Doctoral Dissertation
Doctoral Program in Energy Engineering (36^{th}cycle)

Artificial Intelligence Solutions to Enhance Energy Saving and Battery Management in Electrified and Connected Vehicles

By

Alessia Musa

******

Supervisor(s):
Prof. Daniela Anna Misul, Supervisor
Prof. Ezio Spessa, Co-Supervisor

Doctoral Examination Committee:
Prof. Nicolò Cavina, Referee, Alma Mater Studiorum Università di Bologna
Prof. Robert Prucka, Referee, Clemson University
Prof. Manfredi Villani, The Ohio State University
Prof. Roberto Finesso, Politecnico di Torino
Prof. Mirko Baratta,Politecnico di Torino

Politecnico di Torino
2024

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Alessia Musa
2024

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

To my parents and my sisters.
Thank you, for everything.

Abstract

This thesis explores cutting-edge solutions applied to electrified vehicle technologies, focusing on the interaction between artificial intelligence (AI) and the performance of hybrid electric vehicles (HEVs) and battery electric vehicles (BEVs) in the context of environmental sustainability , operational efficiency and system reliability. It addresses pivotal research questions aimed at optimizing fuel economy and passenger comfort in HEVs, enhancing traffic flow and reducing energy consumption in BEVs through trajectory optimization, and accurately estimating the state of health (SoH) of high-voltage batteries, crucial for battery management and their lifespan. The study transitions from HEVs to BEVs, reflecting regulatory trends and focusing on AI-based control algorithms' adaptability for real-time applications, evaluating their potential to make a significant step toward operational efficiency and environmental sustainability.

The thesis comprises AI techniques in energy management for HEVs, apply AI in cooperative adaptive cruise control (CACC) for BEVs, and delve into estimating the SoH of BEV batteries. It begins with an exploration of energy management in HEVs, utilizing LSTM neural networks and reinforcement learning (RL) to enhance fuel efficiency, engine efficiency, and passenger comfort under varying driving conditions. The narrative then shifts to optimizing BEV trajectories using deep learning and vehicle communication technologies, highlighting the potential of GRU architectures and subsequently RL in developing robust CACC systems amid sensor and communication uncertainties. Lastly, it examines machine learning algorithms' efficiency in estimating the SoH of high-voltage batteries in electric vehicles, considering experimental data from vehicles with diverse mileage conditions.

Through these chapters, the thesis presents a blend of supervised learning and RL approaches, applying machine learning and deep learning techniques to tackle distinct challenges within the electrified vehicle technology domain. The research underlines the transformative potential of AI in advancing electrified vehicle technologies, aiming for a sustainable future in transportation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we outline the motivation behind the thesis, detail the specific research questions it addresses, and provide the necessary background.

1.1 Motivation

In an era increasingly dominated by concerns over environmental sustainability and the advancement of autonomous driving technologies, the role of electrified and connected vehicles has become pivotal for the future of transportation. The evolution of hybrid electric vehicles (HEVs) and battery electric vehicles (BEVs) was identified as crucial for reducing carbon emissions, yet it also posed new challenges in energy efficiency and battery management.

HEVs, which merge a conventional internal combustion engine with an electric propulsion system, enhance fuel economy and lower CO_2 emissions compared to traditional vehicles but still depend on fossil fuels. Conversely, BEVs, powered solely by electricity, emit no tailpipe CO_2 and are thus favored by governments and regulatory bodies aiming to mitigate greenhouse gas emissions and address climate change.

In this scenario, the automotive industry has heavily invested in electrified vehicles (EVs) as a strategy to achieve carbon-neutral transportation, mitigate environmental pollution, and avoid regulatory fees [2, 3]. Amidst this shift, the market dynamics for HEVs are still evolving. By 2026, the hybrid vehicle market

is projected to reach USD 882.88 billion, growing at a compound annual growth rate (CAGR) of 20.6%. The Asia-Pacific region, led by manufacturers such as Toyota, Honda, Nissan, Kia, BYD, and Hyundai, is expected to dominate this growth [4]. The plug-in hybrid segment, in particular, is anticipated to see the highest growth, fueled by increasing demand for electric solutions and better charging infrastructure. The electric vehicle market is also on the rise, projected to hit USD 1393.33 billion by 2027 with a CAGR of 19.19%. European automakers like Volkswagen, Mercedes Group, BMW, Volvo, and Renault are also expanding their market presence, closing in on their Asian competitors [5].

The rapid growth of the HEV and BEV market signifies a significant shift towards environmentally friendly transportation. However, several key technological challenges must be addressed for these vehicles to become the dominant choice. In HEVs, innovative energy management strategies are crucial to maximize fuel efficiency and minimize environmental impact. Conversely, BEVs offer a sustainable alternative, but require breakthroughs in accurately predicting battery health, a critical factor for their long-term viability. BEVs, in addition, present a platform for the development of advanced driver-assistance systems (ADAS) that can further enhance road safety. One such system is cooperative adaptive cruise control (CACC), which optimizes vehicle spacing to reduce energy consumption. However, the effectiveness of CACC is currently challenged by uncertainties in speed profiles and sensor measurements, highlighting areas for further research in this field.

Reflecting these trends, scientific research has increasingly focused on these technologies, leading to a shift in this thesis from HEVs to BEVs in line with regulatory trends over the past five years. Initially, we explored the energy management control problem in HEVs, aiming to boost fuel economy, and so reduce carbon dioxide emissions, without compromising vehicle performance. Subsequently, our attention shifted to BEVs, examining their performance in terms of battery status and their efficiency in connected driving scenarios. We delved into artificial intelligence (AI)-based control algorithms, investigating their adaptability and potential for real-time implementation, which marks a significant step towards not only operational efficiency but also environmental sustainability. The incorporation of AI techniques in this thesis is motivated by their potential to address the complex dynamics of energy efficiency and battery management in electrified vehicles. AI's analytical capabilities and adaptability

are key to developing solutions that could not only enhance vehicle performance but also contribute to environmental sustainability. Indeed, this research aims to assess whether these AI techniques can go beyond the limitations of current state-of-the-art methods by creating specific strategies for each problem being examined. The objective is to determine if it's possible to use these advanced learning approaches to not only match but surpass the capabilities of existing technologies, thereby providing customized solutions tailored to the unique challenges faced.

The remaining of this chapter will present the research questions explored in this thesis, provide a background on AI techniques, and discuss electrified powertrain modeling. Finally, the thesis outline and its main contributions will be detailed.

1.2 Research questions

This thesis investigates the intersection of AI and electrified vehicle performance, with an emphasis on environmental sustainability, operational efficiency, and system reliability. It poses several research questions that are pivotal for advancing the field of electrified vehicle technology:

1. Optimization of HEV performance: how can AI techniques be employed to optimize fuel economy and passenger comfort in HEVs?

2. Trajectory optimization in BEVs: what role does AI play in harnessing data from interconnected vehicles to optimize the longitudinal trajectory, thereby improving traffic flow and reducing energy usage in BEVs?

3. Battery state-of-health estimation: how can AI algorithms accurately estimate the SoH of BEV batteries, and what are the implications for battery management and lifespan?

As scientific research has increasingly focused on these technologies, there has been a shift in this thesis from HEVs to BEVs, aligning with regulatory trends over the past five years. Initially, the research delved into the energy management control problem in HEVs, aiming to enhance fuel economy and

thus mitigate carbon dioxide emissions, without sacrificing vehicle performance. Subsequently, the focus transitioned to BEVs, exploring their performance regarding battery status and their effectiveness in connected driving scenarios.

A blend of AI strategies are implemented to tackle these questions. For energy management of HEVs, this thesis examines both reinforcement learning (RL) with tabular Q-learning and supervised learning with long short-term memory (LSTM) networks. Passenger cars and commercial light-dutyheavy-duty vehicles are modeleled and analyzed, respectively. The methodology employs RL in passenger cars to account for scenarios where the driver may not set the GPS, while utilizing supervised learning for light-commercial vehicles engaged in goods transportation, which operate with predefined destination sequences. In addressing the optimization of BEV trajectories, we evaluate the use of gated recurrent unit (GRU) neural networks within a cooperative adaptive cruise control (CACC) system. To achieve broader applicability, the approach later transitions to a RL framework, explicitly incorporating sensor and communication uncertainties. Lastly, the estimation of battery SoH in BEVs leverages supervised learning on experimental datasets to train predictive models.

Each research question introduces a distinct challenge, requiring tailored AI-based solutions to advance the state of electric vehicle technology within the context of sustainability goals.

1.3 Background

The background of this thesis is structured around two critical pillars: AI techniques and electrified powertrain modeling. It begins with a comprehensive overview of AI, detailing its methodologies, advancements, and wide-ranging applicability. This part establishes a common background for integrating AI into complex system analyses, particularly in the realm of electric vehicle technologies. Following this, the narrative shifts to focus exclusively on electrified powertrain systems. Here, the discussion centers on modeling approaches, aiming to enhance the understanding of how these systems can be simulated and predicted, which is pivotal for applying AI to improve efficiency and foster innovation in electrified powertrains. This section is designed to prepare the

ground for a deeper exploration of the transformative role AI might play in advancing electrified vehicle technologies.

1.3.1 Artifical Intelligence Techniques

Machine learning (ML) is a subset of artificial intelligence (AI), defined as a computer's ability to learn from data[1]. ML includes any algorithmic approach where machines can learn from and make predictions or decisions based on data. This includes a variety of techniques. In this work, the machine learning algorithms that were investigated include linear regression (LR), k-nearest neighbors (KNN), classification and regression tree (CART), random forest (RF), support vector machine (SVM), and dense neural network (DNN). Deep learning, a subset of machine learning, is instead specialized in interpreting complex patterns in data through neural networks. These networks are structured in layers, including input, hidden, and output layers, and are designed to capture the relationship between input and output data.

For the scope of this work, we exclusively focused on types of recurrent neural networks (RNNs) due to their inherent strengths in modeling sequential data. This type of network features a memory cell which manages the information coming from the previous and current inputs to generate the current output(s). To handle vanishing gradient problems related to long–time series[2], two particular types of RNNs were selected namely, LSTM and GRU neural networks. LSTMs have a complex architecture with multiple gates (input (2.6), output (2.8), and forget (2.5) gates), mainly designed to let information pass selectively: their function is to establish which information must be preserved and which one is to be discarded to achieve the learning goal. GRUs combine the input and forget gates into a single update gate and also merge the cell state and hidden state, leading to a more streamlined architecture. This simplicity often leads to faster training times compared to LSTMs, albeit with a

[1]The present section refers to [6, 7].

[2]Vanishing gradient problems describe an issue occurring during backpropagation when the gradient (that is the signal used to update weights), becomes progressively smaller as it is passed back through the network's layers. Due to repeated multiplication, the gradient can shrink to the point where it barely influences the early layers of the network, resulting in negligible weight adjustments. This effect is more acute in networks with extensive depth or those processing long sequences.

potential trade-off in performance, especially for tasks requiring complex data modeling. Both LSTM and GRU are widely used in various applications where sequential data is prominent. This includes different tasks such as language modeling, speech recognition, text generation, and time series analysis. They excel in these areas because of their ability to capture temporal dynamics and dependencies in data, something that is challenging for other neural network architectures.

Both ML and deep learning (DL) can be divided into three main categories that are supervised learning, unsupervised learning and reinforcement learning[3]. Table 1.1 provides a summary of machine learning categories.

Table 1.1 Schematic Overview of Machine Learning Categories

Category	**Learning Technique**	**Dataset**	**Label Required**	**Main Purpose**
Supervised	Directed	Labeled	Yes	Prediction
Unsupervised	Self-organized	Unlabeled	No	Pattern Discovery
Reinforcement	Trial and Error	Interaction Data	No	Decision Making

In this thesis, our primary focus is on supervised learning, encompassing both machine learning and deep learning techniques, along with reinforcement learning. However, for conciseness and to assist the reader in gaining a general, albeit not exhaustive, understanding, we will provide a brief overview of supervised, unsupervised, and reinforcement learning. Subsequently, in each chapter, the techniques employed will be discussed in greater detail. Figure 1.1 provides a broad overview of AI techniques to assist the reader in grasping the foundational concepts, while Figure 1.2 specifically outlines the algorithms utilized within this thesis.

[3]Part of this section has been extracted from: [8–10]

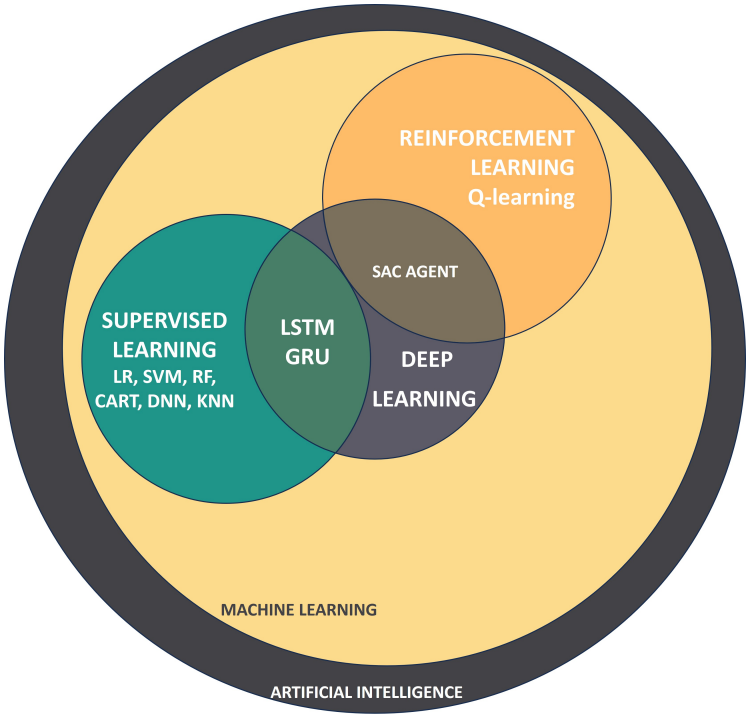


Fig. 1.2 Scheme of the considered AI algorithms.

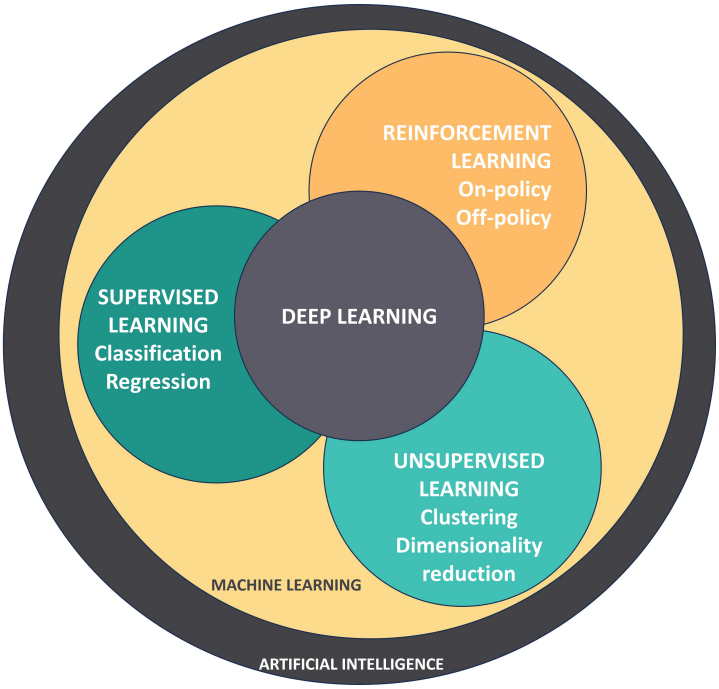


Fig. 1.1 Overview of AI techniques [1].

Supervised Learning

Supervised learning relies on a dataset that has been previously labeled. This dataset is divided into three parts: training, testing, and validation, which are used to train and subsequently validate the model's accuracy. Once trained, the supervised learning model can then be applied to make predictions on new, unlabeled data i.e., the testing dataset. In turn, supervised learning can be divided into two further categories that are regression and classification. The former predicts continuous numerical values whereas the latter predicts discrete labels. Specifically, in regression the model learns to fit a line or curve to the input data points, minimizing the difference between the predicted and actual values. Conversely, in classification, the model learns to draw boundaries between different classes in the input space. The boundaries can be linear (as in logistic regression) or complex (as in neural networks). In its most abstract form, supervised learning working principle can be expressed as follows:

$$y = f(x; \theta) + \epsilon \tag{1.1}$$

where y represents the target variable that the model is trying to predict, x denotes the input features, f(x; θ) is the function that maps input features to the predicted output, θ represents the weights of the model and ϵ is the error term.

Unsupervised Learning

Unsupervised learning is a type of machine learning where the algorithm is trained on a dataset without predefined target values. The primary goal of unsupervised learning is to discover underlying patterns in the data that are not immediately apparent. This can be conceptualized as finding a function f that transforms input data X into a new representation Z that captures some significant features of the data so that Z = f(X). One particular type of unsupervised learning is clustering where data with similar characteristics are aggregated.

Reinforcement Learning

Reinforcement learning is based on the interaction of an agent with its environment and the subsequent rewards or penalties it receives. This approach focus on learning through trial and error and prioritizing long–term rewards. In the context of RL, two primary methods are often discussed: on–policy and off–policy methods. These terms refer to how the learning agent acquires knowledge about the environment. On-policy learning methods update the value of a policy based on the actions taken by the same policy, directly learning from the decisions it makes. Off-policy learning methods, in contrast, learn an optimal policy using data generated from different policies, allowing them to benefit from actions not taken during current policy exploration. In detail, an off–policy method separates the learning process from the policy being evaluated. This allows the agent to learn from experiences generated by following a different, more exploratory policy. Such an approach enables the agent to extensively explore the environment in the early stages of training. As the agent gathers more knowledge, it gradually shifts towards exploiting known rewards. This balance between exploration (seeking new, potentially better rewards) and exploitation (using known rewards) is managed by an ϵ—greedy strategy [11]. In this work, the ϵ—greedy approach is implemented as an exponential decay function, which systematically reduces the likelihood of exploration over time.

$$\begin{aligned} Q(x_{\mathrm{k}}, u_{\mathrm{k}}) \leftarrow\ & Q(x_{\mathrm{k}}, u_{\mathrm{k}}) \\ & + \alpha \left[r_{\mathrm{k+1}} + \gamma \max_{\mathrm{u'}} Q(x_{\mathrm{k+1}}, u') - Q(x_{\mathrm{k}}, u_{\mathrm{k}}) \right] \end{aligned} \tag{1.2}$$

where $\mathrm{Q(x_k, u_k)}$ represents the Q–value of state–action pair $\mathrm{(x_k, u_k)}$, the update rule adjusts this value based on the immediate reward $\mathrm{r_{k+1}}$, the learning rate α, and the discounted future reward $\gamma \mathrm{max_{u'}\, Q(x_{k+1}, u')}$, thereby integrating current experience to refine future action-value estimations. On–policy methods, on the other hand, update the policy based on actions taken during learning, which means that the agent's policy converges with the observed behaviour during training. Specifically, the Q–value is updated using the action actually taken in the next state, $\mathrm{Q(x_{k+1}, u_{k+1})}$, reflecting a learning approach that

assesses and improves upon the policy it employs to make decisions.

$$\begin{aligned} Q(x_{\mathrm{k}}, u_{\mathrm{k}}) \leftarrow & Q(x_{\mathrm{k}}, u_{\mathrm{k}}) \\ & + \alpha \left[r_{\mathrm{k+1}} + \gamma Q(x_{\mathrm{k+1}}, u_{\mathrm{k+1}}) - Q(x_{\mathrm{k}}, u_{\mathrm{k}}) \right] \end{aligned} \tag{1.3}$$

The rewards an agent receives are dependent on its control actions. These rewards are designed to reflect the contribution of each action towards achieving the ultimate goal.

In this thesis, we focus on off-policy methods tailored to the control problem at hand. For managing the energy of HEVs, we employed a Q-learning technique due to the multi-objective nature of the function. This choice was predicated on the belief that a simpler algorithm could achieve convergence despite the problem's complex, multi-objective constraints. Conversely, for CACC management, facing sensor and communication uncertainties, we opted for a soft actor-critic (SAC) agent. This selection was made because of its stochastic nature, which offers robustness against uncertainties.

1.3.2 Electrified Vehicle Powertrains

Unlike traditional vehicles powered exclusively by internal combustion engines and dependent on fossil fuels, electrified powertrains harness electricity, with different levels of electrification, from various sources and in different forms to drive the vehicle. The main types of electrified powertrains include BEVs, HEVs, PHEVs[4], and Fuel Cell Electric Vehicles (FCEVs). In this thesis, our focus will be primarily on HEVs and BEVs, with a specific consideration of PHEVs during their charge sustaining phase[5].

From a technical perspective, hybrid electric vehicles merge the features of traditional ICE-powered and fully electric cars. However, their complex

[4]The distinction between PHEVs and HEVs primarily lies in their charging and operation. PHEVs can be externally charged and operate on electric power alone before engaging the combustion engine, whereas HEVs cannot be externally charged and use a combination of engine power and regenerative braking to charge their battery.

[5]PHEV operation encompasses two primary modes: *charge depleting*, wherein the vehicle prioritizes electric power from the battery for propulsion until reaching a predefined low state of charge, and *charge sustaining*, during which the vehicle's management system regulates the use of the internal combustion engine and electric motor in tandem to prevent further battery depletion, ensuring the battery's state of charge is maintained.

design demands advanced control strategies to efficiently distribute energy between their various power sources. HEVs are categorized in multiple ways: they can be plug-in hybrids (PHEVs) that charge from external sources, or non-plug-in hybrids (HEVs). Additionally, their mechanical propulsion can be configured differently: in a series configuration, where only the electric motor drives the vehicle; a parallel configuration, where both the motor and the internal combustion engine contribute to propulsion; a complex configuration, which allows for switching between series and parallel modes; or utilizing a planetary gearset, a mechanism that facilitates variable power splits between the engine and electric motor for optimized efficiency [12, 13].

Focusing on the parallel configuration, this can be further divided based on the location of the electric motor: P0 if it's belt-connected to the engine, P1 if directly attached to the engine's crankshaft, P2 if situated between the engine and transmission, P3 if it's post-transmission, and P4 if it's on the opposite axle from the engine. These vehicles can operate in four distinct modes: purely thermal (engine-driven), purely electric (motor-driven), power-split (combined engine and motor), or battery charging (engine propels the vehicle and charges the battery simultaneously). Given their complex nature, control algorithms are necessary to select the best operating mode.

On the other hand, BEVs are powered entirely by electric batteries and electric motors and the vehicle is charged through an external power source.

Modeling

Equations common to the powertrains analyzed in this work are presented in this subsection, while equations specific to each powertrain are detailed in their respective paragraphs [6]. The transmission output torque was computed considering the vehicle resistance forces including aerodynamic, rolling, and inertial contributions as a function of the vehicle's speed v and acceleration a.

$$T_{\text{out}} = \left(mg\epsilon_{\text{r}} \cos\theta + 0.5\rho c_x A_{\text{f}} v^2 + mg \sin\theta + ma\right) r_{\text{w}} \tag{1.4}$$

Here, m, ϵ_{r}, c_x and A_{f} refer to the vehicle mass, rolling resistance coefficient, aerodynamic drag coefficient and frontal area; r_{w} is the wheel's radius; g is the

[6] The present section refers to [12, 13]

gravitational acceleration; and ρ is the air density. The main components such as the electric machine and engine were modelled using quasi-static look-up tables. The battery was modeled using an equivalent circuit model, with an internal resistance R_{b}. Hence, the battery current i_{b} and the SOC dynamics $\dot{\sigma}$ were evaluated as follows:

$$i_{\mathrm{b}} = \frac{v_{\mathrm{oc}} - \sqrt{v_{\mathrm{oc}}^2 - 4R_{\mathrm{b}}P_{\mathrm{b}}}}{2R_{\mathrm{b}}}, \tag{1.5}$$

$$\dot{\sigma} = \frac{i_b}{C_b}. \tag{1.6}$$

Battery Electric Vehicles (BEVs)

The BEVs considered for this study were modelled using a road load approach. The vehicle SOC variation along with the vehicle energy consumption have been evaluated as a function of vehicle velocity and acceleration according to (1.7) — (1.6). The electric motor (EM) torque T_{EM} was computed as a function of the transmission output torque T_{out}, the transmission efficiency η and the gear ratio τ:

$$T_{\mathrm{EM}} = \frac{T_{\mathrm{out}}}{\eta^{\mathrm{sign}(T_{\mathrm{out}})}\tau}, \tag{1.7}$$

where the sign of T_{out} is used to correctly discriminate between braking and traction events. The battery power request P_{b} was computed by taking into account the EM power P_{EM} (which also includes its losses) and the auxiliaries' power P_{aux}.

$$P_{\mathrm{b}} = P_{\mathrm{EM}} + P_{\mathrm{aux}}. \tag{1.8}$$

Hybrid ELectric Vehicles (HEVs)

The HEV considered for the purpose of this study is a parallel HEV architecture. The torque wheel $\mathrm{T_w}$ was computed as a function of the engine torque ($\mathrm{T_{eng}}$ or $\mathrm{T_{ICE}}$) and the electric motor torque ($\mathrm{T_{EM}}$) considering the driveline.

$$T_{\mathrm{w}} = T_{\mathrm{eng}}\tau_{\mathrm{g(n_{gear})}}\tau_{\mathrm{f}}\eta_{\mathrm{g}}\eta_{\mathrm{f}} + T_{\mathrm{EM}}\tau_{\mathrm{r}}\eta_{\mathrm{r}}^{\mathrm{sign(T_{EM})}} \tag{1.9}$$

where the subscripts g, f and r refer to the gearbox, the front and rear axle differential efficiencies (η) or transmission ratios (τ), depending on the case. By accounting for the sign of the EM torque as a power factor in the rear differential efficiency, both propulsion and regenerative cases are considered. The gear shifting schedule was determined using a rule-based strategy tied to the road speed, aiming to enhance the vehicle's performance, approach real-time vehicle usage, and meet passenger comfort requirements [14]. The battery power ($\mathrm{P_{batt}}$) was in turn computed considering the electric motor power ($\mathrm{P_{EM}}$) and the overall losses of the electric motor ($\mathrm{P_{EM,loss}}$) along with the power related to the auxiliaries ($\mathrm{P_{aux}}$).

$$P_{\mathrm{b}} = P_{\mathrm{EM}} + P_{\mathrm{EM,loss}} + P_{\mathrm{aux}} \tag{1.10}$$

From a technical point of view, HEVs are characterized by complex structure and require sophisticated control strategies to optimize the power split among the energy sources. In addition to the classification based on the type of recharging process they undergo (plug-in or not), further classification is based on their powertrain configuration, which includes series, parallel, or complex systems. Focusing on parallel hybrids, these are further classified by the electric motor's position relative to the engine, known as P0 to P4 configurations [12, 13]. These configurations dictate the vehicle's operational modes – pure thermal, pure electric, power-split, or battery charging – each offering distinct advantages in terms of performance and efficiency.

1.4 Thesis Outline and Contribution

This thesis is structured as follows: Chapter 2 explores AI techniques in energy management for HEVs. Chapter 3 delves into the application of AI techniques to CACC for strings of BEVs, whereas Chapter 4 considers the estimation of battery state-of-health in BEVs. The thesis concludes with Chapter 5, outlining the conclusions and presenting future research directions. A description of each chapter is provided below.

Chapter 2

In this chapter, we investigate the energy management control in HEVs through two distinct test cases. Initially, we explore a commercial light-duty vehicle's architecture, utilizing a LSTM neural network as the control strategy. Our aim was to enhance fuel efficiency while maintaining charge sustainability. Subsequently, we shift our focus to a passenger car architecture, expanding our objectives to include not only fuel economy but also passenger comfort and engine efficiency across a variety of driving conditions, using a RL based approach. Here, a Q-learning algorithm was employed to fine-tune the activation and torque variations of the internal combustion engine related to the passenger comfort indexes. Building on concepts discussed in earlier sections, this methodology applies RL to passenger cars for scenarios lacking GPS input by the driver, in contrast to using supervised learning for light-commercial vehicles engaged in goods transportation with set destinations. The algorithm was subjected to a thorough parameter optimization process to ensure its effectiveness and adaptability to changing driving conditions.

Part of this chapter is based on the following publication(s):

Musa, A.; Anselma, P.G.; Belingardi, G.; Misul, D.A. Energy Management in Hybrid Electric Vehicles: A Q-Learning Solution for Enhanced Drivability and Energy Efficiency. *Energies* 2024, *17*, 62. doi: https://doi.org/10.3390/en17010062.

Anselma, P.G.; Maino, C.; Musa, A. THEO: A Tailored Hybrid Emission Optimizer for the Drivers of Tomorrow. In the Young Researcher TRA Visions 2020 Award, sponsored by ERTRAC for the road mode category, Politecnico di Torino, 1st Place. Available online: https://www.travisions.eu/TRAVisions/young_researcher_results_2020/;jsessionid=70438b8b2153916744aa54d0656b (accessed on 17 February 2024).

Chapter 3

This chapter introduces a deep learning approach coupled with vehicle communication technology and sensors for developing a real-time CACC system. Initially, a GRU architecture is selected and trained using specialized CACC datasets.

These datasets are created based on an optimal control policy, specifically dynamic programming (DP), targeting a battery electric vehicle. DP primarily focuses on optimizing the longitudinal speed trajectory of the following vehicle in CACC to enhance energy efficiency and passenger comfort.

In the second test case, we explore the impact of sensor and communication uncertainties. Here, a decentralized model predictive control (MPC) serves as the benchmark, while a RL based method, i.e. a SAC agent, is employed as the online control algorithm. This selection was made because of its stochastic nature, which offers enhanced robustness against uncertainties.

Part of this chapter is based on the following publications:

Musa, A.; Anselma, P. G.; Spano, M.; Misul, D. A.; Belingardi, G.; Cooperative Adaptive Cruise Control: A Gated Recurrent Unit Approach. 2022 IEEE Transportation Electrification Conference & Expo (ITEC), Anaheim, CA, USA, 2022, pp. 208–213. doi: 10.1109/ITEC53557.2022.9813990.

Under review:

Seifoddini, A.; Azad, A.; Musa, A.; Misul, D. Design of a Decentralized Control Strategy for CACC Systems Accounting for Uncertainties. SAE CO2 Reduction for Transportation Systems Conference, 2024.

Chapter 4

The present chapter investigates the use of machine learning algorithms to estimate the state of health (SOH) of high-voltage batteries in electric vehicles. The analysis is based on open-circuit voltage (OCV) measurements from 12 vehicles with different mileage conditions and focuses on establishing a correlation between the OCV values, the energy stored in the battery, and the battery SOH. The experimental campaign was conducted at the Technical Center of one of our industrial partner. Based on the data at hand and the project requirements, six machine learning algorithms are evaluated and compared, namely linear regression, k-nearest neighbors, support vector machine, random forest, classification and regression tree, and neural network.

This chapter is based on the following publication:

Lelli, E.; Musa, A.; Batista, E.; Misul, D.A.; Belingardi, G. On-Road Experimental Campaign for Machine Learning Based State of Health Estimation of High-Voltage Batteries in Electric Vehicles. *Energies* **2023**, *16*, 4639. doi: https://doi.org/10.3390/en16124639.

Work Not Included in This Thesis

- Maino, C.; Misul, D.; Musa, A., Spessa E. Optimal mesh discretization of the dynamic programming for hybrid electric vehicles. *Applied Energy* 2021, *Volume 292*, 116920. doi: https://doi.org/10.1016/j.apenergy.2021.116920.

- Spano, M.; Musa, A.; Anselma, P. G.; Misul, D. A.; Belingardi, G.; Battery Electric Vehicles Platooning: Assessing Capability of Energy Saving and Passenger Comfort Improvement. 2021 AEIT International Conference on Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE), Torino, Italy, 2021, pp. 1–6. doi: 10.23919/AEITAUTOMOTIVE52815.2021.9662788.

- Spano, M.; Anselma, P. G.; Musa, A.; Misul, D. A.; Belingardi, G.; Optimal Real-Time Velocity Planner of a Battery Electric Vehicle in V2V Driving. 2021 IEEE Transportation Electrification Conference & Expo (ITEC), Chicago, IL, USA, 2021, pp. 194–199. doi: 10.1109/ITEC51675.2021.9490121.

- Capuano, A.; Spano, M.; Musa, A.; Toscano, G.; Misul, D.A. Development of an Adaptive Model Predictive Control for Platooning Safety in Battery Electric Vehicles. *Energies* 2021, *14*, 5291. doi: https://doi.org/10.3390/en14175291.

- Musa, A.; Pipicelli, M.; Spano, M.; Tufano, F.; De Nola, F.; Di Blasio, G.; Gimelli, A.; Misul, D.A.; Toscano, G. A Review of Model Predictive Controls Applied to Advanced Driver-Assistance Systems. *Energies* 2021, *14*, 7974. doi: https://doi.org/10.3390/en14237974.

- Anselma, P.G.; Musa, A.; Maino, C.; Misul, D.; Belingardi, G. Effect of Temperature Distribution on the Predicted Cell Lifetimes for a Plug-In

Hybrid Electric Vehicle Battery Pack. SAE Technical Paper 2022-01-0712,2022 doi: https://doi.org/10.4271/2022-01-0712.

- Musa, A., Miretti, F., and Misul, D. MPC-Based Cooperative Longitudinal Control for Vehicle Strings in a Realistic Driving Environment. SAE Technical Paper 2023-01-0689,2023 doi: https://doi.org/10.4271/2023-01-0689.

Chapter 2

AI techniques for energy management of HEVs

2.1 Introduction

2.1.1 Motivation

The widespread use of vehicles powered by fossil fuels is often recognized as one of the major contributors to climate change, as well as air and noise pollution. In response, governments in various regions are planning to reduce or eliminate the sale of traditional vehicles, acknowledging the need to tackle these environmental issues. Efforts are underway to diversify energy sources, decarbonize fuels, and embrace electrified powertrain solutions. To effectively join this transition, a variety of powertrain technologies and alternative fuels are being explored, each with its unique pros and cons. For instance, hydrogen-based solutions face challenges due to limited infrastructure, making them less viable in the short term. Similarly, the shift to battery-powered vehicles must consider the need to decarbonize energy production systems. In the current scenario, hybrid electric vehicles (HEVs) offer a transitional solution. Numerous online statistical analyses project a 20% growth in the hybrid market over the next five years, with plug-in hybrids leading the trend [9, 4]. HEVs merge the benefits of conventional and fully electric vehicles. However, their complexity

requires advanced control logic for efficient energy management, making them a popular subject in academic research.

2.1.2 Contribution

In this chapter, our investigation focuses on the optimization of energy management controls within HEVs across two distinct case studies[1]. Initially, our research delves into the energy management architecture of a commercial light-duty vehicle, employing a long short-term memory (LSTM) neural network to govern the system. The primary objective was to improve fuel efficiency while preserving battery charge sustainability. Following this, we pivot our attention towards the energy management system of a passenger car, broadening our goals to encompass fuel economy, passenger comfort, and engine efficiency under various driving conditions through a reinforcement learning (RL) strategy. A Q-learning algorithm was utilized to adjust the internal combustion engine's activation and torque fluctuations in response to passenger comfort metrics[2].

This study expands the discussion by applying RL to passenger vehicles in situations without GPS input from the driver, diverging from supervised learning methods traditionally applied to light and heavy commercial vehicles (HDVs) with predetermined routes. The algorithm underwent extensive parameter optimization to guarantee its efficiency and flexibility in response to fluctuating driving scenarios.

2.1.3 Related works

Different control strategies have been developed for HEVs [13, 18, 19], including rule-based [20–25], optimization-based [26–31, 14], data-driven [32–44], and reinforcement learning approaches [45–56]. Rule-based controllers, while com-

[1] Part of this chapter has been extracted from [15–17].

[2] As a remark, Q-learning is preferred over DP for its ability to handle larger state spaces and its applicability in real-time scenarios, which is not feasible with DP. However, it's important to note that while Q-learning is less susceptible to the curse of dimensionality compared to DP, challenges remain as model complexity increases. Our study uses comparable models for both RL and DP to ensure direct comparability, recognizing that exploring different model complexities with advanced RL techniques could be a valuable direction for future research.

monly used, need extensive calibration and may underperform in scenarios that differ from the calibration ones. In classical approaches, optimizing an objective function is crucial for reducing carbon dioxide emissions, which are directly related to fuel economy. The optimization may involve either a sole focus on carbon dioxide emissions or a weighted average of both carbon dioxide and other pollutant emissions, all while maintaining a charge-sustaining operation [14, 20–59]. Still up for debate, though, is the necessity for a real-time algorithm in this field that can operate in a variety of driving conditions and meet one or more of these objectives, particularly when developing customized controllers. In recent years, approaches based on reinforcement learning have proven to be an appealing option for handling these complex and non-linear control issues. As a matter of fact, the real-time use of the RL agent eliminates the computational costs associated with optimization-based strategies as well as the performance degradation of rule-based solutions when utilized in driving scenarios that differ from those used for calibration [53]. In the realm of HEV energy management, RL has emerged as a promising strategy for optimizing the distribution of power among onboard energy sources. This optimization aims to improve fuel efficiency while complying with the constraints of vehicular components and the battery state-of-charge (SOC) [45–50, 52, 51, 54, 55, 59, 56, 53].

RL techniques are categorized into value-based and policy-based methods. The former leverages acquired knowledge to inform decision-making in given states, while the latter directly models the policy function that maps state-action pairs [11, 60, 6]. Advanced RL algorithms, including Q-learning, deep Q-learning, double Q-learning, and actor-critic methods, differ in complexity and computational demands. Q-learning agents refine their control policies through continuous interaction with the environment, applying the principle of exploration-exploitation. Deep Q-learning uses neural networks to approximate the Q-values of actions for each state, requiring careful calibration to avoid overestimation. To mitigate this, double Q-learning introduces a dual neural network architecture comprising online and target networks for action selection and value estimation, respectively [54]. Actor-critic models utilize one network for policy-based action selection (actor) and another for action outcome evaluation and value function estimation (critic) [52].

For example, Xu et al. applied Q-learning for real-time control to balance fuel economy and charge sustainability [55]. Chen et al. developed an energy

management control that integrates model predictive control with double Q-learning, targeting improved fuel economy and charge maintenance in power-split PHEVs without compromising comfort and drivability [56]. Similarly, Han et al. implemented a double-deep Q-learning algorithm, which significantly enhanced fuel efficiency while keeping the battery SOC near a desired level, showing a 7% improvement over traditional deep Q-learning and achieving nearly 93% efficiency of dynamic programming [54].

Despite extensive research, the integration of drivability and ride comfort into HEV energy management through RL remains underexplored. Studies often focus on traditional metrics like fuel consumption, battery state-of-health, and charge-sustaining SOC performance, with less attention to the broader implications on ride quality and comfort [51–55]. This oversight suggests an area for further investigation within the application of RL techniques to HEV energy management.

2.1.4 Outline

The chapter is structured as follows: Section 2.2 briefly outlines the problem under investigation. Section 2.3 discusses the algorithms utilized. Finally, Sections 2.4–2.5 describe the selected test cases.

2.1.5 Notation

This paragraph outlines the principal symbols and their domains for dynamic programming and Q-Learning algorithms. We define the state vector as $\mathrm{x(t)} \in \mathrm{X} \subset \mathbb{R}^{\mathrm{n}}$, where x(t) is subject to a set of inequality constraints $\mathrm{N(x(t), t)} \leq 0$. The vector x(t) represents the state at time t, with each state lying within an n-dimensional real space. The control vector is denoted by $\mathrm{u(t)} \in \mathrm{U} \subset \mathbb{R}^{\mathrm{m}}$, indicating the action taken at time t within an m-dimensional control space. The time variable is represented by $\mathrm{t} \in \mathbb{R}$, with $\mathrm{t_0}$ and $\mathrm{x_0}$ as the initial conditions. The control set U is assumed to be a closed subset of $\mathbb{R}^{\mathrm{m}}$ that varies over time. The learning rate $\alpha \in [0, 1]$ and discount factor $\gamma \in [0, 1]$ are real numbers that control the learning process. This notation forms the basis for the mathematical formulation of our algorithms.

2.2 Problem Formulation

The control system considered in the present research work takes the form[3]:

$$\dot{x} = f(t, x(t), u(t)), \quad x(t_0) = x_0 \tag{2.1}$$

The objective is to find the optimal control law that minimizes a cost functionals of the form [61]:

$$J(t_0, x_0, t_f, u) := \int_{t_0}^{t_f} L(t, x(t), u(t))dt + K(t_f, x_f) \tag{2.2}$$

where t_f and $x_f := x(t_f)$ are the terminal time and state, L is the running cost whose domain is $\mathbb{R} \times X \times U \to \mathbb{R}$ and K is the terminal cost whose domain is $\mathbb{R} \times X \to \mathbb{R}$. In the context of HEVs, the running cost is usually related to fuel consumption whereas the terminal constraint is designed to account for the charge sustainability requirement. The minimization of J is typically subject to multiple constraints, usually associated with physical limitations of powertrain components, the energy stored in the battery, and requirements related to charge sustainability. Specifically, the charge-sustaining constraint ensures that the vehicle keeps its electrical charge without an external source throughout a given driving mission

$$x(t_f) = x(t_0), \tag{2.3}$$

typically with a certain tolerance to account for practical considerations and to simply maintain energy within predefined boundaries [13]:

$$K(t_f, x_f) = \phi \left[x(t_f) - x(t_0) \right]. \tag{2.4}$$

In addition, usually, the battery SOC is bound within a certain range to avoid premature ageing phenomena. To include drivability and ride quality requirements, we added a component to the running cost representative of the frequency of ICE de/activations.

[3]The present section refers to [13, 12, 61].

2.3 Algorithms

2.3.1 Benchmark Algorithm

We selected dynamic programming (DP), a numerical method for solving multi-stage decision-making problems, to solve this constrained finite-horizon control problem and we used it as a benchmark against which to compare the performance of the proposed algorithm. Given that DP is a well-established approach commonly employed in HEV energy management control problems, we have chosen to omit a formal definition within this study, reporting here just a summary of the algorithm itself (Algorithm 1). We direct interested readers to refer to established sources for a comprehensive understanding [62–64, 13, 61, 57]. We assessed the performance of the DP algorithm on different objective functions, namely:

1. A classical approach where fuel economy and charge sustainability are considered;
2. A trade-off between fuel economy and drivability/comfort requirements ensuring charge sustaining operation [14].

The primary aim was to formulate a control problem that closely emulates real-world driving priorities, thereby creating a benchmark akin to a high-fidelity scenario.

Algorithm 1 Dynamic programming with terminal constraint

Backward Phase:
Initialize $V(t_f, x)$ for all states x at the final time step t_f
for $t \leftarrow t_f - 1$ **downto** 1 **do**
  for each state x at time t **do**
    Consider all possible control u in state x at time t
    Calculate the expected value $V(t, x, u)$ associated with each action
    Apply constraints to eliminate infeasible actions
    Update the value function $V(t, x)$ for state x at time t based on the calculated values
  end for
end for
Forward Phase:
Initialize the optimal policy $\pi(t, x)$ for all states x and time steps t
for $t \leftarrow 1$ **to** $t_f - 1$ **do**
  for each state x at time t **do**
    Choose the control u^* that maximizes $V(t, x, u)$
    Set the policy $\pi(t, x) = u^*$
  end for
end for
Output: Optimal value function $V(t, x)$ and policy $\pi(t, x)$ for all time steps t and states x

2.3.2 LSTM

LSTMs have a complex architecture with multiple gates (input (2.6), output (2.8), and forget (2.5) gates), mainly designed to let information pass selectively: their function is to establish which information must be preserved and which one is to be discarded to achieve the learning goal.

$$f_{\mathrm{t}} = \sigma\left(W_{\mathrm{f}} v_{\mathrm{t}} + U_{\mathrm{f}} h_{\mathrm{t}-1} + b_{\mathrm{f}}\right) \tag{2.5}$$

$$i_{\mathrm{t}} = \sigma\left(W_{\mathrm{i}} v_{\mathrm{t}} + U_{\mathrm{i}} h_{\mathrm{t}-1} + b_{\mathrm{i}}\right) \tilde{C}_{\mathrm{t}} \tag{2.6}$$

$$C_{\mathrm{t}} = f_{\mathrm{t}} \times C_{\mathrm{t}-1} + i_{\mathrm{t}} \times \tilde{C}_{\mathrm{t}} \tag{2.7}$$

$$o_{\mathrm{t}} = \sigma\left(W_{\mathrm{o}} v_{\mathrm{t}} + U_{\mathrm{i}} h_{\mathrm{t}-1} + b_{\mathrm{o}}\right) h_{\mathrm{t}} \tag{2.8}$$

Specifically, the forget gate (f_{t}) decides what information is discarded from the cell state. The input gate (i_{t}) and the input state update ($\tilde{C}_{\mathrm{t}}$) together decide what new information is added to the cell state. Finally, the output gate (o_{t}) determines the next hidden state (h_{t}) that is passed to the network.

Eq. 2.7 represents the candidate cell state, which contributes to the update of the cell state (C_{t}) alongside the input and forget gates.

$$h_{\mathrm{t}} = o_{\mathrm{t}} \times \tanh\left(C_{\mathrm{t}}\right) \tag{2.9}$$

Eq. (2.9) defines how the output gate (o_{t}) and the updated cell state (C_{t}) are used to compute the new hidden state (h_{t}), which is then passed to the next time step or used for generating the output of the LSTM unit. This process allows LSTMs to retain or discard information over long sequences, making them particularly effective for tasks involving long-term dependencies.

2.3.3 Reinforcement Learning based approach

We selected an off-policy temporal difference control algorithm to solve the proposed control problem, namely Q-learning. An off-policy method decouples the learning policy from the policy being evaluated, allowing the agent to learn from experiences generated by following a different exploratory policy (see Section 1.3.1) following the update rule detailed in Eq. (1.2). In the Q–learning framework, a transition in the value function - which indicates the updated value in the Q–table for a specific state-action pair as a result of each reward - is stored in a table. By identifying actions associated with the highest values for each state variable combination, a lookup table of rules can be generated for real-time application. The Q–table is updated by adding a product of the learning rate (α) and the temporal difference error. This error represents the difference between the current Q–value and the combined value of the immediate reward and the discounted maximum Q–value of the subsequent state, with the discount factor (γ). This update process, as detailed in Eq. (1.2), allows the agent to iteratively refine the Q–values based on observed rewards,

state transitions, and potential future rewards, thereby progressively improving its policy. Algorithm (2) outlines the primary structure of this algorithm.

Algorithm 2 Tabular Q–learning

1: Initialize the Q–values for all state–action pairs in a table form
2: Define the set of allowable actions u(x) for each state x
3: Initialize the current state s and choose an initial action u using an exploration strategy (e.g., epsilon-greedy)
4: **for** N_{episodes} **do**
5: Take action u, observe the next state x' and receive a reward r
6: Update the Q_{value} for the current state–action pair using the temporal difference learning rule
7: Choose the next action u' using a policy derived from the Q_{value} and the allowable actions for state x'
8: Set $x = x'$ and $u = u'$
9: **end for**
10: Use the trained Q_{value} to make decisions in the environment

The decision variable, u, is chosen to be the ICE torque. Conversely, the state vector, x, comprises the battery SOC, the required power at the wheels, and the ICE torque. The inclusion of ICE torque within the state vector is critical for the continuous monitoring and regulation of its rate of change. This ensures the consistent and gradual modulation of the controlled torque over time and therefore the comfort of the ride. The reward was designed to include three main components: fuel consumption $\mathrm{m_f}$, the frequency of ICE de/activations ($\mathrm{x_3} < 0 \wedge \mathrm{u} > 0$), and battery SOC charge sustainability ($\mathrm{x_{1,ref}} - \mathrm{x_1}$).

$$r_t = c_1 - [c_2 \cdot m_f + c_3 \cdot |x_{1,\text{ref}} - x_1| + c_4 \cdot (x_3 < 0 \wedge u > 0)] \tag{2.10}$$

The variables $\mathrm{x_1}$ and $\mathrm{x_3}$ represent the first and third state variables, respectively. The weights of each term of the reward function ($\mathrm{c_2}, \mathrm{c_3}, \mathrm{c_4}$) were properly adjusted to achieve the best compromise between fuel economy, charge-sustainability, and frequency of ICE de/activations. $\mathrm{c_1}$ is a non-negative constant introduced to limit numerical problems during the learning process. In addition, local constraints were imposed on state, control variables, and all

the intermediate variables to compute them so as to guarantee the functioning of the vehicle's main components:

$$SOC_{min} \leq SOC(t) \leq SOC_{max} \tag{2.11a}$$

$$P_{batt,min} \leq P_{batt}(t) \leq P_{batt,max} \tag{2.11b}$$

$$T_{v,min} \leq T_v(t) \leq T_{v,max} \tag{2.11c}$$

$$\omega_{v,min} \leq \omega_v(t) \leq \omega_{v,max}, v = ICE, EM \tag{2.11d}$$

Table 2.1 summarizes the parameters and configuration setup of the proposed algorithm.

Table 2.1 Experiment configuration and parameters for tabular Q-learning.

Parameter	**Value**
Learning Rate α	0.9
Discount Factor γ	0.99
ϵ greedy law	Exponential decay
Action(s)	$\{T_{ICE}\}$
State(s)	{SOC, P_w, T_{ICE}}
Reward Function	$c_1 -$ $[c_2 \cdot m_f + c_3 \cdot \lvert x_{1,\text{ref}} - x_1 \rvert + c_4 \cdot (x_3 < 0 \wedge u > 0)]$

The Q–table was initialized as a three-dimensional array, with its elements individually drawn from a normal distribution with a mean of k_1 and a standard deviation of k_2. This configuration was optimized offline for optimal performance.

In modeling the pure electric operation, a hypothetical scenario was created where the ICE torque was assigned a negative value, thus $T_{ICE} = -z$ denoting electric-only mode. For acceleration, a maximum torque change of 80 Nm was allowed within a 1 s sampling period, and for braking, a maximum change of -100 Nm was permitted, ensuring compliance with the defined ride quality standard.

From an algorithmic point of view, the termination condition was established based on a predetermined number of episodes, whereas the early stopping

criteria focused on the evaluation of the cumulative reward and on the value of the cumulative discounted return. As a remark, the cumulative reward represents the total sum of the rewards obtained by the agent during the entire learning process without applying any discount.

$$R = \sum_{t=0}^{T} r_t \tag{2.12}$$

By measuring the cumulative reward, it is possible to evaluate the overall performance and see if it improves over time. On the other hand, discounted cumulative reward calculates the sum of discounted rewards over time, using a discount factor, represented by gamma.

$$R_\gamma = \sum_{t=0}^{T} \gamma^t r_t \tag{2.13}$$

The discount factor reduces the importance of future rewards compared to immediate ones. Consequently, while considered as a potential metric for evaluating the algorithm's performance and guiding early termination, it was not prioritized as the primary criterion for such termination.

Furthermore, a validation strategy solely focused on exploitation was conducted every 500 episodes to assess the agent's overall performance and evaluate the learning phase, and the Q–table was saved for testing purposes. The overall algorithmic system design scheme is depicted in Figure 2.1.

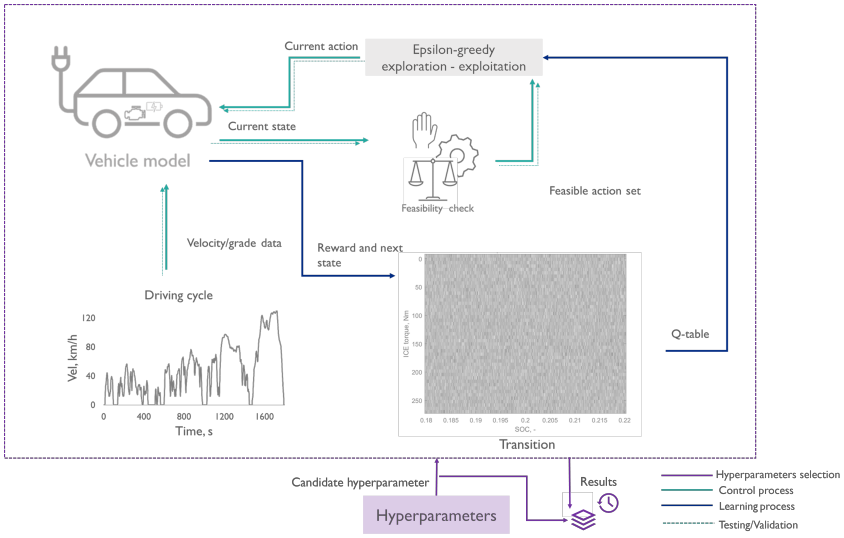


Fig. 2.1 Q–learning design scheme.

2.4 Test Case 1

This section delves into a specific control problem within the domain of fleet management, focusing on light-duty commercial vehicles used in freight or goods transportation. This study hinges on the fundamental assumption that well-defined routes are available for these vehicles. This predictability in their driving cycles is a key aspect, as it allows for the collection of precise data on their operational patterns. Leveraging this data, particularly GPS information and details about driving missions, the research addresses the control problem through a supervised learning methodology. To effectively model and predict the dynamic behaviors of these vehicles, a specialized form of recurrent neural network, namely LSTM, is employed. This choice is driven by LSTM's proven capability in handling time-series data, which is particularly relevant in the context of vehicle route patterns and driving cycles. The approach aims to optimize fleet operations, enhance efficiency, and potentially reduce operational costs by predicting and managing vehicle behaviors more effectively[4].

[4]Part of this research, realized in collaboration with two other researchers from the Interdepartmental Center of Automotive Research and Sustainable Mobility, has been showcased at the Travision2020 competition, road section [16].

2.4.1 Evaluation Metrics

In this initial case study, fuel economy was chosen as the sole metric for evaluation. For the sake of simplicity, no adjustments were made to the final fuel consumption values to account for deviations from the target SOC of the battery, provided that these deviations fell within a 5% tolerance range. A SOC window ranging from 0.4 to 0.8 was established, with an initial SOC set at 0.6.

2.4.2 Comparison Assumptions

The proposed algorithm has been trained and validated on WLTP driving cycles and further tested under real-driving conditions. In this initial test case, the goal was to assess the primary challenges and critical issues associated with this type of approach. As previously clarified, in this initial phase, the energy management control problem was considered as a classification problem and addressed through a supervised learning approach. A benchmark algorithm, DP, optimized for solving a simpler control problem compared to those encountered later, was used to build the datasets for the training phase. This benchmark specifically aimed at minimizing fuel consumption to maintain a targeted SOC for the battery, considering the battery SOC as the state variable, and gear number (GN) and power split /power flow (PF) as the control variables.

Concerning the neural network, its structural details are provided in Table2.2, along with the parameters used during optimization. Meanwhile, the set of input features is detailed in 2.3.

Table 2.2 LSTM architecture and settings.

Layer (type)	**Units**	**Settings**	
LSTM	m	Dropout rate	0.2
Dropout	–	LSTM's Unit values range	128–32
LSTM	m/2	Training	WLTP
Dropout	–	Loss function	Cross entropy
Dense	m/4	Optimizer	Adam

Table 2.3 LSTM input features.

Variable	**Formula**	**Unit**
Actual Velocity (V)	V	km/h
Actual Velocity Variation (ΔV)	$V_t - V_{t-1}$	km/h
Traction/Braking (T/B)	$\begin{cases} 1 & \text{if } \Delta V > 0 \\ 0 & \text{if } \Delta V < 0 \end{cases}$	Boolean
Share of Idle Time (S_{idle})	$\left(\frac{T_{\text{idle}}}{T_{\text{idle}}}\right) \times 100$	%
Gear Number (GN)	GN	-
Battery SOC (SOC_{actual})	SOC_{actual}	%
Battery SOC Variation (ΔSOC)	$SOC_{actual,t} - SOC_{actual,t-1}$	%
Difference in SOC (ΔSOC_{diff})	$SOC_{actual} - SOC^*$	%
Share of Mission Time Left ($S_{mission}$)	$\left(\frac{T_{\text{left}}}{T_{\text{mission}}}\right) \times 100$	%

2.4.3 Vehicle Model

The controller, based on LSTM networks, has been tested considering a P2 parellel HEV architecture. This architecture was developed following a backward-facing modeling strategy and the main components were modeled according to

a map based approach as detailed in Section 1.3.2. The model incorporates an engine with a 5L displacement, an electric motor rated at 125 kW, and a battery capacity of 10 kWh. The vehicle is equipped with an ICE that is diesel-powered, featuring a maximum power output of 132 kW. It has a total mass of 7360 kg, a frontal area of 6.5 m^2, a drag coefficient of 0.703, and tires with a radius of 0.3658 meters. It is important to note that the vehicle layout described here does not correspond to any specific model currently available in the market. Instead, a generic high-performance vehicle design was used. This approach was chosen to avoid the limitations associated with tailoring the system to the precise specifications of existing vehicle powertrains. The primary objective was to qualitatively evaluate the performance of LSTM networks in managing the HEV energy management control problem, rather than optimizing the controller for a specific application. The energy management system algorithm for parallel HEVs selects the gear number (GN) and power-flow (PF) mode at each driving mission step. Seven PFs are considered: 1) pure electric (0% ICE, 100% EM), 2) pure thermal (100% ICE, 0% EM), 3) power-split 25% (75% ICE, 25% EM), 4) power-split 50% (50% ICE, 50% EM), 5) power-split 75% (25% ICE, 75% EM), 6) battery charging 50% (150% ICE, -50% EM), and 7) battery charging 100% (200% ICE, -100% EM).

The vehicle model, consisting of analytical equations, simulates power components (ICE and EM) using operational maps (torque and speed) and an internal resistance model for the battery. Vehicle auxiliary loads are represented by constant power demand, and road resistance forces are calculated using experimental road load coefficients (See Section 1.3.2).

2.4.4 Results

Figures 2.2 and 2.3 illustrate the trends in battery SOC and power distribution, respectively. The DP method accounts for a fuel economy of 4.2L/100km, compared to 4.13L/100km for the LSTM method, which achieves a slightly lower final battery SOC. As previously mentioned, since the difference in the final SOC is less than 5%, no adjustments were made to the final fuel consumption figures. Overall, the LSTM demonstrates good learning capabilities in comparison to the DP algorithm. However, for its application across varied driving scenarios

and potential user differences, it necessitates substantial training and extensive data collection.

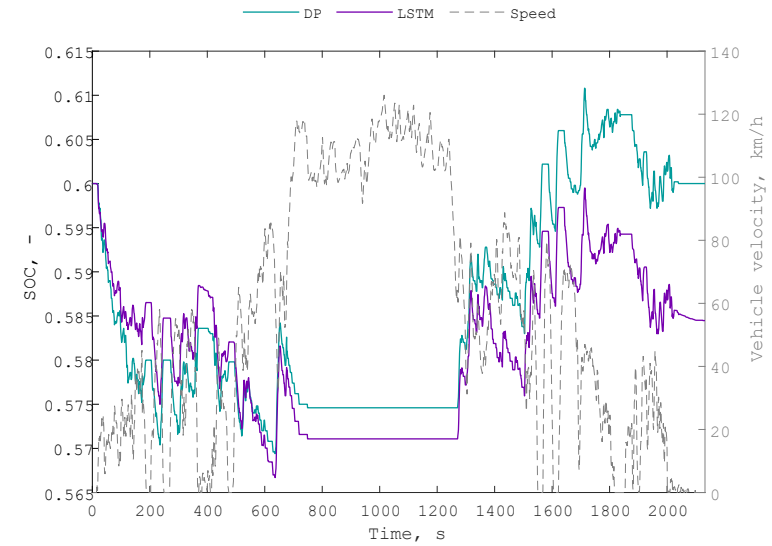


Fig. 2.2 Trends of battery SOC for DP and LSTM on RDE driving cycle.

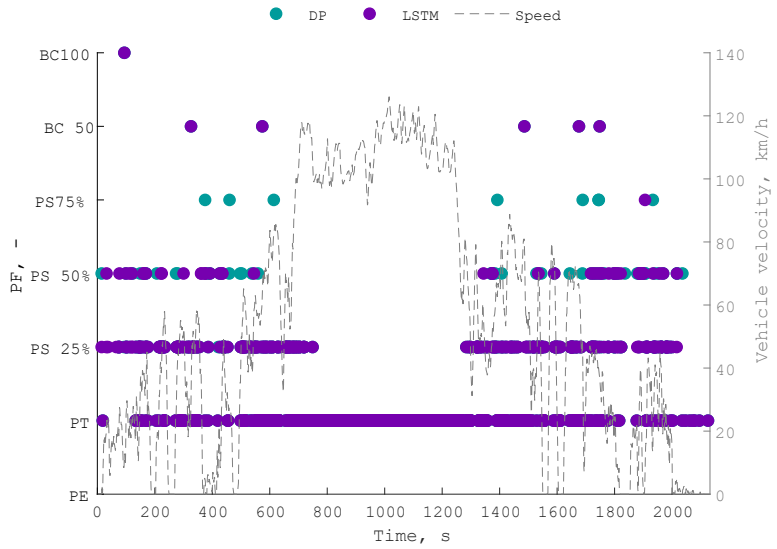


Fig. 2.3 Trends of operating modes for DP and LSTM on RDE driving cycle.

The primary limitation of the approach, stems from the assumption of having access to GPS data alongside the necessity for feature extraction to accurately define driving characteristics. This assumption is critical as it enables the prediction of future driving conditions, which is essential for optimizing energy distribution between the electric motor and internal combustion engine in real-time. While the initial results have been promising, indicating a potential for significant improvements in fuel efficiency and emission reduction, the model's applicability is constrained in scenarios involving dynamic and unpredictable routes. This limitation arises because the LSTM's performance heavily relies on the quality and representativeness of the training data, which may not always encompass the diversity of real-world driving conditions.

Moreover, the current framework addresses a simplified representation of the actual driving environment. It does not incorporate penalties for the activation of the ICE, an aspect that could lead to suboptimal decisions in terms of passenger comfort and noise levels. This omission highlights a gap between the model's operational efficiency and the holistic driving experience, suggesting a need for a more integrated approach.

2.5 Test Case 2

In this section, we present numerical evaluations, validating the proposed control Algorithm 2. Case study 2.5.2 demonstrates the scalability of the algorithm over different driving missions[5].

2.5.1 Evaluation Metrics

From a physical point of view, the performance evaluation encompassed several metrics including the cumulative fuel consumption over the driving mission, the frequency of ICE de/activations, and the final SOC of the battery. Specifically, the fuel consumption per unit of distance travelled (L/100km) was selected as the energy efficiency index; the frequency of ICE activations, measured in occurrences per minute (1/min), was selected as the passengers' comfort index; the final SOC (SOC_f) was selected as the charge-sustaining index.

[5]This section has been extracted from [15].

2.5.2 Comparison Assumptions

To ensure a mathematically rigorous evaluation between two distinct algorithms, it is crucial that both are designed to address the same mathematical problem, incorporating an identical vehicle model, a uniform objective function, and comparable constraints. The expectation is for the off-policy algorithm to align with the selected benchmark, dynamic programming, over an infinite time horizon. Once the two algorithms achieve comparable results in terms of cumulative cost function and cumulative reward function, the comparison should be consistent from a physical point of view as well. However, in practical scenarios, particularly when addressing a multi-objective control problem, identifying a feasible objective function that yields the desired outcomes might be challenging. This is due in part to the fact that the benchmark algorithm allows for the enforcement of a final state, whereas achieving this outcome with the off-policy approach is not straightforward. Consequently, we opted to compare the results obtained from both reinforcement learning (RL) and dynamic programming (DP), utilizing two different objective functions, with a specific emphasis on physics and set goals. Specifically, as shown in the reward function in Table 2.1, the RL agent faced a penalty based on the SOC value compared to the reference one, a penalty every time it starts the engine, and a term related to fuel consumption throughout the driving mission. On the dynamic programming side, the cost function we selected resulting from the best trade-off among the defined objectives, includes a penalty for the frequency of ICE de/activations, a contribution term for consumption, and a final state constraint [14].

Correction of Fuel Consumption to Account for SOC Variation with Respect to the Target Value

In practical implementations, when the final SOC does not reach the target value, we corrected the actual value of fuel consumption by accounting for the net amount of energy variation in the battery, as carried out in [13].

$$\dot{m}_{f,corr} = \dot{m}_f + \theta \Delta SOC \tag{2.14}$$

where θ translates the amount of energy used in the battery into an equivalent fuel consumption considering the ICE efficiency η_{ICE} and the fuel lower heating value $\mathrm{H_{f,LHV}}$.

$$\theta = \frac{P_{batt}}{\eta_{ICE} \cdot H_{f,LHV}} \tag{2.15}$$

2.5.3 Vehicle Model

A Jeep Renegade 4xe represented as a parallel P4 architecture, whose scheme is reported in Figure 2.4, was considered for the purpose of this study [14]. Specifically, the internal combustion engine (ICE) is responsible for powering the front axle, while the electric motor (EM or MGP4) drives the rear axle and is directly connected to the high-voltage battery pack. The main vehicle specifications, obtained by secondary data available online [14], are listed in Table 2.4. The model and algorithm were developed and implemented within the MATLAB® simulation environment [65].

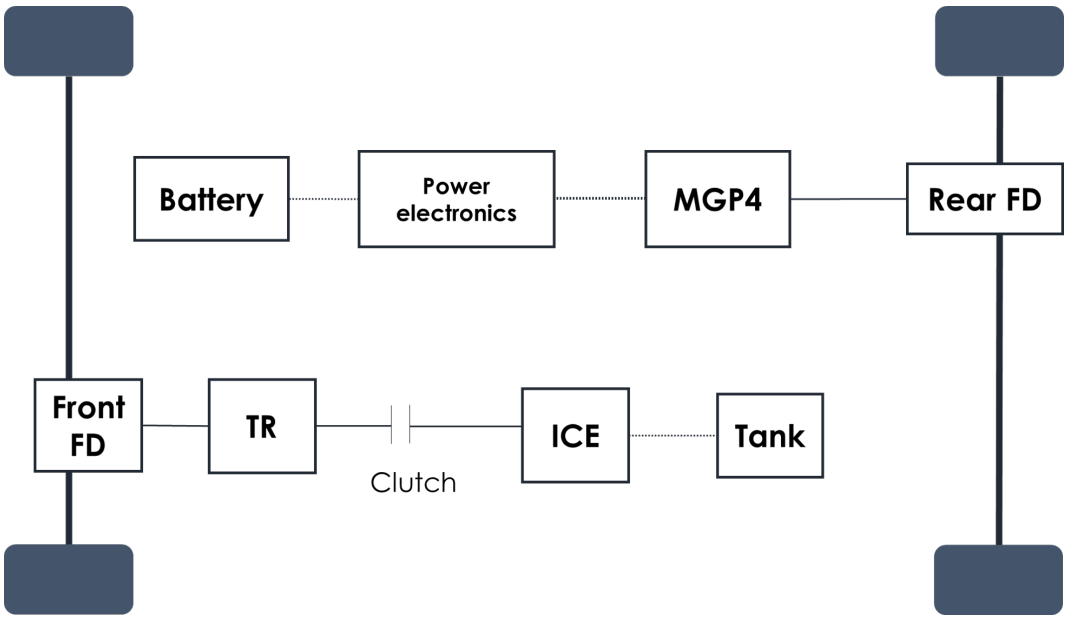


Fig. 2.4 Scheme of the considered electrified architecture.

Table 2.4 Vehicle specifications.

Component	**Parameter**	**Value**
Vehicle	Mass, kg	1850
	RL_a, N	125.22
	RL_b, $\frac{N}{(m \cdot s)}$	1.95
	RL_c, $\frac{N}{(m \cdot s^2)}$	0.59
	Tyre radius, m	0.29
Engine	Displacement, l	1.4
	Rated Power, kW	133
	Maximum torque, Nm	270
EM	Rated Power, kW	44
	Maximum torque, Nm	250
Battery	Type	NMC
	Nominal capacity, Ah	28.4
	Nominal voltage, V	400

2.5.4 Results

DP Results

The main results for the WLTP driving cycle, summarized in Table 2.5, are presented considering the aforementioned metrics. As outlined in Section 2.3.1, the algorithm performance was evaluated considering different objective functions accounting for:

1. Fuel economy and charge sustainability (I);
2. Trade-off between fuel-economy, charge sustainability and drivability (II) [14];
3. Same reward function used for the Q-learning learning algorithm (III) (Please refer to Equation (2.10)).

Complementing this analysis, Figure 2.5 depicts the SOC trends, showcasing the outcomes of employing different objective functions.

Table 2.5 Performance results for DP algorithm on the WLTP driving cycle.

Label [1] -	**FC** [2] **L/100 km**	$\mathbf{f_{ICE}}$ [2] **1/min**	$\mathbf{SOC_f}$ **-**	$\mathbf{FC_{corr}}$ [2,3] **L/100 km**
I	6.69	2.1	0.201	6.71
II	7.08	0.13	0.203	7.18
III	7.6	0.07	0.203	7.74

[1] It indicates a specific objective function; [2] FC = Fuel consumption; f_{ICE} = Frequency of ICE activations; FC_{corr} = Corrected fuel consumption. [3] Correction of fuel consumption to account for SOC variation.

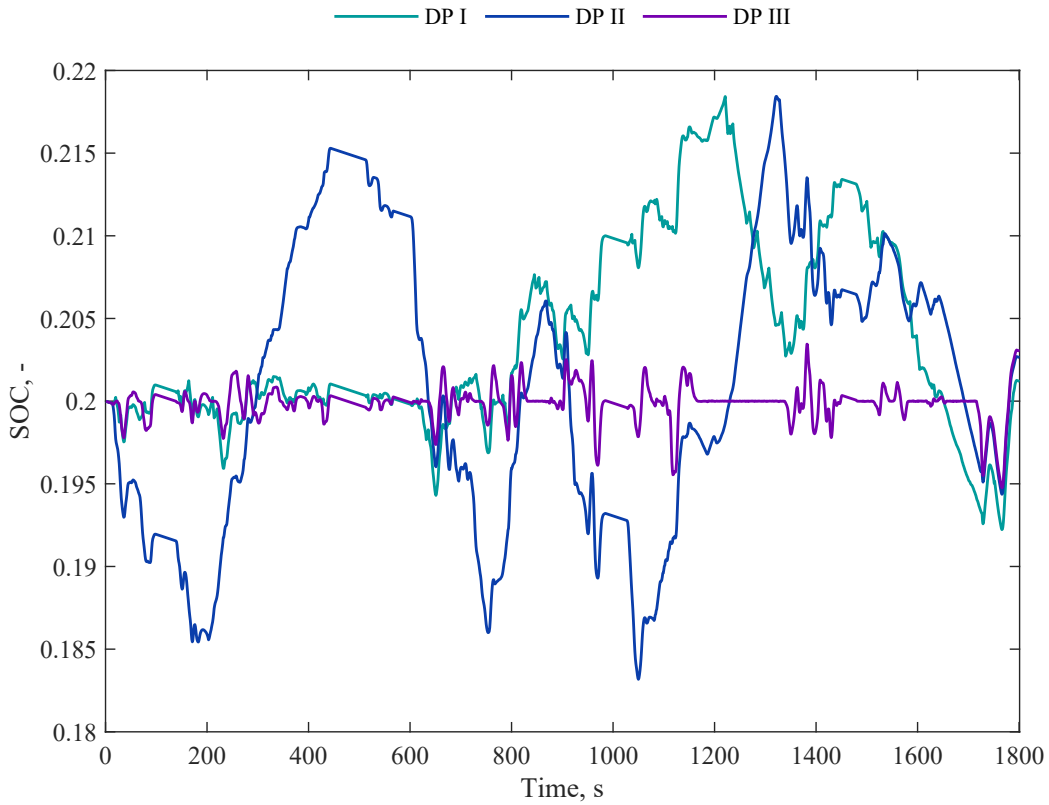


Fig. 2.5 Battery state of charge (SOC) trends for dynamic programming (DP) on WLTP driving cycle, highlighting the impact of three different objective functions.

Q-Learning Results and Discussion

The agent was trained and validated on the worldwide harmonised light vehicle test procedure (WLTP) driving cycle, whereas the testing was performed on the Artemis cycles, including urban (AUDC), rural (ARDC), and motorway (AMDC) segments. The Artemis driving cycles show higher average, and max speeds, and faster acceleration compared to the WLTP cycle. These cycles are designed to adapt to various vehicle types and sizes, incorporating both transient and steady-state driving conditions. The goal is to prove the robustness of the proposed algorithm when applied to driving cycles different from the training one. The performance of the algorithm was assessed considering the cumulative reward, depicted in Figure 2.6. As observed, despite some fluctuations in the trend caused by the absence of regularization techniques, the algorithm exhibits convergence at approximately episode 1500. The figure showcases four stars representing validation episodes focused on pure exploitation, which are conducted every 500 episodes. Specifically, Q_A refers to episode 1000, Q_B to episode 1500, Q_C to episode 2000, and Q_D to episode 2500. On the other hand, the remaining points correspond to the epsilon-greedy approach, where the exploration percentage exponentially decreases during the training phase. During episodes 1 to 850, there is an empty block indicating that the agent was unable to reach the end of the episode.

The observed discrepancy in cumulative rewards between episode 2500 and episode 1500 provides evidence of a potential local minimum, suggesting a limitation in agent performance. While alternative techniques or strategies could have been explored to mitigate this issue, it is worth noting that both outcomes were on the descending part of the exploration-exploitation law curve, with an ϵ lower than 0.2. Therefore, the current configuration was retained.

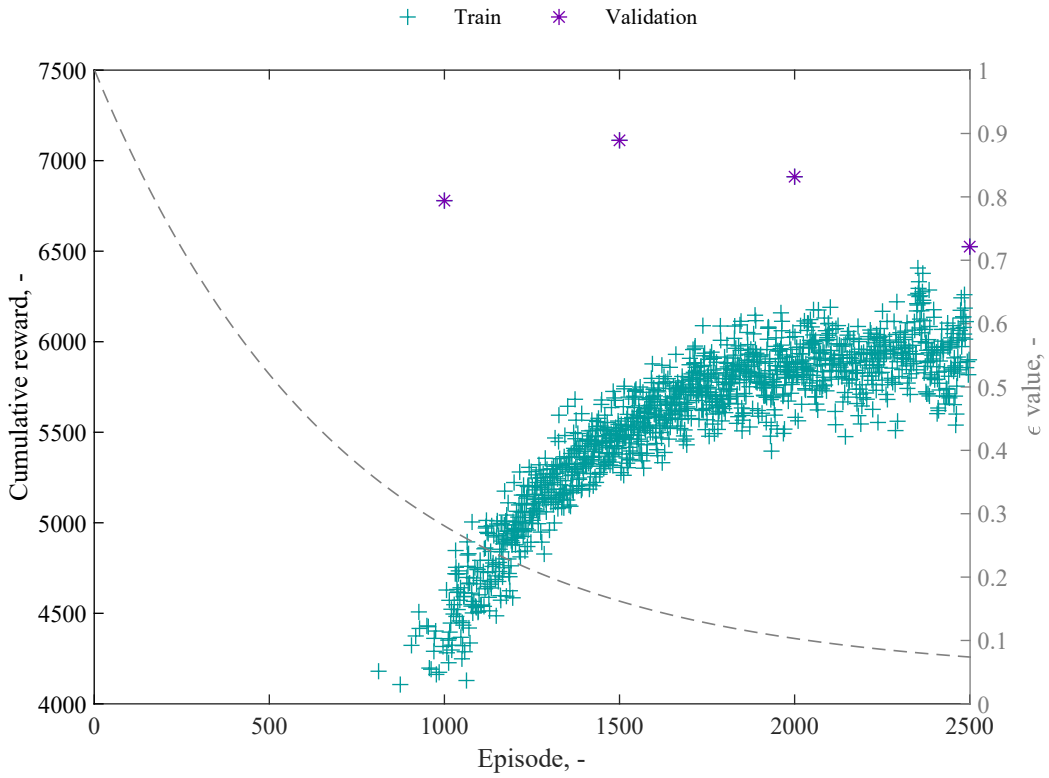


Fig. 2.6 Evolution of cumulative reward over episodes with validation and epsilon-greedy approaches.

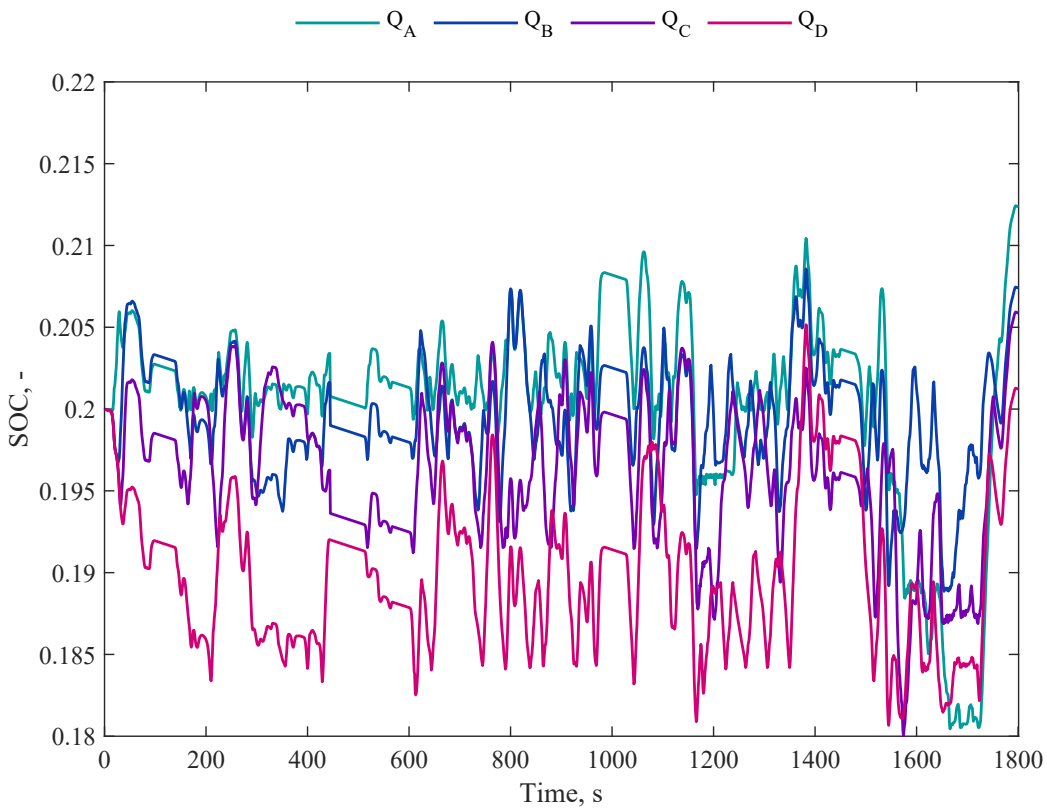


Fig. 2.7 Trends of battery SOC during training with intermittent exploitation introduced every 500 episodes for WLTP driving cycle.

The main training, validation, and testing results are summarized in Figure 2.7 and Tables 2.6–2.8, respectively. The labels indicate the different Q–tables obtained through the pure exploitation episodes. The average fuel consumption achieved during the training and validation processes is approximately 7.58 L/100 km, ranging from 7.62 L/100 km in episode 1000 to 7.53 L/100 km in episode 2500. The average frequency of ICE de/activations is approximately 1.15 1/min, with variations from 1.63 1/min in episode 1000 to 0.8 1/min in episode 2500. On average we obtain a final state of charge (SOC_f) of approximately 0.207, ranging from 0.212 in episode 1000 to 0.201 in episode 2500.

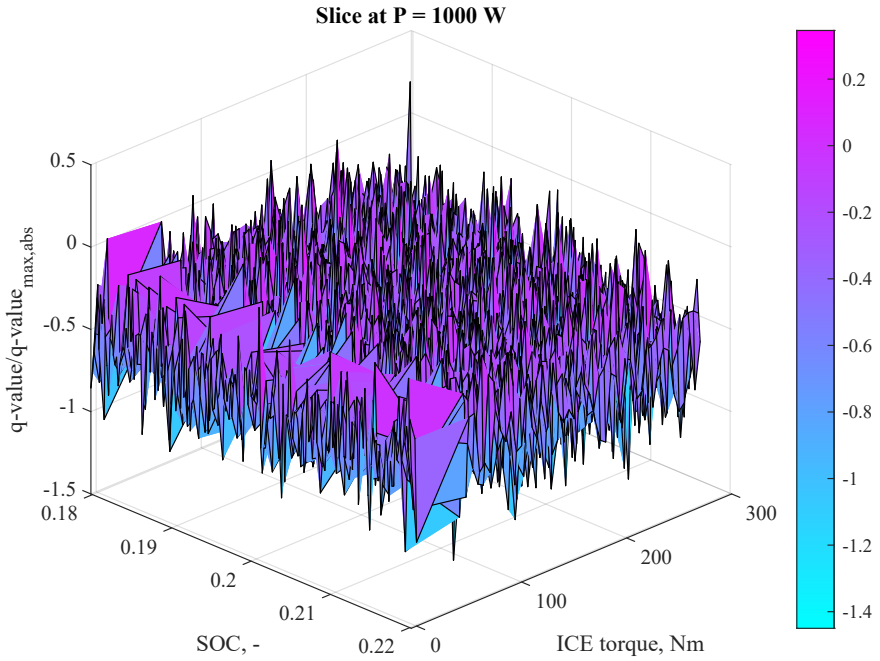
Table 2.6 Performance results for training and validation on the WLTP driving cycle.

$\mathbf{Q_{val}}$ [1] -	Episode -	FC [2] L/100 km	$\mathbf{f_{ICE}}$ [2] 1/min	$\mathbf{SOC_f}$ -	$\mathbf{FC_{corr}}$ [2,3] L/100 km
Q_A	1000	7.62	1.63	0.212	9.89
Q_B	1500	7.59	1	0.207	8.39
Q_C	2000	7.56	1.17	0.206	8.07
Q_D	2500	7.53	0.8	0.201	7.55

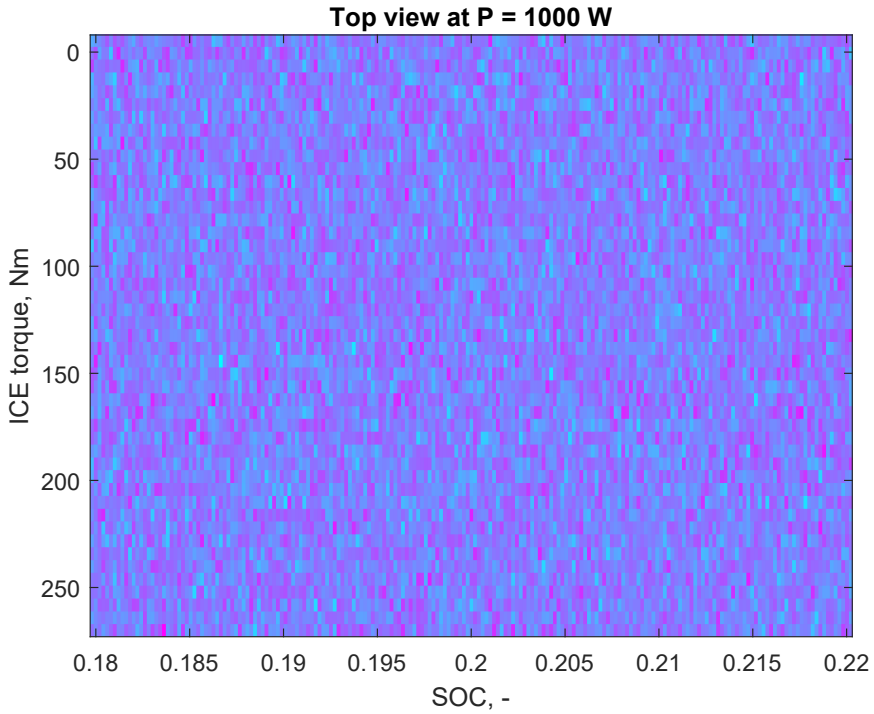
[1] It indicates a specific pure-exploitation validation episode; [2] FC = Fuel consumption; f_{ICE} = Frequency of ICE activations; FC_{corr} = Corrected fuel consumption. [3] Correction of fuel consumption to account for SOC variation.

To enhance result comprehension, Table 2.7 provides a comprehensive summary of performance in relation to the DP algorithm with different objective functions. Among the analyzed results from dynamic programming, DP_{II} stands out as the one that effectively balances fuel consumption, charge sustainability, and drivability. On average, the different episodes of pure exploitation regarding fuel consumption exhibit a mean deviation below 7%, whereas in terms of engine activations, on average they occur approximately 9 times as frequently. On the final SOC side, the comparison was not performed because starting from Q_B, there is a percentage deviation of 3.5% from the target, which we considered within acceptable tolerance ranges.

The Q–table was initialized as a three-dimensional array and each element was independently sampled from a normal distribution, characterized by a certain mean and standard deviation. For the sake of completeness, we reported in Figure 2.8 a sliced view at a specific wheel power request for the Q_B table to give the reader a visual representation of the Q–table. This includes a 3D visualization in Figure 2.8(a) showcasing the distinct shape of the Q–table in a particular section, as well as a 2D top-view representation in Figure 2.8(b) to provide an overview of its contents and stored values.



(a) 3D Section of the Q_B Table at a specific wheel power request, illustrating the structure and value distribution.



(b) 2D Top-View of the same Q_B Table section, showing the overview of data distribution.

Fig. 2.8 Visual representations of the Q_B Table in both 3D and 2D views at a specific wheel power request.

Table 2.7 Comparative performance analysis during the WLTP validation phase for the proposed algorithm and DP.

	Fuel Consumption % Difference		
$\mathbf{Q_{val}}$ [1]	**w.r.t.** [2] $\mathbf{DP_I}$	$\mathbf{DP_{II}}$	**w.r.t.** $\mathbf{DP_{III}}$
Q_A	+13.9	+7.6	+0.2
Q_B	+13.45	+7.2	−0.13
Q_C	+13	+6.78	−0.53
Q_D	+12.56	+6.35	−0.92
	Corrected fuel consumption % difference		
$\mathbf{Q_{val}}$	**w.r.t.** $\mathbf{DP_I}$	**w.r.t.** $\mathbf{DP_{II}}$	**w.r.t.** $\mathbf{DP_{III}}$
Q_A	+46	+37.7	+27.8
Q_B	+25	+16.85	+8.4
Q_C	+20.27	+12.39	+4.26
Q_D	+12.52	+5.15	−2.45
	Frequency of ICE de/activations compared to DP		
$\mathbf{Q_{val}}$	**w.r.t.** $\mathbf{DP_I}$	**w.r.t.** $\mathbf{DP_{II}}$	**w.r.t.** $\mathbf{DP_{III}}$
Q_A	−0.47	+1.5	+1.56
Q_B	−1.1	+0.87	+0.93
Q_C	−0.93	+1.04	+1.1
Q_D	−1.3	+0.67	+0.73

[1] It indicates a specific pure-exploitation validation episode; [2] with respect to.

For the testing phase on the three Artemis driving cycles, we selected the two Q–tables that lead to the highest and lowest cumulative reward, namely Q_B and Q_D. They both achieve a final SOC within the feasible range of 0.18–0.22, with Q_B showing higher proximity to the target value. However, Q_B exhibits higher fuel consumption and frequency of ICE activations compared to Q_D.

Specifically, for the urban driving cycle (AUDC), Q_B consumes 5.77 L/100 km, approximately 16.33% higher than Q_D's fuel consumption of 4.96 L/100

km. Additionally, Q_B has a higher frequency of ICE activations of 0.66 1/min, representing a 36.81% increase compared to Q_D's frequency of 0.483 1/min.

For the rural driving cycle (ARDC) and the motorway one (AMDC), Q_B shows an increase in fuel consumption of approximately 5.6% and 4.3% respectively, compared to the Q_D. Additionally, Q_B exhibits higher frequencies of ICE de/activations, with a 40% increase for ARDC and an 85% increase for AMDC.

Table 2.8 Performance results: training on the WLTP driving cycle and testing on an unknown driving cycle.

$\mathbf{Q_{val}}$ [1] -	**Cycle** -	**FC** [2] **L/100 km**	$\mathbf{f_{ICE}}$ [2] **1/min**	$\mathbf{SOC_f}$ -	$\mathbf{FC_{corr}}$ [2,3] **L/100 km**
Q_B	AUDC	5.77	0.66	0.2	5.77
Q_B	ARDC	6.74	1.72	0.2	6.74
Q_B	AMDC	10.87	1.24	0.202	10.91
Q_D	AUDC	4.96	0.483	0.187	16.26
Q_D	ARDC	6.38	1.22	0.192	7.75
Q_D	AMDC	10.42	0.67	0.192	11.15

[1] It indicates a specific pure-exploitation validation episode; [2] FC = Fuel consumption; f_{ICE} = Frequency of ICE activations; FC_{corr} = Corrected fuel consumption. [3] Correction of fuel consumption to account for SOC variation.

Adopting a conservative approach for the final comparison, the results of Artemis cycles of Q_B were compared to those of DP_{II}, which offers the best trade-off among fuel consumption, the frequency of ICE de/activations, and charge sustainability (Tables 2.9 and 2.10). On the fuel economy side, the proposed algorithm obtains an average increase of around 5%, whereas on the frequency of ICE activations side, we have an average frequency of around 12 times higher. The best performances are observed for the AUDC cycle, and the worst for the AMDC cycle. Figure 2.9 shows the results in terms of SOC for both DP_{II} and Q_B for AUDC (top), ARDC (middle) and AMDC (bottom) cycles. Similarly, Figure 2.10 shows the results in terms of ICE torque. During

the testing phase, the proposed algorithm demonstrates the ability to maintain the charge-sustaining behaviour even on unknown driving cycles. It achieves a fuel economy comparable to the benchmark algorithm optimized for the specific driving mission while keeping the frequency of ICE de/activations below 2 per minute.

Table 2.9 Performance results for $DP_{II}{}^{1}$ algorithm on the Artemis driving cycles.

Cycle	**FC** [1]	$\mathbf{f_{ICE}}$ [1]	$\mathbf{SOC_f}$	$\mathbf{FC_{corr}}$ [1,2]
-	**L/100 km**	**1/min**	**-**	**L/100 km**
AUDC	5.75	0.121	0.2018	5.97
ARDC	6.27	0.167	0.202	6.36
AMDC	10.1	0.06	0.202	10.17

[1] FC = Fuel consumption; f_{ICE} = Frequency of ICE activations; FC_{corr} = Corrected fuel consumption. [2] Correction of fuel consumption to account for SOC variation.

Table 2.10 Comparative performance analysis of $Q_B{}^{2}$ and $DP_{II}{}^{1}$ during the ARTEMIS testing phase.

Cycle	**FC** [1]	$\mathbf{f_{ICE}}$ [1]	$\mathbf{FC_{corr}}$ [1,2]	$\mathbf{SOC_f}$
-	**L/100 km**	**1/min**	**L/100 km**	**-**
AUDC	5.77	0.66	5.77	0.201
w.r.t. [3] DP_{II}	+0.34%	+0.54	−3.35	-
ARDC	6.74	1.72	6.74	0.1998
w.r.t. DP_{II}	+7.49%	+1.55	+5.97	-
AMDC	10.87	1.23	10.91	0.2019
w.r.t. DP_{II}	+7.62%	+1.17	+ 7.27	-

[1] FC = Fuel consumption; f_{ICE} = Frequency of ICE activations; FC_{corr} = Corrected fuel consumption. [2] Correction of fuel consumption to account for SOC variation. [3] with respect to.

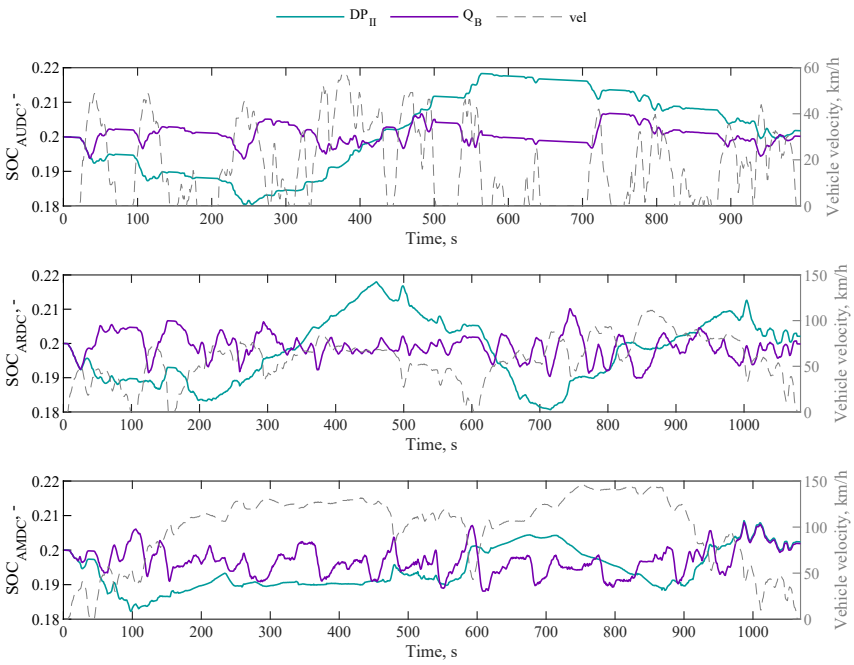


Fig. 2.9 Trends of battery SOC for DP_{II} and Q_B on ARTEMIS driving cycles.

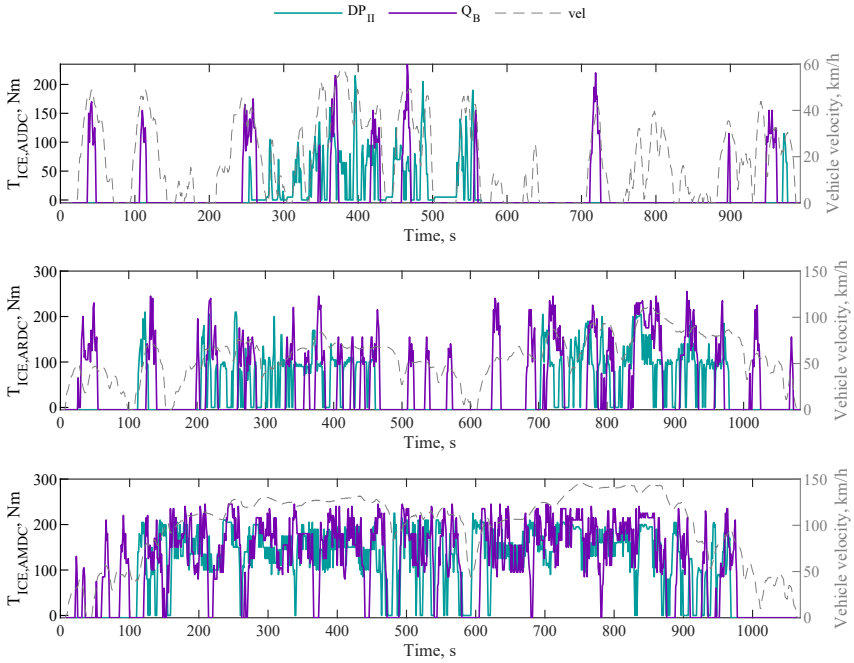


Fig. 2.10 Trends of ICE torque for DP_{II} and Q_B on ARTEMIS driving cycles.

Chapter 3

AI techniques for CACC of BEVs

3.1 Introduction

3.1.1 Motivation

To achieve the so-called sustainable mobility, governments established a series of objectives to be achieved in the short term, with environmental implications as well as improvements in terms of transport efficiency. One example is the introduction at a regulatory level in newly homologated vehicles of advanced driving assistance systems (ADASs), specifically designed to support the driver by ensuring safer, more efficient, and more ecological driving. Adaptive cruise control (ACC), lane keeping (LK), and emergency brake assist are some basic examples. Although some of these are by now consolidated features, the growing possibilities dictated by vehicle-to-everything connectivity have led to redefine their characteristics, paving the way for an interconnected mobility. The potential benefits deriving from ADAS technologies are manifold. Improvements in energy consumption, in passenger safety and comfort along with reduced travel times are among the mainly investigated in the literature [66, 67]. The cooperative adaptive cruise control (CACC) often connected to the concept of eco-driving, allows for adapting the driving trajectory using the information received from other vehicles or from the infrastructure. To this end, several technologies can be used. Vehicle communication technologies such as vehicle-to-everything (V2X) or on-board sensors (RADAR, LiDAR etc) are some examples.

They supply time series data information that can be used to consequently adjust the vehicle longitudinal trajectory. However, managing the powertrain according to the condition of the vehicle as well as the activity of the external environment is a complex task.

3.1.2 Contribution

This chapter presents a methodology that integrates a deep learning framework with vehicular communication technologies and sensor systems to develop a real-time CACC system. Initially, the study selects and trains a gated recurrent unit (GRU) architecture using datasets constructed based on an optimal control strategy, notably dynamic programming (DP), with a focus on a battery electric vehicle. The primary aim of DP in this context is to optimize the longitudinal speed trajectory of the following vehicle within the CACC system, thereby enhancing energy efficiency and improving passenger comfort. The novelty of this work primarily lies in the use of a GRU architecture within CACC systems, integrating deep learning with vehicular dynamics to better understand the potential of these techniques under variable conditions.

In the subsequent test case, the investigation delves into the effects of uncertainties in sensor data and communication links. A decentralized MPC framework is utilized as the standard for comparison. Concurrently, a reinforcement learning (RL) approach, specifically a Soft Actor-Critic (SAC) agent, is implemented to investigate its potential for real-world applications. Our approach to managing multiple uncertainties through a RL framework represents a new direction in CACC research when RL-based algorithms are considered, addressing complexities that have been less explored in previous studies.

3.1.3 Related works

In the literature there are several works related to the cooperative adaptive cruise control application. Targets achievable through this system include energy saving, comfort and safety enhancement along with improvements in traffic throughput. A variety of algorithms can be considered in the definition of the vehicle longitudinal trajectory. Controllers based on model predictive control

(MPC) are widely used [68–71]. As an example, a learning-based stochastic MPC is developed and validated in [70] for several driving scenarios, including cut-in manoeuvres handling. In [71], a robust MPC approach is considered to evaluate CACC system performance in case of packet loss information. The choice of an MPC-based control derives mainly from its retroactive nature; however, the accuracy of the system and the related calculation times are depending on the selected type of MPC (i.e. linear, adaptive, non-linear, robust). Another not trivial point related to MPC-based models lies in the definition of the state equations, especially when trade-off solutions between accuracy and computational times are considered because in general, they require linearization of the main systems considered (not only in dynamics equations but also in components map). To overcome these problems, machine learning techniques were considered as a possible alternative. In 2011, Desjardins et al [72] presented a reinforcement learning-approach for vehicle control. More specifically, they used a policy gradient algorithm and a backpropagation neural network to achieve the desired control trajectory. Speaking about recurrent neural networks, Tian et al [73] proposed a long-short term memory neural network to predict the lead vehicle profiles related to longitudinal dynamics control aiming to compensate for communication delay of communication technology. Main findings refer to improvement in the string stability when the communication delay exceeds a certain threshold.

3.1.4 Outline

The chapter is structured as follows: Section 3.2 briefly outlines the problem under investigation. Section 3.3 discusses the algorithms utilized. Finally, Sections 3.4–3.5 describe the selected test cases.

3.1.5 Notation

In this section, we introduce the notation used throughout the mathematical formulation of the vehicle dynamics under CACC and the algorithms employed in the study. We consider a configuration of n vehicles, with each vehicle indexed by $\mathrm{i} \in \{0, 1, 2, 3, 4\}$, where $\mathrm{i} = 0$ corresponds to the lead vehicle and $\mathrm{i} = \{1, 2, 3, 4\}$ to the follower vehicles.

Let $x_i(t)$ denote the position of the i^{th} vehicle at time t, with $\ddot{x}_i(t)$ representing its acceleration. The control input for each vehicle is given by $u_i(t)$, which is subject to constraints $u_{min} \leq u_i(t) \leq u_{max}$ to ensure feasible acceleration and deceleration values.

For the GRU model, v_t represents the input vector at time t, and h_t denotes the hidden state. The update gate z_t, reset gate r_t, and candidate activation vector $\hat{h}_t$ are crucial components of the GRU. The sigmoid function σ and hyperbolic tangent function tanh are used as activation functions. Weight matrices are denoted by W_z, U_z, W_r, U_r, and W for the respective gates and updates.

In the SAC algorithm, the policy is parameterized by θ, and the critic networks by ϕ_1 and ϕ_2. The target critic networks, used for stable training, are denoted by $Q_{\phi'_1}$ and $Q_{\phi'_2}$. The replay buffer is represented by $\mathcal{D}$, storing experience tuples (s, a, r, s') where s, a, r, and s' stand for state, action, reward, and next state, respectively. The temperature parameter α controls the importance of entropy in the policy's objective function, while γ is the discount factor for future rewards. The target network update rate is represented by τ.

3.2 Problem formulation

In the present study, a configuration consisting of n vehicles is investigated, structured in a string including one lead vehicle and four follower vehicles, all operating under a cooperative adaptive cruise control system. Each vehicle within this configuration is designated by an index i, where i is defined within the range of 0 to 3 in Test Case 3.4 and 0 to 4 in the other.

To narrow the scope of the investigation, the analysis is limited to the first-order longitudinal dynamics of these vehicles. This approach intentionally excludes the lateral dynamics and the potential for overtaking scenarios. The rationale behind this limitation is to mitigate the impact of model non-linearities on the efficacy of the control algorithm.

The dynamic behavior of the i_{th} vehicle in this arrangement is governed by the following set of equations:

$$\tau_i \frac{d}{dt}\ddot{x}_i(t) + \ddot{x}_i(t) = u_i(t) \tag{3.1}$$

$$u_{min} \leq u_i(t) \leq u_{max} \tag{3.2}$$

3.3 Algorithms

3.3.1 Gated Recurrent Unit (GRU)

GRUs feature memory units with two gates, the update and the reset ones. These two gates are mainly designed to let information pass selectively, deciding weather an information must be preserved or discarded. Since for the considered application this approach has not been applied extensively, it might be worth clarifying the functioning of the gates and their structure. For a certain input vector v_t, the characteristic equations of the GRUs can be summarized in (3.3) – (3.6).

$$z_t = \sigma\left(W_z v_t + U_z h_{t-1}\right) \tag{3.3}$$

$$r_t = \sigma\left(W_r v_t + U_r h_{t-1}\right) \tag{3.4}$$

$$\hat{h}_t = \tanh\left(W v_t + r_t \odot h_{t-1}\right) \tag{3.5}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{3.6}$$

where the update gate, reset gate and candidate activation vector are indicated by z_t, r_t and $\hat{h}_t$, respectively. σ and tanh refer to the sigmoid and hyperbolic activation functions respectively whereas $\odot$ represents element–wise multiplication. The weight matrices are labeled as W_z, U_z, W_r, U_r, W, U. While the reset gate r_t and the update gate z_t share a similar structural form, their functions are distinct. The reset gate r_t determines which elements of the previous hidden state (h_{t-1}) should be replaced with the current inputs, as outlined in (3.5). Conversely, the update gate z_t influences the present hidden state h_t, deciding which aspects of the previous hidden state are to be revised with the current candidate activation vector $\hat{h}_t$, detailed in (3.6). For a

comprehensive understanding of the GRU, consulting the relevant specialized literature is recommended [74–77].

3.3.2 Soft-actor-critic Agent

Soft Actor Critic (SAC) algorithm relies on a stochastic policy using an off–policy approach. This policy is developed to optimize the balance between expected returns and entropy, the latter being an indicator of the policy's randomness. The concepts of entropy and the exploration-exploitation trade–off share common traits. Specifically, boosting entropy promotes greater exploration, potentially hastening the learning process in subsequent stages. Additionally, it can prevent the policy from converging to a local optimum [78].

In the SAC algorithm, several key variables play pivotal roles. The actor network is parameterized by θ and represents the policy which outputs a probability distribution over actions. The critic networks is parameterized by ϕ_1 and ϕ_2 and estimate the value of state-action pairs. The target critic networks $\mathrm{Q}_{\phi_1'}, \mathrm{Q}_{\phi_2'}$ are updated versions of the critic networks and are used for stable training. The replay buffer $\mathcal{D}$ is a data structure used to store and retrieve experience tuples $(\mathrm{s}, \mathrm{a}, \mathrm{r}, \mathrm{s}')$. Similarly to the preeceding paragraph, $\mathrm{s}, \mathrm{a}, \mathrm{r}, \mathrm{s}'$ represent state, action, reward, and next state, respectively. The temperature parameter α determines the importance of the entropy term in the objective function. The discount factor γ is used to balance immediate and future rewards. The target network update rate τ controls the rate at which the target networks are updated towards the learned networks.

Algorithm 3 Soft Actor-Critic (SAC)

Initialize actor network $\pi(\theta)$, critic networks Q_{ϕ_1}, Q_{ϕ_2}, and empty replay buffer $\mathcal{D}$
Initialize target networks $Q_{\phi'_1}, Q_{\phi'_2}$ with weights $\phi'_1 \leftarrow \phi_1, \phi'_2 \leftarrow \phi_2$
Initialize temperature parameter α
for each iteration **do**
  for each environment step **do**
    $a \sim \pi(\theta|s)$ {Sample action from the policy}
    $s', r \leftarrow$ step environment with action a
    Store (s, a, r, s') in replay buffer $\mathcal{D}$
  end for
  for each gradient step **do**
    $B \leftarrow$ sample from $\mathcal{D}$ {Sample a batch of transitions}
    for each Q–network **do**
      $y(r, s', d) \leftarrow r + \gamma(1 - d)(\min(Q_{\phi'_1}(s', \pi(\theta|s')), Q_{\phi'_2}(s', \pi(\theta|s'))) - \alpha \log(\pi(\theta|s')))$
      Update Q_{ϕ_i} by minimizing the loss: $L(\phi_i, B) = \mathbb{E}[(Q_{\phi_i}(s, a) - y(r, s', d))^2]$
    end for
    Compute policy loss $L(\theta, B) = \mathbb{E}[\alpha \log(\pi(\theta|s)) - Q_{\phi_1}(s, \pi(\theta|s))]$
    Update θ by maximizing $L(\theta, B)$
    Update target networks:
    $\phi'_1 \leftarrow \tau\phi_1 + (1 - \tau)\phi'_1$
    $\phi'_2 \leftarrow \tau\phi_2 + (1 - \tau)\phi'_2$
    Optionally, adjust α, the temperature parameter
  end for
end for

3.4 Test case 1

The present test case aims at using a deep learning-based approach to control vehicle acceleration based on information from the vehicle immediately ahead in the same carriageway. Specifically, we propose the use of a gated recurrent unit (GRU), i.e. a variant of recurrent neural network designed to avoid

vanishing gradient problem linked to long time series. GRU can extract the main information of the time series and it is capable to find the nonlinear interconnection between the input features and the output ones. In this study the output information refers to the longitudinal control of the battery electric vehicle considered based on the information relative to the vehicle in front. The vehicle considered in the present work is a battery electric vehicle (BEV) and its characteristics data are listed in Table 3.1.

Table 3.1 Vehicle characteristics data.

Vehicle characteristics	**Units**	**Value**
Curb weight	kg	1474
Battery capacity	kWh	42
EM peak power	kW	83

Main powertrain components such as electric machine and energy storage system were modelled using a map-based approach. The vehicle modelling has been described in the authors' previous works, for more details please refer to [79, 80]. The problem under analysis refers to a string of two vehicles on the same carriageway, equipped with V2V technology and on-board sensors such as LiDAR and Radar. They allow for the exchange of information in terms of the distance between the two vehicles as well as the speed of the vehicle in front. The considered driving scenario will refer to the vehicle that follows as ego vehicle, whereas to the vehicle at the front as leading vehicle. The considered flow topology is predecessor following [81] meaning that the ego vehicle only receives the leading vehicle information in terms of velocity and position. For simplicity, a homogeneous string of vehicles has been considered i.e., all vehicles are of the same type. The control problem referred to has as its objective the definition of the optimal ego vehicle longitudinal trajectory by minimizing the energy consumption of the battery[1]. The control problem under analysis was solved by considering a machine learning technique based on recurrent neural networks (RNNs). The reasons behind this choice are related to the

[1]The distance between the lead and ego vehicle is maintained within a specified limit, based on the framework we adopted. This ensures consistency with the authors' previous work [79, 80].

characteristics of the RNNs, mainly designed for time series handling. This type of network is characterized by a memory cell which manages the information coming from the previous and current inputs to generate the current output(s). Referring to the proposed application, the present work will consider standard driving missions characterized by a time duration greater than 1000s. Therefore, to avoid vanishing gradient problems featuring the long time series, a particular type of RNN is selected, namely gated recurrent unit (GRU). The latter, as previously mentioned, is characterized by memory units with two gates, the update and the reset ones, mainly designed to let information pass selectively: their function is to establish which information must be preserved and which ones are to be discarded.

3.4.1 Evaluation Metrics

Energy saving and passenger comfort enhancement result from the formulation of the DP optimization target performed by Anselma and Belingardi in [79] and Spano et al in [82]. The lead vehicle speed and position signals, ego vehicle velocity and position at the previous time instant and inter-vehicle distance (IVD) between the two vehicles were selected as network input features. As an output signal, the network processes the velocity signal, and consequently calculate the acceleration signal, of the ego vehicle. Each feature has been normalized to speed up the training and avoid prioritization phenomena between the different features (mainly dictated by differences in the range of variation of the selected features), as follows:

$$\hat{v_{\mathrm{ij}}^{\mathrm{k}}} = \frac{v_{\mathrm{ij}}^{\mathrm{k}} - \min\left(v_{\mathrm{i}}\right)}{\max\left(v_{\mathrm{i}}\right) - \min\left(v_{\mathrm{i}}\right)} \tag{3.7}$$

where i is the generic i-th feature, j is the j-th sampling point, k represents the driving cycle considered and $\hat{\mathrm{v}}$ the normalized feature. The GRU-based system was evaluated considering different batch sizes and GRU units. The batch size refers to the number of samples used during the training phase to update the neural network weights whereas the GRU units refers to the number of units in GRU 1st layer (and consequently in the other GRU layers since their value is derived from the 1st layer so to reduce the number of tunable parameters). The optimizer and the number of samples in the past were instead

kept constant. The Adam optimizer has been selected owing to its capability to combine the advantages of RMSProp and AdaGrad [83, 84] optimizers. The number of samples in the past, on the other hand, was selected through a trial-and-error approach by selecting the minimum possible value so as not to have any memory problems in view of a possible hardware application. The metrics monitored during the training phase were the mean squared error (MSE) and the mean absolute error (MAE). MSE is defined as the average squared difference between the estimated value and the actual value whereas MAE refers to the average of the absolute error, as shown respectively in (3.8)–(3.9).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{3.8}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{3.9}$$

To improve the network performance, a regularization technique has been employed by reducing the learning rate for the optimizer if the validation loss has not improved since a pre-defined number of epochs.

3.4.2 Results

In this work, CACC control problem was addressed by a regression approach on the speed of the ego vehicle. The resulting acceleration profiles as well as the IVD were derived from the speed prediction profile. A set of conventional driving cycles (Artemis driving cycles, Worldwide Harmonized Light-duty vehicles Test Cycle - WLTP, EPA Highway Fuel Economy Test Cycle - HWFET, Supplemental Federal Test Procedure – US06) has been considered. Table 3.2 sums up their main characteristics in terms of cycle duration, distance, mean and maximum speed. Particularly, Artemis driving cycles have been considered in the GRU training phase. The training dataset has been defined considering diverse driving scenarios represented by different acceleration ranges, mean and maximum velocity. The simulations were performed for three different sets of validation and test cycles. Specifically:

- Validation cycle: WLTP; test cycles: US06 and HWFET

- Validation cycle: US06; test cycles: WLTP and HWFET
- Validation cycle: 20% of Artemis driving cycle (a portion not used for training); test cycles: WLTP, US06 and HWFET

Table 3.2 Main characteristics of training and testing datasets.

Cycle	**Duration**	**Distance**	**Average speed**	**Max speed**
-	**s**	**km**	**km/h**	**km/h**
ARTEMIS URBAN	993	4.87	17.7	57.3
ARTEMIS RURAL	1092	17.3	57.5	111.1
ARTEMIS MOTORWAY	1068	29.6	99.6	150
US06	596	12.8	77.9	129
WLTP	1800	23	45.6	131.3
HWFET	765	16.45	77.7	97

The simulation settings of the GRU-system were derived by applying a grid search approach for the model fine-tuning; the goal was finding the optimal combination of the model's hyperparameters that results in more accurate predictions. For the specific case study under analysis, the hyperparameters considered are batch size and GRU number of units. The hyperparameters considered are only some of the possible choices but they were considered sufficient to demonstrate the potential of the proposed approach based on the experience of the authors and of the main works in the literature. Table 3.3 summarizes the main characteristics of the GRU architecture, its main settings, and the range of variations of the hyperparameters to be tuned in the grid search. The grid search results were analysed considering the GRU's prediction performance in terms of root mean square error (RMSE) of the predicted ego vehicle velocity with respect to DP profiles, thus providing an indication of the standard deviation of the prediction errors. In the following, the results related to the set of simulations using the WLTP cycle as the validation dataset will be analysed. This choice is motivated by the desire to use a driving cycle

representative of different driving scenarios (urban, suburban and highway) as a validation cycle, thus guaranteeing network generalization performance. For the sake of conciseness, Figure 3.2 shows only the results of two test cases evaluated for the entire batch size set. Specifically,Figure 3.2 shows the RMSE values for all the cycles considered when the number of units of the first layer of the GRU is equal to 32 (upper part) and 256 (bottom part) respectively. In the case of 32 units, better performance is obtained for a batch size equal to 32; instead, considering 256 units the best results are obtained for batch size equal to 64. However, the improvement obtained, in our opinion, does not justify the increase in required computational times that result from the increase in the learnable parameters of the network itself (directly linked with the number of GRU units). For this reason, the selected combination of hyperparameters, as the best compromise between accuracy and computational times, is the one with 32 units in the first layer and batch size 32. For the selected combination, the training (blue) and validation (purple) loss trends are shown in Figure 3.1. It is worth recalling that the metric selected for the loss was the MSE

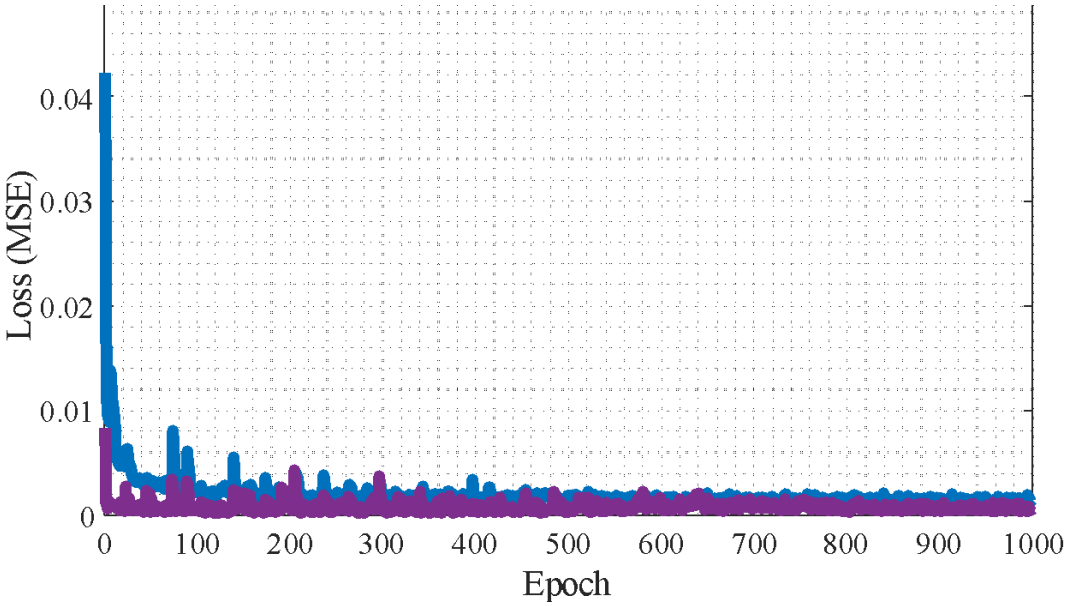


Fig. 3.1 Train and Validation loss trends. Simulation settings: ARTEMIS cycle for training, WLTP for validation, 32 GRU units in the 1st layer, 1000 # of epochs.

and that normalized features are used in the training phase, so the MSE refers to a normalized speed. As it can be seen from the loss trend figure, slightly lower values are obtained in validation than in training. This deviation can

be attributed to the use of the dropout in the training phase but not in the validation phase; moreover, in validation the loss is evaluated at the end of the epoch rather than during the epoch itself. The resulting predicted velocity profiles are shown in dark blue in Figure 3.3 for the test and validation datasets, namely US06, HWFET and WLTP; the same figure shows the profiles obtained with the DP for the ego vehicle (light blue dash-dot line) and the speed profile of the lead vehicle (grey). These profiles were analysed in post-processing, obtaining the relative results in terms of energy consumption and passenger comfort reported in Table 3.4.

Table 3.3 GRU architecture and settings.

Layer (type)	**Units**	**Settings**	
GRU	m	Dropout rate	0.2
Dropout	–	GRU's Unit values range	32–512
GRU	m	Batch size range	32–512
Dropout	–	# of samples	5
GRU	m/2	Optimizer	Adam
Dropout	–	Loss	MSE
Dense	1	Metrics	MAE, MSE

Table 3.4 Results comparison between the proposed GRU approach and DP.

Cycle	**Metric**	**Lead Vehicle**	**Ego Vehicle (GRU)**	**Ego Vehicle (DP)**
WLTP	Energy consumption (kWh/100km)	17.46	16.55	16.55
	RMS of vehicle acceleration (m/s^2)	0.52	0.37	0.34
HWFET	Energy consumption (kWh/100km)	17.05	16.86	16.75
	RMS of vehicle acceleration (m/s^2)	0.3	0.28	0.27
US06	Energy consumption (kWh/100km)	21.31	20.71	20.28
	RMS of vehicle acceleration (m/s^2)	0.99	0.64	0.61

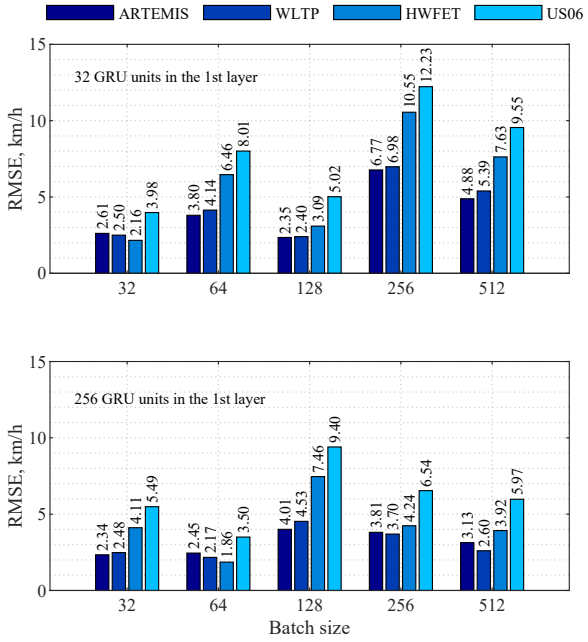


Fig. 3.2 RMSE results with respect to DP profiles for different batch sizes and driving cycles considering different GRU units in the 1st layer i.e 32 GRU units (upper part) and 256 (bottom part).

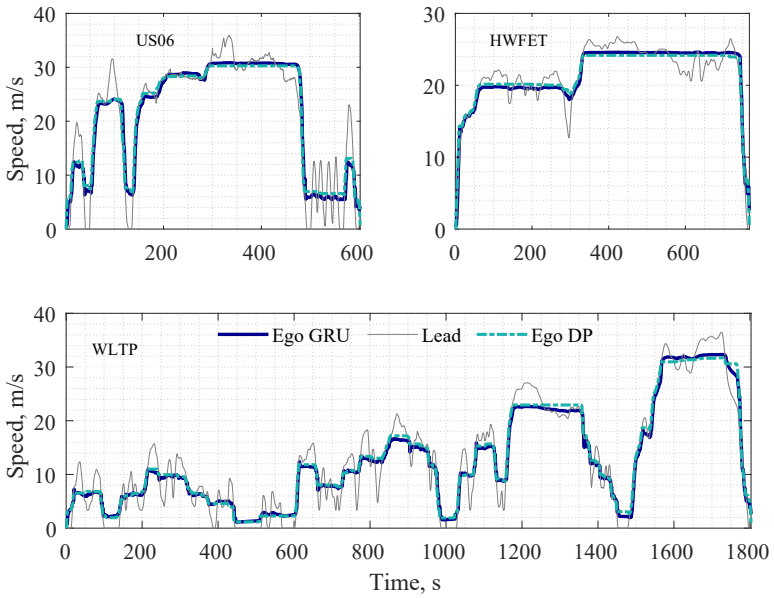


Fig. 3.3 Results comparison between the proposed GRU approach and DP in terms of speed for the test and validation datasets, i.e. US06 (upper left part), HWFET (upper right part) and WLTP (bottom part).

The term RMS of vehicle acceleration refers to the ride quality evaluated based on the root mean square of acceleration signal [85, 86]. The results shown underline the potential in terms of generalization capacity of the neural network on the test datasets (HWFET and US06) as well as on those used in the training phase. However, several challenges and limits remain open and will need to be analysed in future works. Among the main ones, the proposed network should be implemented within a medium-fidelity simulation environment to be able to check the IVD signal between the two vehicles and re-evaluate the speed prediction in the event of any deviations from the established constraints. This consideration is especially true for long driving cycles that show a deterioration in performance towards the end of the cycle, as can be seen in the lower part of Figure 3.3 for the WLTP cycle after 1200s. This deterioration may be explained by the lack of an index indicating the remaining time to travel. However, introducing a feature to represent this information would lead to other considerations related to the user's need to always select a route on the GPS, opening further challenges beyond this work. In addition, the performance obtained in terms of ride comfort does not seem optimal. This lack is attributable to a choice of non-optimal input features to the model as

well as a depth of the architecture that may require further investigation. It is worth recalling that depth of architecture refers to the number of layers in the neural network.

3.5 Test Case 2

As previously described, a CACC system is designed to regulate inter-vehicle distances, boost traffic efficiency, and uphold safety through wireless communications and on-board sensors, thereby minimizing the gap between vehicles in a platoon. Such arrangements have been proven to diminish aerodynamic drag, thereby offering notable fuel savings, especially for heavy-duty vehicles [87]. However, the deployment of CACC encounters several challenges. Its stability is contingent upon the reliability of wireless communications, which must be sufficiently robust to ensure seamless information exchange between vehicles and avert accidents. These communications are susceptible to interference and fluctuations that could undermine the system's reliability. Moreover, challenges such as network latency and sensor inaccuracies can adversely affect the system's reaction times and distance estimation precision, potentially resulting in hazardous driving scenarios. These issues underscore the importance of considering both technological and environmental variables in CACC's functionality.

In response, ongoing research endeavors are directed towards enhancing communication technologies, refining sensor accuracy, and crafting sophisticated algorithms to mitigate these uncertainties, all aimed at bolstering the reliability and safety of CACC systems.

Traditional control strategies often employ frequency domain analysis and PD (proportional-derivative) controllers to enhance the performance of control systems. Alipour et al. [88] explored the effects of communication disruptions, particularly under jamming attacks, by utilizing the wireless Rician fading channel model to depict the system. In this context, jamming means deliberate interference that compromises communication channels, thereby posing substantial challenges to system stability. Xing et al. [89] addressed random delays by implementing a robust controller, tailored to manage a wide spectrum of time delays. Concurrently, Wang et al. [90] introduced a distributed model predictive control (DMPC) strategy to bolster resilience against communication

delays. This method emphasizes synchronization and a consistent spacing policy, aiming to minimize the impact of communication delays on time headway selection, and integrates robustness and string stability into a dual-mode framework.

In the realm of measurement noise, Ploeg et al. [91] developed a Kalman filter-based approach to mitigate the effects of packet data loss, enabling the prediction of the leading vehicle's acceleration by fusing sensor data with acceleration models. Desjardins et al. [92] investigated the application of a reinforcement learning agent, employing function approximation and gradient-descent algorithms for the management of CACC systems. Shi et al. [93] assessed the potential of deep reinforcement learning in handling mixed traffic scenarios, albeit without accounting for uncertainties. Our analysis identifies a research gap in examining the impact of combined sensor and communication uncertainties and contrasts a benchmark solution, such as MPC, with more advanced methodologies[2].

3.5.1 Evaluation Metrics

A string of vehicles is assumed to be stable if, for any set of bounded initial disturbances to all the vehicles, the position fluctuations of all the vehicles remain bounded and these fluctuations approach zeros. These perturbations may originate from a number of sources, including external factors such as road conditions or surrounding traffic dynamics, as well as internal factors such as control system delays, communication delays, sensor noise. Based on [93], a dampening factor was employed as criterion to evaluate the string stability. Specifically, the ratio between the relative L2 norm of the acceleration response of each vehicle and the one of its preeceding vehicle was considered as detailed

[2]Part of the present chapter has been extracted from a conference paper under review: Seifoddini, A.; Azad, A.; Musa, A.; Misul, D. Design of a Decentralized Control Strategy for CACC Systems Accounting for Uncertainties. SAE CO2 Reduction for Transportation Systems Conference, 2024.

in the following:

$$D_{rel(p,i)} = \frac{\|\ddot{x}_i^t\|_2}{\|\ddot{x}_{i-1}^t\|_2} = \frac{\sum_{t=0}^{N} |\ddot{x}_i^t|^2}{\sum_{t=0}^{N} |\ddot{x}_{i-1}^t|^2} \tag{3.10}$$

3.5.2 Comparison Assumption

Communication System Model

CACC system depends on a communication network between vehicles and/or with infrastructure. In this system, each vehicle equipped with CACC transmits information such as speed, acceleration, and other relevant data to nearby vehicles. However, this network is often subject to non-ideal characteristics, which can markedly impact the performance and feasibility of the CACC system itself.

In the current research, two principal sources of communication nonlinearity were examined, namely communication delays and packet data losses. The formers arise from various interrelated factors. Propagation delay, for instance, varies with the distance between vehicles, affecting the time required for signal transmission. Additionally, network latency, which refers to the time for data packet processing and transmission through communication networks, plays a significant role. Further, the processing time of data within each vehicle's onboard systems also contributes to these delays. Similarly, also packet data loss arises from multiple sources. Interference, whether from external devices or obstacles, disrupts vehicle communication, while network congestion in densely populated areas can overwhelm the system. Additionally, increased vehicle separation leads to signal attenuation, and data packets may become corrupted during transmission. To understand and address this challenge, researchers use approaches like Markov models, which assess the probability of packet loss, and simulations, as well as analysis of real-world data, to evaluate and improve system robustness.

Based on MATLAB documentation[94], we employed a probabilistic model to replicate a WIFI network. This model accounts for the stochastic delay linked to transmitting data stored in the network buffer, employing a uniformly

distributed random variable. Additionally, it addresses the possibility of network errors, which can result in periods of downtime or interruptions. Concerning packet loss, the probability of such loss is compared to another uniformly distributed random variable to determine whether a given packet is lost or retained. This approach allows for a realistic simulation of the network's behavior, considering both delay and potential data loss scenarios.

Benchmark

As mentioned previously, the current study examines a decentralized configuration for a CACC system involving five vehicles, taking into account various types of uncertainties. A decentralized control system is a control strategy where decision-making and control actions are distributed among multiple controllers, each having authority over a specific part of the system. This approach contrasts with a centralized control system, where a single controller makes decisions and executes control actions for the entire system. An MPC algorithm has been selected as benchmark solution. The state vector is hereafter reported.

$$X_i = \begin{bmatrix} e_{d_i} \\ e_{\dot{x}_i} \\ e_{\ddot{x}_i} \end{bmatrix} = \begin{bmatrix} x_{i-1} - x_i - d_{s,i} \\ \dot{x}_{i-1} - \dot{x}_i \\ \ddot{x}_{i-1} - \ddot{x}_i \end{bmatrix} \tag{3.11}$$

$$d_{s,i} = d_0 + \dot{x}_i \cdot h_d \tag{3.12}$$

Where e_{d_i} represents the error between the relative distance and the safe inter-vehicular distance (d_s); the latter has been computed considering a constant time headway (h_d) policy that considers the actual vehicle velocity in addition to the default safety distance d_0. The terms time headway and time gap are employed interchangeably throughout this work to maintain technical consistency. A constant time gap (h_d) spacing policy was selected because it effectively simplifies the control law under varying speed conditions, facilitating a more predictable and stable vehicular following behavior across diverse traffic scenarios. Constant time gap policies have been widely validated in literature for providing sufficient responsiveness while ensuring safety, even under tight traffic and rapid speed change conditions. Furthermore, employing a constant time gap approach aligns with the decentralized nature of the system

design, as it allows individual vehicles to react based on locally available data without necessitating frequent communication with a central controller or other vehicles, thus reducing system latency and improving the robustness against communication failures.

Additionally, $e_{\dot{x}_i}$ is the error between the relative velocity and the reference value, which is set to zero, and $e_{\ddot{x}_i}$ denotes the error between relative acceleration and the reference value, also maintained at zero. The state-space representation of the system is given by:

$$\dot{X} = AX + BU + EW \tag{3.13}$$

$$Y = CX + DW \tag{3.14}$$

where X is the state vector, U is the input vector, W is the disturbance vector, and Y is the output vector. The detailed matrices for our system are:

$$\begin{aligned} \dot{X} = & \begin{bmatrix} 0 & 1 & h_d \\ 0 & 0 & 1 \\ 0 & 0 & -\frac{1}{\tau} \end{bmatrix} X \\ & + \begin{bmatrix} 0 & 0 & -h_d \\ 0 & 0 & 0 \\ -\frac{1}{\tau} & \frac{1}{\tau} & 0 \end{bmatrix} \begin{bmatrix} U_i \\ U_{i-1} \\ \ddot{x}_{i-1} \end{bmatrix} \\ & + W_d \end{aligned} \tag{3.15}$$

$$\begin{aligned} Y = & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} X \\ & + W_n \end{aligned} \tag{3.16}$$

U_i denotes the acceleration command of vehicle i which is the manipulated variable, U_{i-1} represents the acceleration command of vehicle $i-1$ which is considered the measured disturbance alongside the actual acceleration $\ddot{x}_{i-1}$. Additional noise terms called W_d and W_n are introduced which account for process disturbances that arise from the inaccuracies of the model and measurement noises that result from inaccuracy of the sensors respectively. The

control problem was set up to optimize a cost function given by the sum of three terms representative of tracking performance, control effort and variation in the control effort required as done in [15].

$$\begin{aligned} J = &(Y - Y_{\text{target}})^T W_y (Y - Y_{\text{target}}) \\ &+ (U - U_{\text{target}})^T W_{\Delta u} (U - U_{\text{target}}) \\ &+ \Delta U^T W_u \Delta U, \end{aligned}$$

where Y and Y_{target} represent the vectors of predicted and target outputs, respectively, over the prediction horizon N_p; U and U_{target} are the vectors of actual and target control inputs, respectively; ΔU is the change in control actions; and W_y, $W_{\Delta u}$, and W_u are the weighting matrices for output error, input deviation, and input change, respectively.

Simulation settings

During the calibration and training phases, encompassing the tuning of MPC parameters and the training of the RL agent respectively, a sinusoidal input signal was employed, characterized by an amplitude of 1 m/s^2 and a frequency of 0.05 Hz. Additionally, two sensitivity analyses were conducted, focusing on the time gap and latency. Specifically, the time gap analysis examined values ranging from 0 to 1.6 s, while the latency analysis explored a range from 0 to 1000 ms. The specific values used are detailed in Table 3.5. In the context of this paper, the time gap is the time interval maintained between vehicles for safety, allowing sufficient reaction and braking time to avoid collisions, which varies with vehicle speed. This analysis refers to a constant time gap spacing policy, which adjusts the following distance based on vehicle speed to ensure safety and improve traffic flow.

3.5.3 Vehicle Model

In the present study, a configuration consisting of five vehicles is investigated, structured in a string including one lead vehicle and four follower vehicles, all operating under a cooperative adaptive cruise control system. Each vehicle

Table 3.5 Simulation settings

Phase	**Parameter**	**Value**
Training	Input signal	$\sin(\omega t), \omega = 0.1\pi$
	Time gap, s	0, 0.5, 1.1, 1.6
	Latency, ms	0, 300, 700, 1000
	Control input range, m/s^2	$[-3, 2]$
Testing	Driving cycle	WLTP
	Curb Weight, kg	1474
	Battery capacity, kWh	42

within this configuration is designated by an index i, where i is defined within the range of 0 to 4 (Figure 3.4).

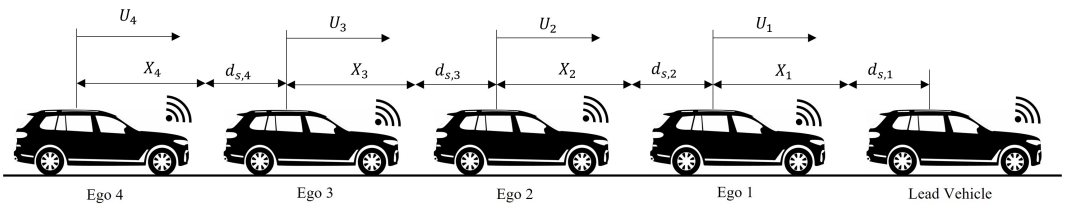


Fig. 3.4 Scheme of the driving scenario and information flow topology

To narrow the scope of the investigation, the analysis is limited to the first-order longitudinal dynamics of these vehicles. This approach intentionally excludes the lateral dynamics and the potential for overtaking scenarios. The rationale behind this limitation is to mitigate the impact of model non-linearities on the efficacy of the control algorithm.

The dynamic behavior of the i_{th} vehicle in this arrangement is governed by the following set of equations:

$$\tau \frac{d}{dt}\ddot{x}(t) + \ddot{x}(t) = u(t) \tag{3.17}$$

$$u_{min} \leq u(t) \leq u_{max} \tag{3.18}$$

Where τ represents the time lag of the vehicle (the acceleration command from the controller cannot be tracked instantaneously due to actuators delay).

3.5.4 Results

The results are presented in two sections. The first one refers to the calibration phase for MPC and the training phase for the RL agent, as previously mentioned. Here, the tuning of MPC parameters and the training of the RL agent are performed under a sinusoidal input signal.

Calibration and Training Results

Relative velocity and dampening factor in presence of network latency without sensor noise Tables 3.6-3.7 showcase the results of a MPC system employing a sinusoidal function as input under different network latency conditions without sensor noise. The dampening factor, defined as the ratio of the L2 norm of the acceleration response between successive vehicles, serves as a measure of string stability within a vehicular platoon. For the scenario without latency, the dampening factor values are relatively stable across four different assessments, starting at approximately 1.01 at a time gap of 0 seconds and decreasing to around 0.79 by 1.6 seconds. This indicates a slight increase in stability (lower dampening factor) as the time gap increases.

In contrast, introducing a latency of 1000 ms results in notably higher initial damping factors (ranging from 1.16 to 1.23 at 0 seconds), which follow a similar trend of decreasing as the time gap increases. The initial increase suggests that latency introduces instability but observing the trend, the system still manages to stabilize over time, albeit less efficiently than in the latency-free scenario. In Fig. 3.5–3.6, we present the velocity profiles of the last vehicle compared to the lead vehicle across different time gaps using the MPC and RL approaches respectively, illustrating the key trends.

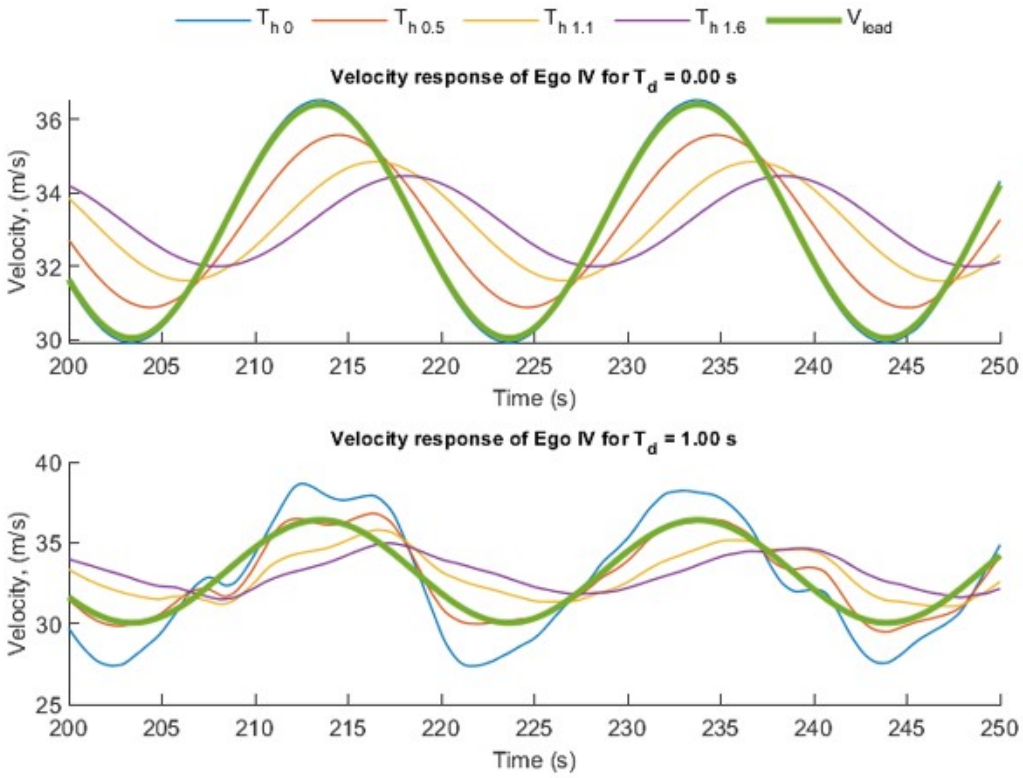


Fig. 3.5 Comparative MPC results with sinusoidal input under 0 ms latency (top) and 1000 ms latency (bottom), without sensor noise.

Table 3.6 MPC results using the sinusoidal function as input and considering a latency of 0 ms without sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	1.011	1.01	1.01	1.01
0.5	0.93	0.93	0.93	0.93
1.1	0.85	0.85	0.85	0.85
1.6	0.79	0.79	0.79	0.79

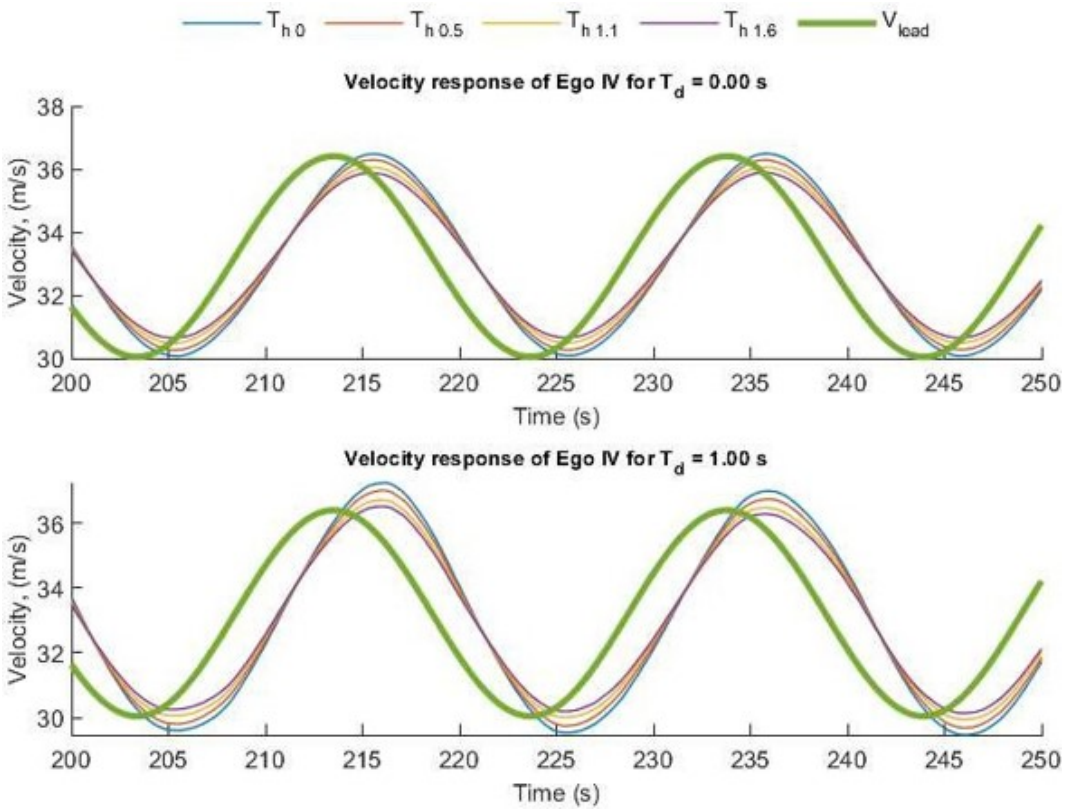


Fig. 3.6 Comparative RL results with sinusoidal input under 0 ms latency (top) and 1000 ms latency (bottom), without sensor noise.

Table 3.7 MPC results using the sinusoidal function as input and considering a latency of 1000 ms without sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	1.16	1.16	1.17	1.23
0.5	1.04	1.041	1.06	1.15
1.1	0.91	0.91	0.91	0.96
1.6	0.83	0.83	0.83	0.84

The RL approach maintains relatively higher dampening factors under no latency conditions compared to the MPC, indicating less variation in stability. However, under latency, both approaches exhibit an increase in dampening factors initially, with a gradual decrease over time. The MPC method appears to adapt slightly better to latency with the increase in time gap, as evidenced

by a more significant decrease in dampening factors, suggesting it may offer marginally improved stability in latency conditions without sensor noise (Tables 3.8–3.13).

Table 3.8 RL results using the sinusoidal function as input and considering a latency of 0 ms without sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	1.01	1.01	1.01	1.01
0.5	0.98	0.98	0.98	0.98
1.1	0.97	0.97	0.97	0.97
1.6	0.95	0.95	0.95	0.95

Table 3.9 RL results using the sinusoidal function as input and considering a latency of 1000 ms without sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	1.04	1.04	1.04	1.04
0.5	1.03	1.03	1.03	1.03
1.1	1.01	1.01	1.01	1.01
1.6	0.99	0.99	0.99	0.99

Relative velocity and dampening factor in presence of network latency and sensor noise

Introducing sensor noise into the system affects both the MPC and RL strategies, leading to varied impacts on the dampening factor under conditions of network latency.

For the MPC approach, the presence of sensor noise slightly increases the dampening factors at zero latency, indicating a marginal decrease in stability. Under latency conditions, the dampening factors increase more significantly, especially at shorter time gaps, suggesting that latency combined with sensor noise exacerbates instability more than latency alone.

In contrast, the RL results show a generally lower dampening factor even with sensor noise, indicating a better maintenance of stability relative to the

MPC approach. However, as latency is introduced, both approaches exhibit increased dampening factors, with RL showing a gradual increase, suggesting that while RL is affected by sensor noise and latency, it potentially adapts better to these conditions compared to MPC. Overall, sensor noise introduces additional challenges to vehicular platoon stability.

For completeness, in Fig. 3.7–3.8, we present the velocity profiles of the last vehicle compared to the lead vehicle across different time gaps using the MPC and RL approaches respectively, illustrating the key trends.

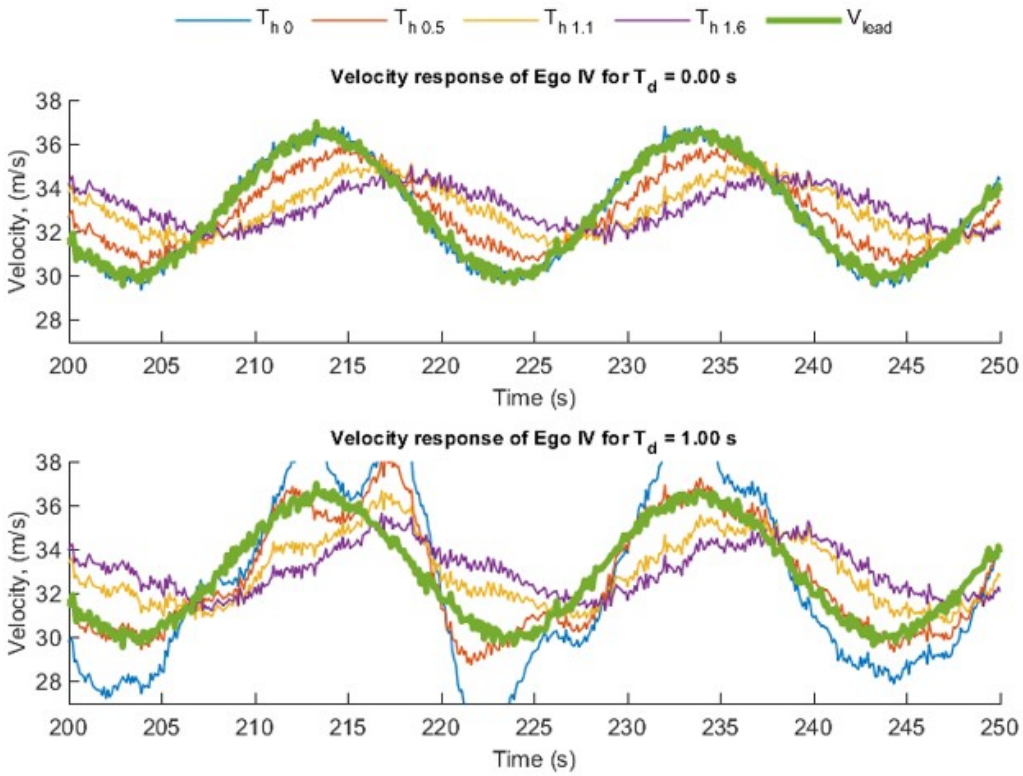


Fig. 3.7 Comparative MPC results with sinusoidal input under 0 ms latency (top) and 1000 ms latency (bottom), with sensor noise.

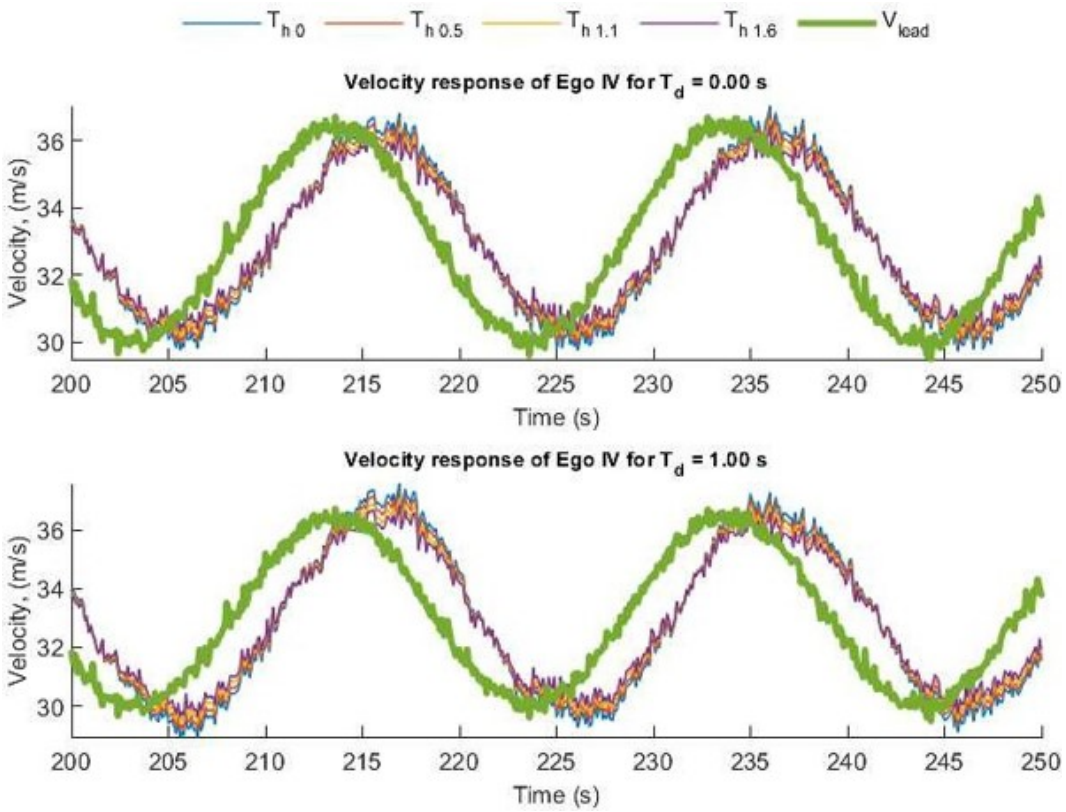


Fig. 3.8 Comparative RL results with sinusoidal input under 0 ms latency (top) and 1000 ms latency (bottom), with sensor noise.

Table 3.10 MPC results using the sinusoidal function as input and considering a latency of 0 ms with sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	1.02	1.01	1.01	1.01
0.5	0.94	0.93	0.93	0.93
1.1	0.86	0.86	0.86	0.87
1.6	0.81	0.81	0.83	0.84

Table 3.11 MPC results using the sinusoidal function as input and considering a latency of 1000 ms with sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	1.18	1.19	1.21	1.28
0.5	1.05	1.07	1.10	1.21
1.1	0.92	0.94	0.973	1.02
1.6	0.85	0.87	0.92	0.94

Table 3.12 RL results using the sinusoidal function as input and considering a latency of 0 ms with sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	0.97	1.01	1.0	1.0
0.5	0.96	1.0	0.98	0.98
1.1	0.94	0.98	0.97	0.97
1.6	0.93	0.97	0.96	0.96

Table 3.13 RL results using the sinusoidal function as input and considering a latency of 1000 ms with sensors noise

Time Gap(s)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
0	1.01	1.05	1.04	1.04
0.5	1.0	1.03	1.03	1.02
1.1	0.98	1.02	1.03	1.01
1.6	0.97	1.01	1.0	1.01

As an observation, the RL-based controller's damping ratio is less affected by sensor noise and latency compared to the MPC controller, as evidenced by the smaller percentage increase in damping ratios under uncertainty (Table 3.6 vs. Table 3.11 for MPC and Table 3.8 vs. Table 3.13 for RL). Specifically, uncertainties caused a 16% increase in the 4th relative damping for the MPC controller, against a 6% increase for the RL controller. These results indicate

the RL controller's lower sensitivity to uncertainties, suggesting the need for revisiting the objective function and calibration of the RL agent to address performance issues.

Testing on WLTP

The calibrated MPC and the trained agent were evaluated using the WLTP driving cycle, employing a forward-looking simulation methodology and focusing on a battery electric vehicle powertrain technology, as previously done in [79**?** , 15]. Interested readers are directed towards those publications for the vehicle model main equations. As in the previous sections, Tables 3.14-3.17 report the results for MPC and RL when tested on the WLTP driving cycle, in scenarios without and with sensor noise. Both strategies show increased dampening values with increased latency, implying reduced stability. Specifically, in the scenario that includes sensor noise, the damping values for the MPC approach at 0 ms latency range from 0.81 to 0.91. At 1000 ms latency, these values rise between 0.91 and 0.97. This gradual increase suggests a reduction in stability as the latency increases; however, all values remain below the instability threshold of 1. The RL strategy shows a similar trend of increasing dampening values with increased latency, but it starts and remains at a higher level across all latency levels compared to MPC. For instance, at 0 ms latency, the dampening values range from 0.91 to 0.96, and at 1000 ms latency, the values are between 0.96 and 1. Notably, the dampening value reaches 1 at 1000 ms latency, indicating a threshold where the system could be considered unstable.

MPC offers a more stable control strategy across the tested latency ranges for this specific vehicle powertrain technology and driving cycle. While both strategies exhibit reduced stability with increased latency, RL's generally higher dampening values, especially at lower latencies, suggest it is less stable than MPC. Both the MPC and RL control strategies were initially calibrated and trained using a sinusoidal signal to expedite the calibration and training phases. Despite the sinusoidal characteristics significantly differing from those of the WLTP driving cycle, this approach demonstrated the strategies' adaptability and potential in managing string stability under uncertainties. The successful application of both strategies under conditions not directly mirrored in their training highlights their capability to generalize and perform effectively in

diverse and uncertain environments, laying a promising foundation for further optimization and application in automotive control systems. In general, for tracking control problems, a feedback-based algorithm like MPC tends to perform better due to its predictive capabilities and ability to handle constraints effectively. This is expected because MPC can optimize control actions over a future horizon, making it highly effective for such applications. However, as we consider the evolution towards scenarios involving nonlinear vehicle dynamics, RL emerges as a promising alternative. Despite potentially offering slightly inferior performance in its current state, RL has the potential to address complex nonlinear problems with comparable efficacy to MPC. This is largely due to RL's ability to learn from interactions with the environment, adapting its strategy to maximize a defined reward function. Moreover, the inherent flexibility and learning capability of RL could simplify dealing with the complexity that an MPC-based strategy would face in such nonlinear contexts. It is also worth noting that there is significant room for improvement in both the reward function design and calibration processes for RL, as well as in the formulation and computational efficiency of MPC, which could further enhance their applicability and performance in future applications.

Table 3.14 MPC results on WLTP driving cycle without sensor noise

Time Gap (s)	Latency (ms)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
1600	0	0.79	0.82	0.86	0.89
1600	300	0.81	0.83	0.86	0.88
1600	700	0.85	0.86	0.87	0.88
1600	1000	0.9	0.9	0.9	0.91

Table 3.15 RL results on WLTP driving cycle without sensor noise

Time Gap (s)	Latency (ms)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
1600	0	0.94	0.95	0.95	0.95
1600	300	0.86	0.96	0.96	0.96
1600	700	0.98	0.98	0.98	0.98
1600	1000	0.99	1	1	1

Table 3.16 MPC results on WLTP driving cycle with sensor noise

Time Gap (s)	Latency (ms)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
1.6	0	0.81	0.84	0.88	0.91
1.6	300	0.83	0.86	0.89	0.91
1.6	700	0.87	0.89	0.92	0.94
1.6	1000	0.91	0.94	0.96	0.97

Table 3.17 RL results on WLTP driving cycle with sensor noise

Time Gap (s)	Latency (ms)	Relative Dampening 1	Relative Dampening 2	Relative Dampening 3	Relative Dampening 4
1.6	0	0.91	0.96	0.96	0.96
1.6	300	0.93	0.98	0.97	0.97
1.6	700	0.96	0.99	0.99	0.99
1.6	1000	0.96	1	1	1

Chapter 4

AI techniques for battery management

The present chapter focuses on the use of machine learning algorithms to estimate the state-of-health (SOH) of high-voltage batteries. The analysis is based on open-circuit voltage (OCV) measurements from 12 electric vehicles with different mileage conditions. SOH is a critical metric that assesses the overall condition of a battery, reflecting its capacity to store energy compared to a new battery. While SOH can also be influenced by factors such as charge and discharge rates, temperature influences, and other variables, this study specifically addresses capacity as the main indicator. The goal is to determine a correlation between the OCV values and the battery SOH. The experimental campaign along with the algorithm development was conducted in the framework of a master thesis work carried in collaboration with an industrial partner[1]. Six machine learning algorithms are evaluated and compared based on the company requirements.

[1]Part of the present chapter has been extracted from [9].

4.1 Introduction

4.1.1 Motivation

In electrified vehicles, battery management systems play a critical role by performing various tasks such as tracking the state of charge (SOC) and state-of-health (SOH), regulating thermal conditions for peak performance, safeguarding against overcharging and overdischarging, and maintaining cell balance within the battery pack [95–102]. These functionalities are designed to enhance battery performance, extend its lifespan, and minimize maintenance and replacement expenses, thereby rendering electrified vehicles more affordable and appealing to consumers.

4.1.2 Contribution

Three perspectives are contributed to the related literature:

1. Establish a correlation between the OCV values and the energy stored in the battery considering experimental data, allowing for the determination of the SOH.

2. Compare different machine learning methodologies.

3. Providing detailed analysis of the employed ML algorithms, highlighting their key differences in terms of complexity, performance, interpretability, data requirements, and preprocessing steps.

4.1.3 Related works

There are two principal approaches to assess battery SOH in high-voltage (HV) batteries: estimation and prediction. SOH estimation utilizes measurements of battery current and voltage, along with a range of diagnostic evaluations, to estimate the current SOH of the battery. This assessment is crucial for determining the operational status of the battery and deciding if maintenance or replacement is necessary. On the other hand, SOH prediction employs historical data and predictive modeling to forecast the battery's future performance and

remaining useful life. This involves analyzing patterns of battery degradation over time and projecting these patterns into the future. The aim of SOH prediction is to predict the remaining lifespan of the battery and to assist in planning for maintenance and replacement [103–106]. Estimation and prediction play a pivotal role in the management of HV batteries, key to maximizing their performance and longevity. SOH assessment can be achieved through model-based or data-driven approaches. Model-based approaches utilize mathematical models to mimic the electrochemical behaviors within the battery. On the other hand, data-driven approaches employ statistical methodologies and AI algorithms. While data-driven methods are flexible, their estimation accuracy hinges on the chosen features and algorithms. Each approach possesses distinct advantages and drawbacks, with the preference for either depending on the specific needs of the intended application.

Model-based approaches to battery management encompass a variety of techniques, such as equivalent circuit models [107–109], electrochemical models [110, 111, 103, 112], and grey box models [113]. Equivalent circuit models feature the battery as an arrangement of resistances and capacitors [107, 108]. By tracking the battery's voltage and current over time, these models are capable of assessing the SOH of the battery. In contrast, electrochemical models focus on replicating the internal chemical and physical processes of the battery. These processes include ion and electron transport, chemical reactions, and heat generation. Electrochemical models are known for their high accuracy in predicting battery behavior. However, they demand significant computational resources, generally unfeasible for real-time applicationss [110, 111, 103, 112]. Grey-box models like the extended Kalman filter blend empirical data with mathematical constructs to accurately predict battery behavior. These models, though simpler than their electrochemical counterparts, are effective in predicting battery performance. They exploit performance data and mathematical frameworks for internal battery parameters estimation, including SOC and internal resistance [113]. However, a primary challenge of model-based algorithms lies in their complexity. Developing precise models to depict battery behavior is often not straightforward and time-consuming. Moreover, these algorithms are highly sensitive to the accuracy of model parameters, with any misestimation leading to unreliable predictions. Another drawback is their limited adaptability to different battery systems, as they are

typically designed for specific battery chemistries under defined operational conditions. Examples of data-driven methods include a support vector machine (SVM) [114–118], random forest (RF) [119, 120], artificial neural networks (ANNs) [121–125], recurrent-neural networks (RNNs) [126] and variants i.e., LSTM [127–132] and a nonlinear autoregressive network with exogenous inputs (NARX) [133–135]. ANNs predict future behavior based on a learning step on historical data, i.e., a dataset representative of the battery behaviour. Both NARX and LSTM are examples of RNNs although there are some differences between the two architectures in the way they handle the temporal dependencies present in the input data. NARX takes advantage of a feedback loop to propagate information from previous time steps to the current one [133–135], whereas LSTM uses a complex memory cell designed to retain or discard information from previous time steps [127–132]. The primary disadvantages of data-driven approaches include the necessity for large amounts of data for training and significant computational resources. Choosing a data-driven method hinges on the unique needs of the application, considering aspects like data availability, computational resource demands, and the complexity of the model. Although each of the presented methods is well known in the literature, there is still a lack of comparative analyses and performance evaluations when real data are employed.Therefore, our study focused on the use of several machine learning algorithms to estimate the SOH of HV batteries in electric vehicles considering real data.

4.1.4 Outline

The chapter is structured as follows: Section 4.2 briefly outlines the problem under investigation. Section 4.3 discusses the algorithms utilized. Finally, Section 4.4 describes the selected test case.

4.1.5 Notation

This paragraph outlines the main symbols and their domains employed in the analysis of battery SOH for electric vehicles, however it can easily be extended

to hybrid electric powertrains. We define the State of Health (SOH) as

$$SOH = \frac{C_{\text{actual}}}{C_{\text{BOL}}} \in \mathbb{R},$$

where C_{actual} represents the current capacity of the battery, and C_{BOL} denotes the initial capacity at the beginning of life, both measured in ampere-hours (Ah). The capacity at various States of Charge (SOC), C_i, is calculated as

$$C_i = \frac{E_{\text{stored}}}{n_{\text{cells}} \cdot v_{\text{ocv}}},$$

with E_{stored} in watt-hours (Wh), n_{cells} as the count of cells, and v_{ocv} the open-circuit voltage in volts (V). The terminal voltage (V) and open-circuit voltage (v_{ocv}) are expressed in volts (V).

4.2 Problem Formulation

Given a dataset $\{(x_i, y_i)\}_{i=1}^{N}$, where $x_i \in \mathbb{R}^d$ represents the feature vector associated with the i-th observation (e.g., various battery usage parameters), and $y_i \in \mathbb{R}$ is the actual State of Health (SOH) for that observation, the goal is to train a model $f : \mathbb{R}^d \to \mathbb{R}$ to predict the SOH, y, based on a new set of battery parameters, x. The prediction made by the model for the i-th observation is denoted as $\hat{y}_i = f(x_i)$.

The loss for each individual prediction is defined as the squared difference between the actual SOH and the predicted SOH:

$$L(f(x_i), y_i) = (\hat{y}_i - y_i)^2$$

To evaluate the performance of the model across the entire dataset, common choices include the mean squared error (MSE) among the others, which is the average of these individual losses:

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$$

The objective in training the model is to minimize the MSE:

$$\min_{f \in \mathcal{H}} MSE = \min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$$

where $\mathcal{H}$ represents the hypothesis space, or the set of all possible models that could be learned given the model architecture and training data.

4.3 Algorithms

In this work, the algorithms that were investigated include linear regression (LR), k-nearest neighbors (KNN), classification and regression tree (CART), random forest (RF), support vector machine (SVM), and dense neural network (DNN).

4.3.1 Linear Regression

Linear Regression — LR is a statistical method employed to establish a linear connection between a dependent variable and an independent variable. In contrast, multiple linear regression explores the connection between a dependent variable and several independent variables, while simple linear regression concentrates on an independent variable only.

4.3.2 SVM

Support vector machine (SVM) — SVM operates by seeking an optimal hyperplane to effectively separate data into distinct classes or estimate a target variable. In SVM, data undergoes transformation into a high-dimensional space, where the primary goal is to define a hyperplane that maximizes the margin between data points and the hyperplane itself. This margin, indicating the distance between the nearest data points and the hyperplane, plays a pivotal role in delineating boundaries between different classes. In regression scenarios, the hyperplane is employed to estimate the target variable.

Algorithm 4 Support Vector Machine Training

Require: Training dataset $D = \{(\mathbf{x}_i, y_i)\}$, $i = 1, \ldots, N$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
Require: Regularization parameter $C > 0$
Require: Kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$
  Initialize Lagrange multipliers $\alpha_i = 0$, $\forall i$
  Initialize threshold $b = 0$
  while termination criteria is not met **do**
    for each α_i in α **do**
      Select α_i and α_j for optimization
      Compute $E_i = f(\mathbf{x}_i) - y_i$ and $E_j = f(\mathbf{x}_j) - y_j$
      Update α_i and α_j using the SMO algorithm steps
      Update threshold b
    end for
    Check for convergence
  end while
  Compute the weight vector $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$ (for linear SVM)
  The decision function is $f(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$ for linear SVM
  For non-linear SVM, use $f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right)$

4.3.3 KNN

K-nearest neighbors — KNN stands as a non-parametric supervised learning classifier, harnessing proximity to make classifications or estimations regarding the grouping of individual data points. Though it can be applied to both regression and classification tasks, its primary use lies in classification. The core principle underlying KNN is the assumption that data points sharing similar attributes tend to cluster together in close proximity.

Algorithm 5 k-Nearest Neighbors Algorithm

Require: Training dataset D, test instance x, number of neighbors k
Ensure: Predicted class label for x
  Initialize an empty list *neighbors*
  for each instance x_i in D **do**
    Compute distance $d(x, x_i)$
    Add $(d(x, x_i), \text{class of } x_i)$ to *neighbors*
  end for
  Sort *neighbors* by distance in ascending order
  Take the first k elements from *neighbors*
  Count the frequency of each class in these k elements
  Assign the class with the highest frequency to x
  return Class label for x

4.3.4 CART

Classification and regression tree (CART) — CART belongs to the category of decision tree methods, where it creates a tree-shaped model for making predictions using input features. The tree's construction involves iteratively dividing the data into smaller subsets based on the feature that minimizes the error most effectively. This recursive process continues until the tree reaches a predefined stopping condition. Predictions are derived by following the path from the tree's root to a leaf node, where the final estimation is determined based on the class label or regression value assigned to that specific node.

Algorithm 6 Classification and Regression Tree (CART) Algorithm

Require: Training dataset D
Ensure: Decision tree model
  Initialize tree T as a single node representing dataset D
  while termination conditions not met **do**
    for each terminal node N in T **do**
      if all instances in N belong to the same class OR no feature improves the split **then**
        Mark N as a leaf node
      else
        Find the best feature and value to split the data in N
        Split N into child nodes based on the best split
        Add child nodes to T
      end if
    end for
  end while
  return Decision tree T

4.3.5 DNN

Dense neural network (DNN) — A DNN consists of an input layer, hidden layers, and an output layer. The input layer receives the input data, whereas the hidden layers process the data through a series of computations known as activation functions. The output layer provides the final estimation based on the processed data. The hidden layers use weights and biases to transform the data. Such parameters are acquired through a process called backpropagation, which adjusts the weights and biases to minimize the estimation error. DNNs can handle complex non-linear relationships between the features and targets variable and can model high-dimensional data. These characteristics, combined with the relative simplicity of DNNs in comparison to more complex architectures such as alternative backpropagation neural networks, RNNs, and their variants, prompted the authors to specifically opt for its implementation, taking into consideration the specific application and available dataset. Nevertheless, it is important to acknowledge that alternative choices could have been considered and pursued.

Algorithm 7 Dense Neural Network (Feed-Forward)

Require: Training dataset D, number of layers L, number of epochs E
Ensure: Trained neural network model
  Initialize network weights and biases randomly
  for each epoch in $1, 2, \ldots, E$ **do**
    for each batch (X, Y) in D **do**
      $A^{(0)} \leftarrow X$ {Input layer activation}
      for each layer l in $1, 2, \ldots, L$ **do**
        $Z^{(l)} \leftarrow W^{(l)} A^{(l-1)} + b^{(l)}$ {Linear transformation}
        $A^{(l)} \leftarrow \text{activation}(Z^{(l)})$ {Apply activation function}
      end for
      Compute loss between $A^{(L)}$ and Y
      Update weights and biases using backpropagation and gradient descent
    end for
  end for
  return Trained neural network

4.3.6 Random Forest

Random forest (RF) — RF belongs to the category of ensemble learning, wherein it aggregates multiple decision trees to arrive at a final estimation. In RF, numerous decision trees are generated, with each tree employing a distinct subset of the training data and a different set of features. These decision trees are subsequently amalgamated to produce a unified estimation. In classification tasks, this amalgamation involves taking a majority vote, while in regression, it entails averaging the individual estimations.

Algorithm 8 Random Forest Algorithm

Input: training dataset, number of trees N, number of features to consider at each split K
Initialize: an empty forest
for $i = 1$ **to** N **do**
  Create a bootstrap sample of the training dataset
  Grow a decision tree from the bootstrap sample
  for each node **do**
    Randomly select K features
    Split the node using the feature that provides the best split
  end for
  Grow the tree until termination criterion is met
  Add the tree to the forest
end for
return forest

4.4 Test case 1

4.4.1 Evaluation Metrics

These algorithms were analysed based on six indexes, namely usability, performance, interpretability, sensitivity to outliers, required data and preprocessing. The main considerations derived from the literature are summarized in Table 4.1.

Complexity — Complexity refers to the level of difficulty or the amount of resources that need to be implemented and use a given algorithm. It may be determined by a number of factors, including the number of parameters or hyperparameters that need to be set, the amount and quality of data required for the training and testing, the computational resources required, and the level of expertise needed to understand and use the algorithm effectively. LR is considered simple and easy to implement, with a straightforward optimization procedure. KNN is also considered simple, with a low number of hyperparameters that need to be tuned. CART is also considered simple to implement, but may require some fine-tuning of the hyperparameters to achieve optimal

Table 4.1 Comparison of ML algorithms.

Type	**Complexity** [1]	**Performance** [2]	**Interpretability** [3]	**Sensitivity to Outliers**	**Data Requirements**	**Preprocessing**
CART	Low	Medium	High	Low	Low–medium	Minimal
DNN	High	High	Low–Medium	Medium	High	Complex
KNN	Low	Low to Medium	Low	High	Low–medium	Minimal
LR	Low	Medium	High	High	Low–medium	Minimal
RF	Medium	High	Medium	Low–medium	Low–medium	Minimal
SVM	Medium	High	Low	Medium	Low–medium	Complex

[1] Understanding, implementing, and utilizing the algorithm; [2] Trade-off between accuracy and efficiency (i.e., computational complexity, processing times, and resource utilization); [3] Degree of transparency and understandability of the model estimations and decision-making process; this includes the ability to explain the relationship between the input features and the target output, as well as the ability to understand why a particular estimation was made

results. RF is more complex to implement than LR, KNN, and CART, but still relatively straightforward. SVM is considered relatively complex but highly effective for certain types of data. DNN is considered the most complex of all the algorithms, with a large number of hyperparameters that need to be tuned and a greater need for computational resources. However, the complexity of each algorithm can vary depending on the user's level of expertise and the specific implementation and configuration of the algorithm.

Performance — Performance refers to the ability to balance accuracy and efficiency, which involves making accurate predictions or classifications while minimizing computational complexity, processing times, and resource utilization. LR is fast to train and make estimations, but its performance may be limited by its linear assumption. KNN is relatively fast for small datasets, but may become slow for large datasets. CART is fast for both training and estimation, but may overfit for certain types of data. RF is relatively fast for both training and estimation, and often provides high accuracy. SVM is relatively slow to train, but highly accurate for certain types of data. DNN is computationally expensive to train, but can achieve state-of-the-art results for many problems.

Interpretability — Interpretability refers to the degree of transparency and understandability of the model estimations and decision-making process; it includes the ability to explain the relationship between the input features and the target outputs, as well as the ability to understand why a particular estimation was made. LR has a clear and straightforward interpretation, with coefficients representing the importance of each feature. KNN has limited interpretability, but its results can be visually represented and understood. The low interpretability derives from the fact that the model is not expressed as a mathematical equation or set of rules, but instead relies on the distances between points to make its estimations. This means that it can be difficult to understand why a particular estimation is made, or the relationship between the input features and the target output. Additionally, the estimation made by KNN depends on the choice of k (i.e., the number of nearest neighbors) and the weighting function used, making it less transparent and less interpretable than other algorithms. CART has a clear interpretation, with each split in the tree representing a decision based on the input features. RF is more difficult to interpret than CART, but its results can be visualized to some extent. SVM is difficult to interpret, with its decision boundaries being represented by complex

mathematical equations. DNN is the most difficult to interpret, with its results represented by a series of weighted connections between nodes.

Sensitivity to outliers — An algorithm is considered sensitive to outliers if its results are significantly impacted by the presence of outliers in the data (the term outlier indicates those points that are significantly different from the rest of the data). LR is generally sensitive to the presence of outliers. Indeed, the algorithm tries to fit a straight line to the data, and outliers can have a significant impact on the slope and intercept of the line. This can lead to a poor fit between the model and the data, resulting in inaccurate estimations. KNN is generally not very sensitive to outliers because the algorithm is based on voting among the k nearest neighbors, so a single outlier would not have a significant impact on the results. Nevertheless, KNN may be sensitive to outliers if k is small and the outlier is close to the data points being classified. RF is robust and insensitive to outliers. Individual decision trees are not significantly impacted by outliers because random forest is an ensemble method that constructs multiple decision trees and aggregates their estimations. However, if there is a large number of outliers, it may be necessary to preprocess the data in order to eliminate or reduce their impact. The CART algorithm is robust and insensitive to outliers. Individual splits are not significantly affected by outliers because CART is a tree-based method that recursively divides the data into smaller subsets based on the values of the features. These measures are designed to be insensitive to the presence of outliers. However, if there is a large number of outliers, it may be necessary to preprocess the data in order to eliminate or reduce their impact. SVM is less sensitive to outliers than linear regression. In fact, SVM tries to find the hyperplane that best separates the data into different classes, and it can effectively ignore outliers in the process. However, if there are many outliers, SVM may still produce inaccurate results. DNNs are less sensitive to outliers compared to linear regression. Neural networks are able to model complex relationships in the data, and they can effectively handle outliers in the data by adjusting the weights of the model. If there are many outliers, it may still be necessary to preprocess the data to remove or mitigate their impact. However, the sensitivity to outliers can vary depending on the specific implementation and configuration of each algorithm, as well as on the characteristics of the considered data.

Data requirements — Data requirements refer to the amount and type of data needed to successfully train a machine learning model. The data requirements can vary depending on the type of algorithm used. Several popular machine learning algorithms have different requirements for the size of the datasets on which they are trained. LR and KNN typically require small to medium-sized datasets, which can range from a few hundred to a few thousand data points. The size of the dataset needed for KNN may depend on factors such as the number of features, the number of classes, and the distribution of the data, whereas for LR it may depend on the complexity of the problem and the number of features in the dataset. Decision trees, such as CART and SVM, also require small to medium-sized datasets, which may depend on the complexity of the problem, the number of features, and the choice of kernel function. RF is similar in this regard, but the number of trees in the ensemble and the depth of each tree may also be relevant. DNNs typically require large datasets with from tens of thousands to millions of data points, depending on the complexity of the network architecture and the choice of activation functions.

Preprocessing — Data preprocessing is the manipulation of data prior to training in order to smooth the learning process of a specific algorithm. Generally, these algorithms are sensitive to inconsistent, missing, and noisy data, which prevents them from identifying the correct relationship between input and output variables. As an example, a duplicate or missing value may result in incorrect data statistics. Data cleaning and transformation are required. Data cleaning entails handling missing values, smoothing noisy data, removing outliers, and resolving inconsistencies. Data transformation entails altering the format, structure, and value of data through the use of procedures such as normalization and standardization. LR requires the data to be clean and properly formatted, which may involve dealing with missing values, handling outliers, and scaling the data if necessary. KNN is a non-parametric algorithm that requires the data to be normalized or scaled so that all features equally contribute to the distance metric used by the algorithm. KNN can be sensitive to noisy or irrelevant features, so that feature selection or engineering may be necessary. CART does not require much preprocessing, but it may be sensitive to noisy or irrelevant features. Feature selection or engineering may be necessary, and they can overfit the data, so regularization techniques may

be used to prevent overfitting. SVM requires properly preprocessed data, which may involve scaling, normalization, and handling missing values. SVM can be sensitive to noisy or irrelevant features, so that feature selection or engineering may be necessary. RF requires a similar preprocessing to that of decision trees, such as handling missing values, feature selection, and pruning. RF is relatively robust to noisy or irrelevant features, but scaling or normalization may be necessary. DNNs require a similar preprocessing to that of SVMs and may also require additional preprocessing steps, such as normalization, regularization, and data augmentation. DNNs are also sensitive to the choice of activation functions, which may require experimentation and fine-tuning.

4.4.2 Comparison Assumptions

4.4.3 Results

The dataset comprises data collected from 12 different vehicles, with 2.5 discharging cycles recorded for each vehicle. The SOC level and battery temperature were recorded at the start of each test, with SOC levels varied across seven different levels ranging from 100% to 15%. Table 4.2 provides a summary of the characteristics of the vehicles used in this project, focusing on the key features such as mileage, battery size, and battery voltage. This table serves as a quick reference for the reader and provides an overview of the data used in this study. As a remark, for confidentiality reasons, the exact number of cells cannot be disclosed, as it pertains to specific types of batteries.

As an example, Figure 4.1 shows a plot of the recorded OCV for $VEH1$ at different SOC levels for all the considered cells. The OCV generally decreases as the SOC decreases, which is consistent with the expected behavior of a lithium-ion battery. Both the OCV values and the cell numbers were properly normalized. More specifically, the OCV values were normalized by dividing each value by the OCV maximum, while the number of cells in each group was divided by the total number of cells in the battery. It should be noted that only slight deviations are observed among the various cells.

Table 4.2 Vehicle database.

	VEH1	**VEH2**	**VEH3**	**VEH4**	**VEH5**	**VEH6**	**VEH7**	**VEH8**	**VEH9**	**VEH10**	**VEH11**	**VEH12**
Mileage (10^3 km)	0.75	1.5	4	6.6	12.5	15.2	16.35	17	17.85	22	130	132
C_{batt} (kWh)	72.6	72.6	77.4	58	72.6	58	72.6	58	72.6	77.4	77.4	58
V_{batt} (V)	653	653	697	522.7	653	522.7	653	522.7	653	697	697	522.7

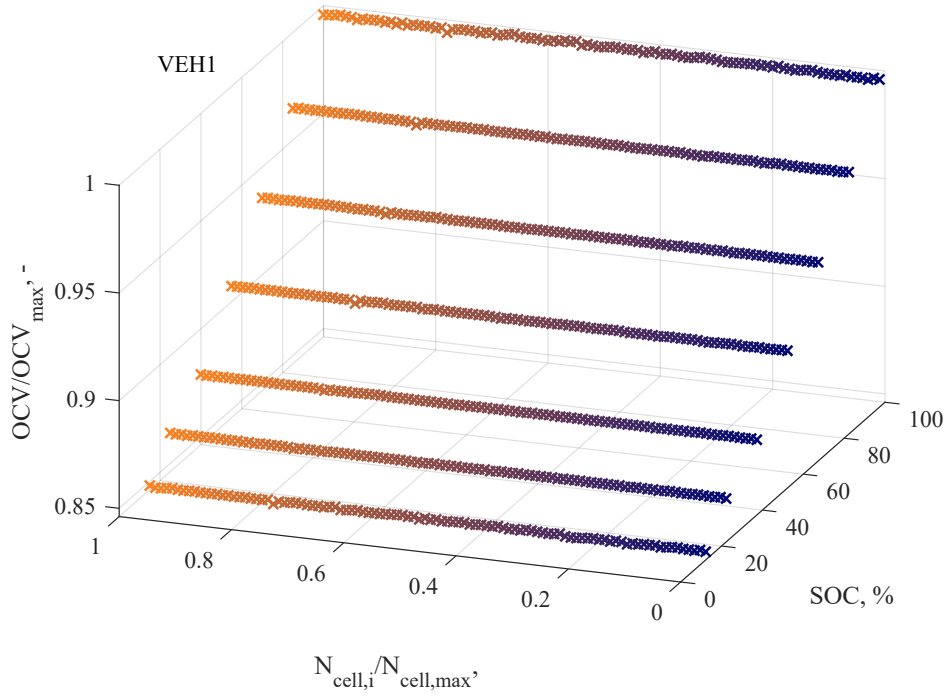
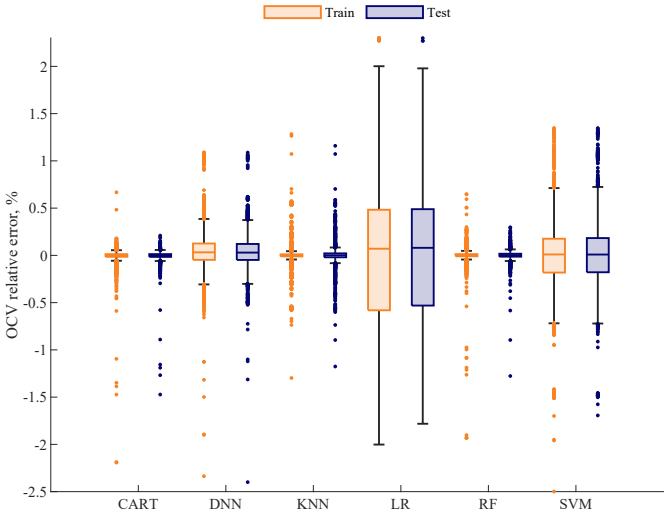


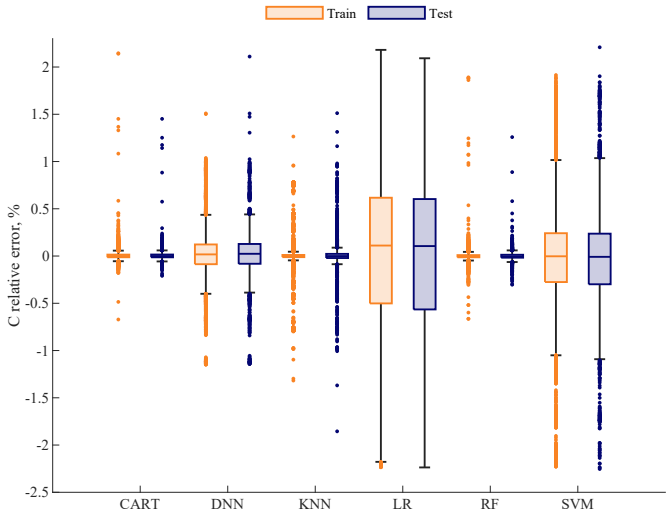
Fig. 4.1 OCV trend VEH1.

The selected algorithms underwent a hyperparametric optimization by analyzing the performance in terms of both training and testing. This comparison was performed to ensure that the selected combination offered good results for the algorithm as a whole and not just due to overfitting on the training data. Overfitting occurs when a statistical model exactly fits its training data, thus impairing the ability of the algorithm to accurately estimate unseen data. In order to avoid overfitting, hyperparameters that result in slightly higher error rates are commonly used. This improves the model's ability to generalize to new data. Figure 4.2 shows the % relative error resulting from hyperparameter optimization during training and testing for the OCV and C, respectively. In the box chart, it can be observed that both the OCV and C exhibit the highest errors for the LR and SVM algorithms, with a slightly worse performance in the case of C compared to OCV. The DNN algorithm follows, with moderately high errors. On the other hand, the RF, KNN, and CART algorithms provided the best performance among the considered models. In addition, for the KNN,

RF, and CART algorithms, except for the outliers, the errors are generally bounded between -0.15 and 0.15, which is an encouraging result. For the DNN algorithm, the errors are slightly larger, with a range from -0.4 to 0.4. In contrast, the SVM algorithm exhibits larger errors, with a range from -1.1 to 1.1, while the LR algorithm shows the largest errors with a range from -2.3 to 2.3.



(**a**)



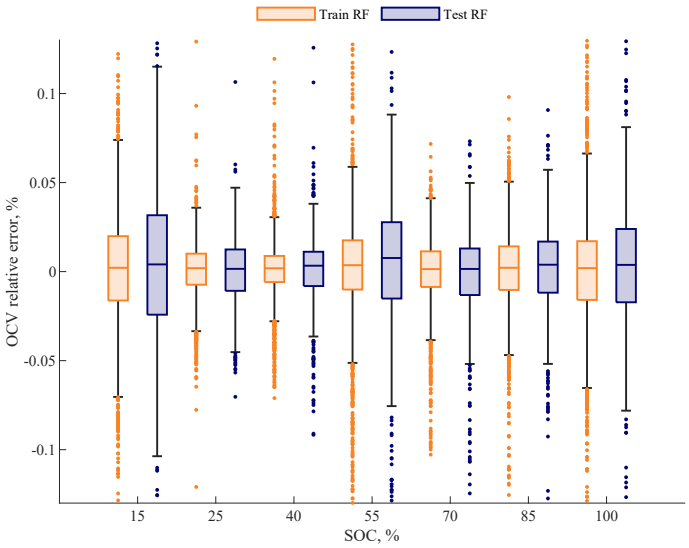
(**b**)

Fig. 4.2 (**a**) OCV relative error resulting from hyperparameters optimization in train and test for all the considered algorithms, (**b**) C relative error resulting from hyperparameters optimization in train and test for all the considered algorithms

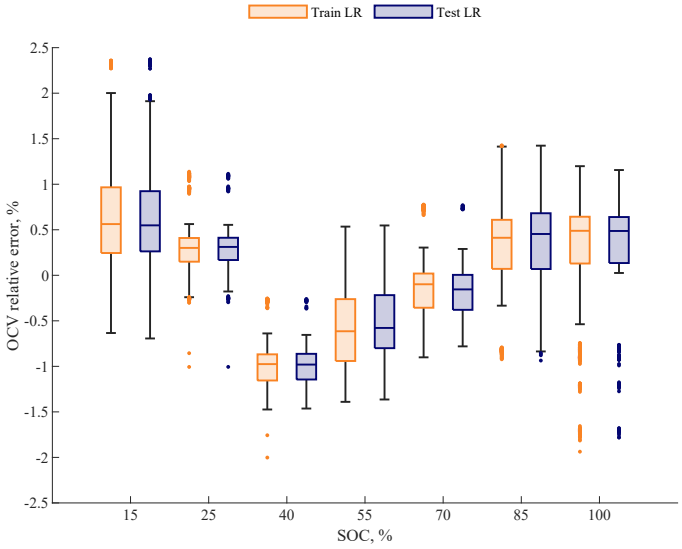
.

For the sake of completeness, a zoom on the OCV trend for all SOC values is reported for the algorithms with the best and worst performance, RF and

LR, respectively (Figure 4.3). It is worth noting that while the RF algorithm demonstrates a comparable performance across all SOC levels, the LR algorithm tends to overestimate the prediction accuracy for samples that have SOC values close to the mean or central SOC levels. The reason for this may be that the LR model is a linear model that assumes a linear relationship between the input variables and the target variable. Therefore, it might not be able to capture the non-linear relationship between the input variables and the outputs, thus producing biased results.



(**a**)



(**b**)

Fig. 4.3 (**a**) A zoom on the OCV relative error for the RF algorithm analyzed for all SOC values considered (**b**) A zoom on the OCV relative error for the LR algorithm analyzed for all SOC values considered.

Table 4.3– 4.4 presents a comparison of the errors generated by all algorithms. The table shows the average and maximum errors computed from the entire

dataset. The results indicate that the random forest (RF) algorithm outperforms the other algorithms in both $\bar{e}_{ocv}$ and $\bar{e}_c$ metrics. Specifically, the RF algorithm achieves the lowest values of both metrics, with only a 0.02% error rate, which is significantly lower than the other algorithms. The DNN algorithm also performs well, achieving the lowest value of $e_{max,ocv}$ with only a 2.4% error rate, but with slightly higher $\bar{e}_{ocv}$ and $\bar{e}_c$ than RF. In contrast, the LR algorithm has the highest $\bar{e}_c$ and $e_{max,ocv}$ values among all algorithms. The SVM algorithm shows a comparable performance with the KNN and CART algorithms in terms of both metrics. The table also includes the percentage improvement of RF over each algorithm. The RF algorithm demonstrates superior performance compared to other machine learning algorithms in various error rate measures. For instance, the RF algorithm achieves an average OCV error rate improvement of -96.67%, -84.62%, and -92% over the LR, DNN, and SVM algorithms, respectively. Moreover, the maximum C error rate improvement in RF over DNN and SVM is -21.89% and -27.5%, respectively, indicating that RF outperforms these algorithms by a significant amount. Additionally, Figure 4.4 displays the correlation between the actual and estimated SOH trends together with the corresponding error as a function of the mileage. The error is expressed as the percentage difference between the real and estimated data. Specifically, for each analyzed vehicle, representing a distinct mileage condition, the variation in SOH was calculated. Although it may be disorienting to observe the SOH fluctuating in the first 22×10^3 km, this is due to variations in the SoC levels of the 12 different vehicles considered in Table 4.2. From 22 to 130×10^3 km, the trend is a clean line due to unavailable intermediate values. These evaluations were then combined into a single graph, along with the error introduced by the presented algorithms as compared to the experimental measurements. The decision to plot all data on a single graph was motivated by readability concerns and the desire to avoid overburdening the reader. Similarly, the same approach is used for the error. The best performance is observed for RF, KNN, and CART, followed by DNN, SVM, and LR, where LR exhibits the poorest performance.

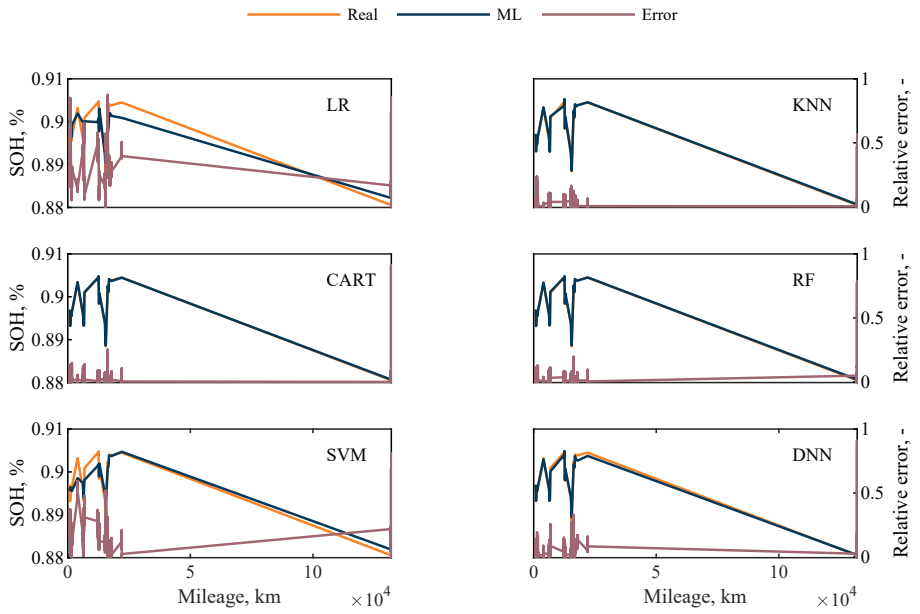


Fig. 4.4 SOH and the error estimation as a function of mileage condition. For each graph, the fluctuations in the first 22×10^3 km are due to variations in the SoC and the 12 different vehicles considered in Table 4.2; instead from 22 to 130×10^3 km the trend is a clean line due to unavailable intermediate values.

Table 4.3 Results comparison (Part 1).

Algorithm	$\bar{e}_{ocv}$	$e_{max,ocv}$	$\bar{e}_c$ (%)	$e_{max,c}$
LR	0.6	2.37	0.75	3.95
DNN	0.13	2.4	0.16	2.97
SVM	0.25	3.00	0.4	3.20
RF	0.02	2.38	0.02	2.32
KNN	0.04	2.62	0.06	2.55
CART	0.02	2.65	0.02	2.60

Table 4.4 Results comparison (Part 2).

RF wrt Algorithm [1]	$\bar{e}_{ocv}$	$e_{max,ocv}$	$\bar{e}_C$	$e_{max,C}$
			(%)	
$\mathrm{RF_{LR}}$	−96.67	+0.42	−97.33	−41.27
RF_{DNN}	−84.62	−0.83	−87.5	−21.89
RF_{SVM}	−92	−20.67	−95	−27.5
RF_{KNN}	−50	−9.16	−66.67	−9.02
RF_{CART}	0	−10.19	0	−10.77

[1] Performance comparison of RF with respect to (wrt) other algorithms evaluated as $\frac{x_{RF}-x_{alg}}{x_{alg}} \cdot 100$

In Figure 4.5, the performance of the six algorithms are compared using a radar diagram [136], and a score between 1 (worst) and 5 (best) is assigned considering some of the index listed in Table 4.1, eliminating common parts such as required data and preprocessing. RF outperformes the other algorithms in terms of accuracy and performance, while exhibiting moderate complexity and interpretability. RF, KNN and CART show similar accuracy and performance, with slightly lower complexity and interpretability. DNN shows good performance, but higher complexity and computational effort, as well as lower interpretability. On all performance metrics but interpretability, LR performs worse than other algorithms, while SVM performs worse than other algorithms but better than LR. A larger dataset would be required to obtain a better performance from DNN. LR is unsuitable for complex and high-performance tasks, while KNN's computational effort would increase with larger datasets. For confidentiality reasons, we are unable to disclose the specific details and measurements of the computational efforts. However, the detailed comparison of the algorithms considered in Section 4.4.1 was introduced to allow for readers to gain an understanding of the computational processes and complexity involved.

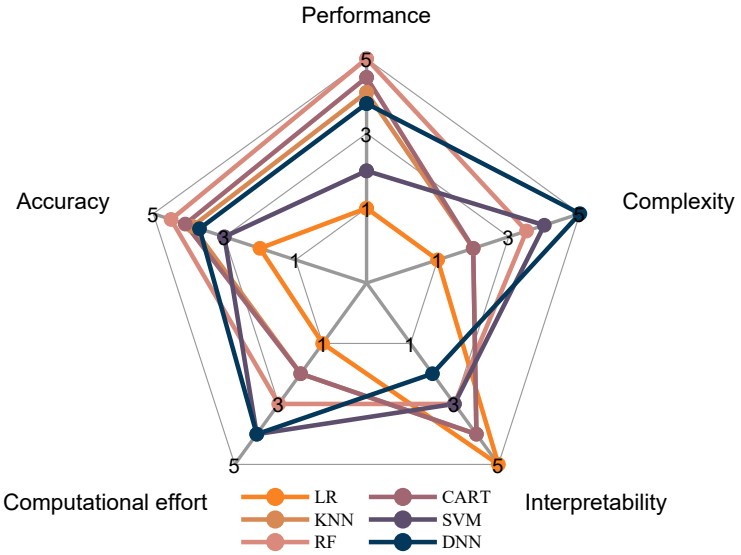


Fig. 4.5 Comparison of the different algorithm results in terms of performance, complexity, interpretability, computational effort, and accuracy. Each algorithm is scored on a scale from 1 (worst) to 5 (best), based on criteria listed in Table 4.1, eliminating common parts such as required data and preprocessing.

Chapter 5

Conclusion and Future Research Directions

This thesis has explored the intersection of artificial intelligence (AI) and electrified vehicle performance, focusing on hybrid electric vehicles (HEVs), battery electric vehicles (BEVs), and the overarching goal of enhancing environmental sustainability, operational efficiency, and system reliability. Through the application of AI techniques such as reinforcement learning, tabular Q-learning, long short-term memory networks, and gated recurrent unit neural networks, we have addressed three core areas: optimizing HEV performance for fuel economy and comfort, improving BEV traffic flow and energy usage through trajectory optimization, and accurately estimating the state-of-health of high-voltage batteries and extend their lifespan.

Our findings demonstrate that AI can significantly contribute to the advancement of electrified vehicle technology. In HEVs, we showed how AI methods can fine-tune energy management to balance fuel efficiency with passenger comfort. For BEVs, our research into trajectory optimization revealed potential for reducing energy consumption and improving traffic efficiency. Moreover, the use of AI for battery SoH estimation has been proven effective, offering insights that could lead to better battery management and longer lifespans.

The methodologies employed in this thesis, ranging from reinforcement learning in scenarios lacking GPS input to supervised learning for vehicles with

predetermined routes, highlight the versatility and potential of AI in addressing diverse challenges within the field of electrified transportation.

5.1 Limitations

This thesis, while making interesting strides in the domain of electrified vehicle performance and AI integration, acknowledges certain constraints across the three main areas of exploration. These limitations, rather than detracting from the value of the research, underscore the complex and evolving nature of the field.

Due to the expansive scope of this work and the time constraints inherent in a three-year research period, some aspects could not be fully addressed. Notably, the rapidly changing regulatory landscape impacted the relevance of certain findings, with some conclusions becoming outdated before the thesis was finalized. This dynamic context presents both a challenge and an opportunity for ongoing research.

Regarding the energy management in hybrid electric vehicles (HEVs), the primary limitation lies in the lack of Hardware-in-the-Loop (HIL) validation for the proposed algorithms.

In the area of Cooperative Adaptive Cruise Control (CACC), the omission of lateral dynamics and the assumption of linear longitudinal dynamics represent key limitations. These simplifications, while necessary for the scope of this study, suggest areas for future research to achieve more comprehensive modeling and increase the meaning-fullness of obtained results.

Lastly, for the estimation of battery State-of-Health (SoH), collaboration constraints with industry partners limited access to certain data profiles, restricting the range of algorithms that could be explored. Gathering further and more diversified experimental data would therefore allow exploring different algorithms and improving the robustness of the proposed approach. This limitation highlights the importance of robust partnerships and open data access in advancing the field.

5.2 Future Research Directions

Given the opportunity to select a focus for future research, advancing the study of HEV energy management and validating the system within a HIL environment emerges as a compelling direction. This research direction holds promise for enhancing both vehicle autonomy and the energy management of HEVs in a realistic simulation environment.

Referring to CACC, incorporating nonlinear dynamics and, similirarly to the HEV case, validating the system within a HIL environment emerges as a next step. To achieve this, it would be essential to first address and mitigate uncertainties, potentially integrating traffic light and infrastructure data to enhance the system's realism and applicability. Then, by incorporating nonlinear dynamics into CACC algorithms, the objective is to achieve a more accurate representation of vehicle behavior across diverse operational conditions.

On the front of SoH estimation, gaining access to comprehensive data collection signals would enable more precise analyses of charging, discharging, and temperature profiles. With a complete dataset, focusing on refining algorithms could lead to significant advancements. Starting with methodologies like Random Forest and/or XGBoost as a foundation, there's potential to further explore more complex networks such as Nonlinear Autoregressive with Exogenous Inputs (NARX) and Gated Recurrent Unit (GRU) models for benchmarking purposes.

Moreover, the application of AI techniques in research introduces complex ethical considerations that warrant a deeper examination. It's essential to approach the recent enthusiasm for AI with a nuanced perspective, acknowledging that these technologies not only build upon but also significantly extend classical statistical and optimization methods. This distinction is critical, as it highlights the innovative capabilities of AI to analyze and interpret complex datasets beyond the scope of traditional methodologies. However, this evolution also necessitates a thoughtful discussion on the ethical implications of AI's reach and its foundational roots in established statistical principles. Equally, the concept of *sustainability*, which has been liberally applied throughout this work, calls for a more discerning evaluation. The term's frequent use underscores the need for a shift towards a more precise and accountable understanding of what

true sustainability entails in the context of technological research and development. As we look to the future, it is imperative that research, particularly in fields as impactful as AI and sustainability, is pursued in collaboration with experts across disciplines, such as Life Cycle Assessment (LCA). In this way, it would be possible to ensure that this research not only advances technological frontiers but does so with a commitment to ethical integrity and a genuine contribution to sustainability.

References

[1] A. Karthikeyan and U.D. Priyakumar. Artificial intelligence: machine learning for chemical sciences. *Journal of Chemical Sciences*, 134(2), 2022. doi:10.1007/s12039-021-01995-2.

[2] Peter Slowik, Nikita Pavlenko, and Nicholas Lutsey. Assessment of next-generation electric vehicle technologies. White paper, International Council on Clean Transportation, 10 2016.

[3] Gustav Fredriksson, Alexander Roth, Simone Tagliapietra, and Reinhilde Veugelers. Is the european automotive industry ready for the global electric vehicle revolution? *Bruegel Policy Contribution*, 2018. URL https://www.bruegel.org/sites/default/files/wp_attachments/PC-26_2018_1.pdf.

[4] MordorIntelligence. Hybrid vehicle market analysis - industry report - trends, size & share. URL https://www.mordorintelligence.com/industry-reports/hybrid-vehicle-market. Accessed on 30 October 2022.

[5] Electric vehicle market analysis - industry report - trends, size & share. URL https://www.mordorintelligence.com/industry-reports/electric-vehicle-market. Accessed: 2022-10-30.

[6] TowardsDataScience. Value-based methods in deep reinforcement learning. URL towardsdatascience.com/value-based-methods-in-deep-reinforcement-learning-d40ca1086e1#:~:text=Value-Based. Accessed: 2023-06-18.

[7] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. http://www.deeplearningbook.org.

[8] Alessia Musa, Pier Giuseppe Anselma, Matteo Spano, Daniela Anna Misul, and Giovanni Belingardi. Cooperative adaptive cruise control: A gated recurrent unit approach. In *2022 IEEE Transportation Electrification Conference & Expo (ITEC)*, pages 208–213, 2022. doi:10.1109/ITEC53557.2022.9813990.

[9] Edoardo Lelli, Alessia Musa, Emilio Batista, Daniela Anna Misul, and Giovanni Belingardi. On-road experimental campaign for machine learning based state of health estimation of high-voltage batteries in electric vehicles. *Energies*, 16(12), 2023. ISSN 1996-1073. doi:10.3390/en16124639.

[10] Alessia Musa, Pier Giuseppe Anselma, Giovanni Belingardi, and Daniela Anna Misul. Energy management in hybrid electric vehicles: A q-learning solution for enhanced drivability and energy efficiency. *Energies*, 17(1), 2024. ISSN 1996-1073. doi:10.3390/en17010062. URL https://www.mdpi.com/1996-1073/17/1/62.

[11] Sutton Richard S. and Barto. Andrew G. *Reinforcement learning: An introduction.* MIT Press, 2018.

[12] Lino Guzzella and Antonio Sciarretta. *Vehicle Propulsion Systems: Introduction to Modeling and Optimization*. 2007. ISBN 978-3-642-35912-5. doi:10.1007/978-3-642-35913-2.

[13] Simona Onori, Lorenzo Serrao, and Giorgio Rizzoni. *Hybrid Electric Vehicles: Energy Management Strategies*. Springer London, London, 2016.

[14] Pier Giuseppe Anselma. Rule-based control and equivalent consumption minimization strategies for hybrid electric vehicle powertrains: a hardware-in-the-loop assessment. In *2022 IEEE 31st International Symposium on Industrial Electronics (ISIE)*, pages 680–685, 2022. doi:10.1109/ISIE51582.2022.9831702.

[15] Alessia Musa, Federico Miretti, and Daniela Misul. Mpc-based cooperative longitudinal control for vehicle strings in a realistic driving environment. In *WCX SAE World Congress Experience*. SAE International, apr 2023. doi:https://doi.org/10.4271/2023-01-0689. URL https://doi.org/10.4271/2023-01-0689.

[16] P.G. Anselma, C. Maino, and A. Musa. Theo: A tailored hybrid emission optimizer for the drivers of tomorrow. In *Young Researcher TRA Visions 2020 Award, sponsored by ERTRAC for the road mode category*, 2020. 1st Place. Available online: https://www.travisions.eu/TRAVisions/young_researcher_results_2020/;jsessionid=70438b8b2153916744aa54d0656b (accessed on 17 February 2024).

[17] P. G. Anselma, C. Maino, A. Musa, G. Belingardi, E. Spessa, and D. A. Misul. Method for controlling a powertrain system of a hybrid electric vehicle and related hybrid electric vehicle. Patent Application, April 2020. Application number ("Ministero dello Sviluppo Economico", Italy): 102020000008686.

[18] Sanjaka G. Wirasingha and Ali Emadi. Classification and review of control strategies for plug-in hybrid electric vehicles. *IEEE Transactions on Vehicular Technology*, 60(1):111–122, 2011. doi:10.1109/TVT.2010.2090178.

[19] Pierluigi Pisu and Giorgio Rizzoni. A comparative study of supervisory control strategies for hybrid electric vehicles. *IEEE Transactions on Control Systems Technology*, 15(3):506–518, 2007. doi:10.1109/TCST.2007.894649.

[20] N. Jalil, N.A. Kheir, and M. Salman. A rule-based energy management strategy for a series hybrid vehicle. In *Proceedings of the 1997 American Control Conference (Cat. No.97CH36041)*, volume 1, pages 689–693 vol.1, 1997. doi:10.1109/ACC.1997.611889.

[21] Theo Hofman, M. Steinbuch, Roell Druten, and Alex Serrarens. A rule-based energy management strategies for hybrid vehicles. *International Journal of Electric and Hybrid Vehicles*, 1, 01 2007. doi:10.1504/IJEHV.2007.014448.

[22] Harpreetsingh Banvait, Sohel Anwar, and Yaobin Chen. A rule-based energy management strategy for plug-in hybrid electric vehicle (phev). In *2009 American Control Conference*, pages 3938–3943, 2009. doi:10.1109/ACC.2009.5160242.

[23] T. Hofman, M. Steinbuch, R.M. van Druten, and A.F.A. Serrarens. Rule-based energy management strategies for hybrid vehicle drivetrains: A fundamental approach in reducing computation time. *IFAC Proceedings Volumes*, 39(16):740–745, 2006. ISSN 1474-6670. doi:https://doi.org/10.3182/20060912-3-DE-2911.00128. 4th IFAC Symposium on Mechatronic Systems.

[24] Daniel Goerke, Michael Bargende, Uwe Keller, Norbert Ruzicka, and Stefan Schmiedler. Optimal control based calibration of rule-based energy management for parallel hybrid electric vehicles. *SAE International Journal of Alternative Powertrains*, 4(1):178–189, apr 2015. ISSN 2167-4191. doi:https://doi.org/10.4271/2015-01-1220.

[25] Jiankun Peng, Hongwen He, and Rui Xiong. Rule based energy management strategy for a series–parallel plug-in hybrid electric bus optimized by dynamic programming. *Applied Energy*, 185:1633–1643, 2017. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2015.12.031. Clean, Efficient and Affordable Energy for a Sustainable Future.

[26] Antonio Sciarretta and Lino Guzzella. Control of hybrid electric vehicles. *IEEE Control Systems Magazine*, 27(2):60–70, 2007. doi:10.1109/MCS.2007.338280.

[27] Shaobo Xie, Xiaosong Hu, Zongke Xin, and James Brighton. Pontryagin's minimum principle based model predictive control of energy management

for a plug-in hybrid electric bus. *Applied Energy*, 236:893–905, 2019. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2018.12.032.

[28] Namwook Kim, Sukwon Cha, and Huei Peng. Optimal control of hybrid electric vehicles based on pontryagin's minimum principle. *IEEE Transactions on Control Systems Technology*, 19(5):1279–1287, 2011. doi:10.1109/TCST.2010.2061232.

[29] Cristian Musardo, Giorgio Rizzoni, Yann Guezennec, and Benedetto Staccia. A-ecms: An adaptive algorithm for hybrid electric vehicle energy management. *European Journal of Control*, 11(4):509–524, 2005. ISSN 0947-3580. doi:https://doi.org/10.3166/ejc.11.509-524.

[30] Simona Onori, Lorenzo Serrao, and Giorgio Rizzoni. Adaptive equivalent consumption minimization strategy for hybrid electric vehicles. In *Dynamic systems and control conference*, volume 44175, pages 499–505, 2010.

[31] S. Delprat, J. Lauber, T.M. Guerra, and J. Rimaux. Control of a parallel hybrid powertrain: optimal control. *IEEE Transactions on Vehicular Technology*, 53(3):872–881, 2004. doi:10.1109/TVT.2004.827161.

[32] Federico Millo, Luciano Rolando, Luigi Tresca, and Luca Pulvirenti. Development of a neural network-based energy management system for a plug-in hybrid electric vehicle. *Transportation Engineering*, 11:100156, 2023. ISSN 2666-691X. doi:https://doi.org/10.1016/j.treng.2022.100156.

[33] Roberto Finesso, Ezio Spessa, and Mattia Venditti. An unsupervised machine-learning technique for the definition of a rule-based control strategy in a complex hev. *SAE International Journal of Alternative Powertrains*, 5, 04 2016. doi:10.4271/2016-01-1243.

[34] Yuanjian Zhang, Zheng Chen, Guang Li, Yonggang Liu, Haibo Chen, Geoff Cunningham, and Juliana Early. Machine learning-based vehicle model construction and validation—toward optimal control strategy development for plug-in hybrid electric vehicles. *IEEE Transactions on Transportation Electrification*, 8(2):1590–1603, 2022. doi:10.1109/TTE.2021.3111966.

[35] Zheng Chen, Chao Yang, and Shengnan Fang. A convolutional neural network-based driving cycle prediction method for plug-in hybrid electric vehicles with bus route. *IEEE Access*, 8:3255–3264, 2020. doi:10.1109/ACCESS.2019.2960771.

[36] Xue Lin, Paul Bogdan, Naehyuck Chang, and Massoud Pedram. Machine learning-based energy management in a hybrid electric vehicle to minimize total operating cost. In *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 627–634, 2015. doi:10.1109/ICCAD.2015.7372628.

[37] M.F. M. Sabri, K.A. Danapalasingam, and M.F. Rahmat. A review on hybrid electric vehicles architecture and energy management strategies. *Renewable and Sustainable Energy Reviews*, 53:1433–1442, 2016. ISSN 1364-0321. doi:https://doi.org/10.1016/j.rser.2015.09.036.

[38] Xi Huang, Ying Tan, and Xingui He. An intelligent multifeature statistical approach for the discrimination of driving conditions of a hybrid electric vehicle. *IEEE Transactions on Intelligent Transportation Systems*, 12(2): 453–465, 2011. doi:10.1109/TITS.2010.2093129.

[39] Yi Lu Murphey, Jungme Park, Leonidas Kiliaris, Ming L. Kuang, M. Abul Masrur, Anthony M. Phillips, and Qing Wang. Intelligent hybrid vehicle power control—part ii: misc intelligent energy management. *IEEE Transactions on Vehicular Technology*, 62(1):69–79, 2013. doi:10.1109/TVT.2012.2217362.

[40] Kuan Liu, Zachary Asher, Xun Gong, Mike Huang, and Ilya Kolmanovsky. Vehicle velocity prediction and energy management strategy part 1: Deterministic and stochastic vehicle velocity prediction using machine learning. In *WCX SAE World Congress Experience*. SAE International, apr 2019. doi:https://doi.org/10.4271/2019-01-1051.

[41] Lu Han, Xiaohong Jiao, and Zhao Zhang. Recurrent neural network-based adaptive energy management control strategy of plug-in hybrid electric vehicles considering battery aging. *Energies*, 13(1), 2020. doi:10.3390/en13010202. Cited by: 29; All Open Access, Gold Open Access, Green Open Access.

[42] Pedro Maroto Estrada, Daniela de Lima, Peter H. Bauer, Marco Mammetti, and Joan Carles Bruno. Deep learning in the development of energy management strategies of hybrid electric vehicles: A hybrid modeling approach. *Applied Energy*, 329:120231, 2023. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2022.120231.

[43] Tao Zhang, Chunqing Zhao, Xiaoxia Sun, Min Lin, and Qidi Chen. Uncertainty-aware energy management strategy for hybrid electric vehicle using hybrid deep learning method. *IEEE Access*, 10:63152–63162, 2022. doi:10.1109/ACCESS.2022.3182805.

[44] Teng Liu, Xiaolin Tang, Hong Wang, Huilong Yu, and Xiaosong Hu. Adaptive hierarchical energy management design for a plug-in hybrid electric vehicle. *IEEE Transactions on Vehicular Technology*, 68(12): 11513–11522, 2019. doi:10.1109/TVT.2019.2926733.

[45] Teng Liu, Xiaosong Hu, Shengbo Eben Li, and Dongpu Cao. Reinforcement learning optimized look-ahead energy management of a parallel hybrid electric vehicle. *IEEE/ASME Transactions on Mechatronics*, 22 (4):1497–1507, 2017. doi:10.1109/TMECH.2017.2707338.

[46] Yue Hu, Weimin Li, Kun Xu, Taimoor Zahid, Feiyan Qin, and Chenming Li. Energy management strategy for a hybrid electric vehicle based on deep reinforcement learning. *Applied Sciences*, 8(2), 2018. ISSN 2076-3417. doi:10.3390/app8020187.

[47] Yuan Zou, Teng Liu, Dexing Liu, and Fengchun Sun. Reinforcement learning-based real-time energy management for a hybrid tracked vehicle. *Applied Energy*, 171:372–382, 2016. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2016.03.082.

[48] Yuankai Wu, Huachun Tan, Jiankun Peng, Hailong Zhang, and Hongwen He. Deep reinforcement learning of energy management with continuous control strategy and traffic information for a series-parallel plug-in hybrid electric bus. *Applied Energy*, 247:454–466, 2019. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2019.04.021.

[49] Xiaosong Hu, Teng Liu, Xuewei Qi, and Matthew Barth. Reinforcement learning for hybrid and plug-in hybrid electric vehicle energy management: Recent advances and prospects. *IEEE Industrial Electronics Magazine*, 13(3):16–25, 2019. doi:10.1109/MIE.2019.2913015.

[50] Bin Xu, Dhruvang Rathod, Darui Zhang, Adamu Yebi, Xueyu Zhang, Xiaoya Li, and Zoran Filipi. Parametric study on reinforcement learning optimized energy management strategy for a hybrid electric vehicle. *Applied Energy*, 259:114200, 2020. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2019.114200.

[51] Jingda Wu, Hongwen He, Jiankun Peng, Yuecheng Li, and Zhanjiang Li. Continuous reinforcement learning of energy management with deep q network for a power split hybrid electric bus. *Applied Energy*, 222:799–811, 2018. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2018.03.104.

[52] Atriya Biswas, Pier Giuseppe Anselma, and Ali Emadi. Real-time optimal energy management of multimode hybrid electric powertrain with misc trainable asynchronous advantage actor–critic algorithm. *IEEE Transactions on Transportation Electrification*, 8(2):2676–2694, 2022. doi:10.1109/TTE.2021.3138330.

[53] Zexing Wang, Hongwen He, Jiankun Peng, Weiqi Chen, Changcheng Wu, Yi Fan, and Jiaxuan Zhou. A comparative study of deep reinforcement learning based energy management strategy for hybrid electric vehicle. *Energy Conversion and Management*, 293:117442, 2023. ISSN 0196-8904. doi:https://doi.org/10.1016/j.enconman.2023.117442.

[54] Xuefeng Han, Hongwen He, Jingda Wu, Jiankun Peng, and Yuecheng Li. Energy management based on reinforcement learning with double deep q-learning for a hybrid electric tracked vehicle. *Applied Energy*, 254:113708, 2019. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2019.113708.

[55] Bin Xu, Xiaolin Tang, Xiaosong Hu, Xianke Lin, Huayi Li, Dhruvang Rathod, and Zhe Wang. Q-learning-based supervisory control adaptability investigation for hybrid electric vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):6797–6806, 2022. doi:10.1109/TITS.2021.3062179.

[56] Zheng Chen, Hongji Gu, Shiquan Shen, and Jiangwei Shen. Energy management strategy for power-split plug-in hybrid electric vehicle based on mpc and double q-learning. *Energy*, 245:123182, 2022. ISSN 0360-5442. doi:https://doi.org/10.1016/j.energy.2022.123182.

[57] Federico Miretti, Daniela Misul, and Ezio Spessa. Dynaprog: Deterministic dynamic programming solver for finite horizon multi-stage decision problems. *SoftwareX*, 14:100690, 2021. ISSN 2352-7110. doi:https://doi.org/10.1016/j.softx.2021.100690.

[58] Federico Miretti and Daniela Misul. Driveability constrained models for optimal control of hybrid electric vehicles. 03 2023. doi:10.48550/arXiv.2303.12603.

[59] Guodong Du, Yuan Zou, Xudong Zhang, Teng Liu, Jinlong Wu, and Dingbo He. Deep reinforcement learning based energy management for a hybrid electric vehicle. *Energy*, 201:117591, 2020. ISSN 0360-5442. doi:https://doi.org/10.1016/j.energy.2020.117591.

[60] openAI. Part 2: Kinds of rl algorithms. URL https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html. Accessed: 2023-06-18.

[61] Liberzon Daniel. *Calculus of Variations and Optimal Control Theory*. Princeton University Press, Princeton, 2012. ISBN 9781400842643. doi:doi:10.1515/9781400842643. URL https://doi.org/10.1515/9781400842643.

[62] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volume I. Athena Scientific, Belmont, MA, USA, 3rd edition, 2005.

[63] A. Brahma, Y. Guezennec, and G. Rizzoni. Optimal energy management in series hybrid electric vehicles. In *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, volume 1, pages 60–64 vol.1, 2000. doi:10.1109/ACC.2000.878772.

[64] Ziyou Song, Heath Hofmann, Jianqiu Li, Xuebing Han, and Minggao Ouyang. Optimization for a hybrid energy storage system in electric vehicles using dynamic programing approach. *Applied Energy*, 139:151–162, 2015. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2014.11.020. URL https://www.sciencedirect.com/science/article/pii/S0306261914011696.

[65] The MathWorks Inc. Matlab version: 9.12.0 (r2022a), 2022. URL https://www.mathworks.com.

[66] Carlos Massera Filho, Marco H. Terra, and Denis F. Wolf. Safe optimization of highway traffic with robust model predictive control-based cooperative adaptive cruise control. *IEEE Transactions on Intelligent Transportation Systems*, 18(11):3193–3203, 2017. ISSN 1524-9050, 1558-0016. doi:10.1109/TITS.2017.2679098.

[67] Didier Rodrigues Lopes and Simos A. Evangelou. Energy savings from an eco-cooperative adaptive cruise control: a BEV platoon investigation. In *2019 18th European Control Conference (ECC)*, pages 4160–4167. IEEE, 2019. ISBN 978-3-907144-00-8. doi:10.23919/ECC.2019.8796226.

[68] Niklas Wikström, Alejandro Parrilla, Stephen Jones, and Anders Grauers. Energy-efficient cooperative adaptive cruise control with receding horizon of traffic, route topology, and traffic light information. *SAE International Journal of Connected and Automated Vehicles*, 2, 05 2019. doi:10.4271/12-02-02-0006.

[69] Defeng He, Tianxiang Qiu, and Renshi Luo. Fuel efficiency-oriented platooning control of connected nonlinear vehicles: A distributed economic mpc approach. *Asian Journal of Control*, 22(4):1628–1638, Jul 2020. doi:10.1002/asjc.2049.

[70] Hadi Kazemi, Hossein Nourkhiz Mahjoub, Amin Tahmasbi-Sarvestani, and Yaser P. Fallah. A learning-based stochastic MPC design for cooperative adaptive cruise control to handle interfering vehicles. *IEEE Transactions on Intelligent Vehicles*, 3(3):266–275, 2018. ISSN 2379-8904, 2379-8858. doi:10.1109/TIV.2018.2843135. Number: 3.

[71] Ellen van Nunen, Jan Verhaegh, Emilia Silvas, Elham Semsar-Kazerooni, and Nathan van de Wouw. Robust model predictive cooperative adaptive cruise control subject to v2v impairments. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8, Yokohama, Oct 2017. doi:10.1109/ITSC.2017.8317758.

[72] C. Desjardins and B. Chaib-draa. Cooperative adaptive cruise control: A reinforcement learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1248–1260, 2011. ISSN 1524-9050, 1558-0016. doi:10.1109/TITS.2011.2157145.

[73] Bin Tian, Guanqun Wang, Zhigang Xu, Yuqin Zhang, and Xiangmo Zhao. Communication delay compensation for string stability of CACC system using LSTM prediction. *Vehicular Communications*, 29:100333, 2021. ISSN 22142096. doi:10.1016/j.vehcom.2021.100333.

[74] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, 2014. doi:10.3115/v1/W14-4012.

[75] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, 2014. doi:10.3115/v1/D14-1179.

[76] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, Dec 2014. URL http://arxiv.org/abs/1412.3555. Consulted: 15th December 2021.

[77] Wuyan Li, Hao Wu, Nanyang Zhu, Yongnian Jiang, Jinglu Tan, and Ya Guo. Prediction of dissolved oxygen in a fishery pond based on gated recurrent unit (gru). *Information Processing in Agriculture*, 8(1):185–193, Mar 2021. doi:10.1016/j.inpa.2020.02.002.

[78] OpenAI. Soft actor-critic. https://spinningup.openai.com/en/latest/algorithms/sac.html, 2019. Accessed: 2024-01-15.

[79] Pier Giuseppe Anselma and Giovanni Belingardi. Enhancing energy saving opportunities through rightsizing of a battery electric vehicle powertrain for optimal cooperative driving. *SAE International Journal of Connected and Automated Vehicles*, 3(2):12–03–02–0007, 2020. ISSN 2574-075X. doi:10.4271/12-03-02-0007.

[80] Matteo Spano, Pier Giuseppe Anselma, Alessia Musa, Daniela Anna Misul, and Giovanni Belingardi. Optimal real-time velocity planner of a battery electric vehicle in v2v driving. In *2021 IEEE Transportation Electrification Conference & Expo (ITEC)*, pages 194–199. IEEE, 2021. ISBN 978-1-72817-583-6. doi:10.1109/ITEC51675.2021.9490121.

[81] Yang Zheng, Shengbo Eben Li, Jianqiang Wang, Dongpu Cao, and Keqiang Li. Stability and scalability of homogeneous vehicular platoon: Study on the influence of information flow topologies. *IEEE Transactions on Intelligent Transportation Systems*, 17(1):14–26, 2016. ISSN 1524-9050, 1558-0016. doi:10.1109/TITS.2015.2402153.

[82] Matteo Spano, Alessia Musa, Pier Giuseppe Anselma, Daniela Anna Misul, and Giovanni Belingardi. Battery electric vehicles platooning: Assessing capability of energy saving and passenger comfort improvement. In *2021 AEIT International Conference on Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, pages 1–6, Torino, Italy, Nov 2021. doi:10.23919/AEITAUTOMOTIVE52815.2021.9662788.

[83] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, Jan 2017. URL http://arxiv.org/abs/1412.6980. Consulted: 15th April 2022.

[84] J. Brownlee. Gentle introduction to the adam optimization algorithm for deep learning. Machine Learning Mastery, Jul 2 2017. URL https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. Consulted: 15th April 2022.

[85] Department of Railway Vehicles, Faculty of Transport, University Politehnica of Bucharest, 313 Splaiul Independenţei, 060042, Bucharest, Romania and M. Dumitriu. Analysis of the dynamic response in the railway vehicles to the track vertical irregularities. part II: The numerical analysis. 8(4):32–39, 2015. ISSN 17919320, 17912377. doi:10.25103/jestr.084.05.

[86] C. C. Smith, D. Y. McGehee, and A. J. Healey. The prediction of passenger riding comfort from acceleration data. 100(1):34–41, 1978. ISSN 0022-0434, 1528-9028. doi:10.1115/1.3426338.

[87] B. McAuliffe, M. Lammert, X.-Y. Lu, S. Shladover, M.-D. Surcel, and A. Kailas. Influences on Energy Savings of Heavy Trucks Using Cooperative Adaptive Cruise Control. In *WCX World Congress Experience*, pages 2018–01–1181, Apr 2018. doi:10.4271/2018-01-1181.

[88] Amir Alipour-Fanid, Monireh Dabaghchian, and Kai Zeng. Platoon stability and safety analysis of cooperative adaptive cruise control under wireless rician fading channels and jamming attacks. 10 2017. doi:https://doi.org/10.48550/arXiv.1710.08476.

[89] Haitao Xing, Jeroen Ploeg, and Henk Nijmeijer. Robust cacc in the presence of uncertain delays. *IEEE Transactions on Vehicular Technology*, 71(4):3507–3518, 2022. doi:10.1109/TVT.2022.3148119.

[90] Xuan Wang, Manjiang Hu, Yougang Bian, Xiaowei Wang, Hongmao Qin, and Rongjun Ding. Dmpc-based string stable platoon control with robustness against communication delays. *Vehicular Communications*, page 100655, 8 2023. ISSN 22142096. doi:https://doi.org/10.1016/j.vehcom.2023.100655.

[91] Jeroen Ploeg, Elham Semsar-Kazerooni, Guido Lijster, Nathan van de Wouw, and Henk Nijmeijer. Graceful degradation of cooperative adaptive cruise control. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):488–497, 2015. doi:10.1109/TITS.2014.2349498.

[92] Charles Desjardins and Brahim Chaib-draa. Cooperative adaptive cruise control: A reinforcement learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1248–1260, 2011. doi:10.1109/TITS.2011.2157145.

[93] Haotian Shi, Yang Zhou, Keshu Wu, Xin Wang, Yangxin Lin, and Bin Ran. Connected automated vehicle cooperative control with a deep reinforcement learning approach in a mixed traffic environment.

Transportation Research Part C: Emerging Technologies, 133, 12 2021. doi:https://doi.org/10.1016/j.trc.2021.103421.

[94] Model wireless message communication with packet loss and channel failure. URL https://it.mathworks.com/help/simulink/ug/message-communication-with-packet-loss.html.

[95] Languang Lu, Xuebing Han, Jianqiu Li, Jianfeng Hua, and Minggao Ouyang. A review on the key issues for lithium-ion battery management in electric vehicles. *Journal of Power Sources*, 226:272–288, 2013. ISSN 0378-7753. doi:https://doi.org/10.1016/j.jpowsour.2012.10.060.

[96] Carlos Vidal, Pawel Malysz, Phillip Kollmeyer, and Ali Emadi. Machine learning applied to electrified vehicle battery state of charge and state of health estimation: State-of-the-art. *IEEE Access*, 8:52796–52814, 2020. doi:10.1109/ACCESS.2020.2980961.

[97] Mina Naguib, Phillip Kollmeyer, and Ali Emadi. Lithium-ion battery pack robust state of charge estimation, cell inconsistency, and balancing: Review. *IEEE Access*, 9:50570–50582, 2021. doi:10.1109/ACCESS.2021.3068776.

[98] Lei Yao, Shiming Xu, Aihua Tang, Fang Zhou, Junjian Hou, Yanqiu Xiao, and Zhijun Fu. A review of lithium-ion battery state of health estimation and prediction methods. *World Electric Vehicle Journal*, 12(3), 2021. ISSN 2032-6653. doi:10.3390/wevj12030113.

[99] Vima Mali, Rajat Saxena, Kundan Kumar, Abul Kalam, and Brijesh Tripathi. Review on battery thermal management systems for energy-efficient electric vehicles. *Renewable and Sustainable Energy Reviews*, 151:111611, 2021. ISSN 1364-0321. doi:https://doi.org/10.1016/j.rser.2021.111611.

[100] Fangfang Yang, Shaohui Zhang, Weihua Li, and Qiang Miao. State-of-charge estimation of lithium-ion batteries using lstm and ukf. *Energy*, 201:117664, 2020. ISSN 0360-5442. doi:https://doi.org/10.1016/j.energy.2020.117664.

[101] Ethelbert Ezemobi, Mario Silvagni, Ahmad Mozaffari, Andrea Tonoli, and Amir Khajepour. State of health estimation of lithium-ion batteries in electric vehicles under dynamic load conditions. *Energies*, 15(3), 2022. ISSN 1996-1073. doi:10.3390/en15031234.

[102] M.S. Hossain Lipu, M.A. Hannan, Aini Hussain, Afida Ayob, Mohamad H.M. Saad, Tahia F. Karim, and Dickson N.T. How. Data-driven state of charge estimation of lithium-ion batteries: Algorithms, implementation factors, limitations and future trends. *Journal of Cleaner Production*, 277:124110, 2020. ISSN 0959-6526. doi:https://doi.org/10.1016/j.jclepro.2020.124110.

[103] Kaveh Khodadadi Sadabadi, Xin Jin, and Giorgio Rizzoni. Prediction of remaining useful life for a composite electrode lithium ion battery cell using an electrochemical model to estimate the state of health. *Journal of Power Sources*, 481:228861, 2021. ISSN 0378-7753. doi:https://doi.org/10.1016/j.jpowsour.2020.228861.

[104] Pier Giuseppe Anselma, Phillip Kollmeyer, Jeremy Lempert, Ziyu Zhao, Giovanni Belingardi, and Ali Emadi. Battery state-of-health sensitive energy management of hybrid electric vehicles: Lifetime prediction and ageing experimental validation. *Applied Energy*, 285:116440, 2021. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2021.116440.

[105] Pier Giuseppe Anselma, Alessia Musa, Claudio Maino, Daniela Misul, and Giovanni Belingardi. Effect of temperature distribution on the predicted cell lifetimes for a plug-in hybrid electric vehicle battery pack. In *WCX SAE World Congress Experience*. SAE International, mar 2022. doi:https://doi.org/10.4271/2022-01-0712.

[106] Lucian Ungurean, Mihai V. Micea, and Gabriel Cârstoiu. Online state of health prediction method for lithium-ion batteries, based on gated recurrent unit neural networks. *International Journal of Energy Research*, 44(8):6767–6777, 2020. doi:https://doi.org/10.1002/er.5413.

[107] Yi-Hsien Chiang, Wu-Yang Sean, and Jia-Cheng Ke. Online estimation of internal resistance and open-circuit voltage of lithium-ion batteries in electric vehicles. *Journal of Power Sources*, 196(8):3921–3932, 2011. ISSN 0378-7753. doi:https://doi.org/10.1016/j.jpowsour.2011.01.005.

[108] Jufeng Yang, Bing Xia, Wenxin Huang, Yuhong Fu, and Chris Mi. Online state-of-health estimation for lithium-ion batteries using constant-voltage charging current analysis. *Applied Energy*, 212:1589–1600, 2018. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2018.01.010.

[109] Gaizka Saldaña, José Ignacio San Martín, Inmaculada Zamora, Francisco Javier Asensio, and Oier Oñederra. Analysis of the current electric battery models for electric vehicle simulation. *Energies*, 12(14), 2019. ISSN 1996-1073. doi:10.3390/en12142750.

[110] J. Li, K. Adewuyi, N. Lotfi, R.G. Landers, and J. Park. A single particle model with chemical/mechanical degradation physics for lithium ion battery state of health (soh) estimation. *Applied Energy*, 212:1178–1190, 2018. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2018.01.011.

[111] James Marcicki, Marcello Canova, A. Terrence Conlisk, and Giorgio Rizzoni. Design and parametrization analysis of a reduced-order electrochemical model of graphite/LiFePO4 cells for SOC/SOH estimation. *Journal of Power Sources*, 237:310–324, 2013. ISSN 03787753. doi:10.1016/j.jpowsour.2012.12.120.

[112] Scott J. Moura, Nalin A. Chaturvedi, and Miroslav Krstić. Adaptive partial differential equation observer for battery state-of-charge/state-of-health estimation via an electrochemical model. *Journal of Dynamic Systems, Measurement, and Control*, 136(1):011015, 2014. ISSN 0022-0434, 1528-9028. doi:10.1115/1.4024801.

[113] Jonghoon Kim and B. H. Cho. State-of-charge estimation and state-of-health prediction of a li-ion degraded battery based on an ekf combined with a per-unit system. *IEEE Transactions on Vehicular Technology*, 60 (9):4249–4260, 2011. doi:10.1109/TVT.2011.2168987.

[114] Verena Klass, Mårten Behm, and Göran Lindbergh. A support vector machine-based state-of-health estimation method for lithium-ion batteries under electric vehicle operation. *Journal of Power Sources*, 270:262–272, 2014. ISSN 0378-7753. doi:https://doi.org/10.1016/j.jpowsour.2014.07.116.

[115] Xuning Feng, Caihao Weng, Xiangming He, Xuebing Han, Languang Lu, Dongsheng Ren, and Minggao Ouyang. Online state-of-health estimation for li-ion battery using partial charging segment based on support vector machine. *IEEE Transactions on Vehicular Technology*, 68(9):8583–8592, 2019. ISSN 0018-9545, 1939-9359. doi:10.1109/TVT.2019.2927120.

[116] Jinhao Meng, Lei Cai, Guangzhao Luo, Daniel-Ioan Stroe, and Remus Teodorescu. Lithium-ion battery state of health estimation with short-term current pulse test and support vector machine. *Microelectronics Reliability*, 88-90:1216–1220, 2018. ISSN 0026-2714. doi:https://doi.org/10.1016/j.microrel.2018.07.025. 29th European Symposium on Reliability of Electron Devices, Failure Physics and Analysis (ESREF 2018).

[117] Zheng Chen, Mengmeng Sun, Xing Shu, Renxin Xiao, and Jiangwei Shen. Online state of health estimation for lithium-ion batteries based on support vector machine. *Applied Sciences*, 8(6), 2018. ISSN 2076-3417. doi:10.3390/app8060925.

[118] Adnan Nuhic, Tarik Terzimehic, Thomas Soczka-Guth, Michael Buchholz, and Klaus Dietmayer. Health diagnosis and remaining useful life prognostics of lithium-ion batteries using data-driven methods. *Journal of Power Sources*, 239:680–688, 2013. ISSN 0378-7753. doi:https://doi.org/10.1016/j.jpowsour.2012.11.146.

[119] Kodjo S.R. Mawonou, Akram Eddahech, Didier Dumur, Dominique Beauvois, and Emmanuel Godoy. State-of-health estimators coupled to a random forest approach for lithium-ion battery aging factor ranking. *Journal of Power Sources*, 484:229154, 2021. ISSN 0378-7753. doi:https://doi.org/10.1016/j.jpowsour.2020.229154.

[120] Niankai Yang, Ziyou Song, Heath Hofmann, and Jing Sun. Robust state of health estimation of lithium-ion batteries using convolutional neural network and random forest. *Journal of Energy Storage*, 48:103857, 2022. ISSN 2352-152X. doi:https://doi.org/10.1016/j.est.2021.103857.

[121] Duo Yang, Yujie Wang, Rui Pan, Ruiyang Chen, and Zonghai Chen. A neural network based state-of-health estimation of lithium-ion battery in electric vehicles. *Energy Procedia*, 105:2059–2064, 2017. ISSN 1876-6102. doi:https://doi.org/10.1016/j.egypro.2017.03.583. 8th International Conference on Applied Energy, ICAE2016, 8-11 October 2016, Beijing, China.

[122] Shuzhi Zhang, Baoyu Zhai, Xu Guo, Kaike Wang, Nian Peng, and Xiongwen Zhang. Synchronous estimation of state of health and remaining useful lifetime for lithium-ion battery using the incremental capacity and artificial neural networks. *Journal of Energy Storage*, 26:100951, 2019. ISSN 2352-152X. doi:https://doi.org/10.1016/j.est.2019.100951.

[123] Angelo Bonfitto. A method for the combined estimation of battery state of charge and state of health based on artificial neural networks. *Energies*, 13(10), 2020. ISSN 1996-1073. doi:10.3390/en13102548.

[124] Ethelbert Ezemobi, Andrea Tonoli, and Mario Silvagni. Battery state of health estimation with improved generalization using parallel layer extreme learning machine . *Energies*, 14(8), 2021. ISSN 1996-1073. doi:10.3390/en14082243. URL https://www.mdpi.com/1996-1073/14/8/2243.

[125] Ephrem Chemali, Phillip J. Kollmeyer, Matthias Preindl, Youssef Fahmy, and Ali Emadi. A convolutional neural network approach for estimation of li-ion battery state of health from charge profiles. *Energies*, 15(3), 2022. ISSN 1996-1073. doi:10.3390/en15031185.

[126] Prakash Venugopal and Vigneswaran T. State-of-health estimation of li-ion batteries in electric vehicle using indrnn under variable load condition. *Energies*, 12(22), 2019. ISSN 1996-1073. doi:10.3390/en12224338.

[127] Penghua Li, Zijian Zhang, Qingyu Xiong, Baocang Ding, Jie Hou, Dechao Luo, Yujun Rong, and Shuaiyong Li. State-of-health estimation and remaining useful life prediction for the lithium-ion battery based on a variant long short term memory neural network. *Journal of Power Sources*, 459:228069, 2020. ISSN 03787753. doi:10.1016/j.jpowsour.2020.228069.

[128] Yitao Wu, Qiao Xue, Jiangwei Shen, Zhenzhen Lei, Zheng Chen, and Yonggang Liu. State of health estimation for lithium-ion batteries based on healthy features and long short-term memory. *IEEE Access*, 8:28533–28547, 2020. ISSN 2169-3536. doi:10.1109/ACCESS.2020.2972344.

[129] Alessandro Falai, Tiziano Alberto Giuliacci, Daniela Anna Misul, and Pier Giuseppe Anselma. Reducing the computational cost for artificial intelligence-based battery state-of-health estimation in charging events. *Batteries*, 8(11):209, 2022. ISSN 2313-0105. doi:10.3390/batteries8110209.

[130] Jikai Bi, Jae-Cheon Lee, and Hao Liu. Performance comparison of long short-term memory and a temporal convolutional network for state of health estimation of a lithium-ion battery using its charging characteristics. *Energies*, 15(7):2448, 2022. ISSN 1996-1073. doi:10.3390/en15072448.

[131] Sungwoo Jo, Sunkyu Jung, and Taemoon Roh. Battery state-of-health estimation using machine learning and preprocessing with relative state-of-charge. *Energies*, 14(21), 2021. ISSN 1996-1073. doi:10.3390/en14217206.

[132] Shaishai Zhao, Chaolong Zhang, and Yuanzhi Wang. Lithium-ion battery capacity and remaining useful life prediction using board learning system and long short-term memory neural network. *Journal of Energy Storage*, 52:104901, 2022. ISSN 2352-152X. doi:https://doi.org/10.1016/j.est.2022.104901.

[133] Sahar Khaleghi, Danial Karimi, S. Hamidreza Beheshti, Md. Sazzad Hosen, Hamidreza Behi, Maitane Berecibar, and Joeri Van Mierlo. Online health diagnosis of lithium-ion batteries based on nonlinear autoregressive neural network. *Applied Energy*, 282:116159, 2021. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2020.116159.

[134] Hicham Chaoui and Chinemerem Christopher Ibe-Ekeocha. State of charge and state of health estimation for lithium batteries using recurrent neural networks. *IEEE Transactions on Vehicular Technology*, 66(10): 8773–8783, 2017. doi:10.1109/TVT.2017.2715333.

[135] Safieh Bamati and Hicham Chaoui. Lithium-ion batteries long horizon health prognostic using machine learning. *IEEE Transactions on Energy Conversion*, 37(2):1176–1186, 2022. doi:10.1109/TEC.2021.3111525.

[136] spider_plot. https://github.com/NewGuy012/spider_plot/releases/tag/20.2. Accessed: 20 January 2023.