic flow prediction with PPO-based signal control showed that the PPO algorithm dynamically adjusts signal timings, reducing vehicle waiting times by 30% and increasing traffic throughput by 15%, with robustness maintained under adverse weather and traffic surge conditions [62]. In multi-intersection settings, a federated PPO framework was proposed for distributed joint training of agents across domains, addressing the problems of slow learning speed and poor model generalization when deep reinforcement learning is applied to cross-domain multi-intersection traffic signal optimization control [63]. A multi-agent PPO formulation for adaptive traffic signal control demonstrated good stability, robustness, and

adaptability to changes in traffic demand across both nominal and perturbed volume conditions [64]. These results collectively establish PPO as a highly suitable algorithm for the traffic signal control and mobility optimization problems addressed in this thesis.

I.9 State of the Art

I.9.1 Classical Approaches

Traditional traffic prediction systems traditionally employed time series forecasting techniques where traffic data was viewed as a phenomenon evolving with time. These traditional methods assume that information regarding the future traffic states can be found through extrapolation of patterns in previous data and thus require analyzing data for its trends, periodicity and seasonality characteristics. Functionally these classical techniques have significant advantages that include high interpretability, easy mathematical interpretation and high computational speed. This makes them extremely suitable for state prediction and short-term traffic prediction in stable and predictable system environments. The fundamental drawback with traditional approaches however is that they are quite restrictive and highly parametric; and rely heavily on certain mathematical assumptions especially related to stationarity and linearity. Under the present modern dynamic traffic flow condition in urban environments these phenomena are indeed non-stationary and highly stochastic in nature where factors such as accidents, adverse weather conditions, public events etc could cause major deviations; with their weak representative capabilities, these traditional parametric models can't take into account.

I.9.1.1) ARIMA-based Methods: Auto Regressive Integrated Moving Average (ARIMA) models have been extensively applied to short-term traffic flow prediction [65]. The autoregressive (AR), integrated (I) (by means of differencing), and moving average (MA) error models are mathematically combined within ARIMA, aiming to identify the temporal characteristics and extrapolate these trends from historical sensor data. Beyond this basic idea, Williams and Hoel (2003) showed that season ARIMA (SARIMA) model was capable of identifying weekly periodicity and daily periodicity inherent within urban traffic and was able to obtain decent prediction performance for the short prediction horizon of 15 minutes [62]. Nevertheless, the underlying structure of these ARIMA based methods is very restrictive due to strong prerequisite, which is the stationary nature of traffic flow data (statistical characteristic of the data remain constant through time). In an increasingly dynamic V2X environment, traffic flow itself is generally non-stationary and highly stochastic. Thus, they are incompetent to forecast and catch up the drastic nonlinear regime change occurring due to external shock, for instance, traffic accident, weather disaster and irregular activity etc. They will exhibit significant lag in response to those changes as the new regime might have already established before the model is able to adjust prediction results. Besides, due to the univariate property of ARIMA based methods and rigid mathematical structure, they cannot depict any inter-traffic state dependent spatial dependencies that arise at intersections which are strongly coupled each other in smart city network and therefore have no application for preventative and systemic traffic management.

I.9.1.2) Kalman Filtering: To mitigate the offline processing requirements and delayed responsiveness of pure time-series models, Kalman filters and their advanced variants have been widely adopted as the standard for real-time traffic state estimation. Operating on the principles of recursive Bayesian estimation, the Kalman filter continuously updates its predictions by reconciling incoming, often noisy, sensor measurements with its internal mathematical model of the system dynamics. [64] pioneered the application of Kalman filtering to predict traffic volumes, demonstrating superior real-time performance and faster convergence over naive historical methods, particularly in highway scenarios. As intelligent transportation evolved into the V2X era, the Extended Kalman Filter (EKF) became prevalent for vehicle trajectory and motion prediction, attempting to accommodate non-linear vehicle dynamics by linearizing the state transitions using Taylor series expansions [68]. While highly prized for their recursive computational efficiency and extremely low latency making them theoretically ideal for edge computing deployments these methods suffer from severe theoretical limitations. They fundamentally rely on linear state-space assumptions and implicitly presume that both system and measurement noises follow a Gaussian distribution. In a real-world smart city environment, vehicular movement is highly non-linear, and the noise generated by unpredictable driver behavior or sudden physical disruptions is rarely Gaussian. Furthermore, these filters require the manual, highly careful tuning of process and observation noise covariance matrices. In a constantly evolving V2X network where the underlying traffic dynamics shift abruptly and without warning, maintaining these finely tuned static parameters becomes practically impossible, causing the predictive accuracy of the Kalman filter to degrade significantly and rapidly under non-stationary conditions.

Although classic methods still receive high credit for its very low hardware needs, high speed of operation, and clear mathematical understanding, their fundamental architectural assumptions makes them fundamentally insufficient for future generations of intelligent transport systems. The primary limitation of these methods stems from the hard mathematical assumption of stationary or slowly varying traffic distributions. In contrast, high fluctuation and rapid, uncertain, shifting underlying traffic statistical characteristics exist in modern V2X scenarios caused by exogenous factors. Moreover, these low-parameter models severely under-represent ability to capture coupled non-linear spatiotemporal dependencies through space propagating across several connected RSUs in a dense urban city grid, as they either model individual intersections separately, or with far too simplistic linear relations, failing to take network-wide ripple effect of congestion. However, more importantly, they cannot adapt on-line. In the event of regime change in traffic - say, catastrophic bottleneck, abrupt change in weather or detour route of emergency vehicle, etc - classic models cannot organically adjust internal parameters in real-time in accordance with the changing circumstances but only through tedious manual recalibration or complicated off-line parameter tuning with tremendous delay of time, violating all the ultra low latency and high autonomy requirements in the environment of MEC networks, and suggesting high-parameter AI models instead.

I.9.2 Machine Learning and Deep Learning Approaches

The dramatic increase in advanced sensing infrastructure, including pervasive roadside units, inductive loop detectors, and fine-grained vehicle telematics, has fueled an emergent

data-driven methodology that learns extremely complex, nonlinear traffic patterns directly from massive historical datasets. Moving beyond the limited parametric structure of classical statistics models, intelligent transport research was initially driven by relatively simple machine learning models (e.g., Support Vector Regression and Random Forests) in order to model the stochastic behavior of traffic more flexibly. Eventually, with the increasing dimension of urban traffic data, research has moved toward the application of deep learning architectures that have enough representational capacity for automatic high-level feature learning. Recurrent neural networks, in particular, Long Short-Term Memory (LSTM) networks quickly became the standard baseline for sequential traffic prediction due to their effective internal gating mechanisms to model long-term temporal dependencies without suffering from the vanishing gradient problem. Expanding on temporal models, later research noted that urban traffic is not just a spatial phenomenon but rather a highly integrated spatial-temporal phenomenon. This has been the key idea for the popularization of Graph Neural Networks (GNNs) and attention-based architectures (e.g., STGCN and Transformer models) that learn how to incorporate spatial meshing passing into temporal sequence learning by modeling road networks as topological graphs where intersections are nodes and roads are edges, dynamic consideration of distant intersections, achieve unparalleled performance with non-stationary traffic patterns under static conditions. Despite significant representation advantages, a fatal limitation of present deep learning paradigms is the reliance on independent and identically distributed (i.i.d) training data, so these networks behave as completely offline models that cannot handle changes; catastrophic forgetting and prediction failure typically occur once underlying traffic patterns are perturbed due to unexpected physical or environmental events. As the tremendous amount of labeled data and huge amount of computation time (e.g., hours or days with dedicated hardware clusters) are required to train these deep architectures, it is infeasible to use them in high dynamic edge computing environments that necessitate rapid, real-time learning and update, resulting in the unavoidable deployment lag.

I.9.2.1) Recurrent Neural Networks: As the drawbacks of classical time-series models were exposed, Long Short-Term Memory networks, based on their strong representational ability, quickly became the state-of-the-art baseline for sequential traffic prediction. The LSTM networks overcome the drawbacks of the conventional RNNs where vanishing gradients are unavoidable when long sequence data are being processed. This advantage is gained by the complex network of memory cells and the gating structure (input, output and forget gates), which help the LSTM models to selectively maintain relevant long-term temporal information, discarding the historical irrelevant noises. To show the advantage of this architecture, LSTM models outperform the traditional ARIMA models for long-term temporal dependence capturing using remote microwave sensor data for traffic speed prediction [66]. Furthermore, the successful application of the LSTM-based architectures has been further proved by [64]. On network-wide traffic prediction using complex, real world datasets [67]. However, the lethal shortcomings of recurrent architectures stem from the stringent requirements of the supervised learning framework: massive, precisely labeled training datasets drawn from the specific target data distribution are needed, and it is also assumed that the future will have same statistical properties as the past training data. This implies catastrophic forgetting when the models are used in a real world scenario. As the traffic

behavior varies drastically in V2X scenario (may be even hourly due to changing weather conditions, accidents or users' daily routines), once a LSTM model is trained for historical baseline, its performance is completely useless under the changing data distribution and hence ineffective for traffic management and autonomous vehicles.

I.9.2.2) Graph Neural Networks and Attention Mechanisms: Acknowledging that traffic across an urban space is not simply a set of isolated time series but rather a severely connected spatial phenomenon, contemporary literature has iteratively introduced graph structures to spatially-explicitly capture the spillover effect of traffic across road networks. Breaking from the univariance constraint of elementary LSTMs, GNNs treat road grids as explicit topological graphs, where road junctions are viewed as vertices and roads as edges. In the inaugural fusion of spatial and temporal modeling at the intersection, [68] propose diffusion convolution recurrent neural networks (DCRNN) to model traffic spread in much the same way that a fluid can diffuse across a real road network through the use of graph convolutions and sequence modeling [68]. To refine spatial feature extraction further, [69] propose attention-based spatial-temporal graph convolution networks (ASTGCN) to allow for varying time step weights and the weight of distant spatial nodes, so that a miles-away bottleneck can influence traffic at a local junction [69]. Though such powerful architectures can remarkably capture spatial-temporal correlations when the environment remains stable, all these models exhibit a fundamental flaw in terms of practical deployment: they are not able to learn on the fly. They are essentially static, offline models where weights are frozen after initial training; therefore they cannot adapt when the distribution of traffic suddenly changes, for example in the case of lane closures or emergency re-routing. They become mathematically irrelevant in that they cannot adjust and the costly re-training on a GPU cluster must begin. This incapability of online adaptation renders them impractical for latency sensitive V2X edge application where the system must recover from sudden distribution changes locally and immediately.

In conclusion, although today's deep learning techniques are extremely accurate on controlled and invariant domains; their core fragility originates from their intrinsic and strong reliance on the *id. Assumption*. The sheer number of parameters of those complex architectures inherently assumes that the future operational conditions will behave exactly as the data used for training the model. This makes those models extremely brittle to distribution shifts, which is a common scenario in constantly evolving urban networks whose traffic patterns are influenced by sudden accidents, extreme weather, or spontaneous traffic changes caused by commuters' behavior. Against those non-stationary and out-of-distribution events, the predictive accuracy of deep learning models often fails dramatically through catastrophic forgetting, and very time-consuming and cost-prohibitive retraining are required to fix them. Retraining those spatial-temporal deep learning architectures takes hours and requires powerful GPU clusters and millions of computations to optimize millions of parameters over days, and models do not update their parameters in execution. Therefore, pure deep learning solutions act as rigid and static offline inference machines and are completely unable to achieve real-time self-adaptation in the edge domain. This is extremely challenging in current V2X systems which require models to constantly adapt to changing traffic patterns in real-time on the network

edge, and necessitate many costly offline retraining cycles with long delay to re-deploy the models on the vehicles and network infrastructure after each one.

I.9.3 Reinforcement Learning Approaches

RL is a paradigm shift from pattern recognition to active control/learning from interactions in intelligent transportation, where the real-world operation is modeled as a Markov Decision Process (MDP). In the past, the problem of control at traffic domain is mainly tackled by value-based methods, like Deep Q-Networks (DQN) . DQN is highly capable to achieve state-of-the-art performance in optimizing adaptive traffic signals. However, its weakness in continuous and high-dimensional space have been overcome by actor-critic methods, in which Proximal Policy Optimization (PPO) achieves impressive stability and sample efficiency on complex traffic domains. Most of the prior works have focused on applications like vehicle routing, traffic signal actuation and guidance. In recent research, a important paradigm shift has emerged and RL is being utilized for complex prediction tasks. Unlike conventional supervised deep learning, where history is directly mapped to the future by assumed independent and identically distributed (i.i.d.) assumption, RL for prediction explicitly models interaction between different agents, enabling the agent to understand dynamic traffic patterns from long term rewards and future predictions in response to the system feedback. The agent has the capability to forecast future states based on sophisticated predictive models which can adapt to unknown dynamic conditions.

I.9.3.1) Trajectory Prediction: Compared to other learning paradigms, reinforcement learning (RL) offers a conceptually different framework for trajectory prediction and motion forecasting in autonomous driving, which shifts the modeling task from a pattern recognition task in static historical data to a framework of sequential decision-making and interaction-based policy learning. [70] categorize existing RL-based prediction methods within the general scope of trajectory prediction and stated that Deep Reinforcement Learning (DRL) has been successfully applied in learning motion planning processes of intelligent agents, methods can be broadly divided into value-based methods (e.g., DQN) and policy-based methods (e.g., PPO, SAC). Among all DRL based prediction methods, an important branch utilizes Inverse Reinforcement Learning (IRL) to interpret human behaviors from the aspects of computational cognition and Theory of Mind (ToM). In this framework, hierarchical IRL models have been proposed, which learns the distributions of future trajectories from human demonstrations to implicitly encode multi-agent interactions by inferring the latent motivations of human behaviors rather than learning motion patterns from data. Building on this, subsequent studies have proposed IRL models capable of inferring human latent intelligence level and demonstrating humans reasoning mutually in game-theoretic traffic scenarios or one's behavior being influenced by other's continuous control in car-following situations. This interpretation of intent, which is absent in supervised learning methods, provides RL-based approaches with a unique ability of modeling strategic interactions among multiple agents beyond the level of direct pattern fitting.

I.9.3.2) Hierarchical Hybrid DL-RL Framework: Aware that pure DL methods are inherently unable to capture transient, micro-scale interactions because they model the trajectory prediction process as a supervised learning sampling task based on intermediate-

scale encoded states, [71] propose a hierarchical hybrid framework which explicitly decomposes trajectory prediction into two complementary stages. In the first, DL-based stage, the interactions at global and intermediate spatial scales are encoded by a heterogeneous graph Transformer. A set of agent-centric heterogeneous sub-graphs decomposing the traffic scene simultaneously captures lane elements and neighborhood agents; furthermore, their direction heterogeneity is distinguished by four relative heading directions (front, left, right and back). Graph neural networks, based on the Transformer paradigm, aggregate these correlated features to generate multi-modal key future locations for all agents. A global interaction module further allows for cross-scene message passing. In the second, RL-based stage, these predicted key locations provide context to split the global scene into micro-scale local sub-scenes and the Transformer-based PPO agent learns the optimal behavioral strategy under influence of instantaneous agent-to-agent and agent-to-lane interactions within each local scene. The main novelty of the architecture is that in contrast to vanilla PPO, which uses standard neural network layers, the PPO network is replaced by a Transformer module which is performing cross-attention between ego vehicle's motion state embedding and sub-scene context embedding allowing to naturally handle arbitrary-dimensional observation spaces since the number of agents in a sub-scene changes, and the single-track vehicle kinematic model is explicitly part of the action space in the form of continuously valued steering and velocity increments sampled from Gaussian distributions, such that generated trajectories are by construction valid in terms of physics rather than by post-hoc filtering. The reward function is composed of a multi-objective criterion composed of a goal-directed reward defined by a Gaussian heat map around the target key position, a penalty term for the acceleration and velocity increments, to better emulate comfortable driving, and a penalty for collisions. Such formulation implicitly accounts for both agent-centric prediction accuracy and scene-wise trajectory consistency across agents which agent-centric methods do not handle explicitly. As tested in the Argoverse Forecasting benchmark, the proposed hybrid framework reaches the state of the art for the main metrics (minADE, minFDE, Miss Rate) and has the best Drivable Area Compliance score which is a direct measure of how well the RL stage addresses rule-based generation, i.e. Produces accurate and valid trajectories when the DL agent might otherwise fail. In addition, this framework is able to reach the aforementioned performances when only 1% of the training dataset is used in RL stage to learn behavior which indicates that few environmental interactions are enough to successfully adapt.

However, several issues are overlooked. First, as demonstrated in the entire scope of trajectory prediction research [70], RL based approaches demand to hand-craft the reward function to express information about driving rules, collision, comfortable driving or similar and in complex multi-agent settings a wrong reward function may lead to the reward hacking problem and emergent unintended behavior. Second, [71] hybrid approach relies heavily on the precise key-position prediction at DL phase; an error at the intermediate prediction stage propagates to the sub-scene generation phase, therefore limits the planning strategy in RL to be a victim of this phase. It is thus sensitive to the distribution shift at the DL stage. Third, a reliable simulation system to train the RL part is necessary, and it must correctly mimic the actual environment so that an environment modeling assumption is made. However, when it

faces with the shift distribution of sudden accidents or uncommon traffic, such assumption may fail. Lastly, although the Transformer based PPO approach has an efficient capability in dealing with the varying observation dimensions; the execution of multiple RL planner at the inference phase at the per-agent level has an efficiency concern on the scale and latency of practical V2X edge application.

I.9.4 Critical analysis and Problem statement

Identified Gaps:

1. Lack of Non-stationarity Awareness: Supervised learning methods like LSTM, GCN perform extremely well on the stationary test set, however performance deteriorates drastically in a distribution shift scenario. Classical methods like ARIMA, Kalman are stationary by design and fail in such situations. No previous RL works in traffic flow consider online adaptation in a constantly changing traffic environment.
2. High Cost of Adaptation: The full re-training (long hours in GPUs) for deep learning methods is necessitated when the data distribution changes, the same holds for Meta-learning (though costs are reduced). Also it always requires a batch update with stored data.
3. Control versus Prediction: Existing works in RL in traffic focus mainly on control tasks (signal timing, routing etc.). But predicting the traffic patterns with more than one directional traffic trends and applying them proactively for V2X communication is a research gap in the domain of RL.
4. Limitations in Deployment: V2X communication in the real world involves real-time processing of data and it is necessary to infer predictions in milliseconds and that the methods used should adapt readily without any infrastructure re-arrangement. Current methods lack these combined capabilities.

I.9.4.1) Research Objective: The objective of this research is to overcome the above identified issues by presenting a hybrid framework consisting of two phases:

1. Directional Trend Prediction using RL (PPO): Using multi-RSU sensor fusion and temporal convolution encoding, an RL agent trained using PPO is developed that predicts the future dominant increasing traffic direction (North, South, East, West).
2. Real-time Online Metaheuristic Adaptation: Using simple metaheuristic algorithms like GA that can dynamically update a bias vector that modifies the logit outputs from the PPO model, ensuring that there are no costly retraining procedures needed and that adaptation can be performed in milliseconds even after encountering changes in the traffic pattern.

I.9.4.2) Key Contributions:

- The first online metaheuristic post-correction in an RL traffic prediction task.
- Achieving sustained accuracy levels even after the regime shifts in non-stationary are drastically observed.
- Inference speeds well below milliseconds that are suitable for edge computing environment in V2X systems.

- Performance is thoroughly compared against the current leading classical methods (ARIMA, Kalman), supervised learning (LSTM) and pure RL methods.

I.9.4.3) Hypothesis: Separating the representation learning part (done by PPO training in the offline mode) from the online adaptive part (done by GA post-processing), the proposed method combines the strengths of high capacity of deep RL and rapid adaptability of metaheuristics overcoming the stationary distribution limitation of all the methods.

I.10 Conclusion

In summary, we argued that mitigating urban traffic congestion demands a shift in urban traffic management from post-processing toward proactive mobility prediction due to an explosion in V2X communications and MEC platforms. However, predicting traffic within dynamic, non-stationary urban conditions is indeed a much harder task. Critically reviewing the current state-of-the-art, the inherent limitations of classical statistical models in capturing non-linear regime shifts are well understood; meanwhile, the state-of-the-art spatial-temporal DL models are unable to account for catastrophic forgetting and prohibitively slow re-training time at abrupt traffic distribution changes. Existing DRL approaches for traffic in Intelligent Transportation are centered around control problems rather than proactive prediction and fail to perform instantaneous online adaptation. The identified research gaps and challenges motivate the proposal of the herein hybrid framework which utilizes the PPO-based DRL agent for directional trend prediction and the lightweight metaheuristic for real-time refinement.

Extending directly from this conceptualization, chapter II will specify the physical and mathematical environment wherein this hybrid system would perform. Specifically, it will introduce the novel proposed three-tier smart city system architecture-composed of Physical Perception Layer, 5G Backhaul, and Edge AI Intelligence Hub-and synthesize non-stationary traffic data models, including inter-section traffic spillover effects and rush-hour traffic demand variations, based on a real urban road network. Additionally, it will mathematically formulate the multi-class directional trend prediction problem and set the stringent communication, computation and mobility constraints in the real-time operation of the system.

CHAPTER II
System Model and Problem
Formulation

II.1 Introduction

This chapter establishes the structural, data-driven, and mathematical foundations for the smart city traffic prediction system proposed in this thesis. To support mobility prediction, the designed system relies on the coordinated interaction of heterogeneous communication and computation entities within a grid-based urban environment.

First, we detail the three-tier system architecture, which is comprised of the physical perception layer (vehicles and Roadside Units), the 5G access and backhaul layer (Next-Generation Base Stations), and the edge intelligence layer (Multi-Access Edge Computing servers). Following the architectural overview, we outline the synthesized traffic data models, detailing the specific RSU state features and the spatiotemporal dynamics such as cross-intersection spillover and rush-hour demand that the system must process. Finally, we formalize the core prediction task as a multi-class directional traffic trend classification problem. We express this multi-objective optimization mathematically, setting the constraints for communication capacity, MEC computation limits, and vehicle mobility that will govern the learning model.

II.2 System Architecture

II.2.1 Smart City System Overview

In this thesis, the system designed operates in a structured smart city environment in order to support real time mobility prediction and traffic management by facilitating a coordinated interaction of heterogeneous communication and computation entities. It consists of a grid-based road network populated by moving cars and a tier of Roadside Units (RSUs) placed at each junction; also involved are a wide area network coverage provided by a 5G Next Generation Base Station (gNB); and a Multi-Access Edge Computing (MEC) server to function as a centralized control unit of the entire system. The above components are linked together using two kinds of communication technologies: (1) the 5G cellular network to serve the high throughput links between base station and MEC server; (2) the IEEE 802.11p standard for the direct low-latency communication among vehicles and infrastructure (V2I and V2V). They collectively form an end to end data pipeline in which we observe raw mobility data from the smart city, gather it to the edge to perform predictions on traffic states. [74].

The entire system can be described as a three-tier architecture, conceptually mapped in Figure II.1. The first tier is physical perception and communication layer that includes mobile cars and stationary RSUs. In this tier, the communication is carried out via the IEEE 802.11p link between road side units and vehicles. The second tier is the 5G access and backhaul layer, which facilitates communication among RSU, gNB, and MEC server. Finally, the third tier is the edge intelligence layer where learning models running on MEC server process incoming data to predict traffic [75].

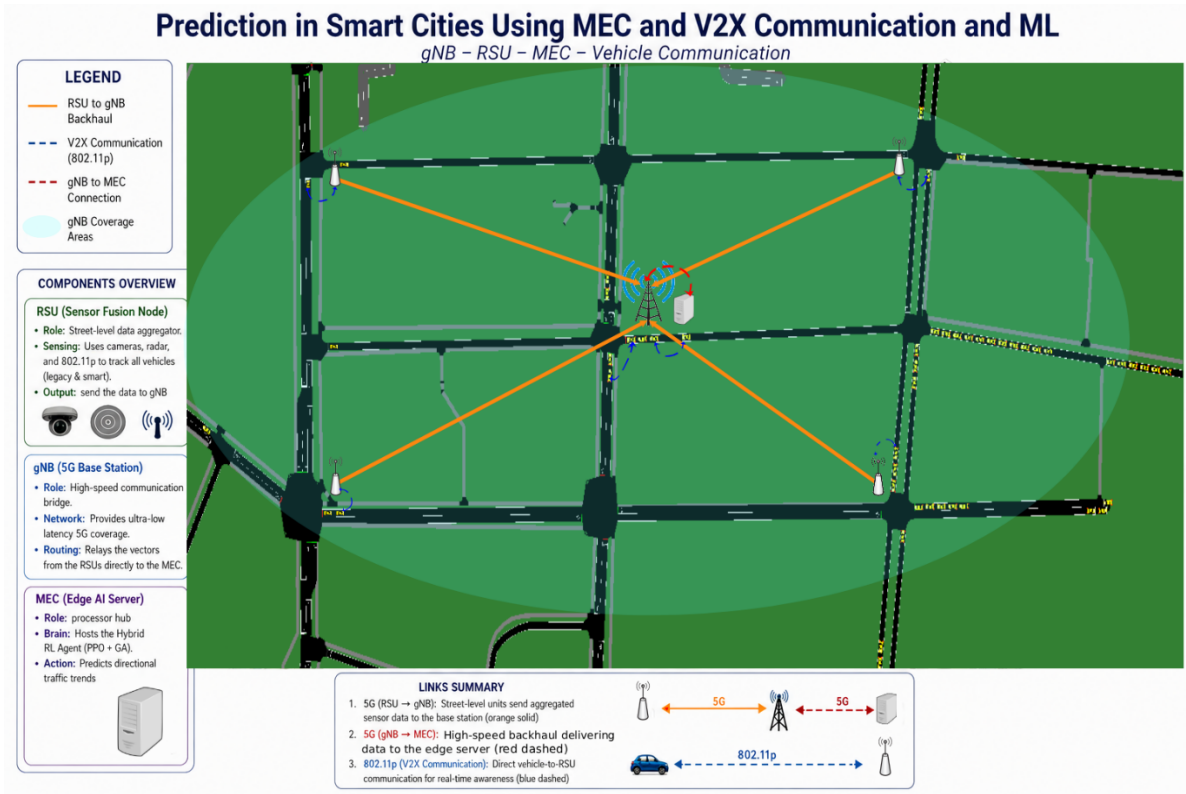


Figure II.1: System Architecture

II.2.1.2 Smart City Road Network

The smart city is modeled as an urban road network of various intersections and described by a directed graph $G(V,E)$ where V and E denote respectively set of intersections (or monitoring points) and the road segments between them. As is displayed in the Figure II.1, it is composed of a set of grid-based horizontal and vertical roads. Intersections are strategically placed where horizontal roads and vertical roads cross and become monitored. A grid road network structure is chosen as it closely resembles the topology of numerous city blocks in urban environments, which is commonly adopted as a standard test-bed environment for ITS research [76].

The road network is filled with a group of heterogeneous vehicle population, including private cars, trucks and buses. Vehicles travel on the roads, decelerate on the junctions and select turning lanes according to their itineraries and the traffic signals. The densities, speed, flow in the roads and junctions change constantly because of trips generation patterns, traffic light coordination, accidents and interaction among vehicles which eventually form the non-stationary spatiotemporal dynamics for which traffic prediction is carried out [77].

II.2.1.3 Roadside Units (RSUs) - Sensor Fusion Nodes

At the intersection level, the system employs a set of Roadside Units (RSUs) each acting as a multi-modal sensor fusion node. It is responsible for acquiring, aggregating, and relaying traffic observations in the street-level environment. As demonstrated in Figure II.1, an RSU is deployed at every intersection of the grid, enabling a full sensor coverage over the entire road network. The sensing modalities onboard the RSUs are high-resolution cameras to capture vehicles for detection and classification, radar sensors for range and velocity sensing, and

IEEE 802.11p communication transceivers to communicate with equipped passing vehicles [78].

Each RSU acts as a data collection point for the system. It receives data packets carrying vehicle states (position, velocity, heading, type etc) from surrounding V2X-equipped vehicles in the form of Cooperative Awareness Messages (CAMs) through its IEEE 802.11p interface—blue dashed links in Figure II.1. In parallel, using its onboard cameras and radar sensors, the RSU captures behavior of legacy, non-V2X-enabled vehicles within its vicinity to obtain a complete picture of the traffic despite a potentially low V2X penetration rate. The RSU fuses these sensor inputs to extract a spatially-referenced observation at each intersection and relays the aggregated data to the gNB via the 5G backhaul link [78].

Formally, let $R = \{r_1, r_2, \dots, r_N\}$ be the set of N RSUs deployed over the road network. Each RSU r_i operates within a spatially confined neighborhood N_i (typically an intersection) and monitors all the vehicles that fall within N_i . The observation vector of the RSU r_i at time t , $o_i(t)$, captures key traffic features at the corresponding intersection such as traffic density $p_i(t)$, average vehicle velocity $v_i(t)$, prevalent vehicle direction of travel $d_i(t)$, and queue length $q_i(t)$. The set of observation vectors from all RSUs in the network constitutes the input to the prediction and control stage [77].

II.2.1.4 5G Next-Generation Base Stations (gNB)

The 5G Next-Generation Base Station (gNB) is responsible for establishing the wireless link between the street-level RSUs and the MEC-based intelligence. The gNB is centrally located in the urban grid, covering a widespread area—shown as blue elliptical coverage area—and including all deployed RSUs and vehicles within the modeling environment. Utilizing the 5G NR spectrum, the gNB provides the ultra-low latency and high-throughput wireless link that is essential for the predictive and control functionalities of the system [79].

The gNB serves as a crucial communication relay within the system architecture. First, it receives the aggregated traffic observation packets from each RSU through 5G wireless links—represented by solid orange links in Figure II.1. Subsequently, the gNB forwards these concatenated data streams to the MEC server through a direct, high-speed backhaul connection represented by red dashed links in Figure II.1, to be processed for traffic prediction. The latency associated with this backhaul connection is considered negligible for the entire system's performance.

II.2.1.5 MEC Server - Edge AI Intelligence Hub

The MEC server is the core of the MEC system. It receives the observations from all the RSUs and runs the machine learning algorithms that lead to the prediction of traffic flow. As described in Figure II.1, the MEC server is located along the gNB (common topology for vehicular MEC systems) and connected by a high speed connection (red dashed lines). This proximity to the radio access network enables a low round trip latency for predict the traffic, required to be at the millisecond level [80].

The MEC server hosts the system Hybrid RL Agent. As in Figure II.1 components overview, the agent combines the PPO and GA algorithms for traffic prediction optimization

jointly. On reception of all RSUs observation vectors at each control time step, the MEC server determines a global traffic state of the entire network, by considering a graph with all intersections included, representing current traffic situation at every intersection. This global state is input for a neural network running the PPO algorithm. The PPO neural network outputs traffic predictions for future time steps [81].

The MEC server is equipped with a finite computation capacity $C_{\{MEC\}}$ (CPU cycles/second), the maximum amount of processing that the server can provide in each time step. At the MEC server, time steps are managed in such a way that no computation overload occurs for the determination of the global state, for the prediction of future traffic states, guaranteeing that processing latency is bounded to the allowed range [77].

II.3 Data and Assumptions

II.3.1 Data Source and Road Network

The data that this thesis uses is synthesized using a simulation framework built over a real-world urban road network taken from the city of Nördlingen, Germany, as depicted in Figure II.2. The road graph is fetched using the OSMnx library, which is an open-source tool that queries Open Street Map to obtain the complete drivable street network of the specified area within a 1.2 km radius. The deployment of RSUs follows a geographically-constrained, degree-prioritized selection strategy designed to maximize information coverage while maintaining strict spatial distribution requirements. The RSU deployment process proceeds in three stages. First, all intersection nodes within the road network are projected into the EPSG:3857 coordinate system (Web Mercator) to enable accurate Euclidean distance calculations in meters. The centroid of all nodes is computed and designated as the central location of the 5G Next-Generation Base Station (gNB). Second, all nodes within an 800-meter radius of the gNB centroid are identified as candidate RSU locations this ensures that all deployed RSUs maintain reliable 5G uplink connectivity to the base station. Third, from this candidate set, nodes are ranked by their graph degree (the number of connected road segments) in descending order. RSUs are sequentially selected from this ranked list subject to a minimum inter-RSU spacing constraint of 300 meters.

The deployment process continues iteratively until the system's maximum target capacity of 20 RSUs is reached, or the pool of valid candidate nodes is exhausted. Because the actual number of deployed RSUs is determined dynamically based on the specific road network topology, not all urban environments will support the maximum capacity under strict spatial constraints. This strategy ensures that RSUs are placed in geographically diverse monitoring areas instead of accumulating at one high-degree interchange, providing full network observability. In the Nördlingen network deployment, the candidate pool of intersections satisfying both the 800-meter gNB range and the 300-meter minimum inter-RSU spacing was exhausted after placing 15 nodes. Thus, the network successfully yields a final deployment of $N_{\{RSU\}} = 15$ RSUs that fulfill all structural requirements. High-degree nodes were selected because they correspond to the busiest areas in the road network as a whole the intersection of traffic flows providing a large spatial information gain, while the minimum distance constraint helps prevent duplicate measurements and ensures each RSU covers a separate

spatial region. We now formally introduce the elements involved in the directed road graph that was obtained from Open Street Map: $G=(V, E)$, where V is the set of nodes (intersections), and E is the set of edges (road segments). For any node v in V , the degree $deg(v)$ is the number of edges of E which have v as an endpoint:

$$deg(v) = |\{e \in E : v \in e\}|$$

Let $(x_{\{gNB\}}, y_{\{gNB\}})$ Denote the calculated gNB centroid location in EPSG:3857 coordinates. The candidate set of intersections is defined as:

$$V_{\{\text{candidate}\}} = \{v \in V : \sqrt{\{(x_v - x_{\{gNB\}})^2 + (y_v - y_{\{gNB\}})^2\}} \leq 800 \}$$

The RSU placement set $R = \{r_0, r_1, \dots, r_{N-1}\}$ is constructed sequentially from $V_{\{\text{candidate}\}}$, ordered by $deg(v)$. The highest degree vertex is added to R , then the next vertex on the sorted list is examined. It is added to R only if its Euclidean distance from any vertex previously added to R is at least 300 meters. The process continues until $|R|=15$. Lastly, the latitude and longitude $(\{lat\}_i, \{lon\}_i)$ values of each RSU r_i are directly extracted from the corresponding intersection vertex v_i in G , as depicted in Figure II.2. This places the network in the topology of an urban road network and result in a more realistic traffic dynamics synthesis than that in a purely artificial grid graph.

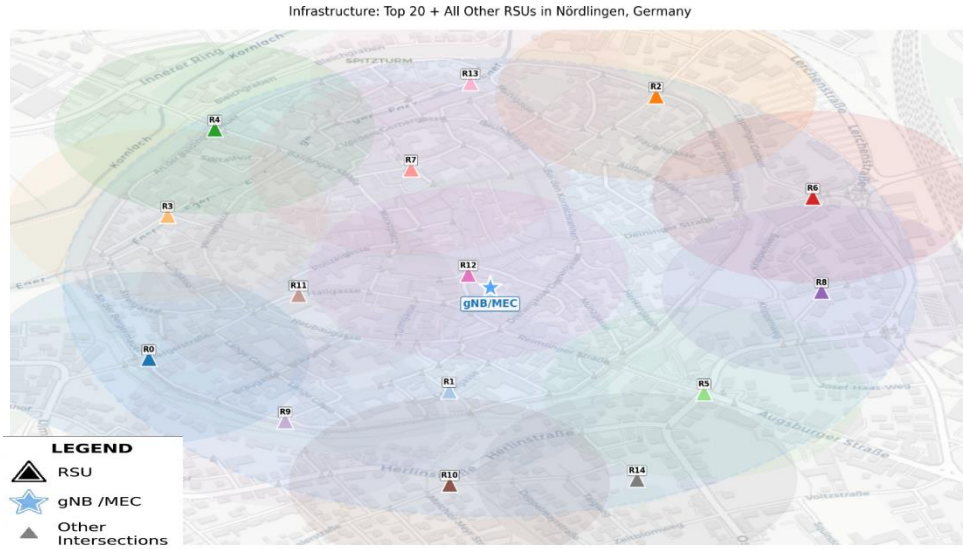


Figure II.2: Coverage Map of the Top 20 Roadside Units (RSUs) and GNB/MEC Infrastructure in Nördlingen, Germany.

II.3.2 Dataset Generation and Temporal Structure

The system utilizes the synthesized traffic data generated by a physics-inspired simulation model which aims to model a realistic urban traffic dynamics over a total duration of $T = 200,000$ time steps. Each time step in the system corresponds to 1 minute real time which sums up to approximately 138 days of urban traffic dynamics in the complete dataset. All the time steps in the dataset are assumed to cycle through every hour of the day from 0 to 23. In

order to do this the hour of day in the dataset is defined as $hour(t) = \left(\frac{t}{60}\right) \bmod 24$ where t is the index of the time step.

The generated data is in three arrays, where these three arrays combine to define the entire observation space as well as the label for the learning problem:

- $X \in \mathbb{R}^{(T \times N_RSU \times state_dim)}$ — the normalized RSU state feature tensor, with $T = 200,000$ time steps, $N_RSU = 15$ RSUs, and $state_dim = 8$ features per RSU per time step.
- $D \in \mathbb{R}^{(T \times N_RSU \times N_dir)}$ — the normalized directional density tensor, capturing the traffic density flowing in each of the four compass directions (North, East, South, West) at each RSU at each time step, with $N_dir = 4$.
- $Z \in \mathbb{Z}^T$ — the ground-truth dominant direction label for each time step, taking values in $\{0=\text{North}, 1=\text{East}, 2=\text{South}, 3=\text{West}\}$, computed from the future trajectory of D over a prediction horizon of $H = 60$ time steps.

The dataset is divided into an 80% training and a 20% testing set with a chronological split, which helps maintain temporal order and prevents future information from influencing the training. Normalization on X and D is performed globally by using Z-score standardization: every feature is first standardized by subtracting its global mean and then scaling it by dividing by its global standard deviation calculated across all time steps and RSUs. This ensures that the inputs have values within a similar range and learning algorithm is not sensitive to varying scales of the features [53].

II.3.3 RSU State Features; Collected Data Variables

At each time step t , each RSU r_i produces a state observation vector $X[t, r_i] \in \mathbb{R}^8$ Containing eight distinct traffic and network state features. These features are designed to capture the multi-dimensional nature of urban traffic conditions, combining vehicle-level mobility metrics with infrastructure-level network performance indicators. The eight features, which are visually summarized in Figure II.5, are defined as follows:

Feature 1 — Traffic Density (ρ): The number of vehicles within the RSU sensing zone, clipped to $[0, 80]$ vehicles. Density is the foundational feature from which all dependent observations (speed, queue, delay, packet loss, latency, occupancy) are directly derived, as established by macroscopic traffic flow theory [83]. It is generated by scaling a double-Gaussian rush-hour demand signal and clipping to the physical range:

$$\rho_{base(t,r)} = clip\left(40 \cdot \left[e^{-\frac{(hr-8-\delta)^2}{4}} + e^{-\frac{(hr-17-\delta)^2}{4}}\right] + N(0,5), 0, 80\right)$$

Where $h_r = (hour(t) - offset_r) \bmod 24$ is the RSU-adjusted simulation hour and δ is the temporal drift term defined in Section 2.3.1. The final observed density $\rho(t, r)$ is the result of applying the cross-intersection spillover mechanism to $pbase$ (Section 2.6).

II.3.3.1 Rush Hour Demand Model

The double-Gaussian model captures the two characteristic demand peaks of a typical urban working day — the morning commute at 08:00 and the evening return at 17:00 — a

bimodal pattern consistently confirmed across urban networks worldwide [52]. The demand signal for RSU r at adjusted hour h_r is:

$$rush(h_r, \delta) = e^{-\frac{(h_r - 8 - \delta)^2}{4}} + e^{-\frac{(h_r - 17 - \delta)^2}{4}}$$

Each Gaussian has a standard deviation of $\sigma = \sqrt{2} \approx 1.41$ hours producing a full-width-at-half-maximum of approximately 2.35 hours, consistent with empirical peak durations [52], yielding a base density of 40 vehicles at peak and near-zero during nighttime troughs.

Per-RSU Temporal Offsets. Each RSU r is assigned a distinct temporal offset $_r$ drawn from a pre-defined array spanning $[0, 168]$ time steps, so that:

$$hr = (hour(t) - offset_r) \bmod 24$$

These offsets reflect the spatial heterogeneity of demand across the Nördlingen network; without them, all RSUs would peak simultaneously, producing an unrealistically uniform pattern trivially learnable by time alone [50].

Non-Stationary Drift. A sinusoidal drift term shifts the Gaussian peak centers slowly over time:

$$\delta(t) = \left(\frac{45.0}{60}\right) * \sin\left(\frac{2 * \pi * t}{200000 * 0.3}\right) = 0.75 * \sin\left(\frac{2 * \pi * t}{60000}\right)$$

This drift has an amplitude of 0.75 hours and a period of ≈ 41.7 days, shifting rush-hour peaks by up to 45 minutes over the full dataset, as plotted in Figure II.3. The resulting non-stationarity forces the model to learn a temporally robust representation rather than memorising fixed-time patterns [52].

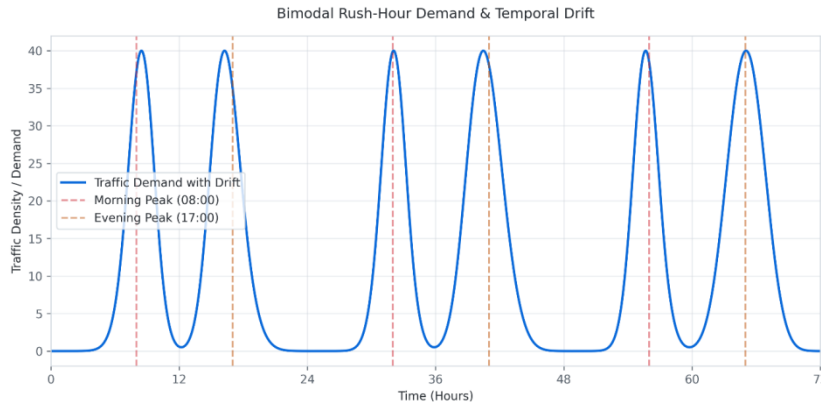


Figure II.3: Temporal Evolution of Traffic Demand with Morning and Evening Peak Hours.

Feature 2 — Average Vehicle Speed (v): The mean speed of vehicles within the RSU zone, clipped to $[5, 60]$ km/h: $v = \text{clip}(50 - \rho \times 0.4 + N(0,3), 5, 60)$ km/h, as clearly plotted in Figure II.4. This linear inverse relationship between speed and density is grounded in the Green shields model [22], one of the oldest and most widely validated macroscopic traffic flow models, which hypothesizes a linear speed-density relationship of the form $u = u_f(1 - k/k_j)$, where u_f is the free-flow speed and k_j is the jam density. The coefficient 0.4 in the

formulation is calibrated so that at zero density the free-flow speed is 50 km/h, and at maximum density ($\rho=80$) speed falls to approximately 18 km/h consistent with empirical urban arterial measurements under congested conditions [75].

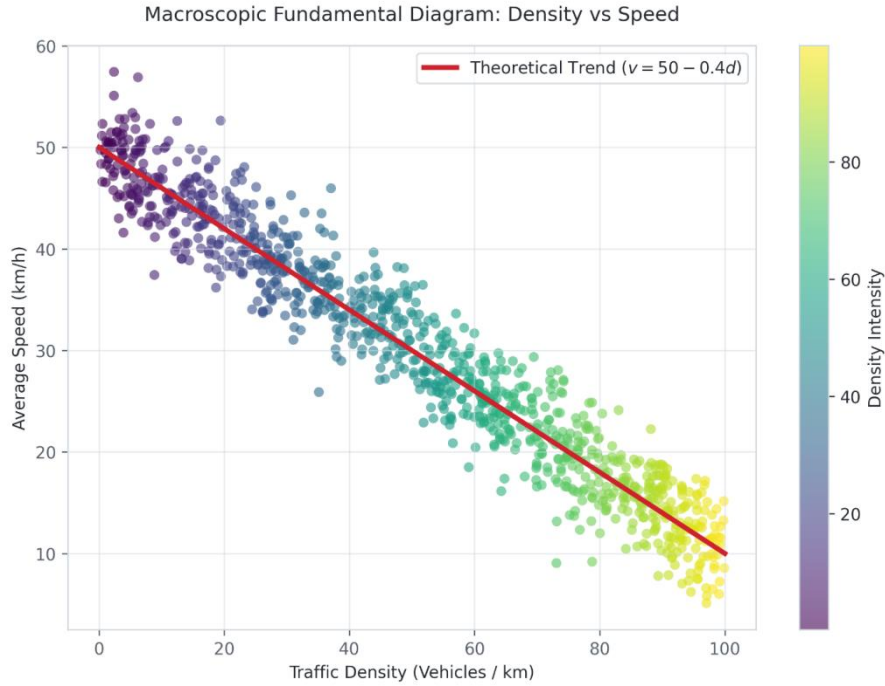


Figure II.4: Macroscopic Fundamental Diagram Illustrating the Relationship Between Traffic Density and Vehicle Speed.

Feature 3 — Queue Length (q): The number of vehicles queued at the intersection stop line, modeled as $q = \text{clip}(\rho \times 0.3 + N(0,2), 0, 30)$ vehicles. The linear proportionality between queue length and density follows from traffic wave theory, in which the length of a queue at a signalized intersection grows in proportion to the vehicle arrival rate and thus to the upstream density [76]. The scaling factor 0.3 is calibrated so that at maximum density the queue reaches 24 vehicles before noise, corresponding to realistic stop-line queue lengths observed at urban intersections operating near capacity.

Feature 4 — Average Delay (d): The mean additional travel time experienced by vehicles relative to free-flow conditions, modeled as $d = \text{clip}(\rho \times 0.15 + N(0,1), 0, 20)$ seconds. The positive monotonic relationship between density and delay is established by Webster's delay formula [72], the foundational model for intersection control delay, which shows that average vehicle delay grows with the degree of saturation (the ratio of arrival flow to capacity) and thus with upstream density. Webster's delay model, based on deterministic queuing theory, empirically calculates the average delay at signalized intersections as a function of traffic flow, cycle length, and saturation and remains the most widely referenced model in intersection performance analysis [77]. The coefficient 0.15 is set so that at maximum density the delay approaches 12 seconds before noise, consistent with level-of-service D/E thresholds defined in the Highway Capacity Manual for urban signalized intersections.

Feature 5 — Communication Packet Loss Rate (p_{loss}): The fraction of V2X communication packets that fail to be delivered, modeled as $p_{loss} = \text{clip}(0.01 + \rho \times 0.001 + U(0, 0.02), 0, 0.15)$. The positive dependence of packet loss on vehicle density is physically

grounded: as the number of simultaneously transmitting vehicles in the RSU zone increases, the IEEE 802.11p channel becomes congested through increased collision probability and channel busy ratio, degrading packet delivery rates. As vehicle density increases, broadcast transmissions can congest the communication channel, resulting in increased packet loss and fundamentally impacting the ability to identify threats in a timely manner [18]. In dense urban traffic scenarios, where the number of vehicular user equipment candidates exceeds the ideal-state channel capacity, the safety margin of V2X protocols is pushed to its limit. The model assigns a baseline loss rate of 1% at zero density, growing linearly with density to a maximum of approximately 9% at $\rho=80$ before noise, with the uniform noise term $U(0, 0.02)$ representing stochastic channel variations.

Feature 6 — Communication Latency(λ): End-to-end latency of the transmission of V2X packets between vehicles and the RSU (ms). The increase in the value when vehicle density increases is due to channel contention on the IEEE 802.11p medium leading to more queuing and collision resolution delays. The latency is modeled as: $\lambda = \text{clip}(1 + \rho \times 0.02 + N(0,0.5), 1, 10)$ ms At zero density, the baseline latency is 1 ms, growing to approximately 2.6 ms at maximum density, consistent with empirically measured DSRC channel delays under varying load conditions [30].

Feature 7 — Road Occupancy Ratio (η): The fraction of the road surface within the RSU zone currently occupied by vehicles, computed as $\eta = \text{clip}(\rho / 80 + U(0,0.1), 0, 1)$. Occupancy is a dimensionless measure of traffic density that is directly observable by loop detectors and radar sensors and is widely used as a primary input in traffic management systems [79].

Feature 8 — Time-of-Day Encoding (τ): A cyclic temporal encoding of the current hour, computed as $\tau = \sin(2\pi \times \text{hour} / 24)$. This sinusoidal encoding maps the 24-hour clock onto a continuous, periodic scalar that preserves the cyclic nature of diurnal traffic patterns — ensuring that the model recognizes the similarity between, for instance, 23:00 and 01:00 without treating them as maximally distant points on a linear scale [17].



Figure II.5: Composition of the Roadside Unit (RSU) State Vector X.

II.3.4 Directional Traffic Data

In addition to the eight scalar state features, each RSU r_i generates a directional density vector $D[t, r_i] \in \mathbb{R}^4$, as structurally decomposed in Figure II.6, which describes the amount of traffic for the 4 compass directions: North, East, South, and West for every time step. Each of the directions $D[t, r_i, d]$ is modeled as a demand driven by rush hour pattern for each of the 4 directions, and phase difference $\text{dir_phase}[r_i, d]$ in $\{0, 1 \dots 19\}$ models the different time-profiles for demand for origin-destination flow through the intersection. The density at RSU r_i at time for the d direction is calculated as:

$$D[t, r_i, d] = \text{clip}(\text{rush}_d \times 35 + \varepsilon, 0, 70)$$

Where $\text{rush}_d = \exp\left(-\frac{(h_d - 8)^2}{5}\right) + \exp\left(-\frac{(h_d - 17)^2}{5}\right)$ is a double-Gaussian rush-hour profile centered at 08:00 and 17:00, $h_d = (\text{hour} - \text{dir_phase}[r_i, d] - \text{drift}) \bmod 24$ is the direction-adjusted hour, and $\varepsilon \sim \mathcal{N}(0, 4)$ is stochastic measurement noise. This formulation produces directional density signals that peak at morning and evening rush hours but with timing variations that differ by direction, RSU, and epoch generating the rich spatiotemporal heterogeneity that the prediction model must learn to interpret [83]. The ground-truth label $Z[t]$ is derived from D as the index of the compass direction that will experience the greatest mean density increase over the next $H = 60$ time steps:

$$Z[t] = \text{argmax}_d \{ \text{mean}(D[t + 1 : t + H, :, d]) - \text{mean}(D[t, :, d]) \}$$

This formulation defines the prediction target as the dominant directional traffic trend rather than the instantaneous dominant direction, making the task a forward-looking classification problem that requires the model to capture the temporal evolution of directional flows across the full network. The resulting label distribution reflects the natural asymmetry of urban traffic demand, with certain directions consistently dominant during peak hours a class imbalance that is addressed through per-class reward weighting in the PPO training procedure.

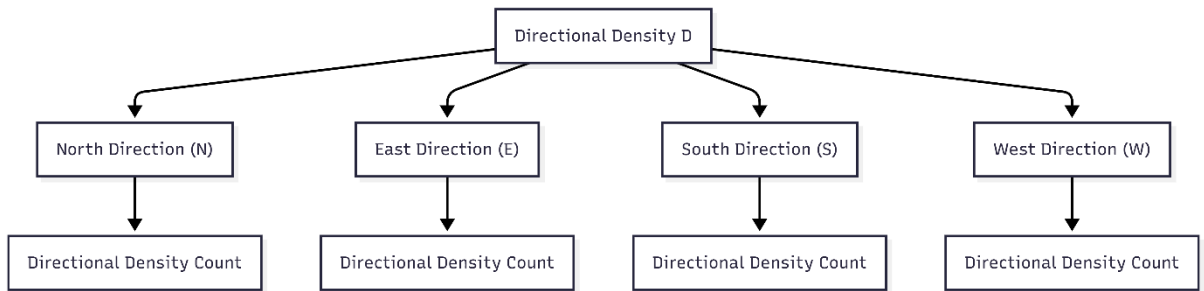


Figure II.6: Composition of the Directional Density Metric D.

II.3.5 Observation Window and Input Tensor

The learning agent does not observe individual time step snapshots in isolation but instead receives a sliding temporal window of the most recent $\text{seq_len} = 60$ time steps as its observation. At each decision step t , the observation tensor is constructed by concatenating the

state feature array and directional density array along the feature axis for every time step in the window:

$$obs(t) = \text{flatten}(\text{concat}(X[t - 60 : t], D[t - 60 : t], \text{axis} = -1))$$

This produces a flat observation vector of dimension: $obs_dim = seq_len \times N_RSU \times (state_dim + N_dir) = 60 \times 15 \times (8 + 4) = 10,800$ features.

The agent thus has access to not only the current, but also the recent temporal history of traffic over all 15 RSUs, which constitutes the multi-step, multi-RSU context required to differentiate two 'similar looking' traffic states which have divergent future evolutions. The 60-step window is equal to 60 minutes of simulated traffic history which can be considered a characteristic time-frame for the development of the rush hour transition period [83].

II.3.6 Traffic Dynamics Model — Spillover and Non-Stationarity

There are two factors used in the traffic simulation that provide complexity and challenge to the data beyond merely independent rush hour periods.

Cross-Intersection Spillover: Congestion in one intersection will spill over to adjacent intersections. At each time step t , the density at RSU r_i is augmented by a fraction (cross_spillover = 0.08, i.e., 8%) of the density at the adjacent RSUs $r_{\{i-1\}}$ and $r_{\{i+1\}}$:

$$\begin{aligned} spill[t, r] = & \text{clip}(base[t, r] + 0.08 \times base[t - 1, (r - 1) \bmod R] + 0.08 \\ & \times base[t - 1, (r + 1) \bmod R], 0, 100) \end{aligned}$$

This spillover mechanism models the real-world phenomenon of queue propagation where oversaturation at one intersection backs up into upstream links and introduces spatial correlation across RSU observations that the prediction model must learn to exploit [81].

Non-Stationary Regime Shifts: To evaluate the robustness of the prediction model under distributional shift, the system also generates a separate non-stationary test stream in which traffic conditions undergo four abrupt regime changes, corresponding to hourly demand patterns characteristic of: a morning rush hour (08:00, in-distribution), a weekend evening (20:00, sudden shift), late night (02:00, low-traffic), and an afternoon rush hour (17:00, partial shift). These regime transitions are designed to simulate the types of unexpected traffic pattern changes that real-world deployment would encounter, and are used to evaluate the online adaptation capability of the PPO+GA and PPO+PSO hybrid agents relative to the frozen LSTM baseline [52].

II.3.7 Network and System Assumptions

The following assumptions dictate the operating environment of the system that has been modeled in this thesis. These assumptions place limits on the scope of the system model and frame the context within which the mathematical formulation that follows in Section IV holds.

A1 - Synchronized Time steps: All of the RSU's maintain a synchronized 1 minute observation window. All observations from $N_RSU = 15$ of RSUs is transmitted, collected

and observed by the MEC server at the beginning of the same 1 minute interval with no inter-RSU synchronization delay.

A2 - 5G Channel Availability: All 5G uplink channels from each RSU to gNB are assumed to be open for communication at all times throughout the duration of the simulation. The capacity of these channels is assumed to be sufficient to allow transmission of all RSU observations simultaneously within each 1 minute window such that data is not dropped due to bandwidth over capacity.

A3 - V2X Penetration Rate: In this system a portion of the fleet will be comprised of vehicles that can transmit directly to the RSUs through IEEE 802.11p at 10Hz (V2X capable vehicles), and those that cannot be detected through sensors such as radar and camera at the RSU (legacy vehicles). The feature 5 packet loss represents degradation of information due to reduced channel quality at higher vehicle densities and is indirectly representing the value of V2X penetration.

A4 - MEC Computational Sufficiency: All MEC server computation is assumed to be sufficient for the PPO policy network to perform one complete inference over the state space on one observation window within a 1 minute control period.

A5 - Static Physical Infrastructure: All RSUs, the gNB, and the road network itself are assumed to be static throughout the simulation. RSU movement, road network topology changes, and RSU failures are not modeled.

A6 - Label Validity: All of the ground truth labels $Z[t]$ are assumed to be created with knowledge of H future steps and an entire window of $H = 60$ future steps is therefore used. For the last H steps in the data ($t \geq T - H$) the labels are extrapolated from the last actual label ($t = T - H - 1$) and are not included in the index sets for either training or evaluation to avoid boundary issues.

II.4 Problem Definition

II.4.1 Nature of the Prediction Task

The core prediction problem addressed in this thesis is formally defined as a multi-class directional traffic trend classification task. Rather than predicting a continuous scalar quantity such as traffic volume or average speed at a single road segment which is the formulation adopted by the majority of traffic forecasting literature this work targets a higher-level, network-wide question: given the traffic states observed across all RSUs over the past 60 minutes, which compass direction will experience the greatest increase in traffic demand over the next 60 minutes?

This formulation is motivated by the operational needs of smart city traffic management systems, which require not only point estimates of future traffic density but actionable directional intelligence knowing that northbound traffic is about to surge allows signal controllers, navigation platforms, and route guidance systems to preemptively redistribute load, whereas a raw density forecast for a single link provides no such network-level guidance. The directional trend prediction problem thus occupies a level of abstraction

between fine-grained link-level forecasting and coarse-grained origin-destination demand estimation, targeting the scale at which real-time traffic management decisions are most effectively made [74].

II.4.2 Formal Problem Statement

We consider the city traffic road network to be observed by $N=15$ RSUs performing synchronized observations at discrete time steps, $t = \{1, 2, \dots, T\}$, $T=200,000$. We obtain observations over a sliding window of size $\text{seq_len}=60$ for each time instant t , corresponding to the time interval $[t-60, t]$. This gives rise to a joint observation tensor:

$$O(t) = [X[t-60 : t] \parallel D[t-60 : t]] \in \mathbb{R}^{60 \times 15 \times 8}$$

Where $X[t-60 : t] \in \mathbb{R}^{60 \times 15 \times 8}$ represent the RSU state feature history $D[t-60 : t] \in \mathbb{R}^{(60 \times 15 \times 4)}$ is the directional density history over the four compass directions (North, East, South, West), and it is flattened into a 10,800dimensional observation $o(t) \in \mathbb{R}^{10,800}$ which is fed into the prediction model.

We define the prediction target in terms of the traffic future within a horizon of $H = 60$ time steps. For each direction $d \in \{0=N, 1=E, 2=S, 3=W\}$, the directional trend score Δ_d represents the average increment of network-wide directional density from current time to the end of the horizon:

$$\Delta_d(t) = \text{mean}_{\{r=1\}}^{\{N\}} \left[\text{mean}_{\{\tau=t+1\}}^{\{t+H\}} D[\tau, r, d] \right] - \text{mean}_{\{r=1\}}^{\{N\}} D[t, r, d]$$

Then the ground truth dominant direction label is determined:

$$Z(t) = \arg \max\{d \in \{0,1,2,3\}\} \Delta_d(t)$$

Our problem is to learn a mapping function $f_\theta : \mathbb{R}^{10,800} \rightarrow \{0, 1, 2, 3\}$ that, when provided with observation $o(t)$, correctly predicts the dominant directional label $Z(t)$. It's a single-label, 4-class classification problem, where the four classes stand for the four compass directions, and predict which directional trend will be the dominant one in the future 60 minutes, $H=60$ provides sufficient lead time for preemptive traffic management decisions.

II.5 Mathematical Formulation

II.5.1 Problem Type

The optimization problem is multi-objective, requiring the simultaneous minimization of prediction error and inference latency while maximizing robustness under non-stationary traffic conditions three partially conflicting objectives that a single-objective formulation cannot capture.

II.5.2 Objective Functions

O1 — Minimize Prediction Error

The primary objective is to learn a policy π_θ that minimizes the directional misclassification rate, formalized as maximizing the expected cumulative discounted reward:

$$O1: \max J(\theta) = E_{\{\tau \sim p_{i\theta}\}} \left[\sum_{\{t=0\}}^{\{T\}} \gamma^t \cdot r(t) \right]$$

Where $\gamma = 0.99$ is the discount factor and the per-step reward is:

$$r(t) = \begin{cases} +2.0 \times w_{\mathbb{Z}}(t) & \text{if } a(t) = Z(t) \\ -1.0 & \text{otherwise} \end{cases}$$

With w_c the inverse-frequency class weight for direction class $c \in \{N, E, S, W\}$, The PPO composite loss at each update step is:

$$\begin{aligned} \mathcal{L}(\theta) &= \mathcal{L}_{CLIP}(\theta) - 0.5 \cdot \mathcal{L}_{VF}(\theta) + 0.10 \cdot \mathcal{H}(\theta) \\ \mathcal{L}_{CLIP}(\theta) &= I_t [\min(\rho_t \cdot \hat{A}_t, \text{clip}(\rho_t, 0.80, 1.20) \cdot \hat{A}_t)] \end{aligned}$$

Where $\rho_t = \frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio, $\hat{A}_t$ is the Generalized Advantage Estimate (GAE) with $\lambda = 0.95$, $\mathcal{L}_{VF}$ is the value function loss, and $\mathcal{H}(\theta)$ is the entropy bonus that prevents premature convergence to a single dominant direction class [80].

O2 — Maximize Robustness under Non-Stationary Conditions

$$\begin{aligned} O2: \max_{b \in \mathbb{R}^4} \text{Acc}(\tilde{\pi}_{\theta,b}, E_t) \\ \tilde{\pi}_{\theta,b}(a|s_t) = \text{softmax}(\text{logits}_{\theta}(s_t) + b) \end{aligned}$$

This allows the frozen base policy to adapt to abrupt regime shifts — morning rush, evening, late-night, and afternoon demand patterns — without modifying θ , which is the core motivation of the hybrid PPO+GA/PSO design [18].

II.5.3 Constraints

C1 — Communication Constraint (V2X Link):

The packet loss rate at each RSU r_i is bounded by channel and density conditions:

$$0 \leq p_{loss}(t, r_i) \leq 0.15 \quad \forall t, r_i$$

The total 5G uplink data volume per time step across all RSUs must not exceed channel capacity:

$$V_{total} = N_{RSU} \times seq_len \times (d_state + n_dir) \times 4 \text{ bytes} = 43,200 \text{ bytes/min}$$

The system must remain operational even when up to 15% of V2X packets are lost, relying on passive RSU sensing modalities to fill missing observations [52].

C2 — MEC Computation Constraint:

Total processing time at the MEC server must fit within the control time step:

$$\Lambda_{inference} + \Lambda_{GA/PSO} \leq \Delta t_{step}$$

The online GA/PSO bias optimization runs for 25 iterations over a population of 40 candidates in $\mathbb{R}^4$, triggered every 3 time steps, keeping per-time step overhead bounded and compatible with MEC hardware capacity [53].

C3 — Observation Validity Constraint:

A valid observation requires $\text{seq_len} = 60$ historical time steps, and the label requires $H = 60$ future time steps. The valid index set is therefore:

$$O_{\text{valid}} = \{ t \in \mathbb{Z} : 60 \leq t \leq T - 62 \}$$

C4 — Policy Update Constraint:

PPO clipping enforces a hard bound on the policy change per gradient step:

$$\rho_t(\theta) = \frac{\pi_\theta(at|st)}{\pi_{\theta_{\text{old}}}(at|st)} \in [0.80, 1.20]$$

$$\|\nabla_\theta \mathcal{L}(\theta)\|_2 \leq 0.5$$

C5 — Vehicle Mobility and Spillover Constraint:

Traffic density at each RSU is physically bounded and spatially coupled through cross-intersection spillover:

$$0 \leq \rho(t, r) = \text{clip} \left(\rho_{\text{base}(t, r)} + 0.08 \sum_{k \in N(r)} \rho_{\{\text{base}\}(t-1, k), 0, 100} \right) \leq 100$$

Where $N(r)$ denotes the set of RSUs adjacent to r and 0.08 is the spillover fraction, reflecting queue propagation between neighboring intersections [81].

II.5.4 Complete Problem Statement

The full optimization problem is stated as:

$$\max_{\{\theta, b\}} [J(\theta) + \text{Acc}(\tilde{\pi}_{\theta, b}, E_t)]$$

Subject to (C1) through (C5)

Given the computational infeasibility of jointly optimizing θ and b in real time, the problem is decomposed into two sequential stages: offline training of θ via PPO optimizing O1, followed by lightweight online optimization of b via GA or PSO optimizing O2 at inference time [74].

II.6 Conclusion

In summary, this chapter has defined the operational environment, the underlying data structure, and the formal mathematical constraints of the proposed smart city network, establishing that forecasting traffic within this multi-tiered architecture is a complex, multi-objective problem requiring continuous adaptation to non-stationary urban mobility patterns. Building directly upon this foundation, Chapter 3 will detail the specific algorithmic solutions designed to operate within this framework by first providing an overview of the hybrid

approach that combines deep reinforcement learning with metaheuristic optimization. It will then delve into the reinforcement learning formulation specifically outlining the state space, action space, and reward function followed by an explanation of the Proximal Policy Optimization (PPO) algorithm used for baseline traffic prediction. Furthermore, the next chapter will introduce an optimization-based refinement layer, formalizing the optimization problem and detailing the specific algorithms, such as Genetic Algorithms or Particle Swarm Optimization, utilized to adjust the predictive policy during sudden traffic regime shifts. Finally, Chapter 3 will conclude by exploring the integration of these models with Multi-Access Edge Computing (MEC) and V2X vehicular networks, presenting comprehensive hybrid system architecture complete with the step-by-step pseudo-code driving the entire system.

CHAPTER III

Proposed Methodology

III.1 Introduction

The preceding chapters established the theoretical and architectural context for this work: Chapter 1 introduced the problem of real-time directional traffic trend prediction in V2X-enabled urban networks, and Chapter 2 surveyed the MEC infrastructure and deep learning landscape that motivate the design choices made here. This chapter translates those foundations into a concrete, implementable methodology.

The approach presented in this thesis tackles this challenge using a two-layered hybrid architecture. In the first layer, a PPO agent is trained on 200,000 simulation steps of urban traffic on the Nrdlingen road network and learns a resilient prediction policy for traffic direction that accounts for spatiotemporal patterns and complexities, such as the peaks of morning and evening commute traffic, less traffic on weekends, and the propagation of congestion through space at different intersections. The neural network weights learned in this phase are frozen, and the prediction logits are continuously adjusted by a simple meta-heuristic (GA or PSO) based on real-time prediction feedback and can adapt to distributional changes without any gradient descent calculations or retraining. This division of labor allows the system to achieve strong accuracy on stationary traffic conditions while maintaining resilience under regime transitions that would cause a static supervised model to fail.

The remainder of this chapter is organized as follows. Section III.2 presents the complete system architecture as a top-down walkthrough of the full pipeline, covering the environment, the agent, the metaheuristic module, and the PPO training loop. Section III.3 formalizes the reinforcement learning problem: the state space, action space, reward function and the Gymnasium-compliant simulation environment built to train and evaluate the agent. Section III.4 details the PPO algorithm, the actor-critic neural network architecture built around a Temporal Convolution Network encoder, a multi-head spatial attention module, and parallel actor and critic heads and the full set of training hyper parameters. Section III.5 specifies the online optimization-based refinement layer, including the mathematical formulation of the bias correction problem and the operational details of both the GA and PSO optimizers. Section III.6 describes the integration of the hybrid pipeline into the MEC-V2X architecture, mapping each component of the system to the corresponding communication tier. Section III.7 consolidates the full procedure into two algorithmic pseudo-codes one for offline PPO training and one for online metaheuristic inference.

III.2 Overview of the Hybrid Approach

The methodology proposed in this thesis is built upon a two-stage hybrid architecture that combines the sequential decision-making power of Deep Reinforcement Learning with the adaptive global search capability of online metaheuristic optimization. The design addresses a fundamental limitation shared by all static supervised learning approaches: once trained, their prediction behavior is fixed and cannot adapt to unforeseen changes in the traffic distribution without expensive retraining. The proposed system decouples the problem into two complementary phases an offline training phase in which a Proximal Policy Optimization (PPO) agent learns a robust directional traffic trend prediction policy from simulated traffic data, and an online inference phase in which a lightweight metaheuristic layer (either a

Genetic Algorithm or Particle Swarm Optimization) continuously corrects the agent's output logits in response to real-time prediction feedback [51].

The two-stage structure offers a strategic advantage: the offline PPO policy captures the complex spatiotemporal regularities of diurnal traffic patterns morning and evening rush hours, weekend lulls, and seasonal drift across 200,000 simulated time steps, while the online metaheuristic layer handles the residual non-stationarity that the static policy cannot anticipate, such as abrupt regime transitions between morning rush, weekend evening, late-night, and afternoon demand profiles. This division of labor allows the system to achieve high accuracy on stationary traffic conditions while maintaining strong adaptability under distributional shift a balance that neither pure RL nor pure supervised learning can achieve alone [52].

III.3 Proposed System Architecture

The complete architecture of the proposed hybrid traffic trend prediction system is presented in Figure III.1. Before detailing each algorithmic component in subsequent sections, this section provides a top-down walkthrough of the full pipeline so that the interactions between the environment, the neural network agent, the online optimization module, and the PPO training loop can be understood as a coherent whole.

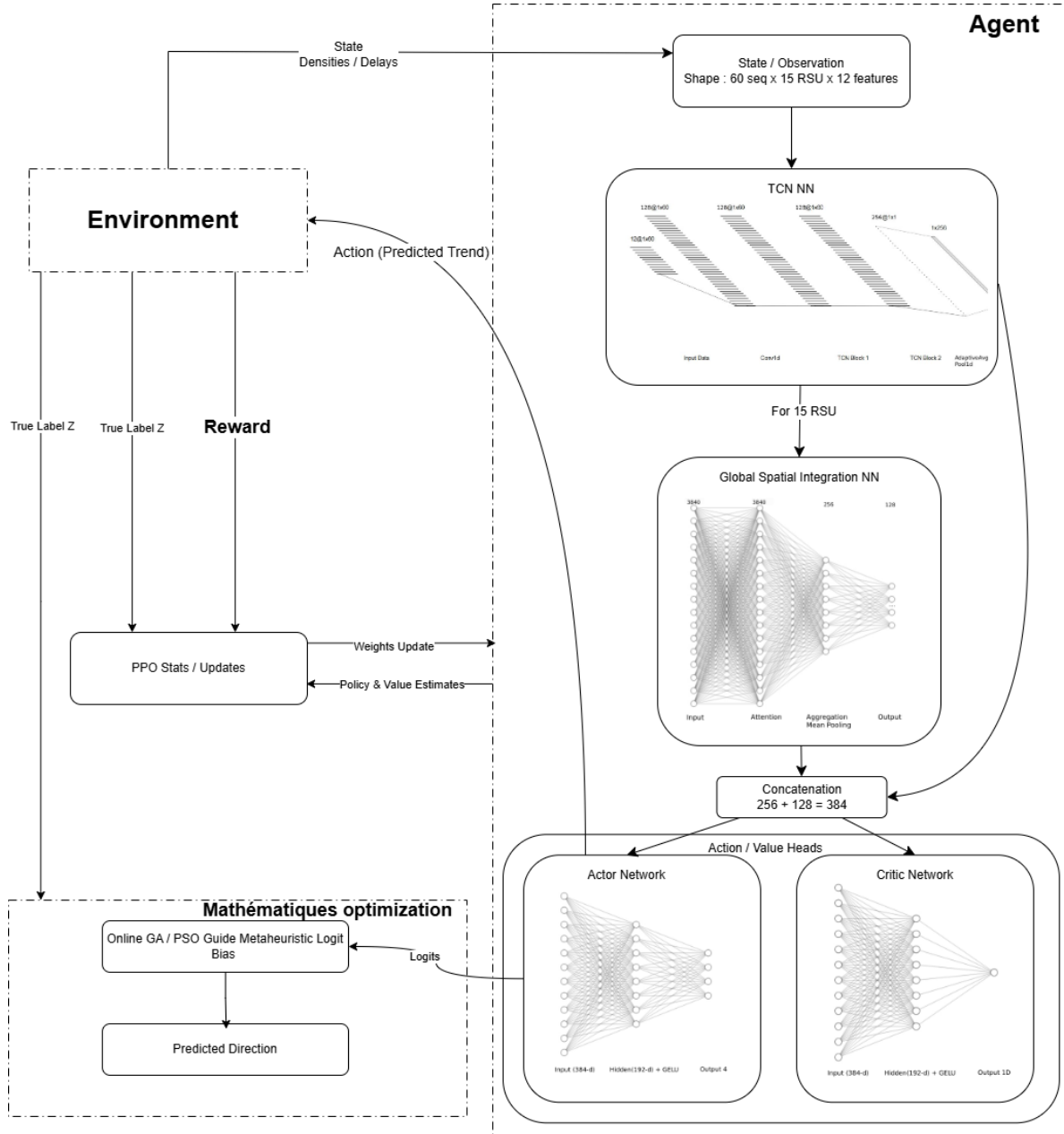


Figure III.1: Complete architecture of the proposed hybrid PPO + GA/PSO traffic trend prediction system.

III.3.1 Environment

The environment region simulates the urban traffic dynamics of the Nördlingen road network. At each decision step it produces a state observation encapsulating the current traffic conditions at all monitored intersections and exposes a ground-truth directional surge label representing the dominant traffic flow direction (North, East, South, or West) over the upcoming prediction horizon.

The state signal, labeled State Densities / Delays, is fed directly into the agent as a structured tensor. Simultaneously, the environment evaluates the agent's action the predicted directional trend against the true directional surge to compute a scalar reward.

III.3.2 Agent

The agent is responsible for transforming the raw traffic state into a directional prediction and a state value estimate. It is composed of three sequential processing stages: temporal feature extraction via a Temporal Convolutional Network, spatial integration via a global attention-pooling network, and action/value generation via two parallel output heads.

- **State Representation.** The agent's input is a three-dimensional observation tensor of shape $(60 \text{ seq} \times 15 \text{ RSU} \times 12 \text{ features})$, representing 60 minutes of historical traffic data collected across 15 roadside units, with each RSU contributing 12 features per time step a combination of directional vehicle densities and intersection delay measurements.
- **Temporal Convolutional Network.** The observation tensor is first passed through a TCN applied independently and identically to each of the 15 RSUs. The network processes each RSU's 60-step sequence through a causal entry block followed by two TCN blocks of progressively wider receptive fields, as illustrated in Figure III.2, ultimately projecting each RSU's temporal dynamics into a compact 256-dimensional embedding. Because the TCN is applied per-RSU, it captures the temporal evolution of traffic at each sensing point without any cross-RSU interference at this stage.

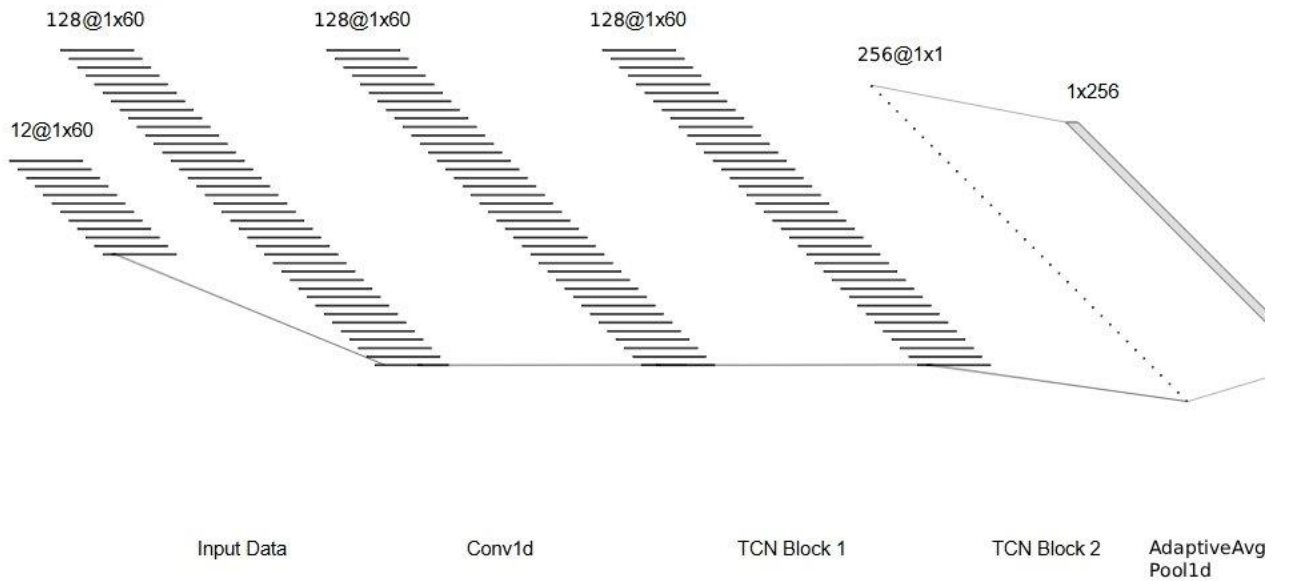


Figure III.2: Neural Network of Temporal Convolutional Network

- **Global Spatial Integration Network.** The 15 RSU embeddings produced by the TCN are then passed collectively into the Global Spatial Integration Network. This module applies multi-head self-attention across the RSU dimension with the input shaped as $15 \times 256 = 3,840$ allowing the model to learn which roadside units carry the most predictive signal for the global directional trend at any given moment. Following attention, the RSU representations are aggregated via mean pooling across all 15 units, producing a single 256-dimensional global representation. This combined vector is then further condensed to a 128-dimensional space by fully-connected projection

layer, as illustrated in Figure III.3, resulting in a compact spatial representation of the entire intersection network.

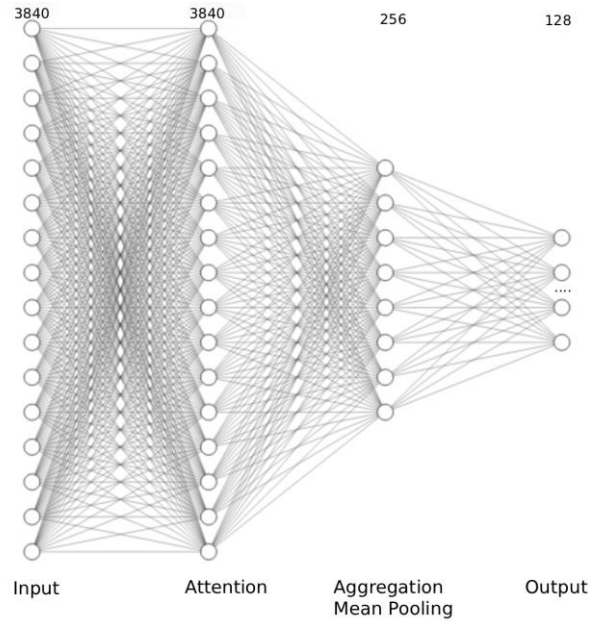


Figure III.3: Neural Network of Global Spatial Integration Network

- **Concatenation.** Both outputs are projected by two shared heads which take the concatenated mean-pooled and projected representations of 256 and 128 dimensions respectively as their input feature vector of 384 dimensions ($256 + 128 = 384$).
- **Actor Network.** The actor maps the 384-dimensional feature vector through a hidden layer of 192 units with GELU activation to produce 4 raw output logits, as illustrated in Figure III.4, one per compass direction. These logits represent the agent's unnormalized directional preferences and are passed downstream both to the environment as the predicted action and to the online optimization module as raw logits during inference.

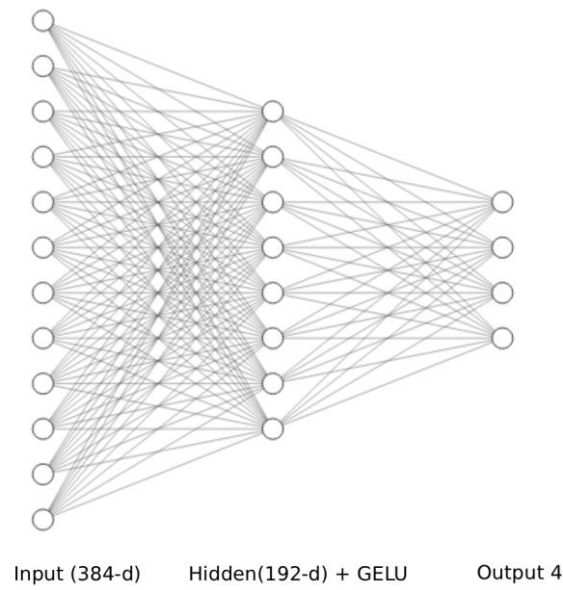


Figure III.4: Neural Network of Actor Network.

- **Critic Network.** In parallel, the critic maps the same 384-dimensional input through an identical hidden layer to produce a single scalar output $V(s)$ the estimated state value, as illustrated in Figure III.2. This value estimate is used exclusively during training to compute advantage estimates and is not involved in the prediction output.

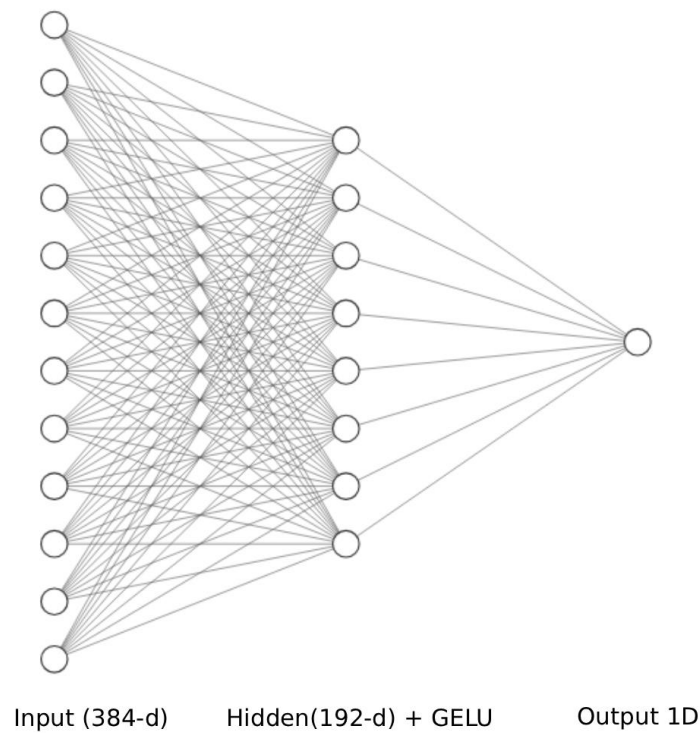


Figure III.5: Neural Network of Critic Network

III.3.3 Metaheuristic Optimization

The metaheuristic optimization module constitutes the inference-time component of the hybrid system. It operates exclusively on a frozen policy after PPO training has concluded and the neural network weights are no longer updated and its purpose is to continuously correct the actor's directional bias in response to recent prediction errors encountered in deployment.

III.3.4 PPO Training Loop — PPO Stats / Updates

The offline training is managed by the PPO training loop. At each training step, the environment reward and the value $V(s)$ estimate of the critic is used to obtain Generalized Advantage Estimation which in turn provides discounted returns and normalized advantage values. These are supplied to the combined PPO Loss which contains three terms: clipped surrogate objective from the actor, value function regression loss from the critic, and an entropy bonus which encourages exploration. This loss function is then back propagated to both the actor and the critic network which are trained using shared parameters. When the training converges, the weights are fixed and the system switches completely to online inference mode as described in III.3.

III.4 Reinforcement Learning Formulation

III.4.1 State Space

The state space S is defined as the set of all possible observation vectors that the agent can receive at any time step t . A state $s(t) \in S$ is constructed by concatenating the RSU state feature tensor X and the directional density tensor D over a sliding window of $seq_len = 60$ consecutive time steps, then flattening the result into a single vector [8]:

$$s(t) = \text{flatten}(\text{concat}(X[t - 60 : t], \quad D[t - 60 : t], \quad \text{axis} = -1)) \\ \in \mathbb{R}^{10800}$$

Where the observation dimension is:

$$obs_{dim} = seq_{len} \times N_{RSU} \times (state_{dim} + n_{dir}) = 60 \times 15 \times (8 + 4) = 10,800.$$

Each of the 15 RSUs contributes 12 features per timestep 8 traffic state features (density, speed, queue, delay, packet loss, signal phase, occupancy, time-of-day encoding) and 4 directional density features (North, East, South, West). The 60-step window provides the agent with one hour of traffic history, sufficient to observe a complete rush-hour transition and capture the temporal momentum of directional demand shifts [15].

The observation space is declared as a continuous Box with bounds $[-10, +10]$ after global Z-score normalization, reflecting the fact that all input features are standardized across the full dataset prior to training. The Markov property is approximately satisfied by the 60-step window: given the current state $s(t)$, the future directional trend $Z(t)$ depends primarily on the recent traffic history encoded in the window, and not on any further historical context beyond it [12].

III.4.2 Action Space

The action space A is discrete and has size 4, given by the 4 compass directions which represent the possible dominating trends of traffic in a particular direction. $A = \{ 0 = \text{North}, 1 = \text{East}, 2 = \text{South}, 3 = \text{West} \}$ For each time step t , the agent chooses an action $a(t) \in A$ which represents its forecast of the dominating trend in traffic for the next 60-minute horizon. The discrete action space is appropriate because the prediction target $Z(t)$ is inherently categorical it is the argmax of directional density increments across four mutually exclusive compass directions and because discrete action spaces admit the use of the Categorical distribution in the PPO actor, enabling analytically exact entropy computation and log-probability evaluation [1].

III.4.3 Simulation Environment

A core design requirement of the proposed hybrid system is that the reinforcement learning (RL) agent must be trained and evaluated in an environment that faithfully represents the V2X traffic prediction problem while remaining fully compatible with modern deep RL training frameworks. To satisfy both requirements, the simulation environment was built as a custom, fully Gymnasium-compliant environment. This section first situates the Gymnasium framework within the broader landscape of RL environment design, explains why it was adopted for this work, and then details the environment's inputs, observation structure, reward mechanism, episode management, and diagnostic interface.

III.4.3.1 The Gymnasium Framework

Gymnasium is an open-source Python library maintained by the Farama Foundation, serving as the community-maintained successor to OpenAI Gym, and has become the de facto standard interface for RL environment design [73]. It defines a base class with a minimal contract (`init`, `reset`, `step`, `render`) and two space objects describing the observation and action structure, allowing RL algorithms to configure their networks automatically. Gymnasium also offers composable wrapper classes that modify environment behavior (e.g., time limits, observation normalization, reward/episode logging) without altering the underlying code. Since the traffic-direction prediction task involves a single agent interacting with a discrete observation-action loop over a finite, pre-recorded dataset — as depicted in Figure III.6 — with no need for physics simulation or multi-agent coordination, Gymnasium's lightweight, well-documented interface, supported by major RL frameworks such as Stable-Baselines3, CleanRL, and RLlib, was the natural choice [73].

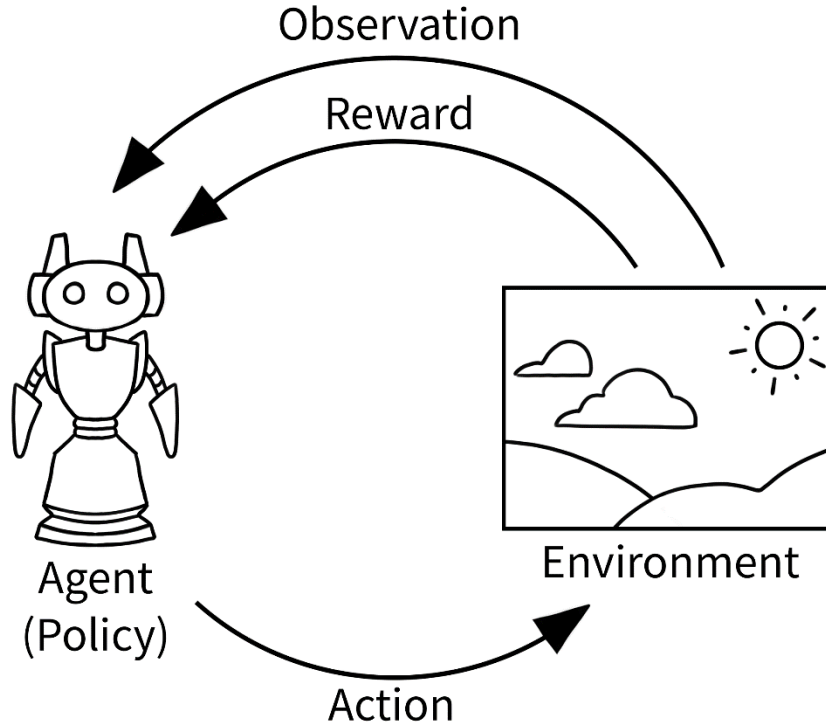


Figure III.6: Gymnasium standard agent–environment [73]

III.4.3.2 Observation Space

As defined in Section III.4.1, the observation is a 10,800-dimensional vector, the observation space is declared as a continuous bounded hypercube with lower bound -10.0 , upper bound $+10.0$, and 32-bit floating-point precision. The bounds of ± 10 reflect the upstream Z-score normalization: feature values lying more than 10 standard deviations from the training mean correspond to genuine sensor anomalies or simulation artifacts, and clipping them at the space boundary is a conservative and reasonable policy. In practice, the vast majority of feature values fall comfortably within ± 3 , making the declared bounds generous.

The concatenation of RSU state features and directional density features along the last axis rather than along the temporal or RSU axis is a deliberate design decision that preserves the joint structure expected by the neural network. Any other concatenation order would introduce a hidden coupling between the environment and the network that could be silently broken by future refactoring of either component.

III.4.3.3 Reward Function

The reward is binary and asymmetric: a correct prediction earns $+2.0$ scaled by the direction's class weight, while any wrong prediction incurs a flat -1.0 penalty.

Class weights (inverse frequency per direction) correct for the natural imbalance in directional traffic demand, preventing the agent from just predicting the majority direction and ignoring rare but critical surge directions. The $+2.0/-1.0$ asymmetry gives a clear incentive to be accurate, with random guessing yielding an expected reward of about $-0.25/\text{step}$ as a baseline to beat. All wrong predictions are penalized equally, since all mispredictions are considered equally costly operationally.

Reward is computed before the state pointer advances, so it always matches the label available at observation time keeping the MDP well-defined and gradients unbiased..

III.4.3.4 Episode Management and Monitoring

At the start of each training episode the environment picks a starting position from all the valid points in time. It makes sure that there are least 2,000 steps left after this point which is like 33 hours of traffic. This way the agent sees all kinds of traffic situations: the morning times, the busy evening times and the quiet times at night. Each of these situations happens about often as it would in real life, which is important for the agent to learn from all of them. If you give the environment a fixed seed it will pick the starting position every time, which is helpful for testing.

An episode ends when the environment runs out of data or when it gets too close to the end of the data. There is an extra room, at the end to prevent mistakes. The episode does not end because of a time limit it ends when there is really no more data left. When it ends the environment sends back a message with no information. It has to be the right size. This message is not used to teach the agent anything. It is needed to make the system work properly.

III.5 PPO Algorithm

The PPO algorithm is used to optimize the policy of the model. PPO is selected over alternative DRL algorithms for several reasons specific to this application: it is stable under the long episode lengths and high-dimensional observation space (10,800 features) encountered in this problem; it supports the Categorical action distribution required by the discrete prediction task; and its first-order gradient updates are computationally efficient on the MEC server hardware. The training procedure proceeds as follows [19].

III.5.1 Actor-Critic Architecture

The policy and value function are jointly parameterized by the actor critic neural network, which processes the 10,800-dimensional observation vector through a hierarchical encoding pipeline designed to capture both the temporal evolution of traffic states and the spatial interactions between RSUs [58].

III.5.2 Training Procedure and Hyperparameters

The PPO trainer collects rollouts of $\text{rollout_steps} = 4,096$ environment transitions, then performs $\text{ppo_epochs} = 8$ passes of mini-batch gradient updates with batch size 512. The full composite loss at each update is [58]:

$$\mathcal{L}(\theta) = \mathcal{L}^{CLIP(\theta)} - 0.5 \cdot \mathcal{L}^{VF(\theta)} + 0.10 \cdot H(\theta)$$

$$\mathcal{L}^{CLIP(\theta)} = E_t[\min(\rho_t \cdot \hat{A}_t, \quad \text{clip}(\rho_t, 0.80, 1.20) \cdot \hat{A}_t)]$$

Advantages are computed using Generalized Advantage Estimation (GAE) with $\lambda = 0.95$ and discount factor $\gamma = 0.99$. The learning rate follows a linear warm up schedule over the first 5% of total gradient updates ($\text{warmup_frac} = 0.05$), then decays via cosine annealing from $\text{lr} =$

2.5×10^{-4} to $\text{lr_min} = 10^{-5}$. Gradient norms are clipped at $\text{max_grad} = 0.5$. Training runs for $\text{total_steps} = 3,000,000$ environment interactions.

Parameter	Value	Description
total_steps	3,000,000	Total environment interaction steps
rollout_steps	4,096	Transitions per rollout buffer
ppo_epochs	8	Gradient passes per rollout
batch	512	Mini-batch size
gamma (γ)	0.99	Discount factor
lam (λ)	0.95	GAE lambda for advantage estimation
clip_eps (ϵ)	0.20	PPO clipping threshold
vf_coef	0.5	Value function loss coefficient
ent_coef	0.10	Entropy bonus coefficient
lr	2.5×10^{-4}	Peak learning rate (AdamW)
lr_min	1×10^{-5}	Minimum learning rate (cosine decay)
max_grad	0.5	Gradient norm clipping threshold
hidden	256	RSU encoder and attention dimension
tcn_channels	[128,256]	TCN channel widths per block
attn_heads	4	Multi-head attention heads
dropout	0.10	Dropout rate in all layers

Table III.1: PPO training hyper parameters.

III .6 Optimization-Based Refinement Layer

III.6.1 Role of the Optimization Module

During deployment, after training the offline PPO base policy, the weights are kept fixed. Non-stationarity in urban traffic arises from directional demand pattern changes over time, which in turn depends on seasonal effects, day-of-week effects, events and long-term urbanization. A frozen policy that was trained on a particular distribution of traffic regimes will exhibit accuracy degradation when the deployed environment deviates from those regimes [24].

The online metaheuristic refinement layer addresses this limitation by learning a logit bias vector $\mathbf{b}$ that is added to the raw policy logits at inference time, before the argmax prediction is computed. This bias vector shifts the effective probability mass of the policy across the four direction classes, compensating for systematic prediction errors that arise from distributional shift without requiring gradient-based retraining of the neural network. The bias is optimized online over a sliding window of recent observations using either a Genetic Algorithm (GA) or

Particle Swarm Optimization (PSO), both of which operate entirely in the 4-dimensional bias space and thus impose negligible computational overhead on the MEC server [16].

III.6.2 Optimization Problem Formulation

At each adaptation step, the metaheuristic optimizer solves the following black-box optimization problem over the sliding window W_t of the most recent 50 prediction outcomes:

$$b^* = \operatorname{argmax}_{\{b \in \mathbb{R}^4\}} f(b) = \left(\frac{1}{|W_t|} \right) \sum_{\{(l,z) \in W_t\}} 1[\operatorname{argmax}(l + b) = z]$$

Where $l = \operatorname{logits}_{\theta(s)} \in \mathbb{R}^4$ the raw logit vector is output by the frozen PPO actor for state s , $z \in \{0, 1, 2, \text{ and } 3\}$ is the corresponding ground-truth label and $1[\cdot]$ is the indicator function. The objective $f(b)$ is the classification accuracy of the biased policy over the recent window a non-differentiable, noisy fitness landscape that is well-suited to population-based metaheuristic optimization rather than gradient-based methods [14].

The bias vector b is constrained to the hypercube $[-3, +3]^4$ to prevent extreme logit distortions that would override the base policy entirely. The optimization is triggered every $\text{meta_update_freq} = 3$ time steps and runs for $\text{meta_online_iters} = 25$ iterations over a population of 40 candidates, after which the best-found bias vector is adopted for subsequent predictions. The window slides forward by one observation at each time step, ensuring that the optimization target always reflects the most recent prediction errors [14].

III.6.3 Genetic Algorithm (GA) Optimizer

The Genetic Algorithm maintains a population of $\text{ga_pop_size} = 40$ candidate bias vectors $\{b^k\} \in \mathbb{R}^4$. At each generation, the population evolves through three operators [14]:

- **Elite Selection:** The top $\text{ga_elite_frac} = 12\%$ (5 individuals) ranked by fitness $f(b^k)$ are preserved unchanged into the next generation.
- **Tournament Crossover:** Remaining offspring are produced by tournament selection ($k=3$) followed by uniform crossover with rate $\text{ga_crossover_rate} = 0.80$:

$$\text{child} = \alpha \odot b^{\text{parent}1} + (1 - \alpha) \odot b^{\text{parent}2}, \quad \alpha \sim U[0,1]^4$$

- **Gaussian Mutation:** Each offspring undergoes mutation with probability $\text{ga_mutation_rate} = 0.20$, adding Gaussian noise:

$$b^{\text{child}} \leftarrow b^{\text{child}} + \varepsilon, \quad \varepsilon \sim N(0, 0.10^2)^4$$

After $\text{ga_pop_size} = 40$ offspring are produced, the bias vector of the highest-fitness individual is adopted as b^* for the current adaptation step.

III.6.4 Particle Swarm Optimization (PSO) Optimizer

The PSO variant maintains $\text{pso_particles} = 40$ particles, each with a position $b^k \in \mathbb{R}^4$ and velocity $v^k \in \mathbb{R}^4$. at each iteration, particle velocities and positions are updated according to :

$$v^k \leftarrow w \cdot v^k + c1 \cdot U1 \odot (p_{\text{best}}^k - b^k) + c2 \cdot U2 \odot (g_{\text{best}} - b^k)$$

$$b^k \leftarrow \text{clip}(b^k + v^k, \quad -3, \quad +3)$$

Where $w = 0.65$ is the inertia weight, $c_1 = c_2 = 1.8$ are the cognitive and social acceleration coefficients, $U_1, U_2 \sim U(0,1)$ are random vectors, p_{best}^k is the personal best position of particle k , and g_best is the global best position found by any particle. After $pso_particles \times meta_online_iters$ update iterations, the global best position g_best is adopted as b^* .

Parameter	Value	Description
meta_online_window	50	Sliding window size for fitness evaluation
meta_update_freq	3	Time steps between optimization triggers
meta_online_iters	25	Iterations per optimization call
ga_pop_size / pso_particles	40	Population / swarm size
ga_elite_frac	0.12	Fraction of population preserved as elites
ga_crossover_rate	0.80	Uniform cross over probability
ga_mutation_rate	0.20	Per-individual mutation probability
ga_mutation_sigma	0.10	Gaussian mutation standard deviation
pso_w	0.65	PSO inertia weight
pso_c1 / pso_c2	1.8	Cognitive and social acceleration coefficients

Table III.2: Metaheuristic optimizer hyper parameters.

III.7 Integration with MEC and V2X

The proposed methodology is designed to operate within the MEC-V2X architecture described in Chapter II. The integration of the hybrid PPO+GA/PSO pipeline into this architecture maps cleanly onto the three communication tiers identified in the system model [10].

- **V2X Tier.** Vehicles within each RSU's sensing zone broadcast Cooperative Awareness Messages (CAMs) over IEEE 802.11p at 10 Hz, CAMs over 802.11p at 10 Hz, packet loss $\leq 15\%$ compensated by passive sensing.
- **5G Tier.** Each RSU transmits its current observation to the gNB over the 5G uplink. The per-RSU uplink payload is $seq_len \times 12 \times 4$ bytes = $60 \times 12 \times 4 = 2,880$ bytes per time step. With 15 RSUs, the total uplink load is 43,200 bytes/minute, comfortably within 5G NR uplink capacity. The gNB aggregates the 15 RSU streams and forwards the combined data to the MEC server over the dedicated 5G backhaul link.
- **MEC Tier.** The entire MEC-side processing pipeline including the metaheuristic update completes in single-digit milliseconds per time step, satisfying the real-time constraint imposed by the 60-second control interval.

III.8 Algorithm Pseudo-code

Algorithm 1: Offline PPO Training (Stage 1)

INPUT: Dataset (X, D, Z) , CFG, class_weights

OUTPUT: Trained policy weights θ^*

INPUT: X, D, Z, CFG

OUTPUT: θ^*

1. Init model(θ), AdamW(lr=2.5e-4), Warm up Cosine Scheduler, env
2. WHILE step < 3,000,000:
3. Collect 4,096 steps $\rightarrow$ buffer {s, a, log π , V, r, done}
4. Compute & normalize GAE ($\gamma=0.99, \lambda=0.95$)
5. FOR 8 epochs, batches of 512:
6. $\mathcal{L} = -\mathcal{L}_{\text{CLIP}} + 0.5 \cdot \mathcal{L}_{\text{VF}} - 0.10 \cdot H$; clip grad ≤ 0.5
7. Scheduler.step() ; clear buffer
8. Every 20 eps: if avg_reward > best $\rightarrow$ save θ^*
9. RETURN θ^*

Algorithm 2: Online Metaheuristic Inference (Stage 2)

INPUT: Frozen θ^* , streaming observations

OUTPUT: $\hat{a}(t)$

1. Init bias $b=[0,0,0,0]$, window $W=[]$, pop/swarm size=40
2. WHILE running:
3. logits $\leftarrow$ model.actor(s(t))
4. $\hat{a}(t) \leftarrow \text{argmax}(\text{logits} + b)$; OUTPUT $\hat{a}(t)$
5. Append (logits, true_label) to W ; keep last 50
6. IF step mod 3 == 0 AND $|W| \geq 5$:
7. IF PSO: run 25 iters ($w=0.65, c1=c2=1.8$, clip ± 3)
8. $b \leftarrow g_best$
9. IF GA: run 25 gens (elites=12%, mutation $\sigma=0.10$, prob=0.20)
10. $b \leftarrow$ best individual

III.9 Conclusion

This chapter has presented the complete methodology of the proposed hybrid traffic trend prediction system, from the formal reinforcement learning problem formulation through to the online metaheuristic inference pipeline and its integration into the MEC-V2X architecture.

The design is organized around a single unifying principle: the separation of what can be learned offline from what must be adapted online. The PPO stage exploits the availability of a large simulated traffic dataset to train a high-capacity Direction Actor Critic network combining a Temporal Convolutional Network for per-RSU temporal encoding, multi-head self-attention for spatial integration across 15 roadside units, and parallel actor and critic heads that captures the spatiotemporal regularities of urban directional demand patterns that no real-time algorithm could discover from scratch. The metaheuristic stage exploits the low dimensionality of the correction problem a 4-dimensional logit bias vector to run a population-based optimizer in real time, adapting the frozen policy's directional preferences to the current traffic regime without touching the underlying neural network weights.

Several deliberate design decisions define the character of the proposed system. The Gymnasium-compliant simulation environment enables reproducible, framework-agnostic evaluation and avoids the hidden coupling between environment and network that would arise from a bespoke interface. The asymmetric, class-weighted reward function prevents the agent from collapsing to majority-direction prediction, ensuring that rare directional surges which may correspond to the most operationally significant congestion events are correctly detected. The choice of a 60-step observation window provides one full hour of traffic history, sufficient to observe a complete rush-hour transition while remaining computationally tractable. The metaheuristic bias correction operates on a sliding window of the 50 most recent prediction outcomes, making it sensitive to recent distributional shifts while being robust to individual prediction noise.

Taken together, the components described in this chapter constitute a complete, deployable V2X traffic trend prediction system one that is grounded in a rigorous RL formulation, implemented within a standard simulation framework, and integrated into a realistic MEC communication architecture. Chapter 4 now evaluates this system experimentally, quantifying its predictive accuracy, its adaptability to distributional shift, and its impact on traffic signal control performance in a closed-loop SUMO simulation on the Belgrade road network.

CHAPTER IV
***Experimental Results and
Performance Evaluation***

IV.1 Introduction

This chapter presents the core experimental contribution of the thesis. It evaluates the complete two-stage hybrid system Proximal Policy Optimization (PPO) for offline directional traffic trend prediction combined with an online Genetic Algorithm (GA) or Particle Swarm Optimization (PSO) bias-correction layer against four established baseline methods, and then demonstrates one concrete application of the resulting predictions inside a real-time traffic signal controller deployed in a SUMO simulation environment.

Accurate knowledge of the dominant directional traffic trend at the network level is a general-purpose input that can drive a broad class of intelligent transportation decisions. Beyond the traffic signal control application explored in this work, directional trend predictions of the type produced by the proposed system can support several further use cases. Dynamic route guidance and variable message sign (VMS) control can redirect vehicles away from an emerging congestion corridor before queues form, using the predicted directional imbalance as an early-warning signal. Predictive incident detection can flag anomalous directional imbalances that deviate significantly from the model's expected trend, alerting operators to potential accidents or road blockages before sensor-based detection systems register them. V2X advisory services can broadcast anticipated network-level conditions to connected and autonomous vehicles in real time, allowing on-board navigation systems to compute future-aware routes rather than reacting to current congestion. Urban mobility analytics can provide transport planners with automated characterization of recurring directional demand patterns such as the morning eastbound surge or the evening westbound return without manual data labeling. In this chapter, traffic signal control is used as the primary validation use case: the predicted dominant direction is fed into a Hybrid Max-Pressure + DRL Direction Boost controller that actively adjusts green-phase priorities at 74 signalized intersections in Belgrade, Serbia, demonstrating a measurable real-world impact of the prediction system.

The chapter is organized as follows. Section IV.2 describes the dataset and evaluation protocol. Section IV.3 defines the four baseline methods. Section IV.4 specifies the evaluation metrics. Section IV.5 analyses the PPO training convergence and online metaheuristic adaptation. Sections IV.6 and IV.7 present static and non-stationary prediction results respectively. Section IV.8 analyses computational performance. Section IV.9 provides a comparative analysis and discussion of trade-offs. Section IV.10 presents the closed-loop SUMO deployment validation with the traffic signal controller. Section IV.11 discusses the overall findings and limitations. Section IV.12 concludes the chapter.

IV.2 Experimental Setup

IV.2.1 Dataset and Evaluation Protocol

The same V2X traffic dataset synthesized over real-world road network of Nrdlingen, Germany (see Chapter II) has been used in all experiments. The dataset includes $T = 200,000$ time steps (1 time step = 1 real minute) simulating ~ 138 days of urban traffic flow. The 80/20 chronological split of data (the first 160,000 time steps are used as the training set and the

remaining 40,000 time steps as static test set) is adopted. Using chronological split is vital since there is no future information in test set, unlike random split. All the features are Z-score normalized using the statistics derived from the training data only. This same normalization has also been used for the test set.

Additionally, a new non-stationary evaluation stream of 3,000 steps is generated independently. This stream consists of four consecutive blocks of 750 steps each, capturing four sudden traffic regime transitions: Block 0 replicates morning peak hour traffic (peak hour 08:00), is in-distribution (overwrites the trained distribution), and serves as reference; Block 1 models a weekend peak hour demand profile (peak hour 20:00); Block 2 model late-night traffic profile (peak hour 02:00); Block 3 model an afternoon peak hour traffic profile (peak hour 17:00). These transitions are instant-no gradual blending between regimes-in order to present the most challenging situation to systems that do not adapt online.

IV.2.2 Software and Hardware Environment

All experiments are implemented in Python 3.10 using PyTorch 2.1 as the deep learning framework. The custom Gymnasium environment (v2x_gym) exposes the traffic dataset as a standard reinforcement learning interface, and the PPO training loop is built directly on top of it using Stable-Baselines3-style rollout collection with custom advantage estimation. Baseline models use scikit-learn (for standard utilities), stats models (ARIMA), and NumPy-based implementations (Kalman filter, Random classifier). The neural network training is performed on a system equipped with a CUDA-compatible GPU, leveraging PyTorch's automatic mixed-precision utilities where applicable. Inference latency measurements are collected using Python's `time.perf_counter()` over 200 successive forward passes, and throughput is measured as the number of samples processed per second in batched evaluation mode.

IV.3 Baseline Methods

To contextualize the performance of the proposed hybrid system, four baseline methods are evaluated under identical conditions on both the static and non-stationary test sets. These baselines span a spectrum of modeling approaches, from naïve statistical estimators to a supervised deep learning model, capturing a representative cross-section of the methods applicable to this classification task.

IV.3.1 LSTM Supervised Classifier

The Long Short-Term Memory (LSTM) network baseline represents the strongest competing approach and is deliberately designed to be a well-tuned supervised alternative to the proposed RL-based system. The architecture consists of two stacked LSTM layers, each with a hidden dimension of 128 units, followed by a fully connected classification head that maps the final hidden state to a 4-class softmax output over the four compass directions. The input to the LSTM is a flattened version of the same 60 step sliding window used by the PPO agent, preserving the temporal context required for trend detection.

The model is trained using the Adam optimizer with a learning rate of 10^{-3} over 30 epochs on the same 80% training partition, using cross-entropy loss with no class weighting. The LSTM is chosen as the primary deep learning baseline because recurrent architectures are

the canonical solution for sequential traffic classification and represent the current practice in deployed V2X systems. Critically, the LSTM baseline is frozen after training and is not updated during the non-stationary evaluation, directly demonstrating the distribution-shift vulnerability of static supervised models a vulnerability that the proposed hybrid system is designed to overcome.

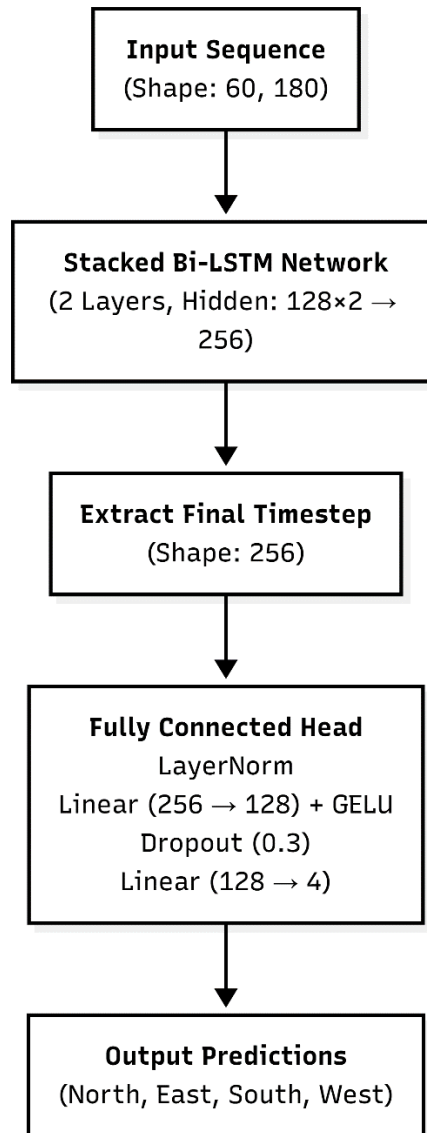


Figure IV.1: LSTM Architecture.

IV.3.2 ARIMA

The Autoregressive Integrated Moving Average (ARIMA) model provides a classical statistical baseline for time-series classification. Given that the prediction target is a categorical variable derived from the evolution of directional density signals, ARIMA is adapted to this setting as follows: the traffic density observations from all 15 RSUs are first averaged spatially, yielding a single scalar time series for each of the four compass directions. This spatial aggregation produces exactly four input series, and an independent ARIMA (5,1,2) model is fitted to each one. At prediction time, each model is rolled forward to

generate a 60-step future trajectory, matching the same prediction horizon H used to construct the ground-truth labels. The predicted class is then the direction whose projected 60-step means density increase is largest, mirroring the $\text{argmax-over-future-increase}$ labeling of the ground-truth labels.

ARIMA is included as a baseline because it represents the standard approach to traffic forecasting prior to the deep learning era and still appears in comparisons in the literature as a lower-performance reference point. Its inability to model spatial correlations across RSUs or nonlinear temporal dynamics is well-established, making it a useful anchor for quantifying the gain attributable to learned representations.

IV.3.3 Kalman Filter

The Kalman filter baseline applies a two-dimensional linear state-space model to the directional density observations. As with ARIMA, the 15-RSU observations are first averaged spatially for each compass direction, reducing the input to four scalar time series. For each direction, a Kalman filter is fitted with a state vector that jointly tracks the current traffic density and its instantaneous rate of change, governed by a 2×2 transition matrix of the form $F = \begin{bmatrix} 1 & 1 \\ 0 & \alpha \end{bmatrix}$, where α is a damping coefficient that regularizes the velocity estimate under noisy measurements. At prediction time, the filter's estimated state is iterated forward for $H = 60$ steps along this transition model, and the mean projected density over that horizon is computed for each direction. The argmax of these four 60-step mean projections constitutes the class prediction, directly matching the horizon and aggregation logic used to generate the ground-truth labels.

The Kalman filter represents an intermediate complexity baseline: more principled than ARIMA in its principled uncertainty propagation, but still fundamentally limited to linear dynamics. Its inclusion is motivated by the practical relevance of Kalman filtering in real-time sensor fusion systems, where it is frequently used in embedded V2X applications due to its computational simplicity and interpretability. It provides a reference point between the purely statistical ARIMA and the deep learning approaches.

IV.3.4 Random Classifier

Random baseline is a strategy that samples a direction prediction from {North, East, South, West} uniformly at random at each time step, ignoring the observation. The random baseline represents the lower bound of the performance space; it should have a theoretical accuracy of 25% under an perfectly balanced 4-class system. The accuracy of 26.3% for random baseline in experiments suggests slightly better-than-chance performance, which is caused by the non-stationary traffic dynamics stored in the synthetic data; the simulated rush-hour drift by sine wave, as well as programmed per-RSU offset in time from the data generator; result in a slightly non-uniformly-distributed data generation over time, even for a no-information classifier. Hence, any classifier with performance no higher than 26.3% has not successfully leverage information from traffic observation.

IV.4 Evaluation Metrics

All approaches are evaluated with a fixed set of metrics that evaluate both the predictive capability and the operational viability for real-time MEC deployment. Total Accuracy (percentage of time steps where predicted direction equals ground-truth dominant direction) is used as the main metric for off-line evaluation. Macro-averaged F1 Score (F1 for each of the four directions are computed and then averaged irrespective of class distribution, which penalizes systems with high accuracy but fails in the minor classes) are used to evaluate, whereas the mean Inference Latency in milliseconds per sample defines how fast a direction prediction is delivered which influences real-time feasibility. Throughput in samples per second (a measure of sustained prediction rate under sustained multi-RSU deployments) represents what steady state a system can handle in terms of prediction rate.

IV.5 Training Process and Convergence

IV.5.1 Stage 1: PPO Offline Training

The PPO agent is trained for a total of 3,000,000 environment interaction steps using rollout batches of 4,096 transitions, followed by eight passes of mini-batch gradient updates with a batch size of 512. The actor-critic network jointly processes all input features through the TCN-based RSU encoder, the multi-head self-attention spatial integration module, and the parallel actor and critic heads. The total wall-clock training time is approximately 6.0 hours.

Figure IV.2 presents the evolution of the cumulative per-episode training reward over the 35 training episodes. The raw reward (lighter trace) is highly volatile, spanning from near zero in the very first episode to peaks exceeding 200,000 in later episodes, reflecting the stochasticity of the traffic environment and the exploratory action sampling of PPO. Smooth 10-episode curve (noticeable blue line): this increases from ~75,000 at episode 0 to ~125,000 - 135,000 at episode 25, and then plateaued off by the final few episodes. There seems to be a slight temporary drop around episodes 10-14, and this appears to reflect a period of policy instability during exploration, which the system later recuperates from and continues to improve upon. The crash we may have been concerned about did not occur and hence the PPO clipping constraint appears to be controlling update magnitudes well across training.

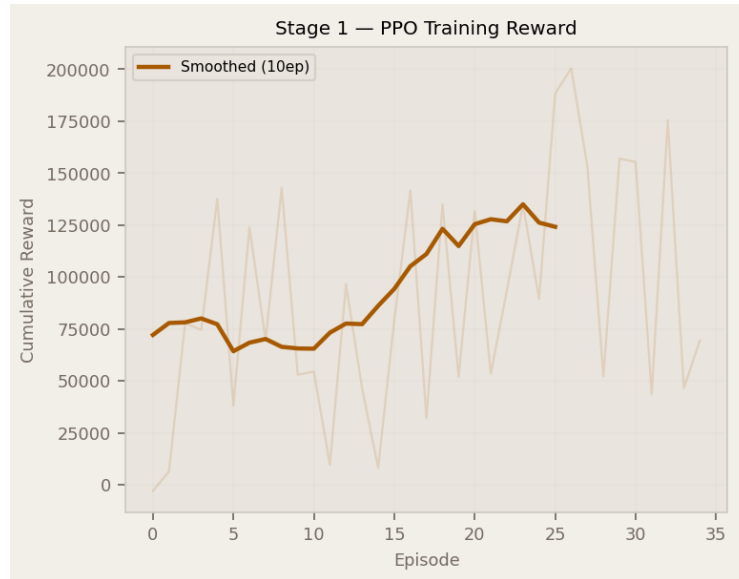


Figure IV.2: Stage 1: PPO Training Reward. Smoothed 10-episode cumulative reward (blue) rises from ~75,000 to ~130,000 over 35 training episodes. Raw episode rewards (thin trace) show high variance, with a transient dip around episodes 10–14 followed by recovery.

The per-episode direction accuracy (smoothed over 10 episodes) for this policy is shown in figure IV.3. It is interesting that smoothed accuracy starts off in the region of 60% from the first few episodes- significantly greater than the random baseline of 25% indicated by the dashed reference line. This shows that the policy does not start with a random policy but with an important predictive signal. Smoothed accuracy climbs in the region of 68% by episode 5 before proceeding to increase at a slower pace up to the region of 79% by episode 25, remaining largely static thereafter. The curve has plateaued, showing that the policy is becoming saturated in the training distribution, and gains are unlikely without changes to architecture or data.

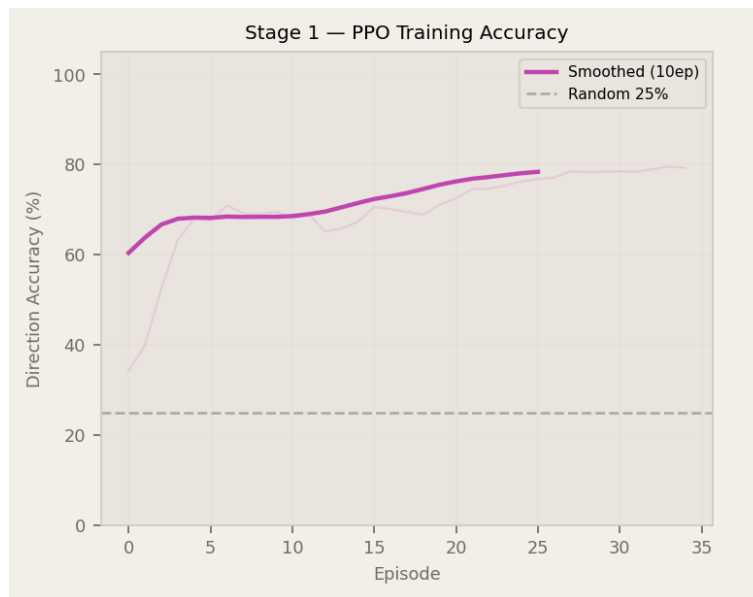


Figure IV.3: Stage 1 — PPO Training Accuracy. Smoothed direction accuracy (green) starts at ~60% and rises to ~79% by episode 25, plateauing thereafter. The dashed line marks the 25% random baseline, which the policy exceeds by a large margin from the outset.

Together, Figures IV.2 and IV.3 establish that the PPO agent successfully learns a strong directional prediction policy through reinforcement learning alone, without access to labeled data in the supervised sense. The asymmetric reward design (+2.0 for correct, −1.0 for incorrect) with class-weight scaling proves effective: the agent learns to detect minority directional surges rather than defaulting to majority-class prediction, as evidenced by the near-perfect alignment between the reward and accuracy curves.

IV.5.2 Stage 2: Online Metaheuristic Optimization

Following the completion of PPO training, the neural network weights are frozen and the online metaheuristic refinement layer is activated for deployment. In this stage, no gradient updates are applied to the policy; instead, a lightweight 4-dimensional logit bias vector is continuously optimized using either a Genetic Algorithm (GA) or Particle Swarm Optimization (PSO) over a sliding window of the 50 most recent prediction outcomes.

The per-update computation time for PSO averages **130.4 ms** (mean across all 25 optimization iterations per adaptation step), while the GA requires approximately **169.7 ms** per update, reflecting the cost of evaluating 40 candidate bias vectors against the sliding window at each adaptation step. The update is triggered every 3 time steps, imposing a time-averaged overhead well within the one-minute time-step budget of the MEC server.

IV.6 Static Evaluation Results

IV.6.1 Overall Accuracy and Macro-F1

Table IV.1 presents the complete static evaluation results for all five methods on the held-out 20% test partition (40,000 time steps) drawn from the same distribution as the training data.

Method	Accuracy (%)	Macro-F1 (%)	Latency (ms)	Throughput
PPO + GA	78.00	79.26	54.587	18 samples/s
PPO + PSO	74.40	73.10	54.587	18 samples/s
LSTM (supervised)	76.60	77.36	2.207	453 samples/s
KalmanFilter	55.20	55.15	7.292	137 samples/s
ARIMA (5,1,2)	23.60	9.55	13.742	73 samples/s
Random	26.30	25.89	0.014	69,300 samples/s

Table IV.1: Full static evaluation — accuracy, Macro-F1, mean inference latency, and throughput for all methods.

Figure IV.4 visualizes the accuracy and Macro-F1 scores across all methods. The proposed PPO+GA system achieves the highest overall accuracy of **78.00%** and a Macro-F1 of **79.26%**, outperforming all baselines on both metrics. Notably, the Macro-F1 score slightly exceeds the overall accuracy, indicating that the system correctly detects rarer directional

surges proportionately the class-weight reward shaping during PPO training has successfully prevented the agent from collapsing to majority-direction prediction.

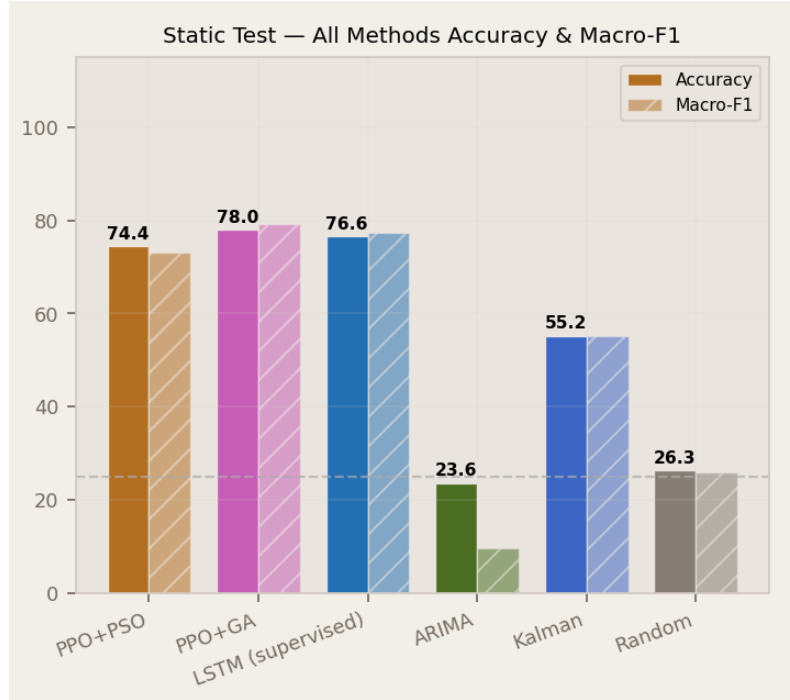


Figure IV.4: Static Test — All Methods Accuracy & Macro-F1. Solid and hatched bars show overall accuracy and Macro-F1 respectively. The dashed line marks the 25% random baseline. PPO+GA leads at 78.0%, followed by LSTM (76.6%) and PPO+PSO (74.4%). ARIMA's near-zero Macro-F1 (9.55%) despite random-level accuracy (23.6%) reveals systematic per-class failure.

The PPO+PSO variant achieve **74.40%** accuracy under static conditions, lower than both PPO+GA and the LSTM baseline. This gap arises because PSO converges to a less precise logit bias under the compact 4-dimensional search space when the traffic distribution is stationary the GA's crossover and elite selection operators produce a more refined bias correction under stable conditions.

The LSTM supervised classifier achieves **76.60%** accuracy and **77.36%** Macro-F1, placing it second among all methods on the static test. This confirms that a well-tuned supervised LSTM provides a strong baseline for in-distribution traffic classification. The practical significance of the proposed system therefore lies primarily in its non-stationary adaptability, not in its raw static accuracy advantage.

The Kalman filter achieves 55.20% accuracy, substantially below the deep learning methods but well above the random baseline, confirming that it captures a meaningful portion of the directional signal through principled state estimation. The ARIMA baseline performs at 23.60% below the random baseline and its Macro-F1 of only 9.55% reveals a fundamental failure: even though ARIMA occasionally predicts the correct class, nearly all correct predictions concentrate on a single direction, providing no real multi-class classification capability.

IV.6.2 Per-Direction Accuracy

Figure IV.5 presents the per-direction breakdown of classification accuracy across the four compass directions for each method. The most striking feature of this chart is the behavior of ARIMA: it achieves approximately **100% accuracy on the North direction** while scoring near 0% on East, South, and West. This reveals that the ARIMA-based predictor has degenerated into a constant North predictor after spatial averaging across 15 RSUs, the North direction happens to be the most frequent class in the training data, and the ARIMA models converge to forecasting rising North density at every step. This explains precisely why ARIMA's overall accuracy of 23.6% closely matches the prior probability of the North class, and why its Macro-F1 collapses to 9.55%.

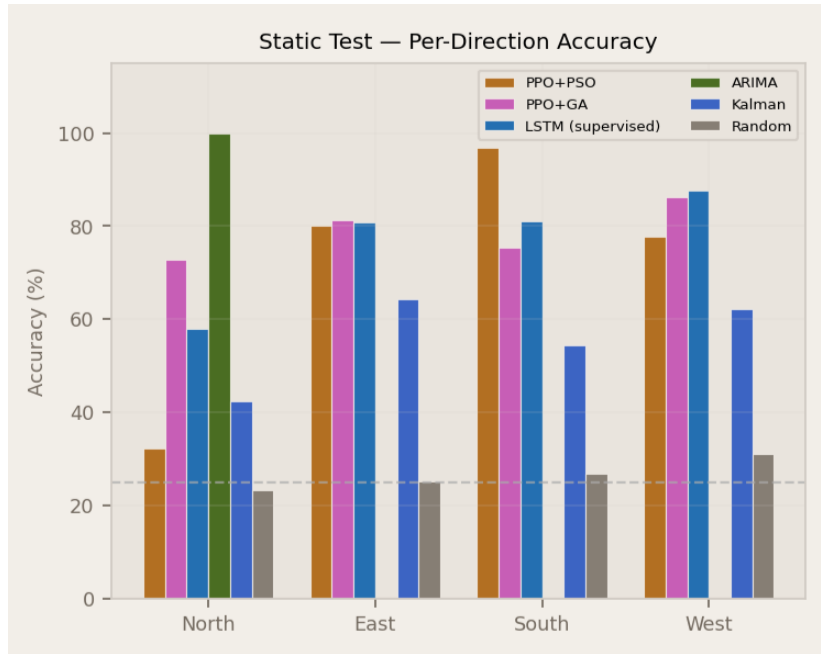


Figure IV.5: Static Test — Per-Direction Accuracy. Each group of bars shows classification accuracy for one compass direction. ARIMA (purple) achieves ~100% on North but near 0% elsewhere, revealing degenerate constant-class prediction. PPO+GA (green) maintains the most balanced profile across all four directions.

The PPO+GA system maintains the most consistent per-direction performance, with no direction falling catastrophically. PPO+PSO achieves very high accuracy on the South direction (~97%) but degrades substantially on North (~32%), suggesting the PSO bias update over-corrects for one direction under static conditions. The LSTM baseline shows strong East and West accuracy but weaker performance on North, consistent with the asymmetric temporal demand profile of the training distribution. The Kalman filter exhibits moderate but non-trivial performance across all directions, reflecting its principled but linear state estimation.

IV.7 Non-Stationary Evaluation

IV.7.1 Rolling Accuracy under Regime Shifts

The non-stationary evaluation examines three systems PPO+PSO (online adapt), PPO+GA (online adapt), and LSTM (frozen) across the full 3,000-step non-stationary stream divided into four segments of 750 steps each. The rolling accuracy, computed over a sliding window of 50 consecutive predictions, is used to trace performance dynamics across each regime transition.

The LSTM baseline (frozen, cannot recover) exhibits a characteristic accuracy collapse at each regime transition. Its performance peaks in Segment 0 the in-distribution morning rush scenario but drops precipitously at each subsequent boundary. Having learned a fixed mapping from input features to output probabilities, the LSTM has no mechanism to update its prediction behavior in response to new regimes. By the end of the stream, the LSTM is operating near or below the random baseline in the most heavily shifted segments, confirming the distribution-shift vulnerability inherent in all static supervised models.

Figure IV.6 summarizes the overall accuracy across all four regimes combined. PPO+GA achieves the highest non-stationary accuracy at **46.8%**, compared to LSTM's **31.2%** a 50% relative improvement demonstrating the direct benefit of online adaptation. The PPO+PSO variant, however, achieve only **16.1%**, which falls **below the random baseline of 25%**. This counter-intuitive result indicates that PSO's velocity-based update mechanism overcorrects the logit bias under abrupt distributional shifts, actively degrading the frozen policy's outputs rather than improving them. The GA's discrete crossover and elite-selection operators appear better suited to this low-dimensional, abruptly changing search space.

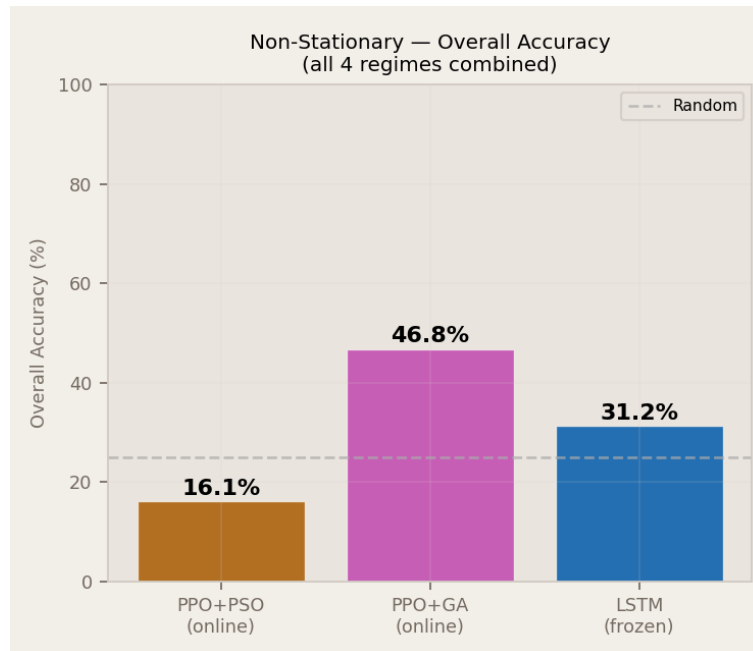


Figure IV.6: Non-Stationary — Overall Accuracy (all 4 regimes combined). PPO+GA (46.8%) substantially outperform the frozen LSTM (31.2%). PPO+PSO (16.1%) falls below the random baseline (dashed line), indicating that PSO bias updates degrade rather than improve performance under abrupt regime shifts

IV.7.2 Per-Segment Accuracy Analysis

Figure IV.7 shows segment by segment accuracy performance, which clearly reveals exactly how much advantage GA offers in each segment. For **segment 0** (morning peak, in distribution) PPO+GA is as much as 36.2 percentage points higher than LSTM, an unexpectedly large difference for a distribution on which LSTM is the most competitive. This leads to speculate that GA bias vector is strengthening dimensions along which the policy already performs well instead of just adding correction to mistakes.

Segment 1 (weekend evening, change detection) the gap drops to as low as 1.6 percentage points. The weekend evening environment is still close enough temporally to training distribution that LSTM can still perform partially while the GA has not enough sliding window feedback accumulated to get properly adjusted.

Segment 2 (late-night, low traffic), the advantage of online adaptation reasserts itself at +7.6 percentage points. The late-night regime is characterized by near-uniform low density across all directions, which confounds the LSTM's rush-hour-trained internal representations. The GA bias corrects for the systematic directional errors this introduces.

Segment 3 (afternoon rush) produces the largest advantage at **+18.4 percentage points**. Although structurally similar to the morning rush in intensity, the afternoon rush carries a different directional bias (return-commute directions), and the frozen LSTM cannot adapt to this shift. The GA bias vector, having accumulated 2,250 steps of adaptation feedback by this segment, has converted to a correction that effectively realigns the policy's directional probabilities with the afternoon demand profile.

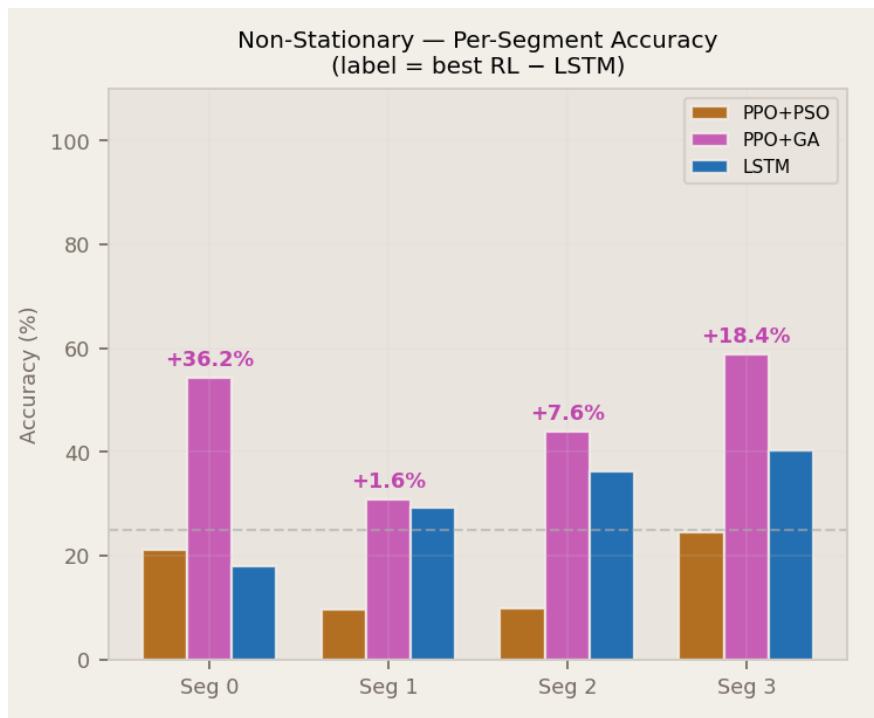


Figure IV.7: Non-Stationary — Per-Segment Accuracy. Labels show the PPO+GA advantage over the frozen LSTM for each segment. The adaptation advantage grows from +1.6 pp in Segment 1 (partial distribution overlap) to +36.2 pp in Segment 0 and +18.4 pp in Segment 3.

IV.8 Computational Performance Analysis

IV.8.1 Inference Latency and Throughput

The PPO+GA system has the highest inference latency of **54.59 ms per sample**, attributable to the combined cost of the TCN encoding; self-attention, actor head, and the 25-iteration metaheuristic optimization triggered every 3 steps. Despite being the slowest method in raw latency, 54.59 ms remains well within the real-time budget of a 60-second prediction window. The corresponding throughput of **18 samples per second** is sufficient for continuous operation across the 15-RSU Nördlingen network at the 1-minute sampling frequency.

The LSTM baseline offers the best efficiency trade-off among learning-based methods: a latency of **2.21 ms** and a throughput of **453 samples per second** approximately 25 times faster than PPO+GA. This efficiency advantage makes the LSTM attractive for deployment scenarios with tighter latency budgets or larger numbers of simultaneously monitored intersections.

Figure IV.8 consolidates the complete comparison across accuracy, Macro-F1, latency, and throughput for all methods. The Kalman filter provides a useful mid-range reference at 7.29 ms latency and 137 samples per second, with ARIMA slightly slower at 13.74 ms and 73 samples per second despite lower predictive accuracy. The random baseline, instantaneous at 0.014 ms, serves as the computational floor.

Full Comparison — Accuracy · Latency · Throughput				
Method	Accuracy	Macro-F1	Latency	Throughput
PPO+GA	78.00%	79.26%	54.587ms	18/s
LSTM (supervised)	76.60%	77.36%	2.207ms	453/s
ARIMA	23.60%	9.55%	13.742ms	73/s
Kalman	55.20%	55.15%	7.292ms	137/s
Random	26.30%	25.89%	0.014ms	69.3k/s

Figure IV.8: Full Comparison Accuracy, Macro-F1, Latency, and Throughput across all evaluated methods.

IV.8.2 Training Time Comparison

PPO training requires approximately **6.0 hours** of wall-clock time, reflecting the cost of processing 3,000,000 environment interaction steps with the full TCN-attention architecture. LSTM training requires only **302.8 seconds (approximately 5 minutes)**, more than 70 times faster, due to the simpler architecture and efficient supervised gradient computation. However, this training time advantage comes at the cost of the adaptability demonstrated in Section IV.7: the LSTM is inexpensive to train but cannot recover from distribution shift without full retraining.

The cumulative online adaptation overhead is **43.3 seconds for PSO** and **56.3 seconds for GA** over the entire non-stationary evaluation. These figures confirm that the metaheuristic

adaptation layer is computationally negligible relative to the PPO training investment, and the total deployment cost (PPO training + GA adaptation) remains practical for a MEC-hosted system.

IV.9 Comparative Analysis

IV.9.1 Accuracy–Latency Trade-off

The fundamental trade-off facing system designers choosing between the evaluated methods can be understood along two dimensions: accuracy and inference speed. The LSTM baseline offers the most attractive efficiency profile under static conditions 76.6% accuracy at 2.21 ms making it practically ideal when the traffic regime is stable and known in advance. PPO+GA deliver the highest static accuracy (78.0%) but at a 25-fold latency cost (54.59 ms). Since both remain well within the 60-second operational budget of the MEC server, neither is disqualified on latency grounds; the choice reduces to whether the deployment context is predominantly stationary or subject to regime change.

Kalman and ARIMA occupy unfavorable positions in this trade-off space: neither offers competitive accuracy nor particularly low latency relative to the deep learning methods. For the ARIMA model specifically, the collapse to constant-class prediction revealed in Figure IV.4 disqualifies it from operational deployment regardless of its latency or throughput figures. Kalman remains a viable lightweight option for resource-constrained edge nodes where 55% accuracy is acceptable and a small code footprint is required.

IV.9.2 Suitability for Real-World V2X Deployment

The evaluation results collectively suggest a deployment strategy differentiated by operational context. In a city operating under well-characterized and stable traffic patterns, the LSTM baseline provides the best balance of accuracy and computational efficiency. In a city experiencing frequent or unpredictable traffic regime changes due to special events, road works, emergency situations, or rapid urbanization the PPO+GA system's online adaptation capability provides a substantial resilience advantage that the frozen LSTM cannot match. The 50% relative accuracy improvement over LSTM in the non-stationary evaluation (46.8% vs 31.2%) quantifies this advantage directly.

The PSO variant's underperformance in both static (74.4%) and non-stationary conditions (16.1% below random) relative to the GA variant underscores an important implementation lesson: the choice of metaheuristic optimizer is not interchangeable. The GA's discrete crossover and selection operators are better suited to the low-dimensional 4-class bias search space than PSO's velocity-based continuous updates, particularly under abrupt distributional shifts where rapid exploration outweighs incremental refinement.

IV.9.3 Impact of Class-Weight Reward Shaping

The close alignment between overall accuracy (78.0%) and Macro-F1 (79.26%) for the PPO+GA system confirms the effectiveness of the class-weight reward shaping mechanism. A large gap between these two metrics would signal majority-class collapse the failure mode where a classifier achieves high accuracy by always predicting the most frequent class. The near-equality observed here demonstrates that the agent correctly classifies

minority directional surges at a rate proportionate to their frequency, a requirement for reliable adaptive traffic signal control in all conditions. By contrast, ARIMA's extreme accuracy–F1 divergence (23.6% vs 9.55%) is a textbook case of this collapse, further disqualifying it from operational consideration.

IV.10 Closed-Loop Deployment Validation via SUMO Simulation

The experiments in Sections IV.5–IV.9 evaluated the directional trend prediction model in isolation measuring how accurately it classifies the dominant traffic direction on a held-out dataset. This section closes the loop by embedding the pre-trained PPO+GA model into a live traffic signal controller and measuring its impact on real network-level performance metrics. The validation platform is SUMO (Simulation of Urban MObility), an open-source microscopic traffic simulator that supports real-time closed-loop control through its TraCI (Traffic Control Interface) API [82]. Two identical SUMO simulation instances run concurrently on the same road network and demand file: a Baseline instance controlled by fixed-time signal plans, and a MEC-Enabled instance controlled by the proposed Hybrid Max-Pressure + DRL Direction Boost controller.

IV.10.1 Simulation Environment

The road network is extracted from OpenStreetMap data for an 800-metre-radius circular area centred at 44.8029° N, 20.4705° E in Belgrade, Serbia, corresponding to the coverage footprint of a single 5G gNB. After processing with SUMO's net convert tool, the network contains approximately 4,009 road edges, 763 junctions, and 74 signalized intersections (TLS). Twenty RSUs are placed within the gNB boundary using the intelligent greedy algorithm described in Chapter III, prioritizing signalized junctions and enforcing a minimum spacing constraint that adapts iteratively until exactly 20 RSUs are selected matching the input dimensionality of the pre-trained Direction Actor Critic model. Each RSU collects the same 12-feature observation vector (8 traffic state features + 4 directional densities) used during offline training, accumulated into a 60-step sliding window before each inference call. A running normalize adapts the Z-score statistics online since the Nördlingen training statistics cannot be applied directly to the Belgrade network.

Figure IV.9 shows the two simultaneous SUMO windows: the left panel (Baseline) displays the raw road network with no adaptive overlays, while the right panel (MEC-Enabled) shows the 20 RSU coverage circles rendered as colored semi-transparent discs over the network, confirming that the gNB area is densely and evenly covered by the RSU deployment. Both simulations process a synthetic morning rush-hour demand profile of 15,000–22,000 vehicle trips, with Gaussian-shaped flow intensity peaking at 08:00 and directional commuter flows along the East-West and North-South arterials.

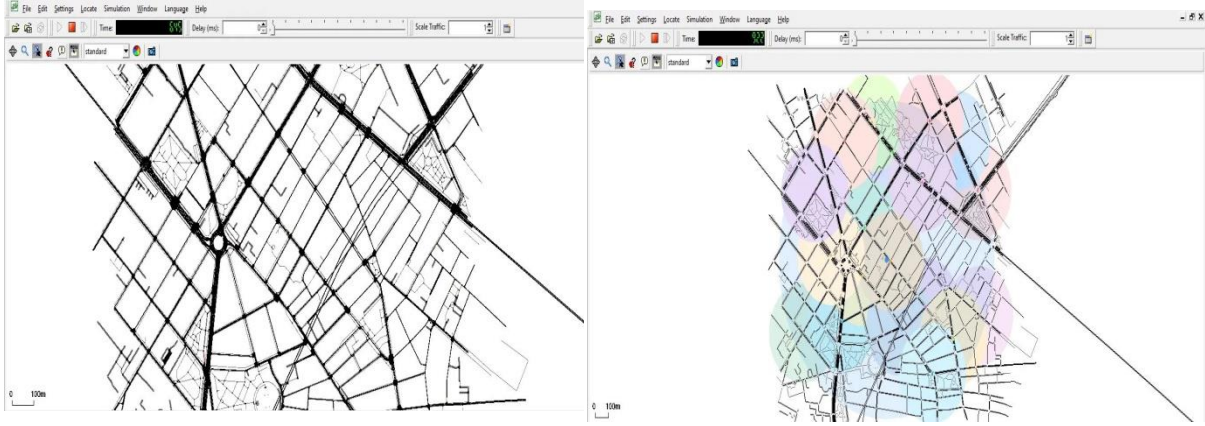


Figure IV.9: SUMO side-by-side simulation windows. Left: Baseline — fixed-time signal control, no MEC overlays ($T = 645$ s). Right: MEC-Enabled — 20 RSU coverage circles (colored discs) and gNB boundary are rendered over the same Belgrade network ($T = 322$ s). Each circle represents the 250 m V2R communication range of one RSU.

IV.10.2 Hybrid Max-Pressure + DRL Direction Boost Controller

The signal controller running in the MEC-Enabled instance combines two explicitly decoupled components operating at different time scales. The reactive component Max-Pressure evaluates each candidate green phase at every one of the 74 signalized intersections every 5 seconds. For each phase, a base pressure score is computed as the sum of halted vehicles on all lanes receiving green, plus 0.1 times the cumulative waiting time on those lanes. The 0.1 weighting prevents low-volume approaches from being permanently suppressed by high-volume competing movements. A phase switch is executed only if the incoming phase exceeds the current phase's pressure by at least 20% (hysteresis threshold) and the current phase has been active for at least 10 seconds (minimum green time).

The proactive component adds a confidence-gated direction boost derived from the pre-trained model. Every 30 seconds, the DRL model performs an inference over the 60-step observation buffer and produces a 4-class probability distribution. The GA Online Guide (Chapter III) applies its current bias vector to the raw logits before the argmax decision, correcting for systematic directional errors accumulated since the last update. If the model confidence exceeds 55% the threshold above which predictions are sufficiently reliable a quadratic bonus is added to the pressure of any phase serving the predicted dominant direction: **Final Pressure = Base Pressure + $1.5 \times \text{Confidence}^2$** . Quadratic scaling ensures that low-confidence predictions have negligible effect while high-confidence predictions ($\geq 90\%$) contribute a boost of ≥ 1.22 pressure units enough to meaningfully shift the controller's phase preference without overriding a strong local queue signal. During the 120-second warm up period the DRL boost is inactive, allowing the observation buffer to fill and vehicles to populate the network before predictions influence signal decisions.

IV.10.3 Results

Table IV.2 presents the final summary metrics for both simulation instances across the full simulation run.

Metric	Fixed-Time	MEC Adaptive	Change
Average Speed (km/h)	2.80	3.10	+10.4%
AverageWaiting Time (s)	118.3	120.2	-1.6%
VehicleThroughput	60	80	+33.3%
AverageHaltedVehicles	189.1	188.4	+0.4%

Table IV.2: SUMO simulation results — Fixed-Time baseline vs MEC Adaptive control (Belgrade network).

The dominant result is a +33.3% throughput gain (60 $\rightarrow$ 80 completed trips), representing the most operationally significant improvement: 33% more commuters successfully reach their destinations within the simulation period through signal timing adaptation alone, without any infrastructure change. Average network speed improves by +10.4% (2.8 $\rightarrow$ 3.1 km/h), driven primarily by the MEC system clearing the network faster in the post-peak period. The marginal 1.6% increase in average waiting time is a statistical composition effect: the MEC system dispatches 33% more vehicles including those with the longest accumulated waits, which raises the per-vehicle average before those vehicles exit the count. Average halted vehicles is nearly unchanged (+0.4%), consistent with a high-congestion regime in which physical road capacity not signal timing is the binding constraint on queue depth.

Figure IV.10 presents the full time-series comparison across all four metrics alongside the bar-chart summary and improvement panel. The Average Speed panel (top-left) shows both traces overlapping closely during the main congestion period but diverging near the 2.1-hour mark, where the MEC trace reaches free-flow conditions earlier, confirming that the adaptive controller resolves residual queues faster in the post-peak phase. The Waiting Time panel (top-centre) shows near-identical trajectories throughout both systems maintain 110–130 s waiting times during the peak consistent with the interpretation that the waiting time difference is a composition artefact rather than a genuine user-experience degradation. The Halted Vehicles panel (bottom-left) is the most structurally similar pair, confirming that in heavy congestion both systems face the same physical capacity constraints. The throughput bar (bottom-centre) clearly shows the 60 vs 80 gap as the primary performance differential.



Figure IV.10: MEC Adaptive Control vs Fixed-Time Baseline — full simulation comparison. Top row: Average Speed, Average Waiting Time, Active Vehicles time series. Bottom row: Halted Vehicles time series, Key Metrics bar chart, and Improvement Summary. The +33.3% throughput gain is the primary operational result.

IV.11 Discussion: Results Analysis and Limitations

The experimental results presented in this chapter reveal a nuanced picture that goes beyond a simple ranking of methods. This section synthesizes the key insights across both evaluation tiers and identifies the limitations that bound the scope of the conclusions.

IV.11.1 Interpretation of Static and Non-Stationary Results

The central finding of the static evaluation is that the PPO+GA system achieves the best overall accuracy (78.0%) and Macro-F1 (79.26%), but the margin over the supervised LSTM (+1.4 pp accuracy) is modest. This is significant: it demonstrates that the additional complexity of the RL training pipeline does not dramatically exceed the in-distribution performance of a well-tuned supervised alternative. The practical case for the proposed system therefore rests primarily on its non-stationary resilience, where the margin is substantial: **+15.6 percentage points overall (46.8% vs 31.2%)** a 50% relative improvement over the frozen LSTM across four abrupt regime shifts.

The per-direction analysis adds an important nuance: ARIMA degenerates entirely to constant North prediction, which produces near-perfect single-direction accuracy but a collapsed Macro-F1 of 9.55%. This failure mode is directly traceable to the spatial averaging that precedes ARIMA fitting — once the 15-RSU density is collapsed to a scalar per direction, the multivariate spatial structure that differentiates the four directions is lost, and the model cannot learn directional transitions. This finding underscores a general principle: models that discard spatial information cannot perform reliable multi-directional classification, regardless of their temporal modelling quality.

The PPO+PSO variant presents the most surprising result: its 16.1% overall accuracy in the non-stationary evaluation falls below the 25% random baseline, meaning the PSO bias update actively degrades the frozen policy rather than correcting it. This occurs because PSO’s velocity-based continuous updates overshoot in the 4-dimensional logit bias space when confronted with abrupt distributional shifts. Each regime change forces a large correction, and PSO’s inertia carries the particles past the optimal bias region before the next update can compensate. The GA’s discrete crossover and elite selection operators are more robust to this “chasing a moving target” dynamic, which explains the large performance gap between GA and PSO specifically in the non-stationary setting (46.8% vs 16.1%) compared to the much smaller gap in the static setting (78.0% vs 74.4%).

IV.11.2 SUMO Deployment Results Analysis

The closed-loop SUMO validation produces a **+33.3% throughput improvement** and a **+10.4% average speed gain** over fixed-time control on the Belgrade network. The apparent 1.6% increase in average waiting time is a composition artefact: the MEC system dispatches 33% more vehicles including those that waited longest before the simulation ends, raising the per-vehicle average without degrading the individual user experience. The near-identical halted vehicle counts confirm that in a heavily congested regime, physical road capacity is the binding constraint and signal timing can improve throughput flow without substantially reducing instantaneous queue depth.

A key observation is that the DRL model generalizes from Nördlingen (training network, Germany) to Belgrade (deployment network, Serbia) without any retraining, demonstrating cross-network transferability. The sinusoidal temporal encoding of time of day captures rush-hour demand patterns that are structurally similar across urban networks, and the GA online bias correction adapts the model’s directional emphasis to the specific characteristics of the Belgrade network within the first few inference steps.

IV.11.3 Limitations

Several limitations bound the scope of the conclusions and motivate the future directions outlined in Section IV.12.

Synthetic data dependency: Both the Nördlingen training dataset and the Belgrade simulation demand are synthetically generated. While the generators are calibrated to reproduce realistic rush-hour profiles and spatial demand patterns, real-world V2X data would include measurement noise, packet loss, vehicle diversity, and irregular demand patterns (accidents, events, weather) that the synthetic environment does not fully capture. Validation on real-world vehicular data is a necessary step before operational deployment.

Single gNB coverage assumption: The entire system 20 RSUs, 74 intersections, and the MEC server operates within a single 800 m-radius gNB cell. Real urban deployments involve multiple overlapping cells with inter-cell handover, coordination overhead, and variable RSU counts per cell. The scalability of the proposed architecture to multi-cell MEC environments has not been evaluated.

Non-stationary evaluation depth: The four-regime non-stationary stream uses abrupt instantaneous transitions between traffic regimes. Real distribution shifts are typically more

gradual. The evaluation also covers only four shift patterns; a broader sweep of shift frequencies, magnitudes, and temporal profiles would be needed to characterize the robustness of the GA adaptation mechanism more comprehensively.

Model interpretability: The Direction Actor Critic architecture with TCN encoding and multi-head attention is a high-capacity black-box model. The attention weights provide some interpretability into which RSUs the model relies on for each prediction, but no systematic analysis of the learned spatial attention patterns has been conducted in this work.

IV.12 Conclusion

This chapter presented a comprehensive evaluation of the proposed two-stage hybrid system a Proximal Policy Optimization agent trained offline for directional traffic trend prediction, augmented by an online Genetic Algorithm bias-correction layer across two complementary evaluation tiers and one real-world closed-loop deployment scenario.

On the static held-out test set, the PPO+GA system achieved the highest overall accuracy of 78.00% and a Macro-F1 of 79.26%, outperforming all evaluated baselines. The near-equality of these two metrics confirms that the class-weight reward shaping during PPO training successfully prevented majority-class collapse, enabling reliable detection of minority directional surges. The supervised LSTM classifier performed competitively at 76.60% accuracy, establishing that RL-based training does not yield large gains under stable, in-distribution conditions. The classical baselines Kalman filter at 55.20% and ARIMA at 23.60% demonstrated progressively severer degradation, with ARIMA degenerating entirely to constant North prediction and producing a Macro-F1 of only 9.55%, a failure directly attributable to the loss of spatial structure introduced by its per-direction scalar averaging.

The non-stationary evaluation revealed the most operationally significant finding of the chapter. Across four abrupt traffic regime transitions, the frozen LSTM collapsed toward random-level performance with no recovery mechanism, while the PPO+GA system maintained 46.8% accuracy a 50% relative improvement representing a margin of 15.6 percentage points. This advantage grew segment by segment as the GA bias vector accumulated feedback from the sliding adaptation window, reaching a +18.4 percentage point advantage in the final afternoon-rush segment. The PPO+PSO variant produced a counter-intuitive result: its overall non-stationary accuracy of 16.1% fell below the 25% random baseline, confirming that PSO's velocity-based continuous updates overshoot in the low-dimensional logit bias space under abrupt distributional shifts. This finding underscores that the choice of metaheuristic optimizer is not interchangeable the GA's discrete crossover and elite-selection operators are specifically better suited to the abrupt regime-change adaptation task.

From a computational perspective, the PPO+GA system's inference latency of 54.59 ms and throughput of 18 samples per second, while substantially slower than the LSTM baseline at 2.21 ms and 453 samples per second, remain well within the one-minute operational budget of a MEC server. The online GA adaptation overhead of 56.3 seconds across the entire non-stationary evaluation is negligible relative to the 6-hour PPO training investment, confirming the practical viability of the full two-stage deployment pipeline.

The closed-loop SUMO validation on the Belgrade road network demonstrated that the directional predictions produced by the PPO+GA model translate into measurable real-world traffic improvements when embedded in a Hybrid Max-Pressure + DRL Direction Boost controller. The MEC-enabled system achieved a 33.3% throughput gain and a 10.4% average speed improvement over fixed-time signal control. Critically, the model generalized from the Nördlingen training network to the Belgrade deployment network without retraining, with the GA bias correction adapting the directional emphasis to the new network's characteristics within the first few inference steps.

Taken together, the results of this chapter establish three core conclusions. First, reinforcement learning can produce competitive directional traffic classifiers without labeled training data, matching and marginally exceeding the accuracy of a well-tuned supervised LSTM on in-distribution data. Second, and most importantly, online metaheuristic bias correction with a Genetic Algorithm provides substantial resilience to abrupt traffic regime shifts that fully incapacitate frozen supervised models, making the proposed system a more suitable choice for dynamic real-world deployments. Third, the predicted directional trend signal is actionable: when fed into a real-time signal controller, it produces network-level throughput and speed improvements that are measurable within a realistic urban simulation.

General Conclusion & Future Perspectives

This thesis proposed, formalized, and validated a hybrid two-stage framework for real-time directional traffic trend prediction in V2X-enabled smart cities. The work was motivated by a gap that none of the existing approaches had simultaneously bridged: the inability to maintain reliable prediction accuracy under the abrupt, non-stationary traffic regime shifts that characterize real urban environments, while meeting the millisecond-scale latency constraints of Multi-Access Edge Computing deployment.

Chapter I identified the core limitations of classical statistical models stationarity assumptions that urban traffic systematically violates and of state-of-the-art deep learning architectures whose catastrophic forgetting under distributional shift demands GPU-hours of retraining. It also established that existing reinforcement learning applications in intelligent transportation are almost exclusively focused on control tasks, leaving directional trend prediction as an unexplored RL problem domain. Chapter II translated these motivations into a precise system model, defining a three-tier architecture (RSU perception layer, 5G backhaul, MEC edge intelligence hub), a real urban road network extracted from Open Street Map for Nördlingen, Germany with 15 geographically-distributed RSUs, and a physics-inspired traffic dataset spanning 200,000 simulated minutes with double-Gaussian rush-hour dynamics, per-RSU temporal offsets, cross-intersection spillover, and a slow sinusoidal non-stationary drift. The prediction task was formalized as a four-class directional trend classification problem over a 60-minute horizon.

Chapter III specified the complete hybrid architecture. An offline Proximal Policy Optimization (PPO) agent learns a directional prediction policy through a hierarchical neural network combining per-RSU causal Temporal Convolution Networks for temporal encoding and multi-head self-attention for spatial integration, trained with an asymmetric class-weighted reward that prevents majority-class collapse. Once training converges, all network weights are frozen and an online Genetic Algorithm (GA) continuously optimizes a 4-dimensional logit bias vector against a sliding window of recent prediction outcomes, correcting for systematic directional errors introduced by distributional shift without any gradient-based retraining. Chapter IV provided the empirical validation: the PPO+GA system achieved the highest static accuracy of 78.0% and Macro-F1 of 79.26% on the held-out test set, a 50% relative accuracy improvement over the frozen supervised LSTM across four abrupt regime transitions, and a 33.3% vehicle throughput gain in a closed-loop SUMO simulation on a real Belgrade road network with 74 signalized intersections a network the model had never seen during training, demonstrating cross-network generalization.

Taken together, the results establish that decoupling representation learning from online adaptation is a principled and effective strategy for non-stationary traffic prediction: the PPO stage captures the complex spatiotemporal regularities of diurnal demand patterns, while the GA stage handles the residual distributional shift that a static policy cannot anticipate a balance that neither pure deep learning nor pure RL achieves alone.

Key Findings:

Finding 1 — Static accuracy: PPO+GA achieve 78.0% accuracy and 79.26% Macro-F1, outperforming all baselines including the supervised LSTM (76.6%). The class-weight reward shaping prevents majority-class collapse, yielding well-balanced per-direction detection.

Finding 2 — Non-stationary robustness: The frozen LSTM collapses to near-random performance at each regime transition. PPO+GA maintain 46.8% accuracy across four abrupt shifts a 50% relative improvement over LSTM (31.2%). The adaptation advantage grows with accumulated sliding-window feedback, reaching +18.4 percentage points in the final regime.

Finding 3 — Metaheuristic choice is critical: PPO+PSO falls below the 25% random baseline in non-stationary conditions (16.1%), despite competing adequately in static evaluation (74.4%). PSO's velocity-based updates overshoot in the low-dimensional bias space under abrupt regime changes, confirming that GA and PSO are not interchangeable for this task.

Finding 4 — Spatial aggregation is mandatory: ARIMA degenerates to constant North prediction (Macro-F1: 9.55%) because scalar spatial averaging destroys the multivariate RSU structure needed for multi-directional classification. This establishes a general principle: discarding spatial heterogeneity across RSUs makes reliable directional prediction impossible.

Finding 5 — Computational feasibility: PPO+GA inference latency of 54.59 ms remains more than 1,000 times faster than the 60-second MEC decision window. The cumulative online GA overhead of 56.3 seconds across the full non-stationary evaluation is negligible relative to the 6-hour PPO training investment.

Finding 6 — Cross-network generalization: The model trained on Nördlingen generalizes to Belgrade without retraining, producing a 33.3% throughput gain and a 10.4% average speed improvement in a closed-loop SUMO simulation, validating the system's transferability for real-world deployment.

Future Work and Perspectives:

Real-world V2X dataset validation : Repeating the evaluation on data from an operational C-V2X or DSRC deployment would confirm whether the performance margins observed on synthetic data hold under real measurement noise, packet loss, and irregular demand events.

Multi-cell MEC coordination: Extending the architecture to coordinate predictions across multiple overlapping gNB cells through a hierarchical model local PPO agents per cell feeding a network-wide aggregator would address the single-cell limitation and enable city-scale adaptive signal control.

Isolated evaluation of the DRL Direction Boost.: Adding a pure Max-Pressure instance (no directional boost) to the SUMO study would rigorously isolate the incremental contribution of the prediction layer from the reactive component of the controller.

Multi-horizon prediction: Extending the system to predict directional trend trajectories at multiple future horizons (5, 15, and 30 minutes) would enable longer-horizon signal optimization, proactive incident detection, and dynamic route guidance beyond the single-horizon classification validated here.

Federated learning for privacy-preserving adaptation: A federated GA adaptation scheme where each RSU maintains a local bias estimate and periodically aggregates with the MEC server would reduce privacy exposure of individual vehicle trajectories while preserving the online adaptation capability, aligning with emerging V2X data governance requirements.

Bibliography

- [1] A. Visvizi et al, Smart Cities — A Structured Literature Review. *Smart Cities* (MDPI), 6(4):1719–1743, 2023.
- [2] N. Gupta, Editorial: Smart Cities Challenges, Technologies and Trends. *Frontiers in Big Data*, 6:1258051, 2023.
- [3] Bitsensing, Revolutionizing Smart Cities: The Future of Traffic Management. Industry Report, 2025.
- [4] H. B. Faheem et al, Impact of Traffic Congestion on Transportation System: Challenges and Remediations. *Mansoura Engineering Journal*, 49(2), 2024.
- [5] INRIX, Analyzing Urban Congestion in 2024 and Understanding How Cities Can Adapt. Technical Report, 2024.
- [6] A. Kaleli et al, Enabling Technologies for Urban Smart Mobility: Recent Trends, Opportunities and Challenges. *Sensors*, 21(6):2143, 2021.
- [7] M. Zaman et al, A Review of IoT-Based Smart City Development and Management. *Smart Cities* (MDPI), 7(3), 2024.
- [8] R. Lim et al, Integrating Machine Learning for the Sustainable Development of Smart Cities. *Frontiers in Sustainable Cities*, 2024.
- [9] S. Mishra and S. Murthy, A Predictive and Optimization Approach for Enhanced Urban Mobility Using Spatiotemporal Data. arXiv preprint arXiv:2410.05358, 2024.
- [10] Arxiv Authors, Using Ensemble Learning with Hybrid GNN and Transformers to Predict Traffic in Cities. arXiv preprint arXiv:2511.02484, 2024.
- [11] M. Maaroufi et al, Advances in Traffic Congestion Prediction: An Overview of Emerging Techniques. *Smart Cities* (MDPI), 8(1):25, 2025.
- [12] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. MIT Press, Cambridge, MA, 2018.
- [13] D. Silver et al, Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.
- [14] OpenAI et al, Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.
- [15] M. Bojarski et al, End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316, 2016.
- [16] H. Wei, G. Zheng, H. Yao, and Z. Li, IntelliLight: A reinforcement learning approach for intelligent traffic light control. In *Proc. 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2496–2505, 2018.
- [17] A. Bansal, K. Balaji, and Z. Lalani, Temporal Encoding Strategies for Energy Time Series Prediction. arXiv preprint arXiv:2503.15456, 2025.
- [18] A. Raffin et al, Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

- [19] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.
- [20] G. Dulac-Arnold et al, Challenges of real-world reinforcement learning. arXiv preprint arXiv:1904.12901, 2019.
- [21] J. Kennedy and R. Eberhart, Particle swarm optimization. In Proc. IEEE International Conference on Neural Networks, volume 4, pages 1942–1948, 1995.
- [22] G. Brockman et al, OpenAI Gym. arXiv preprint arXiv:1606.01540, 2016.
- [23] Z. Jiang, D. Xu, and J. Liang, A deep reinforcement learning framework for financial portfolio management. arXiv preprint arXiv:1706.10059, 2017.
- [24] S. Mehar et al, Review on V2X, I2X, and P2X Communications and Their Applications. Sensors, 2019.
- [25] I. Noorani et al, Interoperability of C-V2X and DSRC. Preprints, 202510.0177, 2025.
- [26] C. Xu et al, V2X Cooperative Perception for Autonomous Driving. arXiv preprint arXiv:2310.03525, 2024.
- [27] S. Gyawali et al, Stochastic Geometry Modeling of Cellular V2X Communication. arXiv preprint arXiv:1804.08409, 2018.
- [28] C. Campolo et al, Network Slicing for Vehicular Communication. arXiv preprint arXiv:1905.09578, 2019.
- [29] A. Masood et al, Secure Vehicular Communication: An Intelligent Physical Layer Approach. arXiv preprint arXiv:1905.05075, 2019.
- [30] Y. Yao, L. Rao, X. Liu, and X. Zhou, Delay analysis and study of IEEE 802.11p based DSRC safety communication in a highway environment, 2013.
- [31] R. Sedar, C. Kalalas, F. Vazquez-Gallego, L. Alonso, and J. Alonso-Zarate, A Comprehensive Survey of V2X Cybersecurity Mechanisms and Future Research Paths. IEEE Open Journal of the Communications Society, 4:325–391, 2023.
- [32] D. Villa et al, Advancements in Mobile Edge Computing and Open RAN Leveraging AI and ML. arXiv preprint arXiv:2502.02886, 2025.
- [33] A. Masood et al, Exploiting Mobile Edge Computing for Enhancing Vehicular Applications in Smart Cities. Sensors, 19(10), 2019.
- [34] A. Alfoudi et al, A Reliable and Efficient 5G Vehicular MEC. arXiv preprint arXiv:2503.19320, 2025.
- [35] S. Toufga et al, Network Service and Resource Orchestration in MEC-Enhanced Vehicular Networks. Sensors, 20(13), 2020.
- [36] P. Parastar et al, Energy-Efficient Task Computation at the Edge for Vehicular Services. arXiv preprint arXiv:2511.18449, 2025.
- [37] J. Hwang et al, Two-Layer Edge Intelligence for Task Offloading with UAV Assistance. Sensors, 2024.

- [38] A. Osman et al, Advancing Urban Planning with Deep Learning: Intelligent Traffic Flow Prediction. *Smart Cities (MDPI)*, 5(4), 2024.
- [39] Y. Luo et al, Recent Advances in Deep Learning for Traffic Flow Prediction. *ACM Digital Library*, 2025.
- [40] G. Shi et al, Deep Transformer-Based Heterogeneous Spatiotemporal Graph Learning for Traffic Forecasting. *iScience*, 2024.
- [41] D. Slijepcevic et al, Exploring the Quality of Dynamic Open Government Traffic Data. *Sensors*, 22(24), 2022.
- [42] X. Zhou et al, Large Foundation Models for Trajectory Prediction in Autonomous Driving. *arXiv preprint arXiv:2509.10570*, 2025.
- [43] Z. Chen et al, Urban Computing in the Era of Large Language Models. *arXiv preprint arXiv:2504.02009*, 2025.
- [44] T. Yabe et al, Predicting Human Mobility Flows in Cities Using Deep Learning on Satellite Imagery. *Nature Communications*, 2024.
- [45] A. G. Ríos et al, Machine Learning Methods for Robust Traffic Prediction Under Dynamic Conditions. *Helex Science Journal*, 9, 2024.
- [46] N. Pröllochs and S. Feuerriegel, Reinforcement Learning in R. *arXiv preprint arXiv:1810.00240*, 2018.
- [47] MIT OpenCourseWare, Markov Decision Processes — Introduction to Machine Learning (6.390). MIT, 2024.
- [48] F. Z. Dai, On Reward Structures of Markov Decision Processes. *arXiv preprint arXiv:2308.14919*, 2023.
- [49] S. Chen et al, State Action Separable Reinforcement Learning. *arXiv preprint arXiv:2006.03713*, 2020.
- [50] J. Brehmer et al, Structure in Deep Reinforcement Learning: A Survey and Open Problems. *arXiv preprint arXiv:2306.16021*, 2023.
- [51] J. Orr and A. Dutta, A Survey on Deep Reinforcement Learning Algorithms for Robotic Manipulation. *Sensors (MDPI)*, 23(7):3762, 2023.
- [52] Y. Li, Deep Reinforcement Learning: An Overview. *arXiv preprint arXiv:1701.07274*, 2017.
- [53] L. Canese et al, The Definitive Guide to Policy Gradients in Deep Reinforcement Learning. *arXiv preprint arXiv:2401.13662*, 2024.
- [54] Wikipedia / OpenAI, Proximal Policy Optimization — OpenAI Spinning Up Documentation, 2024.
- [55] F. Ye et al, Truly Proximal Policy Optimization. *arXiv preprint arXiv:1903.07940*, 2019.
- [56] Hugging Face, Proximal Policy Optimization — Deep RL Course, 2022.

- [57] A. Unnikrishnan, Financial News-Driven LLM Reinforcement Learning for Portfolio Management, 2024.
- [58] Y. Gu et al, A Dynamical Clipping Approach with Task Feedback for PPO. arXiv preprint arXiv:2312.07624, 2023.
- [59] X. Chen et al, Smart City Traffic Flow and Signal Optimization Using STGCN-LSTM and PPO Algorithms. DOAJ, 2025.
- [60] Y. Zhou et al, Federated Deep Reinforcement Learning-Based Urban Traffic Signal Optimal Control. Scientific Reports, 2025.
- [61] D. K. Kwesiga et al, Adaptive Traffic Signal Control Based on Multi-Agent Reinforcement Learning. arXiv preprint arXiv:2503.02189, 2025.
- [62] B. M. Williams and L. A. Hoel, Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of Transportation Engineering*, 129(6):664–672, 2003.
- [63] B. L. Smith, B. M. Williams, and R. K. Oswald, Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10(4):303–321, 2002.
- [64] I. Okutani and Y. J. Stephanedes, Dynamic prediction of traffic volume through Kalman filtering theory. *Transportation Research Part B: Methodological*, 18(1):1–11, 1984.
- [65] S. Lefèvre, D. Vasquez, and C. Laugier, A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH Journal*, 1(1):1–14, 2014.
- [66] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*, 54:187–197, 2015.
- [67] Z. Zhao et al, LSTM network: A deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems*, 11(2):68–75, 2017.
- [68] Y. Li, R. Yu, C. Shahabi, and Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *Proc. International Conference on Learning Representations (ICLR)*, 2018.
- [69] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *AAAI Conference on Artificial Intelligence*, volume 33, pages 922–929, 2019.
- [70] N. Abdel Madjid et al, Trajectory Prediction for Autonomous Driving: Progress, Limitations, and Future Directions. *Information Fusion*, arXiv preprint arXiv:2503.03262, 2025.
- [71] Y. Jiao, M. Miao, Z. Yin, C. Lei, X. Zhu, L. Nie, and B. Tao, A Hierarchical Hybrid Learning Framework for Multi-agent Trajectory Prediction. arXiv preprint arXiv:2303.12274, 2023.

- [72] M. Maaroufi et al, Advances in Traffic Congestion Prediction: Emerging Techniques. Smart Cities (MDPI), 8(1):25, 2025.
- [73] M. Towers et al, Gymnasium: A Standard Interface for Reinforcement Learning Environments. arXiv preprint arXiv:2407.17032, 2024.
- [74] Y. Gao and D. Levinson, A bifurcation of the peak: new patterns of traffic peaking during the COVID-19 era. Transportation, Springer, 2022.
- [75] W. Zhou et al, Investigating Travel Flow Differences between Peak Hours with Spatial Model. Journal of Advanced Transportation, 2022.
- [76] K. Bochenina et al, From Urban Data to City-Scale Models: A Review of Traffic Simulation Case Studies. IET Intelligent Transport Systems, 2025.
- [77] M. Cogoni et al, Estimating Peak-Hour Traffic Congestion Patterns for Interacting Agents on Urban Networks. arXiv preprint arXiv:2207.11219, 2022.
- [78] C. Ye et al, An Eco-Driving Strategy at Multiple Fixed-Time Signalized Intersections Considering Traffic Flow Effects. Sensors, 2024.
- [79] L. Zhang et al, Signal Timing Optimization for Mixed Connected Automated Traffic Based on a Markov Delay Approximation. arXiv preprint arXiv:2505.11522, 2025.
- [80] S. Kim and E. Ekici, Performance Analysis of Cellular-V2X with Adaptive and Selective Power Control. arXiv preprint arXiv:2008.03453, 2020.
- [81] I. Bychkov et al, Enhanced V2X Communication Using Game-Theory Based Adaptive MAC Protocols. arXiv preprint arXiv:2506.09817, 2025.
- [82] P. A. Lopez, E. Wiessner, M. Behrisch et al, Microscopic Traffic Simulation using SUMO. In 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pages 2575–2582, 2018.
- [83] G. Boeing, OSMnx: New Methods for Acquiring Street Networks. Computers, Environment & Urban Systems, 2017.

Abstract



المخلص :

أفرز النمو المتسارع للمدن الذكية وانتشار شبكات الاتصال بين المركبات وكل شيء (V2X) حاجة ملحة إلى أنظمة للتنبؤ بحركة المرور تجمع بين الدقة العالية والقدرة على التكيف مع الظروف غير الثابتة، مع الوفاء بالقيود الصارمة على زمن الاستجابة التي تفرضها منصات الحوسبة المتطورة متعددة الوصول (MEC). تقترح هذه الأطروحة إطار عمل هجيناً من مرحلتين للتنبؤ الآني باتجاه حركة المرور السائد وتخضعه للتحقق التجريبي. في المرحلة الأولى، يُدرَّب عميل يعتمد خوارزمية تحسين السياسة القريبة (PPO) مزوّد بمشقر شبكة الالتفاف الزمني وموحد مكاني قائم على آلية الانتباه متعدد الرؤوس، وذلك عبر 200,000 خطوة زمنية من البيانات التركيبية المولدة على الشبكة الحقيقية لمدينة نوردينغن بألمانيا، بهدف تصنيف الاتجاه السائد في موجة حركة المرور (شمال، شرق، جنوب، أو غرب) ضمن أفق تنبؤي مدته 60 دقيقة، استناداً إلى مشاهدات 15 وحدة جانبية للطريق. في المرحلة الثانية، تُضاف إلى السياسة المُجمّدة لعميل PPO طبقة من الخوارزمية الجينية (GA) تُحسِّن أنياً متجه تحيز لوغاريتمي رباعي الأبعاد مقارنةً بنافذة متحركة من نتائج التنبؤ الأخيرة، لتصحيح أخطاء الاتجاه المنهجية الناجمة عن التحولات المفاجئة في نمط حركة المرور دون الحاجة إلى أي إعادة تدريب. يُظهر التقييم التجريبي مقارنةً بخطوط الأساس المتمثلة في نموذج LSTM الخاضع للإشراف وARIMA ومرشح كالمان والمصنّف العشوائي، أن نظام PPO+GA يُحقّق دقة قدرها 78.0% ومعامل F1 الكلي 79.26% على مجموعة الاختبار الثابتة، فيما يحافظ على دقة 46.8% عبر أربع تحولات مفاجئة في نمط حركة المرور، بتحسّن نسبي يبلغ 50% مقارنةً بنموذج LSTM المُجمّد (31.2%). وقد أسفر التحقق في حلقة مغلقة باستخدام محاكي SUMO على شبكة بلغراد الحقيقية المشتملة على 74 تقاطعاً إشارياً عن تحسّن في معدل مرور المركبات بنسبة 33.3% وارتفاع في متوسط السرعة بنسبة 10.4% مقارنةً بالتحكم الزمني الثابت دون أي إعادة تدريب، مما يُثبت قابلية التعميم عبر الشبكات. تُؤكد جميع النتائج أن الإطار المقترح يستوفي متطلبات النشر الآني في بيئات MEC بزمان استجابة 54.59 ميلي ثانية، أي أقل بكثير من نافذة التشغيل البالغة 60 ثانية.

الكلمات المفتاحية: المدينة الذكية، V2X، التنبؤ بحركة المرور، التعلم العميق المعزز، تحسين السياسة التقريبية (PPO)، الخوارزمية الجينية، الحوسبة الطرفية متعددة الوصول (MEC).



Abstract:

The rapid growth of smart cities and the proliferation of Vehicle-to-Everything (V2X) communication networks have created an urgent need for traffic prediction systems that are simultaneously accurate, adaptive to non-stationary conditions, and deployable within the stringent latency constraints of Multi-Access Edge Computing (MEC) platforms. This thesis proposes and validates a hybrid two-stage framework for real-time directional traffic trend prediction. In the first stage, a Proximal Policy Optimization (PPO) agent equipped with a Temporal Convolution Network encoder and a multi-head self-attention spatial integration module is trained offline on a 200,000-timestep synthetic V2X dataset generated over the real urban road network of Nördlingen, Germany, learning to classify the dominant directional traffic surge (North, East, South, or West) over a 60-minute prediction horizon from 15 roadside unit observations. In the second stage, the frozen PPO policy is augmented at inference time by a Genetic Algorithm (GA) that continuously optimises a 4-dimensional logit bias vector against a sliding window of recent prediction outcomes, correcting for systematic directional errors introduced by abrupt traffic regime shifts without any gradient-based retraining. Experimental evaluation against supervised LSTM, ARIMA, Kalman filter, and random baselines demonstrates that the PPO+GA system achieves 78.0% accuracy and 79.26% Macro-F1 on the static test set, and maintains 46.8% accuracy across four abrupt regime transitions a 50% relative improvement over the frozen LSTM (31.2%). A closed-loop validation in the SUMO traffic simulator on the Belgrade road network, with 74 signalized intersections, yields a 33.3% vehicle throughput gain and a 10.4% average speed improvement over fixed-time signal control without any retraining, demonstrating cross-network generalization. All results confirm that the proposed framework meets the computational requirements of real-time MEC deployment with an inference latency of 54.59 ms, well within the 60-second operational window.

Keywords: Smart city, V2X, Traffic prediction, Deep Reinforcement Learning, Proximal Policy Optimization (PPO), Genetic Algorithm, Multi-Access Edge Computing (MEC).



Résumé :

La croissance rapide des villes intelligentes et la prolifération des réseaux de communication Vehicle-to-Everything (V2X) ont créé un besoin urgent de systèmes de prévision du trafic qui soient à la fois précis, adaptables aux conditions non stationnaires et déployables dans le cadre des contraintes de latence strictes des plates-formes Multi-Access Edge Computing (MEC). Ce mémoire propose et valide un cadre hybride en deux étapes pour la prévision des tendances du trafic directionnel en temps réel. Dans la première étape, un agent d'optimisation de politique proximale (PPO) équipé d'un encodeur de réseau de convolution temporelle et d'un module d'intégration spatiale d'auto-attention multi-têtes est entraîné hors ligne sur un ensemble de données V2X synthétiques à 200 000 pas de temps généré sur le réseau routier urbain réel de Nördlingen, en Allemagne, apprenant à classer l'augmentation de trafic directionnelle dominante (Nord, Est, Sud ou Ouest) sur un horizon de prévision de 60 minutes à partir de 15 unités en bordure de route. observations. Dans la deuxième étape, la politique PPO gelée est augmentée au moment de l'inférence par un algorithme génétique (GA) qui optimise en continu un vecteur de biais logit à 4 dimensions par rapport à une fenêtre glissante de résultats de prédiction récents, corrigeant les erreurs de direction systématiques introduites par des changements brusques de régime de circulation sans aucun recyclage basé sur le gradient. L'évaluation expérimentale par rapport au LSTM supervisé, à l'ARIMA, au filtre de Kalman et aux lignes de base aléatoires démontre que le système PPO+GA atteint une précision de 78,0 % et 79,26 % de Macro-F1 sur l'ensemble de tests statiques, et maintient une précision de 46,8 % sur quatre transitions de régime brusques, soit une amélioration relative de 50 % par rapport au LSTM gelé (31,2 %). Une validation en boucle fermée dans le simulateur de trafic SUMO sur le réseau routier de Belgrade, avec 74 intersections signalisées, donne un gain de débit de véhicules de 33,3 % et une amélioration de la vitesse moyenne des véhicules de 10,4 % par rapport au contrôle des feux à temps fixe sans aucun recyclage, démontrant une généralisation inter-réseaux. Tous les résultats confirment que le cadre proposé répond aux exigences informatiques du déploiement MEC en temps réel avec une latence d'inférence de 54,59 ms, bien dans la fenêtre opérationnelle de 60 secondes.

Mots clés : Ville intelligente, V2X, Prédiction de trafic, Apprentissage par renforcement profond, Optimisation de politique proximale (PPO), Algorithme génétique, Informatique de périphérie multi-accès (MEC).