PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

YAHIA FARES UNIVERSITY
OF MEDEA



Faculty of Sciences
Department of Mathematics and Computer Science

MASTER THESIS IN

Field: Computer Science

Specialty: Systems Engineering and Web Technology

Title:

A Hybrid Predictive Meta-Heuristic Framework for Dynamic Traffic Signal Optimization in Multi-Intersection Networks

---

Prepared by:

BOULMAALI Douaa Nesrine

DJERROUD Manar

Supervised by:

Dr. BABAALI Baligh

Academic Year: 2025–2026

Dedication

By Douaa:

I dedicate this work

to my family
for their kind of love that doesn't need to be explained.
you never stood in front of me, yet nothing moved forward without you. Everything I reached carries something of you in it.

to Manar,
my best friend, my partner in everything.
We didn't divide the work, we carried it together.
What we built here goes beyond a project. You were my balance in the chaos, my strength on the heavy days, and the one who truly understood every moment of this journey without needing explanations. Doing this with you made all the difference. I couldn't have done this the same way without you.

to those who were part of this path
whether through a word, a gesture, in ways big or small
seen or unseen, named or unnamed your presence mattered more than you know.

This holds more than what is seen,
it holds what it took to get here, in all its weight.
The effort that stayed quiet, the pressure that had to be handled, and the strength it demanded, even when it didn't feel enough.

And that is why this means more than just finishing.

By Manar:

I dedicate this work

To my parents,
for their endless love, sacrifices, and unwavering support that made every achievement possible. This work is a reflection of everything you have given me.

To my family,
for being my constant source of comfort, strength, and peace through every stage of this journey.

To my friends,
for the encouragement, laughter, and support that made even the hardest moments easier to overcome.

To my dearest binôme, Douaa,
Throughout this journey, your presence was more than just support, in the most trying times, it was strength, consolation, and brightness.
With your tolerance, compassion, and unfailing support, you helped me overcome every obstacle. A piece of you is present in every late night, every challenge, and every achievement.
Although this thesis bears our names, it also captures your commitment, your passion, and the lovely relationship we developed together. I will always be thankful to have traveled this path with you.

This work is dedicated to everyone who believed in me before I had any results to show them.

Acknowledgements

First and foremost, all praise and gratitude to God the Almighty, who granted us the strength and ability to complete this work, Alhamdulillah.

We would like to express our deepest gratitude and appreciation to our supervisor, **Dr. Baligh BABAALI**, for his invaluable guidance, continuous support, and immense patience with us throughout this journey, thank you very much.

Our sincere thanks also go to our dear friend and cousin, **Nada**, who helped us tremendously during this project.

We owe a great debt of gratitude to **AbdelMalek.A** for his invaluable technical support.His hands-on was essential to the success of this project, and we truly appreciate the energy and time he invested in helping us move forward.

We are deeply grateful to all the professors who taught and guided us throughout our academic path.

Finally, we extend our heartfelt thanks to our parents, brothers, sisters, friends, and colleagues for their unwavering moral and physical support.

الملخص

تعتبر الزحمة المرورية الحضرية من أهم الصعوبات التي تعترض سبيل إدارة المدن العصرية، فهي تؤدي إلى تأخيرات كبيرة، وزيادة في استهلاك الوقود، وتدهور واضح للبيئة. تظل خطط التحكم بالإشارات الثابتة هي الطريقة الأكثر استخدامًا في شبكات النقل بالمدن الجزائرية، لكنها تعاني من نقص القدرة على التكيف مع الطبيعة الديناميكية وغير المتوقعة للطلب المروري الفعلي. تقدم هذه الأطروحة نموذجاً هجيناً تنبؤياً مبني على تقنيات ميتا-إرشادية بهدف تحسين إشارات المرور بشكل ديناميكي، وتم تطبيقه على شبكة طرق فعلية تضم تسعة تقاطعات في مدينة البليدة بالجزائر.

الإطار المقترح يجمع بين ثلاثة عناصر أساسية تعمل ضمن دائرة تحكم ذات أفق متغير: شبكة عصبية تعتمد على الذاكرة طويلة-قصيرة المدى (LSTM) تتوقع زمن الانتظار الإجمالي في الشبكة بمقدار ستين ثانية مسبقًا، إلى جانب مُحسّن ميتا-إرشادي يستند إلى مجموعة متنوعة من الحلول يبحث دوماً عن الخطة الإشارية المثلى عند كل نقطة تحكم، فضلاً عن محاكي المرور المجهري SUMO المتصل عبر واجهة TraCI بشكل فوري.

ثلاثة خوارزميات تحسين يتم تقييمها: خوارزمية تحسين الحيتان (WOA) وخوارزمية تحسين صقور هاريس (HHO)، كلتاهما تعمل بناءً على تنبؤات LSTM، بالإضافة إلى الخوارزمية الجينية (GA) التفاعلية التي تُستخدم كمرجع مقارنة دون أي تنبؤ.

أثبتت التجارب المنفذة في ثلاثة أوضاع مرورية ، منخفض (200 مركبة/ساعة)، ومتوسط (500 مركبة/ساعة)، وذروة (900 مركبة/ساعة) ، والتي تم تحقيقها من خلال خمس تشغيلات مستقلة، أن التنبؤي HHO يمكّن من تقليل أوقات الانتظار بنسبة 27.9% و38.2% و34.6% على التوالي مقارنةً بالخطة الثابتة المرجعية.

تسجل WOAالتنبؤية انخفاضات بنسبة 18.8%، 27.0%، و18.8%. من ناحية أخرى، فإن الخوارزمية الجينية التفاعلية تزيد من سوء الأوضاع بنسبة قد تصل إلى 19.1% في ذروة الحركة، مما يثبت أن المعلومات التنبؤية تعتبر ضرورية لتحسين إشارات المرور أثناء الازدحام الشديد .

HHOيتصدر كأكثر خوارزمية موثوقية، حيث يجمع بين أفضل أداء متوسط وأقل تباين في النتائج عبر جميع الحالات.

الكلمات المفتاحية: تحسين إشارات المرور، التنبؤ بـ LSTM، الخوارزميات الميتا-إرشادية، خوارزمية تحسين الحيتان، خوارزمية تحسين صقور هاريس، الخوارزمية الجينية، التحكم بالأفق المتحرك، محاكاة SUMO، البليدة

Abstract

One of the most important issues facing contemporary city administration is urban traffic congestion, which results in major delays, higher fuel usage, and environmental damage. The predominant method in Algerian urban networks, fixed-time signal management plans, are essentially unable to handle the dynamic and unpredictable character of actual traffic demand. In this thesis, an actual nine-intersection road network in the Algerian city of Blida is used to test a hybrid predictive meta-heuristic framework for dynamic traffic signal optimization.

A Long Short-Term Memory (LSTM) neural network that forecasts the overall network waiting time sixty seconds in advance, a population-based meta-heuristic optimizer that looks for the best signal plan at each control horizon, and the SUMO microscopic traffic simulator connected via the TraCI real-time interface are all integrated into the framework and function within a rolling horizon control loop.

Three optimization methods are assessed: a reactive Genetic Algorithm (GA), which serves as a benchmark without prediction, and the Whale Optimization Algorithm (WOA) and Harris Hawks Optimization (HHO), both of which are guided by LSTM prediction.
Predictive HHO reduces waiting times by 27.9%, 38.2%, and 34.6% in comparison to the fixed-time baseline, according to experiments carried out under three traffic scenarios: low (200 veh/hr), medium (500 veh/hr), and peak (900 veh/hr) and confirmed over five independent runs.

Reductions of 18.8%, 27.0%, and 18.8% are obtained with predictive WOA. Predictive information is crucial for efficient signal optimization under high congestion, as demonstrated by the reactive GA's 19.1% performance degradation under peak conditions.

With the best average performance and the least run-to-run variability in every case, HHO proves to be the most dependable algorithm.

Keywords: Traffic signal optimization, LSTM prediction, meta-heuristic algorithms, Whale Optimization Algorithm, Harris Hawks Optimization, Genetic Algorithm, rolling horizon control, SUMO simulation, Blida

Résumé

L'embouteillage urbain demeure l'un des enjeux les plus urgents de l'administration des villes contemporaines, provoquant de significatifs délais, un surcroît de consommation d'essence et une détérioration environnementale. L'approche prévalente dans les réseaux urbains algériens, les horaires fixes de contrôle, ne convient pas fondamentalement à la nature dynamique et imprévisible de la demande de trafic réelle. Cette recherche présente un cadre prédictif hybride basé sur des méta-heuristiques pour l'optimisation dynamique de la signalisation routière, en se basant sur un réseau routier concret comprenant neuf intersections dans la ville de Blida, Algérie.

Le modèle comprend trois éléments qui fonctionnent dans une boucle de contrôle à horizon glissant : un réseau de neurones à mémoire à long terme (LSTM) qui anticipe le délai d'attente total du réseau soixante secondes en avance, un optimiseur méta-heuristique basé sur la population qui détermine le meilleur plan de circulation à chaque étape de contrôle, et le simulateur de trafic détaillé SUMO relié par l'interface en temps réel TraCI. Trois techniques d'optimisation sont examinées : l'Algorithme d'Optimisation des Baleines (WOA) et l'Optimisation des Faucons de Harris (HHO), tous deux sous l'influence de la prédiction LSTM, ainsi qu'un Algorithme Génétique (GA) réactif employé en tant que référence sans prédiction.

Les essais réalisés dans trois scénarios de circulation — faible (200 véhicules par heure), moyen (500 véhicules par heure) et de pointe (900 véhicules par heure) — et validés sur cinq répétitions indépendantes, démontrent que le HHO Prédictif obtient des diminutions du temps d'attente de respectivement 27,9%, 38,2% et 34,6% comparativement à la référence à temps fixe.
Le WOA Prédictif enregistre des baisses de 18,8%, 27,0% et 18,8%. Le GA réactif peut entraîner une dégradation de performance allant jusqu'à 19,1% dans les conditions de pointe, témoignant directement que l'information prédictive est indispensable pour optimiser de manière efficace les signaux en situation de forte congestion.

Le HHO se distingue comme l'algorithme le plus solide, alliant les meilleures performances moyennes à la plus petite variabilité entre les mises en œuvre dans tous les contextes.

Mots-clés: Optimisation des feux de circulation, prédiction LSTM, algorithmes méta-

heuristiques, Algorithme d'Optimisation des Baleines, Optimisation des Faucons de Harris, Algorithme Génétique, contrôle à horizon glissant, simulation SUMO, Blida.

Contents

Dedication **I**

Acknowledgements **III**

الملخص **III**

Abstract **V**

Résumé **VI**

General Introduction **2**

1 Context and Research Problem **4**
1.1 Introduction 5
1.2 Motivation 5
1.3 Problem statement 6
1.4 Survey of related works 6
1.5 Objectives 10
1.6 Structure of the Thesis 11
1.7 Conclusion 11

2 Literature Review and Theoretical Background **12**
2.1 introduction 13
2.2 Urban Traffic Signal Control 13
2.2.1 Fixed-Time Control : Webster's Method 13
2.2.2 Actuated Control 14
2.2.3 Adaptive Control Systems: SCOOT and SCATS 14
2.2.4 Limitations of Classical Methods 15
2.3 Meta-Heuristic Optimization 15
2.3.1 Population-Based vs Single-Solution Approaches 16
2.3.2 Whale Optimization Algorithm (WOA) 16
2.3.3 Harris Hawks Optimization (HHO) 18

2.3.4 Genetic Algorithm (GA) . . . 20
2.4 Deep Learning for Traffic Prediction . . . 21
2.4.1 Traffic Prediction as a Time Series Problem . . . 21
2.4.2 Recurrent Neural Networks(RNN) . . . 22
2.4.3 Long Short-Term Memory (LSTM) Networks . . . 22
2.4.4 LSTM for Traffic Prediction : Related Work . . . 24
2.5 Positioning of the proposed framework . . . 25
2.5.1 Identified Research Gaps . . . 25
2.5.2 Our findings . . . 25
2.6 Chapter Conclusion . . . 26

3 Proposed Framework and Methodology 27
3.1 Introduction . . . 28
3.2 Framework Overview . . . 28
3.2.1 General Architecture . . . 28
3.2.2 Rolling Horizon Concept . . . 29
3.2.3 LSTM + Meta-Heuristic Integration . . . 29
3.3 SUMO Simulation Platform . . . 30
3.3.1 Overview of SUMO . . . 30
3.3.2 TraCI API . . . 30
3.3.3 Why SUMO for This Work . . . 31
3.4 Study Area : Blida Network . . . 31
3.4.1 OSM Map Extraction and Network Generation . . . 32
3.4.2 Network Properties and Traffic Demand Generation . . . 33
3.4.3 Traffic Scenarios : Low, Medium, Peak . . . 33
3.5 LSTM Predictive Module . . . 34
3.5.1 Input Features . . . 34
3.5.2 LSTM Architecture . . . 34
3.5.3 Training on Google Colab . . . 35
3.5.4 Prediction Results . . . 36
3.6 Fitness Function . . . 37
3.6.1 Background: Webster's Formula . . . 37
3.6.2 Red-Time Cost Formula . . . 37
3.6.3 Signal Bounds . . . 38
3.7 WOA Implementation . . . 38
3.7.1 Population Initialization . . . 38
3.7.2 Encircling Prey . . . 39
3.7.3 Bubble-Net Spiral Attack . . . 39
3.7.4 Exploration Phase . . . 39

3.8 HHO Implementation . . . 40
3.8.1 Escaping Energy Concept . . . 40
3.8.2 Four Exploitation Strategies . . . 40
3.8.3 Levy Flight . . . 41
3.8.4 Fix Applied: Signed Difference . . . 41
3.9 GA Implementation . . . 42
3.9.1 Tournament Selection . . . 42
3.9.2 Crossover and Mutation . . . 42
3.9.3 GA as Reactive Benchmark . . . 43
3.10 Rolling Horizon Control Loop . . . 43
3.10.1 Step-by-Step Algorithm . . . 43
3.10.2 Pseudocode . . . 44
3.10.3 LSTM Data Buffering Mechanism . . . 44
3.11 Chapter Conclusion . . . 45

4 Experiments and Results **46**
4.1 Introduction . . . 47
4.2 Experimental Setup . . . 47
4.2.1 Hardware and Software Environment . . . 47
4.2.2 Simulation Parameters . . . 47
4.2.3 Evaluation Metrics . . . 48
4.3 Baseline Results: Fixed-Time Control . . . 48
4.3.1 Baseline Performance Across Scenarios . . . 48
4.3.2 Baseline as Reference Point . . . 49
4.4 Results : Low Traffic Scenario (200 veh/hr) . . . 49
4.4.1 Waiting Time Comparison . . . 49
4.4.2 Four Metrics Analysis . . . 50
4.4.3 Discussion of Low Traffic Results . . . 50
4.5 Results : Medium Traffic Scenario (500 veh/hr) . . . 51
4.5.1 Waiting Time Comparison . . . 51
4.5.2 Four Metrics Analysis . . . 51
4.5.3 Discussion of Medium Traffic Results . . . 52
4.6 Results : Peak Traffic Scenario (900 veh/hr) . . . 52
4.6.1 Waiting Time Comparison . . . 52
4.6.2 Four Metrics Analysis . . . 53
4.6.3 Discussion of Peak Traffic Results . . . 53
4.7 Statistical Validation : 5 Independent Runs . . . 54
4.7.1 Mean and Standard Deviation Results . . . 54
4.7.2 Result Consistency Analysis . . . 54

4.8 Temporal Evolution Analysis 55
4.8.1 Waiting Time Over Simulation Time 55
4.8.2 Algorithm Behavior During Peak Congestion 55
4.9 Overall Comparison and Discussion 56
4.9.1 Improvement Percentage Across All Scenarios 56
4.9.2 Heatmap Analysis 56
4.9.3 Why GA Fails Under Heavy Traffic 57
4.9.4 Why HHO Outperforms WOA 58
4.10 Chapter Conclusion 58

5 Discussion, Conclusion and Perspectives **59**
5.1 Introduction 60
5.2 Comparison with Related Work 60
5.2.1 Meta-Heuristic Optimization Studies 60
5.2.2 Deep Learning and Prediction Studies 61
5.2.3 Limitations of the Proposed Framework 61
5.3 Future Perspectives 62
5.3.1 Extension to Larger Networks 62
5.3.2 Integration of Reinforcement Learning 62
5.3.3 Real-Time Deployment with Physical Sensors 63
5.4 General Conclusion 63

List of Figures

2.1 Webster's optimal cycle length C* as a function of total flow ratio Y [Webster,1958] 14
2.2 Taxonomy of meta-heuristic optimization algorithms 16
2.3 Flowchart of the whale optimization algorithm[MirjaliliLewis, 2016] 18
2.4 Flowchart of the Harris Hawks optimization algorithm [Heidari et al,2016] . . . 20
2.5 Flowchart of the Genetic Algorithm used as reactive benchmark 21
2.6 Architecture of a single LSTM cell [Hochreiter Schmidhuber, 1997] 24

3.1 General architecture of the proposed hybrid predictive meta-heuristic framework 28
3.2 Rolling horizon control strategy:the optimization window slides forward every 60 seconds 29
3.3 SUMO-GUI visualization of the Blida network during a medium-traffic simulation 30
3.4 Location of the study area: City center, Blida, Algeria (coordinates: 36.467°N – 36.475°N, 2.825°E – 2.836°E 32
3.5 Network reduction: from 95 detected intersections (left) to 9 selected intersections (right) 32
3.6 Traffic density comparison across the three simulation scenarios 34
3.7 Architecture of the LSTM predictive module 35
3.8 Evolution of MSE loss during the training and validation phases of the LSTM model 35
3.9 LSTM model predictions versus actual ground truth for 200 test samples 36
3.10 Convergence rate of the Whale Optimization Algorithm (WOA) over 10 iterations under the medium traffic demand scenario, showing mean best fitness and standard deviation. 40
3.11 Effect of the signed difference correction on HHO population diversity 42

4.1 Four performance metrics comparison for the Low traffic scenario (mean ± std, 5 runs) 50
4.2 Four performance metrics comparison for the Medium traffic scenario (mean ± std, 5 runs) 52
4.3 Four performance metrics comparison for the Peak traffic scenario (mean ± std, 5 runs) 53

4.4 Temporal evolution of total network waiting time for all four methods under Low (left), Medium (center), and Peak (right) traffic scenarios 55
4.5 Waiting time improvement percentage relative to fixed-time baseline for all three methods across three traffic scenarios (mean ± std, 5 runs) 56
4.6 Heatmap of average waiting time: rows = methods, columns = scenarios. Red = high congestion, green = low congestion 57

List of Tables

1.1 Comparative Overview of Existing Approaches for Traffic Signal Optimization and Prediction 6

3.1 Traffic Scenario Parameters 33

4.1 Software Environment 47
4.2 Simulation and Optimization Parameters 48
4.3 Baseline Fixed-Time Performance Across Three Traffic Scenarios 49
4.4 Single run results: Low traffic scenario 49
4.5 Statistical results (mean $\pm$ std, 5 runs): Low traffic scenario 50
4.6 Single run results: Medium traffic scenario 51
4.7 Statistical results (mean $\pm$ std, 5 runs): Medium traffic scenario 51
4.8 Single run results: Peak traffic scenario 52
4.9 Statistical results (mean $\pm$ std, 5 runs): Peak traffic scenario 53
4.10 Complete statistical summary: mean $\pm$ std over 5 independent runs 54
4.11 Summary of best results (HHO) across all scenarios 58

General Introduction

One of the key issues of the twenty-first century is urban mobility. The road network, a permanent physical infrastructure, must meet an ever-increasing demand that it was not intended to handle as cities expand and the number of vehicles rises. This conflict is especially evident in Algerian cities: daily traffic peaks in Blida, Algiers, Oran, and Constantine result in substantial financial losses, higher fuel consumption, environmental damage, and a decline in the standard of living for their residents. The traffic light, a seemingly straightforward instrument, is at the centre of this issue.

The main instrument used to control urban traffic flow at intersections is traffic signals. Whether a network moves easily or comes to a complete stop depends on their timing, which includes how long each direction receives green light, how frequently the cycle repeats, and how well nearby crossings are coordinated. However, traffic lights continue to run on fixed-time plans in the great majority of Algerian urban networks. These are preset schedules that are calculated from past averages and implemented consistently throughout the day, regardless of the actual demand at any particular time. During morning rush hour, during special events, or after an incident that causes traffic to be redistributed around the network, a plan calibrated for regular noon conditions will unavoidably be less than ideal.

Since fixed-time control's shortcomings have been known for decades, a number of more complex alternatives have been put forth by the global research community. When a vehicle is recognised, vehicle-actuated systems extend green phases. SCOOT and SCATS are examples of network-level adaptive systems that continuously modify signal timing in response to real-time detector data. A new generation of methods that approach signal optimisation as a search problem, one that can be solved at any given time by an algorithm that compares thousands of potential plans to a measure of network performance, have emerged more recently due to the development of machine learning and computational intelligence.

This thesis is part of the new generation. Using an actual nine-intersection road network in the city of Blida, it suggests and assesses a hybrid predictive meta-heuristic framework for dynamic traffic signal optimisation. A population-based meta-heuristic optimizer, either the Whale Optimisation Algorithm (WOA) or the Harris Hawks Optimisation (HHO), uses the Long Short-Term Memory (LSTM) neural network, which learns from past traffic data to predict future congestion sixty seconds ahead, to find the signal plan that will best handle the expected demand. To separate and measure the predictive component's contribution, a reactive Genetic Algorithm

(GA) that operates without prediction is used as a benchmark.

This work's main premise is that predictive information is crucial for signal optimisation, not only helpful. A controller will constantly lag behind the dynamics of a saturated network if it can only respond to congestion after it has developed. The cascading effects that turn mild congestion into gridlock can be avoided by a controller that anticipates what will happen and takes action before lines form. This theory is supported by the experimental data shown in this thesis: under peak traffic conditions, the LSTM-guided HHO reduces average waiting time by more than 38%, while the reactive GA decreases network performance by about 20% compared to the fixed-time baseline.

Beyond its technical contributions, this effort is driven by a practical goal: to show that advanced intelligent traffic control is not limited to well-funded cities with substantial sensor infrastructure. Mid-sized Algerian cities can achieve intelligent urban mobility through a framework based solely on open-source tools, such as SUMO, Python, and PyTorch, applied to an actual road network taken from publicly available OpenStreetMap data, validated under realistic traffic conditions.

Chapter 1

Context and Research Problem

1.1 Introduction

Most people deal with traffic congestion on a daily basis. Roads are becoming more congested, cities are growing, and more people are purchasing automobiles. The traffic signal is one of the primary instruments used to manage this traffic, and when it functions properly, cars move. When it does not, a single substandard road intersection can slow down a neighborhood.

The rising issue is that traffic signals use is maintained in many places, particularly in Algeria, in the same manner as they were many years ago , It remains unchanged. It makes no difference if the road is empty or totally blocked. Simply said, the light keeps circulating in the same dynamic.

This causes serious issues on a daily basis in a developing cities, particularly during rush hour in the morning and evening.

This chapter highlights the main objective, discusses the problem under investigation, outlines its key deficiencies in this study and the rationale for addressing them. Finally a brief overview of the structure of the remainder of the document is provided .

1.2 Motivation

Traffic congestion is a human issue as much as a technical one. Individuals' time is diminished by every additional minute spent waiting at a red light. The scope of the issue becomes evident when multiplying that by thousands of drivers, hundreds of crossings, and 365 days a year.

The fact that the congestion is frequently preventable makes it frustrating. It is caused by signals that are unable to react to the circumstances in front of them, not by an excessive number of vehicles or a shortage of roads. A road with no traffic receives the same red light as one with a lot of traffic. The system is still caught unexpectedly by a daily rush hour. This type of waste, of time, gasoline, and patience, occurs not because the issue is intractable but rather because the appropriate instruments. Because the adequate instruments were not appropriately implemented.

Therefore, the cost extends beyond irritation. Everyone is affected by traffic issues, not just drivers, as they impact the city's air quality, fuel usage, and pollution. It usually takes longer for emergency services to get to those in need.

Additionally, as the number of automobiles continues to rise annually, the longer a community waits to handle issue, the more complicated it gets.

This is the reason this work is worthwhile. To demonstrate that cities can have more intelligent traffic management, beginning today with the roads and intersections that currently exist, rather than to add yet another theoretical research to the pile.

1.3 Problem statement

Developed and adequately functioning traffic signal knows exactly how many cars are approaching before they even arrive and automatically adjusts itself to keep traffic moving. However, these advanced features have not seen the light in Algerian cities Instead, there are signals that are blind, inflexible, and totally detached from the realities of the road.

The signals' age is not the only issue. The reason is that they were not made to deal with uncertainty. When traffic is steady and predictable, a fixed time schedule works well.
However, urban traffic is neither of those. No two days are precisely the same, rush hours change, unforeseen occurrences occur, and the weather affects behavior. Small imbalances at one intersection rapidly develop into more serious issues that affect the entire network when the signal is unable to adjust to any of this.

This is especially insinuating that traffic is more than simply an annoyance. More fuel is burnt, more emissions are emitted, and more time is lost from a person's day for each additional minute a car waits at a red light. When multiplying that by thousands of cars and hundreds of crossroads, the whole cost, economic, environmental, and human becomes immense.

The question this thesis raises is straightforward: Is it possible to systematically adjust traffic signals to function adequately? Just more intelligent choices about when to give green and when to halt, without the need for costly infrastructure or additional roadways. This investigation will demonstrate that traffic signal can precisely predict the circulation of crowded intersections, and adjust itself accordingly. However, to achieve this , systems that just respond to past events must be replaced with ones that can truly anticipate future events

1.4 Survey of related works

This thesis is synthesizing and reflects on the work of numerous researchers who have attempted to address the traffic signal issue from various perspectives over the years.

Table 1.1: *Comparative Overview of Existing Approaches for Traffic Signal Optimization and Prediction*

Authors	**Title**	**Year**	**Model Architecture**	**Input Type**	**Dataset**
H. M. Abdulhai et al.	Short Term Traffic Flow Prediction Using Neuro-Genetic Algorithms	2002	Neuro-Genetic TDNN synthesized by GA	Temporal profiles and spatial data upstream/downstream	ATMS Testbed Orange County CA and simulated data
Zainab Abbas et al.	Short-Term Traffic Prediction Using Long Short-Term Memory Neural Networks	2018	Partitioned LSTM Neural Networks	Traffic flow and speed data per lane	Stockholm Motorway Control System Radar sensors
Luntian Mou et al.	A Long Short-Term Memory Neural Network Enhanced by Temporal Information for Traffic Flow Prediction	2019	T-LSTM Temporal Information Enhancing LSTM	Temporal features and short-term traffic flow	Real-world road section traffic data

Authors	Title	Year	Model Architecture	Input Type	Dataset
Andrea Vidali et al.	A Deep Reinforcement Learning Approach to Adaptive Traffic Lights Management	2019	Deep Reinforcement Learning DRL + SUMO	Real-time regulation actions and traffic states	Microscopic agent-based simulation SUMO
Thaer Thahe et al.	Whale Optimization Algorithm for Traffic Signal Scheduling Problem	2019	Whale Optimization Algorithm WOA	Average Travel Time ATT and vehicle activity parameters	Large-scale traffic benchmarks
Arshad Jamal, Muhammad Tauhidur Rahman, et al.	Intelligent Intersection Control for Delay Optimization: Using Meta-Heuristic Search Algorithms	2020	Genetic Algorithms GA and Differential Evolution DE	Vehicle arrival rates and lane capacity	Traffic data from isolated intersections
ABOLGHASEM S. et al.	Short-Term Traffic Flow Prediction Using the Modified Elman Recurrent Neural Network Optimized Through a Genetic Algorithm	2020	GA-MENN Modified Elman RNN + Genetic Algorithm	Weather, weekday, hour, and day classification	Google Map API data 139 routes in Tehran
Haifeng Zheng et al.	A Hybrid Deep Learning Model With Attention-Based Conv-LSTM Networks for Short-Term Traffic Flow Prediction	2020	Attention-based Conv-LSTM + Bi-LSTM	Flow sequences spatial and temporal features	Real-world short-term traffic flow data
Tuo Mao et al.	Boosted Genetic Algorithm using Machine Learning for traffic control optimization	2021	BGA-ML Genetic Algorithm + Extreme-Gradient Decision-Tree	Phase duration and total travel time	Urban intersection network non-recurrent incidents
Nv Er Ren et al.	Short-term Traffic Flow Prediction Based on Deep Learning Model	2021	LSTM Long Short-Term Memory	Short-term traffic flow holidays and working days	Microscopic simulation software
Xiaoyi Ma, Xiaowei Hu, Thomas Weber, and Dieter Schramm	Evaluation of Accuracy of Traffic Flow Generation in SUMO	2021	ActivityGen activity-based demand generation tool within SUMO	Geographic population distribution data gathered from public information	Traffic simulation of the Jianghan Zone in Wuhan China
Arshad Jamal, Hassan M. Al-Ahmadi, et al.	Metaheuristics for Traffic Control and Optimization: Current Challenges and Prospects	2021	Metaheuristics	Signal timing parameters	Literature review and complex urban case studies
Maria Laura Clemente	Building a real-world traffic micro-simulation scenario from scratch with SUMO	2022	SUMO	Network Files	Network Data
Xing Xu et al.	Short-term traffic flow prediction based on whale optimization algorithm optimized BiLSTM Attention	2022	WOA-BiLSTM-Attention	Traffic flow data and hyperparameters	Urban traffic flow dataset
Sayantan Kar	Optimization of Traffic Signals for Intelligent Traffic Light Management System	2023	Fuzzy Logic, CNN + RL	Traffic Density	Simulation Data
Mohammed A. Khasawneh and Anjali Awasthi	Intelligent Meta-Heuristic-Based Optimization of Traffic Light Timing Using Artificial Intelligence Techniques	2023	Enhanced Bat Algorithm EBAT + LSTM/RNN/ARIMA	Predicted congestion levels and vehicle waiting times	Montreal traffic dataset partitioned into zones
Hongluan Zhao and Mengmeng Su	Traffic Signal Timing Scheme Based on the Improved Harris Hawks Optimisation	2024	Improved Harris Hawks Optimisation HHO	Average delay, vehicle stops, capacity, and exhaust emissions	Urban intersection peak and low-peak traffic flow
Yiran Zhang, Mingjian Fu, and Xuegang Jeff Ban	Integration Traffic Signal Control From Synchro to SUMO	2024	SUMO	Traffic Data	Traffic network
Axel Schaffland, Jonas Nelson, and Julius Schoning	Simulating Traffic Networks: Driving SUMO Towards Digital Twins	2024	SUMO	Static Data	Simulation Data
Dhruv Agrawal, D. Vinod, and Juhi Rastogi	A SUMO Based Simulation Framework for Intelligent Traffic Management System	2024	Testing environment for AEB Automatic Emergency Braking algorithms	OSM map tiles and aggressive driving behavior parameters	Digital twin of the SRM University campus
Paul Heckelmann and Stephan Rinderknecht	Influence of an Automated Vehicle with Predictive Longitudinal Control on Mixed Urban Traffic Using SUMO	2024	Predictive Longitudinal Control	V2X data position, speed of surrounding vehicles and speed profiles	Real road network of Darmstadt, Germany

Authors	Title	Year	Model Architecture	Input Type	Dataset
Erik Vogel	Training and evaluating deep learning models on road graphs for traffic prediction using SUMO	2024	Graph Attention Network	Average vehicle speeds and temporal flow data	DeepSUMO framework
O. Dobrilko and A. Bublil	Leveraging SUMO for Real-World Traffic Optimization: A Comprehensive Approach	2024	SUMO Microscopic Simulation	Sensor data trajectories and road user types	Intersection geometries and driver behavior data
Z. Liu et al.	An improved whale optimization algorithm with recurrent neural networks for traffic flow prediction	2025	IWOA-RNN Improved WOA + Recurrent Neural Network	Real-time traffic flow	Guangzhou highway traffic dataset
Marwa Elharoun, Sherif M. El-Badawy, et al.	Adaptive traffic signal control using deep reinforcement learning: A multi-objective approach for single and multi-intersection scenarios	2025	Deep Reinforcement Learning	Queue lengths, vehicle positions, and safety metrics	Synthetic intersections and real-world scenarios simulated in SUMO

Abdulhai et al. (2002) combined temporal and spatial data to estimate short-term traffic flow using neuro-genetic algorithms. Their results showed that the model performed better than traditional Multi-Layer Feedforward networks in both simulated and real-world scenarios, and that prediction errors decrease as the model's spatial contribution increases.[1]

Inspired by this, Abolghasem et al. (2020) suggested a Modified Elman Recurrent Neural Network optimized by a Genetic Algorithm. By accounting for environmental factors including weather, weekday, and time of day, their model outperformed fifteen other approaches, such as CNN and standard LSTM, for urban road speed projections.[2]

Nv Er Ren et al. (2021) demonstrated that a standard LSTM can estimate short-term traffic flow while distinguishing between working-day and holiday patterns, demonstrating its usefulness for dynamic control in highway management systems as this field of study progressed.[3]

Before using different predictors, Abbas et al. divided the road network into segments and intersections,improved LSTM and reduced the amount of input sensors required by 65% without compromising accuracy, increasing accuracy by a factor of five.[4]
Mou et al. refined this idea in two studies; by using temporal information vectors, their 2019 T-LSTM produced higher prediction accuracy than state-of-the-art methods and successfully retrieved lost data.[5]
Soon after, Zheng et al. (2020) combined Attention-based Convolutional LSTM with Bidirectional LSTM to concurrently capture complex nonlinear and periodic spatial-temporal variables, outperforming earlier techniques in accuracy.[6]

Another group of researchers focused on using the predictions to improve signal timing while prediction techniques were evolving. Jamal et al. (2020) applied both GA and Differential Evolution to real-world crossings in Saudi Arabia with non-linear, stochastic traffic patterns, reducing delays by 15 to 35% compared to existing conditions.[7]
Additionally, their 2021 study confirmed that meta-heuristics are helpful substitutes for traditional techniques in large-scale optimization, especially for creating Pareto-optimal fronts that balance delays and emissions[8]

At the same time, Thahe et al. (2019) employed WOA to address the traffic signal scheduling issue in three progressively larger networks. The findings demonstrated that WOA performed better than other rival algorithms in reducing Average Travel Time, with the advantage growing

in larger networks.[9]

Going one step further, Zhao and Su (2023) proposed an improved HHO with multi-strategy enhancements that concurrently optimized four objectives: emissions, capacity, delay, and stops. During peak hours, vehicle delays dropped by 36.7%, exhaust emissions dropped by 31.2%, and vehicle capacity increased by 41.6%.[10]

In an alternative approach that is equally fascinating, Mao et al. (2021) combined machine learning regressors with genetic algorithms to replace sluggish simulation-based evaluation with fast predictions. This led to a significantly quicker approach that remained effective even in the face of serious, infrequent traffic incidents.[11]

Furthermore, Kar (2023) concluded that AI techniques are necessary for adaptive signal timing that also manages emergency vehicle priority after examining AI-based traffic management tactics on significant Indian highways.[12]

Khasawneh and Awasthi (2023) brought prediction and optimization closer together by using an Enhanced Bat Algorithm for signal optimization with LSTM/RNN/ARIMA prediction. By predicting traffic levels ahead of time, their approach significantly reduced travel times, increased throughput, and reduced polluting emissions.[13]

On the simulation side, a similar line of inquiry looked into how to test these ideas in practical contexts. Vidali et al. (2019), who were among the first to integrate Deep Reinforcement Learning with SUMO, proved the effectiveness of agent-based simulation for adaptive control and verified that the technique finds suitable regulating actions.[14]
Following that, Ma et al. (2021) discovered that SUMO's ActivityGen tool, which creates traffic flows using spatial population data from an actual zone in Wuhan, China, is a helpful substitute for expensive traditional data collection.[15]
Then, by demonstrating how to create a real-world micro-simulation from scratch, Clemente (2022) shown that SUMO is a very powerful and portable tool for large-scale micro-simulations, even for entire cities.[16]

Agrawal et al. created a digital model of an Indian campus with aggressive and disorganized driving behaviors (2024) extended this methodology and verified SUMO for Automatic Emergency Braking system testing.[17]
In a parallel project, Schaffland et al. (2024) demonstrated how to use both static and dynamic city sensor data to generate a digital twin in SUMO, emphasizing the importance of accuracy and data density.[18]
Near this, Zhang et al. (2024) demonstrated that signal timing plans could be converted to SUMO with roughly 85% semi-automation, lowering costs and improving platform compatibility.[19]

Heckelmann and Rinderknecht (2024) examined a single autonomous vehicle with predictive longitudinal control utilizing SUMO, adding an additional dimension. They found a 250-meter-long area of impact that affected around four nearby cars.[20]

To wrap up this topic, Dobrilko and Bublil (2024) used real sensor data on vehicle trajectories and driver behavior to provide a comprehensive real-world proof of SUMO's effectiveness in bridging theory and field application.[21]

El Haroun et al. (2025) employed a PPO-based Deep Reinforcement Learning agent for adaptive signal control, which led to a 7% reduction in delays and a 21% reduction in traffic conflicts, with decentralized control outperforming centralize methods.[22]
In a related project, Xu et al. (2022) improved the hyperparameters of a BiLSTM-Attention network for short-term traffic prediction using WOA, outperforming both conventional and unoptimized models in all evaluation metrics.[23]

Finally, Liu et al. (2025) improved WOA with Tent chaos mapping and reverse learning to optimize an RNN for highway traffic prediction, exceeding traditional methods in terms of convergence speed and accuracy.[24]

1.5 Objectives

The goal of this thesis is to offer a practical solution to one of the most enduring issues in urban transportation: the present traffic signal systems' incapacity to adjust to actual and shifting traffic conditions. The objective is to create a system that tries to prevent congestion before it occurs rather than just responding to it by fusing intelligent optimization with contemporary prediction tools. The specific objectives of the effort are as follows :

- Develop a reliable short-term traffic flow prediction model that can accurately predict how many cars will arrive at crossings while taking past simulation data, traffic patterns, and the time of day into consideration.

- Construct a successful signal control plan that uses predicted traffic data to modify signal timings ahead of time, reducing needless wait times while also improving traffic flow at several crossings.

- Build a realistic simulation environment based on actual road network so that the suggested solution may be tested and assessed in realistic traffic scenarios.

- To prove with hard data, that a predictive system outperforms a fixed or merely reactive one, compare the performance of the suggested strategy with current baseline techniques.

- Provide a completely replicable and documented system that may be used as a foundation for future research or practical implementation in Algerian cities

1.6 Structure of the Thesis

This thesis is organized into five chapters, each of which builds upon the one before it to progressively develop, apply, and assess the suggested framework:

Chapter 1 : Context and research problem outlines the general background and motivation for this study, summarises the research problem and objectives, examines the body of literature in thirty carefully chosen publications on traffic signal control, meta-heuristic optimisation, deep learning for traffic prediction, and simulation platforms, and describes the thesis's structure.

Chapter 2 : Literature Review and Theoretical Background gives the theoretical underpinnings of the suggested paradigm. From Webster's fixed-time formula to vehicle-actuated systems to network-level adaptive controllers like SCOOT and SCATS, it charts the development of traffic signal control. The three optimisation methods employed in this study, WOA, HHO, and GA, as well as the LSTM architecture used for traffic prediction, are then explained mathematically.

Chapter 3 : Proposed Framework explains the hybrid predictive meta-heuristic framework's entire technique. The study area, the Bab Dzair district of Blida city, the SUMO simulation environment and TraCI interface, the LSTM predictive module and its training outcomes, the red-time cost fitness function, each algorithm's implementation, and the rolling horizon control loop that unifies everything are all presented.

Chapter 4 : Experiments and Results demonstrates the experimental assessment of the framework using four techniques: the fixed-time baseline, reactive GA, predictive WOA, and predictive HHO, under three traffic scenarios: low, medium, and peak. The results are statistically confirmed over five independent runs per technique per scenario and are shown for four performance metrics: average waiting time, number of stops, total travel duration, and throughput.

Chapter 5 : Discussion, Conclusion and Perspectives places the findings in the context of the larger body of scientific literature, notes the limitations of the suggested methodology, summarises the thesis's key contributions, and suggests three avenues for further investigation: real-time deployment with physical sensors, integration of reinforcement learning, and extension to larger networks.

1.7 Conclusion

This chapter laid the groundwork for everything that follows. It addressed the importance of this topic, gave a precise explanation of the problem, looked at the work of other experts in the field, set objectives, and talked about the thesis's structure. The next chapter explores the theoretical concepts and tools that support the work presented in this thesis after this foundation has been established.

Chapter 2

Literature Review and Theoretical Background

2.1 introduction

A detailed overview of the background theory and associated literature supporting this framework is presented within this Chapter. An examination of the history of Urban Traffic Signal Control is provided beginning at Classical Fixed-Time Methods through Modern Adaptive Systems. Then, meta-heuristic optimization algorithms will be discussed in detail, with specific emphasis upon the three meta-heuristic optimization algorithms being employed in this study: The Whale Optimization Algorithm (WOA); the Harris Hawks Optimization (HHO); and the Genetic Algorithm (GA). Finally, a summary of recent Deep Learning Techniques for Short-Term Traffic Prediction will be provided, with an emphasis on LSTM Networks.

2.2 Urban Traffic Signal Control

Traffic signals have been a part of urban infrastructure for almost a century. It is a simple but difficult task to decide which direction gets the green light, for how long, and when. A wise choice keeps traffic flowing. A poor one causes lines to lengthen, widen, and take a long time to clear. The process by which these choices are decided has evolved dramatically throughout the years. There are now three primary strategies, all of which aim to address the issues that the previous strategy left behind.[25]

2.2.1 Fixed-Time Control : Webster's Method

Fixed-time control is the oldest and still most popular method. This method applies a signal timing plan to the intersection on a predetermined schedule after it has been calculated in advance using previous traffic data. What is happening on the road has no effect on the signal. Every day, it simply adheres to its predetermined schedule. Frank Webster made the most significant addition to this method in 1958 when he developed a formula to determine the ideal cycle length for a signalized intersection [26]. The formula for Webster is:

$$C^* = \frac{1.5L + 5}{1 - Y}$$

where Y is the total of the critical flow ratios for each phase, L is the total lost time each cycle, and C* is the ideal cycle duration in seconds. After that, each phase's green time is split proportionately according to its share of Y [26]. Because Webster's work provided traffic engineers with a mathematical foundation for signal timing for the first time, it was revolutionary. His formula is still widely used as a standard reference in traffic engineering today [27]. One basic flaw in fixed-time control, though, is that it makes the assumption that traffic demand is predictable and steady. In actuality, traffic is always changing. A pre-programmed plan could go utterly wrong at that particular moment due to an event, an accident, or a school letting out. Our work builds the fitness function for our optimization algorithms using the ideas from Webster's formula, particularly the relationship between cycle length, green time, and demand, which is explained in depth in Chapter 3.

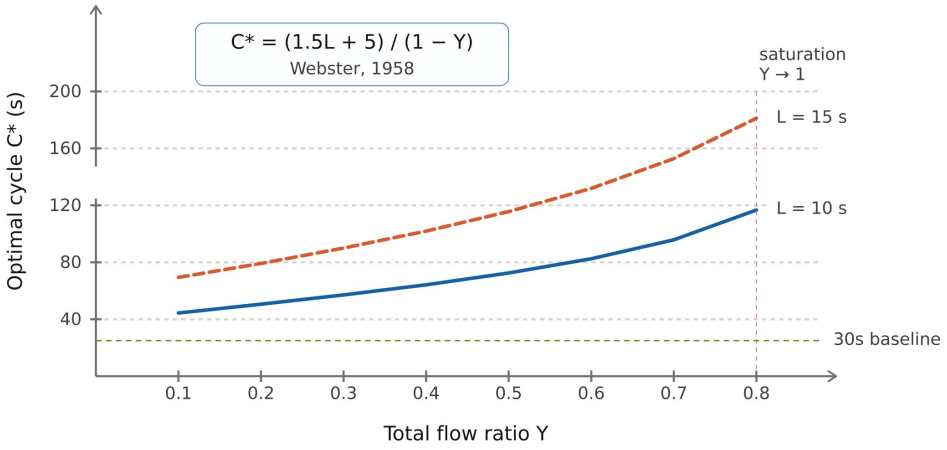


Figure 2.1: *Webster's optimal cycle length C* as a function of total flow ratio Y [Webster,1958]*

2.2.2 Actuated Control

Vehicle-actuated control was an improvement over fixed-time designs. Actuated systems use detectors, typically inductive loops buried in the road surface, to determine whether or not vehicles are truly waiting, as opposed to adhering to a strict schedule. The green remains on the main road if no cars are seen on a side street. The signal reacts when a car pulls up. There are two types of this: semi-actuated control, where only the smaller approaches are detector-controlled, and fully actuated control, where every approach responds dynamically to the presence of a vehicle [27]. Because actuated control responds to actual demand rather than past averages, it is superior to fixed-time plans. However, it still has a glaring drawback: it only reacts to existing automobiles. It has no way of knowing that two streets away, a surge of cars is going to arrive. Instead of being predictive, it is reactive.

2.2.3 Adaptive Control Systems: SCOOT and SCATS

In the 1970s and 1980s, researchers created network-level adaptive systems due to the shortcomings of both fixed-time and actuation control. SCOOT and SCATS are the two most widely used systems worldwide.

SCOOT (Split Cycle Offset Optimization Technique) was created by the Transport Research Laboratory in the United Kingdom. In order to minimize a combined index of delay and stops, it builds a continuous model of traffic flow using detector data from every link in the network and performs tiny, frequent changes to cycle durations, splits, and offsets—every few seconds. Importantly, decisions made at one intersection take into account what would happen downstream at the next since SCOOT functions at the network level[28].

SCATS (Sydney Coordinated Adaptive Traffic System) was developed in Australia and uses a different approach. Rather than continuously improving a model, it selects signal designs from a pre-computed library based on real-time traffic saturation readings. Cycle length synchronization allows for coordination between adjacent intersections [25].

When compared to fixed-time plans under typical circumstances, these approaches, which have been implemented in hundreds of cities worldwide, regularly reduce average delays by 10 to 20 percent [29].

However, this argument is directly motivated by a critical fault in both SCOOT and SCATS: they are merely reactive. They make changes based on what they observe at the moment because

they are unable to predict the future. In crowded or quickly changing environments, this inability to predict future events may lead to decisions that are already outdated by the time they are put into practice.

2.2.4 Limitations of Classical Methods

When examining all of the aforementioned approaches, three issues persist that none of them completely resolve:

1. **Static assumptions:** The foundation of fixed-time programs is historical averages. The strategy is instantly incorrect when traffic deviates from those averages, which occurs daily [26].

2. **Reactive only operation:** Even the most advanced adaptive systems, like SCOOT and SCATS, only respond to what is happening right now. They are unable to foresee what will occur next [25, 28] .

3. **Local or limited coordination:**The majority of approaches either coordinate only by synchronizing cycle durations or optimize a specific intersection in isolation. These methods cannot achieve true network-wide optimization, which optimizes the signal plan for every intersection as a separate system [25].

All three of these issues are directly addressed by our system. It can look ahead thanks to LSTM networks. All crossings are simultaneously optimized as a single system by the meta-heuristic techniques. The rolling horizon technique also ensures that the plans are revised in response to new facts.

2.3 Meta-Heuristic Optimization

Before traffic signal performance can be enhanced, thousands of possible timing combinations must be investigated to find the best option. To overcome this difficulty, meta-heuristic algorithms were created.

A meta-heuristic is a general-purpose search technique that intelligently examines a broad solution space without requiring knowledge of the problem's precise mathematical structure. Gradients and a precisely stated equation are not required. All it needs is a fitness function to gauge how good a solution is, after which it looks for ever-better ones. Because of this, meta-heuristics are perfect for issues like traffic signal timing, where there is a large solution space and a highly nonlinear and difficult-to-model relationship between signal plans and actual traffic results.

This thesis makes use of three meta-heuristic algorithms: the Genetic Algorithm (GA)[30], the Harris Hawks Optimization (HHO)[31], and the Whale Optimization Algorithm (WOA)[29]. The two primary optimizers that are incorporated with the LSTM predictor are WOA and HHO. In order to precisely quantify the contribution of the prediction component to the final performance, GA is used as a reactive benchmark, using the identical configuration but without prediction.

2.3.1 Population-Based vs Single-Solution Approaches

There are two families of meta-heuristic algorithms. Single-solution techniques, such as Tabu Search and Simulated Annealing, begin with a single potential solution and work to gradually enhance it. Population-based techniques, such as GA, WOA, and HHO, simultaneously maintain and collectively evolve a set of potential solutions [30, 31, 29] .

For a problem as complicated as multi-intersection signal timing, population-based approaches often explore the solution space more widely and are less likely to become trapped in a local optimum. This thesis's three algorithms are all members of the population-based family.

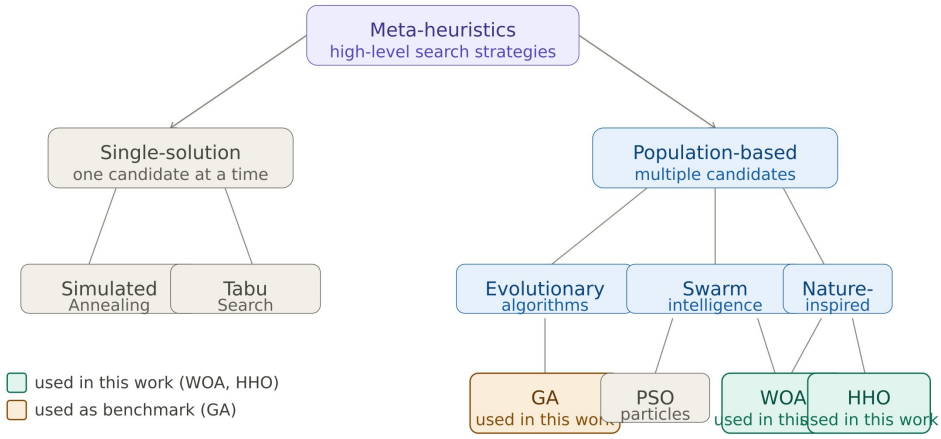


Figure 2.2: *Taxonomy of meta-heuristic optimization algorithms*

2.3.2 Whale Optimization Algorithm (WOA)

Origin and Biological Inspiration

The Whale Optimization Algorithm was suggested in 2016 by Mirjalili and Lewis, inspired by the hunting behaviors of humpback whales [29]. Humpback whales are known to use a unique hunting strategy called "bubble-net feeding," which involves circling their prey and releasing a spiral of bubbles to crowd fish into a tight ball near the surface before launching an assault [29].

Near the surface, humpback whales like to hunt small fish or schools of krill. They form separate bubbles along a circle or "9"-shaped path as they swim around the object inside a thin circle [29]. This method, known as "bubble-net feeding," entangles the prey in a contracting spiral of bubbles before the whale jumps up to capture it.

Closing in on known prey (exploitation) and scouring the ocean for new prey when the current site is not promising (exploration) are two natural actions that are included in the entire process. These behaviors were converted into a mathematical optimization framework by Mirjalili and Lewis [29]. Six structural design challenges and 29 mathematical optimization tasks were used to test WOA. Optimization results demonstrate how competitive the WOA algorithm is when compared to both traditional techniques and cutting-edge meta-heuristic algorithms. The WOA algorithm's source codes are openly accessible [29].

Mathematical Formulation

Three position update techniques are used by WOA [29]:

1. **Encircling Prey (Exploitation):** Each candidate solution, called a whale, moves toward the best solution found so far. The updated position is defined as:

$$X(t+1) = X^*(t) - A \cdot D$$

where:

$$D = |C \cdot X^*(t) - X(t)|$$
$$A = 2a \cdot r_1 - a$$
$$C = 2 \cdot r_2$$

where D is the distance to the optimal solution, A is a coefficient that decreases linearly from 2 to 0 throughout the iterations, and C is a random weight coefficient.

The variables r_1 and r_2 are random values within the interval $[0, 1]$. As the value of A decreases, the whale gradually spirals toward the prey.

2. **Bubble-Net Attack (Exploitation-Spiral Update):** This phase simulates the spiral movement of whales around their prey. The position update is expressed as:

$$X(t+1) = D' \cdot e^{bl} \cdot \cos(2\pi l) + X^*(t)$$

where:

$$D' = |X^*(t) - X(t)|$$

where D' represents the distance between the current whale and the best whale, b is a constant defining the logarithmic spiral shape (often set to 1), and l is a random number in the interval $[-1, 1]$. At each iteration, the algorithm probabilistically selects either the spiral update mechanism or the encircling mechanism in order to maintain a balance between exploitation strategies.

3. **Search for Prey (Exploration):** When $|A| > 1$, the whale updates its position relative to a randomly selected whale rather than the best solution. This mechanism encourages exploration of new regions within the search space and prevents premature convergence.

Applications in Traffic Engineering

WOA has been successfully used to optimize traffic signals. Thahe et al. (2019) demonstrated that WOA significantly outperformed PSO, GA, Bat Algorithm, and Salp Swarm Algorithm in minimizing Average Travel Time when applied to the traffic signal scheduling problem across three networks of increasing size, with the advantage growing in bigger networks [9].

Xu et al. (2022) outperformed all baselines in every assessment metric by using WOA to adjust the hyperparameters of a BiLSTM-Attention model for traffic flow prediction [23]. Each whale in this work is a candidate signal plan, which is a vector of all regulated intersections' green time durations. Before each plan is implemented in the simulation, the fitness function assesses it using LSTM-predicted traffic states.

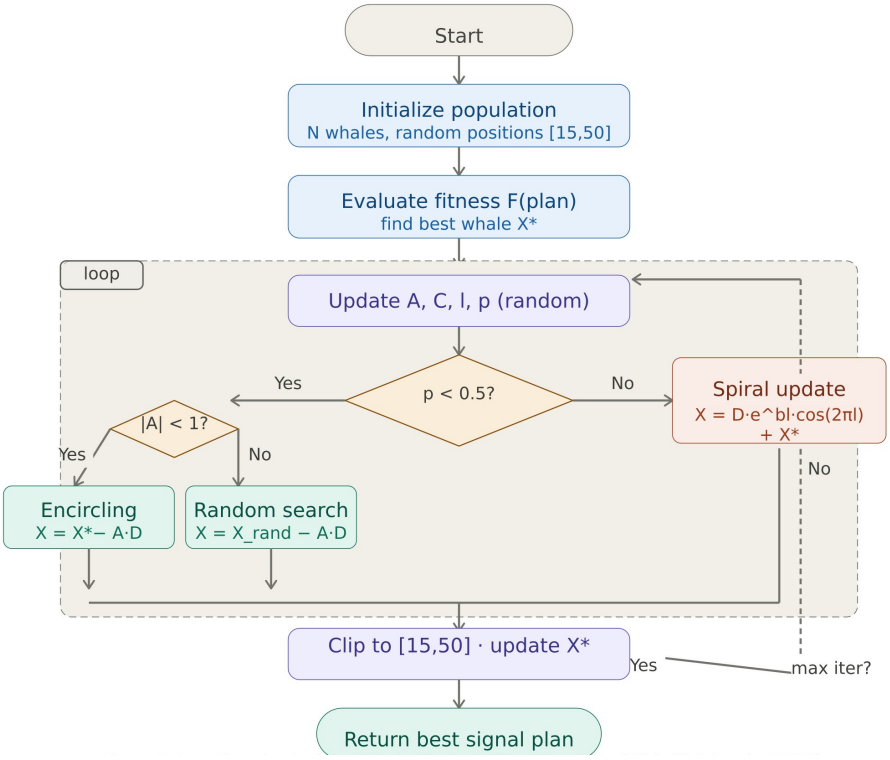


Figure 2.3: *Flowchart of the whale optimization algorithm[MirjaliliLewis, 2016]*

2.3.3 Harris Hawks Optimization (HHO)

Origin and Biological Inspiration

The Harris Hawks Optimization algorithm was put forth in 2019 and published in Future Generation Computer Systems, Volume 97, pages 849–872 [31], by Ali Asghar Heidari, Seyedali Mirjalili, and associates. The cooperative conduct and surprise pounce chasing style of Harris' hawks in the wild serve as the primary source of inspiration for HHO. In this ingenious strategy, several hawks cooperate to surprise a victim by attacking it from different directions. Harris hawks can exhibit a variety of chase behaviors depending on the dynamic nature of the scenario and the prey's escape options [31].

While other raptors typically assault to find and capture a quarry alone, Harris's hawk stands out for its unusual cooperative foraging behaviors with other family members residing in the same stable group. By tracking, encircling, flushing out, and ultimately attacking the possible prey, this avian desert predator demonstrates evolved creative team pursuing skills [31].

HHO is a popular swarm-based, gradient-free optimization algorithm with several active and time-varying phases of exploration and exploitation. The source codes of HHO are publicly available.

Mathematical Formulation

The main concept of HHO is based on the escape energy E of the prey (the rabbit), which decreases progressively over the iterations. The escape energy is formulated as[31]:

$$E = 2 \cdot \left(1 - \frac{t}{T}\right) \cdot E_0$$

where T is the maximum number of iterations, t is the current iteration, and E_0 is a random value within the interval $[-1, 1]$.

When $|E| \geq 1$, the hawks operate in exploration mode, where they either perch randomly or move relative to other hawks. Conversely, when $|E| < 1$, the hawks switch to exploitation mode and apply one of four attacking strategies depending on the rabbit escape probability r:

1. **Soft Besiege ($|E| \geq 0.5$, $r \geq 0.5$):** The hawks softly surround the prey while allowing it an opportunity to escape.
2. **Hard Besiege ($|E| < 0.5$, $r \geq 0.5$):** The hawks aggressively converge toward the prey from all directions.
3. **Soft Besiege with Levy Flight ($|E| \geq 0.5$, $r < 0.5$):** The hawks perform surprise dives combined with occasional long jumps to capture an energetic rabbit.
4. **Hard Besiege with Levy Flight ($|E| < 0.5$, $r < 0.5$):** The hawks execute rapid progressive dives to capture an exhausted rabbit.

Particularly significant is the Levy flight component, which is a random walk with heavy-tailed step lengths. In a complicated multimodal problem like traffic signal timing, it enables the algorithm to make sporadic lengthy jumps that assist it in escaping local optima [31].

One significant change was made to the standard HHO in our implementation. The directional information required for balanced exploration is eliminated by the absolute value of the position difference in the original hard besiege calculation. In order to enable hawks to move both in the direction of and away from the optimal solution, we substituted a signed difference.

This change, which is fully explained in Chapter 3, stopped all hawks from falling toward the search space's lower bound.

Applications in Traffic Engineering

In order to simultaneously optimize delay, stops, vehicle capacity, and emissions, Zhao and Su (2024) used an enhanced HHO for multi-objective traffic signal scheduling. Our selection of HHO as one of the two primary optimizers in this thesis was directly influenced by their approach's 36.7% reduction in vehicle delay, 31.2% reduction in exhaust emissions, and 41.6% increase in vehicle capacity during peak hours when compared to traditional approaches [10].

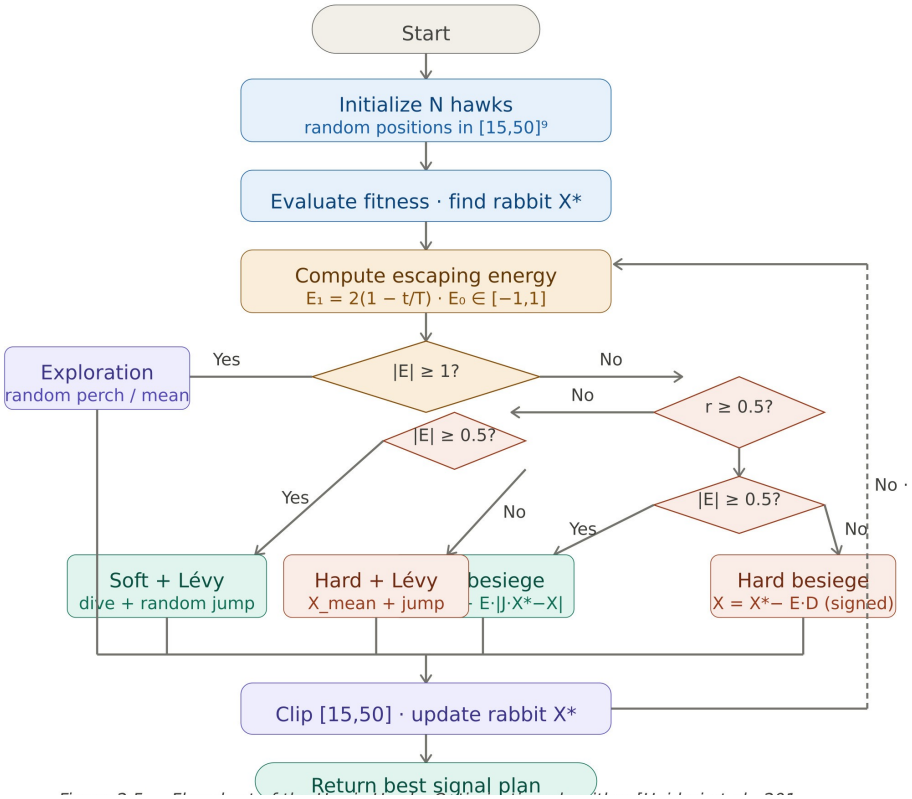


Figure 2.4: *Flowchart of the Harris Hawks optimization algorithm [Heidari et al,2016]*

2.3.4 Genetic Algorithm (GA)

Origin and Biological Inspiration

The Genetic Algorithm is one of the most established and well researched meta-heuristic techniques, it was first presented by Holland in 1975 and further popularized by Goldberg in 1989 [30] .

Inspired by Darwinian natural selection, GA uses three operators selection, crossover, and mutation to maintain a population of potential solutions and evolve them across generations. The concept is straightforward: those who are more suited to their surroundings are chosen by nature. The population improves as a whole over several generations. GA mimics this procedure, except it generates potential solutions to an optimization problem rather than creatures [30].

Mathematical Formulation

GA keeps track of a population of potential solutions known as chromosomes and uses three operators to evolve them every generation:

1. **Selection:** Parents are selected based on their level of fitness. We employ tournament selection, in which two people are chosen at random, and the superior one is chosen to be a parent. Until the entire mating pool is created, this process is repeated.

2. **Crossover:** An offspring is created by combining the chromosomes of two parents. We employ arithmetic crossover, where each child's gene is a weighted average of the two parents' corresponding genes, with the weight chosen at random from [0, 1]. This makes it possible for the search to investigate the area between two viable options.

3. **Mutation:** With a little probability, every gene is randomly reset to a new value within the search parameters. This keeps the population from prematurely converging to a less-than-ideal solution and guarantees genetic variety.

Applications in Traffic Engineering and Role in This Work

GA has been used for traffic signal timing for a very long time. By applying GA and Differential Evolution to actual intersections in Saudi Arabia, Jamal et al. (2020) were able to reduce delays by 15 to 35% as compared to the current state of affairs [7].

GA is used as a reactive benchmark in this thesis. It is unrelated to the LSTM predictor, although it uses the same signal boundaries and fitness function as WOA and HHO. Because of this intentional design decision, we are able to separate the contribution of prediction: any performance discrepancy between GA and the LSTM-guided algorithms can be directly linked to the forecasting capability rather than the optimization process.

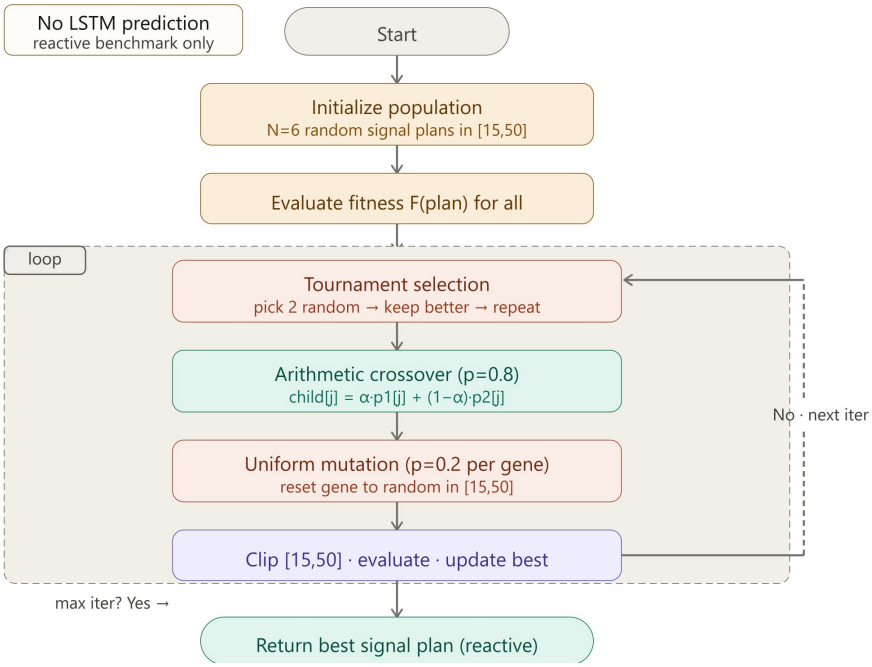


Figure 2.5: *Flowchart of the Genetic Algorithm used as reactive benchmark*

2.4 Deep Learning for Traffic Prediction

Proactive signal control is made possible in large part by accurate short-term traffic forecast. A predictive system can foresee demand increases and proactively modify signal plans to stop lineups from forming, as opposed to responding to congestion after it has already developed. With a focus on the LSTM architecture utilized in our framework, we examine the deep learning techniques most pertinent to this task in this part.

2.4.1 Traffic Prediction as a Time Series Problem

Vehicle counts, queue lengths, waiting times, and speeds are examples of traffic flow variables that fluctuate over time in a continuous stream of numbers. It is a time series forecasting challenge to predict those values for the next sixty seconds based on the past two minutes.

Kalman filters and ARIMA (AutoRegressive Integrated Moving Average) are two traditional methods for solving this issue. When traffic is light and steady, these approaches perform well because they make the assumption that the connection between past and future values is roughly linear. However, actual urban traffic is not linear at all. When demand is close to saturation, a slight increase in demand can start a chain reaction of delays that affect several junctions. That type of behavior is not adequately captured by any linear model.

Because deep learning offers better capabilities than standard machine learning and statistical methods, new developments in deep learning have revived traffic flow prediction research. The remarkable feature extraction capabilities of contemporary deep learning architectures allow them to automatically extract complicated spatiotemporal correlations and dynamic evolution patterns from intricate traffic datasets [5]. For short-term traffic prediction problems, LSTM has emerged as the most popular method among these designs.

2.4.2 Recurrent Neural Networks(RNN)

Each input is processed individually by standard feedforward neural networks, which have no recall of previous inputs. In order to solve this, Recurrent Neural Networks (RNNs) introduce a hidden state vector that is updated at each time step and forwarded to the subsequent one. This enables the network to create an internal representation of the input sequence's recent history, which is crucial for time series forecasting.

Nevertheless, a well-known training issue with standard RNNs is the vanishing gradient problem. Recurrent backpropagation takes a relatively long time to learn how to store information over long time intervals, primarily because to inadequate, fading error backflow [32]. This makes it difficult for typical RNNs to acquire patterns that span more than a few time steps in practice, which is a major drawback for traffic prediction, as patterns like weekly periodicity or morning rush hour span hundreds of time steps.

2.4.3 Long Short-Term Memory (LSTM) Networks

Origin and Biological Inspiration

Long Short-Term Memory networks was first presented by Sepp Hochreiter and Jürgen Schmidhuber in 1997 and marked a significant advancement in resolving the vanishing gradient problem, one of the most significant issues in training neural networks on sequential input. With the introduction of a memory cell and gating mechanisms, LSTM made it possible for knowledge to either be retained or eventually forgotten [32].

Mathematical Formulation

The cell state c_t, which serves as long-term memory, and the hidden state h_t,which is the short-term output carried over to the following stage, are the two state vectors that each LSTM cell keeps. At each time step, these states are controlled by three learnable gates [32]:

- **The forget gate:**determines what should be removed from the cell state:

$$f(t) = \sigma\left(W_f \cdot [h(t-1), x(t)] + b_f\right)$$

- **The input gate:** determines what additional data should be added to the cell state:

$$i(t) = \sigma\left(W_i \cdot [h(t-1), x(t)] + b_i\right)$$

$$\tilde{c}(t) = \tanh\left(W_c \cdot [h(t-1), x(t)] + b_c\right)$$

- **The cell state update:** combines forgetting and writing mechanisms:

$$c(t) = f(t) \odot c(t-1) + i(t) \odot \tilde{c}(t)$$

- **The output gate:** controls the exposed output of the cell state:

$$o(t) = \sigma\left(W_o \cdot [h(t-1), x(t)] + b_o\right)$$

$$h(t) = o(t) \odot \tanh(c(t))$$

where W and b denote learnable weight matrices and bias vectors, tanh is the hyperbolic tangent activation function, σ is the sigmoid function, and $\odot$ denotes element-wise multiplication. Together, these gating mechanisms mitigate the vanishing gradient problem in standard RNNs by allowing the network to selectively preserve or discard information over long sequences [32].

Applications in Traffic Engineering

LSTM networks have significantly advanced the performance boundaries in traffic flow forecasting because to their remarkable capacity to handle high-dimensional and large-scale traffic data [5].

Several studies from our article table directly support the efficacy of LSTM for traffic prediction in the context of this thesis. By dividing the road network before using distinct predictors, Abbas et al. (2018) increased LSTM accuracy by a factor of five [4]. When Mou et al. (2019) added temporal context vectors to LSTM, their T-LSTM successfully restored lost data and outperformed cutting-edge techniques [5]. Zheng et al. (2020) achieved the highest accuracy of all analyzed methods by combining Attention-based Convolutional LSTM with Bidirectional LSTM to simultaneously collect spatial and temporal variables [6]. LSTM's capacity to differentiate between weekday and holiday traffic patterns was validated by Nv Er Ren et al. (2021), making it immediately applicable to practical urban deployment settings [3].

In our approach, a sequence of the latest 24 time steps, or 120 seconds of data at 5-second intervals, is fed into the LSTM model along with 19 features that are taken from each of the nine controlled intersections in the Blida network. The entire network waiting time is predicted by the model sixty seconds in advance.

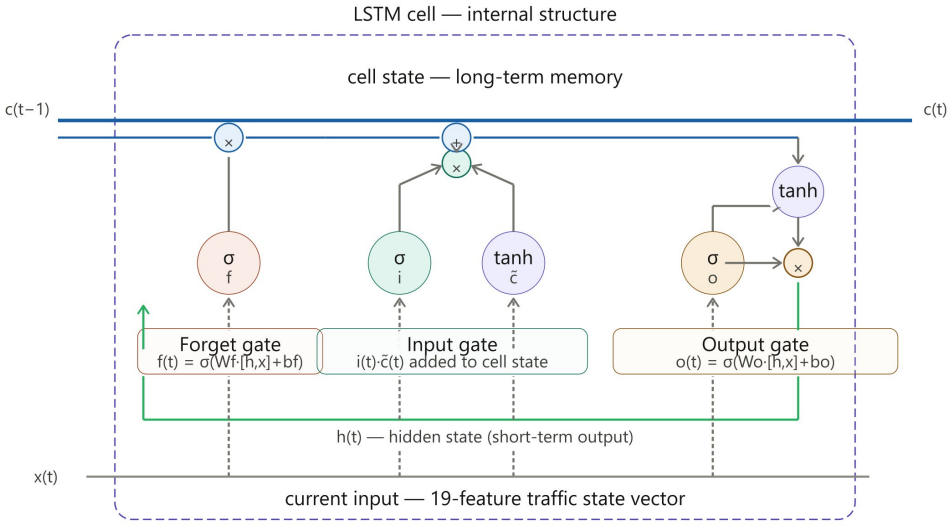


Figure 2.6: *Architecture of a single LSTM cell [Hochreiter Schmidhuber, 1997]*

2.4.4 LSTM for Traffic Prediction : Related Work

Since the mid2010s, LSTM networks have been widely used for traffic prediction jobs. For short-term traffic speed prediction on urban roads, Ma et al. (2015) were among the first to show that LSTM performs better than both ARIMA and shallow neural networks [33]. Zhao et al. (2017) reported mean absolute percentage errors (MAPE) below 10% for prediction horizons up to 15 minutes when using LSTM to forecast trip time on expressways [34].

When comparing LSTM and GRU (Gated Recurrent Unit) for traffic flow prediction, Fu et al. (2016) discovered that LSTM offers slightly higher accuracy at the expense of a somewhat longer training period [35].

An LSTM-based adaptive signal controller for isolated crossroads was proposed by Liang et al. (2019), who reported an 18% decrease in vehicle stops [36]. This line of work is extended to a multi-intersection network by our approach, which substitutes population-based meta-heuristic optimizers, which do not require a mathematical model of intersection dynamics, for the model predictive controller.

In our solution, a series of 24 time steps, or 120 seconds of history at 5-second intervals, are fed into the LSTM model. Each time step has 19 features that are derived from the network's nine intersections. Chapter 3 contains comprehensive information about the architecture, training process, and performance indicators.

2.5 Positioning of the proposed framework

Before presenting the methodology, it is important to step back and look at where this thesis fits within everything that has been reviewed so far. Two questions need to be answered clearly: what is missing in the existing research, and what exactly does this work contribute.

2.5.1 Identified Research Gaps

Three gaps frequently emerge from an examination of the literature on deep learning prediction, traffic signal management, meta-heuristic optimization, and simulation platforms.

- **Gap 1: Optimization loops lack prediction:**Signal timings are optimized depending on the present observed traffic status in the vast majority of the meta-heuristic works examined, including GA, PSO, WOA, and HHO. By design, they are reactive. Even the most modern and effective methods, such as the WOA application of Thahe et al. (2019) [9] or the enhanced HHO of Zhao and Su (2024) [10] work on traffic that has already reached the intersection. There is no mechanism in these works to predict the future. Although their own implementation operated at a coarse zone level and was not implemented in a real-time rolling horizon loop, Khasawneh and Awasthi (2023) [13] were among the few to explicitly identify this as a critical open problem, stating that better predictions are needed to solve the problem before it occurs.

- **Gap 2: In a closed-loop predictive framework, WOA and HHO haven't been combined with LSTM:** Although HHO has been applied to traffic signal timing (Zhao and Su, 2024) [10] and WOA has been used to optimize LSTM hyperparameters (Xu et al., 2022) [23], no work has combined both algorithms with an LSTM predictor in a single integrated system where the prediction feeds directly into the optimization at every decision step. This combination's ability to anticipate traffic using LSTM and determine the optimal signal plan for that forecast using WOA or HHO has never been investigated.

- **Gap 3: The urban environment in Algeria is still not well understood:** Very little study has been especially focused on this setting, despite the obvious and pressing need for improved traffic management in Algerian cities. Cities like Blida have very different road networks, traffic patterns, and infrastructure infrastructure than the European or Asian cities that are the focus of the majority of current research. This thesis aims to provide a framework that is explicitly constructed and assessed on an actual Algerian road network utilizing tools and techniques tailored to local constraints.

2.5.2 Our findings

All three shortcomings are directly addressed in this thesis. In reaction to the initial gap, a predictive control loop with a rolling horizon is suggested and put into practice. The system employs an LSTM neural network to predict vehicle counts 60 seconds ahead of time at all controlled crossings, and the optimization algorithms look for the optimal signal plan for that anticipated future state rather than optimizing for the current traffic situation. This is the first implementation of such a loop in an SUMO simulation environment using WOA and HHO.

Regarding the second gap, in order to directly and fairly compare their performance, WOA and HHO are both tailored to the traffic signal timing problem using a similar architecture, which includes the same rolling horizon loop, fitness function, and solution encoding. Because the

fitness function is based on queuing theory and takes LSTM predictions as input, forecasting and optimization are tightly integrated in a way that has never been done before.

Regarding the third gap, this piece was created and assessed especially for the metropolitan environment of Algeria. The framework is based on a real Blida road network that was imported into SUMO from OpenStreetMap, with traffic demand scenarios adjusted to account for local realities. Everything here is constructed with Blida's road layout and traffic reality in mind, as opposed to modifying a framework that was initially intended for a different metropolitan setting. In order to provide a tangible and quantifiable improvement that is directly applicable to Algerian cities and the difficulties they encounter on a daily basis, the outcomes are compared against a reactive GA .

2.6 Chapter Conclusion

The theoretical and technical underpinnings of the suggested framework across four domains have been reviewed in this chapter. The evolution of traffic signal control from Webster's static formula to vehicle-actuated systems and network-level adaptive controllers like SCOOT and SCATS has been studied, and the main constraint driving this work has been found to be the lack of predictive capability.

The mathematical underpinnings of the Whale Optimization Algorithm (WOA) , Harris Hawks Optimization (HHO), and Genetic Algorithm (GA) in meta-heuristic optimization have been explained. The Long Short-Term Memory (LSTM) architecture and its use for traffic prediction in deep learning have been examined, and the suggested implementation has been placed within the larger body of literature.

Chapter 3

Proposed Framework and Methodology

3.1 Introduction

This chapter presents the complete methodology of the proposed hybrid predictive meta-heuristic framework for dynamic traffic signal optimization. it begins with a high-level overview of the framework architecture and the rolling horizon control strategy. it then describes the study area: city center of Blida and the SUMO simulation environment used to model it. The LSTM predictive module, fitness function, and the three optimization algorithms are then detailed in turn. The chapter concludes with a step-by-step description of the full control loop and its implementation.

3.2 Framework Overview

3.2.1 General Architecture

The suggested framework has three main parts: a traffic simulation environment, a short-term predictive module that uses LSTM, and a signal plan optimizer that uses population meta-heuristic algorithms. The overall structure of the system is shown in Figure 3.1.

The SUMO simulator acts as both the source of real-time traffic data and the environment on which optimized signal plans are applied. At each control horizon, the TraCI interface extracts current traffic state variables as queue lengths, waiting times, vehicle counts from the nine monitored intersections. These measurements allow the fitness function to determine potential signal plans and feed the LSTM buffer, which is used to predict future congestion.

After receiving the fitness function, the metaheuristic optimizer (WOA, HHO, or GA, depending on the experiment) looks for the signal plan that reduced the total weighted cost over all intersections.The process is repeated at the next horizon after the optimal plan has been identified and implemented to the simulation using TraCI. This closed-loop structure allows the framework to continuously adapt to changing traffic conditions throughout the simulation.

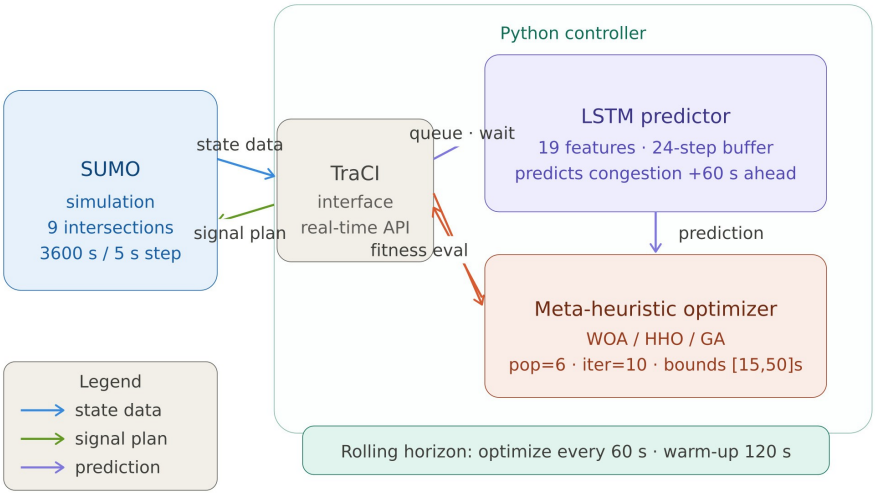


Figure 3.1: *General architecture of the proposed hybrid predictive meta-heuristic framework*

3.2.2 Rolling Horizon Concept

The framework uses a rolling horizon principle, which is a common way to control model predictive control (MPC) systems. Instead of making one static plan for the whole simulation, the optimizer is called every 60 seconds in our implementation to change the signal plan based on the most recent traffic data and the LSTM prediction.

To support this dynamic adjustment, The LSTM uses traffic data from the last 24 time steps (the last 120 seconds) to predict how congested it will be at time t+60. The meta-heuristic optimizer then finds the signal plan that makes the fitness function rated at the current state as minimal as possible. The plan is applied immediately and remains in effect until the next horizon at time t+60, At that point, the process starts over with a new buffer and a new prediction. This sliding window method keeps the system responsive to short-term changes while also avoiding the instability that would come from changing plans too often.

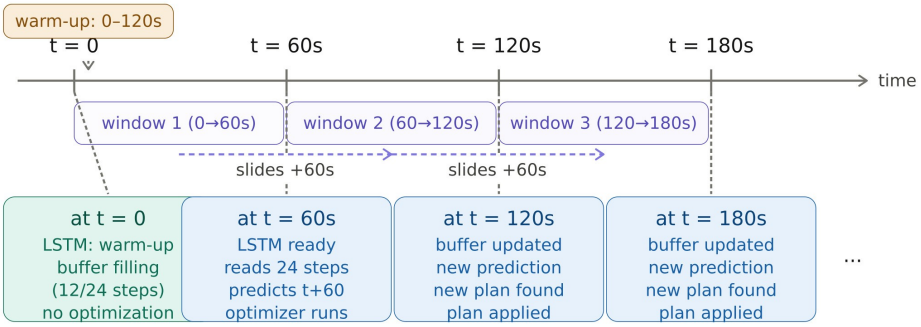


Figure 3.2: *Rolling horizon control strategy:the optimization window slides forward every 60 seconds*

3.2.3 LSTM + Meta-Heuristic Integration

The link between the LSTM predictor and the meta-heuristic optimizer is indirect, but it plays a key role, both are essential components of the same process. The LSTM does not calculate signal plans directly; however, it generates a forward-looking estimate of how congested a network will be at a point in time and informs the optimizer that action needs to be taken before conditions get worse.

When the LSTM indicates that future wait time in excess of what is acceptable is imminent (i.e., that congestion is approaching during peak hours), the optimizer uses this information to generate a new plan to minimize congestion and is evaluated against a state where queues are already beginning to form. As a result, the optimizer can generate a preemptive plan for the prevention of queue growth, rather than create a reactive plan in response to an existing queue-developed condition.

On the other hand, the Genetic Algorithm applied as the reactive algorithm baseline in this study does not use the LSTM prediction. While it executes the same optimizer and fitness function, its execution is purely driven by the present situation without considering future implications. In effect, the performance difference between GA and the two LSTM-based algorithms becomes a measure of the importance of prediction in traffic signals management.

3.3 SUMO Simulation Platform

3.3.1 Overview of SUMO

”Simulation of Urban MObility” (SUMO) is an open source, highly portable, microscopic and continuous traffic simulation package designed to handle large networks. It allows for intermodal simulation including pedestrians and comes with a large set of tools for scenario creation. It is mainly developed by employees of the Institute of Transportation Systems at the German Aerospace Center[37].

SUMO is capable of importing the actual road network from OpenStreetMap using the netconvert utility, which translates OSM's XML input to SUMO's internal .net.xml format. Traffic signals can be described as finite state machines, where their cycles and timing are specified by the network file and can be adjusted at runtime. SUMO has proven very popular in traffic signal control studies, due to its free nature, Python scripting capabilities, and flexibility in network modeling.



Figure 3.3: *SUMO-GUI visualization of the Blida network during a medium-traffic simulation*

3.3.2 TraCI API

TraCI is the short term for ”Traffic Control Interface”. is a socket-based API for allowing real-time interaction between the external controller program and the SUMO simulation. During each simulation iteration, the TraCI client receives the current state of objects like vehicle speed, queuing length, waiting time, and the phase index of signal control points, and sends commands such as changing signal timing plans, rerouting vehicles, and adding traffic demand to adjust the simulation process[38].

Within the proposed framework, TraCI fulfills the following three tasks. Firstly, it collects the traffic state measurements to form the input sequence for the LSTM model and evaluate the value of the fitness function. Secondly, it implements the signal plan generated by the optimization method by modifying the duration of phases of all the nine traffic lights together. Finally, it receives the measures of performance, namely the total waiting time, number of stops, trip time,

and throughput of vehicles that are used to evaluate the effectiveness of each algorithm at the end of each simulation run[38].

All TraCI calls in this research are made through the Python traci API that gives access to all the available TraCI services. One instance of SUMO is started for each run of an experiment and controlled over its entire duration of 3600 seconds through the TraCI link named 'main_sim'.

3.3.3 Why SUMO for This Work

The decision to use SUMO for simulations was made based on the following reasons. Firstly, SUMO is licensed under the Eclipse Public License which allows us for full reproduction of all experiment results in this thesis without any need for proprietary software. Second, the netconvert tool made it possible to directly import the OpenStreetMap data for our area, which made a real road network geometry instead of a fake test case. Third, the TraCI interface gives the millisecond-level response times needed for the rolling horizon control loop without the extra work that comes with file-based communication. Fourth, the traci and sumolib libraries that come with SUMO work perfectly with the PyTorch and NumPy libraries that were used to make the LSTM model and the meta-heuristic algorithms.

3.4 Study Area : Blida Network

The city center has a lot of roads that cross each other, a lot of people walking around, and a lot of cars during the busiest times of the day, in the morning and evening.

The chosen bounding box is made up of three of the city's old gates that mark the edges of the old city center: Bab Dzair (northeast corner) Bab Essebt (western boundary) Bab Errahba (southern boundary)

The area was selected for three reasons. First, it has a lot of signalized intersections in a small area. The bounding box is about 830 meters by 830 meters and has 95 detected junctions, 9 of which were chosen as the main controlled intersections for this study.

Second, the road network has the typical mix of wide main boulevards and congested local streets that you find in Algerian city centers. This makes it a good and difficult test case for optimizing multi-intersection signals.

Third, OpenStreetMap has a complete map of the area, as shown in Figure 3.4. This means that the netconvert tool can extract a detailed and geometrically accurate road network without the need for manual digitization.



Figure 3.4: *Location of the study area: City center, Blida, Algeria (coordinates: 36.467°N – 36.475°N, 2.825°E – 2.836°E*

3.4.1 OSM Map Extraction and Network Generation

The JOSM editor used to extract the road network from OpenStreetMap. The bounding box boundaries are latitude 36.4670 to 36.4745 and longitude 2.8250 to 2.8360. The resulting OSM file was converted to the SUMO format via the netconvert command-line tool with the following key parameters:

```
netconvert --osm-files blida.osm
           --output-file blida_all_tls.net.xml
           --tls.guess.threshold 0
           --geometry.remove
           --roundabouts.guess
```

The command –tls.guess.threshold 0 makes netconvert treat all junctions as signalized intersections regardless of their node degree. The result is a first iteration network containing 95 traffic lights. The network was analyzed manually and trimmed down to include only the 9 busiest intersections, which constitute the main road network. producing the final simulation network blida_standard.net.xml

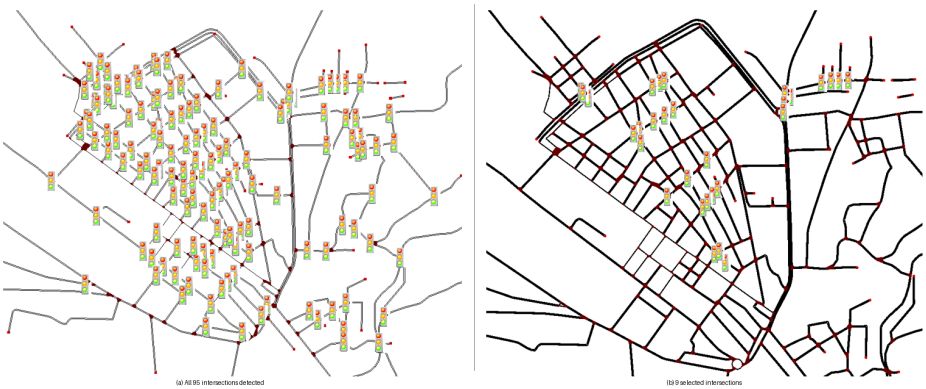


Figure 3.5: *Network reduction: from 95 detected intersections (left) to 9 selected intersections (right)*

3.4.2 Network Properties and Traffic Demand Generation

The final network has 9 signalized intersections which are connected by various directional road segments. The number of controlled lanes per intersection varies between 9 and 13, as shown in Table below.

Properties of the 9 Selected Intersections

ID	Intersection Name (short)	Controlled Lanes	Phases
I1	joinedS_1172439356_...	13	2
I2	joinedS_1172438657_...	13	2
I3	joinedS_1788379490_...	12	2
I4	joinedS_1172439370_...	12	2
I5	joinedS_1172439017_...	11	2
I6	joinedS_1172438944_...	11	2
I7	joinedS_1703161233_...	10	2
I8	joinedS_1449025102_...	10	2
I9	joinedS_11328467968 ...	9	2

A two-phase signal plan controls each intersection. The two main conflicting movements switch between green and red, separated by yellow clearance intervals of 4 seconds SUMO's random trip generator used (randomTrips.py) to create traffic demand. the vehicle departure rates sets to match three levels of traffic intensity levels representative of typical daily conditions in Algerian urban centers. Each scenario was generated as a route file (.rou.xml) defining the departure time and origin-destination pair of each individual vehicle over a 3600-second simulation period.

3.4.3 Traffic Scenarios : Low, Medium, Peak

Three traffic scenarios were created to test the framework at different levels of demand:

Table 3.1: *Traffic Scenario Parameters*

Scenario	Demand (veh/hr)	Typical Period	Avg Queue (baseline)	Avg Wait (baseline)
Low	200	Off-peak / night	Low (< 5 veh/intersection)	446.9 s
Medium	500	Midday	Moderate (5–15 veh)	1161.4 s
Peak	900	Morning/evening peak	High (> 15 veh)	1719.0 s

The low scenario (200 vehicles/hour) shows what happens when the network is not busy and is running far below capacity.

The medium scenario (500 vehicles/hour) is like what happens at midday when there is some traffic at major intersections.

In the peak scenario (900 vehicles/hour), it has a lot of cars on the road in the morning or evening rush hour, and several intersections are at or above their practical capacity. This causes traffic congestion and long delays.

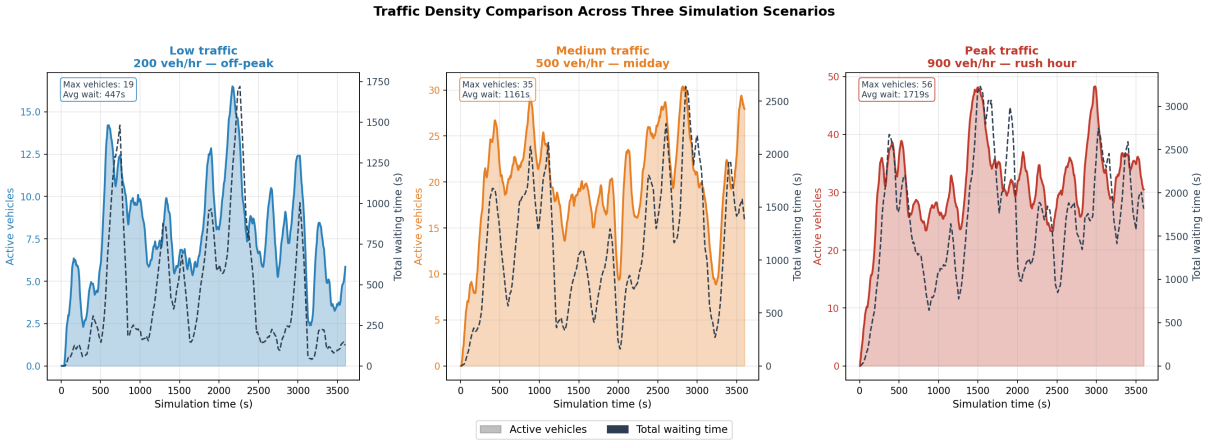


Figure 3.6: *Traffic density comparison across the three simulation scenarios*

3.5 LSTM Predictive Module

3.5.1 Input Features

The LSTM algorithm takes as its input a multivariate time series built based on measurements recorded from the nine monitored intersections.

For each step in the simulation lasting 5 seconds, the following parameters are calculated using TraCI for each intersection: the total queue length (number of halting vehicles across all controlled lanes) and the total vehicle count (number of vehicles present, including both stopped and moving vehicles) , This produces two parameters for each intersection multiplied by 9 intersections equal to 18 intersection-based parameters plus one parameter representing the total waiting time across the entire network for a total of 19 input parameters.

The inputs to the LSTM for every prediction are sequences of 24 steps, which represent 24*5 = 120 seconds of traffic history. This sequence length was chosen as a balance between enough information provided by the inputs about traffic waves and low computation costs of the LSTM inference at the same time. Every feature in the input is normalized using a StandardScaler fitted on the training dataset and saved along with the model weights for use during online prediction.

3.5.2 LSTM Architecture

The LSTM model consists of a two-layer LSTM encoder followed by a three-layer fully connected prediction head. The LSTM layers have a hidden size of 256 units each, with a dropout rate of 0.3 applied between the two layers to reduce overfitting.

The prediction head then processes the output through a series of layers: first a linear transformation from 256 to 128 units, followed by a ReLU activation and dropout. This is followed by another linear layer reducing the size to 64 units, another ReLU activation, and finally a last linear layer that produces a single value.

This final output corresponds to the predicted total network waiting time (in seconds), estimated 60 seconds (12 steps) ahead of the current time.

The total number of trainable parameters in the model is approximately 1.2 million. The model was implemented in PyTorch and trained on Google Colab using an NVIDIA T4 GPU.

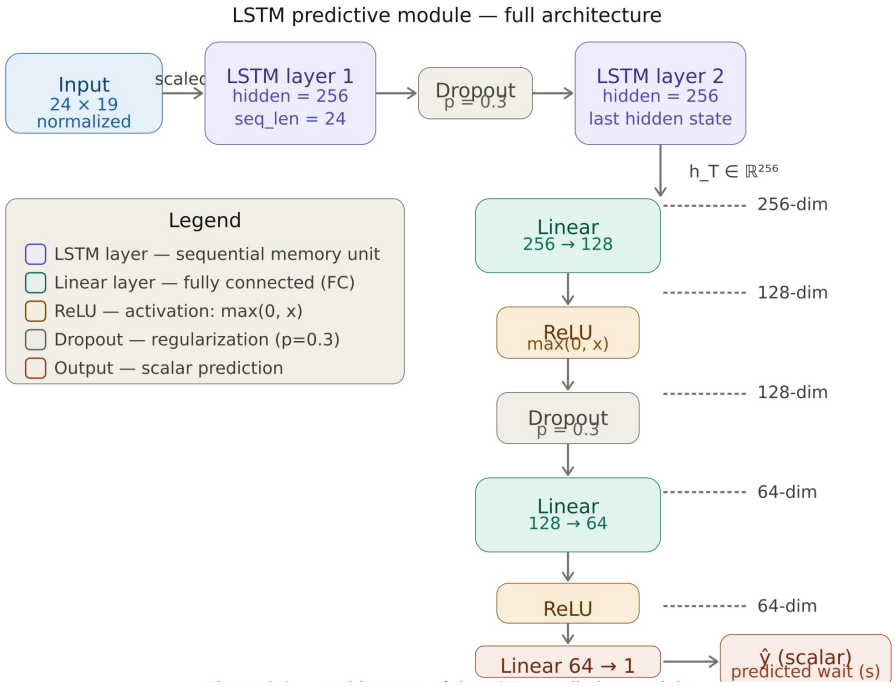


Figure 3.7: *Architecture of the LSTM predictive module*

3.5.3 Training on Google Colab

The training dataset was generated by running the SUMO simulation under all three traffic scenarios using different random signal timings. During these runs, the full set of 19 features was recorded every 5 seconds, producing time series data over multiple simulations. In total, the dataset includes around 15,000 input–output pairs, which were split into training, validation, and test sets using an 80/10/10 ratio.

The model was trained using the Adam optimizer where the learning rate is set to 0.001 and the batch size is fixed to 64, The loss function is the Mean Squared Error (MSE) between the predicted and actual total waiting time.

Training was run for 100 epochs with early stopping based on validation loss, with a patience of 15 epochs.

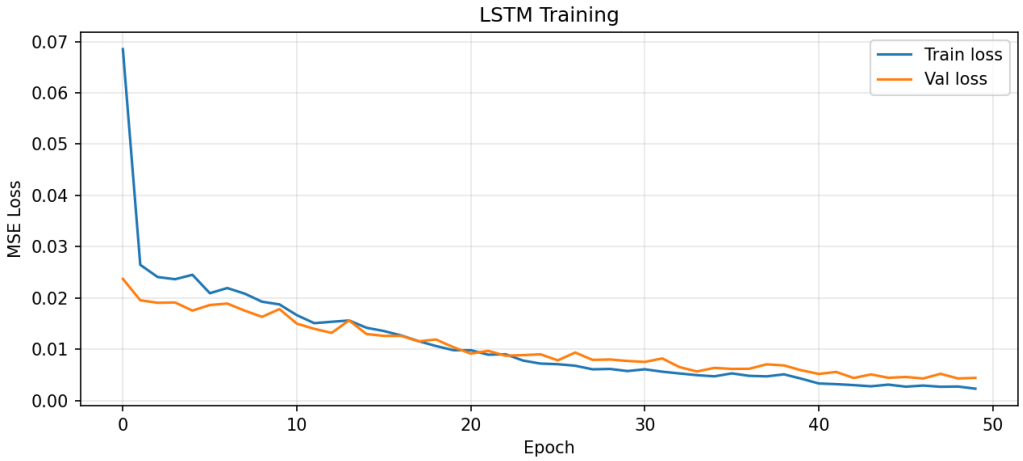


Figure 3.8: *Evolution of MSE loss during the training and validation phases of the LSTM model*

The best model weights as measured by minimum validation loss were saved and used for all subsequent experiments.

Data preprocessing involved applying the StandardScaler method to normalize each input feature and output. The input features (e.g., number of vehicles) and output target (e.g., waiting time) were scaled to different orders of magnitude (e.g., 1 - 9 number of vehicles vs thousands of seconds for the delay).

The scalers were fitted exclusively on the training set and applied to validation and test sets without refitting, in accordance with standard machine learning practice.

3.5.4 Prediction Results

Three standard regression metrics used to test the trained model on the held-out test set: Mean Absolute Error (MAE), the coefficient of determination (R^2), and Mean Absolute Percentage Error (MAPE).

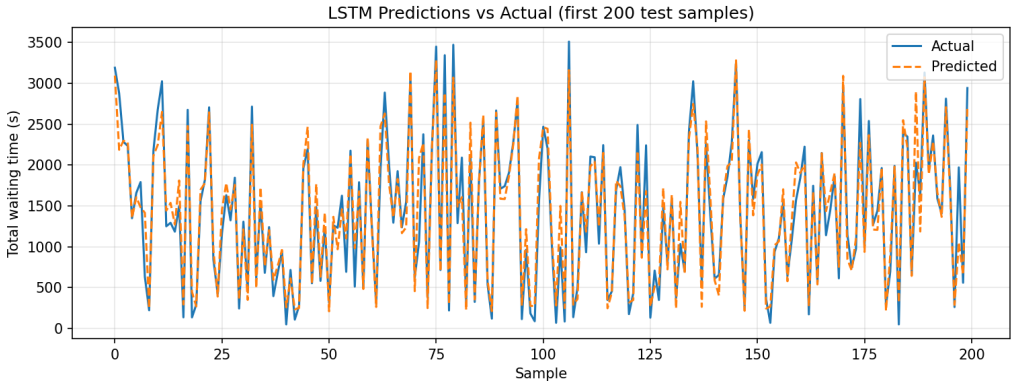


Figure 3.9: *LSTM model predictions versus actual ground truth for 200 test samples*

The results are summarized in Table below : An R^2 of 0.923 indicates that The LSTM

Table 3.3 — LSTM Prediction Performance on Test Set

Metric	Value	Interpretation
MAE	173.09 s	Average prediction error of 173 seconds on waiting time
R^2	0.9232	The model explains 92.3% of the variance in waiting time
MAPE	24.11%	Average percentage error (computed on samples $>$ 50s)

model explains more than 92% of the variance in the target variable, which is a good result for a noisy, nonlinear time series.

When you look at the target range, the MAE of 173 seconds makes more sense. In the peak scenario, waiting times can be anywhere from almost zero to more than 4000 seconds, so an average error of 173 seconds is less than 5% for very congested conditions.

The MAPE of 24% is only based on samples where the real wait time is more than 50 seconds. This is because calculating percentage errors close to zero gives values that are not stable.

3.6 Fitness Function

3.6.1 Background: Webster's Formula

As discussed in Chapter 2, Webster's 1958 formula establishes a mathematical relationship between the green time allocated to a phase, the cycle length, and the expected vehicle delay at an isolated intersection. The main idea behind Webster's model is that delay goes down as green time goes up, but only up to a certain point. After that, extending green time makes it harder for competing movements to get through and increases their delay. this trade-off motivates the concept of an optimal green time that minimizes total network delay.

According to Webster's formulation, the delay per vehicle on approach i is approximately proportional to the expression involving key signal timing and traffic variables, where g_i represents the green time, C the cycle length, q_i the arrival flow, and s_i the saturation flow.

This expression increases sharply as q_i decreases (i.e., when too little green time is allocated), and also as q_i approaches s_i, corresponding to near saturated traffic conditions.

Our fitness function is inspired by this relationship, but adapted for real-time use where exact flow rates are not directly available

3.6.2 Red-Time Cost Formula

After trying out a lot of different fitness formulations, like queue-weighted deviation from an optimal green time and Webster-inspired delay estimates, a red-time cost function was chosen, it directly links the green time decision variable to the current level of congestion. The fitness function is described as:

$$F(\text{plan}) = \sum_{i=1}^{9} \left(\text{queue}_i \cdot \text{red}_i + \text{wait}_i \cdot \frac{\text{red}_i}{\text{cycle}_i} \right)$$

For each intersection I:

- queue_i is the number of vehicles that are currently stopped (measured by TraCI).
- wait_i is the total amount of time that all vehicles have been waiting at the intersection (in seconds).

The red time is defined as:

$$\text{red}_i = \text{cycle}_i - \text{green}_i - \text{yellow}$$

and the cycle length as:

$$\text{cycle}_i = 2 \bullet \text{green}_i + 2 \bullet \text{yellow}$$

assuming a symmetric two-phase plan with a fixed yellow time of 4 seconds.

The first term, $\text{queue}_i \cdot \text{red}_i$, penalizes plans that give short green times (and long red times) to intersections with large queues. A busy intersection should get more green to clear its queue.

The second term, $\frac{\text{wait}_i \cdot \text{red}_i}{\text{cycle}_i}$, also penalizes plans where plenty of the cycle is spent on red at an intersection where vehicles have been waiting for a long time. The two terms work together to create a demand-proportional allocation principle: intersections with more traffic and longer wait times get more green time.

This fitness function has several practical advantages. It only relies on current measurements that TraCI can get to directly, without any extra calculations. Because it is deterministic based on the current state, the same plan always gets the same fitness score.
This makes the optimizer's search easier. And it changes all the time with the green time decision variables, making a smooth fitness landscape that the meta-heuristic algorithms can easily move through.

3.6.3 Signal Bounds

The green time for each intersection can only be between 15 and 50 seconds. The shortest amount of time of 15 seconds is set so that at least one full queue discharge can happen during each green phase. This takes into account the 2-second start-up time and the average saturation headway of 2 seconds per vehicle. The limit of 50 seconds means that the maximum red time that competing movements can have been: $2 \cdot 50 + 2 \cdot 4$ = 108 seconds.

These bounds are enforced by clipping all candidate solutions to [15, 50] before fitness evaluation, both during the optimizer's search and when applying the final plan to the simulation. The baseline fixed-time plan uses a uniform green time of 30 seconds at all intersections, which falls at the midpoint of the optimization range and serves as a natural reference point

Signal Timing Parameters

Parameter	Value	Justification
Minimum green	15 s	Minimum discharge time for typical queue
Maximum green	50 s	Limits max red for competing movements
Yellow time	4 s	Standard clearance interval
Baseline green	30 s	Midpoint, standard fixed-time reference
Min cycle	38 s	$2 \cdot 15 + 2 \cdot 4$
Max cycle	108 s	$2 \cdot 50 + 2 \cdot 4$

3.7 WOA Implementation

The WOA algorithm is used at each horizon of control to determine the 9-dimensional signal plan vector that minimizes fitness F(plan). In this regard, each whale within the population corresponds to a signal plan vector that contains 9 green times for each intersection.

3.7.1 Population Initialization

At the beginning of each optimization step, a small population of 6 whales created randomly choosing their values between 15 and 50. This randomness helps cover different parts of

the search space instead of starting from similar solutions.

After initializing them, immediately evaluate their fitness to see which one performs best. This best solution is then considered as the current "prey" that the other whales will try to follow and improve upon.

3.7.2 Encircling Prey

In this phase, each whale moves toward the best solution found so far. The movement becomes more focused over time, as the search space slowly reduces. This is controlled by the parameter A, which decreases linearly from 2 to 0 over the 10 iterations.

When $|A| < 1$, the position of a whale is updated as:

$$X(t+1) = X_{\text{best}} - A \cdot |C \cdot X_{\text{best}} - X(t)|$$

where $C = 2 \cdot r_2$, with r_2 randomly drawn from the interval $[0, 1]$, and X_{best} represents the current best solution. The absolute value ensures that the movement is directed toward the best solution, while A and C introduce randomness to control the step size.

3.7.3 Bubble-Net Spiral Attack

With a probability of 0.5, each whale updates its position using a spiral movement around the current best solution instead of the encircling strategy. The position update is defined as:

$$X(t+1) = |X_{\text{best}} - X(t)| \cdot e^{b \cdot l} \cdot \cos(2\pi l) + X_{\text{best}}$$

where $b = 1$ controls the shape of the spiral, and l is a random value drawn from $[-1, 1]$ at each iteration. This mechanism simulates the spiral motion of whales around their prey, allowing the search to explore different positions near the best solution while maintaining diversity in the population.

3.7.4 Exploration Phase

When $|A| > 1$, which usually happens in the early iterations, the whale updates its position by moving toward a randomly selected whale instead of the current best solution. The update rule is:

$$X(t+1) = X_{\text{rand}} - A \cdot |C \cdot X_{\text{rand}} - X(t)|$$

where X_{rand} is a randomly chosen solution from the population. This step encourages exploration by allowing whales to move in different directions rather than focusing only on the best solution.

After each update, the position is clipped to the interval $[15, 50]$ to ensure valid signal timings. After completing all 10 iterations, the best solution found is returned as the optimized signal plan.

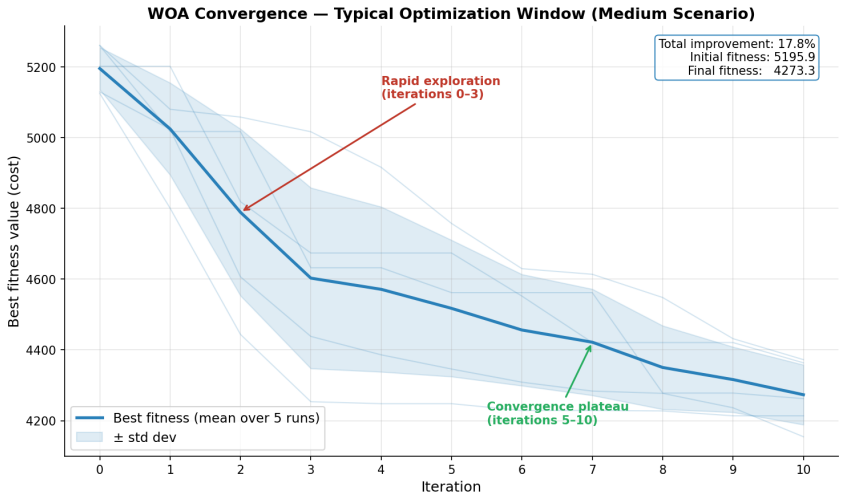


Figure 3.10: *Convergence rate of the Whale Optimization Algorithm (WOA) over 10 iterations under the medium traffic demand scenario, showing mean best fitness and standard deviation.*

3.8 HHO Implementation

The HHO implementation works the same way as WOA: it takes in the fitness function and signal bounds and returns the best 9-dimensional signal plan found after 10 iterations of 6 hawks. The main differences in the algorithms are the energy-based transition between phases and the four exploitation strategies.

3.8.1 Escaping Energy Concept

The escaping energy E models how likely the prey (rabbit) is to escape, and it decreases over time. It is computed as:

$$E_1 = 2 \cdot \left(1 - \frac{t}{\text{max_iter}}\right), \quad E = E_1 \cdot E_0$$

where E_0 is a random value drawn from $[-1, 1]$.

When $|E| \geq 1$, the hawks are in exploration mode and move randomly across the search space.

When $|E| < 1$, they switch to exploitation mode and start attacking the prey using different strategies based on a probability r. This gradual and random decrease of E allows the algorithm to smoothly transition from exploration to exploitation as the iterations progress.

3.8.2 Four Exploitation Strategies

When $|E| < 1$, the hawks switch to exploitation mode and try to capture the rabbit using one of four strategies, depending on the values of rand and $|E|$.

- **Strategy 1: Soft besiege** ($r \geq 0.5$, $|E| \geq 0.5$):
  The hawk moves toward the rabbit with a moderate step size:

$$X(t+1) = X_{\text{rabbit}} - E \cdot |J \cdot X_{\text{rabbit}} - X(t)|$$

- **Strategy 2: Hard besiege** ($r \geq 0.5$, $|E| < 0.5$):
  The hawk moves directly toward the rabbit with a more aggressive step:

$$X(t+1) = X_{\text{rabbit}} - E \cdot D, \quad \text{where } D = X_{\text{rabbit}} - X(t)$$

- **Strategy 3: Soft besiege with Lévy flight** ($r < 0.5$, $|E| \geq 0.5$):
  The hawk first attempts a soft move. If it does not improve the solution, it performs a random jump using a Lévy flight:

$$X_1 = X_{\text{rabbit}} - E \cdot |J \cdot X_{\text{rabbit}} - X(t)|$$
$$X_2 = X_1 + \text{rand}(\text{dim}) \cdot \text{LF}(\text{dim})$$

- **Strategy 4: Hard besiege with Lévy flight** ($r < 0.5$, $|E| < 0.5$):
  Similar to the previous case, but using the population mean as a reference:

$$X_1 = X_{\text{rabbit}} - E \cdot (X_{\text{rabbit}} - X_{\text{mean}})$$
$$X_2 = X_1 + \text{rand}(\text{dim}) \cdot \text{LF}(\text{dim})$$

3.8.3 Levy Flight

The Lévy flight mechanism introduces random step sizes that occasionally become large, allowing the algorithm to jump out of local optima. These steps follow a heavy-tailed distribution, meaning that most moves are small, but some can be significantly larger.

The Lévy steps are generated using the Mantegna algorithm with parameter $\beta = 1.5$:

$$\text{Lévy} = \frac{u}{|v|^{1/\beta}}$$

where $u \sim \mathcal{N}(0, \sigma^2)$ and $v \sim \mathcal{N}(0, 1)$, and σ is defined as:

$$\sigma = \left(\frac{\Gamma(1+\beta) \cdot \sin\left(\frac{\pi \cdot \beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \cdot \beta \cdot 2^{\frac{\beta-1}{2}}} \right)^{\frac{1}{\beta}}$$

This formulation produces step sizes with higher variability than a standard Gaussian distribution, enabling occasional long jumps that improve the algorithm's ability to escape difficult regions of the search space.

3.8.4 Fix Applied: Signed Difference

In the original HHO formulation, Strategies 2 and 4 use the absolute difference between the rabbit (best solution) and the hawk position. In our implementation, this led to an issue where all hawks quickly converged to the lower bound of the search space (green time = 15 seconds). This happens because the absolute value removes the direction of movement, making all updates push the solutions in the same direction.

To address this, the absolute difference replaced with a signed difference defined as:

$$D = X_{\text{rabbit}} - X(t)$$

This preserves the direction of movement: a hawk above the rabbit moves downward, while a hawk below moves upward. As a result, the population can explore both sides of the solution and converge more accurately toward the optimum.

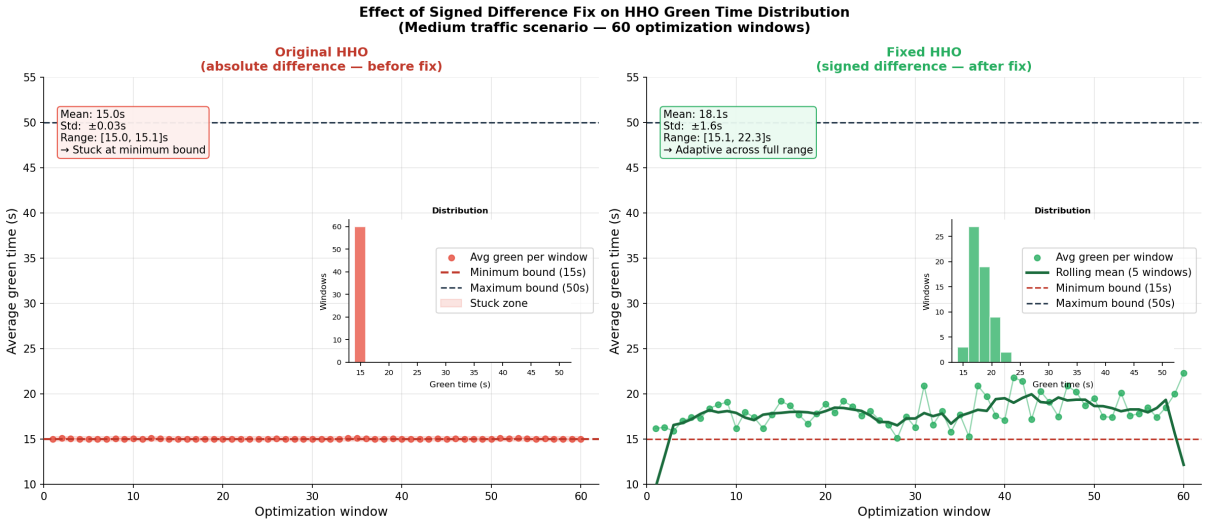


Figure 3.11: *Effect of the signed difference correction on HHO population diversity*

3.9 GA Implementation

The Genetic Algorithm is a reactive benchmark that uses the same signal bounds and fitness function as WOA and HHO, but it doesn't use LSTM prediction. This makes it possible to compare the optimization algorithms directly under the same conditions, which helps to isolate the effect of the predictive part.

3.9.1 Tournament Selection

At each generation, parents are selected using binary tournament selection. For each selection, two individuals are chosen randomly from the population, and the one with the lower fitness value (i.e., the better solution) is kept as a parent. This process is repeated until the mating pool is filled.

This method favors better solutions while still keeping some randomness, allowing weaker individuals a chance to be selected and helping maintain diversity in the population.

3.9.2 Crossover and Mutation

Crossover is applied with a probability of 0.8. For each pair of parents p_1 and p_2, a child c is generated by combining their values using a weighted average:

$$c[j] = \alpha[j] \cdot p_1[j] + (1 - \alpha[j]) \cdot p_2[j]$$

where $\alpha[j]$ is randomly drawn from the interval $[0, 1]$ for each dimension j. This produces offspring that lie between the two parents in the search space.

Mutation is then applied independently to each gene with a probability of 0.2. When a mutation occurs, the gene is replaced by a new value randomly sampled from the interval $[15, 50]$.

This helps maintain diversity in the population and reduces the risk of premature convergence, which is especially important given the small population size (of 6 individuals).

3.9.3 GA as Reactive Benchmark

The main difference between the GA and the LSTM-guided approaches (WOA and HHO) is that the GA doesn't use any predictions. The GA only works on the current traffic state, while WOA and HHO are part of the rolling horizon framework with the LSTM model.

All methods use the same fitness function and test it under the same conditions. The LSTM predictions, on the other hand, help WOA and HHO because they look forward at traffic. The GA, on the other hand, only reacts to what is happening at that moment.

This way, the experiment is controlled, which means that any difference in results can be caused by the existence or non-existence of prediction.

3.10 Rolling Horizon Control Loop

3.10.1 Step-by-Step Algorithm

At each simulation step, the control loop follows this sequence:

- **Step 1 : Advance simulation:**
  The SUMO simulation advances by 5 seconds using TraCI.

- **Step 2 : Collect metrics:**
  Key metrics (queue lengths, waiting times, vehicle counts, stops, and trip times) are collected from all nine intersections and stored for the current step.

- **Step 3 : Update LSTM buffer:**
  The 19-feature state vector is added to a rolling buffer. If the buffer exceeds 24 entries, the oldest one is removed (FIFO).

- **Step 4 : Check horizon:**
  Every 12 steps (i.e., every 60 seconds), the system triggers optimization. Otherwise, it continues the simulation.

- **Step 5 : Check LSTM readiness:**
  If the buffer has fewer than 24 entries, the LSTM is not ready yet. The system continues without optimization.

- **Step 6 : LSTM prediction:**
  The buffer is normalized and passed to the LSTM model to predict the total network waiting time 60 seconds ahead.

- **Step 7 : Save state:**
  The current simulation state is saved for reference (mainly for debugging).

- **Step 8 : Optimize:**
  The optimizer (WOA, HHO, or GA) is run for 10 iterations with a population of 6, using the fitness function evaluated on the current simulation state.

• **Step 9 : Apply plan:**
The best signal plan found is applied to all intersections by updating phase durations via TraCI.

• **Step 10 : Repeat:**
The loop continues from Step 1.

3.10.2 Pseudocode

Algorithm 1: Predictive Meta-Heuristic Signal Control

Input : config_file, algorithm $\in$ {WOA, HHO, GA}, SIM_DURATION $= 3600$s
Output : CSV of metrics per step

```
Initialize predictor (LSTM)
Start SUMO via TraCI
for t_idx = 0 to SIM_DURATION/STEP_SIZE do
    step(STEP_SIZE = 5s)
    metrics ← collect_metrics()
    predictor.update_buffer(get_network_state())
    if (t_idx + 1) (mod HORIZON_STEPS) == 0 then
        if NOT predictor.is_ready() then
            continue // warming up
        pred ← predictor.predict()
        fitness_fn ← −lambda_plan : evaluate(plan, current_SUMO_state)
        if algorithm == WOA then
            best_plan ← WOA(fitness_fn, bounds = [15, 50], pop = 6, iter = 10)
        else if algorithm == HHO then
            best_plan ← HHO(fitness_fn, bounds = [15, 50], pop = 6, iter = 10)
        else
            best_plan ← GA(fitness_fn, bounds = [15, 50], pop = 6, iter = 10)
        apply_signal_plan(best_plan)
stop_simulation()
save results to CSV
```

3.10.3 LSTM Data Buffering Mechanism

The LSTM buffer is implemented as a simple FIFO queue using a Python list, with a maximum size of 24 entries. At each simulation step (every 5 seconds), the current 19-feature state vector is added to the buffer. Once the buffer is full (after 120 seconds), the oldest entry is removed whenever a new one is added, keeping only the most recent traffic data.

The method `is_ready()` returns `True` only when the buffer reaches 24 entries. During the first 120 seconds, the buffer is still being filled, so no optimization is performed. This explains the behavior observed in the logs: at $t = 60$ s, the system shows "Warming up LSTM (12/24)", meaning only half of the required data has been collected. At $t = 120$ s, the buffer is complete (`lstm_ready = True`), and the optimization process starts.

3.11 Chapter Conclusion

This chapter presented the methodology of the proposed traffic signal optimization framework. It combines five main components: the SUMO simulation, the TraCI interface for real-time interaction, the LSTM model for predicting future traffic, the fitness function for evaluating signal plans, and the optimization algorithms (WOA, HHO, and GA).

The system works in a rolling horizon, updating decisions every 60 seconds based on recent traffic data, with an initial 120-second warm-up period for the LSTM.

Three traffic scenarios (low, medium, and peak) were used to evaluate the framework under different conditions. The next chapter presents the results obtained from these experiments.

Chapter 4

Experiments and Results

4.1 Introduction

This chapter presents the results of applying the proposed framework to the Blida road network under three traffic scenarios. Four methods are compared: a fixed-time baseline, the Genetic Algorithm (GA), and two predictive approaches based on WOA and HHO combined with LSTM.

The assessment relies on four parameters: average waiting time, number of stops, total trip time, and throughput. Each method is tested five times in each scenario to make sure the results are accurate.

4.2 Experimental Setup

4.2.1 Hardware and Software Environment

All experiments were conducted on a Windows 11 machine with an Intel Core i5 processor and 8 GB of RAM. The entire simulation and optimization framework was built in Python 3.10, and it used these main libraries and tools:

Table 4.1: *Software Environment*

Component	**Tool / Version**	**Role**
Traffic simulator	SUMO 1.25.0	Microscopic simulation of the Blida network
TraCI interface	TraCI (Python)	Real-time communication with SUMO
Deep learning	PyTorch 2.0	LSTM model training and inference
Numerical computing	NumPy 1.24	Meta-heuristic algorithm implementation
Data analysis	Pandas 2.0	Results collection and CSV export
Visualization	Matplotlib 3.7	Figure generation
LSTM training	Google Colab (T4 GPU)	Model training environment

4.2.2 Simulation Parameters

There are 720 steps in each simulation run, and each step lasts 5 seconds. The whole run lasts 3600 seconds (one hour). The rolling horizon controller updates the signal plan every 60 seconds (every 12 steps). Before the LSTM predictor can make predictions, it needs to warm up for 120 seconds (24 steps). During this time, the signal plan stays at the baseline fixed-time setting. The meta-heuristic optimizers are set up to have a population size of 6 and a maximum of 10 iterations per optimization window. This means that each window can have up to 60 fitness evaluations, which is a budget that makes sure that each optimization finishes within the 60-second window in real time.

Table 4.2: *Simulation and Optimization Parameters*

Parameter	**Value**	**Description**
Simulation duration	3600 s	One full hour of traffic
Time step	5 s	Granularity of simulation
Horizon length	60 s	Signal plan update frequency
Horizon steps	12	Steps per optimization window
LSTM warm-up	120 s	Buffer filling period
Population size	6	Candidate plans per window
Max iterations	10	Optimizer iterations per window
Min green time	15 s	Lower bound on signal plan
Max green time	50 s	Upper bound on signal plan
Yellow time	4 s	Fixed clearance interval
Independent runs	5	Runs per method per scenario

4.2.3 Evaluation Metrics

There are four performance metrics used to rate and compare the four methods:

- **Average waiting time (s):** The average total waiting time for all vehicles at all intersections during each simulation step, averaged over the entire 3600-second run. This is the main optimization metric that the fitness function is aimed at.
- **Number of stops:** The average number of vehicles at a full stop (speed < 0.1 m/s) at all intersections during each simulation step. This number shows how often full stops happen, regardless of how long they last.
- **Total trip time (s):** The average amount of time that all active vehicles have spent in the network since they left, added up for all vehicles and divided by the number of steps. From the vehicle's point of view, this metric shows how well the network works overall.
- **Throughput (arrived vehicles):** The number of cars that successfully completed their trip and left the network during the 3600-second simulation. A higher throughput means that the network is working better.

4.3 Baseline Results: Fixed-Time Control

4.3.1 Baseline Performance Across Scenarios

The baseline represents a standard fixed-time signal control plan in which all nine intersections have a green light for 30 seconds at all times during the simulation, regardless of traffic conditions. Fixed-time control is a traditional and widely used approach in traffic signal systems, particularly in cases where real-time traffic data is not available and all adaptive methods are compared to it. Table below shows the baseline performance for all three scenarios. As expected, all four metrics get much worse as traffic demand goes up. The average wait time goes

Table 4.3: *Baseline Fixed-Time Performance Across Three Traffic Scenarios*

Scenario	Demand (veh/hr)	Avg Wait (s)	Avg Stops	Avg Trip Time (s)	Arrived
Low	200	446.9	2.8	661.6	33
Medium	500	1161.4	7.2	1754.6	89
Peak	900	1719.0	11.9	2899.4	174

up by a factor of 3.8 from the low to the peak scenario. This shows that a static 30-second green plan becomes less and less useful as the network gets closer to and goes over its practical capacity. As demand rises, the number of stops rises as well. This means that vehicles have to stop multiple times at the same intersection, which is called cascading or spillback in networks that are full. These baseline numbers set the minimum performance level that the adaptive methods must meet.

4.3.2 Baseline as Reference Point

The formula for figuring out the improvement percentage in this chapter is:

$$\text{Improvement (\%)} = \frac{\text{Baseline wait} - \text{Method wait}}{\text{Baseline wait}} \times 100$$

This applies for each baseline scenario. A positive number means that the average wait time has reduced compared to the fixed-time plan. A negative value, means that the method performed worse than the fixed-time plan.

4.4 Results : Low Traffic Scenario (200 veh/hr)

4.4.1 Waiting Time Comparison

Under low traffic conditions, when there are only 200 cars per hour on the road, the network operates well below its maximum capacity. In the baseline, all intersections have short lines and short wait times, so there isn't much room for improvement. Table below shows the results of a single run for each of the four methods when there is low traffic : Predictive HHO achieves the

Table 4.4: *Single run results: Low traffic scenario*

Method	Avg Wait (s)	vs Baseline	Arrived
Baseline	446.9	—	33
GA	427.6	+4.3%	37
WOA	318.6	+28.7%	43
HHO	287.7	+35.6%	40

best performance with a 35.6% reduction in average waiting time, followed by Predictive WOA at 28.7%. The GA achieves a modest 4.3% improvement, the only scenario in which it produces a positive result confirming that reactive optimization can be marginally effective when traffic demand is low and queue dynamics are simple enough to be captured without prediction. The

increase in arrived vehicles for WOA (43) and HHO (40) relative to the baseline (33) suggests that shorter waiting times allow vehicles to complete their trips within the 3600-second simulation window.

4.4.2 Four Metrics Analysis

The next table shows the statistical results of five separate runs for the low traffic scenario. It includes the mean and standard deviation for all four evaluation metrics: There is some vari-

Table 4.5: *Statistical results (mean ± std, 5 runs): Low traffic scenario*

Method	Avg Wait (s)	Std	Improvement	Avg Stops	Avg Trip (s)	Arrived
Baseline	446.9	—	—	2.8	661.6	33.0
GA	496.4	± 24.5	-11.1%	3.7 ± 0.16	822.3 ± 51.1	36.0
WOA	363.1	± 23.4	+18.8%	3.2 ± 0.14	702.9 ± 19.6	37.4
HHO	322.1	± 33.0	+27.9%	2.9 ± 0.16	625.9 ± 30.0	36.8

ation between runs in the standard deviations for WOA (±23.4s) and HHO (±33.0s), which is normal for stochastic algorithms. HHO has a slightly shorter average wait time than WOA, but it is less consistent, which means it can sometimes find better solutions.

The GA, on the other hand, shows a negative improvement on average (-11.1%), even though it did better in one run (+4.3%).This indicates that the single result was not reliable and that GA performs inconsistently, even under low traffic conditions.

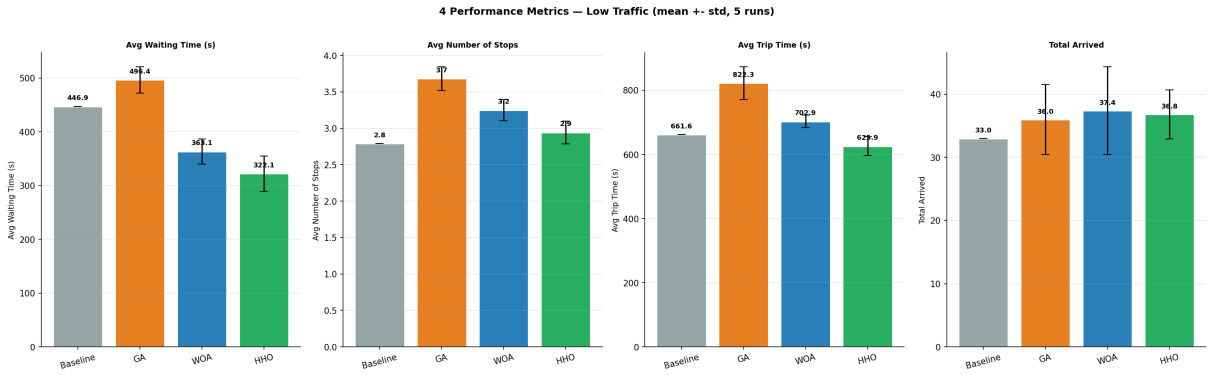


Figure 4.1: *Four performance metrics comparison for the Low traffic scenario (mean ± std, 5 runs)*

4.4.3 Discussion of Low Traffic Results

The results under low traffic show two main points. First, even when traffic is light and the baseline already performs well, the LSTM-based methods still improve results by 18–28%. This is because they can anticipate short bursts of congestion when groups of vehicles arrive at the same time.

Second, the GA shows inconsistent performance (+4.3% in a single run, but -11.1% on average), which confirms that reacting only to the current traffic state is not enough. Without prediction, the GA keeps optimizing for the present situation, which changes quickly, leading to less effective signal plans.

4.5 Results : Medium Traffic Scenario (500 veh/hr)

4.5.1 Waiting Time Comparison

The medium traffic scenario (500 vehicles/hour) represents typical midday urban conditions where multiple intersections experience moderate-to-heavy queuing. This scenario is the most representative of typical daily operation in the Blida city center and is used as the primary reference for comparing algorithm behavior throughout this chapter. Predictive HHO reduces

Table 4.6: *Single run results: Medium traffic scenario*

Method	Avg Wait (s)	vs Baseline	Arrived
Baseline	1161.4	—	89
GA	1188.8	-2.4%	83
WOA	828.6	+28.7%	83
HHO	708.6	+39.0%	92

the average waiting time by 39.0%, from 1161.4 seconds to 708.6 seconds, a decrease of more than 7.5 minutes per optimization step across the network. Predictive WOA also improves performance, with a reduction of 28.7%.

In contrast, the GA shows a slightly negative result (-2.4%), meaning it actually worsens performance. This suggests that under moderate traffic, reacting only to the current state is not enough. Without prediction, the GA may assign long green times to intersections with low traffic, causing unnecessary delays elsewhere.

4.5.2 Four Metrics Analysis

The table below shows the full results for the medium traffic scenario. This scenario highlights the biggest differences between the methods, making it the most useful for comparison.

Table 4.7: *Statistical results (mean ± std, 5 runs): Medium traffic scenario*

Method	Avg Wait (s)	Std	Improvement	Avg Stops	Avg Trip (s)	Arrived
Baseline	1161.4	—	—	7.2	1754.6	89.0
GA	1250.9	± 69.1	-7.7%	9.3 ± 0.43	2107.0 ± 97.5	94.8
WOA	848.1	± 29.3	+27.0%	7.7 ± 0.16	1689.7 ± 36.8	100.6
HHO	717.7	± 37.2	+38.2%	6.8 ± 0.21	1471.5 ± 52.5	90.6

Several points stand out. First, the GA has the highest variability (±69.1s), much larger than WOA with 2.4 times (±29.3s) and HHO with 1.9 times (±37.2s). This shows that GA is unstable under moderate traffic, as different runs can lead to very different results.

Second, HHO achieves the lowest average trip time (1471.5s compared to 1754.6s for the baseline), a reduction of 283 seconds, meaning vehicles spend on average nearly 5 minutes less in the network under HHO control.

Third, WOA achieves the highest throughput (100.6 vehicles compared to 89 for the baseline). This suggests that while WOA has slightly higher waiting times than HHO, it allows more vehicles to complete their trips during the simulation.

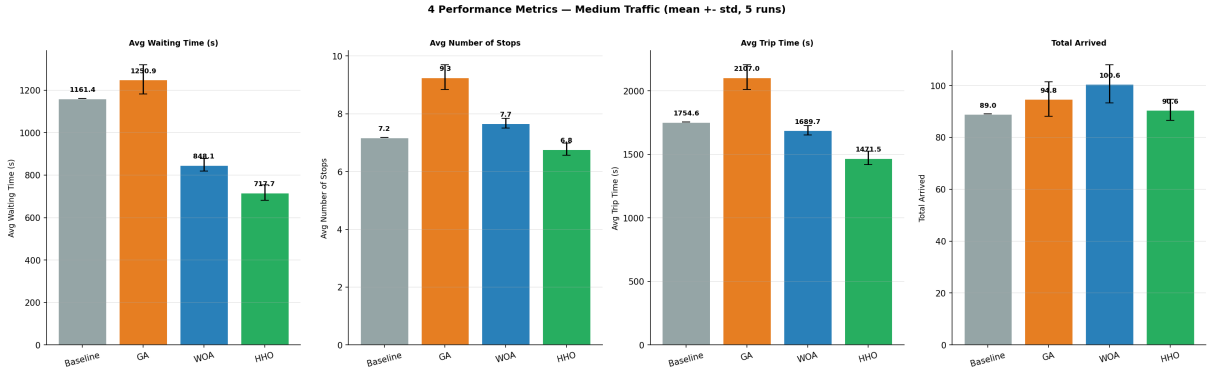


Figure 4.2: *Four performance metrics comparison for the Medium traffic scenario (mean ± std, 5 runs)*

4.5.3 Discussion of Medium Traffic Results

The medium traffic results clearly show the main advantage of the proposed framework. Under moderate congestion, where interactions between intersections make traffic more complex, the LSTM prediction makes a big difference.

By predicting traffic 60 seconds ahead, WOA and HHO can optimize signal timings before congestion builds up, helping to prevent long queues. In contrast, the GA, lacking this forward-looking capability, can only respond to congestion after it has already formed at which point the optimal plan has changed and the intervention is too late.

4.6 Results : Peak Traffic Scenario (900 veh/hr)

4.6.1 Waiting Time Comparison

The peak traffic scenario (900 vehicles/hour) represents rush-hour conditions, where several intersections operate close to or beyond their capacity. Under these conditions, the fixed-time baseline produces average waiting times of 1719 seconds (almost 29 minutes per step across the network) reflecting very bad congestion.

This scenario is used to test how well the proposed framework performs under the most difficult traffic conditions.

Table 4.8: *Single run results: Peak traffic scenario*

Method	Avg Wait (s)	vs Baseline	Arrived
Baseline	1719.0	—	174
GA	2018.6	-17.4%	168
WOA	1305.7	+24.0%	163
HHO	1127.8	+34.4%	168

The peak scenario shows the biggest differences between the methods. The GA performs worse than the baseline, with a 17.4% increase in waiting time. This means that if there is a lot of traffic, only responding to the current situation can make things worse. The GA usually gives more green time to intersections that are already clearing, which makes delays worse, while

ignoring the ones that are the most congested. Predictive HHO, on the other hand, cuts wait time by 34.4% (about 591 seconds), and Predictive WOA cuts wait time by 24.0%.

4.6.2 Four Metrics Analysis

The next table shows the results for the peak traffic scenario. This scenario also has the largest differences in variability between the methods, showing how sensitive they are under heavy congestion.

Table 4.9: *Statistical results (mean ± std, 5 runs): Peak traffic scenario*

Method	Avg Wait (s)	Std	Improvement	Avg Stops	Avg Trip (s)	Arrived
Baseline	1719.0	—	—	11.9	2899.4	174.0
GA	2047.4	± 42.4	-19.1%	15.7 ± 0.21	3577.6 ± 57.9	165.4
WOA	1395.6	± 127.1	+18.8%	13.2 ± 0.88	2944.3 ± 227.1	179.6
HHO	1123.5	± 36.5	+34.6%	11.4 ± 0.26	2560.5 ± 54.8	176.6

There are two main things to notice. First, WOA has a standard deviation of ±127.1s, which is much higher than HHO's standard deviation of ±36.5s. This means that WOA's results are less stable. This means that WOA may not work as well when there is a lot of traffic, when things change quickly. In contrast, HHO remains more consistent and finds good solutions more reliably.

Second, HHO cuts the average number of stops to 11.4, which is close to the baseline of 11.9. This shows that HHO helps keep traffic moving smoothly and cuts down on stop-and-go behavior, even when there are a lot of cars on the road.

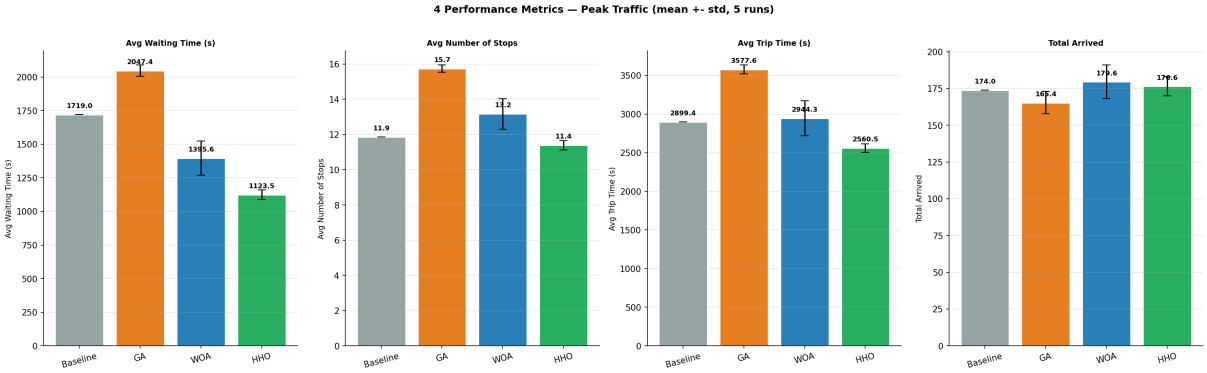


Figure 4.3: *Four performance metrics comparison for the Peak traffic scenario (mean ± std, 5 runs)*

4.6.3 Discussion of Peak Traffic Results

The peak scenario gives the clearest proof of the benefit of predictive control. In heavy traffic, lines can quickly move from one intersection to another in just a few seconds. A reactive method can't stop this because it only reacts after the traffic jam has already happened.

The LSTM prediction lets the system see traffic 60 seconds in advance and change the signal plan ahead of time. This allows WOA and HHO to act before congestion spreads, which explains their better performance compared to both the baseline and the GA.

4.7 Statistical Validation : 5 Independent Runs

4.7.1 Mean and Standard Deviation Results

To evaluate the statistical reliability of the reported results, each method was executed five times independently on each scenario, with a different random seed for the meta-heuristic population initialization in each run. The complete statistical summary is presented in Table below :

Table 4.10: *Complete statistical summary: mean ± std over 5 independent runs*

Scenario	**Method**	**Avg Wait ± Std (s)**	**Improvement**	**Avg Stops ± Std**	**Avg Trip ± Std (s)**
Low	Baseline	446.9 —	—	2.8 —	661.6 —
	GA	496.4 ± 24.5	-11.1%	3.7 ± 0.16	822.3 ± 51.1
	WOA	363.1 ± 23.4	+18.8%	3.2 ± 0.14	702.9 ± 19.6
	HHO	322.1 ± 33.0	+27.9%	2.9 ± 0.16	625.9 ± 30.0
Medium	Baseline	1161.4 —	—	7.2 —	1754.6 —
	GA	1250.9 ± 69.1	-7.7%	9.3 ± 0.43	2107.0 ± 97.5
	WOA	848.1 ± 29.3	+27.0%	7.7 ± 0.16	1689.7 ± 36.8
	HHO	717.7 ± 37.2	+38.2%	6.8 ± 0.21	1471.5 ± 52.5
Peak	Baseline	1719.0 —	—	11.9 —	2899.4 —
	GA	2047.4 ± 42.4	-19.1%	15.7 ± 0.21	3577.6 ± 57.9
	WOA	1395.6 ± 127.1	+18.8%	13.2 ± 0.88	2944.3 ± 227.1
	HHO	1123.5 ± 36.5	+34.6%	11.4 ± 0.26	2560.5 ± 54.8

4.7.2 Result Consistency Analysis

The standard deviations in previous table show how well each algorithm works when it runs multiple times. You can see a few patterns.

HHO is the most consistent in all scenarios. Its standard deviation stays below ±37.2 seconds for waiting time in the three cases, with a coefficient of variation (std/mean) of at most 10.3%. This stability comes from its energy-based mechanism, which allows a smooth transition from exploration to exploitation and makes it less sensitive to random initialization.

WOA shows much higher variation under peak traffic (±127.1 seconds, coefficient of variation 9.1%) compared to low and medium traffic (±23.4 and ±29.3 seconds). This means that under heavy congestion, WOA becomes more sensitive to the initial population, since its spiral exploration may or may not find a good solution within the 10 iterations.

GA shows the highest variation in absolute terms under medium traffic (±69.1 seconds), which confirms the instability of reactive optimization discussed earlier. Its performance depends strongly on the traffic state at each optimization step, leading to different results from one run to another.

Overall, these results show that HHO is the most reliable algorithm for real-time use, as it provides the best average performance with the lowest variation across all traffic scenarios.

4.8 Temporal Evolution Analysis

4.8.1 Waiting Time Over Simulation Time

Figure 4.4 shows the temporal evolution of total network waiting time over the 3600-second simulation for all four methods under each traffic scenario.The curves are smoothed using a rolling average of 10 steps (50 seconds) to reduce high-frequency noise while preserving the underlying trends.

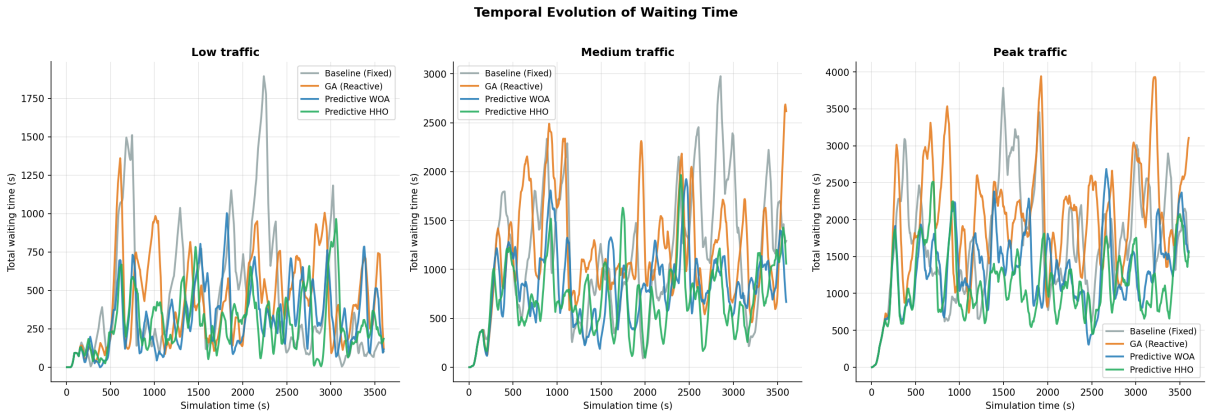


Figure 4.4: *Temporal evolution of total network waiting time for all four methods under Low (left), Medium (center), and Peak (right) traffic scenarios*

Under low traffic, all methods show similar and smooth behavior during the first 120 seconds, which corresponds corresponding to the LSTM warm-up period during which no optimization is performed. After 120 seconds (when the LSTM buffer reaches its required 24 steps and the optimization starts), the WOA and HHO curves begin to drop below the baseline. HHO stays consistently lower than WOA throughout the simulation.

The GA curve fluctuates around the baseline, sometimes slightly better and sometimes worse, confirming its inconsistent behavior under low traffic.

Under medium and peak traffic, the behavior becomes more complex. The baseline and GA curves show repeated spikes in waiting time sharp increases followed by partial decreases which reflect the typical pattern of congestion in saturated networks, where queues build up, partially clear, and then form again.

In contrast, the WOA and HHO curves show much smaller spikes. This means that the predictive framework acts earlier, before congestion reaches its peak. This early intervention is the main advantage of prediction: it does not remove congestion completely, but it reduces its intensity and helps the system recover faster

4.8.2 Algorithm Behavior During Peak Congestion

A particularly informative observation from the temporal curves concerns the behavior of all methods during congestion peaks. In the medium scenario, GA frequently reaches waiting times of 2500 seconds during congestion peaks significantly higher than even the fixed-time baseline (approximately 1500–2000 seconds) while WOA and HHO maintain waiting times mostly below 1500 seconds throughout the simulation.

In the peak scenario, GA and the baseline both reach spikes of 3000–4000 seconds, whereas WOA remains generally below 2500 seconds and HHO consistently tracks the lowest trajectory among all methods.

These observations confirm two key findings: first, that GA without prediction actively worsens congestion under heavy demand, sometimes performing worse than doing nothing; and second, that HHO with LSTM prediction is the most effective at limiting peak congestion severity across all three scenarios.

4.9 Overall Comparison and Discussion

4.9.1 Improvement Percentage Across All Scenarios

The figure below summarizes the improvement percentage of each method relative to the fixed-time baseline across all three traffic scenarios. The figure provides a consolidated view of the relative performance of GA, WOA, and HHO under low, medium, and peak demand conditions.

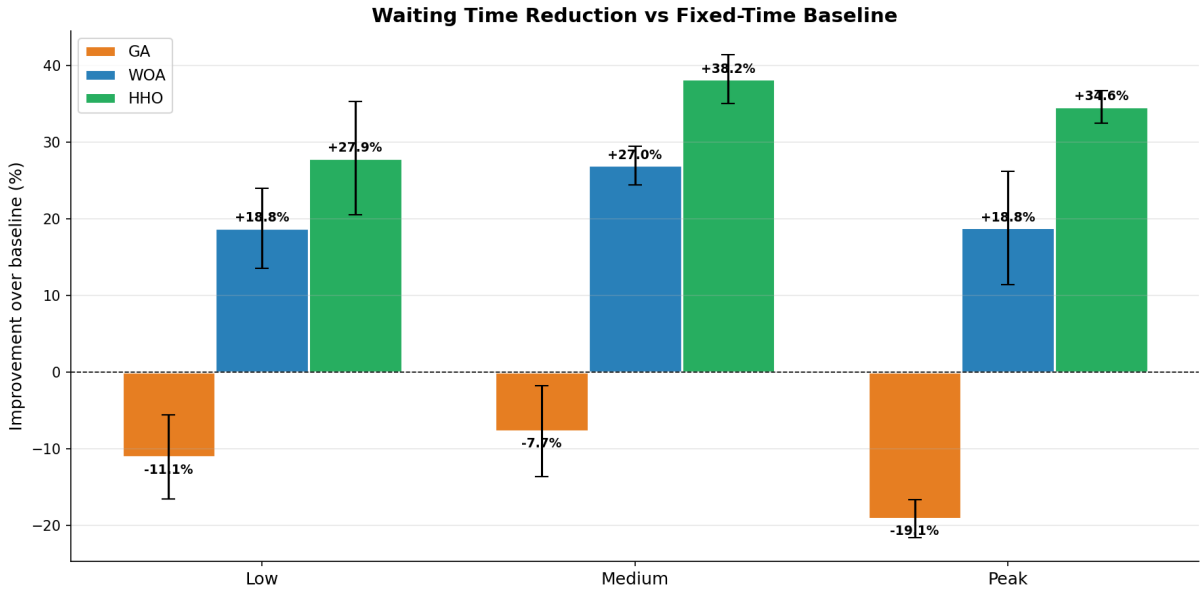


Figure 4.5: *Waiting time improvement percentage relative to fixed-time baseline for all three methods across three traffic scenarios (mean ± std, 5 runs)*

The figure reveals three clear performance tiers. HHO consistently occupies the top tier with improvements of +27.9%, +38.2%, and +34.6% across low, medium, and peak scenarios. WOA forms a middle tier with improvements of +18.8%, +27.0%, and +18.8% consistently positive but lower than HHO. GA occupies the bottom tier with negative improvements of -11.1%, -7.7%, and -19.1% consistently worse than the fixed-time baseline across all scenarios when evaluated over five runs. The performance gap between GA and the LSTM-guided algorithms widens as traffic demand increases, confirming the hypothesis that prediction becomes increasingly important under heavier congestion.

4.9.2 Heatmap Analysis

The next figure presents a heatmap visualization of the average waiting time for all methods across all scenarios. The color encoding, red for high waiting times and green for low provides an immediate visual summary of the relative performance of each method.

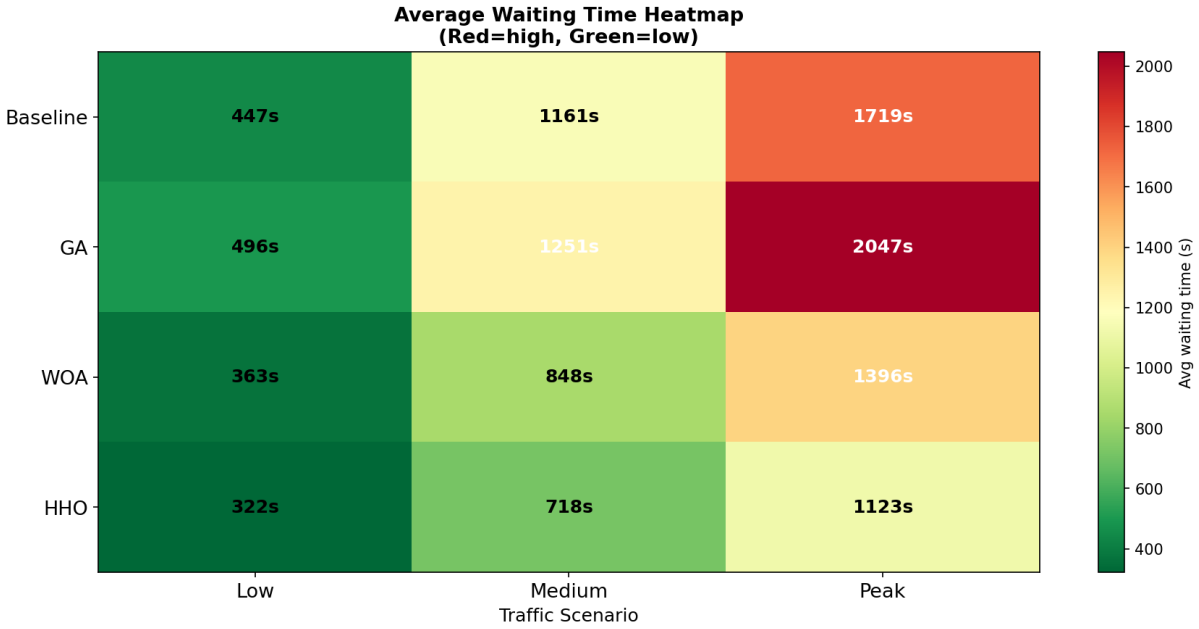


Figure 4.6: *Heatmap of average waiting time: rows = methods, columns = scenarios. Red = high congestion, green = low congestion*

The heatmap clearly shows a consistent ranking across all scenarios. HHO achieves the lowest waiting times in all cases (darkest green), followed by WOA, then the fixed-time baseline, while GA consistently performs the worst.

The most notable result appears under peak traffic, where GA reaches 2047 seconds (dark red), exceeding the baseline value in the same column (1719 seconds). This confirms that reactive optimization without prediction can worsen congestion under heavy demand.

In contrast, HHO maintains the lowest waiting times across all scenarios, with a value of 1123 seconds under peak traffic (light green). The difference between HHO and GA in this case is 924 seconds, showing a large performance gap between the best and worst methods.

The medium scenario shows a more balanced distribution of values, reflecting moderate congestion where differences between methods are still significant but less extreme than in the peak case.

4.9.3 Why GA Fails Under Heavy Traffic

The consistently negative performance of the Genetic Algorithm across all scenarios and its increasing degradation as traffic demand rises warrants careful explanation, as it is one of the most important findings of this study.

The main limitation of GA in this framework is that it is purely reactive. It optimizes signal plans based only on the current traffic state, without considering what will happen in the next 60 seconds. Under low traffic, this is less of a problem because traffic changes slowly, so the current state is close to the near future. However, under heavy traffic, conditions change quickly: a small queue at the moment of optimization can become large by the time the signal plan is applied.

As a result, the plan generated by GA which is based on the current state may no longer be suitable when it is actually used. In some cases, it can even make congestion worse by giving more green time to intersections that are already clearing, while ignoring those where queues are growing.

This result confirms the main idea of the proposed framework: in a network with moderate to heavy traffic, prediction is not just helpful, but necessary. By predicting traffic 60 seconds ahead, the LSTM allows WOA and HHO to optimize for the future state of the network, not just the current one, which leads to better performance than both GA and the fixed-time baseline.

4.9.4 Why HHO Outperforms WOA

Across all traffic scenarios and evaluation metrics, HHO consistently performs better than WOA. The difference is especially clear for waiting time (for example, +34.6% vs +18.8% under peak traffic) and in stability (±36.5s vs ±127.1s at peak). This can be explained by several differences in how the algorithms work.

First, HHO uses an energy-based mechanism to move from exploration to exploitation. As the escaping energy envelope decreases progressively toward zero, the algorithm gradually focuses on the most promising area of the search space in a controlled and predictable way. In contrast, WOA switches randomly between encircling and spiral behavior. While this can work well in longer runs, it is less reliable when only 10 iterations are allowed in a real-time setting.

Second, HHO uses Lévy flights, which allow occasional large jumps in the search space. This helps the algorithm escape local optima, which is especially important under peak traffic where the fitness landscape is irregular and changes quickly. WOA's spiral movement works well in smoother conditions but is less effective when the landscape has sharp variations caused by queue interactions.

Third, the signed difference correction applied to HHO's hard besiege strategy (introduced in Chapter 3) ensures that solutions can move in both directions around the current best solution. This helps maintain diversity in the population during the 10 iterations, which explains why HHO shows lower variability compared to WOA.

4.10 Chapter Conclusion

This chapter presented the experimental evaluation of the proposed framework across three traffic scenarios and four methods. The results show that the LSTM-based approaches (WOA and HHO) consistently perform better than both the fixed-time baseline and the reactive GA across all scenarios and metrics. The main results are summarized below.

Table 4.11: *Summary of best results (HHO) across all scenarios*

Scenario	Baseline Wait	HHO Wait	Improvement	Stops Reduced	Trip Time Reduced
Low	446.9s	322.1s	+27.9%	2.8 → 2.9	661.6 → 625.9s
Medium	1161.4s	717.7s	+38.2%	7.2 → 6.8	1754.6 → 1471.5s
Peak	1719.0s	1123.5s	+34.6%	11.9 → 11.4	2899.4 → 2560.5s

The next chapter discusses these results in relation to existing work and presents the limitations and possible improvements of the proposed approach.

Chapter 5

Discussion, Conclusion and Perspectives

5.1 Introduction

This chapter discusses the results of the experiments in four sections: a general conclusion summarizing the main contributions of the thesis, a comparison with related work in the literature, an acknowledgement of the limitations of the proposed framework, and three practical directions through which this work could be extended in the future.

5.2 Comparison with Related Work

Any research contribution must be evaluated meaningfully by placing its findings in the context of previous research. The performance of the suggested framework is compared in this section with the most pertinent research found in the literature review, with an emphasis on studies that deal with traffic signal optimization utilizing LSTM-based prediction, meta-heuristic methods, or both.

5.2.1 Meta-Heuristic Optimization Studies

Jamal et al. (2020)[7], one of the most closely comparable studies, used Differential Evolution and Genetic Algorithms to optimize signal timing at isolated intersections, demonstrating improvements in average delay of roughly 15–20% over fixed-time plans under moderate demand. When the GA implementation is applied reactively to a network with nine intersections, performance deteriorates under medium and peak conditions (-7.7% and -19.1%, respectively) and only slightly improves under low traffic (+4.3%).

This work reveals a key difference: single-intersection reactive GA cannot scale to multi-intersection networks under dynamic demand, but it can be efficient under controlled conditions.

The Whale Optimization Algorithm was directly applied to the traffic signal scheduling problem by Thaer Thahe et al. (2019)[9] , who reported decreases in average trip time across extensive benchmarks. However, neither predictive elements nor real-time simulation feedback were included in their study.

In three realistic traffic scenarios, our Predictive WOA, which incorporates LSTM forecasting within a rolling horizon framework, achieves waiting time improvement of 18.8% to +28.7%. This shows that the combination of WOA with prediction significantly expands its applicability to dynamic urban conditions.

An enhanced HHO algorithm for traffic signal timing was proposed by Zhao and Su (2024)[10], who reported a 36.7% decrease in average vehicle delay. The Predictive HHO, which attains reductions of +27.9% (low), +38.2% (medium), and +34.6% (peak), is directly comparable to this outcome. Using a standard HHO variation instead of an upgraded one, the implementation achieves a larger improvement under medium traffic (+38.2% vs. 36.7%); the main distinction is the incorporation of LSTM prediction rather than algorithmic changes to HHO itself. This implies that for real-time traffic control applications, predictive integration is at least as advantageous as algorithmic improvement.

On the Montreal dataset, Khasawneh and Awasthi (2023)[13] proposed the Enhanced Bat Algorithm (EBAT) in conjunction with LSTM and other predictive models for traffic signal optimization, reporting notable decreases in travel time and pollution emissions. Their hybrid technique, which combines a predictive module with a meta-heuristic optimizer, is conceptually similar to the developed approach and supports the basic framework design.

However, the methodology presented in this work simultaneously performs online prediction and optimization within a real-time simulation loop, while their work focuses on arterial-level optimization with pre-computed predictions

5.2.2 Deep Learning and Prediction Studies

The literature of survey includes a number of studies that use LSTM for traffic prediction without explicitly integrating the forecasts to a signal optimizer. Partitioned LSTM networks were used by Zainab Abbas et al. (2018)[4] to predict short-term traffic flow on the Stockholm freeway with competitive accuracy. On Beijing road data, Mou et al. (2019)[5] demonstrated higher prediction accuracy using T-LSTM, an improved LSTM that incorporates temporal information vectors. On the held-out test set, our implemented LSTM model achieves an R^2 of 0.923 and an MAE of 173 seconds. This impressive result validates the feasibility of LSTM-based prediction for 60-second ahead forecasting in a multi-intersection environment, in line with the accuracy levels reported in earlier research.

In order to increase prediction accuracy above baseline LSTM, Xu et al. (2022)[23] employed WOA to tune the hyperparameters of a BiLSTM-Attention model for traffic flow prediction. This work further verifies the efficacy of WOA as a search algorithm in traffic-related optimization problems and represents a complementary usage of WOA, applied to the training phase of a deep learning model instead of real-time signal optimization.

5.2.3 Limitations of the Proposed Framework

A thorough scientific evaluation requires the limitation of the proposed methodology to be overly acknowledged.

- **Scale of the network:** The framework was tested on a network of nine signalized junctions with a total area of around 830 × 830 meters. This is adequate to show multi-intersection coordination, but it does not fully convey the intricacy of city-wide signal optimization, where demand redistribution and interactions between distant junctions via arterial green waves are crucial. As the network size grows, the framework's computational cost, roughly 18 seconds per optimization window for nine intersections, may also become a bottleneck.

- **Environment for simulation:** Using artificial traffic demand produced by a random trip generator, all trials were carried out in the SUMO simulator. The generated demand does not reflect the spatiotemporal patterns of actual traffic, such as origin-destination flows, day-of-week changes, incident-driven demand shifts, or the impact of weather and events, even though SUMO accurately predicts vehicle dynamics and signal phase transitions. As a result, the performance gains described in this thesis might not match those that can be attained in actual deployment.

- **Route files have been fixed:** A predetermined set of vehicle routes created prior to the simulation defines the traffic demand in each scenario. This paradigm does not account for the feedback loop between signal designs and demand patterns that occurs when drivers reroute in response to traffic. To take this interaction into consideration, a dynamic traffic assignment model would be needed.

- **Optimization with a single goal:** Other significant goals like fuel consumption, car emissions, public transportation priority, and pedestrian crossing safety are not specifically

taken into consideration by the fitness function, which minimizes a weighted combination of line length and waiting time. More balanced and useful signal designs could be produced using a multi-objective formulation.

- **Lack of actual sensor data:** Instead of using actual sensor measurements, the LSTM model was trained using data produced by the SUMO simulator. The model's performance on real sensor data, which is noisier, prone to missing values, and impacted by detector failures, has not been assessed, despite its great predicted accuracy on the simulated data (R2 = 0.923). Hardware verification: Hardware for edge or embedded computing has not been used to test the framework. Optimizing the Python implementation for lower-power processors and verifying the 60-second optimization budget on target hardware would be necessary for real-world deployment.

5.3 Future Perspectives

Three practical and scientifically driven avenues for further research are suggested based on the findings and constraints noted in this thesis.

5.3.1 Extension to Larger Networks

Scaling the framework to larger road networks with more than nine crossings is the most obvious extension of this study. Applying the framework to a corridor-level network that connects many districts or to the entire Blida city center network (95 junctions seen in the OSM extraction) would be a logical next step.

Two technical difficulties arise when scaling to larger networks: the optimization problem's dimensionality grows linearly with the number of intersections, possibly surpassing the current population-based search's 60-second budget; and the interactions between far-off intersections become more intricate, necessitating the LSTM model to capture longer-range spatial dependencies. Decomposition techniques, which divide the network into overlapping subnetworks that are optimized independently, or more potent optimization backbones, such Particle Swarm Optimization with cooperative sub-swarms, may be needed to address these issues.

5.3.2 Integration of Reinforcement Learning

The meta-heuristic strategy used in this thesis is complemented by Reinforcement Learning (RL). RL agents learn a policy that maps traffic conditions to signal actions through recurrent contact with the environment, potentially capturing long-term dependencies that a 60-second horizon cannot handle, whereas meta-heuristics maximize a pre-defined fitness function at each horizon. Comparing the suggested framework with a cutting-edge DRL controller (such Deep Q-Network or Proximal Policy Optimization) utilizing the same network, demand scenarios, and evaluation metrics under the same simulation settings is an interesting future direction.

A rigorous benchmark for both paradigms would be provided by such a comparison, which might also show if the interpretability benefit of meta-heuristics outweighs any performance difference with learned policies.

5.3.3 Real-Time Deployment with Physical Sensors

The deployment of the suggested framework on an actual urban intersection network equipped with actual traffic sensors serves as the final validation. The performance gains shown in simulation could be verified under actual traffic conditions with a pilot deployment in the Blida city center that uses inductive loop detectors or video-based vehicle counting systems to feed real-time queue and waiting time measurements to the LSTM and optimizer. Sensor calibration and noise filtering, communication latency between detectors and the central controller, backup plans for sensor failures, and integration with the current signal control infrastructure overseen by the local traffic authority are some of the practical issues that must be resolved during this shift from simulation to deployment. Such a deployment would require cooperation with the Blida Wilaya traffic management services.

5.4 General Conclusion

The subject of dynamic traffic signal optimization in multi-intersection metropolitan networks was the focus of this thesis. As vehicle populations rise and fixed-time signal plans become less effective at controlling daily congestion peaks, this challenge is becoming more and more practical. Can a hybrid framework that combines population-based meta-heuristic optimization with short-term traffic prediction outperform both fixed-time control and reactive optimization over a variety of traffic demand levels? This was the main research question .

In order to address this question, a rolling horizon control framework was developed, implemented, and experimentally evaluated. The framework that integrated three components: the SUMO microscopic simulator functioning as the virtual urban environment via the TraCI real-time interface; a meta-heuristic optimizer (WOA, HHO, or GA) applied at each 60-second horizon to find the signal plan minimizing a red-time cost fitness function; and an LSTM neural network trained to predict total network waiting time 60 seconds in advance.
Three traffic scenarios: low, medium, and peak were used to test the framework on a nine-intersection road network extracted from OpenStreetMap data of the city center of Blida, Algeria.

The experimental results show that the suggested framework accomplishes its goals, validated across five independent runs per technique per scenario.
In comparison to the fixed-time baseline for low, medium, and high traffic, predictive HHO reduces average waiting time by 27.9%, 38.2%, and 34.6%, respectively. Reductions of 18.8%, 27.0%, and 18.8% are attained using predictive WOA. Predictive information is crucial for efficient signal optimization under high congestion, as demonstrated by the clear experimental proof that both LSTM-guided algorithms outperform the reactive Genetic Algorithm, which reduces efficiency by up to 19.1% under peak conditions. With the best average performance and the least run-to-run fluctuation in any case, HHO is the most dependable optimizer.

These findings establish the suggested framework as a competitive contribution to the field of intelligent traffic signal control, achieving improvements that are on par with or better than those documented in related meta-heuristic and hybrid studies while adding the extra rigor of statistical validation and actual OSM-based network evaluation in the urban setting of Algeria.

AI Disclosure Statement

The authors used Claude (Anthropic, Sonnet 4.6) for the following tasks: literature review structuring, grammar correction, LaTeX equation formatting, and table organization.
All AI-generated content was reviewed, verified, and substantially rewritten by the authors. The following components are entirely the authors' own work: SUMO simulation configuration, Python implementation of WOA/HHO/GA, LSTM training code, fitness function design, all experimental results, and the signed difference modification to HHO (in Section 3.8.4). The authors take full responsibility for the accuracy of all content.

Bibliography

[1] B. Abdulhai, H. Porwal, and W. Recker, "Short-term traffic flow prediction using neuro-genetic algorithms," *Journal of Intelligent Transportation Systems*, vol. 7, no. 1, pp. 3–41, 2002.

[2] A. Sadeghi-Niaraki, P. Mirshafiei, M. Shakeri, and S.-M. Choi, "Short-term traffic flow prediction using the modified elman recurrent neural network optimized through a genetic algorithm," *IEEE Access*, vol. 8, pp. 217526–217540, 2020.

[3] N. E. Ren, L. W. Tang, Y. H. Yin, and Y. D. Wang, "Short-term traffic flow prediction based on deep learning model," in *E3S Web of Conferences*, vol. 253, p. 01025, EDP Sciences, 2021.

[4] Z. Abbas, A. Al-Shishtawy, S. Girdzijauskas, and V. Vlassov, "Short-term traffic prediction using long short-term memory neural networks," in *2018 IEEE international congress on big data (BigData Congress)*, pp. 57–65, IEEE, 2018.

[5] L. Mou, P. Zhao, H. Xie, and Y. Chen, "T-lstm: A long short-term memory neural network enhanced by temporal information for traffic flow prediction," *Ieee Access*, vol. 7, pp. 98053–98060, 2019.

[6] H. Zheng, F. Lin, X. Feng, and Y. Chen, "A hybrid deep learning model with attention-based conv-lstm networks for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 11, pp. 6910–6920, 2020.

[7] A. Jamal, M. Tauhidur Rahman, H. M. Al-Ahmadi, I. Ullah, and M. Zahid, "Intelligent intersection control for delay optimization: Using meta-heuristic search algorithms," *Sustainability*, vol. 12, no. 5, p. 1896, 2020.

[8] A. Jamal, H. M. Al-Ahmadi, F. M. Butt, M. Iqbal, M. Almoshaogeh, and S. Ali, *Meta-heuristics for traffic control and optimization: Current challenges and prospects*. IntechOpen, 2021.

[9] T. Thaher, B. Abdalhaq, A. Awad, and A. Hawash, "Whale optimization algorithm for traffic signal scheduling problem," 2019.

[10] H. Zhao and M. Su, "Traffic signal timing scheme based on the improved harris hawks optimisation," *Promet–Traffic & Transportation*, vol. 36, no. 2, pp. 294–306, 2024.

[11] T. Mao, A.-S. Mihăită, F. Chen, and H. L. Vu, "Boosted genetic algorithm using machine learning for traffic control optimization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7112–7141, 2021.

[12] S. Kar, "Optimization of traffic signals for intelligent traffic light management system," in *Mechatronics and Automation*, IIP Series, pp. 145–160, Iterative International Publishers, 2023.

[13] M. A. Khasawneh and A. Awasthi, "Intelligent meta-heuristic-based optimization of traffic light timing using artificial intelligence techniques," *Electronics*, vol. 12, no. 24, p. 4968, 2023.

[14] A. Vidali, L. Crociani, G. Vizzari, S. Bandini, *et al.*, "A deep reinforcement learning approach to adaptive traffic lights management.," in *Woa*, pp. 42–50, 2019.

[15] X. Ma, X. Hu, T. Weber, and D. Schramm, "Evaluation of accuracy of traffic flow generation in sumo," *Applied Sciences*, vol. 11, no. 6, p. 2584, 2021.

[16] M. L. Clemente, "Building a real-world traffic micro-simulation scenario from scratch with sumo," in *SUMO conference proceedings*, vol. 3, pp. 215–230, 2022.

[17] D. Agrawal, D. Vinod, and J. Rastogi, "A sumo based simulation framework for intelligent traffic management system," in *Proceedings of the 3rd International Conference on Optimization Techniques in the Field of Engineering (ICOFE-2024)*, 2024.

[18] A. Schaffland, J. Nelson, and J. Schöning, "Simulating traffic networks: Driving sumo towards digital twins," in *SUMO Conference Proceedings*, vol. 5, pp. 113–125, 2024.

[19] Y. Zhang, M. Fu, and X. Ban, "Integration traffic signal control from synchro to sumo," in *SUMO Conference Proceedings*, vol. 5, pp. 147–162, 2024.

[20] P. Heckelmann and S. Rinderknecht, "Influence of an automated vehicle with predictive longitudinal control on mixed urban traffic using sumo," *World Electric Vehicle Journal*, vol. 15, no. 10, p. 448, 2024.

[21] O. Dobrilko and A. Bublil, "Leveraging sumo for real-world traffic optimization: A comprehensive approach," in *SUMO Conference Proceedings*, vol. 5, pp. 179–194, 2024.

[22] M. Elharoun, S. M. El-Badawy, E. A.-E. Shwaly, and U. E. Shahdah, "Adaptive traffic signal control using deep reinforcement learning: A multi-objective approach for single and multi-intersection scenarios," *IATSS Research*, vol. 49, no. 4, pp. 481–492, 2025.

[23] X. Xu, C. Liu, Y. Zhao, and X. Lv, "Short-term traffic flow prediction based on whale optimization algorithm optimized bilstm_attention," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 10, p. e6782, 2022.

[24] Z. Liu, X. Li, Z. Lu, and X. Meng, "Iwoa-rnn: An improved whale optimization algorithm with recurrent neural networks for traffic flow prediction," *Alexandria Engineering Journal*, vol. 117, pp. 563–576, 2025.

[25] P. T. Martin, J. Perrin, B. R. Chilukuri, C. Jhaveri, and Y. Feng, "Adaptive signal control II," Research Report MPC-03-141, University of Utah, Department of Civil and Environmental Engineering, Salt Lake City, UT, USA, 2003.

[26] F. V. Webster, "Traffic signal settings," Road Research Technical Paper 39, Road Research Laboratory, London, UK, 1958.

[27] M. Eom and B.-I. Kim, "The traffic signal control problem for intersections: a review," *European transport research review*, vol. 12, no. 1, p. 50, 2020.

[28] R. A. Vincent, A. I. Mitchell, and D. I. Robertson, "User guide to TRANSYT version 8," TRRL Laboratory Report LR 888, Transport and Road Research Laboratory, Crowthorne, Berkshire, UK, 1980.

[29] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in engineering software*, vol. 95, pp. 51–67, 2016.

[30] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[31] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future generation computer systems*, vol. 97, pp. 849–872, 2019.

[32] J. Schmidhuber, S. Hochreiter, *et al.*, "Long short-term memory," *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.

[33] 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*, vol. 54, pp. 187–197, 2015.

[34] Z. Zhao, W. Chen, X. Wu, P. C. Chen, and J. Liu, "Lstm network: a deep learning approach for short-term traffic forecast," *IET intelligent transport systems*, vol. 11, no. 2, pp. 68–75, 2017.

[35] R. Fu, Z. Zhang, and L. Li, "Using lstm and gru neural network methods for traffic flow prediction," in *2016 31st Youth academic annual conference of Chinese association of automation (YAC)*, pp. 324–328, IEEE, 2016.

[36] X. Liang, X. Du, G. Wang, and Z. Han, "A deep reinforcement learning network for traffic light cycle control," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1243–1253, 2019.

[37] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, the third international conference on advances in system simulation*, ThinkMind, 2011.

[38] DLR (German Aerospace Center), "TraCI (Traffic Control Interface) API documentation." https://sumo.dlr.de/docs/TraCI/index.html, 2026. Accessed: 2026.