ORIGINAL RESEARCH **Open access**

A Predictive Energy Management Strategy for Extended-Range Electric Vehicles Using GA-LSTM and Double Deep Q-Networks

Samuel Boateng[1*], Kwesi Mensah[1]

Abstract

This study introduces an energy management model that uses a double deep Q-network (DDQN) combined with a long short-term memory (LSTM) neural network for velocity forecasting within a model predictive control (MPC) architecture. A genetic algorithm (GA) is employed to optimize the initial learning rate and the LSTM-based velocity prediction model's neuron dropout rate. Predictive outcomes indicate that across various velocity prediction timeframes, the root-mean-square error yielded by the GA-LSTM approach is lower than that of the SVR method. The agent's state inputs comprise the forecasted required power, the current state of charge (SOC), and the immediate power demand. At the same time, real-time strategy regulation is executed via the MPC approach. Simulation outcomes reveal that the introduced regulation scheme decreases equivalent fuel usage by 0.0354 kg relative to DDQN, by 0.8439 kg relative to ECMS, and by 0.742 kg relative to the power-following management scheme. The discrepancy between the presented method and the dynamic programming strategy amounts to a mere 0.0048 kg, or 0.193%, concurrently ensuring that the power battery's SOC stays steady. Lastly, a hardware-in-the-loop simulation validates the strong real-time execution capabilities of the developed strategy.

Keywords Deep reinforcement learning, Genetic algorithm, Extended-range vehicle, LSTM speed prediction, Model predictive control (MPC), Double deep Q-network (DDQN)

*Correspondence:
Samuel Boateng
samuel.boateng@ucc.edu.gh
[1] Department of Intelligent Energy Systems, Faculty of Engineering, University of Cape Coast, Cape Coast, Ghana

Introduction

Driven by the escalating gravity of ecological and resource challenges, alternative energy vehicles have garnered heightened interest lately. Consequently, academic exploration focusing on mitigating emissions and conserving energy has expanded. Zhao *et al*. [1] evaluated the environmental impacts of unregulated emissions from internal combustion engine vehicles and advocated for the "Oil-to-Gas" initiative in Qingdao. To address fuel depletion and emissions, pure electric options are being actively deployed. Although pure electric vehicles (PEVs) offer prominent benefits like minimal emissions and superior efficiency, their driving range remains limited by low battery capacities, lengthy recharging durations, and a sparse distribution of charging stations, which severely hinders their widespread adoption. Consequently, extended-range electric vehicles have emerged as a focal point of investigation, given that integrated range extenders can persistently replenish the onboard battery, substantially boosting total travel range. Energy management within these systems is achieved by an optimization strategy that distributes power loads between the battery and the range extender. This control mechanism dictates the operational status (on/off) and the precise power output of the range extender across varying time intervals.

This is an open access article distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License (CC BY-NC-SA 4.0), which permits non-commercial use, distribution, reproduction, and adaptation in any medium or format, provided that appropriate credit is given to the original author(s) and source, a link to the license is provided, and any changes made are indicated. Adapted material must be distributed under the same license.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

Coinciding with recent breakthroughs in artificial intelligence, deep learning technologies have found increasingly widespread utility in the domain of energy management [2-5]. Shen introduced a velocity forecasting approach blending BP neural networks with Markov chains, incorporating model predictive control to align speed predictions with minimized equivalent fuel usage and enabling an adaptive equivalence factor [6]. To achieve an adaptive equivalence factor, Wang *et al*. [7] developed a management framework based on generalized regression neural networks (GRNN) within an MPC layout. A hybrid forecasting technique integrating convolutional neural networks (CNN) and long short-term memory (LSTM) networks was put forward by Xing *et al*. [8], who constructed a corresponding energy management scheme within an MPC structure. To identify driving conditions, Chen *et al*. [9] devised an LSTM-centric energy management methodology that dynamically calibrates and optimizes engine velocity across distinct operational environments [9]. Utilizing velocity forecasting alongside MPC-based energy regulation, Han *et a*l. [10] explored how electric machine thermodynamics influence overall energy efficiency. Lin *et al.* [11] introduced an energy distribution strategy using ensemble learning for velocity prediction under the tenets of model predictive control (MPC), incorporating an adaptive reference state of charge tailored to the driving mode. To handle energy management in hybrid electric drivetrains, Ritter presented a novel methodology for embedding extended, uncertain prediction horizons into a stochastic MPC scheme [12]. To elevate the energy economy of transit buses in city environments, Li *et al*. [13] advanced a tiered predictive energy management approach centered around driver classification and conduct.

To address the necessity for automobiles to operate under distinct conditions across various driving modes and to facilitate more precise mode transitions for enhanced fuel conservation, Yu *et al.* [14] established vehicle operation modes via torque forecasting and refined the energy control scheme to bolster efficiency. Incorporating Bayesian regularization, Chen *et al*. [15] introduced an intelligent equivalent consumption minimization strategy (ECMS) utilizing dual neural networks, alongside a novel correction technique that dynamically tunes the equivalence factor to yield near-optimal fuel efficiency without relying on a reference state-of-charge. Wei *et al*. [16] formulated a novel driving style recognition methodology aimed at enhancing fuel economy and optimizing power allocation by categorizing driving segments via a K-means clustering algorithm. Numerous other investigators have developed distinct approaches to overcome the challenges associated with calculating the equivalence factor across diverse vehicular operating states. For power distribution across the multiple energy components of plug-in hybrid electric vehicles (PHEVs), Chen *et al*. [17] proposed an energy management model using double-Q learning within a model predictive control framework. Lin *et al*. [18] detailed a journey-adaptive ECMS that pairs neural networks with a forgetting-factor least squares algorithm to forecast vehicle speed and road slope, dynamically modifying the equivalence factor based on remaining travel distance to enhance overall vehicle efficiency. Zhao *et al.* [19] applied a parameter identification technique to resolve a “black-box” hybrid vehicle model. They developed a two-level model predictive control strategy to optimize and regulate the operation of the hybrid system, validating its efficacy against a benchmark problem that demonstrated notable fuel savings.

While these management tactics integrate neural networks with ECMS principles to predict and regulate future operating states to a certain extent, calculating the precise equivalence factors within ECMS remains problematic. In contrast to ECMS, reinforcement learning models can undergo training across a vast array of operating profiles to determine superior operational parameters. Furthermore, the training phase is decoupled from the execution phase, meaning that saving and deploying the trained agent incurs minimal computational delay.

By embedding reinforcement learning into an MPC architecture and coupling it with a novel velocity forecasting model, Yang *et al*. [20] established a fresh energy management approach that markedly improves fuel efficiency; however, despite incorporating reinforcement learning, this strategy features insufficient enhancements to the underlying velocity prediction framework.

Conventional rule-based control logic struggles to deliver optimal performance under intricate driving scenarios. In contrast, reinforcement learning techniques focused solely on immediate power demands cannot effectively plan across the time domain. Building upon prior literature, this study addresses the economic efficiency challenges of extended-range vehicles through the following contributions:

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

(1) Construct a mathematical model of the extended-range vehicle encompassing the complete powertrain configuration, and determine the most efficient operating curve for the range extender.

(2) Develop both an LSTM and a support vector regression (SVR) velocity forecasting model, establishing a training dataset composed of diverse driving cycles including NEDC, WLTC, UDDS, HEFET, FTP75, and EUDC.

(3) Refine the starting learning rate and neuron dropout probability within the neural network via a genetic algorithm to heighten the accuracy of the LSTM velocity predictions, subsequently comparing its performance against the SVR prediction technique.

(4) Apply the MPC framework to regulate the power output of the range extender by integrating DDQN deep reinforcement learning energy management with the velocity forecasting module, thereby bolstering total vehicle efficiency.

(5) Perform Simulink simulations to benchmark the regulatory performance of the introduced method against alternative control schemes, followed by hardware-in-the-loop (HIL) testing to confirm the real-time operational capability of the strategy.

Powertrain Modeling

This investigation centers on a plug-in extended-range electric vehicle. The battery pack and the range extender jointly deliver energy for the total system. The traction motor supplies propulsion for the vehicle, while the range-extending unit is fundamentally composed of an internal combustion engine mechanically coupled to a generator. Via a DC/DC converter, the energy storage system links to both the drive motor and the range-extending component. The architectural layout of the vehicle investigated herein is delineated in **Eq. 1,** while the specifications of the primary components are detailed in **Table 1 (Figure 1).**

The operational regime of the range-extended vehicle is illustrated in **Figure 2**. At the initial phase of travel, the vehicle operates purely on electric power. Once the State of Charge (SOC) hits the activation threshold for the range extender, the vehicle switches to range-extension mode. In this state, the range extender operates under a single-point profile because single-point operation yields peak efficiency. However, this high-efficiency envelope does not coincide with the high-power zone, which prevents high-power battery discharging when the SOC rises. When the SOC declines to its minimal limit, the excessively depleted state compromises the battery's operational lifespan. Simultaneously, further reduction in SOC leads to challenges such as an inadequate power supply from the battery pack. Consequently, the range extender engages the power sustenance mode. Under this condition, the range extender shifts away from strict single-point operation and aims to stabilize the SOC to prevent any subsequent drop. Thus, managing the power sustenance mode requires a balanced optimization of fuel economy alongside battery health. Drawing from previous investigations, dropping below an SOC of 0.3 dramatically degrades battery performance and impairs its overall life cycle. Accordingly, this study establishes the target SOC at 0.3 for the power sustenance mode, focusing on enhancing economic performance and sustaining SOC stability throughout this low-SOC preservation phase.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

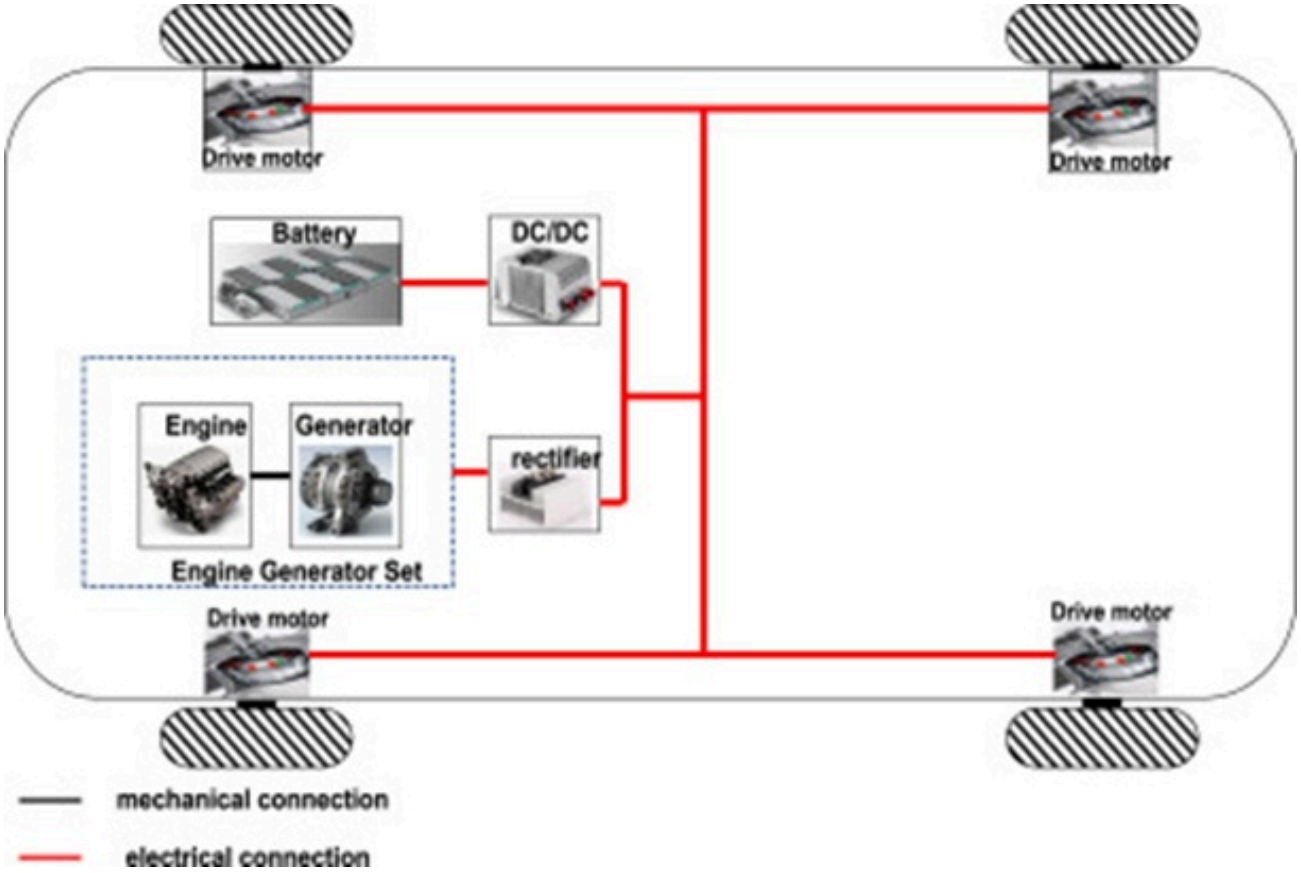


Figure 1. Extended-range vehicle structure.

Table 1. Vehicle powertrain parameters.

Component	Specification	Value	Additional specification	Value
Vehicle system	Gross vehicle weight	7000 kg	Frontal projected area	9 m²
	Drag coefficient	0.55	—	—
Engine unit	Fuel type	Diesel engine	Peak torque	2200 N·m
	Maximum rotational speed	2000 r/min	—	—
Electric motor	Rated type	Permanent magnet synchronous motor	Maximum torque output	1000 N·m
	Maximum operating speed	2670 r/min	—	—
Generator	Peak torque capability	2180 N·m	Maximum rotational speed	2000 r/min
Battery pack	Nominal voltage	550 V	Energy storage capacity	200 Ah

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

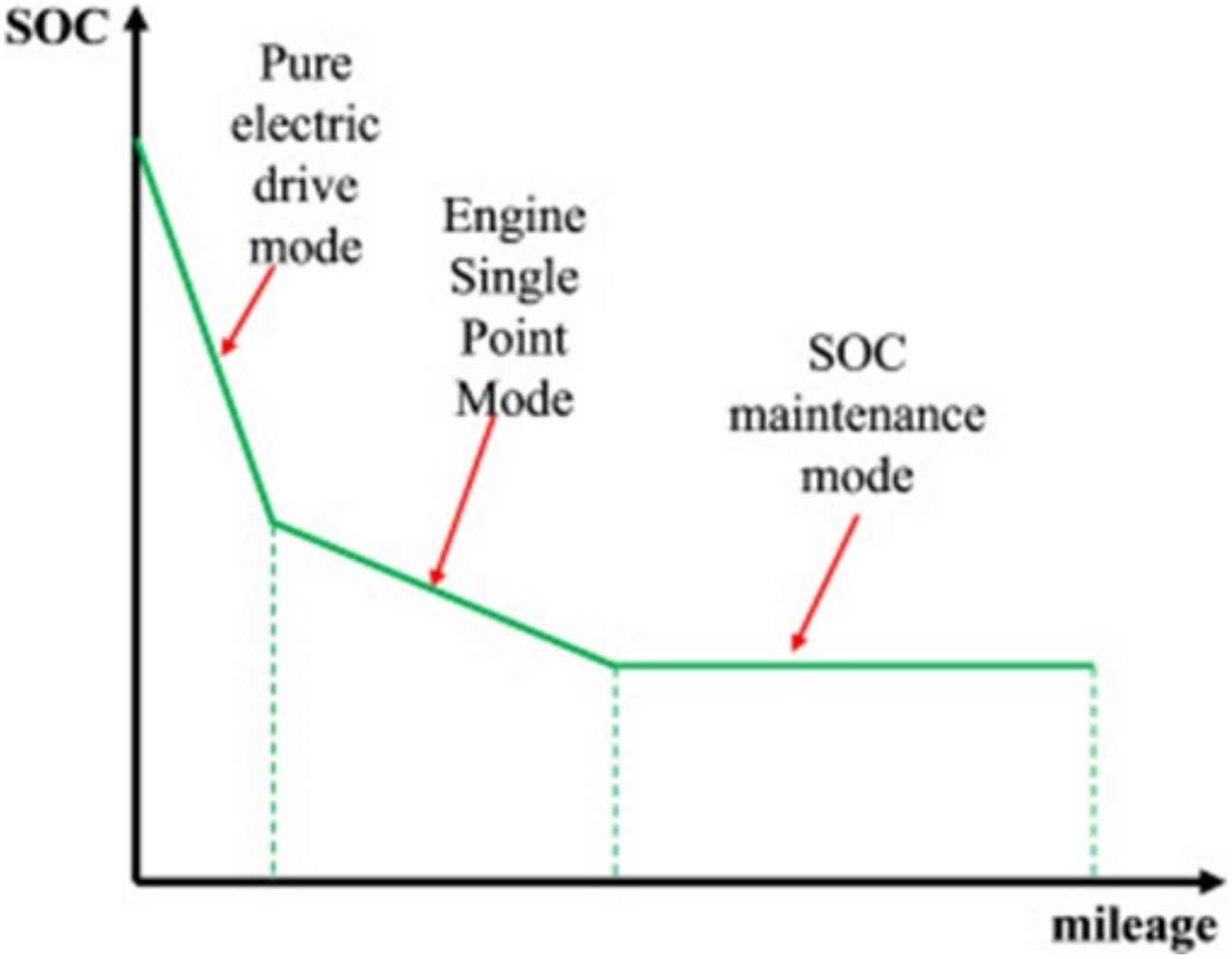


Figure 2. Extended-range vehicle operating modes.

Longitudinal dynamics model of the vehicle

The full-vehicle configuration is used to compute the motion resistance encountered during travel using longitudinal dynamics principles.

$$F = Gf\cos\alpha + \frac{C_D A}{21.15}u^2 + G\sin\alpha + \delta m\frac{du}{dt} \quad (1)$$

In this context, G represents the vehicle's gravitational force (N); \alpha denotes the road gradient; C_D signifies the aerodynamic drag coefficient; A indicates the frontal cross-sectional area m^2; \delta represents the rotational mass correction factor; m is the total mass (kg); $\frac{du}{dt}$ and the remaining term symbolizes the forward driving acceleration (m/s^2).

Engine generator set

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

In this study, the efficiency characteristics of both the engine and generator are depicted in **Figure 3.**

The rate of engine fuel consumption is determined via the following approach:

$$m_f = T_{eng} \times n_{eng} \times be(T_{eng}, n_{eng}) \quad (2)$$

Here, m_f represents the fuel consumption rate (kg/h);T_{eng} is the next variable that denotes engine torque N m;n_{eng} is the engine rotational speed (r/min); and *be* stands for the brake specific fuel consumption rate (kg/kWh).

Given that the engine and generator are coupled mechanically, they maintain identical rotational velocities and torque levels. As a result, the optimal operational pathway of the range extender can be derived from the combined engine-generator efficiency maps. Consequently, **Figure 4** illustrates the optimized fuel economy curve for the range-extending unit [21].

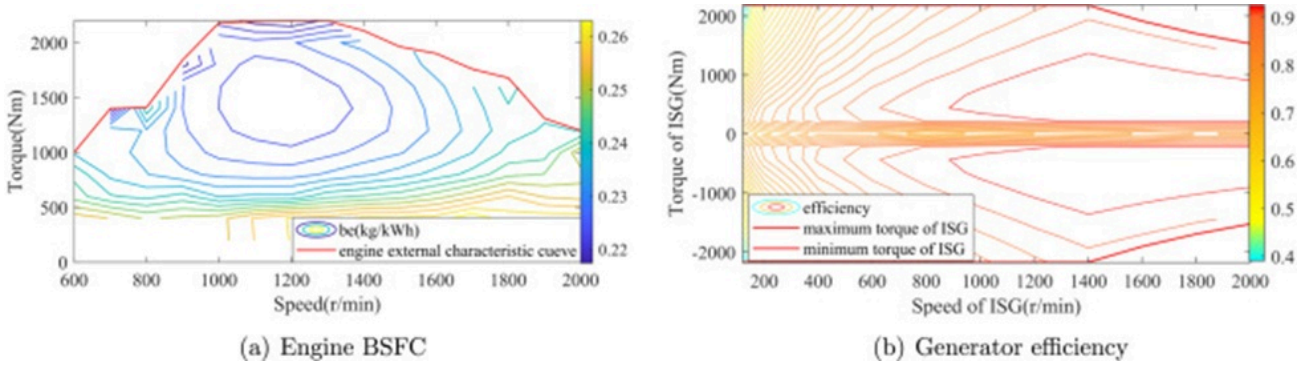


Figure 3. Engine and generator efficiency map.

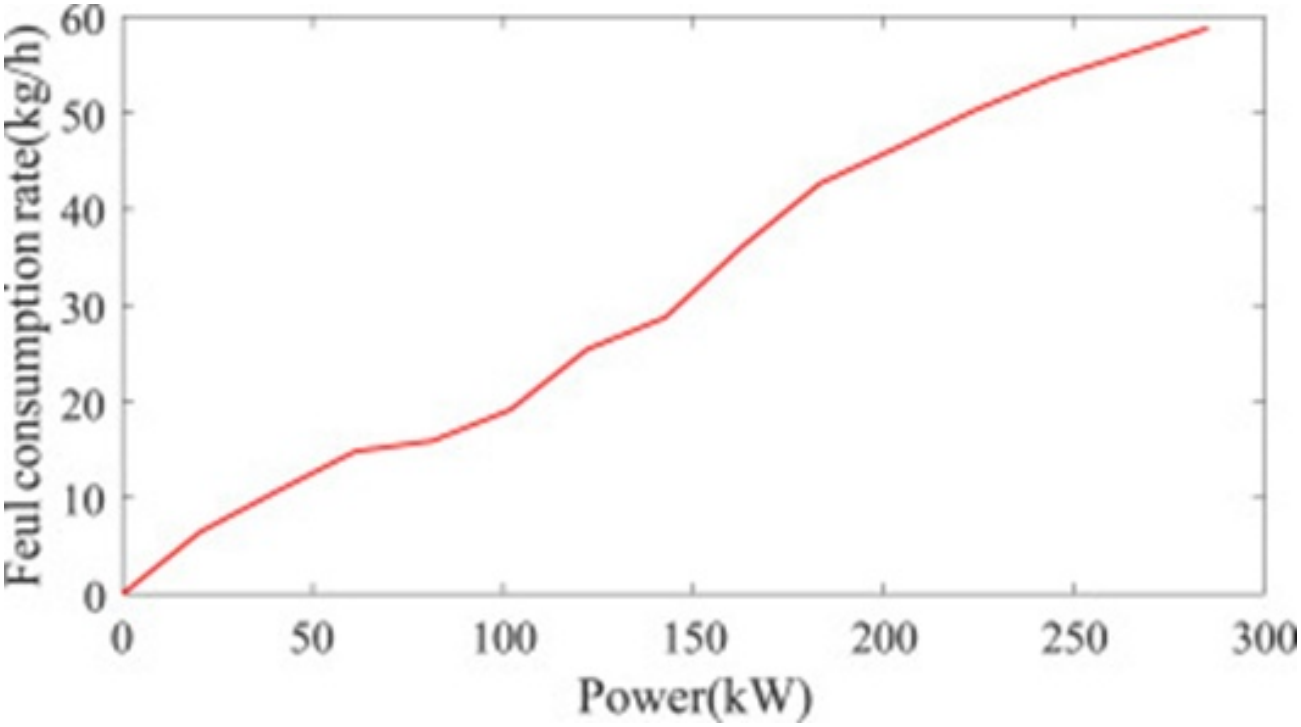


Figure 4. Range extender's optimal fuel consumption curve.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

Battery model

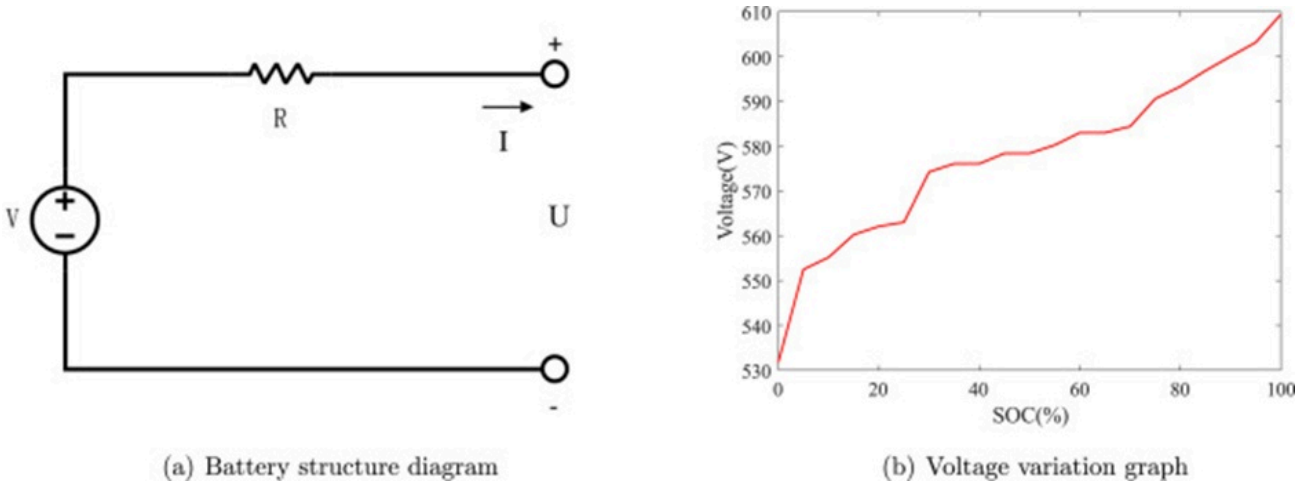
The traction battery is linked to both the propulsion motor and the generator via an inverter, enabling it to supply electrical power to the motor or store energy generated by the generator. This study models the power battery using an equivalent circuit configuration [22], treating the battery as an open-circuit voltage source positioned in series with an internal resistance. The schematic layout of the battery and the relationship between its terminal voltage and SOC are detailed in **Figure 5**, where V represents the open-circuit voltage of the traction battery, I denotes the current, and R signifies the internal resistance.

The calculations for the battery current and current SOC follow the equation below [23]:

$$I(t) = \frac{U_{VOC}(t)}{2R} - \sqrt{\frac{U_{VOC}(t)^2}{2R^2} - \frac{P_b(t)}{R}} \quad (3)$$

$$SOC(t) = SOC_{init} + \int_{t_0}^{t} \frac{U_{VOC} - \sqrt{U_{VOC}^2 - 4RP_b}}{2RQ} dt \quad (4)$$

In this relation, the first term SOC_{init} signifies the initial SOC value;U_{voc} the second denotes the battery's open-circuit voltage; R represents the internal resistance; P_b indicates the output power; and Q stands for the overall capacity.



(a) Battery structure diagram (b) Voltage variation graph

Figure 5. Battery model.

Speed Prediction

In this research, model predictive control (MPC) serves as the bridge integrating the velocity forecasting module with the DDQN-driven energy management architecture [24]. First, a dedicated training dataset is assembled to train the neural network. The dataset utilized here incorporates six distinct driving schedules: NEDC, WLTC, UDDS, HWFET, FTP75, and EUDC, as illustrated in **Figure 6.** To maximize the predictive performance of the LSTM velocity model, a genetic algorithm is deployed to tune its hyperparameters, using the WTVC driving cycle as a validation dataset for hyperparameter adjustment **(Figure 7)**. Once these hyperparameters are established, the LSTM velocity forecasting model is trained. The trained network then generates speed predictions utilizing a rolling window technique. Ultimately, the control strategy computes the required power across the predictive horizon based on the forecasted vehicle speed. This anticipated

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

power demand and the corresponding SOC over the prediction horizon are supplied to the deep reinforcement learning agent as state inputs. The trained agent then determines the optimal power output for the range extender via its interactively trained network, which is continuously updated within the MPC framework to ensure real-time energy management [25].

The overall workflow diagram illustrating the velocity forecasting and energy management approach is depicted in **Figure 8.**

This section evaluates two primary machine learning models for vehicle speed forecasting:

(1) A vehicle speed forecasting model utilizing a GA-LSTM approach.

(2) A vehicle speed forecasting model utilizing Support Vector Regression (SVR).

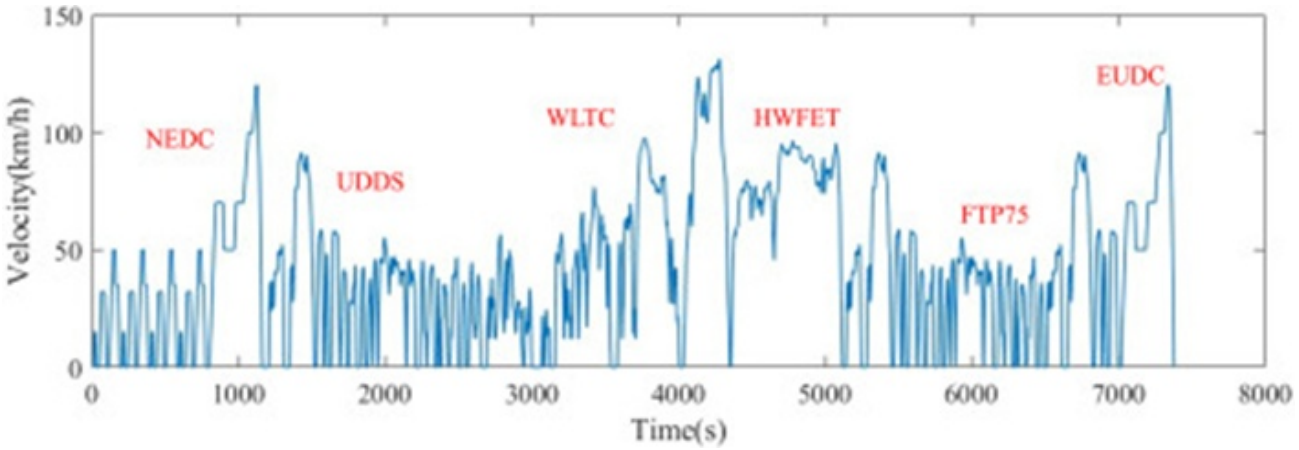


Figure 6. Vehicle speed prediction training set data.

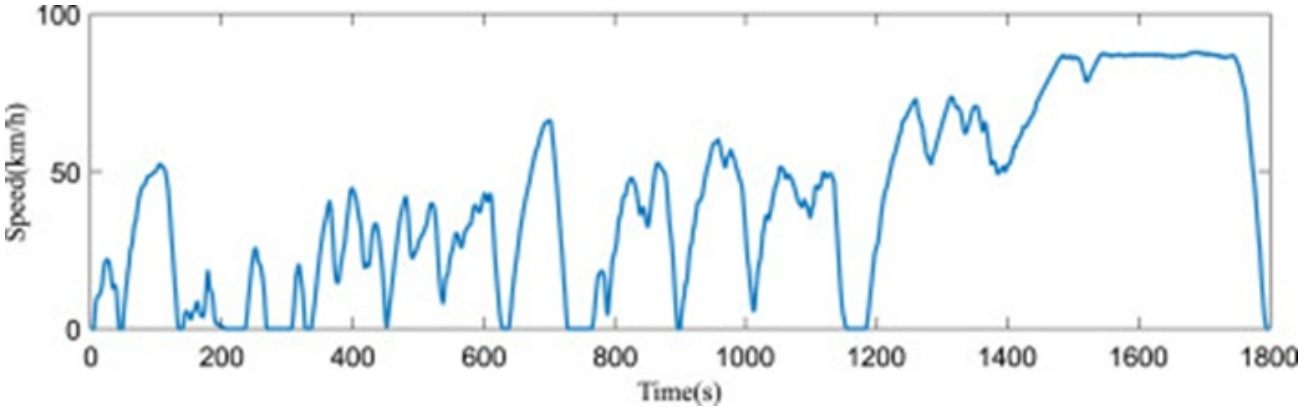


Figure 7. Validation set WTVC.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

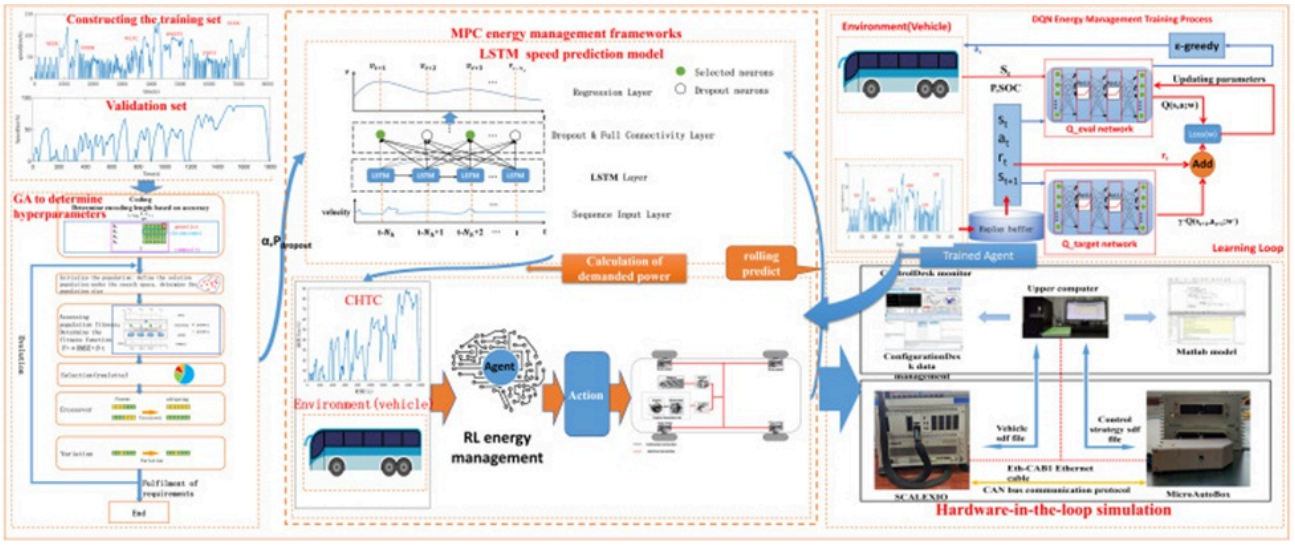


Figure 8. Flow chart of speed prediction and DDQN energy management study.

SVR-based speed prediction

Support Vector Regression (SVR) represents a machine learning methodology derived from the Support Vector Machine (SVM) algorithm, tailored for addressing regression tasks and forecasting continuous numerical targets. It finds widespread application across domains like finance, meteorology, medicine, and diverse predictive analyses.

In the SVR framework, the algorithm maximizes the margin separating the training datasets from the predicted values by establishing an optimal hyperplane, ensuring that the prediction error remains below a predefined boundary. The model incorporates an \epsilon-insensitive loss function (l_\epsilon) to tolerate prediction discrepancies within a specific margin. By applying dual formulation and the Karush-Kuhn-Tucker (KKT) conditions, the Lagrangian function is converted into the core SVR solving equation. Concurrently, a kernel function is used to map data from a low-dimensional space into a higher-dimensional space to solve non-linear characteristics more easily. Due to the high adaptability of the Gaussian kernel function, its profile can be modified via parameters without requiring explicit feature mapping, resulting in low computational demand. Consequently, this study selects the Gaussian kernel function for SVR-based vehicle speed forecasting, with its mathematical expression detailed in **Eq. 5.**

$$\kappa(x, x_i) = \exp(-g\|x - x_i\|^2) \quad (5)$$

Where x_i denotes an individual sample data point, and g represents the bandwidth of the kernel function.

The algorithm optimizes the model by calibrating two primary hyperparameters: the penalty factor C associated with the insensitive loss and the kernel function bandwidth g. **Figure 9** presents a structural schematic of low-dimensional linear regression prediction via SVR.

Figure 10 outlines the systematic SVR workflow for solving the velocity forecasting problem, where historical velocity across a time horizon h is supplied as an input using a rolling window, and the future velocity across a predictive horizon p is generated as an output via over-loop forecasting.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

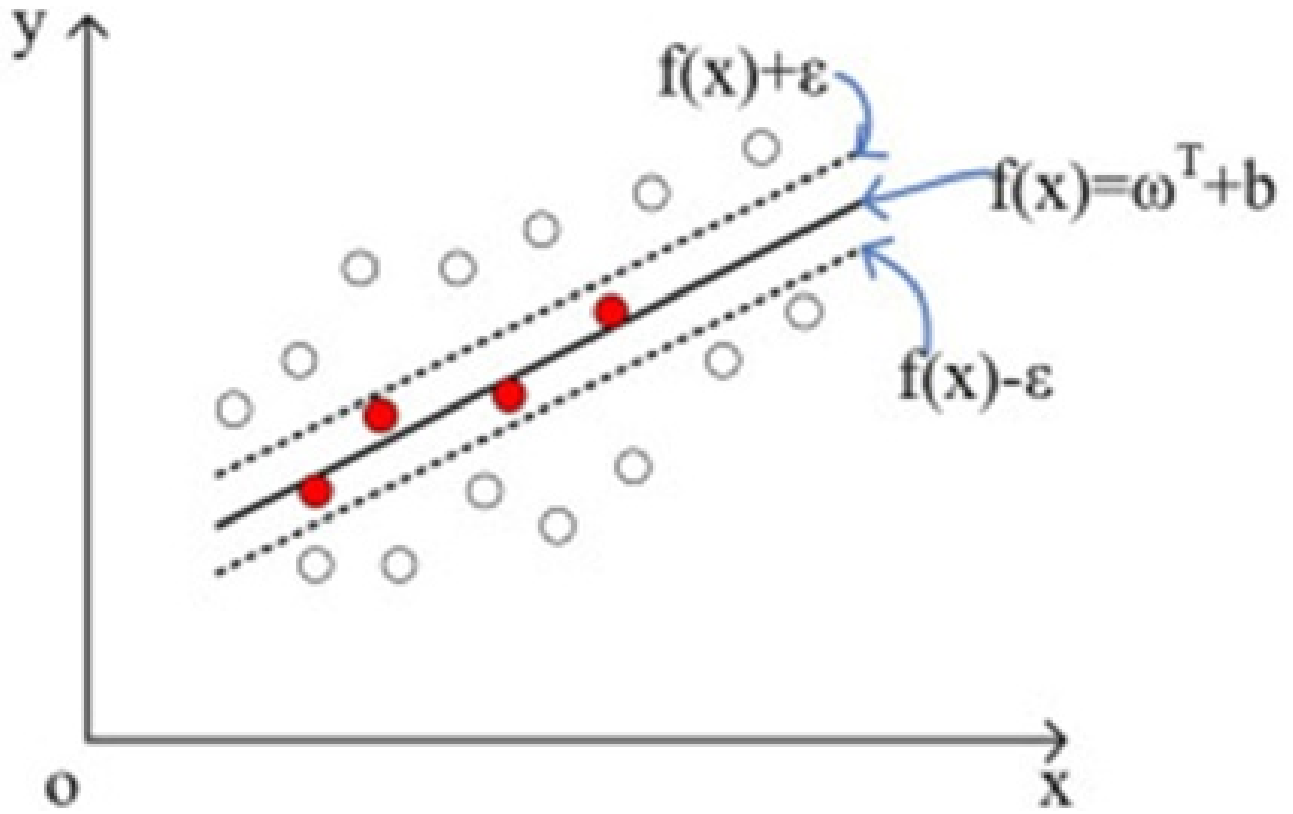
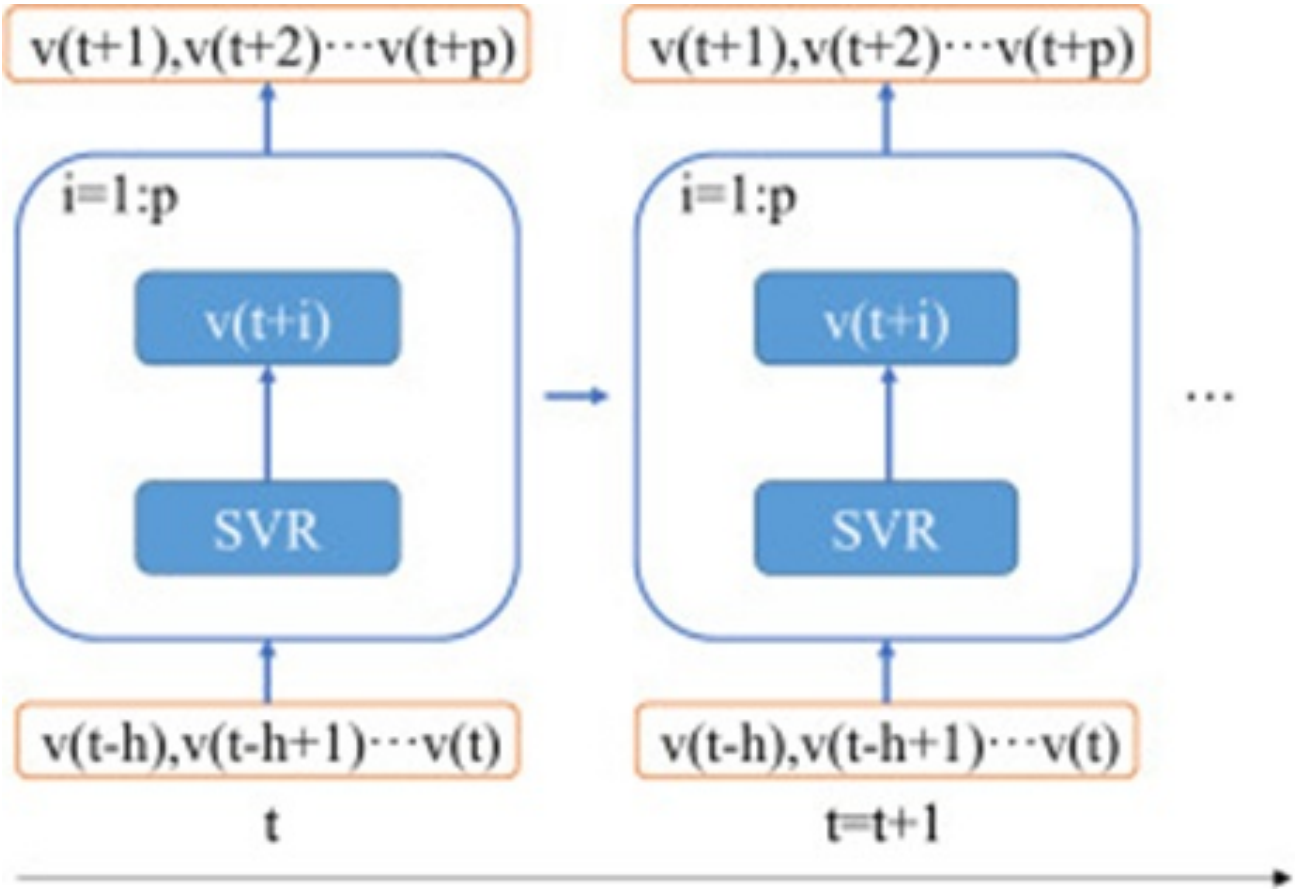


Figure 9. Schematic of SVR low-dimensional linear regression prediction.



© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

Figure 10. Flow chart of SVR speed prediction.

LSTM-based vehicle speed prediction

The LSTM network is a specialized variant of recurrent neural networks (RNNs), featuring a sophisticated architecture equipped with gate mechanisms that preserve memory across extended time steps and permit gradient propagation through time, thereby mitigating the gradient vanishing issues prevalent in standard RNN models. The LSTM architecture is illustrated in **Figure 11** [10]. The LSTM incorporates a cell state unit C to retain long-term information. At any given time step t, the LSTM processes three distinct inputs: the current network input value xt, the hidden state from the previous step h _ {t-1}, and the prior cell state C _ {t-1}. Following internal computations, the network yields the current hidden state h_{t} and the updated cell state C_{t}. In contrast to traditional RNN architectures, the LSTM regulates the cell state C by utilizing three distinct gating systems [26]:

· The forget gate determines the proportion of the previous cell state C{t-1} that should be carried over into C_t.

· The input gate regulates the extent to which the new input xt is incorporated into C_t.

· The output gate controls how much of the current computed state is reflected in the final output value based on [h_{t-1}, xt].

The mathematical expressions for the parameters within this architectural framework are defined in **Eq. 6:**

$$\begin{cases} f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \\ i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \\ g_t = \tanh(W_g[h_{t-1}, x_t] + b_g) \\ C_t = f_t \cdot C_{t-1} + i_t \cdot g_t \\ o_t = \tanh(W_o[h_{t-1}, x_t] + b_o) \\ h_t = o_t \cdot \tanh(C_t) \end{cases} \quad (6)$$

Where ft signifies the output of the forget gate; it represents the output of the input gate; ot denotes the output of the output gate; σ represents the gate activation function; and W stands for the respective weight matrix parameter.

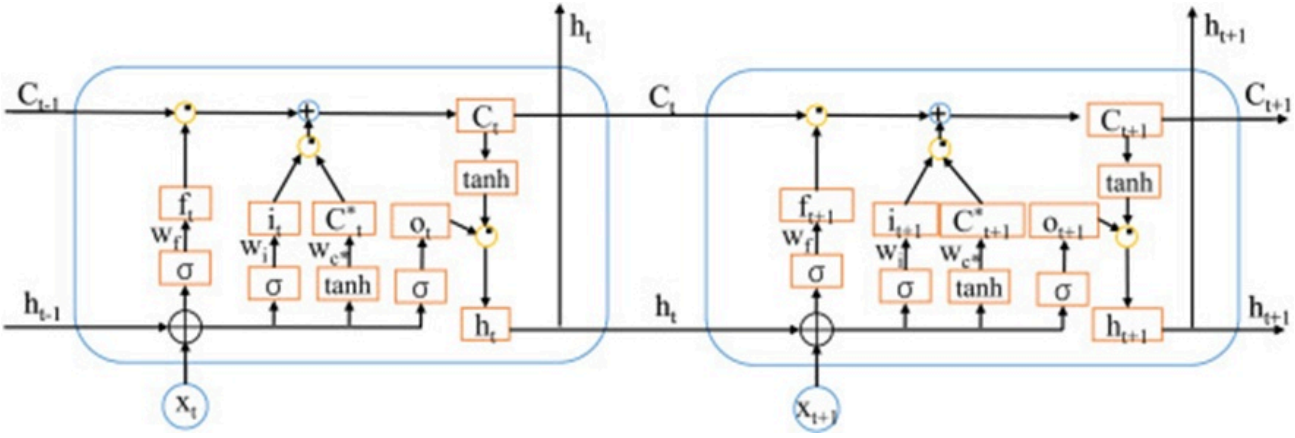


Figure 11. LSTM structure diagram.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

Figure 12 illustrates the LSTM velocity forecasting sequence, where Nh represents the step count of the historical time domain. Using velocity data from the historical horizon h as input, the model forecasts velocity across the predictive horizon p, then updates future steps via a rolling window. To prevent overfitting during training, this work incorporates a dropout layer into the baseline architecture. The neural network randomly deactivates hidden-layer nodes, along with their forward and backward connections, with probability p, constructing an altered network architecture to bolster the model's generalization capabilities and mitigate overfitting. Selecting an excessively low dropout probability makes the model vulnerable to overfitting, whereas an excessively high probability complicates network training. This study employs a genetic algorithm to determine an optimal neuron dropout probability, ensuring that the neural network retains an appropriate regularization rate.

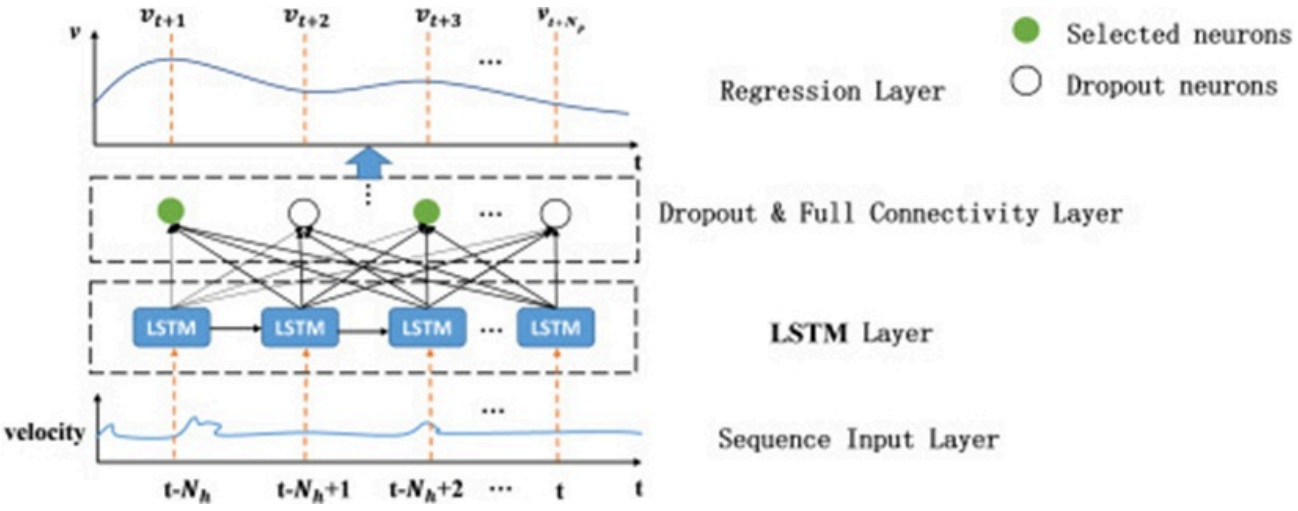


Figure 12. LSTM vehicle speed prediction diagram.

LSTM hyperparameter determination based on genetic algorithm

The Genetic Algorithm (GA) is an optimization and computational search framework modeled after the mechanisms of biological evolution and natural genetics, designed to pinpoint optimal or near-optimal solutions [26, 27]. Its operational loop replicates natural evolutionary processes by generating novel candidate solutions via selection, crossover, and mutation, and preserving high-fitness individuals for subsequent generations through rigorous fitness evaluation. When applying neural networks to regression problems, numerous hyperparameters must be calibrated, a process that is highly time-consuming and laborious when conducted manually using a validation set. In this work, a genetic algorithm is deployed to optimize both the initial learning rate and the dropout probability of the LSTM velocity forecasting network, using the WTVC cycle as a validation benchmark and defining the fitness function as a weighted combination of training duration and root-mean-square error (RMSE). The validation configuration is shown in **Figure 7**. The systematic workflow for GA-driven hyperparameter optimization is detailed in **Figure 13**. First, based on the designated hyperparameter search boundaries, the genetic algorithm's precision and encoding scheme are established. The mathematical relationship between the interval span and encoding precision is defined in **Eq. 7:**

$$\frac{L}{2^n - 1} \leq E \quad (7)$$

Where L denotes the interval span, n represents the length of the binary string, and E signifies the target precision. Subsequently, the actual value corresponding to the binary sequence is computed according to **Eq. 8:**

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

$$X = a + T * \frac{b-a}{2^n - 1} \quad (8)$$

Where X represents the real value mapped from the binary string within the interval [a, b], n is the bit length of the binary string, and T is the decimal equivalent of that binary sequence.

Following the encoding stage, the overall population size is set, and the fitness metric is formulated. This study uses the weighted combination of execution time and root-mean-square error as the fitness function, as shown in **Eq. 9:**

$$f(x_1, x_2) = \alpha t + (1 - \alpha) RMSE \quad (9)$$

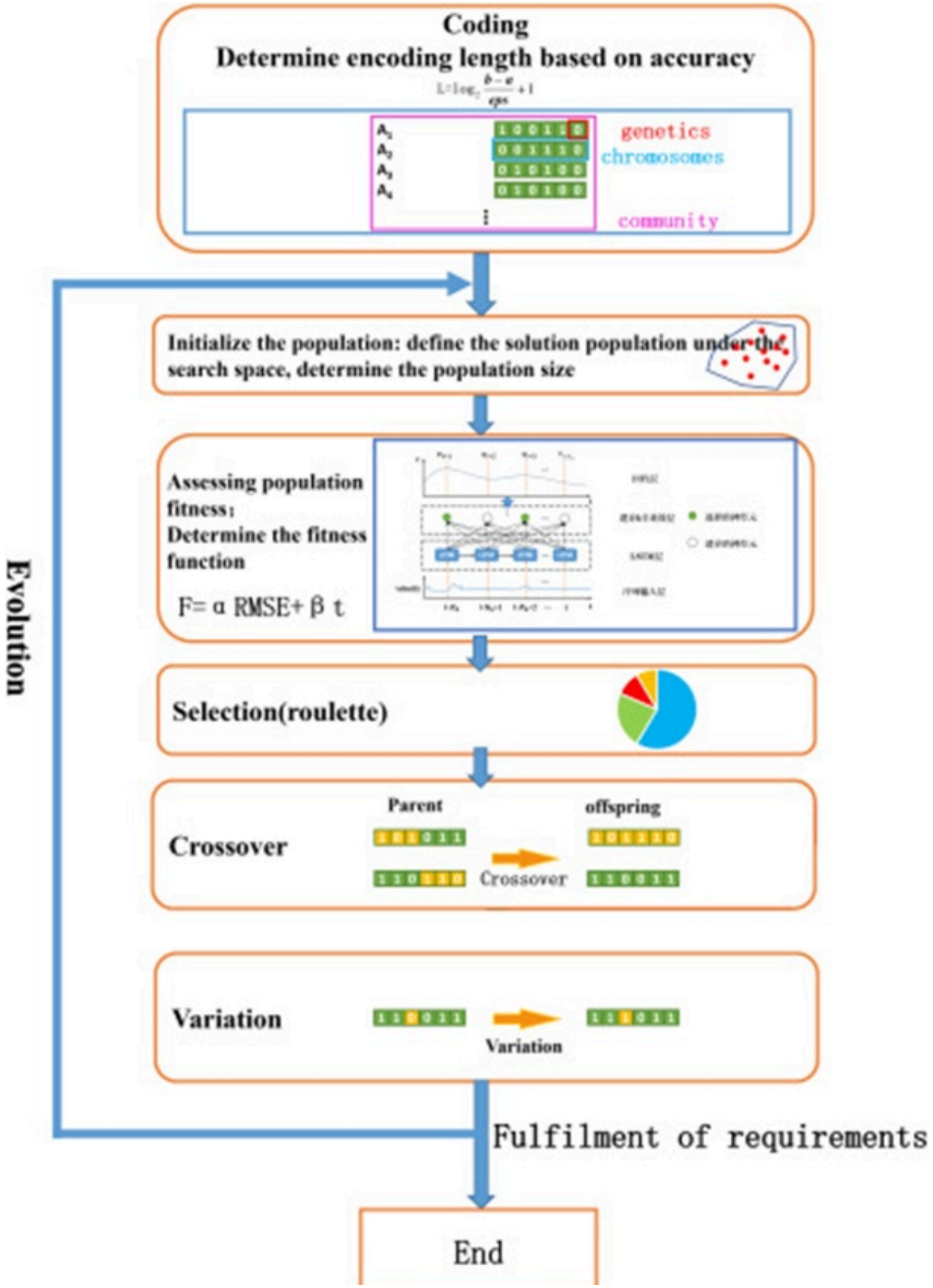
Where α represents the weighting coefficient for the prediction time requirement, RMSE denotes the root-mean-square error of the forecasted velocity evaluated against the validation WTVC profile, and *f* (.) represents the fitness evaluation function.

Thereafter, the genetic algorithm selects candidate individuals based on their evaluated fitness scores. A roulette wheel selection mechanism is employed, where the likelihood of an individual transmitting its genetic traits to the next generation depends on its fitness value. Because the objective here is to minimize the fitness score, lower fitness values yield a higher probability of genetic transmission, thereby filtering out candidates characterized by high root-mean-square errors or lengthy processing times. Next, the algorithm executes crossover and mutation operations across the population.

The crossover and mutation mechanisms are illustrated in **Figure 12**, where pairs of chromosomes swap genetic fragments to form novel offspring, and mutation randomly flips a chromosome's bit from 0 to 1 or 1 to 0 [28]. Both crossover and mutation are carried out stochastically according to predefined probabilities; in this study, the crossover probability is set to 0.35, and the mutation probability is set to 0.15 to preserve the stability of the generated population.

Through continuous evolutionary screening, the optimal initial learning rate is determined to be 0.0064, and the neuron dropout probability is found to be 0.3889. The convergence history showing the best and mean fitness function values across the iterations is presented in **Figure 14**. After 100 iterations, the fitness function stabilizes. However, minor fluctuations persist due to inherent stochastic variance when training neural networks, even under identical hyperparameter sets.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press



© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

Figure 13. Genetic algorithm optimization neural network hyperparameters flow chart.

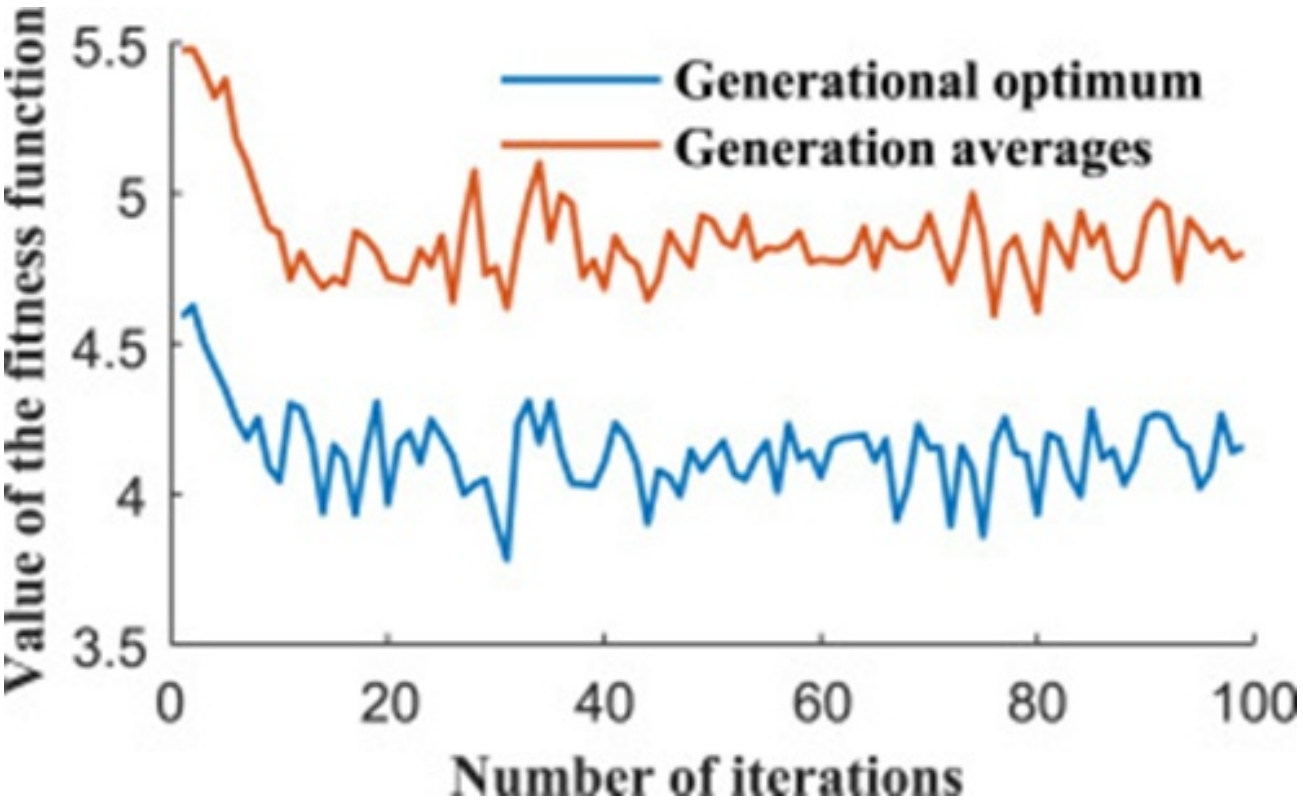


Figure 14. Variation of fitness function.

Speed prediction results and performance comparison

In this study, the vehicle simulation framework was developed using Matlab/Simulink. The velocity forecasting experiments were executed utilizing MATLAB 2022b on a personal computer equipped with an Intel i7-12700H CPU operating at 2.3 GHz and 16 GB of RAM. Once the optimal initial learning rate and neuron dropout rate were mapped out via the genetic algorithm, this specific setup was applied to retrain the LSTM velocity prediction network across 1000 training epochs. Concurrently, the SVR velocity forecasting model was processed using the identical training set illustrated in **Figure 6.**

A forecasting network can only deliver highly accurate estimations if it effectively leverages velocity fluctuations across the historical timeline; however, an overly extended history window balloons the processing overhead, threatening real-time execution. Consequently, drawing from empirical testing and iterative adjustments, the historical window was set to 30 s. Thus, this study fixes the historical timeline at 30 s, gathering the preceding 30 s of vehicular velocity as the primary input matrix for the prediction architecture.

For a prediction horizon of 5 s, the root-mean-square error (RMSE) for the GA-LSTM network reached 2.8842. Meanwhile, the SVR model (c = 0.76, g = 0.6) yielded an RMSE of 5.1028. The corresponding training loss trajectories and RMSE profiles are plotted in **Figure 15.**

The forecasting outcomes for these distinct velocity prediction strategies are presented in **Figure 16,** which contrasts the performance across predictive horizons of 3 s, 5 s, and 7 s. A comprehensive breakdown of processing times and prediction precisions across these configurations is compiled in **Table 2**. The data in **Table 2** demonstrate that the GA-

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

LSTM method yields a minimized RMSE alongside heightened accuracy, effectively mirroring upcoming speed variations. By evaluating the accuracies of these varying prediction spans and assessing how velocity precision impacts energy distribution performance, the predictive horizon for the MPC framework in this study was set to 5 s to enable the DDQN energy management model with superior trajectory planning. Extending the forecasting window excessively introduces massive prediction deviations, which degrade the reinforcement learning agent's capacity for accurate planning and weaken fuel efficiency. Conversely, an overly restricted prediction window yields insufficient reference value for effective long-term scheduling.

Following the training cycle, the root-mean-square error converged to 0.28688 with a corresponding loss value of 0.0411, successfully satisfying the established accuracy benchmarks. As illustrated in the graphical analysis, both the loss function and the RMSE stabilize effectively after dropping sharply during the initial 50 epochs.

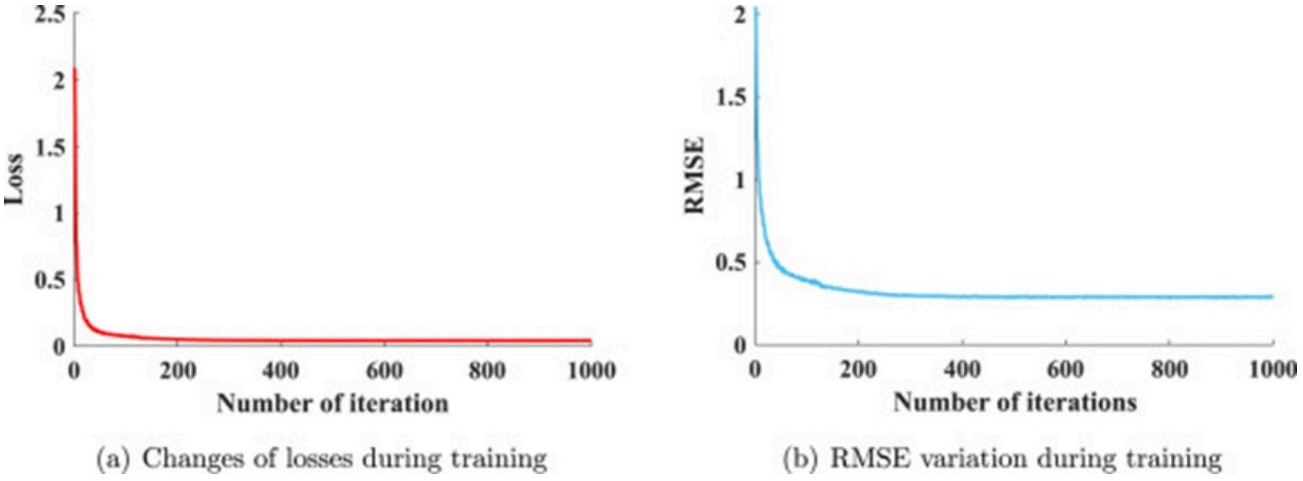


Figure 15. Schematic of loss and root mean square error variation during training.

To establish a clear baseline and contrast the findings, a parallel set of LSTM velocity forecasting simulations was conducted without genetic algorithm optimization, with the comparative results illustrated in **Figure 17**. The graphical trends emphasize the performance leap achieved by the GA-LSTM strategy over the standard LSTM model. Furthermore, **Table 3** offers a structured comparative analysis of the operational parameters for both approaches. This comparison confirms that the GA-LSTM forecasting model exhibits superior performance regarding both algorithmic execution time and estimation accuracy.

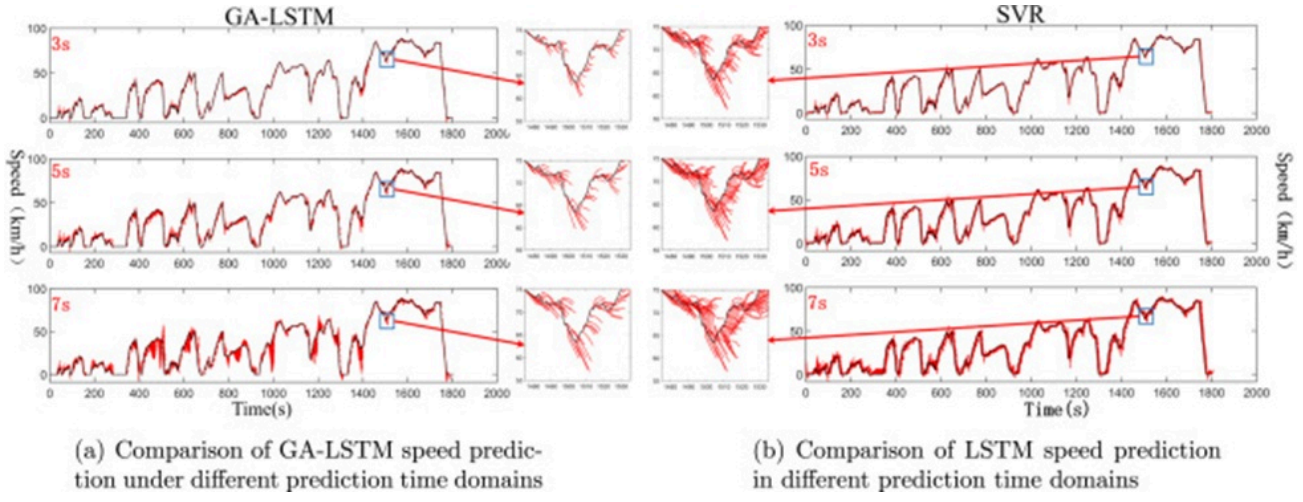


Figure 16. Vehicle speed prediction effects of different prediction methods.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

To dive deeper into these forecasting methods, graphical density profiles were generated for the SVR, GA-LSTM, and baseline LSTM techniques, culminating in the velocity prediction heat maps shown in **Figure 18**. Here, the horizontal axis denotes the measured actual speed. The vertical axis represents the projected speed. Regions transitioning toward a red hue signify a higher frequency of precisely predicted velocity values. In contrast, the solid red diagonal line represents the ideal condition where actual velocity matches the predicted velocity perfectly. The plots demonstrate that the velocity tracking under the GA-LSTM methodology aligns much more closely with the true values than the SVR alternative, displaying a tight concentration along the ideal trajectory line. Additionally, the distribution heat maps verify that the predicted velocities generated by the GA-LSTM framework are significantly more clustered around the actual vehicle speeds than those from the standard LSTM model lacking genetic algorithm tuning.

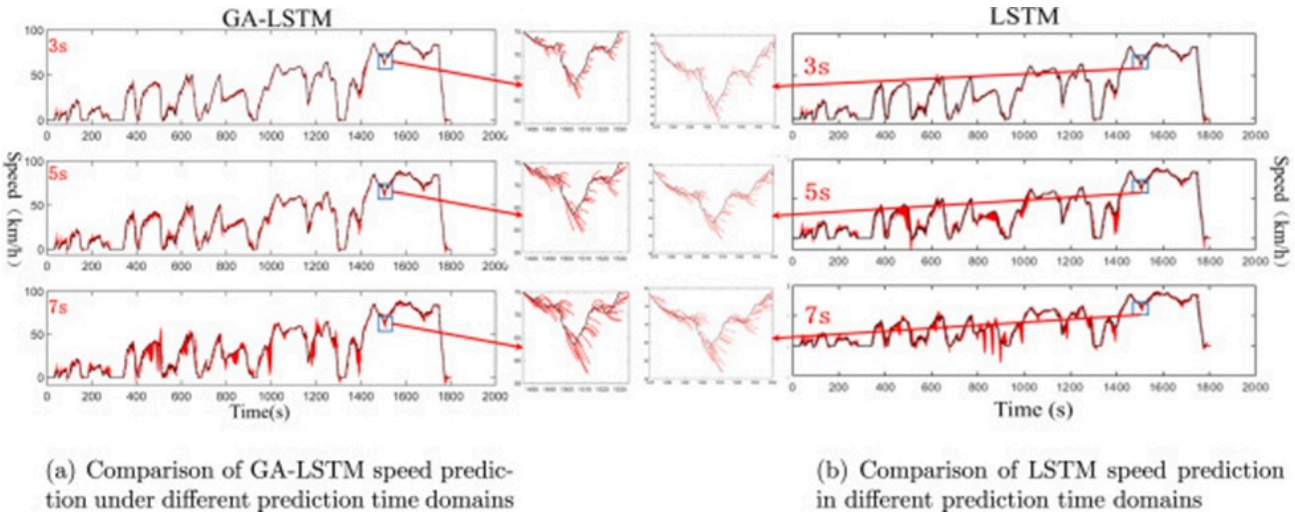


Figure 17. Vehicle speed prediction effects of LSTM and GA-LSTM prediction methods.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

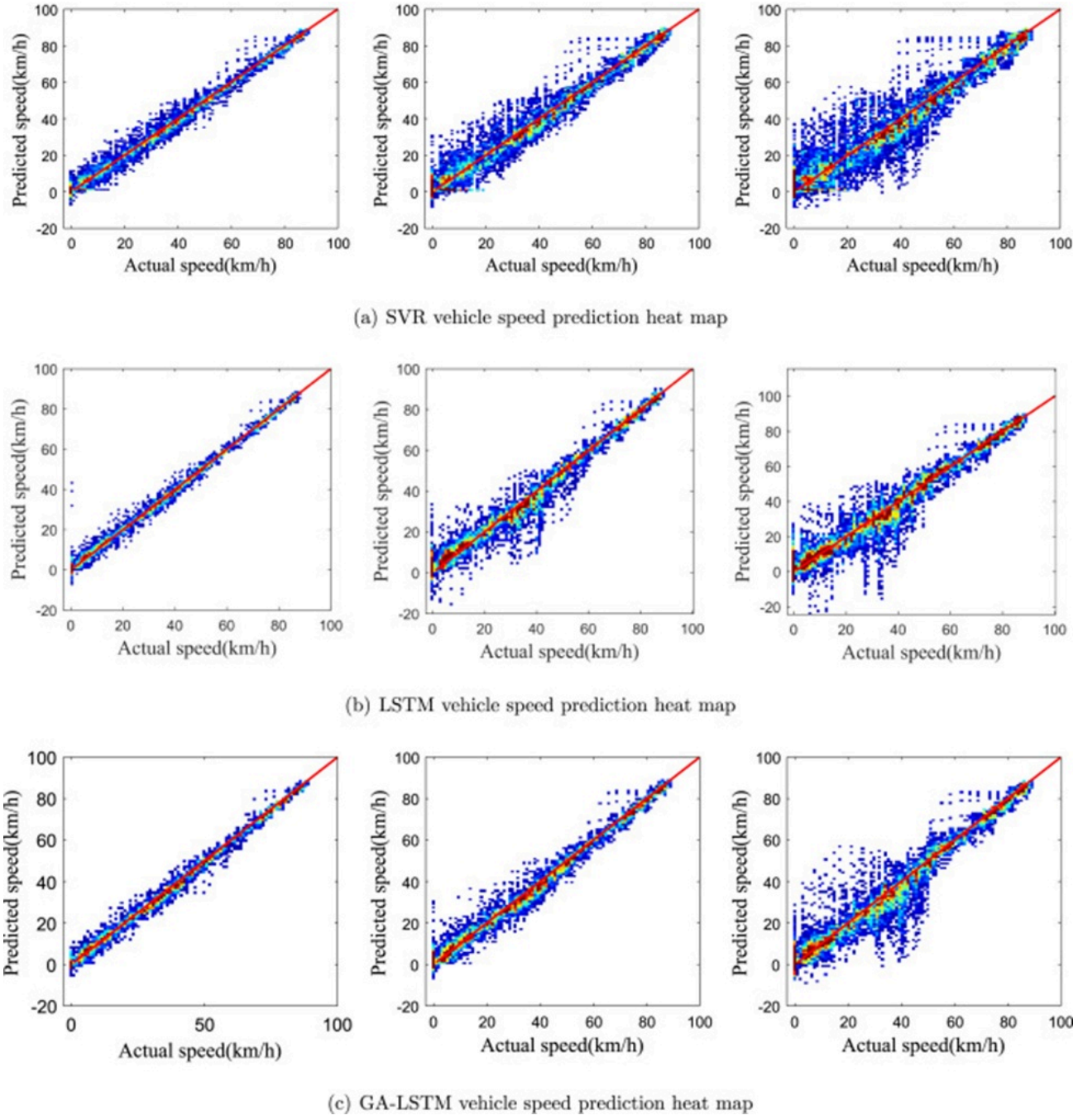


Figure 18. Heat map of vehicle speed prediction with different prediction methods (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

Table 2. Performance comparison of different vehicle speed prediction methods.

Predictive time domain	SVR		LSTM	
	RMSE	Average prediction time (s)	RMSE	Average prediction time (s)
3 s	2.9869	0.006218	1.994	0.003618
5 s	5.1028	0.013402	2.8842	0.004467
7 s	7.135	0.02163	5.6356	0.004591

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

Table 3. Performance comparison of GA-LSTM and LSTM vehicle speed prediction methods.

Predictive time domain	SVR		LSTM	
	RMSE	Average prediction time (s)	RMSE	Average prediction time (s)
3 s	2.4125	0.004328	1.994	0.003618
5 s	4.1873	0.009532	2.8842	0.004467
7 s	6.254	0.01573	5.6356	0.004591

Deep Reinforcement Learning Energy Management in the MPC Framework

In this research, real-time velocity forecasting and deep reinforcement learning-based energy allocation are integrated via an MPC architecture. MPC is a highly adaptable control framework that can be combined with diverse control algorithms to execute real-time operations. The core concept behind MPC relies on leveraging a representative model, the instantaneous system state, and prospective control variables to project future system outputs. This management sequence is built around three integral phases: the predictive model, rolling horizon optimization, and feedback error correction, with the systematic MPC execution flow illustrated in **Figure 19.**

Selecting the future prediction horizon is essential for effectively merging velocity forecasting with energy regulation. The velocity forecasting metrics indicate that expanding the predictive horizon degrades estimation precision. Although an overly brief prediction span offers excellent accuracy, it restricts the control framework's ability to map out optimized plans over a meaningful timeframe, thereby diminishing the core benefits of a prediction-driven strategy. For these reasons, horizons of 3 s and 7 s were deemed inadequate for this study. Moving forward, guided by insights from previous literature, the predictive horizon for this system is defined as 5 s.

In contrast to supervised and unsupervised learning paradigms, reinforcement learning functions through continuous environmental interaction and iterative trial-and-error adjustments to refine its governing policy. Rather than categorizing data or mapping features based on predefined training sets, reinforcement learning computes an optimal sequence of decisions by driving continuous interactions between an autonomous agent and its surrounding environment [29]. The operational framework of reinforcement learning is detailed in **Figure 20.**

Initially, the agent observes the current state s_t from the environment and executes an action a_t corresponding to that state s_t. Subsequently, the environment updates to a new state based on the impact of action a_t. It provides a feedback loop to the agent in the form of a reward R_{t+1}, which can take a positive or negative value depending on the action's effectiveness. Through this ongoing interaction loop, the agent learns to select increasingly optimized actions. The energy distribution module in this paper focuses specifically on the Double DQN architecture.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

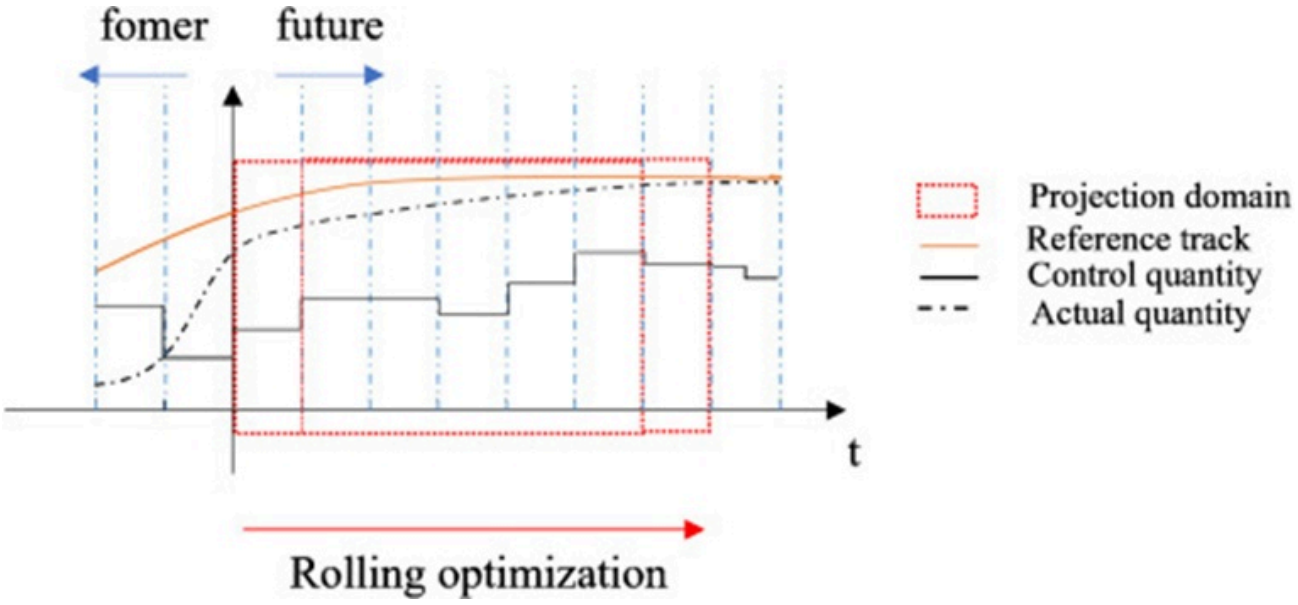


Figure 19. MPC schematic diagram.



Figure 20. Schematic diagram of the reinforcement learning framework.

Markov processes for deep reinforcement learning

A deep reinforcement learning problem is typically framed as a Markov Decision Process (MDP). A standard Markov process transitions into a Markov reward process with the addition of a reward structure. Subsequently, it incorporates a set of allowable actions to establish a full Markov decision process. Within an MDP, the state domain S, action domain A,

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

state transition probability matrix P, reward function R, and discount multiplier \gamma form a characteristic quintuple \langle S, A, P, R, \gamma \rangle.

In this study, we construct a deep reinforcement learning energy management architecture in which battery SOC and predicted velocities across the forecasting horizon serve as state inputs. At the same time, the power output of the range extender defines the action domain. Within the reinforcement learning space, both action selection and state transitions are fundamentally stochastic; specifically, actions are stochastically distributed given a specific state s. The variable \pi defines the probability density of executing action a when encountering state s, while state transitions follow a stochastic distribution given state s. P denotes the probability distribution of these state shifts. Throughout this transition process, the environment yields distinct rewards R corresponding to changes in state, generating a specific reward value R at every designated sampling interval [30].

The action executed at the current time step is evaluated by calculating the discounted return G_t, which represents the discounted weighted sum from the current step t through to the terminal step of the horizon, defined by the expression in **Eq. 10:**

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \quad (10)$$

In this formula, G_t represents the discounted cumulative reward, γ is the discount factor where $\gamma \in$ [0,1], and R_{t+k+1} signifies the reward yielded at the (t+k+1) th step. This formulation ensures that events further into the future carry less weight and have a reduced impact on immediate evaluations.

Because the individual reward R is a stochastic variable, the cumulative discounted return Gt is likewise a random variable, meaning the precise value of Gt cannot be definitively known at the instant t. Consequently, it is necessary to compute the mathematical expectation of Gt to assess the current action and state configuration properly. Taking the expectation of this random variable yields the value function, with **Eq. 11** defining the mathematical expression for the action-value function:

$$Q_\pi(s_t, a_t) = E[G_t | S_t = s_t, A_t = a_t] \quad (11)$$

Where Q_π represents the action-value function, which corresponds to the mathematical expectation of the discounted return Gt, serving as an indicator of the expected value of subsequent states originating from the current state.

By combining **Eqs. 10 and 11**, the Bellman expectation equation for the action-value function can be derived as shown in **Eq. 12:**

$$Q_\pi(s_t, a_t) = E_\pi[G_{t+1} + \gamma Q_\pi(S_{t+1}, A_{t+1}) | S_t] = s, A_t = a \quad (12)$$

Here, s and a represent the specific state and executed action, respectively. To decouple the system from the specific policy function \pi and achieve an objective assessment of the action chosen in the current state, it is necessary to determine the optimal action-value function, as outlined in **Eq. 13:**

$$Q^* = \max_\pi Q_\pi(s_t, a_t) \quad (13)$$

At this stage, the agent's optimized action a^* is defined as:

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

$$a^* = \arg\max_a Q^*(s, a) \quad (14)$$

DQN

Unlike conventional Q-learning, which relies on look-up tables to query the Q-values corresponding to specific states and actions, Deep Q-Networks (DQN) utilize a neural network Q(s, a; w) to approximate the optimal action-value function Q * (s, a) [31]. The neural network continuously adjusts its parameters w to bring the outputs of Q(s, a; w) closer to Q * (s, a). Unlike standard deep learning methods, reinforcement learning uses Temporal-Difference (TD) learning to update these network weights. The TD algorithm does not require running through a simulation completely; instead, it trains the network dynamically through ongoing environmental interactions.

The TD mechanism trains the model primarily by evaluating the discrepancy between predicted Q-values and empirical rewards. This relationship, mapping the estimated Q-values at time step t and step t + 1 within a DQN framework, is represented in **Eq. 15:**

$$Q(s_t, a_t; w) \approx r_t + \gamma Q(s_{t+1}, a_{t+1}; w) \quad (15)$$

Here, Q (st, at; w) signifies the current Q-value estimated by the neural network, while Q (s_{t+1}, a_{t+1}; w) represents the estimated Q-value for the subsequent step. The term rt stands for the immediate reward granted to the agent following an action at the current step, and w indicates the internal network weights. The value Q (s_t, a_t; w) serves as the prediction made at step t. By the time the system reaches step t + 1, the actual reward rt becomes known, making the right side of the expression a more accurate reflection of reality. This right-hand component is designated as the TD target: yt. Throughout the optimization procedure, the DQN selects the specific action a that maximizes the value of Q (s_{t+1}, a_{t+1}; w) for its next move, as formulated in **Eq. 16:**

$$y_t = r_t + \gamma Q(s_{t+1}, a_{t+1}; w) = r_t + \gamma \cdot \max_a Q(s_{t+1}, a; w_t) \quad (16)$$

To minimize the discrepancy between the network's output and the TD target yt, a loss function is computed using the squared deviation between these two values, as outlined in **Eq. 17**. This calculated loss drives subsequent parameter updates:

$$L = \frac{1}{2}(Q(s_t, a_t; w) - y_t)^2 \quad (17)$$

During training operations, DQN frequently incorporates experience replay to maximize data efficiency. This mechanism avoids discarding valuable data by repeatedly recycling past transitions, while simultaneously scrambling sequence patterns to sever temporal correlations between consecutive steps. Network performance can be tuned before training by modifying the replay buffer capacity n. In this study, n is configured to 1,000,000. Once this buffer volume is fixed, the reinforcement learning agent extracts randomized mini-batches from the stored experiences to execute training updates.

DDQN

Value estimation errors are inherently present within standard DQN frameworks. As deduced from **Eq. 17**, calculating the TD target involves a maximization step that, when coupled with bootstrapping, systematically induces overestimation bias. Double DQN (DDQN) was developed to mitigate the detrimental effects of this overestimation phenomenon.

This architecture employs a dual-network configuration:

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

· Q_{eaval} **:** The current evaluation value network.

· Q_{target} **:** The target value network.

While both networks share an identical structure, their parameter update schedules differ; Q_{eaval} modifies its weights during each training iteration, whereas Q_{target} periodically duplicates the weights of Q_{eaval} at longer intervals. As illustrated in **Figure 21**, the DDQN workflow initiates by calculating Q-values via the evaluation network, followed by determining the optimal action a* through a dynamic programming approach, formalized in **Eq. 18:**

$$a^* = \arg\max_a Q(s_{t+1}, a; w) \quad (18)$$

Subsequently, the target network computes the target value yt using this optimal action a*, according to **Eq. 20**. The TD framework then uses this yt value to determine the loss function required for updating the evaluation network.

$$y_t = r_t + \gamma Q(s_{t+1}, a^*; w^-) \quad (19)$$

The DDQN loss formulation is detailed in **Eq. 20:**

$$L(w) = E_{(s,a,r,s_{t+1})\sim D}[r_t + \gamma Q(s_{t+1}, a^*; w^-) - Q(s_t, a_t; w)^2] \quad (20)$$

Where:

· D represents the experience replay buffer.

· γ denotes the discount factor.

· $E_{(s,a,r,s_{t+1})\sim D}$ defines the expected loss across the sampled data batches.

Network parameter optimization is handled via the Root Mean Square Propagation (RMSProp) algorithm. In contrast to Stochastic Gradient Descent with Momentum (SGDM), RMSProp dynamically scales the learning rate. This provides superior handling of non-stationary objectives, mitigating problematic oscillations and fostering steady model convergence.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

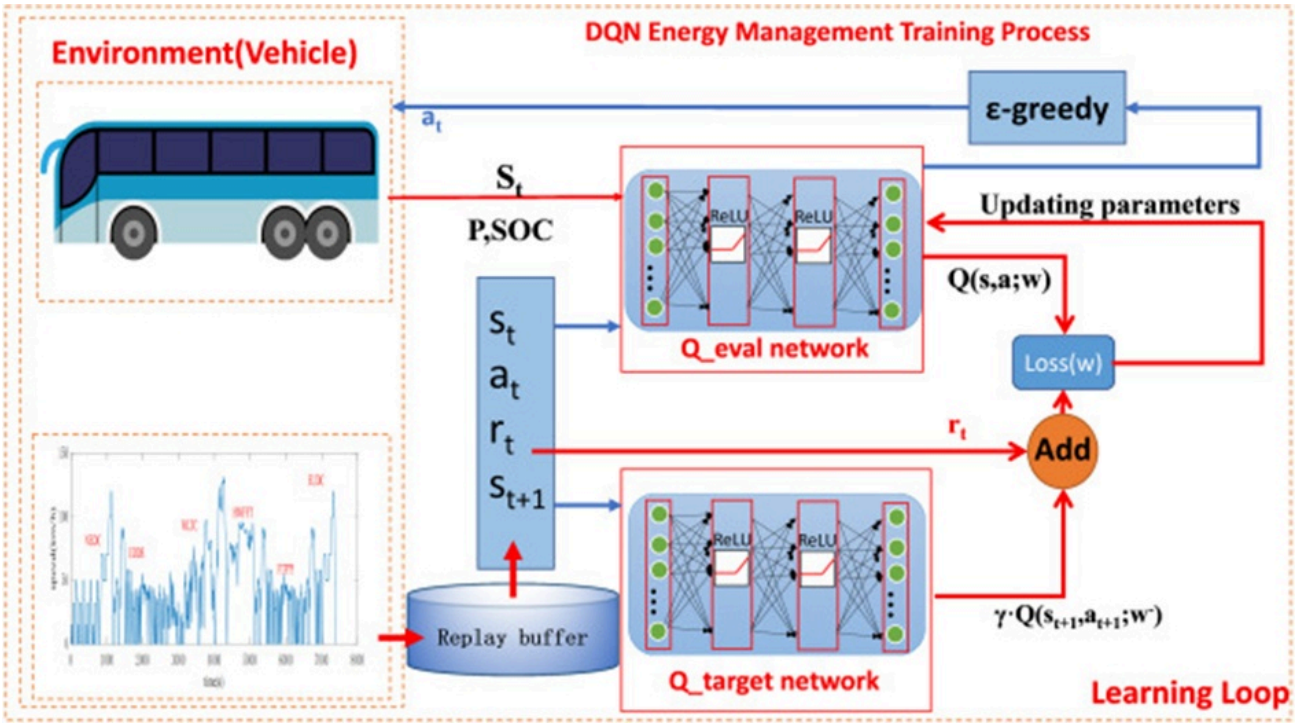


Figure 21. Double DQN training process.

MPC-DDQN

In this study, future vehicle velocity profiles are generated via an LSTM velocity prediction model to compute the required power across the predictive horizon. The agent's state vector inputs comprise the immediate power demand, the projected future power demands, and the current State of Charge (SOC). Through training, the agent learns to regulate the next-step power output of the range extender based on these state variables, operating within an output range of [0 kW, 285 kW].

Tuning the reward function parameters guides the control strategy toward superior decision-making. To optimize training efficiency and save time, the training of the velocity predictor is decoupled from the optimization of the energy management agent. The output from the prediction model is fed directly into the Double DQN environment, with the operational logic for the MPC-Double DQN energy management system detailed in **Table 4:**

Within deep reinforcement learning frameworks, designing an appropriate reward function is critical, as it directly governs both learning efficiency and model convergence. To maintain the SOC close to a target of 0.3 while maximizing fuel economy, the reward function utilized here incorporates two primary objectives, as presented in **Eq. 21:**

$$r_t = \beta(SOC(t) - SOC_0)^2 + \gamma(\alpha_{equal} \cdot (SOC(t) - SOC_0) + fe_{ins}) \quad (21)$$

Where:

- rt represents the current step reward.
- SOC(t) is the instantaneous state of charge.
- SOC0 is the starting SOC level.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

· , and γ denote scaling constants. Following extensive simulation trade-offs assessing control efficacy, \beta is fixed at −160 to guarantee a consistently upward trend for the reward function over training iterations. Furthermore, the parameter \gamma is set to 1.5.

· α_{equal} signifies the equivalent fuel-to-electricity conversion metric.

· fe_{ins} corresponds to the real-time fuel consumption rate.

Table 4. MPC-DDQN energy management pseudo-code.

Step	Revised MPC-DDQN procedure
1	Define the key hyperparameters, including the ε-greedy exploration factor (ϵ), replay memory capacity (D), and discount coefficient (γ).
2	Initialize the online network weights w and duplicate them to establish the target network parameters w^{-}.
3	Specify the maximum number of training episodes, denoted as Emax.
4	Execute the training loop for i = 1 to Emax episodes.
5	Randomly initialize the state trajectory S, consisting of the forecasted power demand sequence and battery state of charge (SOC).
6	For each time step j = 1:T, choose an action a with probability ϵ; otherwise, with probability (1 − ϵ), select the optimal action a* corresponding to the highest Q-value. The target network then computes the target value: $y_t = r_t + \gamma Q\text{target}(s_{t+1}, a^*; w^{-})$.
7	Determine the temporal-difference (TD) error using $\delta_t = y_t - Q(s_t, a_t; w)$.
8	Store the generated transition in the replay memory according to the prioritized experience replay strategy and its associated sampling probability.
9	Optimize the current network by training on minibatches sampled from the replay buffer.
10	After every predefined update interval C, synchronize the target network by copying the online network parameters w to Qtarget(S, a; w^{-}).
11	Complete the current time-step iteration.
12	Complete the current training episode.
13	Apply the first control action from the optimal output sequence as the range extender power command for the subsequent time instant, enabling continuous control through a receding-horizon (rolling-window) mechanism.

Comparison of Simulation Results

This investigation evaluates an LSTM-MPC-DDQN energy management approach configured with a 30 s historical window and a 5 s predictive horizon, benchmarking its performance against five alternative control schemes. All evaluated strategies are tested under the China Heavy-duty Commercial Vehicle Test Cycle (CHTC) profile, starting with an initial SOC setting of 0.3.

Iterative training process

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

Deep reinforcement learning seeks to maximize cumulative rewards, meaning DDQN training quality is judged by whether the estimated Q-values rise and settle reliably near an optimal upper bound. Consequently, precise hyperparameter tuning is vital for the performance of the reinforcement learning-based energy management system.

The baseline DDQN setup uses a three-layer backpropagation (BP) network with 84 hidden-layer neurons. Conversely, the predictive MPC-DDQN variant leverages a four-layer network containing 124 hidden neurons. Across both architectures, parameter optimization relies exclusively on the RMSProp algorithm, applying a learning rate of 0.0001 alongside a discretization step of 0.1 s. **Figure 22** shows the training trajectory for the standalone DDQN controller:

As illustrated, rewards climb steeply during early episodes. However, past the 5th iteration, further training fails to improve the reward metrics, which instead stall and oscillate around a fixed baseline. At this stage, parameter updates stagnate, and reward yields occasionally degrade. This limitation stems from both the shallow network architecture and the restricted input space of the basic DDQN, which minimizes the observable environmental state and limits the information available for feature extraction.

Incorporating velocity forecasting via the MPC-DDQN framework circumvents these issues. Here, the velocity predictor interprets past driving records to extract motion features, passing these velocity trends to the controller to establish future power requirements. The DDQN then modulates the range extender's output based on this anticipatory data. This strategy enriches the state inputs for the DDQN. It empowers the reinforcement learning model to coordinate actions over a broader time domain, maximizing reward outcomes and boosting vehicle fuel economy. The comparative training histories of the LSTM-MPC-DDQN and SVM-MPC-DDQN architectures are provided in **Figure 23.**

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

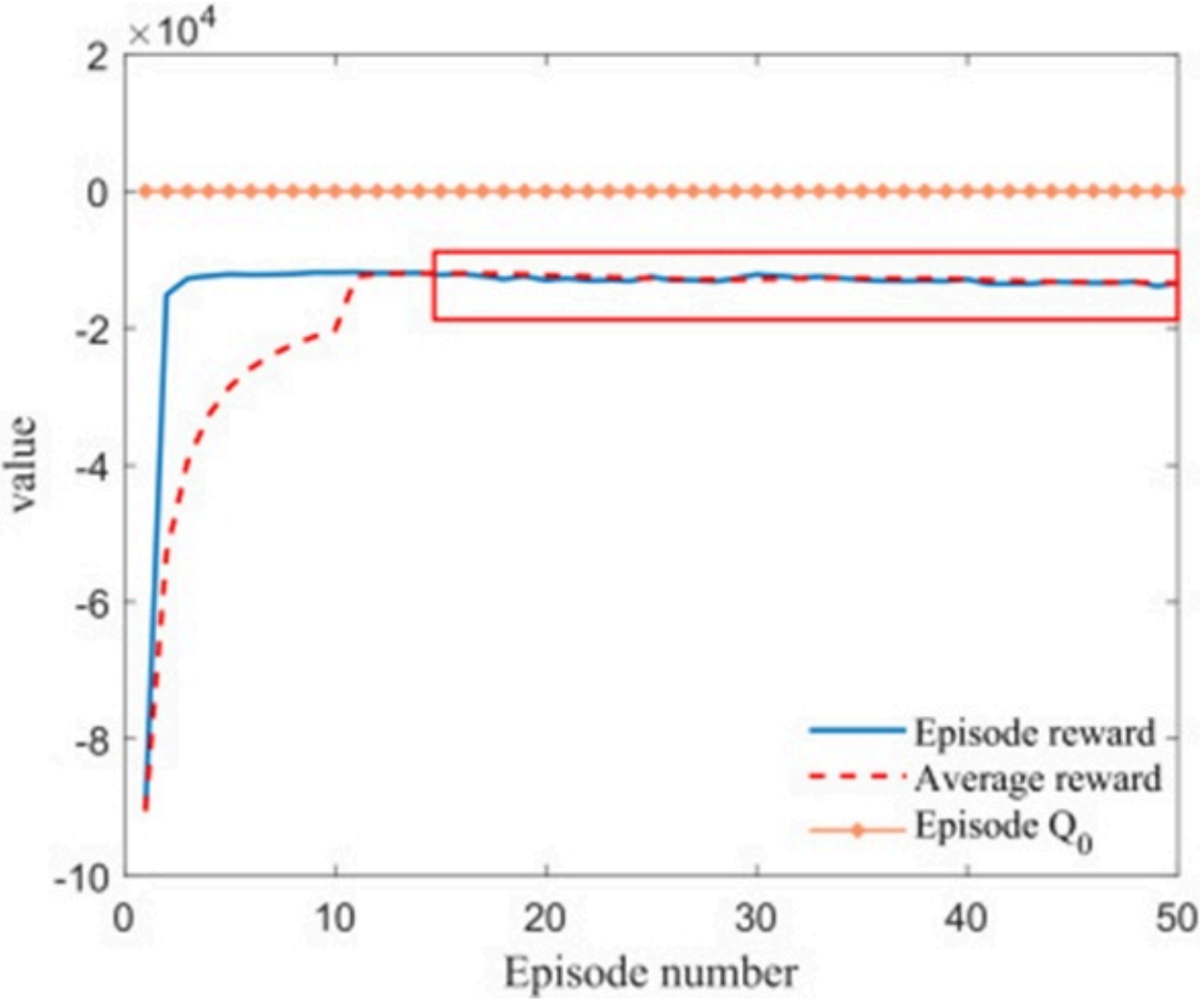


Figure 22. Double DQN training process diagram.

Upon completing 300 training cycles for both agents, the terminal iteration yields an average reward of −5159.5 for the LSTM-MPC-DDQN framework, compared to −9836.8 for the SVR-MPC-DDQN strategy. Because the SVR velocity predictor underperforms relative to the LSTM model, the agent discounts the future predictive horizon during training. This forces the neural network to downplay upcoming velocity changes, resulting in diminished final reward performance.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

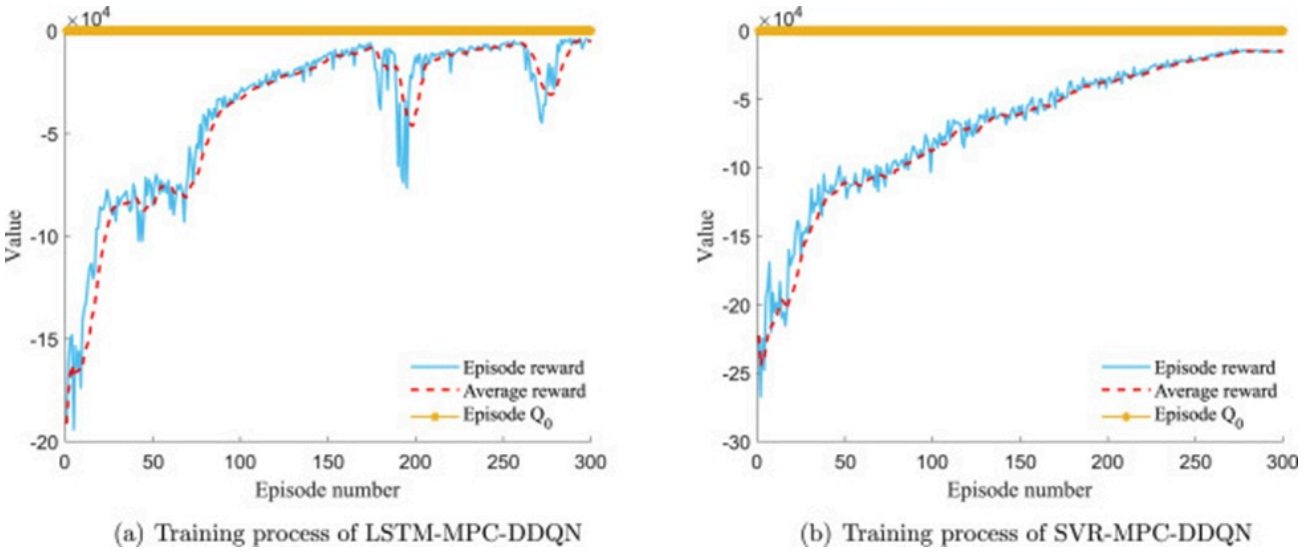


Figure 23. Training process of speed prediction energy management for different vehicle speed prediction methods.

Simulation experiment results

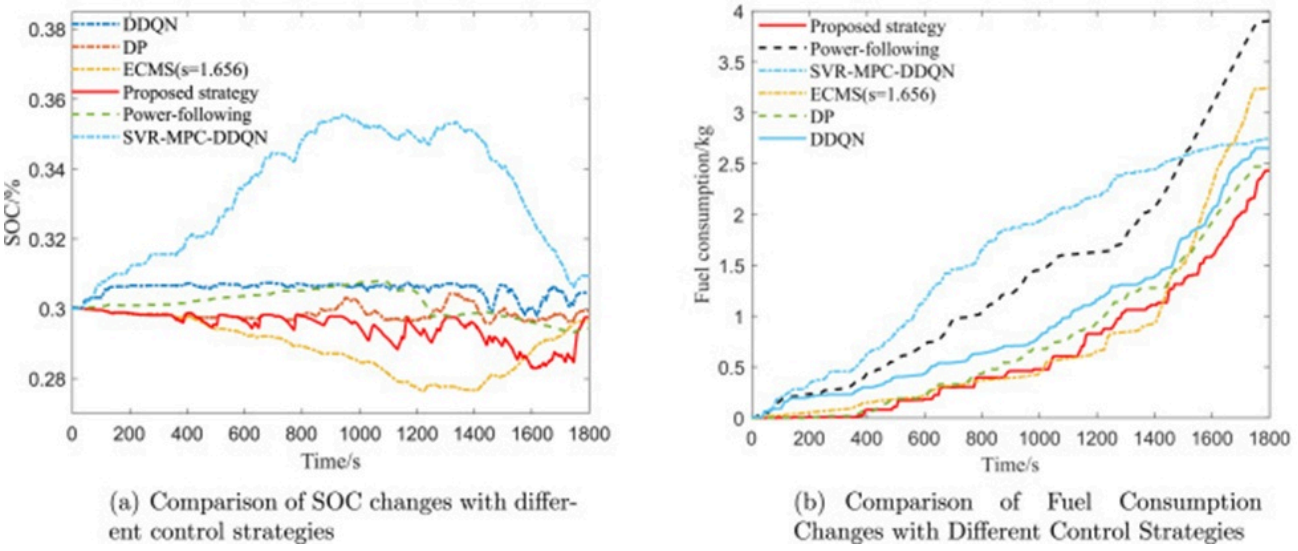
The profiles tracking SOC trajectories and fuel depletion metrics across the various control frameworks are illustrated in **Figure 22**. As shown in Figure (a), both the SVR-MPC-DDQN and ECMS schemes experience substantial SOC fluctuations throughout the simulation. Yet, every tested controller manages to drive the terminal SOC back to approximately 0.3.

The performance metrics for these control methodologies are summarized in **Table 5**, which indicates that while the dynamic programming (DP) model yields the absolute lowest equivalent fuel consumption, the LSTM-MPC-DDQN framework provides the closest approximation to this DP baseline. As a global optimization tool, the dynamic programming algorithm establishes the theoretical performance limit and serves as the primary benchmark for validation. To maintain a fair evaluation, the objective function defined for the DP model mirrors that of our proposed control methodology.

The data indicates that predictive, reinforcement-learning-driven energy management substantially improves fuel efficiency relative to traditional power-following and ECMS approaches. However, because the SVR velocity model provides less accurate predictions than the LSTM alternative, the SVR-MPC-DDQN strategy fails to outperform either the standard DDQN or the superior LSTM-MPC-DDQN configuration **(Figure 24).**

Figure 25 shows the output power of the range extender under various control strategies. Notably, during low-demand periods, the dynamic programming, LSTM-MPC-DDQN, and ECMS methods minimize range extender usage. This occurs because DP and ECMS exploit global optimization. At the same time, the LSTM-MPC-DDQN framework relies on anticipatory velocity forecasting to prevent the range extender from operating within low-efficiency regions by reasoning across the look-ahead horizon. Conversely, the ECMS strategy exhibits severe power output volatility; this aggressive, high-frequency switching throughout its operation represents a primary factor behind its elevated fuel consumption.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press



(a) Comparison of SOC changes with different control strategies

(b) Comparison of Fuel Consumption Changes with Different Control Strategies

Figure 24. Variation of SOC and fuel consumption for different control strategies.

Table 5. Simulation results of different control strategies.

Control strategy	Equivalent fuel consumption/kg	Fuel consumption/kg	SOC final value/%
DP	2.4885	2.4699	0.2993
DDQN	2.5287	2.6495	0.3046
ECMS	3.3372	3.2423	0.2964
LSTM-MPC-DDQN	2.4933	2.4299	0.2976
Power-following	3.2353	3.085	0.2943
SVR-MPC-DDQN	2.9925	2.742	0.3095

The LSTM-MPC-DDQN scheme demonstrates stable power transitions, keeping the bulk of its generation bounded below 200 kW—a high-efficiency operating envelope for the range extender that directly minimizes overall fuel use. For the dynamic programming baseline, power delivery centers primarily around 80 kW, which aligns with the peak efficiency zones shown in the optimal fuel curves of **Figure 4,** validating its role as the ideal benchmark.

In contrast, the power-following approach mirrors the immediate traction demands without any temporal planning, leading to higher fuel penalties. Meanwhile, the SVR-MPC-DDQN alternative suffers from constrained predictive accuracy, which prevents the controller from executing optimal management decisions.

Regarding cumulative energy distribution, the total mechanical work delivered by the range extender is calculated by integrating instantaneous power over the entire cycle, as visualized in **Figure 26**. When analyzed alongside fuel consumption metrics, these results confirm that both dynamic programming and our proposed predictive framework achieve superior operating efficiency compared to competing control strategies.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

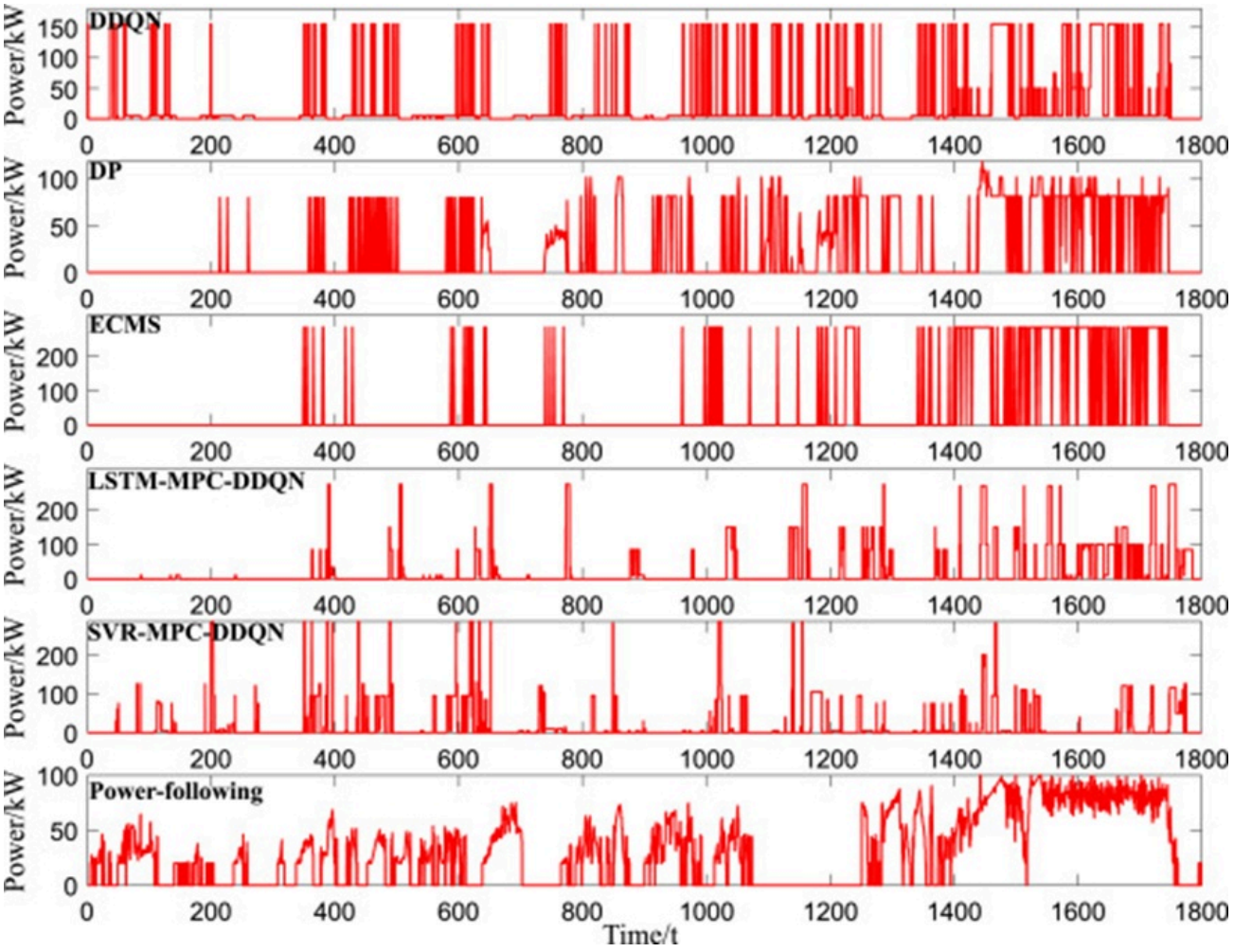


Figure 25. Variation of output power with different control strategies.

Figure 27 illustrates where the primary operating points for the engine and generator fall across different control strategies, with the dashed line highlighting the range extender's ideal operating curve. Because the engine and generator collectively establish this optimal curve, the control scheme must rely on it to determine the ideal torque and rotational speed based on power requirements. Observing the distribution, it is evident that the operating points for DP, DDQN, and LSTM-MPC-DDQN are largely clustered around this optimal trajectory. Conversely, the ECMS and power-following strategies frequently drift into low-efficiency areas due to shifting operational conditions, ultimately driving up their fuel consumption. Because DP operates as a global optimizer, it consistently selects the most efficient operating points from a comprehensive perspective, thereby preventing the range extender from falling into inefficient zones. Similarly, the LSTM-MPC-DDQN framework utilizes speed forecasting to execute short-term planning, successfully bypassing these low-efficiency regions. In contrast, the SVR-MPC-DDQN strategy struggles with inaccurate speed predictions, causing it to operate inefficiently. This underscores how crucially speed prediction accuracy dictates the overall effectiveness of reinforcement learning strategies reliant on such forecasts.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

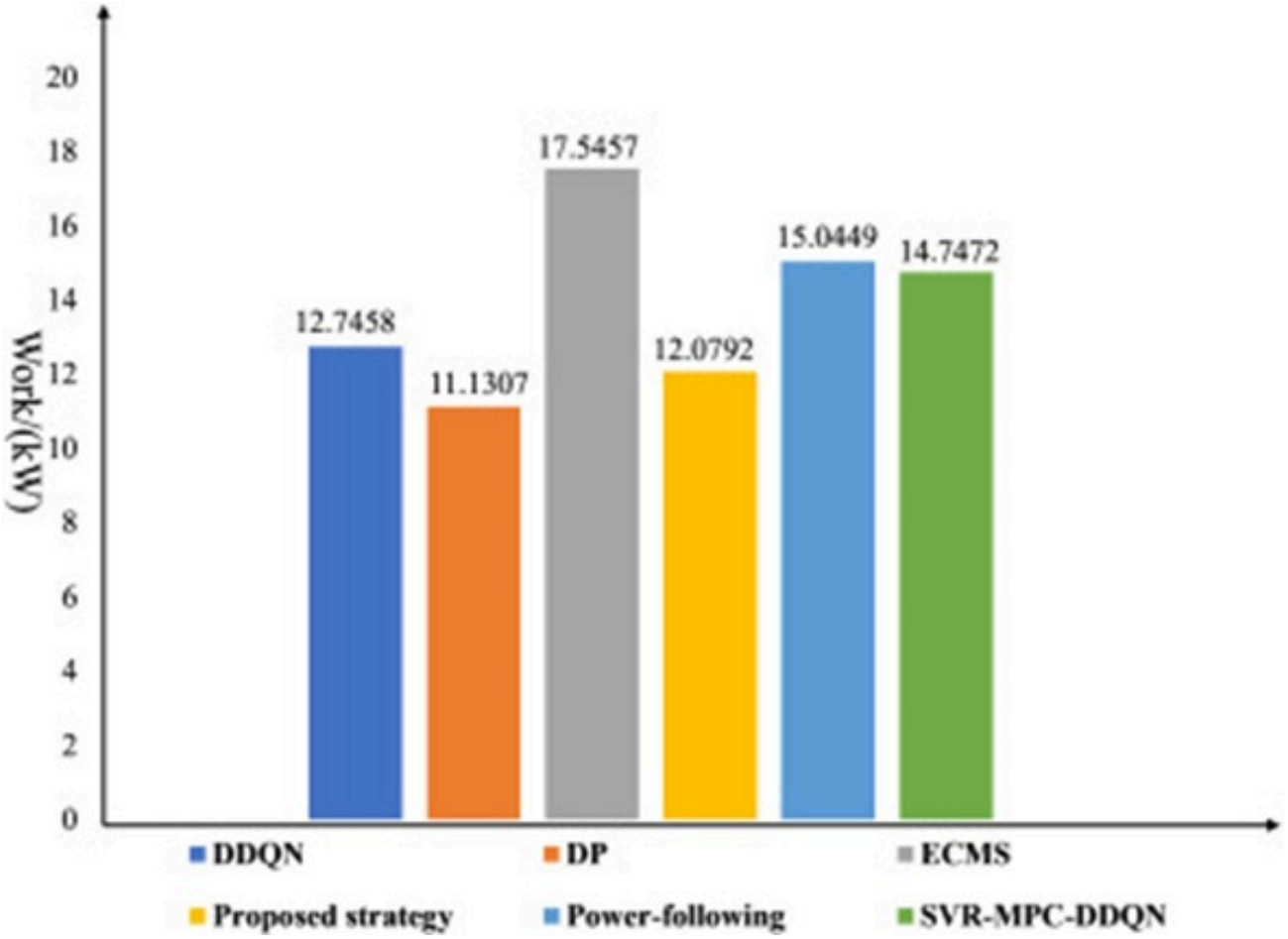
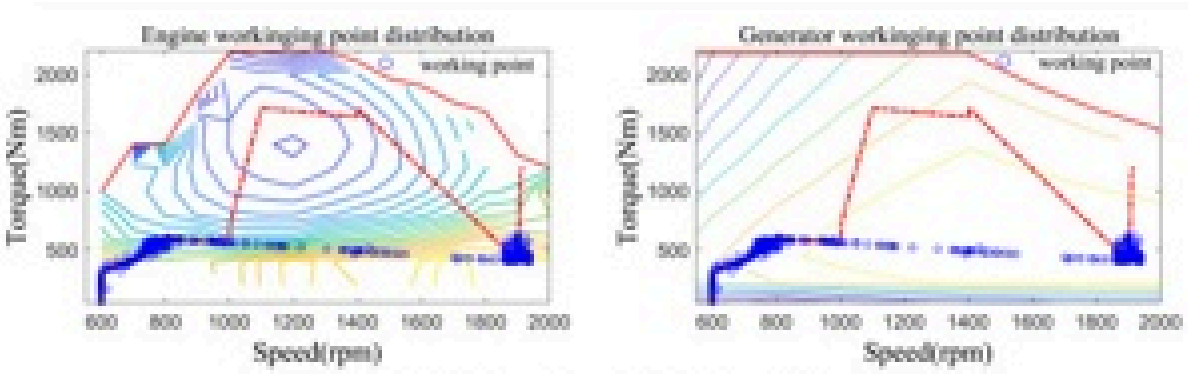


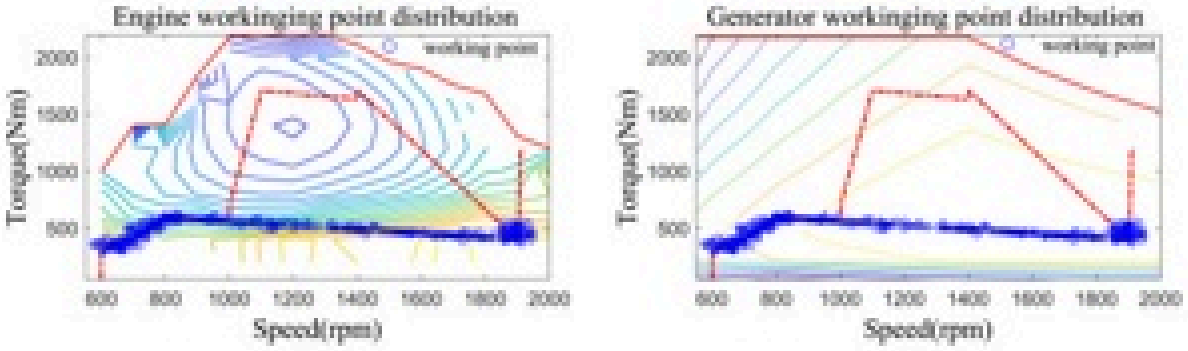
Figure 26. The cumulative output power of the range extender under each control strategy.

To confirm the viability of the introduced control method, simulations under WTVC driving conditions were conducted to benchmark it against competing strategies. As depicted in **Figure 28**, the State of Charge (SOC) for our advanced strategy stays highly stable. Meanwhile, the power-following approach shows an increasing SOC, but this comes with a penalty of higher fuel usage. This finding is further supported by the metrics in **Table 6,** which show that the power-following framework yields substantially higher equivalent fuel consumption than the alternatives.

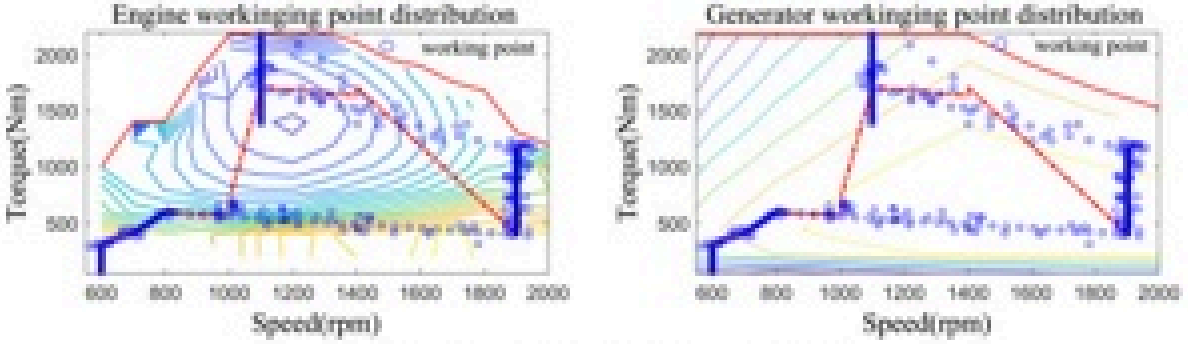
© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press



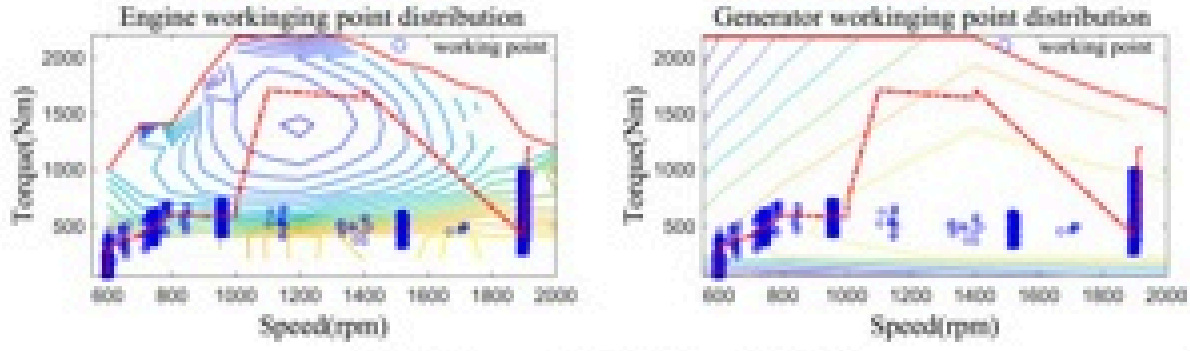
(a) Working points distribution of DP



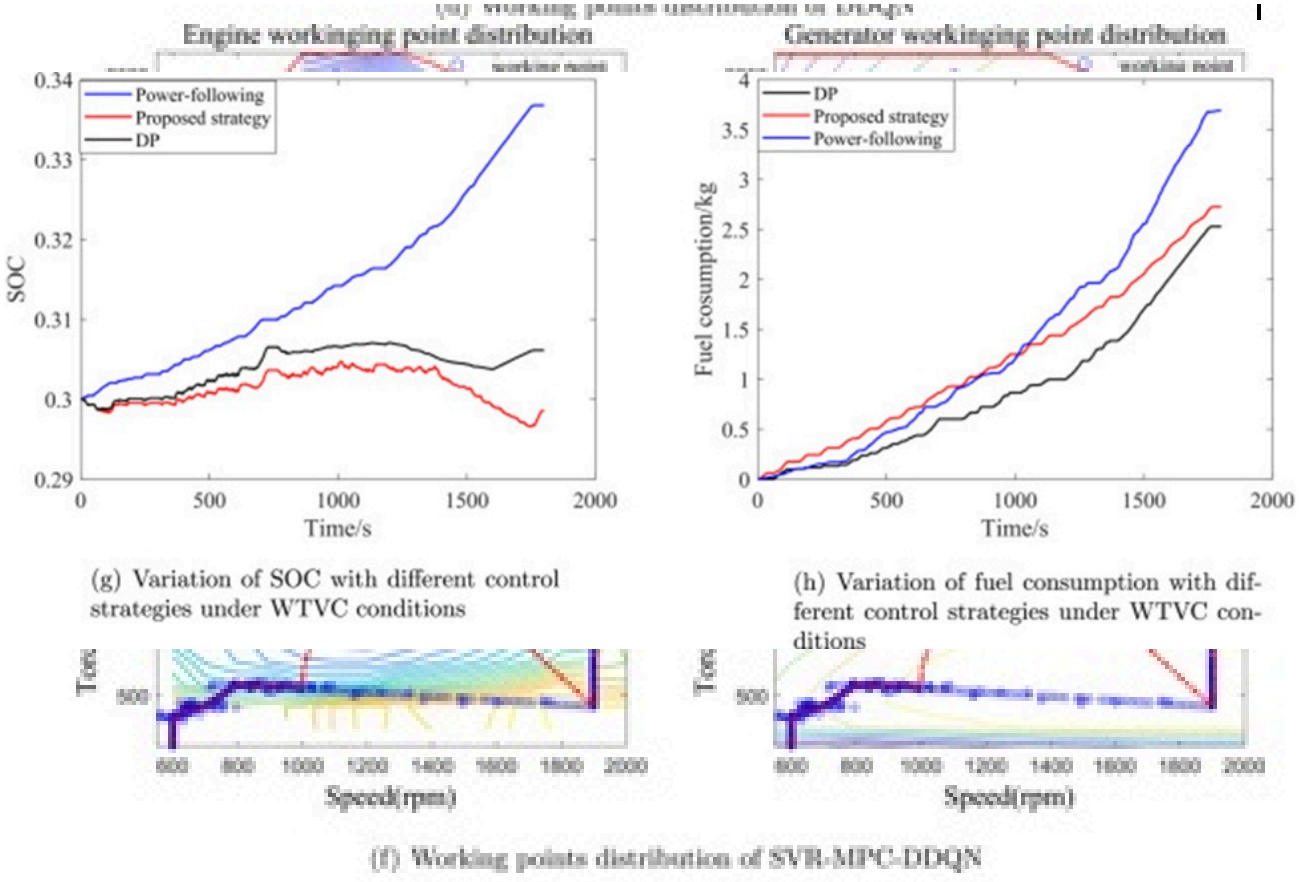
(b) Working points distribution of Power-following



(c) Working points distribution of ECMS



© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press



(g) Variation of SOC with different control strategies under WTVC conditions

(h) Variation of fuel consumption with different control strategies under WTVC conditions

(f) Working points distribution of SVR-MPC-DDQN

LSTM–MPC–DDQN method	2.7629	2.7260	0.2986
Power-following scheme	2.9684	3.6902	0.3268

Experimental Verification of HIL Simulation

To evaluate the real-time capabilities and precision of the LSTM-MPC-DDQN framework, we constructed a Hardware-in-the-loop (HIL) testing environment. The structural layout of this HIL simulation is presented in **Figure 29**, utilizing the CHTC test profile. The primary components of the HIL setup include a SCALEXIO real-time hardware simulator, a MicroAutoBox controller, and a primary host computer. The host computer oversees the entire simulation while recording output data; the controller acts as the electronic control unit, executing decisions based on the designated control algorithms; and the SCALEXIO system emulates the full vehicle dynamics, generating a realistic environment for the Hybrid Control Unit (HCU). Network cables link both the AUTOBOX and SCALEXIO units to the host computer. A harness interface connects SCALEXIO to the expansion board, while CAN cables link both SCALEXIO and AUTOBOX to this same board. Data transmission for the AUTOBOX is facilitated via the CAN bus.

Figure 30 contrasts the HIL experimental outcomes of our proposed LSTM-MPC-DDQN algorithm against standard Simulink software simulations. The data indicates that the HIL findings closely mirror the Simulink projections; specifically, the terminal SOC error is merely 0.03%, and the discrepancy in fuel consumption is just 0.026%. Furthermore, the speed output from the driver module tracks the target velocity accurately, keeping experimental deviations well within acceptable limits. Consequently, the deep reinforcement learning (RL) energy management approach—powered by GA-LSTM speed forecasting—demonstrates strong real-time execution, successfully satisfying practical driving demands.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

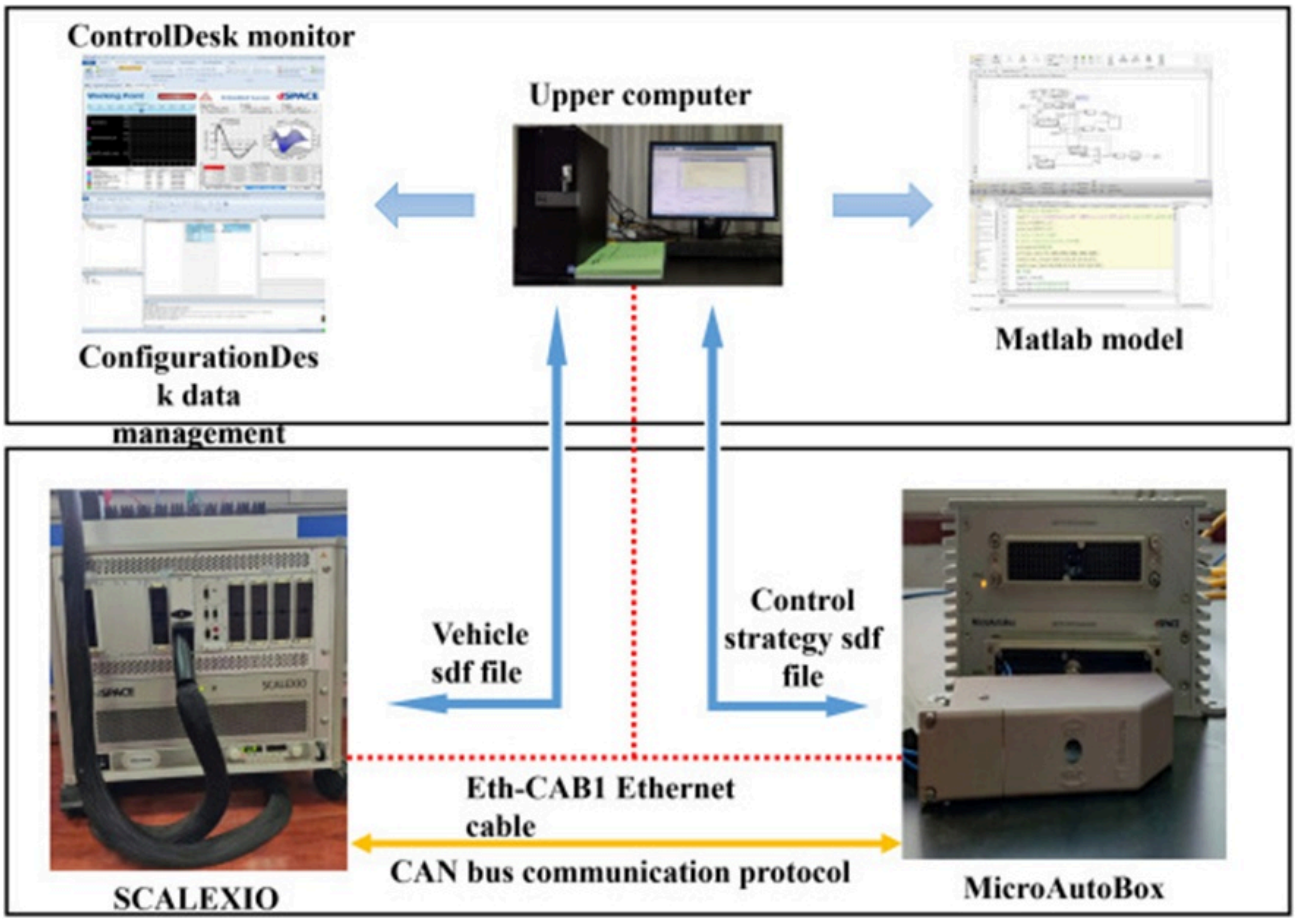
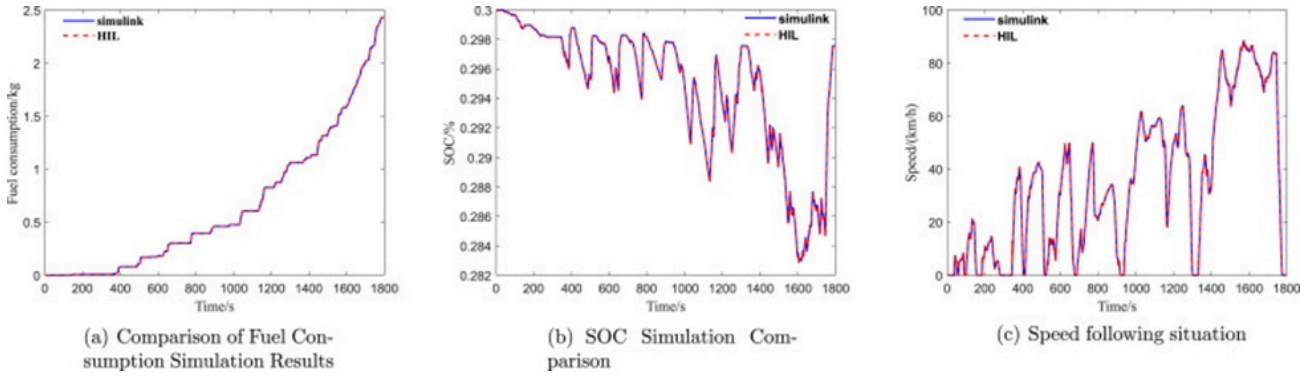


Figure 29. Hardware-in-the-loop simulation schematic.



(a) Comparison of Fuel Consumption Simulation Results (b) SOC Simulation Comparison (c) Speed following situation

Figure 30. Comparison of hardware-in-the-loop Simulink and Simulink simulation results.

Conclusion

This research introduces a reinforcement learning (RL) energy management framework relying on GA-LSTM speed forecasting, focusing specifically on a four-wheel power-driven extended-range bus. Initially, we established an LSTM-based velocity prediction model and applied a genetic algorithm to extract optimal hyperparameters, thereby maximizing the neural network's predictive precision. When using the CHTC profile for validation, the GA-LSTM model demonstrated vastly superior accuracy across multiple predictive horizons, coupled with faster computation times compared with SVR-

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

based forecasting. Once the speed forecast is generated, the controller computes the required power across the upcoming horizon. Using the current power demand, the projected power demand, and the battery's SOC as state inputs, the RL agent determines the necessary power output from the range extender. The MPC framework facilitates this through continuous rolling optimization.

For the speed-predictive RL energy management detailed here, the training involved 300 iterations. Simulation outputs reveal that our LSTM-MPC-DDQN strategy achieves a profound upgrade in fuel efficiency over standard power-following and ECMS methods. Its fuel economy closely rivals that of the global DP baseline, marking a 1.34% enhancement over the basic DDQN approach. Furthermore, the SOC maintained by the LSTM-MPC-DDQN system is noticeably more consistent than those produced by ECMS or SVR-MPC-DDQN. The findings also highlight that deep RL energy management is heavily dependent on the exactness of the speed predictions. Any loss in prediction accuracy compromises the power demand calculations for the future horizon, ultimately degrading the agent's forward-planning capabilities. Lastly, rigorous hardware-in-the-loop testing successfully validated the robust real-time applicability of this proposed strategy.

Because reinforcement learning frameworks naturally accommodate multivariate state inputs, future studies could integrate traffic density into the control logic. This could be achieved by using onboard cameras to monitor leading vehicles, applying object detection algorithms to quantify surrounding traffic density, and feeding this data directly to the agent as an additional state variable.

Acknowledgements

None

Conflict of interest

None

Financial support

This work was supported by the National Natural Science Foundation of China (No. 52175236), Research and development and demonstration application of heavy-duty diesel vehicle exhaust emission testing technology, China (No. 24-8-cspz-18-nsh) and Qingdao Civi Science and Technology Plan, China (No. 19-6-1-88-nsh).

Ethics statement

None

Received: 25 Jun 2022 Revised: 01 Sep 2022 Accepted: 16 Sep 2022
Published online: 15 November 2023

Rights and permissions

Open Access The author(s) retain copyright. This article is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. It may be shared and adapted for non-commercial purposes with appropriate attribution, an indication of changes, and

distribution of adaptations under the same license. Third-party material may be subject to separate terms identified in its credit line. View the license at https://creativecommons.org/licenses/by-nc-sa/4.0/.

References

1. Zhao H, Mu L, Li Y, Qiu J, Sun C, Liu X. Unregulated emissions from natural gas taxi based on IVE model. Atmosphere. 2021;12(4):478. https://doi.org/10.3390/atmos12040478.

2. Fayyazi M, Abdoos M, Phan D, Golafrouz M, Jalili M, Jazar RN, et al. Real-time self-adaptive Q-learning controller for energy management of conventional autonomous vehicles. Expert Syst Appl. 2023;222:119770. https://doi.org/10.1016/j.eswa.2023.119770.

3. Zhu W. Optimization strategies for real-time energy management of electric vehicles based on LSTM network learning. Energy Rep. 2022;8:1009-19. https://doi.org/10.1016/j.egyr.2021.12.045.

4. Yang D, Wang L, Yu K, Liang J. A reinforcement learning-based energy management strategy for fuel cell hybrid vehicle considering real-time velocity prediction. Energy Convers Manage. 2022;274:116453. https://doi.org/10.1016/j.enconman.2022.116453.

5. Vignesh R, Ashok B. Intelligent energy management through neuro-fuzzy based adaptive ECMS approach for an optimal battery utilization in plugin parallel hybrid electric vehicle. Energy Convers Manage. 2023;280:116792. https://doi.org/10.1016/j.enconman.2023.116792.

6. Shen P, Zhao Z, Zhan X, Li J, Guo Q. Optimal energy management strategy for a plug-in hybrid electric commercial vehicle based on velocity prediction. Energy. 2018;155:838-52. https://doi.org/10.1016/j.energy.2018.04.182.

7. Wang W, Guo X, Yang C, Zhang Y, Zhao Y, Huang D, et al. A multi-objective optimization energy management strategy for power split HEV based on velocity prediction. Energy. 2022;238:121714. https://doi.org/10.1016/j.energy.2021.121714.

8. Xing J, Chu L, Hou Z, Sun W, Zhang Y. Energy management strategy based on a novel speed prediction method. Sensors. 2021;21(24):8273. https://doi.org/10.3390/s21248273.

9. Chen R, Yang C, Han L, Wang W, Ma Y, Xiang C. Power reserve predictive control strategy for hybrid electric vehicle using recognition-based long short-term memory network. J Power Sources. 2022;520:230865. https://doi.org/10.1016/j.jpowsour.2021.230865.

10. Han J, Shu H, Tang X, Lin X, Liu C, Hu X. Predictive energy management for plug-in hybrid electric vehicles considering electric motor thermal dynamics. Energy Convers Manage. 2022;251:115022. https://doi.org/10.1016/j.enconman.2021.115022.

11. Lin X, Wu J, Wei Y. An ensemble learning velocity prediction-based energy management strategy for a plug-in hybrid electric vehicle considering driving pattern adaptive reference SOC. Energy. 2021;234:121308. https://doi.org/10.1016/j.energy.2021.121308.

12. Ritter A, Widmer F, Duhr P, Onder CH. Long-term stochastic model predictive control for the energy management of hybrid electric vehicles using pontryagin's minimum principle and scenario-based optimization. Appl Energy. 2022;322:119192. https://doi.org/10.1016/j.apenergy.2022.119192.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

13. Li M, He H, Feng L, Chen Y, Yan M. Hierarchical predictive energy management of hybrid electric buses based on driver information. J Clean Prod. 2020;269:122374. https://doi.org/10.1016/j.jclepro.2020.122374.

14. Yu X, Lin C, Tian Y, Zhao M, Liu H, Xie P, et al. Real-time and hierarchical energy management-control framework for electric vehicles with dual-motor powertrain system. Energy. 2023;272:127112. https://doi.org/10.1016/j.energy.2023.127112.

15. Chen Z, Liu Y, Zhang Y, Lei Z, Chen Z, Li G. A neural network-based ECMS for optimized energy management of plug-in hybrid electric vehicles. Energy. 2022;243:122727. https://doi.org/10.1016/j.energy.2021.122727.

16. Wei C, Chen Y, Li X, Lin X. Integrating intelligent driving pattern recognition with adaptive energy management strategy for extender range electric logistics vehicle. Energy. 2022;247:123478. https://doi.org/10.1016/j.energy.2022.123478.

17. Chen Z, Gu H, Shen S, Shen J. Energy management strategy for power-split plug-in hybrid electric vehicle based on MPC and double Q-learning. Energy. 2022;245:123182. https://doi.org/10.1016/j.energy.2022.123182.

18. Lin X, Zhang J, Su L. A trip distance adaptive real-time optimal energy management strategy for a plug-in hybrid vehicle integrated driving condition prediction. J Energy Storage. 2022;52:105055. https://doi.org/10.1016/j.est.2022.105055.

19. Zhao Z, Xun J, Wan X, Yu R. MPC based hybrid electric vehicles energy management strategy. IFAC-PapersOnLine. 2021;54(10):370-5. https://doi.org/10.1016/j.ifacol.2021.10.188.

20. Yang N, Ruan S, Han L, Liu H, Guo L, Xiang C. Reinforcement learning-based real-time intelligent energy management for hybrid electric vehicles in a model predictive control framework. Energy. 2023;270:126971. https://doi.org/10.1016/j.energy.2023.126971.

21. Lu L, Zhao H, Liu X, Sun C, Zhang X, Yang H. MPC-ECMS energy management of extended-range vehicles based on LSTM multi-signal speed prediction. Electronics. 2023;12(12):2642. https://doi.org/10.3390/electronics12122642.

22. Du R, Hu X, Xie S, Hu L, Zhang Z, Lin X. Battery aging- and temperature-aware predictive energy management for hybrid electric vehicles. J Power Sources. 2020;473:228568. https://doi.org/10.1016/j.jpowsour.2020.228568.

23. Shangguan J, Guo H, Yue M. Robust energy management of plug-in hybrid electric bus considering the uncertainties of driving cycles and vehicle mass. Energy. 2020;203:117836. https://doi.org/10.1016/j.energy.2020.117836.

24. Jinquan G, Hongwen H, Jiankun P, Nana Z. A novel MPC-based adaptive energy management strategy in plug-in hybrid electric vehicles. Energy. 2019;175:378-92. https://doi.org/10.1016/j.energy.2019.03.083.

25. Li J, Wu X, Xu M, Liu Y. A real-time optimization energy management of range extended electric vehicles for battery lifetime and energy consumption. J Power Sources. 2021;498:229939. https://doi.org/10.1016/j.jpowsour.2021.229939.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press

26. Xue J, Jiao X, Yu D, Zhang Y. Predictive hierarchical eco-driving control involving speed planning and energy management for connected plug-in hybrid electric vehicles. Energy. 2023;129058. https://doi.org/10.1016/j.energy.2023.129058.

27. Min D, Song Z, Chen H, Wang T, Zhang T. Genetic algorithm optimized neural network based fuel cell hybrid electric vehicle energy management strategy under start-stop condition. Appl Energy. 2022;306:118036. https://doi.org/10.1016/j.apenergy.2021.118036.

28. Wang Y, Zhang Y, Zhang C, Zhou J, Hu D, Yi F, et al. Genetic algorithm-based fuzzy optimization of energy management strategy for fuel cell vehicles considering driving cycles recognition. Energy. 2023;263:126112. https://doi.org/10.1016/j.energy.2022.126112.

29. Yang C, Lu Z, Wang W, Wang M, Zhao J. An efficient intelligent energy management strategy based on deep reinforcement learning for hybrid electric flying car. Energy. 2023;128118. https://doi.org/10.1016/j.energy.2023.128118.

30. Du G, Zou Y, Zhang X, Guo L, Guo N. Energy management for a hybrid electric vehicle based on prioritized deep reinforcement learning framework. Energy. 2022;241:122523. https://doi.org/10.1016/j.energy.2021.122523.

31. Wang Z, He H, Peng J, Chen W, Wu C, Fan Y, et al. A comparative study of deep reinforcement learning based energy management strategy for hybrid electric vehicle. Energy Convers Manage. 2023;293:117442. https://doi.org/10.1016/j.enconman.2023.117442.

© 2023 The Author(s). | Published by Institute of Smart Energy Systems Press