People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Mohamed El Bachir El Ibrahimi – Bordj Bou Arréridj
Faculty of Mathematics and Computer Science
Department of Computer Science

THESIS
Submitted in partial fulfillment for the degree of
Master in Computer Science
Specialty : Networks and Multimedia

TOPIC

Deep Learning Model for Intelligent Location Prediction in VANETs : LSTM-CNN A Hybrid Approach

Presented by :
Maroua DJOUDI
Hanane BENHAMMOUDA

Defended publicly on : June 10, 2025
Before the jury composed of :
President : Dr. Nadjib BENAOUDA
Examiner : Dr. Adel SAHA
Supervisor : Dr. Boubakeur MOUSSAOUI

Academic Year : 2024/2025

Dedication

I dedicate this humble work :

To my dear parents, whose love, sacrifices, support, and prayers have guided me throughout my studies and my life.

To my beloved brothers : Mohammed Nadjib, Ayoub, Alilou, Djamel, and Mohammed Amine.

To my dear sisters and dear friends : Rania, Yousra, Sirine, Hanane, Baya, Nadine, and Afaf — thank you for your constant encouragement and presence.

A special and heartfelt thanks to Dr. Moussaoui Boubakeur, my supervisor, for his continuous support, trust, and dedication. His encouragement and wise guidance have been a constant source of motivation throughout this journey.

Finally, to all those who have inspired me, supported me, or simply believed in me directly or indirectly, I am deeply grateful.

Djoudi Maroua

Dedication

First and foremost, thank God for giving me strength and guidance throughout this journey.

To my dear parents : Your love, sacrifices, and constant belief in me gave me the strength to keep going. I owe this journey to you.

To my brothers : Zakaria, Saber, Karim, and Rami. Thank you for your quiet support and presence along the way.

To my friends : Baya, Maroua, Nadine, Ichrak, Ibtissem, Leila, and Naima. You've been my light, my laughter, and my strength. Your friendship carried me through every hard moment.

To my precious aunt Nawja, Your quiet love and kind heart have always been a source of peace for me.

To my supervisor, Dr. Moussaoui Boubakeur, Thank you for your guidance, patience, and invaluable support. Your mentorship made this work possible.

This work is a part of me, and it belongs to all of you, with all my love.

Benhammouda Hanane

Acknowledgments

First and foremost, to **Allah**, the Most Merciful and the Most Generous, for granting us strength, patience, and guidance throughout this journey.

We warmly thank our supervisor, **Dr. Boubakeur Moussaoui**, for his availability, his valuable advice, his kindness and for the trust he placed in us throughout this work.

We also extend a special thank you to **Ms. Soumia Bibi**, who accompanied us with care and kindness from the very beginning of our project. Her initial support and guidance greatly helped us get this work off to a good start and we are sincerely grateful.

We also extend our sincere thanks to the members of the jury for their interest in our work, their constructive comments, and their contribution to our scientific progress.

We also thank our instructors for the quality of the teaching provided throughout our university studies, and the Computer Science Department for the resources made available to us.

Finally, we express our gratitude to our families and friends for their moral support, patience, and constant encouragement throughout our studies.

Résumé

Cette recherche porte sur la prédiction de la localisation des véhicules dans le contexte des systèmes de transport connectés et intelligents (STI-C), notamment au sein des réseaux ad hoc véhiculaires (VANET). Une localisation précise et fiable est essentielle à la sécurité, au routage et à la coordination des véhicules autonomes et connectés. Cependant, les signaux GPS sont souvent dégradés ou indisponibles dans des scénarios réels tels que les canyons et les tunnels urbains.

Pour relever ce défi, nous proposons une architecture hybride basée sur l'apprentissage profond, combinant des réseaux de neurones convolutifs (CNN) et des modèles de mémoire à long terme (LSTM). Le système est entraîné et évalué sur des jeux de données de trajectoires synthétiques simulant des schémas de conduite réalistes (circulaires, hélicoïdaux et en L). Diverses mesures d'erreur, telles que MSE, RMSE, MAE, R^2 Score, RPE et MADE, sont utilisées pour évaluer les performances.

Les résultats démontrent que l'architecture LSTM-CNN surpasse les autres modèles en termes de précision de localisation, ce qui en fait une solution prometteuse pour les applications en temps réel dans des environnements véhiculaires dynamiques.

Abstract

This research focuses on vehicle location prediction in the context of connected and intelligent transportation systems (C-ITS), particularly within vehicular ad hoc networks (VANETs). Accurate and reliable location is essential for the safety, routing, and coordination of autonomous and connected vehicles. However, GPS signals are often degraded or unavailable in real-world scenarios such as urban canyons and tunnels.

To address this challenge, we propose a hybrid architecture based on deep learning that combines convolutional neural networks (CNNs) and long short-term Memory (LSTM) models. The system is trained and evaluated on synthetic trajectory datasets simulating realistic driving patterns (circular, helical, and L-shaped). Various error metrics such as MSE, RMSE, MAE, R^2 Score, RPE, and MADE are used to evaluate performance.

The results demonstrate that the LSTM-CNN architecture outperforms other models in terms of localization accuracy, making it a promising solution for real-time applications in dynamic vehicular environments.

ملخص

يُركّز هذا البحث على مسألة التنبؤ بموقع المركبات ضمن سياق أنظمة النقل المتصلة والذكية (C-ITS)، لا سيّما في إطار الشبكات المخصصة للمركبات (VANETs). يُعدّ الحصول على موقع دقيق وموثوق به عاملاً جوهريًا لضمان السلامة وتوجيه المركبات المتصلة وذاتية القيادة وتنسيق حركتها. إلا أن إشارات نظام التموضع العالمي (GPS) قد تكون ضعيفة أو غير متوفرة في بعض البيئات الواقعية مثل الأنفاق أو المناطق الحضرية العميقة.

لمعالجة هذا التحدي، يقترح هذا العمل بنية هجينة تعتمد على تقنيات التعلم العميق، تجمع بين الشبكات العصبية الالتفافية (CNN) ونماذج الذاكرة طويلة وقصيرة المدى (LSTM). وقد تم تدريب النموذج وتقييمه باستخدام مجموعات بيانات لمسارات اصطناعية تُحاكي أنماط قيادة حقيقية (دائرية، حلزونية، وعلى شكل حرف L)، وتم الاعتماد على مؤشرات أداء متنوعة مثل: MSE وRMSE وMAE وR^2 Score وRPE وMADE لقياس الدقة والفعالية.

أظهرت النتائج أن النموذج الهجين (LSTM-CNN) يتفوق على النماذج الأخرى من حيث دقة تحديد الموقع، مما يجعله حلاً واعدًا للتطبيقات الزمنية الفعلية ضمن البيئات الديناميكية لأنظمة المركبات الذكية.

Table des matières

Table des figures

Liste des tableaux

General Introduction

In recent years, roads and vehicles have become smarter. With the increasing number of cars on the road and the generation of data every second, the transportation sector is rapidly evolving towards intelligent and automated systems. Among these innovations, cooperative intelligent transportation systems (C-ITS) have garnered attention for their ability to improve safety, reduce traffic congestion, and support the future of autonomous driving. At the heart of these systems is vehicular ad hoc network technology (VANET), a system in which vehicles communicate directly with each other and with the road infrastructure, without the need for centralized control.

In VANETs, each vehicle must constantly know its current position. To achieve this, vehicles regularly broadcast periodic control messages to nearby nodes, including their current position. Accurate and reliable positioning is therefore crucial. To avoid collisions and efficiently plan routes through cooperative driving, the ability to determine the precise position of vehicles in real time is fundamental, enabling a wide range of safety and intelligent transportation applications.

However, traditional positioning systems such as the Global Positioning System (GPS) face significant limitations, particularly in urban canyons, tunnels, or in adverse weather conditions, where signal loss can occur. These limitations often lead to positioning errors or uncertainties. Some post-processing techniques, such as the Kalman filter or RDF (Relative Distance Filtering), have been proposed to refine GPS data. However, their effectiveness is limited by their reliance on GPS data, meaning they can only enhance the data, but not replace it.

In scenarios where GPS signals are intermittent or completely unavailable, researchers have proposed predictive models that use historical trajectory data to estimate a vehicle's future position. These predictive approaches aim to improve positioning accuracy and consistency, particularly in environments without GPS. In this project, we propose a deep learning approach to predict vehicle location in a VANET, without the need for physical sensors or constant GPS data.

This work contributes to ongoing efforts to improve vehicle positioning systems, particularly in environments degraded by GPS. The goal is not only to improve accuracy, but also to design more robust, affordable systems capable of operating with minimal hardware. In the long term, this could help improve road safety, support autonomous driving, and make intelligent transportation systems more accessible worldwide.

Chapitre 1

Cooperative Intelligent Transport Systems and VANETs

1.1 Introduction

The increasing number of vehicles on the road and the growing demand for safer, more efficient, and more environmentally friendly transportation have led to the development of cooperative intelligent transportation systems (C-ITS). These systems extend the concept of intelligent transportation systems(ITS) by enabling real-time cooperation and communication between vehicles, infrastructure, and other stakeholders in the transportation ecosystem. At the heart of C-ITS is the vehicular ad hoc network (VANET), a dynamic, decentralized wireless network formed between vehicles and roadside units (RUs) to support data exchange and decision-making on the move.[1] In this chapter, we highlight the core components, architecture, and communication models of C-ITS and VANETs. We also examine the security challenges related to the exchange of sensitive information.

1.2 Definition

The term ”C-ITS” describes a new generation of transportation systems that use wireless technologies to facilitate real-time information exchange, enabling cooperative behavior among road users.

VANETs form the core communication layer of C-ITS. Unlike traditional mobile networks, VANETs are highly dynamic and infrastructure-less and require rapid message dissemination among high-speed mobile nodes. The system must cope with frequent topology changes and ensure reliable and low-latency communication under challenging conditions.

1.3 Objectives

The main objectives of VANETs can be summarized as follows.

- Improve road safety : C-ITS enables vehicles to respond faster, which lowers the chance of accidents, by exchanging vital information like collision alerts, road condition warnings, and sudden braking
- Enhance traffic efficiency : by providing real-time traffic updates to vehicles, which then dynamically modify their routes to avoid traffic jams. The real flow of traffic can also be used to modify traffic signals.
- Reduce environmental impact : By reducing unnecessary idling and acceleration, applications such as traffic smoothing, platooning, and eco-driving support help reduce emissions and fuel consumption.
- Allow independent decision-making : By providing autonomous vehicles with access to environmental context beyond their sensors, C-ITS enhances safety and decision-making.
- Support emergency management : VANETs reduce reaction times and enhance coordination by warning surrounding vehicles and emergency services in the event of an accident or other hazard.

1.4 Types of Communication

VANETs supports several communication types[2] :

- **Vehicle-to-Vehicle (V2V) :** Direct exchange of position, speed, and alerts between nearby vehicles.
- **Vehicle-to-Infrastructure (V2I) :** Interaction with RSUs to receive traffic light status and road updates.
- **Vehicle-to-Network (V2N) :** Connection to cloud services and back-end systems.
- **Vehicle-to-Pedestrian (V2P) :** Enhances safety by alerting pedestrians through smartphones or wearable devices.

1.5 Architecture of VANETs

The VANET architecture consists of important physical and logical elements that work together to ensure that communication and coordination occur in real time.[1]

1.5.1 On-Board Units (OBUs)

Vehicles are equipped with on-board devices called OBUs. The hardware consists of processors, memory units, DSRC or LTE antennas, sensors, and GPS modules. Message exchange and security are managed by software, which also runs C-ITS applications and communication protocols.

OBUs exchange messages such as CAM and DENM with each other (V2V). In addition, via cellular relays or RSUs, they communicate with cloud services or central authorities (V2N) and RSUs (V2I) to obtain infrastructure data.

1.5.2 Road-Side Units (RSUs)

Static units, called RSUs, are positioned at intersections or along roadsides. They ensure data transmission between vehicles and infrastructure and are equipped with antennas, controllers, and power systems.

RSUs communicate with the central authority for policy updates and secure key distribution, with other RSUs for wide area coverage, and with OBUs for sending updates (traffic lights, road conditions). They serve as entry points between the transportation control center and mobile users.

1.5.3 Central Authority (CA)

The CA is a centralized backend that manages misbehavior detection, pseudonym issuance, and certificate management. It guarantees security and trust throughout the VANET. The CA facilitates secure communication by coordinating with RSUs and offering digital credentials, even though it is not directly involved in real-time exchanges.

Also, Figure 2 [1] illustrates the general architecture of a VANET.

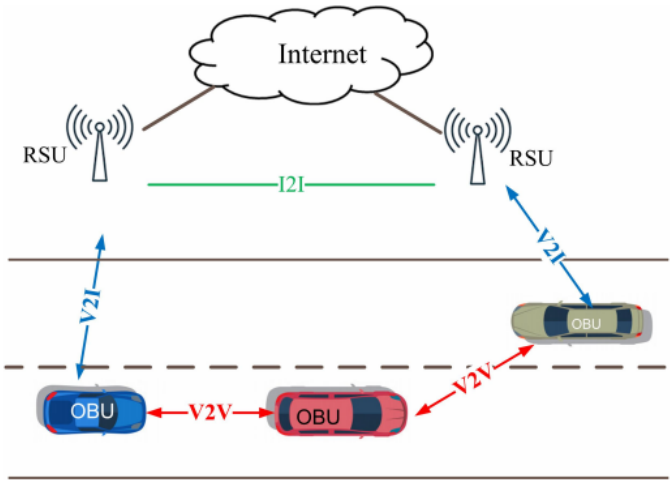

FIG. 1 : Basic Architecture of a VANET

1.6 WAVE Architecture (Wireless Access in Vehicular Environments)

WAVE is a communications architecture specifically designed for vehicular networks to facilitate secure, reliable, and fast V2V and V2I data transfer. It operates primarily on the physical and MAC layers of the IEEE 802.11p standard and is standardized by the IEEE 1609 family.[1]

1.6.1 Standards

The development of VANETs is the result of communication and resource systems. The Institute of Electrical and Electronics Engineers (IEEE) created a system architecture called WAVE to enable wireless access in vehicular environments. In October 1999, the U.S. Federal Communications Commission approved 75 MHz (5.85-5.925 GHz) in the 5.9 GHz band as the new DSRC spectrum for vehicular communications. The European Telecommunications Standards Institute (ETSI) has also allocated a radio frequency of 30 MHz (5875-5905 GHz) to 5.9 GHz. Similar groups exist in Japan, the IEEE 802.11 TGP working group began work on an amendment to the standard in 2004 to incorporate vehicular environments. The IEEE 802.11 group focuses on the physical layer standard (DSRC PHY) and the Media Access Control (MAC) sublayer for unrestricted access in the automotive environment. The creation of specifications covering additional layers of the protocol suite has been assigned to another IEEE team (1609 working group). The main purpose of these protocols is to present the communication architecture, frequency sharing, application management, security algorithms, and messaging. The communication protocol using these standards is shown in figure 2 in relation to the OSI model's nine couches.[1]

1.6.2 Communication Protocols

Figure 2 illustrates the WAVE architecture. These protocol stacks are managed by this architecture. The following elements are present in WAVE,

according to the terminology of the OSI model :[1]

- The WAVE architecture is based on the IEEE 802.11 standard, which describes the physical layer and part of the data layer link of the protocol stack. IEEE 802.11p is a modification of the 802.11 standard. Because it specifies not only the data transmission part of the protocols, but also the management functions related to the corresponding layer, this new standard takes the vehicular environment into account. Two entities are responsible for managing the physical layer : PLME (Physical Layer Management Entity) and MLME (MAC Layer Management Entity).

- IEEE 1609.4: The level of both layers is enhanced by a new sub-layer. It helps control multi-channel operations. WAVE units can divide their time between channel control and channel services.

- IEEE 802.2: Logical Link Control (LLC) adheres to this obsolete standard.

- IEEE 1609.3: At the transport and network levels, WAVE manages the traditional Internet Protocol version 6 (IPv6) stack and a new stack adapted to the vehicle context. This new stack includes a WAVE protocol for short messages, also called the WAVE ShortMessage Protocol (WSMP). Traditional and less demanding exchanges are managed by TCP/UDP, while priority and urgent communications are managed by WSMP. This IEEE 1609.3 standard also includes a set of management functions called WAVE Management Entity (WME), which provide network configuration services. Consider an application that requires real-time execution and sends urgent alert messages. WSMP allows this application to send short messages and directly control certain radio resource parameters to ensure that all recipients receive the message on time. However, standard Internet applications are necessary to attract investors to a particular technology. For typical Internet applications, WSMP is insufficient, which is why IPv6 is included.

- IEEE 1609.1: This resource management tool is not part of the OSI

model. It describes the application that allows a vehicle to interact with limited computing resources and complex processes running outside the vehicle.

- IEEE 1609.2: This offers WAVE security services and does not integrate with the OSI model. This standard covers secure message formats and potential features such as encryption and authentication.

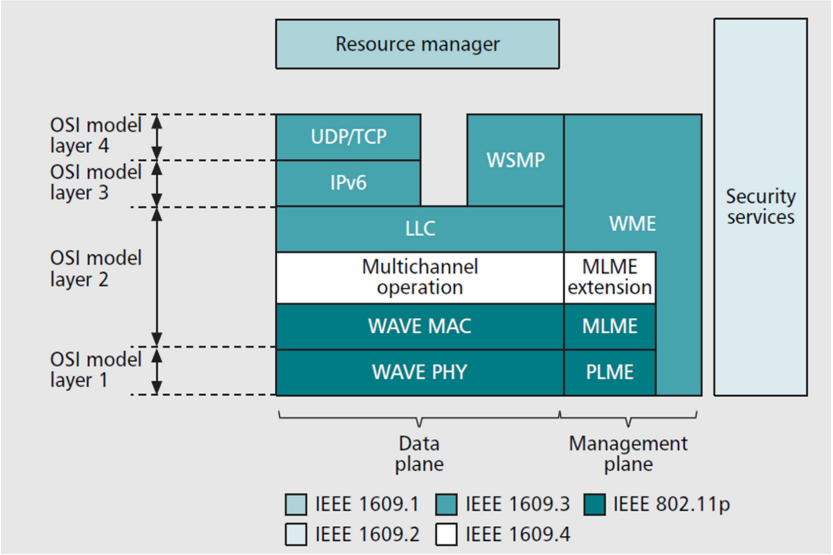

FIG. 2 : Pile de protocoles WAVE [1]

1.7 Comparison between Trusted Authority, Central Authentication, and PKI

Protecting sensitive data, confirming identities, and stopping malicious activity in VANETs all rely on secure and reliable communication. To achieve this, several mechanisms have been introduced, including Public Key Infrastructure (PKI), Central Authentication (CA), and Trusted Authority (TA). Each C-ITS mechanism addresses a different aspect of network security, although their functions sometimes overlap.

1.7.1 Trusted Authority (TA)

A centralized and trusted organization, called the TA, is responsible for overseeing the overall security architecture of the VANET. Its main responsibilities include identity management, certificate issuance, and cryptographic key distribution.

These functions include :

- Issuance and revocation of digital certificates ;
- Certificate Revocation Lists (CRLs) ; Defining and enforcing security policy ;

By controlling the identities and authorizations of participating vehicles and road units, the VANET TA often acts as a representative of a government or institutional entity (such as a security agency or the Ministry of Transportation), thus ensuring the legitimacy of these entities.

1.7.2 Central Authentication (CA)

The real-time process of confirming an entity's identity when it tries to join a network or access a particular service is known as central authentication. In contrast to the TA, which handles identities and certificates over an extended period of time, the CA usually participates in dynamic session validation.

Functions :

- Real-time identity checks (such as challenge-response protocols).
- Secure login or access control during communication.

To prevent unwanted access, the CA participates in the authentication of RSUs or vehicles (OBUs) in the context of VANETs. However, this strategy can lead to latency issues, especially in large-scale or decentralized networks.

1.7.3 Public Key Infrastructure (PKI)

PKI is a well-known framework for managing digital certificates and cryptographic keys. In many systems, including VANETs,it serves as the basis for secure communication.

Functions :

- Generation and management of public/private key pairs
- Issuance of digital certificates (e.g., in X.509 format)
- Digital signature verification
- Certificate renewal and revocation

PKI ensures authenticity, integrity, and, sometimes, anonymity in VANETs by signing security messages such as Cooperative Awareness Messages (CAMs) and Decentralized Environmental Notification Messages (DENMs). To protect confidentiality, the certificate authority usually issues a batch of temporary pseudonymous certificates to vehicles.[1]

1.7.4 Summary Comparison

TAB. I : Comparison of TA, CA, and PKI in VANETs

Aspect	**Trusted Authority (TA)**	**Central Authentication (CA)**	**Public Key Infrastructure (PKI)**
Role	Identity and certificate management	Real-time authentication	Secure key and certificate framework
Focus	Global policy and identity control	Session-based access validation	Data encryption, signature, and verification
In VANETs	Vehicle registration and pseudonym issuance	Login/authentication for OBUs and RSUs	Signing CAM/-DENM, certificate revocation
Pros	Strong control and trust enforcement	Effective session validation	Secure, scalable, and privacy-preserving
Cons	May introduce central point of failure	Can cause delay in decentralized systems	Certificate management overhead

1.8 Applications

- Safety Applications :Accidents are decreased by services like Intersection Collision Warning (ICW), Electronic Emergency Brake Light (EEBL), and Forward Collision Warning (FCW). High reliability and low latency are necessary for these applications.
- Traffic Management : Applications like dynamic routing, congestion prediction, and adaptive traffic signals improve traffic flow and reduce delays[1].
- Environmental Efficiency :Green-wave support, platooning, and eco-driving advice lower fuel consumption and emissions.
- Infotainment : VANETs improve passenger comfort by supporting location-based services, media streaming, and internet access.

1.9 Challenges

Despite their promising potential, the deployment of VANETs faces several critical challenges that hinder large-scale adoption and practical implementation. These challenges are both technical and organizational, requiring continued research and cross-sector coordination to overcome them effectively.

- **High Mobility :** Rapid and unpredictable changes in network topology result from vehicles' high speeds and frequent direction changes. Maintaining steady and uninterrupted communication channels between nodes is challenging due to the sporadic connectivity and frequent link failures caused by this dynamic environment.
- **Scalability :** Hundreds or even thousands of vehicles may be communicating at the same time in urban areas. As a result, there are more message collisions and congestion in the wireless communication channels. When creating protocols that can effectively handle massive data volumes without sacrificing performance, scalability becomes a key concern.
- **Security and Privacy :** VANETs are exposed to various security threats such as message spoofing, denial-of-service (DoS) attacks, and false data injection. Moreover, the exchange of location and identity data raises significant privacy concerns. Without proper safeguards, malicious actors could exploit the system to track vehicles or manipulate traffic behavior.
- **Interoperability :** Compatibility problems arise from the absence of standardized system architectures and communication protocols among various manufacturers and geographical areas. A truly C-ITS requires smooth communication between heterogeneous vehicles and infrastructure components, which is not possible due to this fragmentation.

Researchers and engineers are actively developing cutting-edge solutions to these problems, such as lightweight and safe authentication methods, data

sharing strategies that protect privacy (like pseudonym schemes), and flexible communication protocols that can adapt to changing conditions. Moreover, global initiatives to standardize protocols seek to enhance interoperability and guarantee uniform system performance across borders and platforms [2].

1.9.1 Security Requirements in C-ITS

In C-ITS networks, where sensitive data like vehicle location, speed, and identity are permanently changed, security is a fundamental requirement. Without the proper authentication and integrity mechanisms, these data may be intercepted, fabricated, or manipulated, leading to critical failures like false alarms, incorrect predictions, or even accidents.

In our work, we view data security and reliability as critical pillars for making IA-based decisions. Therefore, we start with the idea that the data used for learning and prediction travels through secure transmission channels that are guarded by protocols like PKI, TA, or centralized authentication frameworks.

1.10 Sensitive Information Exchange in VANETs

Using messages like CAM and DENM, most communications in VANET environments are conducted via wireless broadcasts between vehicles and infrastructure. Among the vital data these messages convey are vehicle position, speed, heading, acceleration, and contextual alerts on road hazards or traffic conditions. These messages are naturally susceptible to many risks including interception, manipulation, or spoofing since they are sent over open wireless channels. The sensitivity of information shared in VANETs is therefore seen from several important angles :

- **Accuracy and Integrity :** The effectiveness of safety applications is directly affected by the accuracy of the data sent. Errors in speed or

location data can lead to poor decisions and dangerous consequences.

- **Reliability :** Vehicles must be able to determine whether a received message comes from a reliable source. Without strong trust models, fake messages could spread and disrupt traffic flow or even cause accidents.
- **Confidentiality and Privacy :** Messages often contain personally identifiable information, such as vehicle identification or location history, which poses significant privacy concerns. Encryption and pseudonymization systems must be used to prevent unauthorized vehicle tracking or profiling.

While these aspects are fundamental to the secure operation of VANETs, this project focuses specifically on using shared vehicle data for predictive purposes. We do not directly address security, encryption, or message content verification ; rather, our work involves analyzing trajectory data to predict vehicle movements and support smart mobility decisions.

1.11 Conclusion

These fundamental elements discussed in this chapter highlight the limitations of current systems, particularly in terms of real-time decision-making and data reliability.

To address these challenges, we propose leveraging artificial intelligence (AI) to improve vehicle trajectory prediction, ensure precise localization, and optimize decision-making. The following chapter presents the fundamentals of AI and explains how to integrate it into C-ITS to overcome the limitations discussed here.

Chapitre 2

Artificial Intelligence and its role in C-ITS

2.1 Introduction

AI has developed quickly in recent years as a game changing technology that can solve difficult problems in a variety of fields. An increasing amount of real-time data and a growing need for smarter, safer, and more efficient mobility solutions have made the transportation sector a prime candidate for AI integration. AI is radically changing contemporary transportation systems, from enabling autonomous vehicles to optimizing routing strategies.[3]

We focus on Machine Learning (ML) and Deep Learning (DL), which are widely used in applications such as trajectory prediction and anomaly detection. This chapter sets the foundation for understanding how AI techniques can be applied to improve the performance and reliability of VANET-based systems.

2.2 Definition

AI is a scientific discipline focused on creating machines capable of performing tasks that require some form of human intelligence. Marvin Minsky

defines it as "the science of enabling machines to perform tasks that would require intelligence if performed by humans." It relies on mathematical formulas and algorithms that use probability and statistics to mimic cognitive functions such as learning and decision making.

2.3 History

The origins of AI date back to 1956, initiated by pioneers such as John McCarthy at Dartmouth College, with the aim of replicating human intelligence. Significant milestones have marked its evolution, notably the Turing Test in 1950, which challenged machines to appear indistinguishable from humans, and the development of Eliza in 1965, a precursor to modern chatbots, illustrating the vision and aspiration of AI since its early days.Today, AI is reshaping the world in unprecedented ways, transforming industries at an extraordinary pace. According to a PwC (PricewaterhouseCoopers) report, AI could contribute up to 15.7 trillion USD to the global economy by 2030. Modern AI is characterized by its ability to learn from large amounts of data, which allows major advances in numerous sectors, including law, healthcare, finance, and the automotive industry.

2.4 Domains and Applications of AI

AI is a broad term that includes a number of important fields, all of which aim to mimic particular facets of human intelligence and judgment. These areas have produced useful systems that are used in industries such as robotics, healthcare, finance, and transportation. An outline of the main areas of AI and the technologies associated with them is provided below.

2.4.1 Principal Domains of AI

- Computer Vision : is concerned with making it possible for machines to comprehend and evaluate visual data from the outside world, like

pictures or videos. Traffic monitoring, facial recognition, and object detection are just a few examples of applications.

- Natural Language Processing (NLP) :Systems can comprehend, interpret, and produce human language thanks to natural language processing, or NLP. Information extraction, text classification, machine translation, and dialogue systems (such as chatbots) are among the tasks that fall under this category.

- Expert Systems : Using a knowledge base and inference rules, these systems mimic the logic of human experts. They are frequently employed in recommendation systems, decision support, and diagnostics

- Speech Recognition :Enables voice-based system interaction by translating spoken language into text. Voice commands, accessibility technologies, and virtual assistants make great use of it.

- Planning and Decision Making : this field involves developing algorithms that can generate strategies or action sequences to accomplish particular objectives, frequently in dynamic or uncertain environments.

- Robotics and Intelligent Agents : In order to allow machines (robots or autonomous agents) to interact with the physical world, make decisions, and carry out tasks, robotics and intelligent agents (AI) are combined with sensors and actuators.

- Predictive Analytics : Using statistical and machine learning techniques, predictive analytics makes predictions about future occurrences or actions by analyzing past data. [4]

2.5 AI in Transportation Systems

AI is now a revolutionary force driving innovation in C-ITS. By automating complex tasks and streamlining transportation operations, AI technologies are making mobility smarter, safer, and more sustainable. Traffic forecasting and control are one of the most important uses of AI in C-ITS.

Machine learning models analyze historical and current data to predict traffic congestion, dynamically change traffic signals, and improve overall traffic flow. In addition, machine learning-based computer vision is used for incident detection, lane tracking, and vehicle and pedestrian detection, all of which improve road safety. AI is essential for perception, navigation, and decision-making in autonomous vehicles. It combines information from cameras, LiDAR, GPS, and on-board sensors to communicate with the environment. In addition,predictive maintenance programs use AI to track the condition of the infrastructure and vehicles to anticipate breakdowns.[3]

2.6 Branches of AI

AI is mainly divided into two branches : ML and DL. ML enables machines to learn from data and improve their performance. DL is inspired by the functioning of the human brain through artificial neural networks, allowing the processing and interpretation of massive amounts of information.[4]

The diagram below provides a visual representation of the hierarchy and functioning of AI, highlighting the relation between ML and DL.

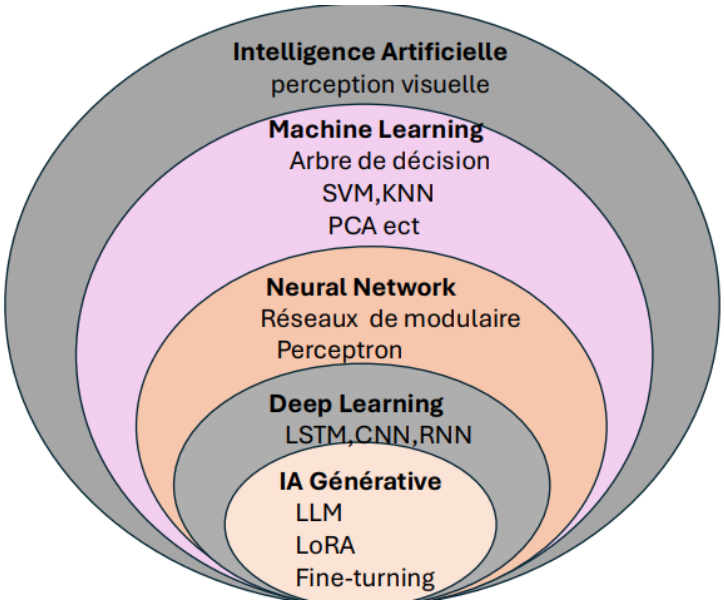

FIG. 3 : Overview of AI, ML, and DL

2.7 Machine Learning

ML is a branch of science, and more specifically, a branch of AI. It means letting algorithms find ”patterns,” or things that happen over and over again, in data sets. This information can be numbers, words, pictures, statistics, and so on.Data for ML can come from anything that can be stored on a computer. Algorithms learn and get better at doing a certain job by finding patterns in this data.In short, ML algorithms learn on their own how to do a job or make predictions based on data and get better at it over time. The algorithm will be able to find patterns in new data after it has been trained.[5]

2.7.1 How does ML work ?

There are four primary steps involved in creating an ML model. This process is usually managed and supervised by a data scientist.Choosing and preparing a training dataset is the first step. The ML model will be fed this data in order to learn how to solve the problem for which it was created.To tell the model which features it should look for, the data can be labeled. Additionally, it may be unlabeled, in which case the model will need to recognize and extract recurrent features independently.The data needs to be meticulously cleaned, arranged, and prepared in both situations. Otherwise, there is a chance that the ML model will be biased during training. It will have a direct effect on the outcomes of its future forecasts. Choosing an algorithm to use on the training dataset is the second step. The kind of problem to be solved and the kind and volume of training data determine the kind of algorithm to use.Training the algorithm is the third step. The process is iterative. The algorithm is applied to variables, and the output is contrasted with what it ought to have generated. The accuracy of the outcome can then be improved by adjusting the weights and bias.After that, the variables are run once more until the algorithm typically yields the right result. The ML model is the trained algorithm. Using and refining the model is the fourth and last step. Depending on the issue being resolved, the model is applied to fresh data. For instance, emails would be subjected to an ML model intended to identify spam.A robot vacuum cleaner’s machine learning model, on the

other hand, absorbs information from interactions with the outside world, like moving furniture or putting new items in the room. Accuracy and efficiency can also improve with time.[5]

2.7.2 Supervised and unsupervised learning

There are two main types of ML : supervised and unsupervised.In supervised learning, the algorithm is guided by prior knowledge of what the model's output values should be. Consequently, the model adjusts its parameters to reduce the gap between the results obtained and the expected results. The margin of error is thus reduced as the model is trained, allowing it to be applied to new cases.In contrast, unsupervised learning does not use labeled data. It is therefore impossible for the algorithm to reliably calculate a success score. Its objective is therefore to infer the clusters present in our data. Let's take the example of a dataset of flowers ; we are trying to group them into classes. Here, we don't know the plant species, but we want to try to group them ; for example, if the flower shapes are similar, then they are related to the same corresponding plant. There are two main areas of models in unsupervised learning to find clusters :Partitioning methods : k-means algorithms.And hierarchical clustering methods : hierarchical ascending clustering (HAC).[5]

Since machine learning is the foundation of many predictive applications and is a basic area of AI, it was introduced in this chapter. Predicting a vehicle's future position without GPS data is the main challenge in our work. The theoretical and practical underpinnings for creating models that can generalize to unknown future movements by learning from historical trajectory data are provided by machine learning. However, traditional machine learning algorithms frequently need to manually engineer features and have trouble identifying temporal dependencies in sequential data.This restriction prompted us to investigate more sophisticated deep learning-based models that are capable of automatically extracting temporal and spatial features from unprocessed sequences. These DL architectures are introduced in the following section along with an explanation of how they improve prediction

performance in the context of C-ITS.

2.8 Deep Learning

DL is one of the core technologies of ML. DL refers to algorithms capable of mimicking the actions of the human brain using artificial neural networks. The networks are composed of dozens or even hundreds of ”layers” of neurons, each receiving and interpreting information from the previous layer.

2.9 Deep Learning and Neural Networks

Neural networks are the core of modern DL systems. They draw inspiration from the composition and operation of the human brain, which is made up of interconnected neurons that exchange information and pick up patterns from stimuli. Neural networks in AI are made up of layers of artificial neurons, or nodes, that analyze input data to produce predictions and extract valuable features.

2.9.1 Artificial Neural Networks (ANNs)

The human brain serves as the inspiration for artificial neural networks, or ANNs. Each neuron in their many layers of interconnected neurons applies an activation function to the weighted sum of its inputs. These networks are widely used to solve complex problems, identify patterns, and make predictions [6]. Three primary layers make up a typical ANN :

- The input layer, which receives the raw input data.
- The hidden layers, responsible for extracting features and learning patterns using weights and activation functions.
- The output layer, which produces the needed prediction or classification result.

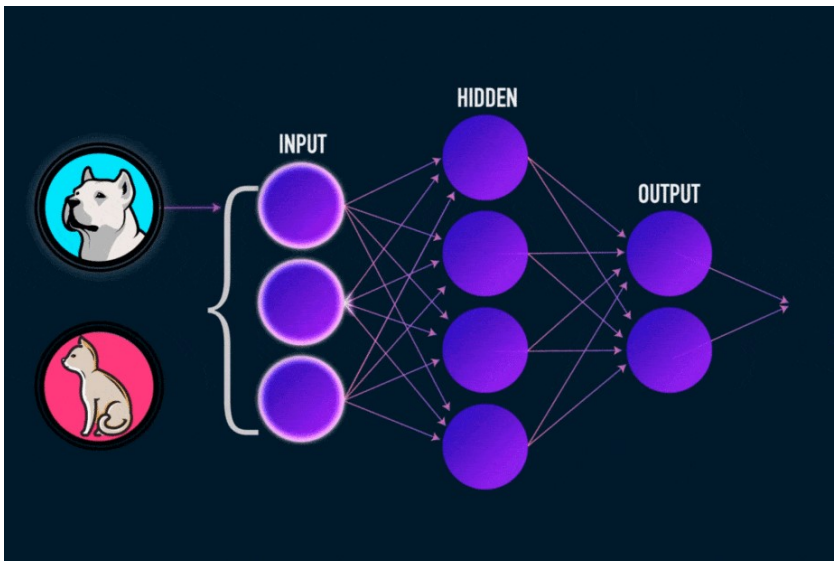

FIG. 4 : Illustration of a neural network architecture used in DL.

The mathematical formulation of a neuron is given by :

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.1}$$

where : w_i represents the weights, x_i denotes the inputs, b is the bias, and $f(\cdot)$ is the activation function. The activation functions most commonly used are ReLU (Rectified Linear Unit) and the Sigmoid function.

Later, recurrent neural networks (RNNs) were created to handle sequential data by keeping a copy of earlier inputs. Recurrent temporal dependencies in the form of hidden states are used by RNNs to handle sequential data. RNNs, in contrast to conventional feed-forward networks, feature loops that enable them to transfer data from earlier inputs to subsequent inputs. Regular RNNs have a drawback, though, in that they struggle to learn long-term dependencies in sequences because they are unable to adequately handle the vanishing gradient issue.

2.9.2 Convolutional Neural Networks (CNNs)

Specialized deep learning models called Convolutional Neural Networks (CNNs) are made to process spatial data. They are excellent at extracting hierarchical features from structured inputs. CNNs apply a set of learnable filters to convolutional layers in order to recognize and extract spatial patterns. These are followed by pooling layers that employ downsampling to lower the spatial dimensions of the data. By doing this, overfitting is prevented, the computational load is reduced, and the most crucial features are preserved. Finally, high-level features are extracted by the convolutional and pooling layers, and fully connected layers transform them into a dense vector representation. as illustrated in Figure 5

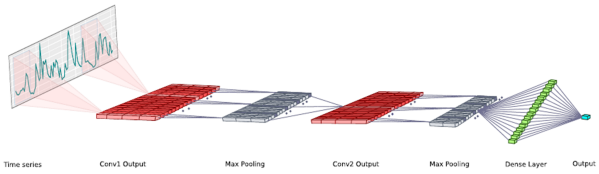

FIG. 5 : Overview of AI, ML, and DL[7]

This representation is then used to make predictions by mapping the features to the output.

In the context of trajectory prediction, CNNs are used to detect spatial dependencies between position sequences. They apply convolutional filters to extract localized features across time windows, allowing the model to detect movement patterns.[8]

The fundamental operation in CNNs is the convolution, defined as :

$$(I * K)(x, y) = \sum_{m} \sum_{n} I(x + m, y + n)K(m, n) \tag{2.2}$$

where I represents the input, K denotes the kernel, and (x, y) are spatial coordinates. This process enables CNNs to capture spatial relationships in data, thereby improving prediction accuracy.

2.9.3 Long Short-Term Memories(LSTMs)

A subclass of RNNs called long short-term Memory (LSTM) networks was created expressly to represent temporal sequences and preserve long-term dependencies. [8] [9](Figure 6)

An LSTM unit consists of three main gates :

- The forget gate, which decides what information to discard.
- The input gate, which updates the cell state with new information.
- The output gate, which determines the final output.

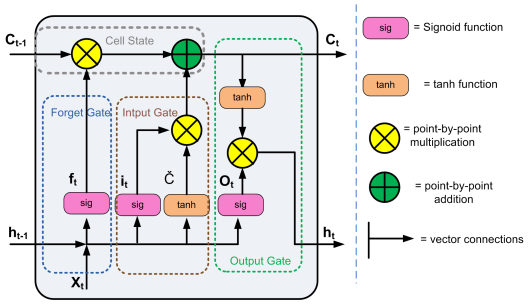

FIG. 6 : Structure of an LSTM cell[7]

To illustrate how LSTMs effectively capture long-term dependencies, making them convenient for sequential location prediction, we present the following equations :

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.3}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.4}$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{2.5}$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.6}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.7}$$
$$h_t = o_t \odot \tanh(C_t) \tag{2.8}$$

Table II provides a brief overview of the main symbols and notations used in the previous equations.

TAB. II : SUMMARY OF SYMBOLS AND NOTATIONS

Symbols and Notations	Description
$f^{(t)}, i^{(t)}, o^{(t)}$	The forget, input, and output gate at time t, respectively
σ	Logistic sigmoid function
W_f, W_i, W_o	Weight matrices of the forget, input and output gates, respectively
b_f, b_i, b_o, b_C	The biases of the forget, input, output gates, and the cell state, respectively
$h^{(t-1)}, h^{(t)}$	The hidden state of the previous and timestamp, respectively
$x^{(t)}$	The current input
$C^{(t)}$	The current cell state
$\widetilde{C}^{(t)}$	The new memory cell state candidate
$\odot$	Element-wise (Hadamard) product

LSTMs are perfect for tasks involving time-dependent patterns like speech recognition, anomaly detection, and vehicle trajectory prediction because of their gated structure, which balances memory retention and forgetting.They have been extensively used in the context of C-ITS to forecast vehicle trajectories in situations where GPS signals are erratic or unavailable. They are particularly effective in capturing non-linear movement patterns and learning from historical location data to forecast future positions accurately.

Furthermore, as demonstrated by hybrid models used in VANET-based location prediction, LSTM models can learn both spatial and temporal dependencies when combined with other architectures (such as CNNs).

2.9.4 Transformers

Sequential modeling has been transformed by attention-based architectures called Transformers. Unlike RNNs or LSTMs, Transformers do not require sequential processing of input data, allowing for parallelization and

faster training. This makes them highly efficient for applications involving temporal context, such as autonomous driving and traffic flow prediction. The key to their success is the self-attention mechanism, which enables the model to weigh the importance of different elements in the input sequence when generating each output token. This mechanism captures long-range dependencies and relationships in the data more effectively than traditional architectures.

Introduced by Vaswani et al. in "Attention is All You Need" (2017), the Transformer architecture consists of an encoder-decoder structure. The encoder processes the entire input sequence and generates a set of context-rich representations. These representations are then passed to the decoder, which generates the output sequence one token at a time—each step using self-attention and encoder-decoder attention to decide what to generate next.[10]

The figure below provides a simplified example of a Transformer translating a French sentence into English. It shows how each input word is mapped and attended to when generating each output word, illustrating the internal transformation process from input to output using attention weights.

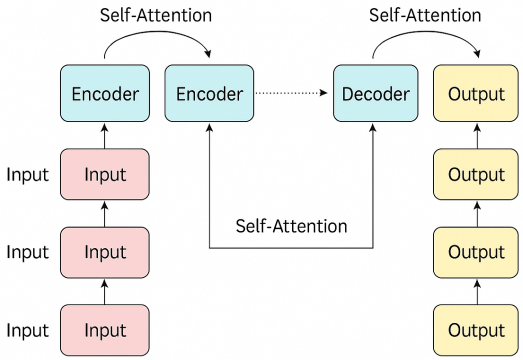

FIG. 7 : Transformer model illustrating how input sequences are transformed into output sequences using self-attention mechanisms.

2.9.5 Autoencoders :

Autoencoders are neural network-based models primarily used for unsupervised representation learning and reduction of dimensionality. They work especially well for feature compression and anomaly detection in time-series data in the context of C-ITS. By encoding the input into a lower-dimensional latent space and decoding it back with the least amount of reconstruction error, traditional autoencoders seek to reconstruct the input.

In recent studies, more sophisticated autoencoder-based models designed for temporal anomaly detection have been introduced. For example, Singh (2017) showed how LSTM-based autoencoders could effectively handle sequential dependencies by identifying irregular patterns in temporal vehicle data [11]. More recent methods, such as AER (Autoencoder with Regression), combine an autoencoder and a regression head to optimize reconstruction and prediction goals simultaneously, improving anomaly detection [12].Similarly, the Multivariate Time Series Anomaly Detection (MTSAD) model incorporates ConvLSTM architectures to capture spatial and temporal dependencies in multivariate datasets, achieving state-of-the-art results in industrial settings [13].

These advances highlight the increasing relevance of autoencoders in vehicle trajectory modeling and abnormal behavior detection in VANETs, especially when dealing with high-dimensional, time-dependent datasets.

2.9.6 Gated Recurrent Units

A simplified version of LSTMs called GRUs was developed to lower computational complexity without sacrificing sequence learning performance. Unlike LSTMs, GRUs do not keep a separate memory cell and combine the input and forget gates into a single update gate. GRUs are easier to deploy and train thanks to their simplified architecture, particularly on low-resource devices.

Recent research indicates that GRUs are computationally more efficient than LSTMs while providing results that are comparable. In real-time appli-

cations, their use has grown, especially when latency is an issue.[11]

2.10 Deep Learning Techniques and Algorithms

Effective training of deep learning models relies on various techniques and algorithms :

2.10.1 Optimization Algorithms :

During training, algorithms like Adam, and Stochastic Gradient Descent (SGD) are used to minimize the loss function. Adam specifically provides an optimization algorithm that combines the advantages of two additional extensions of SGD, Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp), to handle sparse gradients on noisy problems.

2.10.2 Regularization Techniques :

Techniques such as L1 and L2 regularization apply a penalty term to the loss function to prevent overfitting, depending on the size of the model's weights. By ignoring randomly selected neurons during training, dropout is another regularization technique that prevents the network from becoming unduly reliant on any one neuron.

2.10.3 Batch Normalization :

With this approach, the inputs for each layer are set to have a standard deviation of one and a mean of zero. Batch normalization reduces the need for additional regularization techniques by speeding up training and offering some regularization.

2.11 Large Language Models (LLMs)

Advanced deep learning models called Large Language Models (LLMs) are made to comprehend, produce, and work with human language. To capture intricate patterns and contextual meanings in language, they employ transformer-based architectures and are trained on enormous volumes of text data. Text generation, translation, summarization, question answering, and other natural language processing tasks are among the many tasks that LLMs are capable of.

2.11.1 Challenges and Limitations in LLMs and AI Systems

LLMs have demonstrated remarkable capabilities but also present several challenges such as :

- LLMs without memory :Every interaction with an LLM is autonomous and stateless, just like a REST API call. The incapacity of the model to remember past exchanges affects the continuity of long-term interactions. The need for fully self-contained inputs leads to repetitive or fragmented interactions.
- Synchronous invocations : One by one, LLMs process and respond to each input. This synchronous process limits concurrent query processing and real-time communication. The inability to parallelize processing could be a drawback in scenarios that require quick responses.
- Hallucinate : LLMs can generate information that is factually incorrect or illogical. They use large datasets to find patterns rather than ensuring factual accuracy. This could lead to the presentation of false information with assurance, creating the appearance of knowledge.
- Internet connection :LLMs can only use the data they were trained on and cannot access real-time information from the internet. They cannot provide the most recent news updates or access the most recent

research. Because of this, they are less helpful for tasks requiring up-to-date information.

- LLMs are bad at Mathematics :they struggle with complicated problem-solving and precise computations, which makes them poor mathematicians. They are capable of basic arithmetic but lack the structured reasoning required for advanced math. This limitation affects their reliability in performing accurate multi-step calculations.

- LLMs produce non deterministic output :Identical inputs can produce different outputs due to their probabilistic nature. This variability makes it difficult to obtain consistent results. Particularly affected are applications that require consistent response formatting, such as report generation.[14]

- Limitations of traditional prediction methods : Classical approaches such as the Kalman filter often struggle to handle nonlinearities and long-term dependencies in complex temporal data. To overcome these limitations, advanced deep learning models like LSTM networks have been introduced. LSTMs are capable of learning temporal dynamics from sequential data, making them well-suited for tasks such as anomaly detection and trajectory prediction.[11]

2.12 Local and Regional Large Language Models (LLMs)

In recent years, many countries have begun developing large language models (LLMs) that are specific to their own linguistic and cultural contexts, either locally or regionally. These efforts aim to enhance the understanding and generation of human language by AI systems that consider regional dialects, expressions, and usage. Algeria has started down this path with programs like DziriBERT and DarjaBERT, which are specifically designed for Algerian Arabic (Darja).

2.12.1 DziriBERT

A pre-trained language model designed to comprehend Algerian dialectal Arabic is called DziriBERT. It was trained using text data in Algerian Arabic gathered from social media sites such as Facebook and Twitter, and it is based on the BERT (Bidirectional Encoder Representations from Transformers) architecture. A team of researchers, comprising members of the Algerian research community and Meta AI contributors, created the model. Text classification, sentiment analysis, named entity recognition, and dialect identification are just a few of the NLP tasks that DziriBERT can handle. It is a domain-specific model that shows how transformer-based architectures can be modified to fit low-resource dialects like Algerian Darja, despite not being a large-scale LLM like GPT-3.[15]

2.12.2 DarjaBERT

Another Algerian project called DarjaBERT aims to create transformer models that are trained on spoken Arabic in Algeria. This model focuses on informal dialectal texts that are frequently encountered in daily conversations and social media. It has a significant part in : Chatbot creation,Understanding informal language, Transliteration assignments. Although there isn't much information available, DarjaBERT is presently being researched and developed as part of regional AI research projects at Algerian universities.[16]

2.12.3 Regional Arabic LLMs

Arabic-specific LLMs are being developed in the larger Arab world, including :

- Noor (Saudi Arabia) : A potent Arabic LLM with extensive training in dialectal, modern standard, and classical Arabic.
- Jais (UAE) : The UAE's Technology Innovation Institute created this bilingual LLM in Arabic and English.

- The MENA region makes extensive use of the well-known Arabic natural language processing models CAMeLBERT and AraBERT.

These models give governments, businesses, and educational institutions the means to implement AI in their own languages while also bridging the language gap in AI applications.[17]

2.12.4 To Algerian LLMs

Despite not being full-scale LLMs just yet, DziriBERT and DarjaBERT set the stage for future, more sophisticated models. Algeria could create its own multilingual and culturally appropriate LLM with assistance from government agencies, research facilities, and business. This would help with healthcare, education, and the creation of content in French, Arabic, and Tamazight.

2.13 Conclusion

This chapter illustrated the potential of AI and deep learning to transform transportation systems. We examined how these techniques use data to learn and make decisions in real time, ranging from basic neural networks to more intricate models like Transformers and Autoencoders. We also discussed optimization and regularization strategies that improve model performance.

These AI tools are not just theoretical, they lay the groundwork for real-world applications. One such application is vehicle location prediction, which is an essential part of modern traffic systems.

Moving from theory to practice, we apply AI to VANET-based vehicle localization systems in the following chapter. We'll see how vehicles communicate with each other and their surroundings, and how deep learning models can predict future locations with high accuracy.

Chapitre 3

Vehicle Localization Prediction

3.1 Introduction

The accurate location of the vehicle is a fundamental requirement for the operation of C-ITS. In VANETs, knowing the precise position of each vehicle is crucial for safety applications, traffic optimization, and autonomous driving. However, traditional positioning systems, such as GPS, are often limited by signal loss, urban obstacles, or environmental conditions. To overcome these challenges, prediction techniques have emerged as a complementary solution to improve localization accuracy and reliability.

This chapter introduces localization concepts in VANETs, presents the main challenges of traditional methods, and highlights the importance of predictive models. We then explore conventional and deep learning-based prediction techniques and review recent work on vehicle localization using AI.

3.2 Concepts of Localization in VANETs

Vehicle localization refers to the process of determining the position of a vehicle in a geographical space. In VANETs,this position information is essential for many applications, including cooperative awareness, traffic op-

timization, emergency vehicle routing, and accident prevention. Localization systems provide position coordinates (usually latitude and longitude) that are continuously updated as the vehicle moves.

3.2.1 Overview of GPS Positioning

Vehicle localization in VANETs now is heavily dependent on GPS, a satellite-based navigation system. A GPS receiver solves a system of equations involving the receiver's location and clock offset to determine its position by calculating pseudoranges, or the estimated distances, from at least four satellites. The basic idea is based on time-of-arrival measurements with coded signals, like the Coarse/Acquisition (C/A) code, which is transmitted at 1575.42 MHz in the L1 band. The GPS signal includes three main components : a pseudo-random code for satellite identification, ephemeris data for satellite positioning, and almanac data for system health and orbits. A GPS receiver must first receive the satellite signal, then use delay-locked loops (DLL) and phase-locked loops (PLL) to track the signal code and carrier phase before decoding the navigation message to extract location data.[18] Even though GPS offers constant worldwide coverage, several factors, such as hardware limitations, multipath propagation, satellite geometry, and ionospheric delays, affect its accuracy. Civilian GPS receivers can reach 3-10 m accuracy in optimal circumstances. However, this precision can drastically deteriorate or even vanish in obstructed environments such as tunnels or urban canyons.[19] The GPS acquisition process is also susceptible to noise and signal strength, as covered in [20]. Although they still depend on the signal availability, faster acquisition architectures, such as those based on FFT or systolic correlators, have been proposed to improve performance. Because of this, GPS is not enough for VANETs that need real-time, highly accurate, and highly available localization systems.

3.2.2 Principal components of localization in VANETs

GPS is not the only method used for localization in VANETs. Rather, it is frequently improved or supported by other elements and tactics that work in concert to ensure consistent, precise, and reliable positioning. The primary components of a VANET localization framework are as follows :

- OBUs : Each vehicle has an OBU with a GPS receiver, a communication module (such as the DSRC or C-V2X), and perhaps an Inertial Measurement Unit (IMU). To determine position, the OBU analyzes the the GNSS signals and communicates this information to otvehicles or nodes of the infrastructure.infrastructure.
- RSUs : RSUs are stationary nodes located along the road infrastructure that can help localize by acting as known anchors for range-based localization methods, providing map references, or providing correction data (e.g., in differential GPS).
- HELLO Messages :Vehicles send out HELLO beacons on a regular basis, which allow other cars to create a situational map by providing localization data (position, velocity, and direction). Cooperative localization strategies are based on these messages.
- Inertial Sensors :In the event that GPS signals are not available, displacement can be estimated by integrating the linear acceleration and angular velocity data provided by IMUs. Dead reckoning offers short-term robustness in locations with GPS issues, despite drift over time.
- Sensor Fusion Modules :VANETs frequently use multi-sensor data fusion, combining GPS, IMU, wheel encoders, and map data, to guarantee high accuracy and availability. Fusion techniques vary from basic KFs to more sophisticated methods like hybrid Bayesian approaches or particle filters.[19]

Finally , how well these elements work together to offset the shortcomings of individual systems determines how effective localization is in VANETs.

We look more closely at the intrinsic drawbacks of conventional GPS-based methods in the context of vehicles in the following section.

3.2.3 Objectives of VANET Localization Systems

For a VANET localization system to be effective in practice, it must meet the following performance objectives :

- Accuracy : For lane-level positioning, particularly in safety-critical applications, sub-meter to meter-level precision is frequently needed.
- Availability : Even in challenging settings like tunnels or cities, localization data must always be accessible.
- Robustness : The system should be able to withstand multipath, noise, and partial sensor failures while still achieving a respectable level of accuracy.
- Latency : To guarantee safe and responsive vehicle behavior, real-time applications need low-latency updates, usually less than 100 milliseconds.

It is difficult to meet these requirements using GPS alone. The following sections will concentrate on the investigation of more robust and predictive localization techniques, which are driven by the combination of environmental restrictions, signal processing limitations, and hardware flaws.

3.3 Challenges of Traditional Localization Techniques

Accurate vehicle location is a fundamental requirement for VANET-enabled applications such as collision avoidance and route optimization. However, achieving reliable real-time location in highly dynamic road environments remains a challenge [17]. The main issues can be summarized as follows :

1. **Sensitivity to Environmental Conditions**– Obstacles like tunnels, buildings, and thick vegetation can especially weaken GPS signals. Signal loss occurs from total blockage, while multipath errors are caused by signal reflection in so-called "urban canyons." In a similar vein, tunnels and underpasses may completely block satellite visibility, making position data unavailable. In urban VANET applications, such situations are common, rendering GPS unreliable on its own.

2. **Delays in Signal Acquisition and Processing** – Before they can decode the navigation data and calculate position, traditional GPS receivers need to go through a signal acquisition phase. This procedure could take a few seconds, especially if there is noise and interference present or if the engine is cold-starting. The author of [Salih Alj, 2003] emphasizes that traditional acquisition architectures are frequently too sluggish for real-time automotive applications. Even though optimizations like systolic and FFT-based correlator architectures have been proposed to speed up acquisition, they still rely heavily on uninterrupted, clean signal reception.

3. **Hardware and Algorithmic Constraints** – Commonly found in civilian vehicles, single-frequency receivers are vulnerable to clock drift, Doppler shift, and ionospheric delays, all of which reduce positioning accuracy. Furthermore, while traditional filtering methods like Recursive Density Filters (RDF) and KFs work well in typical situations, they may not work well in sparse, inconsistent, or noisy data. Additionally, these models depend on Gaussian noise and linearity assumptions, which are not always true for VANETs.

4. **Inaccuracy in High-Speed or Rapidly Changing Scenarios** – The dynamics of fast-moving vehicles present new difficulties. Safety requires regular position updates, but GPS systems may not deliver quick enough or precise location updates, particularly when jittered or signal strength is poor. Under these circumstances, the error margin could be more than a few meters, which is not enough for lane-level accuracy.[20]

In light of these limitations, it becomes evident that relying solely on traditional localization techniques, especially GPS, cannot meet the demanding requirements of modern VANET applications.An alternate strategy is required to fill in the blanks and preserve precise position estimates in situations where signals are unreliable or unavailable.

Prediction-based localization, which uses motion models, historical trajectories, and machine learning to estimate vehicle positions even in the absence of GPS data, is introduced in the following section.

3.4 Importance of prediction

By making up for delays, signal loss, or inaccurate sensor data, prediction is essential to enhancing vehicle localization. Predictive models can use a vehicle's past behavior and current trajectory to estimate its future location when current data is unavailable or unreliable.

Trajectory data can be used to model temporal dependencies using deep learning techniques, particularly RNNs like LSTM and GRU. These models are perfect for forecasting vehicle movements in intricate environments because they gradually pick up patterns. Prediction can be incorporated into localization systems to :

- During GPS outages, keep your position estimates accurate.
- Make safety applications more resilient
- Encourage proactive decision-making in collision avoidance and routing

3.5 Prediction Methods

Traditional prediction methods rely heavily on mathematical modeling and filtering, where vehicle motion is approximated by physical models and iteratively updated based on sensor measurements.

3.5.1 Traditional Methods

- KF : A recursive estimator that combines predictions and measurements to estimate position. Gaussian noise and linear dynamics are assumed. According to [21], the Kalman Filter performs well in smooth environments, but its error rate is high in dynamic and cluttered situations, where deep learning techniques are more robust. To address some of its limitations, adaptations such as the Extended Kalman Filter (EKF), which handles moderate nonlinearities through linearization, and the Unscented Kalman Filter (UKF), which uses the unscented transform for better nonlinear modeling, are commonly used.

- Relative Distance Filtering (RDF) :A GPS trajectory data preprocessing method that eliminates outliers based on their distance from nearby points. A point is filtered or corrected if it deviates significantly from the local motion pattern (i.e., if its distance from neighboring points is greater than a predetermined threshold). To improve the performance of prediction models such as LSTM and GRU, RDF enhances trajectory continuity and data quality.

We will discuss this in more detail in the next section (4.6 State of the art)

3.5.2 Modern Deep Learning Methods :

DL has become a potent substitute for conventional model-based methods for vehicle localization in recent years, especially in GPS-challenged environments where traditional filtering techniques are complicated by non-linearity, noise, and missing data. RNNs, and in particular LSTM networks, are among the most efficient architectures. They have proven to be highly effective at learning motion patterns from sequential sensor data and modeling temporal dependencies. Because LSTMs don't require manual feature engineering, they can capture both short-term dynamics and long-term dependencies, making them ideal for vehicle trajectory prediction. Additionally, CNNs have been used to process spatial features, like environmental maps or sensor grids,

and are frequently used in hybrid models with LSTMs to take advantage of both temporal and spatial correlations. Transformer architectures, which are well-known for their effectiveness in natural language processing, have more recently been modified for vehicle localization tasks. This adaptation offers enhanced performance in attention-based modeling of complex driving scenarios and long-range prediction. Simultaneously, Graph Neural Networks (GNNs) have become popular in cooperative vehicular contexts. In these contexts, GNNs model vehicle interactions by representing the network as a graph of agents and their communications. V2V message exchange is advantageous for these models because it allows for collective localization that is robust against individual sensor failures. For deployment in safety-critical vehicle systems, deep learning techniques typically need substantial annotated datasets, high processing power for training, and careful management of real-time constraints, despite their accuracy and adaptability.However, they are very promising for robust and adaptive localization in contemporary VANETs due to their capacity to generalize across different environments and capture non-linear patterns.

3.6 State of the Art

The problem of vehicle localization in partially observable or GPS-denied environments has been the subject of several studies in recent years. These studies suggest a number of methods that can be broadly classified into three main categories :

3.6.1 Kalman Filter-Based Approaches

The classical Kalman Filter (KF) is widely used in vehicle localization for fusing sensor data and filtering noise in linear systems. Several studies have adapted and optimized it for vehicular contexts :

- In [22], a KF was used to enhance the localization of a two-wheeled mobile robot by fusing odometry, IMU, and ultrasonic sensor data.

The filter effectively reduced noise through predictive correction.

- Study [23] proposed a KF with threshold-based data triggering to dynamically generate location updates between nearby vehicles. This reduced message overhead and improved communication reliability and bandwidth usage.
- In [24], a memory-efficient KF algorithm was introduced for embedded systems. The approach avoids storing all previous data by discarding the second-to-last state, enabling real-time use of GPS and video data.
- Authors of [25] presented the Enhanced IAEKF (EIAE-KF), which combines kinematic information with VANET-based measurements. It adaptively updates vehicle states and demonstrated superior performance in dynamic and unstable environments.
- In [26], KF was used to predict vehicle location in urban scenarios, coping with signal degradation.
- The authors in [27] integrated GPS and motion sensor data to improve location estimation in dynamic environments.
- [28] proposed an adaptive KF-based model to mitigate environmental noise and enhance accuracy under high mobility.
- In [29] and [30], KF was employed within VANET architectures to estimate positions based on vehicle-to-vehicle communications, reducing reliance on GPS alone.

These approaches demonstrate how traditional KFs can be adapted to vehicular systems, though they are often limited in handling nonlinear motion or high sensor uncertainty.

The Extended Kalman Filter (EKF) extends the classical KF to handle nonlinear dynamics, making it more suitable for real-world vehicle localization scenarios :

- In [31], an EKF was used to predict vehicle positions under nonlinear motion using RSSI and SINR data. The method outperformed a benchmark algorithm (ESCL-VNET) in both accuracy and speed.
- The work in [32] addressed urban vehicle movement with speed variability by employing an EKF. The system assumed IEEE 802.11p-based communication and access to real-time data such as position, velocity, and acceleration.
- In [33], EKF and KF were compared using GPS and inertial navigation data. EKF achieved better accuracy, particularly on highways, due to its ability to model vehicle steering dynamics.
- Study [34] integrated EKF with reduced inertial sensor systems and trace traffic data to limit localization error without requiring synchronized trace information.
- Authors in [35] applied an EKF to fuse GPS, infrared ultrawide band V2V distance measurements, and turning data from IMUs. The approach aimed to reduce the impact of Geometric Dilution of Precision (GDOP) and improve position estimation robustness.
- Study [36] proposed a 3D Time-of-Arrival method under VANET constraints. While not strictly EKF-based, it emphasized accuracy in high-noise, low-infrastructure environments, sharing similar goals with EKF approaches.

These EKF-based methods show strong performance in environments with moderate to strong non linearity but still rely on careful tuning and accurate motion models.

3.6.2 Neural Network-Based Approaches

More recently, NNs have been introduced as learning-based models capable of learning complex motion patterns and handling noisy or incomplete data. These models leverage data such as GPS coordinates, accelerometer readings, and velocity to learn vehicle dynamics over time.

- In [26] and [27], feedforward and deep neural networks were trained on vehicle trajectory data to predict positions when GPS signals were weak or unavailable.
- In [26], a hybrid system combining Neural Networks (NN) and KF was proposed. It analyzed vehicle positions in four directions and modeled real-world vehicle dynamics, accounting for noise and directionality.
- The authors of [37] employed a pure NN-based solution for nonlinear optimization of noisy distance and angle observations between vehicles. This real-time approach effectively mitigated localization inaccuracies caused by environmental disturbances

3.6.3 Machine Learning

- Study [38] compared classical Kalman-based localization systems with machine learning models, highlighting that ML approaches can outperform filters in nonlinear, high-noise scenarios.
- This study [39] reviews the various machine learning techniques used for traffic management in VANETs. Algorithms such as random forests, support vector machines (SVMs), and Bayesian algorithms are compared. The paper identifies major challenges such as scalability, latency, and routing optimization, and shows how ML models can effectively overcome them.
- This paper [40] provides an overview of machine learning applications in intelligent vehicle networks. It covers several use cases, including mobility prediction, dynamic resource management, and distributed decision-making. It highlights the potential of ML models in complex and mobile environments, while addressing limitations in terms of real-time and security.

3.7 Conclusion

A thorough review of vehicle localization in VANETs was given in this chapter, covering basic ideas, contemporary techniques, significant obstacles, and the increasing demand for prediction-based solutions. We found promising avenues for improving localization performance by reviewing recent research and investigating both traditional and contemporary methods.

We transition from theoretical underpinnings to real world application in the following chapter. We outline the preprocessing procedures and evaluation metrics we employed in our experiments, present our suggested system architecture, and discuss the models we chose for trajectory prediction.

Chapitre 4

Implementation and Results

4.1 Introduction

This chapter presents the entire implementation process of our vehicle trajectory prediction framework, from raw data to model evaluation. Our workflow has been designed to ensure robustness, consistency, and clarity. The objective is to demonstrate the methodical process of transforming historical GPS trajectory data into accurate and understandable future location predictions using deep learning techniques. After presenting the overall architecture through a schematic workflow, we review each step of the implementation process on three different databases and with three deep learning models.

4.2 Workflow Overview

In this section, we explain the core of our contribution. Figure 8 illustrates the structured workflow of our proposed vehicle position prediction framework. Our system consists of four key stages :

1. **Data Preprocessing**, which enhances the consistency and quality of input data. Essential tasks like data cleaning, handling missing values,

feature normalization, and sequence segmentation are carried out during this phase with the aid of a lookback window. In order to improve model convergence and standardize input features, this initial step is essential.

2. **Model Development**, in which the right layers, activation functions, and hyperparameters are chosen to create the deep learning architecture to capture temporal and spatial dependencies between vehicle positions. In this study, three models are examined : CNN, CNN-LSTM, and LSTM-CNN. They are all made to capitalize on various facets of vehicle position prediction.

3. **Training**, in which the model gains better generalization and robustness by learning from past data by reducing prediction errors through optimization strategies like cross-validation, early stopping, and batch optimization.

4. **Evaluation**, uses performance metrics like mean square error (MSE), root mean square error (RMSE), and relative position error (RPE) to evaluate the predictive accuracy of trained models. This gives a quantitative indication of how well the model captures vehicle positions.

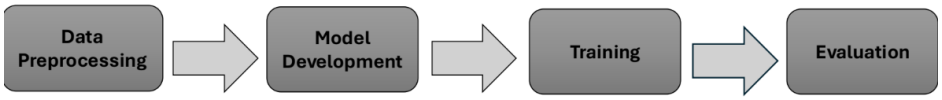

Fig. 8 : Deep Learning Workflow for position Prediction

4.3 Data Preprocessing

We used three different data sets to assess how well our deep learning model predicts the location of vehicles in CITs : **Circular, Helical, and**

L-shaped. [1] [13] These datasets provide different movement patterns, allowing us to assess the model's generalization ability across diverse vehicle movements. Each dataset consists of **40,000 time-stamped samples** recorded at a fixed interval of **0.5 seconds per sample**. The data include four features : Px, Py which represent the coordinates of the position relative to x and y, respectively, and Vx, Vy, which correspond to the velocity components along the x and y axes, respectively.The datasets are structured in a tabular format as shown in Table III.

TAB. III : Sample Structure of the Datasets

Px	**Py**	**Vx**	**Vy**
0.59	6016.33	1	1
1.58	6010.74	1	1
0.05	6002.62	1	1
0.42	5996.39	1	1
1.38	5990.74	1	1

The datasets represent different vehicle motion patterns. The Circular dataset follows a circular trajectory, where the position values (Px, Py) evolve in a circular pattern. Meanwhile, the velocity components (Vx, Vy) remain constant at 1, making it useful to evaluate the performance of the model on periodic movements. The Helical dataset follows a helical motion with simultaneous horizontal and vertical displacement, where the position values increase progressively, reflecting a more complex movement pattern. The L-shaped dataset consists of an L-shaped path with sharp turns, testing the model's ability to handle abrupt directional changes.

To better understand the nature of each dataset, Figure 9 provides a graphical representation of the three types of trajectories.

[1] Dataset Link

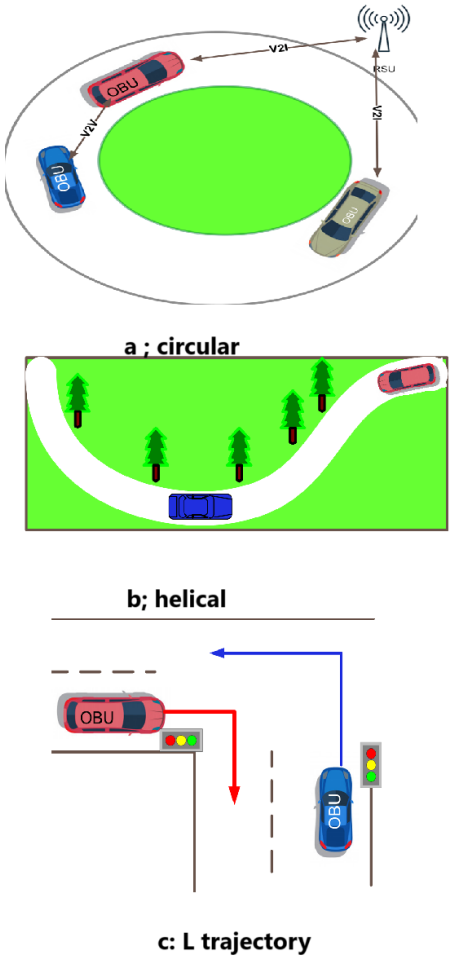

FIG. 9 : Representation of the three trajectory types

These datasets enable us to evaluate our model in various motion scenarios, guaranteeing its suitability for actual vehicular environments. The preprocessing stage ensures that the data are optimized for deep learning model training and properly formatted. It includes two main substeps : (i) data cleaning and (ii) normalization and segmentation. We initially checked the dataset for anomalies, duplicate entries, and missing values. According to the analysis, the dataset was already clean and well-structured, so no further data cleaning was necessary.

To standardize feature scales and improve model convergence, Min-Max

normalization was applied to all input features (*Px, Py, Vx, Vy*). Given an input feature X, the normalization is computed using the following formula :

$$X_{norm} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{4.1}$$

where X represents the original value, and $X_{\min}$ and $X_{\max}$ are the minimum and maximum values of that feature. This transformation scales all values to the [0, 1] range, ensuring stability during training.

A lookback mechanism was used to organize the dataset into overlapping time windows because vehicle location prediction is a sequential problem. To predict the next vehicle position, each input sequence contains the previous N time steps, with each step representing 0.5 seconds. In this study, a 20-time-step lookback window was used, meaning each input sample consists of 20 consecutive observations of (*Px, Py, Vx, Vy*), and the output is the next position (*Px, Py*) at time $t + 1$.

The dataset is divided into 80% training and 20% validation data to ensure efficient learning and performance evaluation.

These preprocessing parameters were specifically chosen for their relevance and effectiveness in time-series modeling tasks. Other preprocessing alternatives, such as feature engineering (e.g., adding speed magnitude or acceleration) or statistical outlier removal, were experimentally tested but showed no significant improvement in performance and were therefore discarded. This selection ensures that only the most impactful parameters were retained to enhance learning and generalization.

The preprocessing parameters used in this study are summarized in Table III.

Tab. III : Data Preprocessing Parameters and Justifications

Parameter	**Value**	**Justification**
Lookback Window Size	20 time steps	Empirically selected to capture 10 seconds of historical data (0.5s × 20), providing enough temporal context for accurate prediction.
Training Set Ratio	80%	Standard ratio in machine learning to ensure sufficient data for training while retaining enough for model validation.
Validation Set Ratio	20%	Complements the 80% training split and allows effective monitoring of generalization performance.
Normalization Method	Min-Max Scaling	Chosen for its simplicity and efficiency in bringing all input features to the [0, 1] range, which speeds up and stabilizes training.

We will apply the CNN, LSTM-CNN, and CNN-LSTM models to each of the three datasets.

4.4 Model Development

4.4.1 CNN model

As previously stated, the CNN model can be used to extract spatial dependencies from vehicle position sequences (Px, Py), improving the precision of future position predictions. In order to capture spatial patterns, the input is two-dimensional (2D) spatial matrices that represent historical positions. These are then processed through a sequence of one-dimensional (1D) convolutional layers with increasing filter widths of 32, 64, and 128. High-level spatial features are extracted by these convolutional layers, which are followed by the ReLU activation function. Max-pooling layers with a window size of two aid in lowering the dimensionality and avoiding overfitting. The extracted spatial features are then flattened and passed through fully connected layers ($256 \rightarrow 128 \rightarrow 64$), further refining the learned representations. The final output layer, which contains two neurons, predicts the coordinates of the next position.

4.4.2 Hybrid models

Hybrid deep learning models combine the advantages of different architectures to enhance trajectory prediction. In this study, two hybrid models were implemented : LSTM-CNN and CNN-LSTM, each integrating both LSTM and CNN layers in different sequences to capture spatial and temporal dependencies.

The LSTM-CNN architecture first processes time-series data through two LSTM layers (with 128 and 64 units, respectively) to extract long-term temporal dependencies. The refined sequential features are then passed to a set of three convolutional layers with increasing filter sizes of 32, 64, and 128 to capture spatial structures. Each convolutional layer is followed by a ReLU activation function and a max-pooling layer with a window size of 2, reducing dimensionality while preserving essential information. The output is then flattened and passed through fully connected layers ($256 \rightarrow 128 \rightarrow$

64), before reaching the final output layer, which predicts future position coordinates.

In contrast, the CNN-LSTM model first applies three convolutional layers (32, 64, and 128 filters) to extract spatial features from input sequences. These extracted spatial patterns are followed by ReLU activations and max-pooling layers (window size = 2), before being fed into two LSTM layers (128 and 64 units), which model the temporal evolution of movement. The final layers include fully connected layers ($256 \rightarrow 128 \rightarrow 64$), leading to an output layer predicting the next position coordinates.

Figure **??** presents a visual comparison between two hybrid deep learning architectures : LSTM-CNN (top) and CNN-LSTM (bottom), both used for sequence-based tasks like vehicle trajectory prediction.

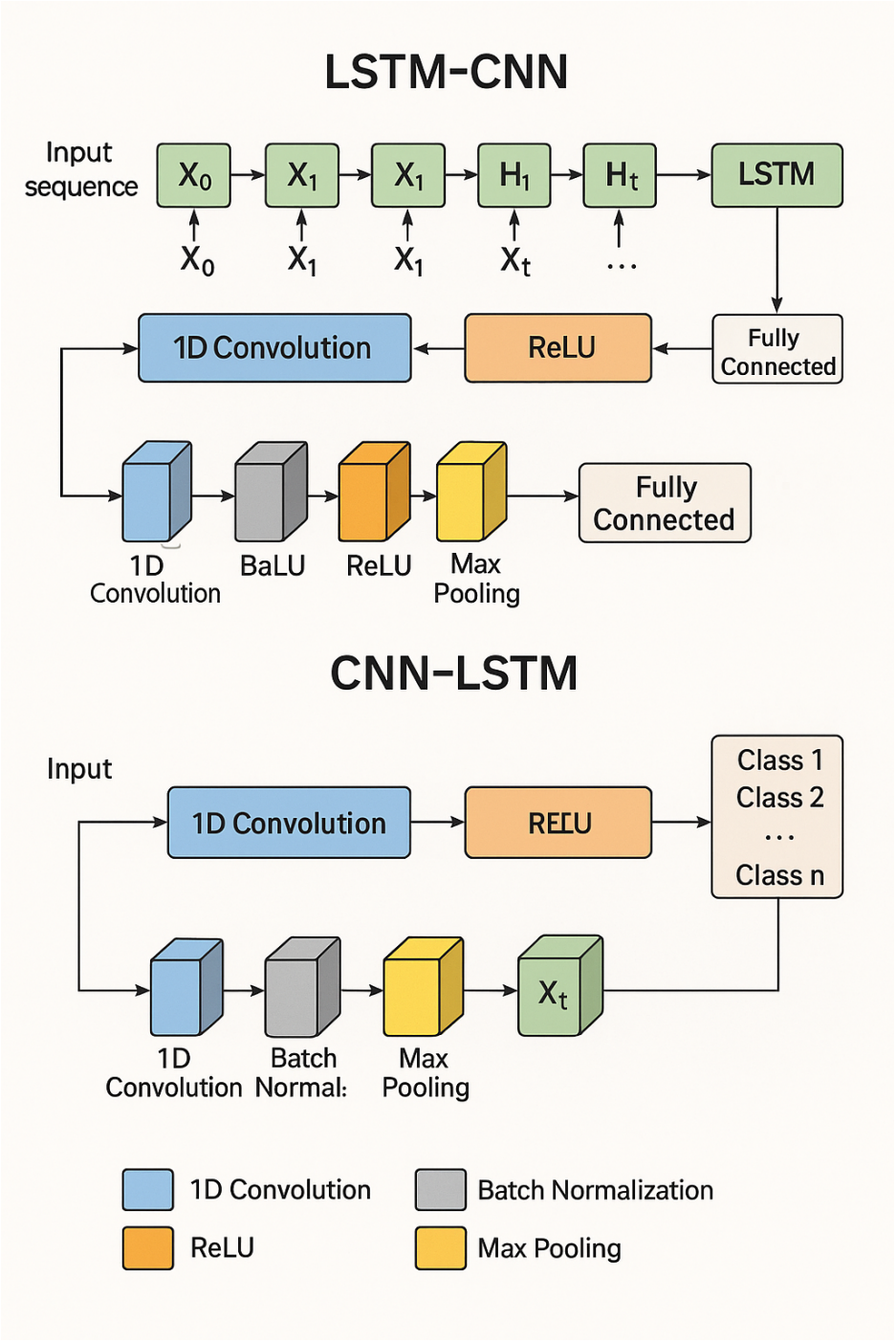

FIG. 10 : Hybrid architecture

4.5 Training Strategy

The training process involved optimizing the model using the Adam optimizer and minimizing the prediction error through the MSE loss function. Various techniques such as data augmentation and early stopping were incorporated to improve generalization and prevent overfitting. While CNNs demonstrated strong predictive capabilities, their main limitation was the lack of explicit temporal modeling. As a result, they may struggle to capture long-term sequential dependencies, which are crucial for vehicle trajectory forecasting.

The models were trained using cross-validation (k=5) to ensure robustness and prevent overfitting. The choice of Adam optimizer and MSE loss function was maintained for consistency across models. Dropout layers were also included to enhance generalization performance. A comparative analysis of these architectures revealed that LSTM-CNN achieved superior performance, particularly in capturing long-term dependencies in vehicle trajectories, while CNN-LSTM struggled with complex motion patterns such as those found in helical movements.

4.6 Evaluation Metrics

In this section, we discuss how to evaluate the performance of each model.

4.6.1 Performance metrics

To assess the effectiveness of the proposed models, multiple performance metrics were used.

The MSE was selected as the primary loss function measuring the average squared difference between predicted and actual positions. This metric provides an indication of large errors due to its quadratic nature.To complement this, the RMSE was used to measure the standard deviation of prediction errors, while the Mean Absolute Error (MAE) provided an alternative ac-

curacy metric, representing the mean absolute difference between predicted and actual values, making it less sensitive to outliers compared to MSE.

Additionally, the R^2 Score was used to evaluate how well the predicted positions matched the actual trajectory, with values closer to 1 indicating better predictive performance. The RPE was introduced to assess the model's ability to maintain positional accuracy relative to the true trajectory, particularly in cases involving abrupt directional changes. Lastly, the Mean Absolute Directional Error (MADE) was employed to quantify angular deviations in trajectory predictions, providing insight into the model's capability to predict motion direction accurately.

4.6.2 Tests and results

To facilitate a structured analysis of models performances, results are presented table V. maintains correct numbering and summarizes additional evaluation metrics, providing deeper insights into the robustness and predictive reliability of the models.

Tab. V : Comprehensive Comparison of Prediction Models Across Datasets

Model	**Dataset**	**MSE**	**RMSE**	**MAE**	**R^2 Score**	**RPE**	**MADE**
CNN	Circular	0.00006	0.00739	0.00576	0.99956	0.01545	0.00936
LSTM-CNN	Circular	0.00003	0.00569	0.00466	0.99972	0.01239	0.00673
CNN-LSTM	Circular	0.00028	0.01669	0.01220	0.99776	0.02329	0.00895
CNN	Helical	0.00005	0.00689	0.00551	0.99937	0.02782	0.01330
LSTM-CNN	Helical	0.00003	0.00538	0.00422	0.99960	0.01979	0.01143
CNN-LSTM	Helical	0.00031	0.01749	0.01220	0.99597	0.04868	0.01170
CNN	L-shaped	0.00006	0.00739	0.00467	0.99931	0.03253	0.00988
LSTM-CNN	L-shaped	0.00008	0.00866	0.00483	0.99901	0.02665	0.01306
CNN-LSTM	L-shaped	0.00018	0.01310	0.00795	0.99785	0.08152	0.00853

This results highlight the comparative performance of three deep learning models—CNN, LSTM-CNN, and CNN-LSTM—evaluated across three syn-

thetic trajectory datasets : Circular, Helical, and L-shaped. Each model was assessed based on a set of error and performance metrics, including MSE, RMSE, MAE, R^2 Score, Relative Positioning Error (RPE), and Mean Absolute Drift Error (MADE).

On the Circular dataset, the LSTM-CNN model achieved the best performance, with the lowest MSE (0.00003), RMSE (0.00569), and MAE (0.00466), along with a very high R^2 score of 0.99972. These results indicate a strong capacity of the LSTM-CNN to capture the periodic dynamics of vehicle movement. In contrast, the CNN-LSTM model exhibited significantly higher error values (MSE = 0.00028, RMSE = 0.01669), although it still maintained a reasonable R^2 score of 0.99776, suggesting that while the model generalized moderately well, its point-wise predictions lacked precision.

On the Helical dataset, the same trend was observed. LSTM-CNN again outperformed the other two models with a minimum MSE of 0.00003 and RMSE of 0.00538, accompanied by a high R^2 score of 0.99960. The CNN model performed reasonably well with a slightly higher RMSE of 0.00689, while the CNN-LSTM model struggled, producing the highest error values (MSE = 0.00031, RMSE = 0.01749) and the lowest R^2 score (0.99597) among the three. This suggests that CNN-LSTM has more difficulty modeling continuous, upward-spiraling trajectories typical of helical motion.

On the L-shaped dataset, results showed a slight deviation. Although CNN achieved the lowest MAE (0.00467) and a high R^2 score of 0.99931, the LSTM-CNN and CNN-LSTM models showed a small performance drop compared to the Circular and Helical scenarios. Interestingly, CNN-LSTM, despite having a higher MSE (0.00018), yielded the lowest MADE (0.00853), implying that while it might not produce precise absolute predictions, it performs relatively well in capturing local motion continuity. However, its RPE was significantly higher (0.08152), reflecting a potential issue with overall trajectory estimation.

In general, across all datasets, the LSTM-CNN architecture consistently achieved the best performance, particularly in terms of RMSE and MAE. This suggests that combining spatial feature extraction (CNN) with tempo-

ral sequence learning (LSTM) in the right order leads to more accurate and robust localization predictions. The CNN-LSTM, by contrast, performed less reliably, indicating that the architectural placement of convolutional layers before recurrent ones (as in LSTM-CNN) is more effective for this task. Overall, the models showed strong generalization capabilities, with R^2 scores above 0.995 in all cases, demonstrating their potential for real-world vehicle localization tasks.

4.7 Tools and Computational Environment

To assess the computational efficiency of our proposed models, we evaluated both inference time and resource utilization during model execution. All experiments were conducted using the Kaggle cloud platform without GPU acceleration. Training duration varied depending on the complexity of the models and the nature of the datasets, generally ranging from 1 hour to 1 hour and 30 minutes for medium-scale experiments.

Regarding computational resources, the maximum RAM usage fluctuated between 296.7 MiB and 1.2 GiB, while disk space utilization remained relatively constant at around 2.3 to 2.4 GiB. CPU usage ranged from 60% to 77%, indicating moderate computational demand during model training and evaluation phases.

Our implementation was carried out using the Python programming language, with deep learning frameworks such as TensorFlow and Keras. The experiments and visualizations were managed through Jupyter Notebooks, and all executions were performed using Kaggle Kernels or Google Colab environments for cloud-based computing.

4.8 Conclusion

We described in detail how we used deep learning techniques to implement our vehicle trajectory prediction system in this chapter. To optimize model

performance, we started with thorough data preprocessing, making sure the input was properly normalized, structured, and segmented. Then, in order to capture temporal and spatial patterns in vehicle trajectories, we designed and developed a number of neural network architectures.Through thorough testing and training, we were able to identify each model's advantages and disadvantages on various datasets. Predictive accuracy was evaluated using metrics like RMSE, MAE, R2 score, and RPE. The results showed that the LSTM-CNN hybrid model continuously performed the best. Even though the experiments were carried out without GPU acceleration using cloud-based platforms like Kaggle, they produced accurate and dependable results in terms of computational resource usage and accuracy.

Deep learning for vehicle location prediction in VANET environments is shown to be both feasible and effective by this implementation, especially in situations where GPS signals may be weak or nonexistent. The results obtained and their implications for actual intelligent transportation systems are thoroughly analyzed and discussed in the following chapter.

General Conclusion

The main objective of this research was to improve vehicle localization within VANETs by leveraging deep learning techniques, particularly in GPS-demanding environments. We began by exploring the foundations of C-ITS and VANETs, identifying their key components, communication patterns, and the challenges they face in ensuring safe and efficient transportation.

To address these challenges, we presented AI as a promising solution, focusing on its key areas, branches, and major applications in the transportation sector. We studied ML and DL techniques, focusing specifically on architectures such as CNN, and hybrid models such as CNN-LSTM and LSTM-CNN.

Our experimental study, conducted on different datasets, demonstrated that the hybrid LSTM-CNN model outperforms traditional and standalone models in accurately predicting vehicle positions. By replacing or supplementing GPS localization, our approach ensures more robust and reliable positioning, particularly in complex or congested urban environments.

Furthermore, we examined the limitations of traditional localization methods such as the Kalman filter and highlighted the advantages of learning-based models, particularly for handling nonlinearities and long-term dependencies.

This work also opens up avenues for future research. Integrating real-time data from multiple sources, improving model generalization to different scenarios, and incorporating privacy protection mechanisms represent potential avenues for advancing intelligent transportation systems.

This study reaffirms the transformative potential of AI and deep learning

to shape the future of vehicle networks and smart mobility.

Bibliographie

[1] M. Boubakeur. *Sécurité et protection de la vie privée dans les réseaux véhiculaires*. UNIVERSITÉ MOHAMED BOUDIAF - M'SILA, 2022.

[2] T. L. Willke, P. Tientrakool et N. F Maxemchuk. *A survey of inter-vehicle communication protocols and their applications*. IEEE Communications Surveys & Tutorials., (2009).

[3] A. Jain et V. P. Singh. "Cooperative Intelligent Transport System : An Application Domain of VANET". In : *Advances in Data and Information Sciences*. Sous la dir. de K. Chauhdary et al. Springer, 2021, p. 171-179. doi : 10.1007/978-981-16-3067-5_16.

[4] Christopher Mutschler et al. *Unlocking Artificial Intelligence : From Theory to Applications*. Springer Nature, 2024. isbn : 978-3-031-64832-8. url : https://link.springer.com/book/10.1007/978-3-031-64832-8.

[5] DataScientest. *Machine Learning : tout savoir sur cette technologie d'IA*. Accessed : 2025-05-16. 2023. url : https://datascientest.com/machine-learning-tout-savoir.

[6] B. Saoud et al. "Artificial Intelligence, Internet of Things and 6G Methodologies in the Context of Vehicular Ad-Hoc Networks (VANETs) : Survey". In : *ICT Express* (2024). In press.

[7] Boubakeur Moussaoui, Maroua Djoudi et Hanane Benhammouda. "Deep Learning Model for Intelligent Location Prediction in VANETs : LSTM-CNN A Hybrid Approach". In : *Proceedings of the IEEE International Conference on Pattern Analysis and Intelligent Systems (PAIS)* (2025). Submitted.

[8] Jason Brownlee. *Deep Learning for Time Series Forecasting : Predict the Future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery, 2018.

[9] S. Bibi et al. "An LSTM-based Outlier Detection Approach for IoT Sensor Data in Hierarchical Edge Computing". In : *2023 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2023, p. 1-6.

[10] Jay Alammar. *The Illustrated Transformer*. https://jalammar.github.io/illustrated-transformer/. Accessed : 2025-05-27. 2018.

[11] AKASH SINGH. *Anomaly Detection for Temporal Data using Long Short-Term Memory (LSTM)*. Master's Thesis at KTH Information and Communication Technology. 2017.

[12] Lawrence Wong et al. "AER : Auto-Encoder with Regression for Time Series Anomaly Detection". In : *arXiv preprint arXiv :2212.13558* (2022).

[13] "Enhancing Autoencoder Models for Multivariate Time Series Anomaly Detection". In : *The Journal of Supercomputing* (2025). doi : 10.1007/s11227-025-07044-w.

[14] Arghya Mukherjee. *Understanding the Challenges of Large Language Models (LLMs) and Their Solutions*. https://medium.com/@arghya05/understanding-the-challenges-of-large-language-models-llms-and-their-solutions-arghya-mukherjee-5e154b93cca4. Accessed May 2025. 2023.

[15] Yacine Jernite et Collaborators. *DziriBERT : A Pre-trained Language Model for Algerian Arabic*. Available at https://huggingface.co/DziriBERT. 2022.

[16] Université d'Alger et Collaborators. *DarjaBERT : A Transformer Model for Algerian Dialect NLP*. Internal project or unpublished prototype (to be confirmed). 2023.

[17] King Abdullah University of Science et Technology. *Noor : Arabic Large Language Model by Saudi Arabia*. Available at https://www.kaust.edu.sa/en/news/kaust-launches-noor. 2022.

[18] Kai Borre et al. *A Software-Defined GPS and Galileo Receiver : A Single-Frequency Approach*. Boston, MA : Birkhäuser, 2007. ISBN : 978-0-8176-4390-4. DOI : 10.1007/978-0-8176-4540-3.

[19] Stanislas Boutoille. "Systèmes de fusion pour la segmentation hors-ligne de signaux GPS multi-porteuses". Thèse de doctorat en génie informatique, automatique et traitement du signal. Thèse de doct. Calais, France : Université du Littoral Côte d'Opale, 2007.

[20] Yassine Salih Alj. "Conception d'un système d'acquisition GPS rapide". Mémoire de maîtrise en génie électrique. Mém. de mast. Montréal, Canada : École de technologie supérieure, Université du Québec, 2003.

[21] Nafiseh Rezazadeh, Mohammad Ali Amirabadi et Mohammad Hossein Kahaei. "Deep learning-based location prediction in VANET". In : *IET Intelligent Transport Systems* 18 (2024). Available at https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/itr2.12529, p. 1574-1587. DOI : 10.1049/itr2.12529.

[22] N.Z.M. Nasir et al. "Autonomous mobile robot localization using Kalman filter". In : *MATEC Web of Conferences*. T. 90. Les Ulis, France : EDP Sciences, 2017, p. 01069.

[23] Y. Mo et al. "Vehicle position updating strategy based on Kalman filter prediction in VANET environment". In : *Discrete Dynamics in Nature and Society* 2016 (2016), p. 1404396. DOI : 10.1155/2016/1404396.

[24] P. Emami, L. Elefteriadou et S. Ranka. "Tracking vehicles equipped with dedicated short-range communication at traffic intersections". In : *Proceedings of the 6th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications*. New York : ACM, 2017, p. 9-16.

[25] F.A. Ghaleb et al. "Improved vehicle positioning algorithm using enhanced innovation-based adaptive Kalman filter". In : *Pervasive and Mobile Computing* 40 (2017), p. 139-155.

[26] H. Feng et al. “Location prediction of vehicles in VANETs using a Kalman filter”. In : *Wireless Personal Communications* 80.2 (2015), p. 543-559.

[27] F. B. Günay et al. “Vehicular ad hoc network (VANET) localization techniques : A survey”. In : *Archives of Computational Methods in Engineering* 28 (2021), p. 3001-3033.

[28] R. Yaduwanshi et S. Kumar. “Location Accuracy and Prediction in VANETs using Kalman Filter”. In : *Emerging Technologies in Data Mining and Information Security : Proceedings of IEMIS 2022, Volume 3*. Sous la dir. de C. Panigrahi, C. Hota et R. Buyya. Singapore : Springer, 2022, p. 565-575.

[29] J. Liu et al. “Failure detector based on vehicle movement prediction in vehicular ad-hoc networks”. In : *IEEE Transactions on Vehicular Technology* 72.9 (2023), p. 11657-11667. doi : 10.1109/TVT.2023.3266106.

[30] K. Jiang et al. “Vehicle self-positioning via Kalman filter using multi-station non-circular signals”. In : *Signal Processing* (2024), p. 109658.

[31] H.H.A. Malki, A.I. Moustafa et M.H. Sinky. “An Improving Position Method Using Extended Kalman Filter”. In : *Procedia Computer Science* 182 (2021), p. 28-37.

[32] R.K. Jaiswal et C.D. Jaidhar. “Location Prediction Algorithm for a Nonlinear Vehicular Movement in VANET Using Extended Kalman Filter”. In : *Wireless Networks* 23 (2017), p. 2021-2036.

[33] D. Fox, W. Burgard et S. Thrun. “Markov Localization for Mobile Robots in Dynamic Environments”. In : *Journal of Artificial Intelligence Research* 11 (1999), p. 391-427.

[34] M. Elazab, A. Noureldin et H.S. Hassanein. “Integrated Cooperative Localization for Connected Vehicles in Urban Canyons”. In : *2015 IEEE Global Communications Conference (GLOBECOM)*. Piscataway : IEEE, 2015, p. 1-6.

[35] G.M. Hoang et al. “Mitigating Unbalanced GDoP Effects in Range-Based Vehicular Cooperative Localization”. In : *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*. Piscataway : IEEE, 2017, p. 659-664.

[36] A.R. Ansari et al. “Accurate 3D Localization Method for Public Safety Applications in Vehicular Ad-hoc Networks”. In : *IEEE Access* 6 (2018), p. 20756-20763.

[37] J. Eom et al. “DNN-Assisted Cooperative Localization in Vehicular Networks”. In : *Energies* 12.14 (2019), p. 2758. doi : 10.3390/en12142758.

[38] L. N. Balico et al. “Localization Prediction in Vehicular Ad Hoc Networks”. In : *IEEE Communications Surveys & Tutorials* 20.4 (2018), p. 2784-2803. doi : 10.1109/COMST.2018.2846698.

[39] D.K. Sharma et al. “Machine learning models and techniques for VANET based traffic management : A survey”. In : *Peer-to-Peer Networking and Applications* 14 (2021), p. 1732-1757. doi : 10.1007/s12083-020-00993-4.

[40] Y. Zhou et al. “Machine learning in vehicular networking : An overview”. In : *Digital Communications and Networks* 7.4 (2021), p. 558-567. doi : 10.1016/j.dcan.2021.04.001.

Annex : Our Published Paper

Deep Learning Model for Intelligent Location Prediction in VANETs: LSTM-CNN A Hybrid Approach

1st Boubakeur Moussaoui
Computer Science Department
University of Mohamed el Bachir el Ibrahimi
Bordj Bou Arreridj, Algeria
Laboratory of Informatics and its Applications of M'sila (LIAM)
b.moussaoui@univ-bba.dz

2nd Maroua Djoudi
Computer Science Department
University of Mohamed el Bachir el Ibrahimi
Bordj Bou Arreridj, Algeria
marouadjoudi677@gmail.com

3rd Hanane Benhammouda
Computer Science Department
University of Mohamed el Bachir el Ibrahimi
Bordj Bou Arreridj, Algeria
hananebenhammouda4@gmail.com

***Abstract*—In Cooperative Intelligent Transport Systems (C-ITS), vehicles and/or RSUs exchange various messages to improve traffic management, promote collision prevention, and ensure safety on our roads. These messages are broadcast within the vehicle's vicinity. Each message must contain the accurate position of the vehicle, along with other relevant information such as the velocity and the vehicle identifier. Accurate information on the vehicle's position significantly affects the majority of sensitive applications. Finding an approach to predict timely and accurately the exact position of a vehicle has become a great challenge in CITS. This study addresses the problem of Vehicle Location Prediction (VLP) using a deep learning approach. A hybrid solution based on Convolutional Neural Networks (CNNs) and Long-Short-Term-Memory (LSTM) is proposed. While CNN excels at extracting spatial patterns from positions sequences, LSTM captures long-term temporal dependencies, providing a comprehensive spatiotemporal representation. A comparative analysis has been conducted to evaluate the performance of our approach; the results showed that the CNN-LSTM model outperforms other models in terms of localisation error.**

***Index Terms*—Deep Learning, VLP, CNN, LSTM, VANET.**

I. Introduction

The rapid growth of Vehicular Ad-hoc Networks (VANETs), particularly in CITS, promises to improve road safety, traffic management, and transportation efficiency. VANETs enable communication between vehicles (V2V) and/or with the infrastructure (V2I). These networks support a variety of critical applications, including accident prevention, cooperative driving, and real-time traffic updates. Using wireless communication technologies such as Dedicated Short-Range Communications (DSRC) and 5G, VANETs facilitate seamless data exchange between mobile nodes. This improves road safety and enhances transportation efficiency. For effective decision-making, many CITS applications require the exchange of sensitive information (such as identity, location, speed, direction, etc.) [1]. The accuracy of node locations within the network is critical, as this information directly affects the system's reliability, particularly in safety applications and location-based services. However, security and privacy issues arise when an attacker exploit this accurate positioning to track a vehicle during its journey [2], [3], [4].

Connected and autonomous vehicles are equipped with a Global Positioning System (GPS) unit. This unit allows real-time tracking and navigation by receiving signals from satellites to calculate the vehicle's position. The OBU processes these data to provide route guidance, monitor vehicle movements, and support various applications in intelligent transportation systems. An enhanced GPS method, called Differential GPS (DGPS), was developed to improve location accuracy. These systems have certain limitations in CITS, such as low accuracy in urban areas, signal disturbance due to weather conditions, and failures in tunnels.

In this research, we develop an efficient deep learning-based localisation model that ensures high accuracy and robustness in CITS. Using hybrid deep learning models, we aim to mitigate GPS limitations, improve real-time positioning, and support the advancement of intelligent transportation systems. This study will explore various localisation models, evaluate their performance under different scenarios, and provide information on the effectiveness of AI-driven localisation in modern vehicular networks. The findings will contribute to the development of reliable and intelligent transportation solutions, paving the way for safer and more autonomous vehicle environments. The main contributions of this study are described as follows:

- Implementation of three different neural network models

979-8-3315-2625-2/25/$31.00 ©2025IEEE

- Training of these models using three datasets.
- Analysis and evaluation of the prediction errors of each model.

This paper is organized as follows: in Section II, we present the main related works. Section III describes a theoretic background of deep-learning. In Section IV, we present our proposed approach until the evaluation. Finally, we conclude the work in Section V.

However, in real-world scenarios, GPS signals can be disrupted or degraded by different factors. These include urban canyons, where buildings block the signal; tunnels or underground areas; and dense forests or regions with poor satellite visibility.

II. Related works

VLP is widely addressed in many studies related to CITS area. Several approaches have been proposed. In fact, it is difficult to achieve an accurate position that can cope with the high mobility of vehicles and communication link failures. One potential solution is to use GPS embedded in the On-Board Units (OBU) of connected vehicles. GPS alone, in ideal conditions, such as clear satellite visibility, can provide accurate and reliable position data. However, in real-world scenarios, GPS signals can be disrupted or degraded by different factors. These include urban canyons, where buildings block the signal; tunnels or underground areas; and dense forests or areas with poor satellite visibility. In such cases, GPS alone can no longer meet the demand for vehicle positioning, which is why Dead Reckoning (DR) or Kalman Filtering (KF) is often used to improve accuracy.

The Kalman Filter is one of the most interesting approaches proposed in this scope [5], [6], [7], [8], [9], [10]. KF considers the linear nature of the mobility pattern in VANETs, and has been a common practice. Due to the quick convergence of KF, enhanced accuracy can be attained in real-time scenarios, as it only needs the previous state and does not store any historical data.

Other studies are based on the Dead Reckoning method, which relies on estimating the vehicle's current position based on its previous position, speed, and heading [11]. While it does not directly depend on GPS, GPS may be used to periodically update the DR estimates to correct any accumulated errors.

Without GPS updates, DR and KF become less accurate over time due to errors accumulating from sensor inaccuracies. However, Neural Networks (NN) are known as learning-based approaches. These approaches can learn to predict localisation based on patterns in data; they include, among other inputs, such as accelerometer readings and the value received from the GPS signal. After training, the NN models can estimate a vehicle's position even when GPS signals are weak or unavailable, as long as the model has been trained with enough data and other sensor inputs [12], [13], [14], [15].

The authors in [16] present a hybrid localisation scheme, combining optimized techniques with a robust MAC scheme for efficient beaconing. The proposed approach minimises delays, adapts to traffic conditions, and ensures accurate, GPS-independent localization.

III. Theoretic background

A. Background on Neural Networks

Artificial Neural Networks (ANNs) are computational models inspired by the human brain. They consist of multiple layers of interconnected neurons, where each neuron applies an activation function to a weighted sum of its inputs. These networks are widely used to recognize patterns, make predictions, and solve complex problems [17]. A standard ANN consists of three main layers:

- The input layer, which receives the raw input data.
- The hidden layers, responsible for extracting features and learning patterns using weights and activation functions.
- The output layer, which produces the needed prediction or classification result.

The mathematical formulation of a neuron is given by:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{1}$$

where : w_i represents the weights, x_i denotes the inputs, b is the bias, and $f(\cdot)$ is the activation function. The activation functions most commonly used are ReLU (Rectified Linear Unit) and the Sigmoid function.

Recurrent Neural Networks (RNN) were later designed for handling sequential data, by storing a copy of previous inputs. RNNs manage sequential data by using recurrent temporal dependencies in the form of hidden states. Unlike traditional feed-forward networks, RNNs have loops that let them transmit information from previous inputs to next inputs. However, one a limitation of regular RNNs is that they do not effectively address the vanishing gradient problem, making it difficult for them to learn long-term dependencies in sequences.

B. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are specialised deep learning models created to process spatial data. They are particularly effective at extracting hierarchical features from structured inputs. CNNs exploit convolutional layers that apply a set of learnable filters to detect and extract spatial patterns. These are followed by pooling layers that reduce the spatial dimensions of the data using downsampling. In this manner, it reduces computational load and helps prevent overfitting while retaining the most critical features. Finally, fully connected layers take the high-level features extracted by the convolutional and pooling layers and transform them into a dense vector representation. This representation is then used to make predictions by mapping the features to the output as illustrated in Figure 1.

The fundamental operation in CNNs is the convolution, defined as:

$$(I * K)(x, y) = \sum_{m} \sum_{n} I(x + m, y + n) K(m, n) \tag{2}$$

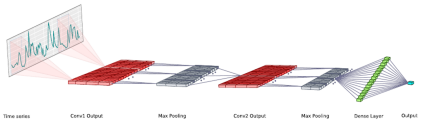

Fig. 1: CNN Architecture

where I represents the input, K denotes the kernel, and (x, y) are spatial coordinates. This process enables CNNs to capture spatial relationships in data, thereby improving prediction accuracy.

C. Long Short-Term Memorys (LSTMs)

Long-Short-Term Memorys (LSTMs) networks are a specialised type of RNN designed to capture temporal dependencies in sequential data. Unlike standard RNNs, which suffer from the vanishing gradient problem, LSTM networks include memory cells and gating mechanisms that regulate the flow of information over long time steps. An LSTM cell consists of three key gates (Figure 2) [18]:

- The forget gate, which decides what information to discard.
- The input gate, which updates the cell state with new information.
- The output gate, which determines the final output.

To illustrate how LSTMs effectively capture long-term dependencies, making them convenient for sequential location prediction, we present the following equations:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5}$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{6}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{7}$$
$$h_t = o_t \odot \tanh(C_t) \tag{8}$$

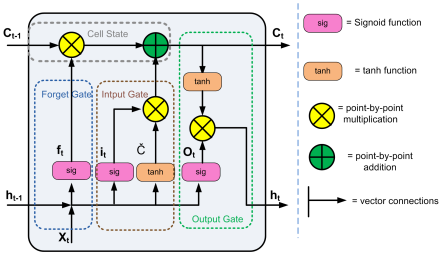

Fig. 2: Structure of an LSTM cell

Table I provides a brief overview of the main symbols and notations used in the previous equations.

TABLE I: SUMMARY OF SYMBOLS AND NOTATIONS

Symbols and Notations	Description
$f^{(t)}, i^{(t)}, o^{(t)}$	The forget, input, and output gate at time t, respectively
σ	Logistic sigmoid function
W_f, W_i, W_o	Weight matrices of the forget, input and output gates, respectively
b_f, b_i, b_o, b_C	The biases of the forget, input, output gates, and the cell state, respectively
$h^{(t-1)}, h^{(t)}$	The hidden state of the previous and timestamp, respectively
$x^{(t)}$	The current input
$C^{(t)}$	The current cell state
$\tilde{C}^{(t)}$	The new memory cell state candidate
$\odot$	Element-wise (Hadamard) product

To effectively model vehicle location, we leverage CNNs to extract spatial features and LSTMs to capture temporal dependencies.

IV. Proposed Approach

In this section, we explain the core of our contribution. Figure 3 illustrates the structured workflow of our proposed position prediction framework. Our scheme consists of four key stages:

1) **Data Preprocessing**, which improves the quality and consistency of input data. We perform in this phase essential operations such as data cleaning, missing value handling, feature normalization, and sequence segmentation using a lookback window. This first step is crucial for standardizing input features and improving model convergence.
2) **Model Development**, where the deep learning architecture is designed by selecting the appropriate layers, activation functions, and the hyperparameters to capture spatial and temporal dependencies between vehicle positions. In this study, we explore three models: CNN, LSTM-CNN and CNN-LSTM. Each of them is designed to take advantage of different aspects of predicting vehicle's position.
3) **Training**, where the model learns from historical data by minimizing prediction errors using optimization techniques such as cross-validation, early stopping and batch optimization, ensuring improved generalization and robustness.
4) **Evaluation**, the final step, which assesses the predictive accuracy of trained models using performance metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Relative Position Error (RPE), providing a quantitative measure of the effectiveness of the model in capturing vehicle positions.

A. Step 1 :Data preprocessing

To evaluate our deep learning model for accurately predicting vehicle's location in CITs, we use three distinct datasets:

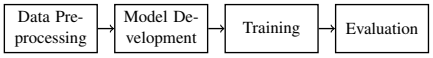

Fig. 3: Deep Learning Workflow for position Prediction

Circular, Helical, and L-shaped. [1]

These datasets provide different movement patterns, allowing us to assess the model's generalization ability across diverse vehicle movements. Each dataset consists of **40,000 time-stamped samples** recorded at a fixed interval of **0.5 seconds per sample**. The data include four features: Px, Py which represent the coordinates of the position relative to x and y, respectively, and Vx, Vy, which correspond to the velocity components along the x and y axes, respectively.

The datasets represent different vehicle motion patterns. The Circular dataset follows a circular trajectory, where the position values (Px, Py) evolve in a circular pattern. Meanwhile, the velocity components (Vx, Vy) remain constant at 1, making it useful to evaluate the performance of the model on periodic movements. The Helical dataset follows a helical motion with simultaneous horizontal and vertical displacement, where the position values increase progressively, reflecting a more complex movement pattern. The L-shaped dataset consists of an L-shaped path with sharp turns, testing the model's ability to handle abrupt directional changes.

The preprocessing phase ensures that the data are well-formatted and optimised for training deep learning models. It consists of two main sub-steps: (i) data-cleaning and (ii) normalization and segmentation.

Before applying any transformations, we examined the dataset for missing values, duplicate entries, and anomalies. The analysis revealed that the dataset was already clean and well-structured. Therefore, no additional data cleaning operations were required.

To standardize feature scales and improve model convergence, we applied Min-Max normalisation to all input features (*Px, Py, Vx, Vy*). Given an input feature X, the normalization is computed as follows:

$$X_{norm} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{9}$$

Where X represents the original feature value, and $X_{\min}$ and $X_{\max}$ denote the minimum and maximum values of that feature, respectively. This transformation scales all data values between 0 and 1, ensuring stable training.

Since VLP is a sequential problem, the dataset is structured into overlapping time windows using a lookback mechanism. Each input sequence consists of the past N time steps (each lasting 0.5 seconds) to predict the next vehicle position. In this study, we use a lookback window of 20 time steps, which means that the input to the model is a sequence of 20 consecutive observations (*Px, Py, Vx, Vy*), while the output corresponds to the next position (*Px, Py*) at time $t + 1$. To ensure effective training, the dataset is split into 80% training data and 20% validation data. Additionally, Min-Max scaling is applied to normalise all input features, ensuring that values remain within a standard range to enhance model convergence.

The preprocessing parameters used in this study are listed in Table II.

we will apply the CNN, LSTM-CNN, and CNN-LSTM models to each of the three datasets.

B. Step 2 :Model development

1) CNN model: In this section, a CNN model will be used to predict future vehicle positions in the three available movement patterns. As mentioned earlier, the CNN model is useful for extracting spatial dependencies from sequences of vehicle positions (Px, Py), thus enhancing the accuracy of future position predictions.

The input consists of two-dimensional (2D) spatial matrices, representing past positions, which are processed through a series of one-dimensional (1D) convolutional layers with increasing filter widths of 32, 64, and 128 to capture spatial patterns. These convolutional layers, followed by the ReLU activation function, extract high-level spatial features, while max-pooling layers with a window size of 2 help to reduce the dimensionality and to prevent overfitting.

The extracted spatial features are then flattened and passed through fully connected layers ($256 \rightarrow 128 \rightarrow 64$), further refining the learned representations. The final output layer, which contains two neurons, predicts the coordinates of the next position.

2) Hybrid models: Hybrid deep learning models combine the advantages of different architectures to enhance trajectory prediction. In this study, two hybrid models were implemented: LSTM-CNN and CNN-LSTM, each integrating both LSTM and CNN layers in different sequences to capture spatial and temporal dependencies.

The LSTM-CNN architecture first processes time-series data through two LSTM layers (with 128 and 64 units, respectively) to extract long-term temporal dependencies. The refined sequential features are then passed to a set of three convolutional layers with increasing filter sizes of 32, 64, and 128 to capture spatial structures. Each convolutional layer is followed by a ReLU activation function and a max-pooling layer with a window size of 2, reducing dimensionality while preserving essential information. The output is then flattened and passed through fully connected layers ($256 \rightarrow 128 \rightarrow 64$), before reaching the final output layer, which predicts future position coordinates.

TABLE II: Data preprocessing parameters

Parameter	**Value**
Lookback Window Size	20 time steps
Training Set	80%
Validation Set	20%
Normalization Method	Min-Max Scaling

[1]Dataset Link

In contrast, the CNN-LSTM model first applies three convolutional layers (32, 64, and 128 filters) to extract spatial features from input sequences. These extracted spatial patterns are followed by ReLU activations and max-pooling layers (window size = 2), before being fed into two LSTM layers (128 and 64 units), which model the temporal evolution of movement. The final layers include fully connected layers (256 $\rightarrow$ 128 $\rightarrow$ 64), leading to an output layer predicting the next position coordinates.

C. Step 3 : Training

The training process aimed to optimise model performance while ensuring generalisation across different trajectory patterns. Each model was trained using the Adam optimiser (learning rate = 0.001) with MSE as the loss function to minimise prediction errors. To enhance generalization and prevent overfitting, a dropout rate of 0.3 was applied, and cross-validation (k=5) was implemented to ensure robustness across datasets. For all models, early stopping was incorporated, halting training when validation loss stopped improving for a predefined number of epochs. In Addition, data augmentation techniques were applied by introducing slight modifications to position values to enhance model robustness.

D. Step 4 : Evaluation

In this section, we discuss how to evaluate the performance of each model.

1) Performance metrics: To assess the effectiveness of the proposed models, multiple performance metrics were used.

The MSE was selected as the primary loss function measuring the average squared difference between predicted and actual positions. This metric provides an indication of large errors due to its quadratic nature.To complement this, the RMSE was used to measure the standard deviation of prediction errors, while the Mean Absolute Error (MAE) provided an alternative accuracy metric, representing the mean absolute difference between predicted and actual values, making it less sensitive to outliers compared to MSE.

Additionally, the R^2 Score was used to evaluate how well the predicted positions matched the actual trajectory, with values closer to 1 indicating better predictive performance. The RPE was introduced to assess the model's ability to maintain positional accuracy relative to the true trajectory, particularly in cases involving abrupt directional changes. Lastly, the Mean Absolute Directional Error (MADE) was employed to quantify angular deviations in trajectory predictions, providing insight into the model's capability to predict motion direction accurately.

2) Tests and results: To facilitate a structured analysis of models performances, results are presented in two tables. The first one (Table III) reports common error metrics such as MSE, RMSE, and MAE, which quantify the accuracy of the predictions. The second one (Table IV) maintains correct numbering and summarizes additional evaluation metrics, including the R^2 Score, RPE, and MADE, providing deeper insights into the robustness and predictive reliability of the models.

TABLE III: Comparison of Error Metrics

Model	Dataset	MSE	RMSE	MAE
CNN	Circular	0.00006	0.00739	0.00576
LSTM-CNN	Circular	0.00003	0.00569	0.00466
CNN-LSTM	Circular	0.00028	0.01669	0.01220
CNN	Helical	0.00005	0.00689	0.00551
LSTM-CNN	Helical	0.00003	0.00538	0.00422
CNN-LSTM	Helical	0.00031	0.01749	0.01220
CNN	L-shaped	0.00006	0.00739	0.00467
LSTM-CNN	L-shaped	0.00008	0.00866	0.00483
CNN-LSTM	L-shaped	0.00018	0.01310	0.00795

TABLE IV: Comparison of Performance Metrics

Model	Dataset	R^2 Score	RPE	MADE
CNN	Circular	0.99956	0.01545	0.00936
LSTM-CNN	Circular	0.99972	0.01239	0.00673
CNN-LSTM	Circular	0.99776	0.02329	0.00895
CNN	Helical	0.99937	0.02782	0.01330
LSTM-CNN	Helical	0.99960	0.01979	0.01143
CNN-LSTM	Helical	0.99597	0.04868	0.01170
CNN	L-shaped	0.99931	0.03253	0.00988
LSTM-CNN	L-shaped	0.99901	0.02665	0.01306
CNN-LSTM	L-shaped	0.99785	0.08152	0.00853

The results indicate that LSTM-CNN consistently achieves the lowest MSE, RMSE, and MAE across all datasets, demonstrating its superior ability to model both spatial and temporal dependencies. CNN performs slightly worse but remains a strong alternative, particularly for datasets with less complex temporal variations. In contrast, CNN-LSTM exhibits the highest errors, suggesting that extracting spatial features first may not be optimal for position prediction.

In addition to the error metrics, the R^2 Score confirms the model's ability to fit the data, with LSTM-CNN achieving values closest to 1, indicating better predictive accuracy. The RPE further supports this finding, as LSTM-CNN consistently produces the lowest RPE, meaning its predicted positions are the closest to the actual trajectory. On the other hand, CNN-LSTM shows the highest RPE, suggesting difficulties in maintaining precise positional accuracy, particularly in helical and L-shaped movements.

Lastly, the MADE highlights the model's ability to predict motion direction accurately. LSTM-CNN exhibits the lowest MADE, confirming its effectiveness in capturing both spatial and directional dependencies. CNN maintains reasonable performance, while CNN-LSTM struggles with directional accuracy, further reinforcing its limitations in handling complex motion patterns.

These findings emphasize the importance of hybrid architectures in achieving accurate trajectory predictions, with LSTM-CNN emerging as the most effective model in this study.

V. Conclusion

The accurate and timely prediction of vehicle positions in CITS plays a crucial role, particularly in safety and emergency applications, which require the exact position of alert. Precise position prediction not only enhances route planning and traffic management but also reduces the risk of accidents by enabling proactive interventions. Therefore, developing robust and efficient deep learning models for VLP task holds immense potential to improve CITS applications.

In this study, we explored only deep learning-based approaches for VLP in VANETs. CNN can extract spatial patterns from trajectory sequences, while LSTM can capture temporal dependencies. The performance was assessed using multiple evaluation metrics, including MSE, RMSE, MAE, R^2 Score, RPE, and MADE.

Results revealed that the combination LSTM-CNN consistently outperformed the other models across all used datasets, achieving the lowest errors MSE, RMSE, and MAE. This indicates that prioritizing temporal dependencies before extracting spatial features enhances prediction accuracy. Future research can further optimize these models by exploring attention mechanisms to enhance temporal dependency learning or integrating additional contextual information such as road constraints, traffic density, and external environmental factors. Additionally, evaluating these models on real-world vehicular datasets rather than simulated trajectories would provide deeper insights into their robustness and practical applicability in VANET environments.

References

[1] C. Creß, Z. Bing, A. C. Knoll, Intelligent transportation systems using roadside infrastructure: A literature survey, IEEE Transactions on Intelligent Transportation Systems (2023).

[2] M. AlMarshoud, M. Sabir Kiraz, A. H. Al-Bayatti, Security, privacy, and decentralized trust management in vanets: A review of current research and future directions, ACM Computing Surveys 56 (10) (2024) 1–39.

[3] B. Moussaoui, N. Chikouche, H. Fouchal, An efficient privacy scheme for c-its stations, Computers and Electrical Engineering 107 (2023) 108613.

[4] I. Ullah, M. A. Shah, Sgo: Semantic group obfuscation for location-based services in vanets, Sensors 24 (4) (2024) 1145.

[5] H. Feng, C. Liu, Y. Shu, O. W. Yang, Location prediction of vehicles in vanets using a kalman filter, Wireless personal communications 80 (2015) 543–559.

[6] P. Emami, L. Elefteriadou, S. Ranka, Tracking vehicles equipped with dedicated short-range communication at traffic intersections, in: Proceedings of the 6th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications, 2017, pp. 9–16.

[7] F. B. Günay, E. Öztürk, T. Çavdar, Y. S. Hanay, A. u. R. Khan, Vehicular ad hoc network (vanet) localization techniques: a survey, Archives of Computational Methods in Engineering 28 (2021) 3001–3033.

[8] R. Yaduwanshi, S. Kumar, Location accuracy and prediction in vanets using kalman filter, in: Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2022, Volume 3, Springer, 2022, pp. 565–575.

[9] J. Liu, S. Chen, G. Gui, H. Gacanin, H. Sari, F. Adachi, Failure detector based on vehicle movement prediction in vehicular ad-hoc networks, IEEE Transactions on Vehicular Technology 72 (9) (2023) 11657–11667. doi:10.1109/TVT.2023.3266106.

[10] K. Jiang, Y. Shi, Z. Yang, X. Zhang, Vehicle self-positioning via kalman filter using multi-station non-circular signals, Signal Processing (2024) 109658.

[11] R. Raghu, R. Prabhushankar, J. Rajaram, M. Vaiyapuri, Efficient dead reckoning approach for localization prediction in vanets, J Appl Sci Comput 6 (3) (2019) 2093–2099.

[12] M. Dörterler, Ö. Faruk Bay, Neural network based vehicular location prediction model for cooperative active safety systems, Promet-Traffic&Transportation 30 (2) (2018) 205–215.

[13] J. Eom, H. Kim, S. H. Lee, S. Kim, Dnn-assisted cooperative localization in vehicular networks, Energies 12 (14) (2019) 2758.

[14] Amirabadi, M. Ali, Deep learning-based location prediction in vanet, IET Intelligent Transport Systems 18 (2024).

[15] N. Rezazadeh, M. A. Amirabadi, M. H. Kahaei, Deep learning-based location prediction in vanet, IET Intelligent Transport Systems 18 (9) (2024) 1574–1587.

[16] B. Amira, F. Lazhar, Hybrid accurate localization approach for dense and highly mobile vehicular networks, International Journal of Communication Systems 38 (4) (2025) e6139.

[17] B. Saoud, I. Shayea, A. E. Yahya, Z. A. Shamsan, A. Alhammadi, M. A. Alawad, Y. Alkhrijah, Artificial intelligence, internet of things and 6g methodologies in the context of vehicular ad-hoc networks (vanets): Survey, ICT Express (2024).

[18] S. Bibi, C. Titouna, F. Titouna, F. Naït-Abdesselam, An lstm-based outlier detection approach for iot sensor data in hierarchical edge computing, in: 2023 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), IEEE, 2023, pp. 1–6.