Reinforcement learning approach for optimal control of multiple electric locomotives in a heavy-haul freight train: A Double-Switch-Q-network architecture

Huiyue Tang[a,c], Yuan Wang[b,c,*], Xiang Liu[c], Xiaoyun Feng[a]

[a]*School of Electrical Engineering, Southwest Jiaotong University, Chengdu, China*
[b]*School of System Design and Intelligent Manufacturing, Southern University of Science and Technology, Shenzhen, China*
[c]*Department of Civil and Environmental Engineering, Rutgers, The State University of New Jersey, New Jersey, U.S.A.*

Abstract

Electric locomotives provide high tractive power for fast acceleration of heavy-haul freight trains, and significantly reduce the energy consumption with regenerative braking. This paper proposes a reinforcement learning (RL) approach for the optimal control of multiple electric locomotives in a heavy-haul freight train, without using the prior knowledge of train dynamics and the pre-designed velocity profile. The optimization takes the velocity, energy consumption and coupler force as objectives, considering the constraints on locomotive notches and their change rates, speed restrictions, traction and regenerative braking. Besides, since the problem in this paper has continuous state space and large action space, and the adjacent actions' influences on states share similarities, we propose a Double-Switch Q-network (DSQ-network) architecture to achieve fast approximation of the action-value function, which enhances the parameter sharing of states and actions, and denoises the action-value function. In the numerical experiments, we test DSQ-network in 28 cases using the data of China Railways HXD3B electric locomotive. The results indicate that compared with table-lookup Q-learning, DSQ-network converges much faster and uses less storage space in the optimal control of electric locomotives. Besides, we analyze 1)the influences of ramps and speed restrictions on the optimal policy, and 2)the inter-dependent and inter-conditioned relationships between multiple optimization objectives. Finally, the factors that influence the convergence rate and solution accuracy of DSQ-network are discussed based on the visualization of the high-dimensional value functions.

Keywords: reinforcement learning, Double-Switch Q-network, optimal control, electric locomotive, heavy-haul freight train

1. Introduction

Heavy-haul freight trains with electric locomotives are of significant importance for long-distance and large-capacity freight transportation in China [1]. The high-power electric locomotives in China, such as HXD3B, can 1) provide large tractive power for fast acceleration of heavy-haul freight trains, for example, the tractive power of a HXD3B locomotive is 9600kw, 2) use regenerative braking to transform the kinetic energy into electric energy, and return it to the traction power supply system, and 3) reduce the coupler forces through the coordination of traction and braking forces from different

*Corresponding author
Email address: wang.skoud@gmail.com (Yuan Wang)

locomotives. The control strategies of electric locomotives have significant influences on the safety and efficiency of heavy-haul freight trains.

The optimal control of heavy-haul freight trains requires a tradeoff between multiple objectives. We always expect to maximize the velocity while minimizing the energy consumption and coupler force. However, these objectives cannot be achieved simultaneously. For example, when a heavy-haul freight train is running on an upward ramp, the energy consumption increases significantly with the increase of velocity. Besides, the acceleration of wagons depends on the coupler force, and the mechanical resistance and aerodynamic drag increase with the increase of velocity. As a result, the coupler force also increases with the increase of velocity.

Moreover, it is difficult to find the optimal control strategies for electric locomotives in a heavy-haul freight train, since the in-train dynamics of heavy-haul freight trains is much more complicated than the passenger trains, in three aspects,

1) *The large number of railway vehicles in a train* [2]: The total number of railway vehicles in a heavy-haul freight train can be greater than 200, including different types of locomotives and wagons. A heavy-haul freight train can weigh more than 10000 tons, and reach several kilometers in length. Thus, a train can cover several ramps and curves, a small change in control strategies may result in drastic changes of in-train impulse. In contrast, a CRH2A high-speed train consists of eight vehicles, and weighs about 400 tons when carrying passengers. Hence, the train dynamics is much simpler.

2) *The nonlinear constraints on the traction/braking force and its change rate* [3]: Taking the HXD3B electric locomotive as an example, the control handle of the locomotive has discrete notches. At each notch, the traction/braking force changes nonlinearly with the increase of velocity. Besides, for a heavy-haul freight train, the response of mechanical system is slow. Thus, the control signals should not change rapidly considering the safety. By contrast, some high-speed trains and metro trains can run at a constant acceleration, so the traction/braking force can even be continuous.

3) *Unpredictable sensor faults and actuator faults* [4]: Since heavy-haul freight trains are mainly used for long-distance transport of coals or other minerals, generally, the rail lines are far from cities and influenced by uncontrollable environment factors, such as weather, topography, the condition of infrastructure. The sensors can be faulty, and it is not easy for the traction/braking forces output from the mechanical system to track the control signals. By contrast, the high-speed trains and metro trains are mainly used for passenger transport, and runs in or between cities. Thus, the environment is much more comfortable.

Considering the aforementioned difficulties, a method based on machine learning has been proposed for heavy-haul train operation [5]. However, the existing work is focus on the pneumatic brake control on steep descent, the control strategies of electric locomotives are not discussed in detail. In this paper, we aim to develop an efficient algorithm for an intelligent agent to learn the optimal control strategies of multiple electric locomotives in a heavy-haul freight train, which can be used without the prior knowledge of train dynamics and pre-designed velocity profiles, and satisfy the nonlinear constraints on the control inputs.

Reinforcement learning (RL) is one of the machine learning methods, which is specific in decision optimization and prediction. The agent interacts with the environment, learns the knowledge and optimizes the decisions for achieving a goal [6, 7]. RL agent stores the knowledge in different forms, such as lookup tables, and parameterized functional form. For the optimal control of electric locomotives in a heavy-haul freight train, the state space is continuous, action space is large and adjacent actions share similarities, so the neural network, one of the nonlinear functional form, will get good performance. In particular, we propose a novel architecture, named Double-Switch-Q-network (DSQ-network), for a fast approximation of the action-value function. DSQ-network is an improvement

based on the traditional Q-network [8], which encoding the action space, and is designed specifically for the optimal control of multiple electric locomotives in a heavy-haul freight train. It gets excellent performance in the simulation environment when using the data of China Railways HXD3B electric locomotives. Notably, encoding both the state space and action space is common, however, the DSQ-network architecture changes the way of using state features and action features in the parameterized functional form compared with the existing literatures. To our best knowledge, the DSQ-network architecture has never been used elsewhere.

2. Literature review

The literature review covers three aspects: 1) the train control methods and their dependency on the train's dynamic model. 2) the advantages of using RL approach to solve our problem. 3)Why DSQ-network is effective in the control of multiple electric locomotives in a heavy-haul freight train.

Energy-efficient train operation can be achieved with different methods [9, 10], such as Lagrangian analysis [11], maximum principle [12, 13, 14, 15], genetic algorithm [16, 17, 18], pseudospectral method [19, 20], dynamic programming [21], and ant colony optimization [22]. The results provide useful guidance for safe, efficient and environment-friendly train operation. However, the aforementioned literatures are focus on minimizing the energy consumption, where the coupler force between the connected railway vehicles in a train isn't taken into consideration, but for heavy-haul freight trains, the coupler force has significant influences on the safety and maintenance cost [23].

Taking into account the coupler force, the operation strategies of heavy-haul trains are improved with the evolution of the multi-particle train model. In early works, the linearized train model is used for the closed-loop control design based on Linear Quadratic Regulation (LQR) theory [2, 24, 25]. Meanwhile, the optimal scheduling methods are proposed to obtain the equilibrium point, and thereby improve the performance of closed-loop controllers [26]. Besides, based on the nonlinear regulator theory, velocity regulator has been designed specifically for the situation where only locomotives' velocities are measurable [27], and based on which, fault-tolerant control scheme is developed for handing the velocity sensor faults and actuator faults [4]. Moreover, based on a discrete train model, the optimal control of heavy-haul trains is achieved by using Model Predictive Control (MPC) [3, 28].

In addition to the aforementioned approaches developed specifically for heavy-haul trains, we can also draw inspirations from some other emerging approaches that can cope with complicated train dynamics. Firstly, since Neural Networks (NNs) are capable of nonlinear function approximation, they have been utilized to approximating the nonlinear train dynamics when taking into account actuator failures [29] and actuator saturations [30, 31] in the closed-loop control design. Secondly, robust control methods, including sampled-data control [32], guaranteed cost control [33], H_∞ control [34, 35] have been used for velocity tracking control of high-speed trains, where the unknown time-varying delays, uncertain parameters and stochastic disturbance are considered. Thirdly, Iterative Learning Control (ILC) provides a mechanism for improving the tracking control accuracy using the repetitive train operation information, which has been used for train control under iterative varying wind resistance parameters [36], speed delays and input saturations [37, 38]. Moreover, recently, progress has been made in adaptive fuzzy control of nonlinear systems [39, 40]. However, these methods require a pre-designed velocity profile, for heavy-haul freight trains which have complicated dynamics, it takes considerable efforts to provide a well-designed velocity profile.

From another prospective, instead of using the pre-designed velocity profiles and the prior knowledge of train dynamics, RL [41] and data-mining [42] use the sampled driving experiences to optimize the operation strategies of metro trains. However, since a metro train is much shorter than a heavy-haul freight train, the coupler force isn't considered as an optimization objective. Taking the

coupler force as the optimization objective, the optimal air brake control for heavy-haul trains running on downward slope has been achieved with approximate dynamic programming algorithm [43] and machine learning method for data imbalance [5]. Notably, although in [43], there is an simulation environment based on the train's dynamic model, the train operation strategies are learned from the sampled driving experiences generated by the environment, rather than the prior knowledge of train dynamics. However, this research is specific in the air brake control and the minimization of coupler force, neglecting a detailed analysis on locomotive operation strategies and energy consumption.

One of the most promising approaches to solve the problem in this paper is RL [6, 7], since there are two important features:

1) RL is based on trial and error. The agent learns knowledge about the environment through exploration and exploitation. Hence, RL approach can be used for optimization and prediction without prior knowledge about the environment. RL has strong adaptability for different objectives and constraints, which has been used for traffic flow control [44], train rescheduling [45], and flight taxi-out time prediction [46].

2) RL maximizes long-term reward, rather than immediate reward. Based on function approximation, RL is able to optimize the global objective function in large scale multi-stage decision problem. For example, the computer programs developed based on RL have achieved master-level in playing backgammon [47], Atari 2600 games [8, 48, 49] and the game of Go [50, 51].

In this paper, instead of storing the action values in a table, we use a nonlinear function to approximate the action-value function. It is because, according to a brief computational complexity analysis given in subsection 5.2 of this paper, our problem has continuous state space and large action space. Besides, compared with tables, nonlinear function approximation has advantages in parameter sharing of states and actions, and denoising the action-value function, thus it can get higher convergence rate. Moreover, when using nonlinear function approximation, the action values are calculated based on the parameters of the nonlinear functions, while the table-based methods store each action value independently, the storage space can be saved a lot by using nonlinear function approximation.

Deep Q-network (DQN) is a successful nonlinear function approximator for playing Atari 2600 games [8], but its performance needs to be improved when handling RL problems which have uncertainties, large or continuous action space. By using Bayesian deep model, the uncertainties in RL problems can be handled [52]. Besides, to address RL problems which have large discrete action space or continuous action space, several policy-gradient-based RL algorithms are proposed [53, 54, 55]. However, in this paper, we propose a value-iteration-based RL algorithm for large discrete action space. Furthermore, to construct discrete features for continuous state space, several coding techniques can be used, such as sparse coarse coding [56] and tile coding [57], both of which pertain to binary coding techniques. Besides, Continuous U Tree is proposed for the discretization of large continuous state space [58]. Since the RL problem formulated in this paper has a continuous two-dimensional state space, it is efficient to use tile coding to construct the state features [6, 7]. Notably, different from previous work, tile coding is also used to construct the features of large discrete action space.

Moreover, this paper proposes a novel nonlinear function approximator, named Double-Switch Q-network (DSQ-network), for fast approximation of the action-value function in a large action space. In contrast to the existing Q-network [8] that takes the state features as the inputs of a feedforward neural network and outputs the value of each action independently, DSQ-network uses the state features and action features as two switch layers in the feedforward neural network, aiming to enhance the parameter sharing of different actions. DSQ-network can significantly improve the convergence rate, because the adjacent locomotive notches' influences on the velocities and positions of railway vehicles share similarities.

3. Knowledge gaps and contributions

It is found from the literature review that the increase of the complexity of train model and nonlinear constraints on the control inputs and train states will significantly increase the difficulties of solving the train control problem with model-based methods, while the data-driven methods, such as RL and data mining, don't suffer much from the model complexity and nonlinear constraints. There is lack of data-driven method focusing on the control strategies of multiple electric locomotives in a heavy-haul freight train. Moreover, the traction/regenerative braking force of a high-power electric locomotive is controlled by a large number of discrete notches, and varies nonlinearly with the velocity. so even if the optimal control of multiple electric locomotives in a heavy-haul freight train is transformed into a RL problem, it is still challenging due to the continuous state space and large action space. In this paper, a novel architecture for nonlinear function approximation, DSQ-network, is proposed to improve the convergence rate when approximating the optimal action-value function.

The contributions of this work are presented as follows.

1) This paper proposes a novel architecture for nonlinear function approximation in a RL problem, named DSQ-network, to enhance the parameter sharing of adjacent actions in a large action space. Compared with table-lookup Q-learning, DSQ-network gets much higher convergence rate with much less computation resources in a RL problem when the state space is continuous, action space is large, and the adjacent actions' influences on the states share similarities. The DSQ-network architecture can also be used for intelligent control of cars, trucks and some robots which have large or continuous action space. Notably, encoding state space and action space is common in RL, but it is the first time to use the state features and action features as two separated switch layers, and insert the action features between hidden layer and output layer of a feedforward neural network.

2)Without the prior knowledge of train dynamics and the pre-designed velocity profiles, this paper achieves the optimal control of multiple electric locomotives in a heavy-haul freight train, considering multiple optimization objectives, i.e., velocity, energy consumption and coupler force, and satisfying the nonlinear constraints on locomotive notches and their change rates, speed restrictions, and the traction and regenerative braking characteristics of electric locomotives.

3)This paper demonstrates the effectiveness and robustness of DSQ-network by using 28 cases in numerical experiments, where the data of China Railways HXD3B electric locomotives are used. Based on the results, we study the influences of ramps and speed restrictions on the optimal policy for the control of electric locomotives in a heavy-haul freight train. Moreover, by changing the reward weights, we investigate the inter-dependent and inter-conditioned relationships of velocity, energy consumption and coupler force. Finally, based on the visualization of the high-dimensional output of DSQ-network, we gain insights into how the speed restrictions and ramps affect the convergence processes and results of DSQ-network.

The rest of this paper is organized as follows. In section 4, the optimal control of multiple electric locomotives in a heavy-haul freight train is formulated as a standard RL problem. In section 5, firstly, the basic principles of Q-learning and Q-network are given. Then the DSQ-network architecture is introduced. In section 6, the algorithm for the implementation of DSQ-network in the optimal control of electric locomotives is given. Section 7 demonstrates DSQ-network with numerical experiments. Section 8 provides insights into the factors that influence the convergence of DSQ-network in the optimal control of electric locomotives. Section 9 gives conclusions.

4. Problem formulation

In this section, firstly, we give brief definitions for the agent, environment, state, action and reward in the optimal control of multiple electric locomotives during the operation of a heavy-haul freight train. Then we provide mathematical description of the state space, action space and reward function.

To facilitate the understanding of the application scenario, some terminologies regarding heavy-haul freight train are briefly introduced.

Notch: The discrete level of the control handle of a locomotive. For the HXD3B locomotive, there are totally 25 notches, where -12, -11, ..., -1 are the levels for regenerative braking, 0 means there is neither traction nor braking, and 1, 2, ..., 12 are the levels for traction.

Coupler: a connector between every two adjacent vehicles of a train.

Coupler force: the interactive force between every two adjacent vehicles. Excessive coupler force will result in the deformation or broken of couplers.

Traction/regenerative braking characteristics: The relationship between the traction/regenerative braking force and velocity. Generally, for each notch of a locomotive, the traction/regenerative braking force is a piecewise function of the velocity.

4.1. Definition of the RL elements

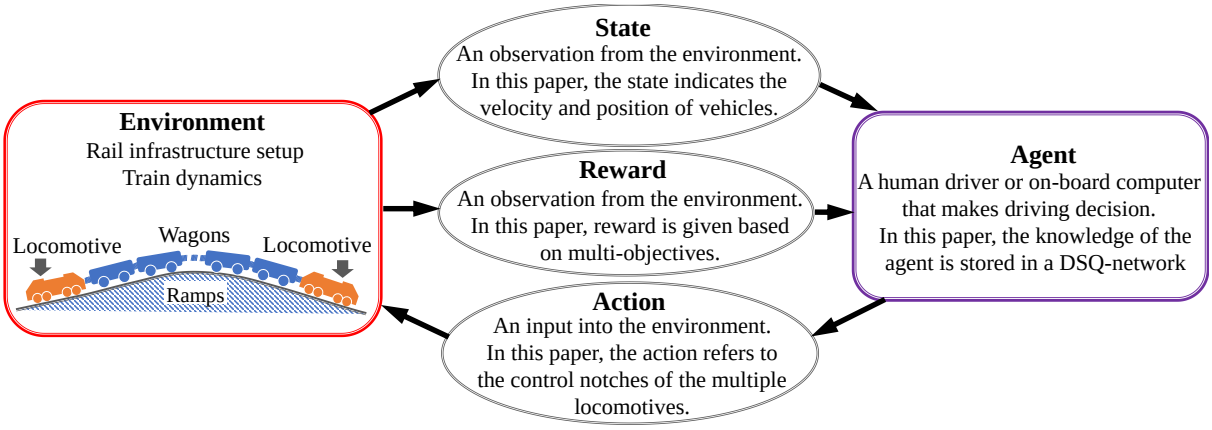

Figure 1: The RL framework for the optimal control of multiple electric locomotives during the heavy-haul freight train operation.

As shown in Fig.1, agent is defined as a human driver or an on-board computer that can change the locomotive notches, which can influence the velocities and positions of all the railway vehicles in the freight train. Environment is defined as the freight train system that gives responses to the locomotive notches selected by the agent, with the railway vehicles' velocities and positions and the evaluation signals. The velocities and positions of the railway vehicles are defined as states. The locomotive notches are defined as actions. The evaluation signals, including the velocities, energy consumption and coupler force, are defined as rewards.

Environment should establish the relationship between the action and the state transition. In the optimal control of heavy-haul freight train, environment should contain the relationship between the locomotive notches and the railway vehicles' velocities and positions. This relationship is influenced by the traction and regenerative braking characteristics of locomotives, mechanical and aerodynamic resistance, and infrastructure resistance. The traction and regenerative braking characteristics of the locomotives determine the locomotive forces based on the locomotive notches and velocities. Figs.2

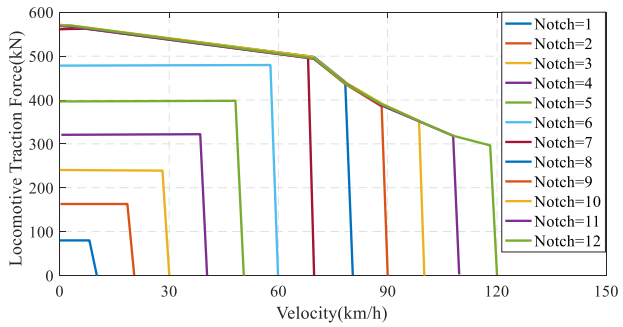

Figure 2: The traction characteristics of China Railways HXD3B electric locomotive.

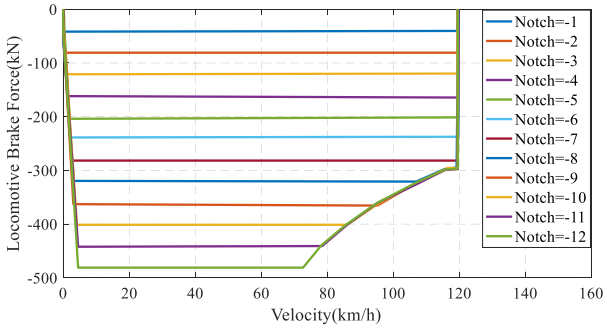

Figure 3: The regenerative brake characteristics of China Railways HXD3B electric locomotive.

and 3 respectively show the traction and regenerative braking characteristics of the China Railways HXD3B electric locomotive, which is a high-power electric locomotive used for freight transport on the main lines in China. From Figs.2 and 3 we can see, the locomotive notches are discrete, and the traction/regenerative braking force varies nonlinearly with the velocity. When using the regenerative braking mode, the kinetic energy of the heavy-haul freight train can be transformed into electrical energy, which can be recycled by the traction power supply system. In this paper, the environment is simulated based on the heavy-haul freight train dynamic model in [59]. Besides, this paper simulates the actuator faults, velocity sensor errors and coupler force sensor errors happening in the train system.

Notably, the agent does not have any prior knowledge about the train dynamics, and the environment only provides sampled data of driving experiences for the agent when interaction.

4.2. State

Considering a heavy-haul freight train which consists of N railway vehicles, and an episode which has N_e state transition steps, the state space at the end of the kth step is defined as

$$S(s_k) = \left\{ \begin{bmatrix} v_{k,1} & p_{k,1} \end{bmatrix}^T \mid p_{k,1} \in [0, P_{\mathrm{m}}], v_{k,i} \in \left[V_L^{\lim}(p_{k,i}), V_H^{\lim}(p_{k,i}) \right] \right\} \tag{1}$$

where, $k = 1, 2, ..., N_e$ and $i = 1, 2, ..., N$. T_k and s_k denote the time point and the state at the end of the kth step in an episode, respectively. $v_{k,i}$ and $p_{k,i}$ denote the ith railway vehicle's velocity and position at time point T_k, respectively. The first vehicle's velocity $v_{k,1}$ and position $p_{k,1}$ are selected as the state variables. P_{m} is the upper bound for the first railway vehicle's position $p_{k,1}$. $V_H^{\lim}(p_{k,i})$ and $V_L^{\lim}(p_{k,i})$ denote the highest and lowest permitted velocities at the position $p_{k,i}$, where $V_H^{\lim}(p_{k,i})$ is the speed restriction, and $V_L^{\lim}(p_{k,i})$ is used to prevent reverse driving when the explorations are performed.

There are two considerations in the definition of the state space, firstly, this paper uses the first vehicle's velocity $v_{k,1}$ and position $p_{k,1}$ as the state variables, rather than those of all the vehicles in the freight train. This way significantly reduces the dimensions of the state space, and improves the convergence rate. Although the influence of coupler force is not taken into consideration in the definition of state, the agent can learn it by receiving rewards, and consider it when selecting locomotive notches. Secondly, $v_{k,i} \in \left[V_L^{\lim}(p_{k,i}), V_H^{\lim}(p_{k,i}) \right]$ considers the speed restrictions for all the vehicles simultaneously, rather than only the first vehicle's speed restriction, because a freight train which consists of hundreds of vehicles can cover several sections with different speed restrictions. Although there are $N - 1$ vehicles whose velocities are not defined as state variables, the speed restrictions for them also put constraints on the velocity of the first vehicle.

In an exploration episode, it is required to define the starting state s_0 and terminal state s_{N_e}. In this paper, s_0 is selected randomly from the state space, because the diversity of the starting states can improve the solving efficiency and avoid local optimum. The terminal states form a set

$$S(s_{N_e}) = \left\{ \begin{bmatrix} v_{N_e,1} & p_{N_e,1} \end{bmatrix}^T \;\middle|\; p_{N_e,1} = P_{\mathrm{m}}, v_{k,i} \in \left[V_L^{\lim}(p_{N_e,i}), V_H^{\lim}(p_{N_e,i}) \right] \right\} \tag{2}$$

4.3. Action

Considering a heavy-haul freight train that is pulled by N_l electric locomotives. For the i_lth locomotive in the freight train, the notch at state s_k is denoted by M_{k,i_l}^l, where $i_l = 1, 2, ..., N_l$. The notches of all the locomotives in the freight train are defined as action variables. During the period between s_{k-1} and s_k, action a_{k-1} is performed. The action is changed from a_{k-1} to a_k at state s_k. The action a_k is selected from the action space

$$A(s_k) = \left\{ \begin{bmatrix} M_{k,1}^l & M_{k,2}^l & \cdots & M_{k,N_l}^l \end{bmatrix}^T \;\middle|\; M_{k,i_l}^l \in \left[M_{\min}^l, M_{\max}^l \right], \Delta M_{k,i_l}^l \in \left[\Delta M_{\min}^l, \Delta M_{\max}^l \right] \right\} \tag{3}$$

where, $A(s_k)$ is a set that contains all the legal actions when encountering state s_k, and $a_k \in A(s_k)$. $M_{\max}^l$ and $M_{\min}^l$ are respectively the upper and lower bound of the locomotive notches, which are determined by the type of locomotives. $\Delta M_{k,i_l}^l$ is the increment or decrement of locomotive notches at state s_k. $\Delta M_{\max}^l$ and $\Delta M_{\min}^l$ are respectively the upper and lower bound of $\Delta M_{k,i_l}^l$. $\Delta M_{\max}^l$ and $\Delta M_{\min}^l$ restrict the change rates of the locomotive notches, to improve the smoothness of train operation, and the tolerance for the slow response of the mechanical system of a heavy-haul freight train.

4.4. Reward

After the state of a heavy-haul freight train is transformed from s_k to s_{k+1} by taking action a_k, the agent will receive a reward signal to evaluate the train's velocity, energy consumption and coupler force during the state transition from s_k to s_{k+1}. A reward signal that takes into account the actuator faults, the measurement errors of velocity sensors and coupler force sensors is defined as

$$r_{k+1} = \mu_v r_{k+1}^v + \mu_e r_{k+1}^e + \mu_c r_{k+1}^c + \mu_h r_{k+1}^h + \mu_l r_{k+1}^l \tag{4}$$

where, r_{k+1}^v, r_{k+1}^e and r_{k+1}^c are the evaluation indicators of velocity, energy consumption and coupler force, respectively. r_{k+1}^h and r_{k+1}^l are the penalty signals for the agent's explorations higher than the highest permitted velocity $V_H^{\lim}$ and lower than the lowest permitted velocity $V_L^{\lim}$, respectively. μ_v, μ_e, μ_c, μ_h and μ_l are the corresponding weights for r_{k+1}^v, r_{k+1}^e, r_{k+1}^c, r_{k+1}^h and r_{k+1}^l. The mathematical expressions are

$$r_{k+1}^v = -\left| \hat{v}_{k+1,1} - V_{\mathrm{m}} \right|, \quad r_{k+1}^e = -\int_{T_k}^{T_{k+1}} F(\hat{\boldsymbol{f}}_l)^T \hat{\boldsymbol{v}}_l dt, \quad r_{k+1}^c = -\max_{T_k \le t \le T_{k+1}} \left\| \hat{\boldsymbol{f}}_c \right\|_\infty \tag{5}$$

$$r_{k+1}^h = \begin{cases} -c_1 \left(v_{k+1,1} - V_H^{\lim}(p_{k+1,1}) + c_2 \right)^{c_3} - c_4, & v_{k+1,1} - V_H^{\lim}(p_{k+1,1}) > 0 \\ 0, & v_{k+1,1} - V_H^{\lim}(p_{k+1,1}) \le 0 \end{cases} \tag{6}$$

$$r_{k+1}^l = \begin{cases} -c_5 \cdot \left(V_L^{\lim}(p_{k+1,1}) - v_{k+1,1} + c_6 \right)^{c_7} - c_8, & V_L^{\lim}(p_{k+1,1}) - v_{k+1,1} > 0 \\ 0, & V_L^{\lim}(p_{k+1,1}) - v_{k+1,1} \le 0 \end{cases} \tag{7}$$

In r^v_{k+1}, $\hat{v}_{k+1,1}$ denotes the measured velocity of the first vehicle at the time point T_{k+1}. V_{m} is the maximum velocity of locomotives, which is determined when the locomotives are designed. It can be found that r^v_{k+1} increases with the increase of velocity.

In r^e_{k+1}, $\hat{\boldsymbol{v}}_l = [\hat{v}^l_1 \quad \hat{v}^l_2 \quad \dots \quad \hat{v}^l_{N_l}]^T$, $\hat{\boldsymbol{f}}_l = [\hat{f}^l_1 \quad \hat{f}^l_2 \quad \dots \quad \hat{f}^l_{N_l}]^T$, $F(\hat{\boldsymbol{f}}_l) = [F(\hat{f}^l_1) \quad F(\hat{f}^l_2) \quad \dots \quad F(\hat{f}^l_{N_l})]^T$. For $i_l = 1, 2, ..., N_l$, $\hat{v}^l_{i_l}$ denotes the measured velocity of the i_lth locomotive in the heavy-haul freight train, $\hat{f}^l_{i_l}$ denotes the actuator force of the i_lth locomotive, and $F(\cdot)$ is a piecewise function defined as

$$F(x) = \begin{cases} \eta_t^{-1} x, & x \geq 0 \\ \eta_b x, & x < 0 \end{cases} \tag{8}$$

where, η_t and η_b are the energy efficiency ratio of the electric traction drive system and the regenerative braking system, respectively. $F(\hat{f}_{i_l})$ indicates that the heavy-haul freight train consumes electrical energy when $\hat{f}_{i_l} \geq 0$, and recycles the electrical energy transformed from the kinetic energy when $\hat{f}_{i_l} < 0$.

In r^c_{k+1}, $\hat{\boldsymbol{f}}_c = [\hat{f}^c_1 \quad \hat{f}^c_2 \quad \dots \quad \hat{f}^c_{N-1}]^T$. For $i_c = 1, 2, ..., N-1$, $\hat{f}^c_{i_c}$ denotes the measured coupler force between the i_cth and the $i_c + 1$th vehicles in the heavy-haul freight train. $\left\|\hat{\boldsymbol{f}}_c\right\|_\infty = \max_{i_c} \left|\hat{f}^c_{i_c}\right|$, where $\left|\hat{f}^c_{i_c}\right|$ is the absolute value of $\hat{f}^c_{i_c}$. The evaluation indicator of coupler force is defined as the maximum value of $\left\|\hat{\boldsymbol{f}}_c\right\|_\infty$ during the period from T_k to T_{k+1}, because the main cause of coupler broken accidents is excessive coupler force.

In r^h_{k+1} and r^l_{k+1}, $c_1, c_2, \cdots, c_8$ are all positive constants. Hence, when the agent explores the state space beyond $V^{\lim}_H(p_{k+1,1})$, r^h_{k+1} decreases with the increase of $v_{k+1,1} - V^{\lim}_H(p_{k+1,1})$. Similarly, when the agent explores the state space below $V^{\lim}_L(p_{k+1,1})$, r^l_{k+1} decreases with the increase of $V^{\lim}_L(p_{k+1,1}) - v_{k+1,1}$.

Notably, in practice, we expect to maximize the velocity while minimizing the energy consumption and coupler force. However, in a standard RL problem, we always hope to maximize the expectation of accumulative rewards. Therefore, r^v_{k+1}, r^e_{k+1}, r^c_{k+1}, r^h_{k+1} and r^l_{k+1} are all defined as negative values.

5. Methodology

This section introduces the DSQ-network architecture used for approximating the optimal action-value function. Firstly, the basic principles of Q-learning and Q-network are introduced. Then we give a brief analysis on the computational complexity of table-lookup Q-learning. Finally, the DSQ-network architecture is introduced.

5.1. Principles of Q-learning and Q-network

Q-learning [60] is an off-policy Temporal Difference (TD) control algorithm. It is used to approximate the optimal action-value function and the optimal policy in a RL problem. Q-learning stores all the action values independently in a table. The action values will be updated when the agent encounters the corresponding state-action pairs. Q-network [8] follows the basic principles of Q-learning, but uses feedforward neural network to approximate the optimal action-value function. Different state-action pairs share the parameters in the neural network. When an action value is updated, the values of many other state-action pairs are also updated. The following are the basic principles of Q-learning and Q-network.

Firstly, let $Q^\pi(s, a)$ denote the action-value function. $Q^\pi(s, a)$ is defined as the expectation of the discounted sum of the rewards after selecting action a at state s, when policy π is followed [6]. The mathematical definition is

$$Q^\pi(s, a) = E_\pi \left[r_{k+1} + \gamma r_{k+2} + ... + \gamma^{N_e - k - 1} r_{N_e} \;\middle|\; s_k = s, a_k = a \right] \tag{9}$$

where, γ is the discount factor for the future rewards, and satisfys $0 \leq \gamma \leq 1$. Policy $\pi = P(a|s)$ gives the probability of selecting an action $a \in A(s)$ when state s is encountered.

The optimal policy π^* in a RL problem should satisfy

$$Q^{\pi^*}(s,a) \geq Q^{\pi}(s,a), \quad \text{for all } \ s \in S(s) \ \text{ and } \ a \in A(s). \tag{10}$$

where, $Q^{\pi^*}(s,a)$ is the optimal action-value function, which satisfys the Bellman optimality equation

$$Q^{\pi^*}(s,a) = E_{\pi^*}\left\{ r_{k+1} + \gamma \max_{a'} Q^{\pi^*}(s_{k+1}, a') \;\middle|\; s_k = s, a_k = a \right\} \tag{11}$$

Based on (11), both table-lookup Q-learning and Q-network obtain the optimal policy π^* and the optimal action-value function $Q^{\pi^*}(s,a)$ through iteratively updating $Q(s,a)$. At each iteration, if there is a state transition (s,a,r,s'), $r + \gamma \max_{a'} Q(s',a')$ is used as the target value for the update of $Q(s,a)$.

For table-lookup Q-learning [60], the formula for updating $Q(s,a)$ is

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \tag{12}$$

where, α denotes the learning rate, more specifically, α determines the weights of the current $Q(s,a)$ and the estimated target value $r + \gamma \max_{a'} Q(s',a')$ in the updated $Q(s,a)$.

Q-network is a combination of the principles of Q-learning and feedforward neural network. The feedforward neural network receives the features of state, and outputs the action values, which is called function approximator, and denoted by $Q(s,a;\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a vector consisting of the weights of the neural network.

According to [8], the target network and experience replay can improve the stability of Q-network. Thus, the loss function for training Q-network is

$$J(\boldsymbol{\theta}) = E_{(s,a,r,s')}\left[\left(r + \gamma \max_{a'} Q(s',a',\hat{\boldsymbol{\theta}}) - Q(s,a,\boldsymbol{\theta}) \right)^2 \right] \tag{13}$$

where, E denotes expectation, and (s,a,r,s') denotes a state transition step. $\hat{\boldsymbol{\theta}}$ is the parameter vector of the target network. We need to minimize the loss function $J(\boldsymbol{\theta})$ through iteratively updating the parameter vector $\boldsymbol{\theta}$. Generally, the optimal parameter vector $\boldsymbol{\theta}^*$ is obtained by using stochastic gradient descent method

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \left[r + \gamma \max_{a'} Q(s',a',\hat{\boldsymbol{\theta}}) - Q(s,a,\boldsymbol{\theta}) \right] \nabla_{\theta} Q(s,a;\boldsymbol{\theta}) \tag{14}$$

The following discussions are all based on the above principles.

5.2. Brief discussion on table-lookup Q-learning

Table-lookup Q-learning uses a high-dimensional table to store all the action values, as shown in Fig.4. Each element of the table is the action value for a state-action pair (s,a). When encountering state s, the agent will select an action a based on $Q(s,a)$, then update $Q(s,a)$ with the formula (12).

Notably, table-lookup Q-learning requires discretization of the continuous variables in state space and action space. Considering a situation where the length of the rail line is 40km, and two HXD3B locomotives, with a maximum velocity of 120km/h, are used in a freight train, if the velocity resolution and position resolution are respectively set to 1km/h and 0.1km, the number of elements in state space is $400 \times 120 = 48000$, and in action space is $25 \times 25 = 625$. Hence, the total number of elements in the table of $Q(s,a)$ is 30 million. In the problem of this paper, table-lookup Q-learning has a low convergence rate, and requires large memory space. Moreover, since each $Q(s,a)$ is updated independently, there are a lot of noises in the action-value function.

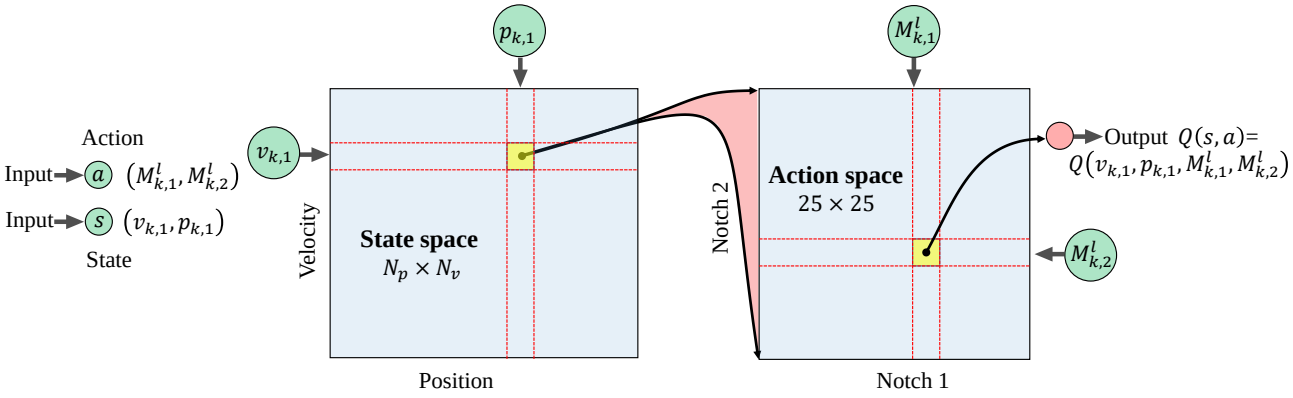

Figure 4: Table-lookup Q-learning.

5.3. DSQ-network architecture

Taking advantages of parameter sharing of states and actions, nonlinear function approximation can improve the convergence rate, use less storage space and denoise the action value function. Q-network is a promising nonlinear function approximator, since its stability has improved a lot with the target network and experience replay [8].

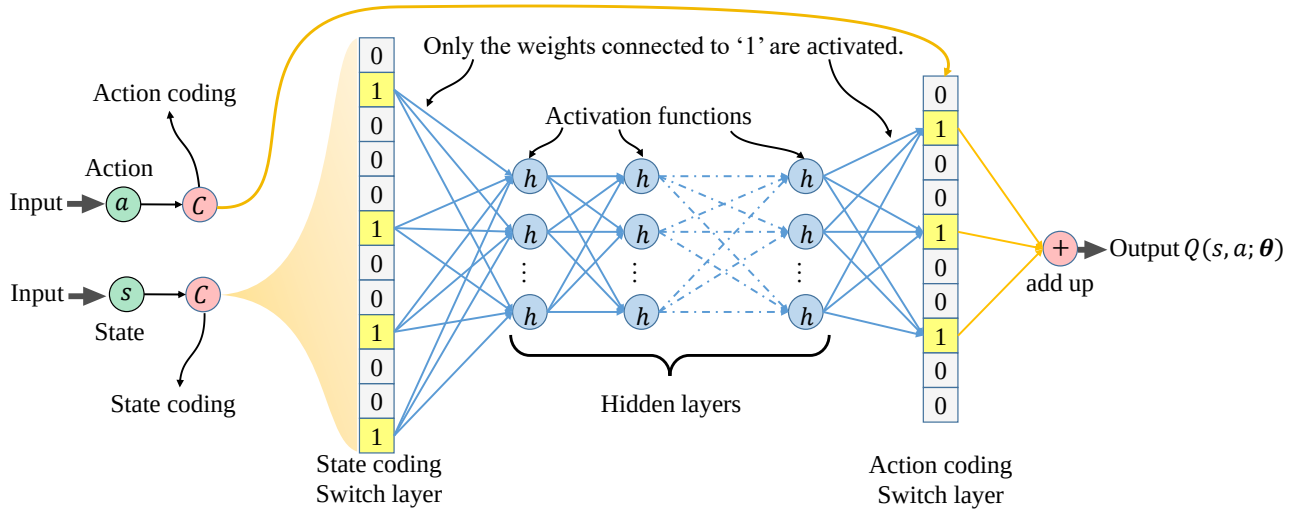

Figure 5: DSQ-network architecture.

Unlike the existing Q-network [8] that uses state features as inputs, and outputs the value of each action independently, the DSQ-network architecture takes the binary features of state and action as two switch layers, as shown in Fig.5, where both the state features and action features are constructed by using tile-coding [6, 7].

Tile-coding converts the state vector (v, p) and action vector (M_1, M_2) into binary feature vectors, where v, p, M_1 and M_2 denote velocity, position, notch 1 and notch 2, respectively. The two binary feature vectors serve as two switch layers in a feedforward neural network. The state switch layer determines which weights between the input layer and the first hidden layer are activated, where the weights connected to 1 are activated, and those connected to 0 are ignored. The input of each node of the first hidden layer is a weighted sum of the network inputs which are connected to the activated weights. The action switch layer determines which nodes in the last hidden layer should offer the values that will be added to the final $Q(s, a; \boldsymbol{\theta})$.

Notably, there is a difference between the implementation processes of state coding and action coding. Since the state space is continuous, the offsets of each tiling along the position dimension and velocity dimension are randomly selected. By contrast, the action space is a discrete space, so it must be ensured that each action is represented by a unique feature vector, which means each feature vector cannot be shared by more than one action. Therefore, the offsets of the tilings for action coding are specified.

For state coding, a tiling is a two-dimensional grid that covers the whole state space, which is cut into $N_v \times N_p$ uniform gridlike tiles. The size of each tile is $\Delta v \times \Delta p$, where Δv and Δp are the tile's lengths along the velocity dimension and position dimension, respectively. There are some basic constraints on N_v, N_p, Δv and Δp, as shown in (15).

$$N_v \cdot \Delta v \geq V_{\mathrm{m}}, \;\; N_p \cdot \Delta p \geq P_{\mathrm{m}} \tag{15}$$

If one tiling is used for encoding the state, there are $N_v \times N_p$ binary elements in the feature vector. Let $\boldsymbol{\varphi}(s)$ denote the feature vector of state s, and $\varphi_{i_s}(s)$ denote the i_sth element in $\boldsymbol{\varphi}(s)$, $i_s = 1, 2, \cdots, N_v \times N_p$. We have $\boldsymbol{\varphi}(s) = [\varphi_1(s) \quad \varphi_2(s) \quad \cdots \quad \varphi_{N_v \times N_p}(s)]^T$, where $\varphi_{i_s}(s)$ is determined by

$$\varphi_{i_s}(s) = \begin{cases} 1, & \text{if } s = (v, p) \text{ present at the } i_s\text{th tiles.} \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

When there are N_t tilings used for encoding the state, and $N_t > 1$, firstly, we obtain the i_tth tiling's feature vector $\boldsymbol{\varphi}_{i_t}(s)$ by using (16), where $i_t = 1, 2, ..., N_t$. Then we can construct the feature vector of the N_t tilings as

$$\boldsymbol{\varphi}(s) = \left[\boldsymbol{\varphi}_1^T(s) \quad \boldsymbol{\varphi}_2^T(s) \quad \cdots \quad \boldsymbol{\varphi}_{N_t}^T(s)\right]^T \tag{17}$$

Besides, the multiple tilings need to be offset by different amounts, in order to extract the features more accurately. The offsets of the tilings for state coding are randomly selected from $[0, \Delta v]$ and $[0, \Delta p]$. The legal offsets of the tilings for state coding form a set

$$\Delta_s^{\mathrm{offset}} = \left\{ (v_{i_t}^{\mathrm{offset}}, p_{i_t}^{\mathrm{offset}}) \;\middle|\; v_{i_t}^{\mathrm{offset}} \in U(0, \Delta v), \; p_{i_t}^{\mathrm{offset}} \in U(0, \Delta p), \; i_t = 1, 2, ..., N_t \right\} \tag{18}$$

The action coding also follows the rules in (15)–(17), but the set of the offsets are specified. When the tile's size is $\Delta M \times \Delta M$, the set of the tiling offsets are

$$\Delta_a^{\mathrm{offset}} = \left\{ (\Delta M - i_t, i_t) \;\middle|\; \Delta M = M_{\max}^l - M_{\min}^l + 1, \; i_t = 1, 2, ..., N_t \right\} \tag{19}$$

where, $M_{\max}^l$ and $M_{\min}^l$ are the upper and lower bound of locomotive notches, respectively. Each tiling for action coding consists of $N_{M_1} \times N_{M_2}$ uniform gridlike tiles.

Notably, in the following sections, to distinguish the N_t for action from state, we let N_t^s and N_t^a denote the number of tilings for state coding and action coding, respectively.

From this section we can see, the problem in this paper has continuous state space and large action space. Table-lookup Q-learning suffers from low convergence rate, large memory space, and noises. Hence, we propose a novel nonlinear function approximator, DSQ-network, to improve the efficiency of solving the peoblem, reduce the required memory space and denoise the action-value function. DSQ-network can enhance the parameter sharing of state-action pairs, denoise the action-value function, and have higher accuracy than the linear function approximator. Unlike the existing Q-network that outputs the value of each action independently, DSQ-network enhances the parameter sharing of actions by tile coding.

Notably, it is common to use the features of states and actions as the inputs of function approximator, but it is the first time to use the state features and action features as separated switch layers, and insert the action features between the hidden layer and output layer of a feedforward neural network. DSQ-network is specially for the large action space where adjacent actions share similarities.

6. Algorithm

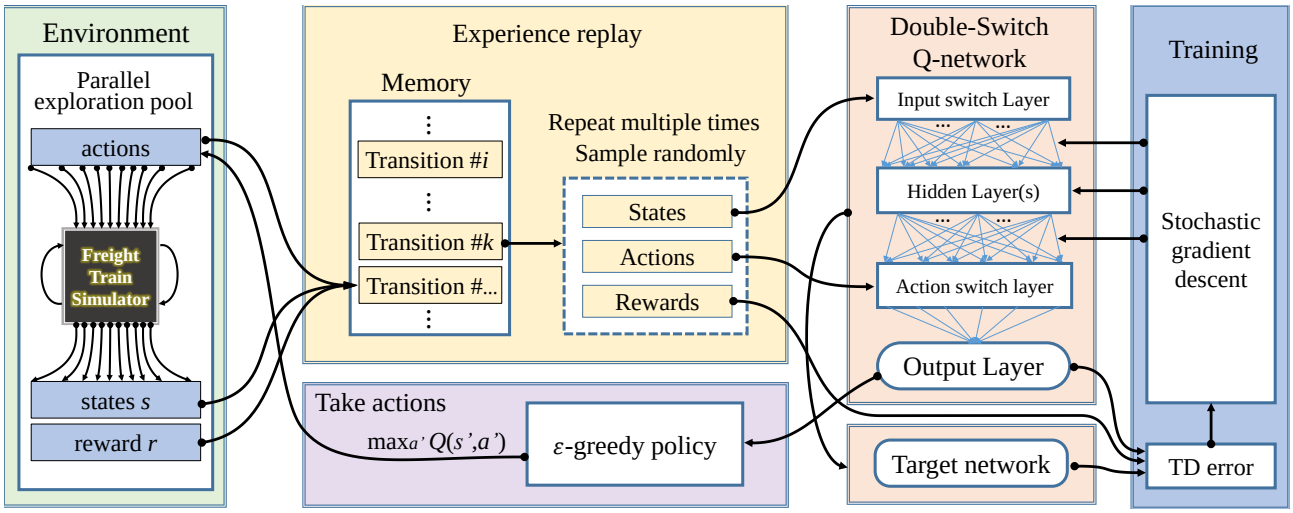

Figure 6: The computation framework of the optimal control of multiple electric locomotives in a heavy-haul freight train using DSQ-network.

Fig.6 shows the computation framework for using DSQ-network to learn the optimal control strategies of multiple electric locomotives in a heavy-haul freight train. The computation framework mainly includes: firstly, when encountering a state s, the agent selects an action a by using ε-greedy policy and $Q(\boldsymbol{\varphi}(s), \boldsymbol{\varphi}(a); \boldsymbol{\theta})$, for all $a \in A(s)$. Secondly, the state transition (s, a, r, s') is calculated by using the freight train simulator in the environment, where parallel explorations are performed simultaneously, in order to generate the episodes fast. The starting states of all the episodes are selected randomly from the state space, and the terminal states locate at P_{m}. Thirdly, the state transitions are stored independently in the experience replay memory D, and randomly selected for updating the online network parameter vector $\boldsymbol{\theta}$. $\boldsymbol{\theta}$ is updated using the stochastic gradient descent method. The target network parameter vector $\hat{\boldsymbol{\theta}}$ is copied periodically from $\boldsymbol{\theta}$.

The pseudo code of the algorithm is presented as follows.

Algorithm: DSQ-nework for optimal control of electric locomotives in a heavy-haul train

Main Inputs:

• *Environment*: train marshalling, gradients of ramps, speed restrictions.

• *Tile Coding*: tile length along velocity dimension Δv, tile length along position dimension Δp, tile length along notch dimension ΔM, number of tilings for state coding N_t^s, number of tiles along velocity dimension N_v, number of tiles along position dimension N_p, number of tilings for action coding N_t^a, number of tiles along notch 1 dimension N_{M_1}, number of tiles along notch 2 dimension N_{M_2}.

• *Function approximation*: learning rate α, discount factor γ, activation function $h(\cdot)$, number of hidden layers, number of nodes in each hidden layer.

• *Training configuration*: simulation step Δt, state transition step ΔT, the maximum number of state transition steps, the size of experience relay memory, the update frequency of target network, sampling batch_size.

Initialize:

• action-value function Q with random weights $\boldsymbol{\theta}$.

• target action-value function $\hat{Q}$ with weights $\hat{\boldsymbol{\theta}}$.

• experience replay memory D.

• simulation environment (**env**), random initial state $s = (v, p)$ and action $a = (M_1, M_2)$ for each episode in the parallel exploration pool.

For step = 1: the maximum number of state transition steps, **do**

For each of the parallel exploration episodes, **do**

With probability ε randomly select a legal action $a = (M_1, M_2)$,
otherwise select $a = \arg\max_a Q(\boldsymbol{\varphi}(v, p), \boldsymbol{\varphi}(M_1, M_2); \boldsymbol{\theta})$.

Execute action a in **env** for ΔT, then observe the new state $s' = (v', p')$, and read $\hat{\boldsymbol{v}}_l$, $\hat{\boldsymbol{f}}_l$ and $\hat{\boldsymbol{f}}_c$ during the period from s to s'.

Store transition $(v, p, M_1, M_2, r, v', p')$ in D.

If (p', v') is the terminate state, **do**

Randomly initialize state $s = (v, p)$ and action $a = (M_1, M_2)$.

End If

End For

For _in range(batch_size), **do**

Select a transition $(v, p, N_1, N_2, r, v', p')$ randomly from D.

Calculate the state feature vectors $\boldsymbol{\varphi}(v, p)$, $\boldsymbol{\varphi}(v', p')$ and action feature vector $\boldsymbol{\varphi}(M_1, M_2)$.

In particular, calculate $\boldsymbol{\varphi}(M_1', M_2')$ for all the actions in set $A(s')$.

Set the target value,

$$y = \begin{cases} r, & \text{if } (v', p') \text{ is the terminal state.} \\ r + \gamma \max\limits_{(M_1', M_2')} \hat{Q}(\boldsymbol{\varphi}(v', p'), \boldsymbol{\varphi}(M_1', M_2'); \hat{\boldsymbol{\theta}}) \end{cases} \tag{20}$$

Perform stochastic gradient descent using

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \left[y - Q(\boldsymbol{\varphi}(v, p), \boldsymbol{\varphi}(M_1, M_2); \boldsymbol{\theta}) \right] \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\varphi}(v, p), \boldsymbol{\varphi}(M_1, M_2); \boldsymbol{\theta}) \tag{21}$$

Reset $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}$ every C state transition steps.

End For

End For

7. Numerical experiment

This section demonstrates DSQ-network by numerical experiments in MATLAB. Based on the results, we investigate 1) the influences of ramps and speed restrictions on the optimal policy for the control of multiple electric locomotives in a heavy-haul freight train, and 2) the inter-dependent and inter-conditioned relationships of velocity, energy consumption and coupler force.

In the numerical experiments, the heavy-haul freight train consists of two HXD3B electric locomotives and 200 wagons, where the two locomotives respectively locate at the head and end of the train. Each locomotive weighs 150 tons, and each wagon weighs 70 tons. Hence, we can obtain that the total weight of the heavy-haul freight train is14300 tons. The length of the train is 2.45km. The traction and regenerative braking characteristics of HXD3B locomotive are used, as shown in Figs.2 and 3. The parameters for the implementation of DSQ-network are given in Table.1.

Table 1: Parameters for the implementation of DSQ-network

Parameter	Value
Tile length along velocity dimension Δv	40km/h
Tile length along position dimension Δp	4km
Number of tiles along velocity dimension N_v	4
Number of tiles along position dimension N_p	11
Number of tilings for state coding N_t^s	30
Tile length along notch 1 dimension, ΔM_1	7
Tile length along notch 2 dimension, ΔM_2	7
Number of tiles along Notch 1 dimension, N_{M_1}	2
Number of tiles along Notch 2 dimension, N_{M_2}	2
Number of tilings for action coding N_t^a	25
Lower bound of locomotive notch's change rate $\Delta M^l_{\min}$	–1
Upper bound of locomotive notch's change rate $\Delta M^l_{\max}$	1
Lower bound of locomotive notch $M^l_{\min}$	–12
Upper bound of locomotive notch $M^l_{\max}$	12
Discount factor γ	0.95
Learning rate α	10^{-5}
Exploration Probability ε	Decrease from 0.5 to 0.1, and be linear with the state transition steps.
Number of hidden layers in DSQ-network	1
Number of nodes in each hidden layer	256
Activation function $h(\cdot)$	Rectified linear unit
Number of parallel episodes	32
State transition step ΔT	10 seconds
Simulation step Δt	0.1 seconds
The maximum number of state transition steps	10000
The size of experience replay memory	16000 state transitions
The update frequency of target network	Every 20 state transition steps
sampling batch_size	900 state transitions
The maximum velocity of locomotive V_{m}	120km/h
The lowest permitted velocity $V_L^{\lim}$	15km/h
Energy efficiency ratio of electric traction drive system η_t	0.9
Energy efficiency ratio of regenerative braking system η_b	0.8
The reward weight of coupler force μ_c	2×10^{-6}
The reward weight of energy consumption μ_e	7.5×10^{-6}
The reward weight of velocity μ_v	0.3
Parameters of the penalty signals for explorations beyond $V_H^{\lim}$ or $V_L^{\lim}$	$\mu_h = 1, \mu_l = 1$ c_1 =0.05, c_2=3, c_3=3, c_4=10 c_5=0.05, c_6=6, c_7=3, c_8 =10

7.1. Influences of ramps and speed restrictions

Four cases are used to investigate the influences of ramps and speed restrictions on the optimal policy for the control of multiple electric locomotives in a heavy-haul freight train, including case 1 (Ramp 1/Speed restriction 1), case 2 (Ramp 2/Speed restriction 1), case 3 (Ramp 1/Speed restriction 2), and case 4 (Ramp 2/Speed restriction 2). The data of Ramp 1 and Ramp 2 are shown in Tables.2, and Speed restriction 1 and Speed restriction 2 are given in Table.3.

Fig.7 shows the changes of the approximated optimal action values with the increase of state transition steps in cases 1–4. It can be found that the approximated optimal action values converge at around 2000 state transition steps in all of the four cases.

Figs.8–11 show the dynamic performance of heavy-haul freight train in cases 1–4 when the opti-

Table 2: Gradients of Ramps

Ramp 1		Ramp 2	
Mileage	Gradient	Mileage	Gradient
0km – 5km	0‰	0km – 5km	2.5‰
5km – 12km	–4‰	5km – 10km	5‰
12km –15km	4‰	10km –15km	–2.5‰
15km – 25km	–4‰	15km – 21km	2.5‰
25km – 28km	4‰	21km – 24km	–5‰
28km – 37km	–4‰	24km – 32km	5‰
37km – 38km	0‰	32km – 38km	2.5‰
38km – 40km	0‰	38km – 40km	0‰

Table 3: Speed restriction

Speed restriction 1		Speed restriction 2	
Mileage	$V_H^{\lim}$	Mileage	$V_H^{\lim}$
0km – 8km	70km/h	0km – 8km	85km/h
8km – 20km	85km/h	8km – 16km	65km/h
20km – 28km	75km/h	16km – 32km	90km/h
28km – 40km	95km/h	32km – 40km	70km/h

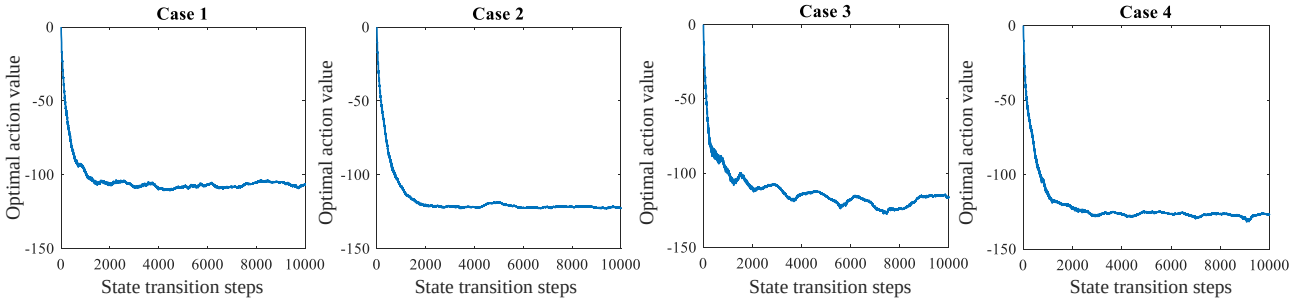

Figure 7: The changes of the approximated optimal action value for cases 1-4 with state transition steps. (The data is the average of optimal action values of 42 states.)

mal policies that are learned from 10000 state transition steps are followed, respectively. The computation time of cases 1-4 is respectively 6449 seconds, 6538 seconds, 6471 seconds, 6558 seconds(CPU/I5 4210H(2CPUs×2.9GHz), RAM/8GB, MATLAB R2016b). Here the episode that follows the optimal policy is called exploited episode. Each figure has four subfigures showing the changes of the locomotive velocities, the locomotive notches, the maximum and minimum coupler forces, and the energy consumption with the increase of mileage. Notably, in the first subfigure, the length of the train is considered, so there is a position difference between the two locomotives, while in the other three subfigures, the horizontal coordinate is the position of locomotive #1.

From Fig.8 we can see, firstly, regenerative braking is utilized to keep the train from exceeding the speed restrictions, because case 1 has three longer downhill sections and two shorter uphill sections, where the gravitational potential energy is converted into kinetic energy, increasing the train velocity. Secondly, the rear locomotive gives lower notches than the head locomotive between 15km and 25km, because there is a downhill-uphill connection where the heavy-haul freight train suffers from compressive coupler forces. When the rear locomotive provides larger regenerative braking force than the head locomotive, the freight train can be stretched, and the coupler forces are reduced. Conversely, between 30km and 40km, after the operation on an uphill-downhill connection that makes the train stretch, the rear locomotive gives higher notch than the head locomotive, to make the train compressive and reduce the stretching coupler forces. Compared with case 1, case 2 has longer uphill sections and shorter downhill sections. It can be found from Fig.9 that the difference between the locomotive notches is much smaller than that in Fig.8, especially on the downhill-uphill connections and the uphill-downhill connections. The reason is that the acceleration of the heavy-haul freight train on long uphill sections requires both locomotives provide large traction forces, otherwise the train will decelerate, and the agent will receive a bad reward due to low velocity. Besides, according to

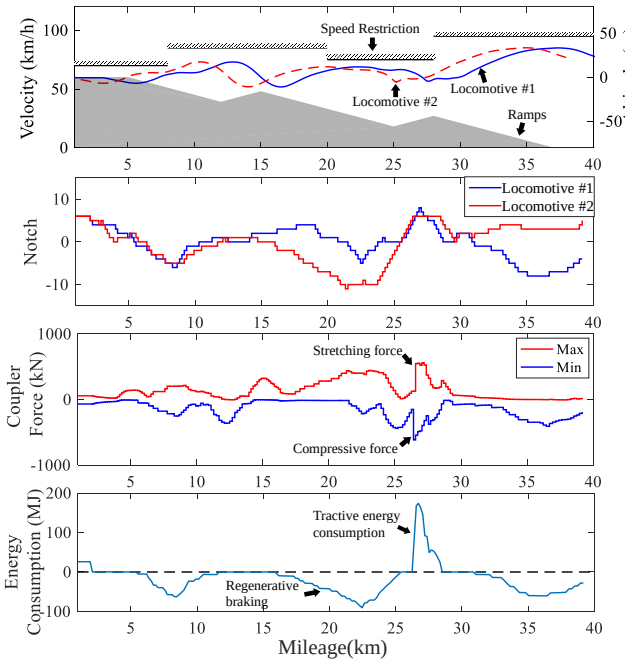

Figure 8: The dynamic performance of heavy-haul freight train under the optimal policy: Case 1.(The first subfigure considers the position difference between the two locomotives, while in the other three subfigures, the horizontal coordinate is the position of the locomotive #1.)

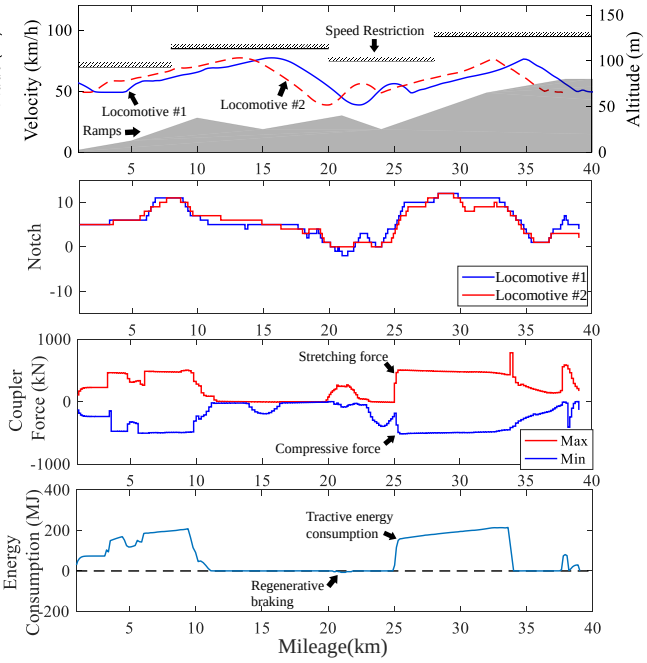

Figure 9: The dynamic performance of heavy-haul freight train under the optimal policy: Case 2.(The first subfigure considers the position difference between the two locomotives, while in the other three subfigures, the horizontal coordinate is the position of the locomotive #1.)

the traction characteristics of HXD3B locomotive, a small difference between the notches of different locomotives may result in a situation where the force of one locomotive is large, while the force of the other locomotive is zero. Consequently, the heavy-haul freight train cannot pass the long uphill sections due to lack of traction force. Moreover, in case 2, the total energy consumption is 18067.77 MJ and the average coupler force (the average of the absolute value of coupler force rewards in the exploited episode) is 283.81kN, while those of case 1 are –2719.17 MJ and 212.60kN, respectively. It is found that on the long uphill sections, the energy consumption rewards have greater influences on the action-value function than the coupler force rewards.

Cases 3 and 4 use the same ramps as cases 1 and 2, respectively, and a lower average speed restriction. In case 3, the operation time, energy consumption and the average coupler force are respectively 2321.7s, -3911.58MJ and 276.40kN, while those for case 1 are respectively 2211.9s, –2719.17MJ, and 212.60kN. Naturally, the decrease of average speed restriction results in more operation time and less energy consumption. However, it is interesting to find that the average coupler force increases. From Fig.10 we can see, a sharp coupler force happens when the train encounters a dramatic increase of speed restriction at 16km, from 65km/h to 90km/h. This is because the penalty signal disappears with the increase of speed restriction, and the velocity reward encourages higher velocity. Since the acceleration of wagons depends on the coupler force, a sharp increase of velocity will lead to an increase of coupler force. Moreover, the uphill-downhill connection at 15km and the downhill-uphill connection at 12km also exacerbate the longitudinal in-train impulse.

In case 4, the operation time, energy consumption and average coupler force are respectively 2406.6s, 17896.15MJ and 351.42kN, while those for case 2 are respectively 2463.2s, 18067.77MJ, and 283.81kN. Firstly, it is interesting to find that the operation time decreases with the drop of average speed restriction. This is because there exists a 16km long section with speed restriction of

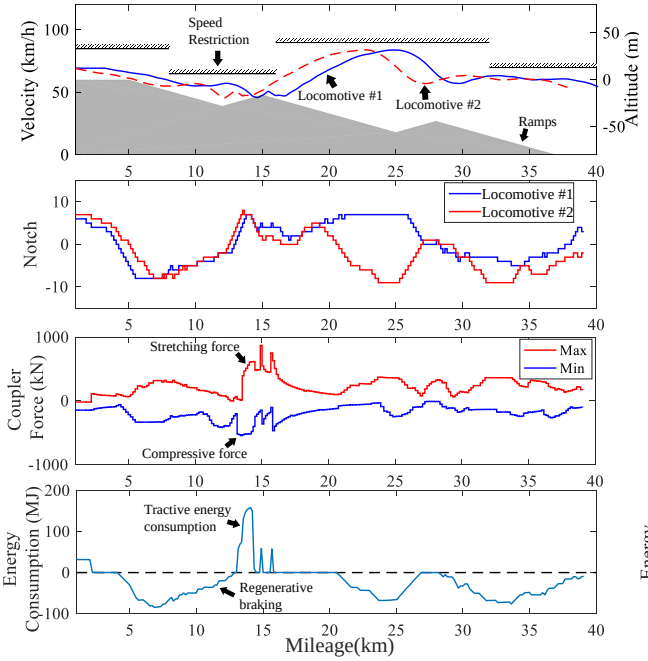

Figure 10: The dynamic performance of heavy-haul freight train under the optimal policy: Case 3.(The first subfigure considers the position difference between the two locomotives, while in the other three subfigures, the horizontal coordinate is the position of the locomotive #1.)

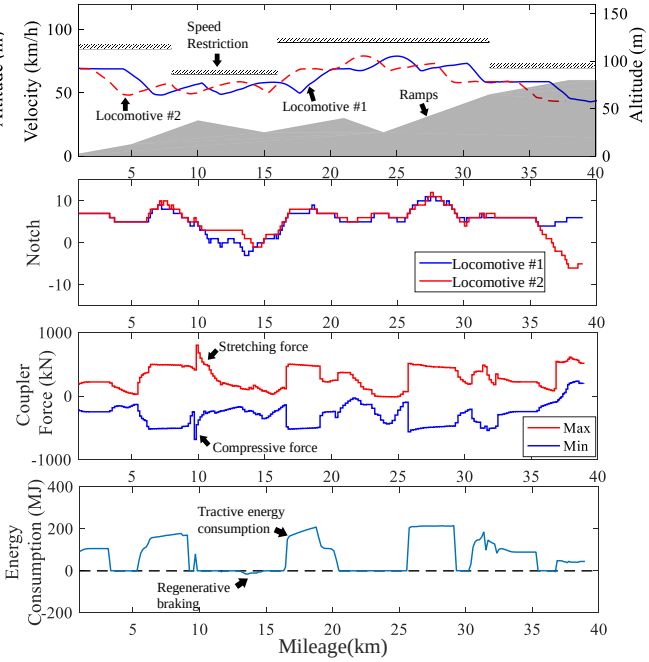

Figure 11: The dynamic performance of heavy-haul freight train under the optimal policy: Case 4.(The first subfigure considers the position difference between the two locomotives, while in the other three subfigures, the horizontal coordinate is the position of the locomotive #1.)

90km/h in case 4, which is longer than the 12km section with speed restriction of 95km/h in case 2, giving more time for the heavy-haul freight train to run at a velocity close to the highest permitted velocity. Compared with downhill section, the acceleration on uphill section is much lower. For uphill sections where the speed restriction decreases, and the section with high speed restriction is short, the heavy-haul freight train always decelerates before reaching the highest permitted velocity. Consequently, although the average speed restriction in case 4 is lower than that in case 2, the average velocity of the train is higher, resulting in less operation time. Secondly, the energy consumption in case 4 is lower than that in case 2. It is because the speed restriction in case 2 increases when the heavy-haul freight train runs on the long uphill sections at 0km–10km and 25km–34km, and the acceleration on long uphill sections requires large traction force and consumes considerable energy, while in case 4, the speed restriction decreases at 0km–10km and 25km–34km, so the required traction force is much lower, resulting in less energy consumption. Thirdly, the average coupler force in case 4 is higher than that in case 2. From Fig.11 we can see, in case 4, there is an uphill section at 15km–21km, and the speed restriction increases, so the acceleration of wagons requires large coupler force, while in case 2(Fig.9), the speed restriction decreases, the required tractive force for wagons is smaller, so the coupler force is between 15km and 21km is much smaller than that in case 4.

7.2. Influences of reward weights

The following tests investigate the inter-dependent and inter-conditioned relationships between velocity, energy consumption and coupler force by changing the reward weights μ_v, μ_e and μ_c. Seven Groups of reward weights are used in each of the four cases in 7.1. Here we use Ci/Wj to denote the case that uses the jth group of weights in case i in 7.1, where i=1, 2,$\cdots$, 4 and j=1, 2,$\cdots$, 7. The reward weights in the tests are defined as $\mu_v = K_v\Delta\mu_v$, $\mu_e = K_e\Delta\mu_e$ and $\mu_c = K_c\Delta\mu_c$, where K_v, K_e and

K_c are non-negative integers, and $\Delta\mu_v$, $\Delta\mu_e$ and $\Delta\mu_c$ are the units for changing the reward weights. The reward weights in 7.1 are used as the units for changing the reward weights, so we have $\Delta\mu_v = 0.3$, $\Delta\mu_e = 7.5 \times 10^{-6}$, $\Delta\mu_c = 2 \times 10^{-6}$. Table.4 shows the seven groups of K_v, K_e and K_c in the tests.

Table 4: The operation time, total energy consumption and coupler force of 28 cases using reward weights $\mu_v = K_v\Delta\mu_v$, $\mu_e = K_e\Delta\mu_e$, $\mu_c = K_c\Delta\mu_c$. (Ci/Wj denotes the case that uses the jth group of weights in case i in 7.1, where i=1, 2,$\cdots$, 4 and j=1, 2,$\cdots$, 7. Percentage* is the increasing rate compared with the case that uses the first Group of weights, Average coupler force* is the average of the absolute value of coupler force rewards in the exploited episode.)

Case	Weight coefficients			Operation time		Total energy consumption		Average coupler force*	
				Value(s)	Percentage*	Value(MJ)	Percentage*	Value(kN)	Percentage*
C1/W1	$K_v = 1$	$K_e = 0$	$K_c = 0$	2136.4	0%	-1954.69	0%	445.55	0%
C1/W2	$K_v = 1$	$K_e = 0$	$K_c = 1$	2069.0	-3.2%	-1784.72	+8.7%	342.09	-23.2%
C1/W3	$K_v = 1$	$K_e = 1$	$K_c = 0$	2181.3	+2.1%	-2484.13	-27.1%	317.46	-28.7%
C1/W4	$K_v = 1$	$K_e = 1$	$K_c = 1$	2211.9	+3.5%	-2719.17	-39.1%	212.60	-52.3%
C1/W5	$K_v = 1$	$K_e = 1$	$K_c = 2$	2234.8	+4.6%	-1534.69	+21.5%	191.95	-56.9%
C1/W6	$K_v = 1$	$K_e = 2$	$K_c = 1$	2217.5	+3.8%	-2347.61	-20.1%	196.74	-55.8 %
C1/W7	$K_v = 1$	$K_e = 2$	$K_c = 2$	2426.0	+13.6%	-3195.73	-63.5%	210.63	-52.7%
C2/W1	$K_v = 1$	$K_e = 0$	$K_c = 0$	2138.5	0%	20688.94	0%	401.24	0%
C2/W2	$K_v = 1$	$K_e = 0$	$K_c = 1$	2150.6	+0.6%	20879.05	+0.9%	341.73	-14.8%
C2/W3	$K_v = 1$	$K_e = 1$	$K_c = 0$	2274.5	+6.4%	19601.10	-5.3%	431.43	+7.5%
C2/W4	$K_v = 1$	$K_e = 1$	$K_c = 1$	2463.2	+15.2%	18067.77	-12.7%	283.81	-29.3%
C2/W5	$K_v = 1$	$K_e = 1$	$K_c = 2$	2427.4	+13.5%	18849.24	-8.9%	294.35	-26.6%
C2/W6	$K_v = 1$	$K_e = 2$	$K_c = 1$	3886.9	+81.8%	16921.34	-18.2%	304.62	-24.1%
C2/W7	$K_v = 1$	$K_e = 2$	$K_c = 2$	4384.6	+105.0%	16475.85	-20.4%	376.71	-6.1%
C3/W1	$K_v = 1$	$K_e = 0$	$K_c = 0$	2198.2	0%	-2626.57	0%	402.18	0%
C3/W2	$K_v = 1$	$K_e = 0$	$K_c = 1$	2280.0	+3.7%	-3392.74	-29.2%	287.61	-28.5%
C3/W3	$K_v = 1$	$K_e = 1$	$K_c = 0$	2420.5	+10.1%	-4488.07	-70.9%	292.49	-27.3%
C3/W4	$K_v = 1$	$K_e = 1$	$K_c = 1$	2321.7	+5.6%	-3911.58	-48.9%	276.40	-31.3%
C3/W5	$K_v = 1$	$K_e = 1$	$K_c = 2$	2336.0	+6.2%	-3707.63	-41.2%	280.25	-30.3%
C3/W6	$K_v = 1$	$K_e = 2$	$K_c = 1$	2432.8	+10.7%	-4403.79	-67.7%	249.54	-38.0%
C3/W7	$K_v = 1$	$K_e = 2$	$K_c = 2$	2511.6	+14.3%	-4160.48	-58.4%	177.36	-55.9%
C4/W1	$K_v = 1$	$K_e = 0$	$K_c = 0$	2231.0	0%	19318.06	0%	506.03	0%
C4/W2	$K_v = 1$	$K_e = 0$	$K_c = 1$	2250.5	+0.9%	18958.48	-1.9%	397.72	-21.4%
C4/W3	$K_v = 1$	$K_e = 1$	$K_c = 0$	2349.1	+5.3%	17796.91	-7.9%	507.54	+0.3%
C4/W4	$K_v = 1$	$K_e = 1$	$K_c = 1$	2406.6	+7.9%	17896.15	-7.4%	351.42	-30.6%
C4/W5	$K_v = 1$	$K_e = 1$	$K_c = 2$	3367.6	+50.9%	16858.95	-12.7%	283.04	-44.1%
C4/W6	$K_v = 1$	$K_e = 2$	$K_c = 1$	3842.7	+72.2%	16571.13	-14.2%	323.42	-36.1%
C4/W7	$K_v = 1$	$K_e = 2$	$K_c = 2$	4625.1	+107.3%	16262.65	-15.8%	302.36	-40.2%

Table.4 shows the train operation time, total energy consumption, and average coupler force in the 28 cases when the seven groups of reward weights are used. Besides, the increasing rates of W2-W7 compared with W1 are presented in the form of percentages. Table.5 shows the computation time of the 28 cases.

The velocity reward weight μ_v mainly influences the train operation time and energy consumption, since with the increase of velocity, the operation time gets shorter and the velocity reward shows an upward trend, but the energy consumption reward decreases. Firstly, the operation time in W1–W7 shows an upward trend with the decrease of the ratio of K_v to the sum of K_v, K_e and K_c. Significant increases of operation time happen in C2 and C4, where the operation time of W7 increases respectively by 105% and 107.3% compared with W1. Secondly, in C1–C4, with the increase of operation

Table 5: The computation time of the 28 cases at 10^3 and 10^4 state transition steps. (The numerical experiments are performed in MATLAB R2016b. Computer configuration: CPU/I5 4210H(2CPUs×2.9GHz), RAM/8GB, MATLAB R2016b)

	C1 CPU time (seconds)		C2 CPU time (seconds)		C3 CPU time (seconds)		C4 CPU time (seconds)	
	10^3 steps	10^4 steps	10^3 steps	10^4 steps	10^3 steps	10^4 steps	10^3 steps	10^4 steps
W1	654	6535	647	6467	636	6356	642	6416
W2	648	6481	627	6266	623	6234	645	6449
W3	643	6427	640	6396	644	6441	625	6247
W4	645	6449	654	6538	647	6471	656	6558
W5	624	6240	640	6397	622	6218	678	6775
W6	644	6438	638	6377	640	6396	641	6411
W7	640	6404	625	6254	632	6320	640	6403

time, the energy consumption shows an evident downward tendency, although some fluctuations exist, where the energy consumption of W7 is respectively 1241.04MJ, 4213.09MJ, 1533.91MJ and 3055.41MJ less than W1.

The energy consumption reward weight μ_e shows different influences on the operation time, energy consumption and coupler force. Firstly, with the same μ_v and μ_c, W2 and W4 of C1–C4 show that the energy consumption decreases with the increase of μ_e. A significant decrease of energy consumption happens in C1, where the energy consumption of W4 decreases by 39.1% compared with W1, while W2 increases by 8.7%. This relationship can also be seen from the results of W5 and W7 of C1–C4, and W4 and W6 of C2–C4. Secondly, the increase of μ_e can reduce the coupler force, which can be found by comparing the performance of W2, W4 and W6 of C1, C3 and C4, respectively. The changes of the average coupler force are respectively 342.09kN→212.60kN→196.74kN, 287.61kN→276.40kN→249.54kN, and 397.72kN→351.42kN→323.42kN in C1,C3 and C4. This is because the increase of μ_e requires lower energy consumption, resulting in a decrease of velocity and an increase of operation time. Besides, the velocities of wagons are directly controlled by the forces of couplers connected to them, and the mechanical resistance and aerodynamic drag of wagons decrease when the velocities decrease. Therefore, the lower velocities can lead to smaller coupler forces.

The coupler force reward weight μ_c can influence the coupler force and energy consumption. Firstly, with the same μ_v and μ_e, the coupler force shows a downward trend with the increase of μ_c. This trend can be found by comparing the coupler force of W1–W2 of C1–C4, W3–W5 of C1 and C4, W3–W4 of C2–C3, W3 and W5 of C2–C3, W6–W7 of C3–C4. The largest change of coupler force happens in W3–W5 of C4, where the average coupler force decreases from 507.54kN to 283.04kN, and the increasing rate compared with W1 decreases from +0.3% to −44.1%. Secondly, it is found that the increase of μ_c leads to higher energy consumption in W3–W5 of C3, where the change of energy consumption is -4488.07MJ→-3911.58MJ→-3707.63MJ, and the increasing rate compared with W1 is -70.9%→-48.9%→-41.2%. This trend can also be found from W1–W2 and W4–W5 of C1, W1–W2 of C2, W3–W5 and W6–W7 of C3, W3–W4 of C4. The reasons for the change of coupler force and energy consumption are: firstly, the differences of locomotive forces can counteract the coupler force caused by ramps, and therefore the in-train impulse is reduced. Secondly, compared with the same locomotive forces, different locomotive forces consume extra energy, because the force from one locomotive also counteract that of the other locomotive.

8. Discussion and insights

8.1. The high-dimensional output of DSQ-network

The control of multiple electric locomotives in a heavy-haul freight train is formulated as a RL problem that has continuous state space and large action space, and the adjacent actions' influences on the states share similarities. When the length of the rail line is 40 km, DSQ-network converges around 2000 state transition steps, while in our test of the table-lookup Q-learning, the action values converge only when the length of the rail line is not greater than 3 km. For a rail line with a length of 3 km, the table-lookup Q-learning starts to converge around 7.5×10^5 state transition steps, much slower than DSQ-network. This is because, for Q-learning, the action values are stored in a high-dimensional table, and updated independently during the learning process. Hence, the agent can only learn the knowledge about the state-action pairs that have been visited. However, for DSQ-network, the elements in feature vectors and the weights of neural network are shared by different state-action pairs. Each update of $Q(s, a; \boldsymbol{\theta})$ is based on only one state transition (s, a, r, s'), but it can change the values of many other state-action pairs, including those state-action pairs that have never been visited. Sharing of elements in feature vectors and the weights of neural network makes it possible to generalize the limited experiences to large state-action space, and denoises the action-value function.

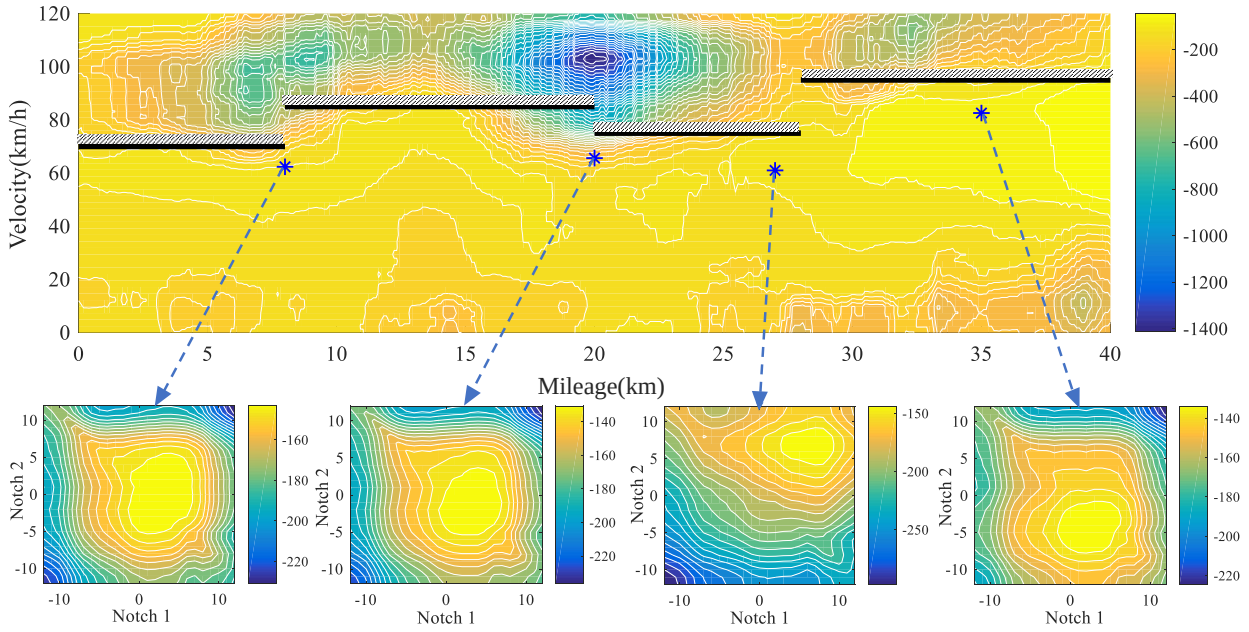

Figure 12: Visualization of the high-dimensional output of the DSQ-network: Case 1 in 7.1. (The upper figure: The estimated optimal action values for all states. The lower figures: the estimated action-value function for four states which are randomly selected from the exploited velocity-position trajectory, after 10^4 state transition steps.)

With the contour maps, Figs.12–15 show the high-dimensional output of DSQ-network in cases 1-4 in 7.1, respectively. In each figure, the upper subfigure shows the estimated optimal action values of all the states, and the four lower subfigures respectively show the estimated action-value function of four states which are randomly selected from the exploited velocity-position trajectories in Figs.8–11. When the heavy-haul freight train encounters a state, ε-greedy policy is used to select an action from all the legal locomotive notch pairs, so the velocity-position trajectories always escape from the blue areas (low values) to the yellow areas (high values). The action-value function provides guidance for selecting locomotive notches at arbitrary states in the state space, rather than the specified states.

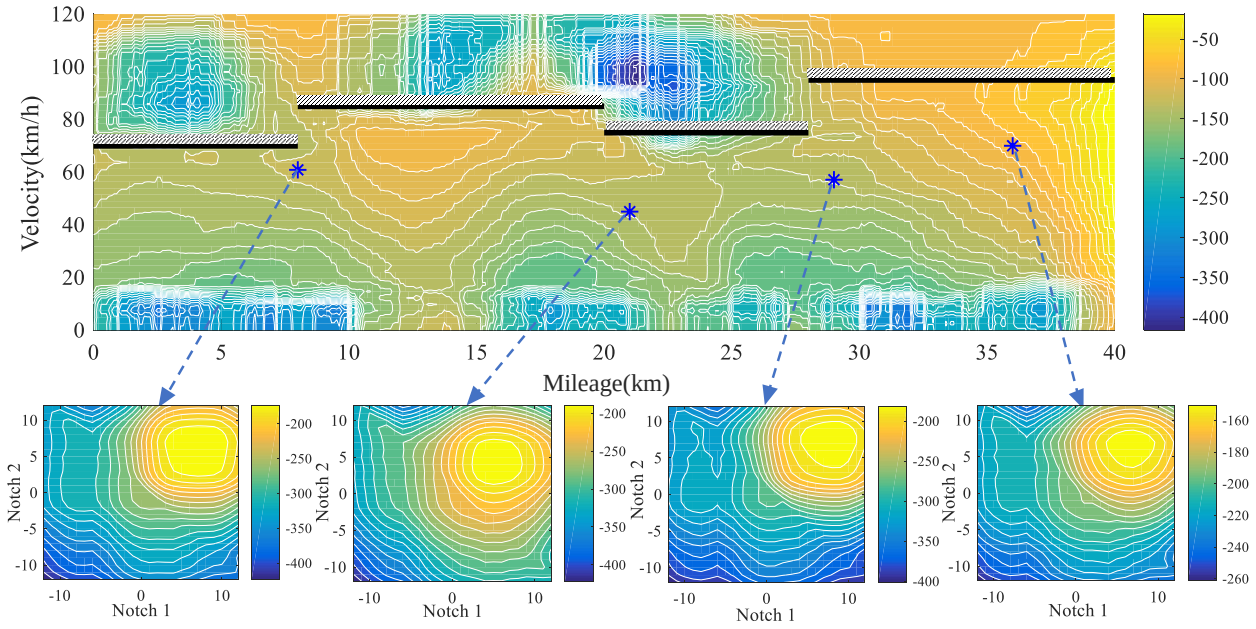

Figure 13: Visualization of the high-dimensional output of the DSQ-network: Case 2 in 7.1. (The upper figure: The estimated optimal action values for all states. The lower figures: the estimated action-value function for four states which are randomly selected from the exploited velocity-position trajectory, after 10^4 state transition steps.)

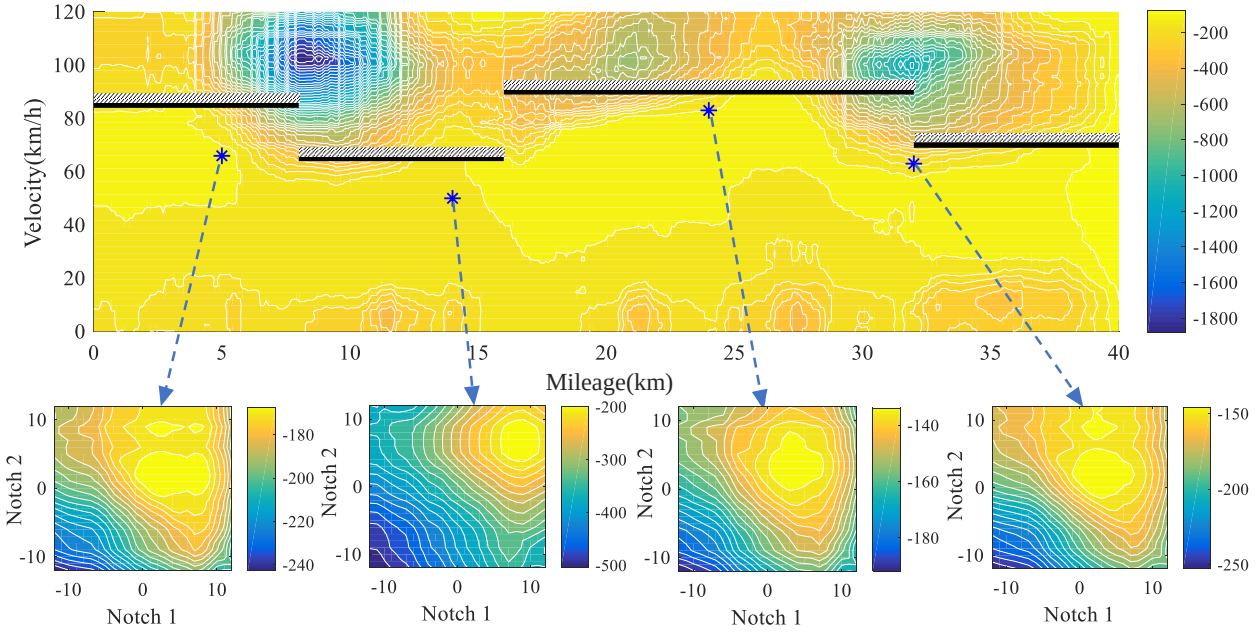

Figure 14: Visualization of the high-dimensional output of the DSQ-network: Case 3 in 7.1. (The upper figure: The estimated optimal action values for all states. The lower figures: the estimated action-value function for four states which are randomly selected from the exploited velocity-position trajectory, after 10^4 state transition steps.)

The ramps and speed restrictions do have influences on the efficiency and accuracy of DSQ-network. DSQ-network shows the most excellent performance in case 2, which can be found from the convergence trajectories of the estimated optimal action values (Fig.7) and the contour maps showing the estimated optimal action values and action-value function (Figs.12–15). In Fig.7, after 2000 state transition steps, the estimated optimal action values in case 2 tend to converge, while those in cases 1,3

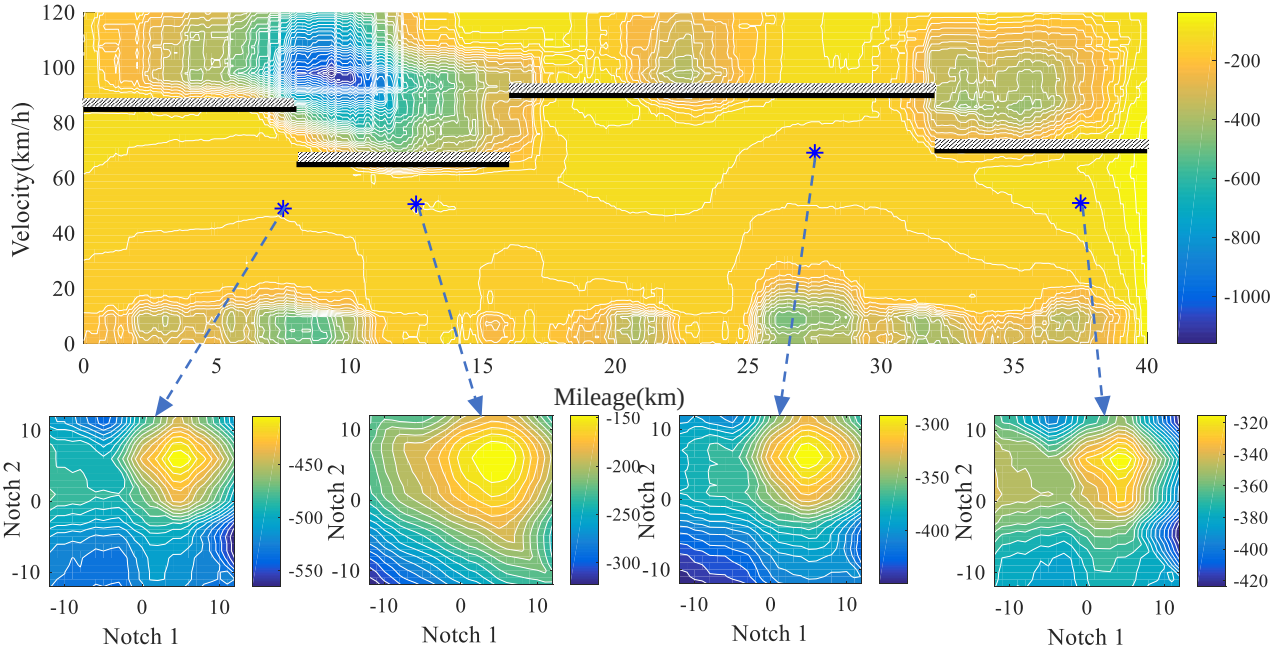

Figure 15: Visualization of the high-dimensional output of the DSQ-network: Case 4 in 7.1. (The upper figure: The estimated optimal action values for all states. The lower figures: the estimated action-value function for four states which are randomly selected from the exploited velocity-position trajectory, after 10^4 state transition steps.)

and 4 still show fluctuations. This is because: firstly, on long uphill sections, the heavy-haul freight train can accelerate only if the both locomotives provide large traction forces. Besides, compared with case 4, the average speed restriction of case 2 is higher, so the agent has less probability to explore the state space beyond the highest permitted velocity due to lack of tractive force, resulting in an increase of the sampled data from the state space below the highest permitted velocity. Secondly, in case 2, optimal locomotive notch pair is located at the corner of the action space, and value of a notch pair decreases monotonously with the increase of its distance from the optimal notch pairs in the action space. Therefore, the optimal notch pairs can be distinguished from the other notch pairs efficiently and accurately. Moreover, from the upper subfigures of Figs.12–15 we can see, the difference between the maximum and minimum of the optimal action values in case 2 is much smaller than those in cases 1,3 and 4. This is because: the agent in cases 1,3 and 4 explores the space beyond the highest permitted velocity more frequently than that in case 2, which produces low-value rewards, resulting in the large difference between the maximum and minimum of the optimal action values.

8.2. Comparison between DSQ-network and DQN

On the one hand, DSQ-network and DQN share some common points: 1) Both methods use the principles of Q-learning to update the action-value function; 2) Both methods use feedforward neural network to approximate the action-value function. On the other hand, the application scenarios of DSQ-network and DQN are different. DSQ-network is designed for RL problems that have continuous state space, large action space, and the influences of adjacent actions on states share similarities. Particularly, tile coding is used to construct the features of actions, so that DSQ-network is able to handle high-dimensional action space. In contrast, DQN is suited for dealing with large and high-dimensional state space, such as image and time series. However, DQN outputs the values of each action independently, so that it is difficult for DQN to learn the relationship between adjacent actions in a high-dimensional action space.

9. Conclusion and further study

This paper has proposed a novel RL approach to achieve the optimal control of multiple electric locomotives in a heavy-haul freight train. A novel nonlinear function approximator, named DSQ-network, has been proposed for fast approximation of the action-value function. DSQ-network can optimize the heavy-haul freight train's performance in velocity, energy consumption and coupler force, without any prior knowledge of train dynamics and pre-designed velocity profiles. Besides, we have discussed the influences of ramps and speed restrictions on the optimal policy for the control of electric locomotives, and the inter-dependent and inter-conditioned relationships between multiple objectives. Moreover, DSQ-network converges much faster, and uses much less storage space than the table-lookup Q-learning when the problem has continuous state space, large action space, and the adjacent actions' influences on states share similarities. Although encoding the state space and action space is common, DSQ-network changes the way of using state and action features, and is extremely suitable for the problem in our paper. DSQ-network can also be used for intelligent control of cars, trucks, and some robots which have large or even continuous action space.

Acknowledgement

This work was supported in part by the National Key Technologies Research and Development Program of China under Grant 2016YFB1200502, and in part by the China Scholarship Council CSC under grants 201707000037 and 201707000036.

References

[1] X. Huang, L. Zhang, M. He, X. You, Q. Zheng, Power electronics used in chinese electric locomotives, in: the 6th IEEE International Power Electronics and Motion Control Conference, 2009, pp. 1196–1200.

[2] P. Gruber, M. M. Bayoumi, Suboptimal control strategies for multilocomotive powered trains, IEEE Trans. Autom. Control 27 (3) (1982) 536–546.

[3] L. Zhang, X. Zhuan, Development of an optimal operation approach in the MPC framework for heavy-haul trains, IEEE Trans. Intell. Transp. Syst. 16 (3) (2015) 1391–1400.

[4] X. Zhuan, X. Xia, Fault-tolerant control of heavy-haul trains, Vehicle Syst. Dyn. 48 (6) (2010) 705–735.

[5] X. Wang, S. Li, T. Tang, X. Wang, J. Xun, Intelligent operation of heavy haul train with data imbalance: A machine learning method, Knowl.-Based Syst. 163 (2019) 36 – 50.

[6] R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction, The MIT Press, 1998.

[7] R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction(Second Edition), The MIT Press, in press.

[8] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.

[9] J. Yin, T. Tang, L. Yang, J. Xun, Y. Huang, Z. Gao, Research and development of automatic train operation for railway transportation systems: A survey, Transport. Res. Part C: Emerg. Technol. 85 (2017) 548 – 572.

[10] G. M. Scheepmaker, R. M. P. Goverde, L. G. Kroon, Review of energy-efficient train control and timetabling, Eur. J. Oper. Res. 257 (2) (2017) 355 – 376.

[11] P. Howlett, Optimal strategies for the control of a train, Automatica 32 (4) (1996) 519–532.

[12] A. Albrecht, P. Howlett, P. Pudney, V. Xuan, P. Zhou, The key principles of optimal train control–part 1: Formulation of the model, strategies of optimal type, evolutionary lines, location of optimal switching points, Transport. Res. Part B: Methodol. 94 (2016) 482–508.

[13] A. Albrecht, P. Howlett, P. Pudney, V. Xuan, P. Zhou, The key principles of optimal train control–part 2: Existence of an optimal strategy, the local energy minimization principle, uniqueness, computational techniques, Transport. Res. Part B: Methodol. 94 (2016) 509–538.

[14] E. Khmelnitsky, On an optimal control problem of train operation, IEEE Trans. Autom. Control 45 (7) (2000) 1257–1266.

[15] X. Lin, Q. Wang, P. Wang, P. Sun, X. Feng, The energy-efficient operation problem of a freight train considering long-distance steep downhill sections, Energies 10 (6) (2017) 794–819.

[16] C. Sicre, A. P. Cucala, A. Fernández-Cardador, P. Lukaszewicz, Modeling and optimizing energy-efficient manual driving on high-speed lines, IEEJ Trans. Electr. Electron. Eng. 7 (6) (2012) 633–640.

[17] S. Açıkbaş, M. T. Söylemez, Coasting point optimisation for mass rail transit lines using artificial neural networks and genetic algorithms, IET Electr. Power Appl. 2 (3) (2008) 172–182.

[18] K. K. Wong, T. K. Ho, Coast control of train movement with genetic algorithm, in: Congress on Evolutionary Computation, Canberra, Australia, 2003, pp. 1280–1287.

[19] P. Wang, R. M. P. Goverde, Multiple-phase train trajectory optimization with signalling and operational constraints, Transport. Res. Part C: Emerg. Technol. 69 (2016) 255–275.

[20] Y. Wang, B. D. Schutter, T. J. J. van den Boom, B. Ning, Optimal trajectory planning for trains–a pseudospectral method and a mixed integer linear programming approach, Transport. Res. Part C: Emerg. Technol. 29 (2013) 97–114.

[21] H. Ko, T. Koseki, M. Miyatake, Application of dynamic programming to the optimization of the running profile of a train, Computers in Railways 74 (2004) 103–112.

[22] S. Lu, S. Hillmansen, T. K. Ho, C. Roberts, Single-train trajectory optimization, IEEE Trans. Intell. Transp. Syst. 14 (2) (2013) 743–750.

[23] V. K. Garg, R. V. Dukkipati, Dynamics of railway vehicle systems, Toronto: Academic Press, 1984.

[24] X. Zhuan, X. Xia, Optimal scheduling and control of heavy haul trains equipped with electronically controlled pneumatic braking systems, IEEE Trans. Control Syst. Technol. 15 (6) (2007) 1159–1166.

[25] M. Chou, X. Xia, Optimal cruise control of heavy-haul trains equipped with electronically controlled pneumatic brake systems, Control Eng. Pract. 15 (5) (2007) 511–519.

[26] X. Zhuan, X. Xia, Cruise control scheduling of heavy haul trains, IEEE Trans. Control Syst. Technol. 14 (4) (2006) 757–766.

[27] X. Zhuan, X. Xia, Speed regulation with measured output feedback in the control of heavy haul trains, Automatica 44 (1) (2008) 242–247.

[28] L. Zhang, X. Zhuan, Optimal operation of heavy-haul trains equipped with electronically controlled pneumatic brake systems using model predictive control methodology, IEEE Trans. Control Syst. Technol. 22 (1) (2014) 13–22.

[29] Q. Song, Y.-D. Song, Data-based fault-tolerant control of high-speed trains with traction/braking notch nonlinearities and actuator failures, IEEE Trans. Neural Netw. 22 (12) (2011) 2250–2261.

[30] S. Gao, H. Dong, B. Ning, X. Sun, Neural adaptive control for uncertain MIMO systems with constrained input via intercepted adaptation and single learning parameter approach, Nonlinear Dynamics 82 (3) (2015) 1109–1126.

[31] S. Gao, H. Dong, Y. Chen, B. Ning, G. Chen, X. Yang, Approximation-based robust adaptive automatic train control: An approach for actuator saturation, IEEE Trans. Intell. Transport. Syst. 14 (4) (2013) 1733–1742.

[32] S. Li, L. Yang, K. Li, Z. Gao, Robust sampled-data cruise control scheduling of high speed train, Transport. Res. Part C: Emerg. Technol. 46 (2014) 274–283.

[33] S. Li, L. Yang, Z. Gao, K. Li, Optimal guaranteed cost cruise control for high-speed train movement, IEEE Trans. Intell. Transport. Syst. 17 (10) (2016) 2879–2887.

[34] H. Tang, X. Ge, Q. Liu, Q. Wang, Robust H∞ control of high-speed trains with parameter uncertainties and unpredictable time-varying delays, in: Proceedings of the 35th Chinese Control Conference (CCC), 2016, pp. 10173–10178.

[35] H. Tang, Q. Wang, X. Feng, Robust stochastic control for high-speed trains with nonlinearity, parametric uncertainty, and multiple time-varying delays, IEEE Trans. Intell. Transp. Syst. 19 (4) (2018) 1027–1037.

[36] Z. Li, C. Yin, S. Jin, Z. Hou, Iterative learning control based automatic train operation with iteration-varying parameter, in: Proceedings of the 10th IEEE International Conference on Control and Automation (ICCA), 2013, pp. 51–56.

[37] L. Fan, Iterative learning and adaptive fault-tolerant control with application to high-speed trains under unknown speed delays and control input saturations, IET Control Theory and Applications 8 (9) (2014) 675–687.

[38] H. Ji, Z. Hou, R. Zhang, Adaptive iterative learning control for high-speed trains with unknown speed delays and input saturations, IEEE Trans. Autom. Sci. Eng. 13 (1) (2016) 260–273.

[39] K. Sun, S. Mou, J. Qiu, W. Tong, H. Gao, Adaptive fuzzy control for nontriangular structural stochastic switched nonlinear systems with full state constraints, IEEE Trans. Fuzzy Syst. 27 (8) (2019) 1587–1601.

[40] J. Qiu, K. Sun, T. Wang, H. Gao, Observer-based fuzzy adaptive event-triggered control for pure-feedback nonlinear systems with prescribed performance, IEEE Trans. Fuzzy Syst.doi:10.1109/TFUZZ.2019.2895560.

[41] J. Yin, D. Chen, L. Li, Intelligent train operation algorithms for subway by expert system and reinforcement learning, IEEE Trans. Intell. Transport. Syst. 15 (6) (2014) 2561–2571.

[42] J. Yin, D. Chen, Y. Li, Smart train operation algorithms based on expert knowledge and ensemble cart for the electric locomotive, Knowl.-Based Syst. 92 (2016) 78–91.

[43] X. Wang, T. Tang, H. He, Optimal control of heavy haul train based on approximate dynamic programming, Adv. Mech. Eng. 9 (4) (2017) 1–15.

[44] F. Zhu, S. V. Ukkusuri, Accounting for dynamic speed limit control in a stochastic traffic environment: A reinforcement learning approach, Transport. Res. Part C: Emerg. Technol. 41 (2014) 30–47.

[45] D. Šemrov, R. Marsetič, M. Žura, L. Todorovski, A. Srdic, Reinforcement learning approach for train rescheduling on a single-track railway, Transport. Res. Part B: Methodol. 86 (2016) 250–267.

[46] P. Balakrishna, R. Ganesan, L. Sherry, Accuracy of reinforcement learning algorithms for predicting aircraft taxi-out times: A case-study of tampa bay departures, Transport. Res. Part C: Emerg. Technol. 18 (6) (2010) 950–962.

[47] G. Tesauro, TD-Gammon, a Self-Teaching Backgammon Program, Achieves Master-Level Play, MIT Press, 1994.

[48] H. V. Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double Q-learning, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16), 2016, pp. 2094–2100.

[49] Z. Wang, T. Schaul, M. Hessel, H. V. Hasselt, M. Lanctot, N. D. Freitas, Dueling network architectures for deep reinforcement learning, in: Proceedings of the 33rd International Conference on Machine Learning, 2015, pp. 1995–2003.

[50] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. V. D. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.

[51] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. baker, M. Lai, A. bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. V. D. Driessche, T. Graepel, D. Hassabis, Mastering the game of go without human knowledge, Nature 550 (7676) (2017) 354–359.

[52] J. Xuan, J. Lu, Z. Yan, G. Zhang, Bayesian deep reinforcement learning via deep kernel learning, Int. J. Comput. Intell. Syst. 12 (2018) 164–171.

[53] G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, T. Mann, T. Weber, T. Degris, B. Coppin, Deep reinforcement learning in large discrete action spaces, arXiv:1512.07679 [cs.AI].

[54] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, Deterministic policy gradient algorithms, in: Proceedings of the 31st International Conference on Machine Learning (ICML), 2014, pp. 1–9.

[55] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, in: Proceedings of 4th International Conference on Learning Representations (ICLR), 2016, pp. 1–14.

[56] R. S. Sutton, Generalization in reinforcement learning: Successful examples using sparse coarse coding, in: International Conference on Neural Information Processing Systems, 1995, pp. 1038–1044.

[57] A. A. Sherstov, P. Stone, Function approximation via tile coding: Automating parameter choice, in: International Symposium on Abstraction, Reformulation and Approximation (SARA-05), Berlin, Heidelberg, 2005, pp. 194–205.

[58] W. T. B. Uther, M. M. Veloso, Tree based discretization for continuous state space reinforcement learning, in: Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence, 1998, pp. 769–774.

[59] M. Chou, X. Xia, C. Kayser, Modelling and model validation of heavy-haul trains equipped with electronically controlled pneumatic brake systems, Control Eng. Pract. 15 (4) (2007) 501–509.

[60] C. J. C. H. Watkins, Learning from Delayed Rewards, Ph.D. thesis, Cambridge University, 1989.