Algorithm of Automatic Train Operation System Based on Reinforcement Learning

Yanmei Guo[1], Ziheng Wu[1], Xiangxian Chen[1], Zhujun Ling[2]
[1]Department of Instrumentation Science and Engineering, Zhejiang University, Hangzhou, China
[2]Zhejiang Train Intelligent Engineering Technology Research Center Co., Ltd

Abstract. Achieving good speed control is the key to the smooth operation of the automatic driving system. In this paper, the image processing method in deep learning is introduced, combined with the classic deep neural network algorithm, and the CNN is strengthened through the reinforcement learning method. A speed controller with good tracking effect is designed. This method enhances the universality of the fuzzy control method, meanwhile, it creatively puts forward the idea of using the combination of neural network and reinforcement learning to achieve a good effect of automatic train operation control.

Keywords: ATO control, reinforcement learning, CNN network

1. Introduction

ATO (automatic train operation) is an important part of the control of train operation. Its primary goal is to adjust the tractive force and braking force of train in real time according to the different operating environments of train so as to realize the safe, reliable and efficient operation of train in accordance with predetermined instructions. At the same time, a good automatic train operation system can also reduce energy consumption, achieve high-precision speed and position control, and improve on-time train punctuality and comfort [1]. The most important thing about ATO systems is speed tracking.

The construction ideas of ATO system are generally established by the train model, simulation verification, control algorithms designed according to demand. Among them, the accurate mathematical description of the running process of a train is the basis for realizing an ATO system with high efficiency and good performance [2]. The train speed, the interaction between train and catenary, wheel-rail, air and others are significantly aggravated. And the dynamic environment of train operation system is obviously enhanced. The time-varying nonlinearity of train operation process is more obvious. And therefore, model is difficult to apply to different operating occasions. In the future, it may even be possible to have complex train models that depart from simple parametric methods.

At present, the research of automatic train control algorithm is quite common. The classic control algorithm represented by PID cannot achieve the ideal control effect when the train operating conditions change. For the time-varying train model, the control algorithm lacks flexibility and cannot satisfy our demand. The train operation control method based on fuzzy control has no requirement for the mathematical model of the controlled object and is able to solve the order or parameter requirements of the traditional control algorithm to a certain extent, however, the control accuracy depends on design of the membership function and language variables. Existing solutions do not have precise theoretical guidance in this section, and usually rely on expert experience. As a result, their generalization ability to different train models don't work well.

[1] Yanmei Guo. Tel.: + 86 17367078393.
E-mail address: gym12840@163.com

The point of this paper lies in combining with the method of deep reinforcement learning, trying to put forward a solution to enhance the generalization ability of fuzzy control thought. Through deep reinforcement learning, the fuzzy control scheme is automatically generated for the existing train model, the required target speed curve and the demand strategy to replace the design of traditional expert experience and enhance the universality of fuzzy control. In this process, the control method of this paper introduces the hot image representation method in deep learning to describe our state space, and uses the classical CNN algorithm to get the corresponding output states, and then strengthens the CNN network through the reinforcement learning method, and finally gets the best output based on the strategy we designed. The method breaks away from the traditional fuzzy control based on the projection method designed of language variables and membership function, and the image representation of the processing methods expand the input state space. Finally, we enhance the generalization ability of the fuzzy control, thereby enhancing the characterization ability.

2. Design of train simulator

Train is the control object of ATO system. The controlled input information is traction command and braking command. According to Newton's law of mechanics, train operation equation of motion can be established as follow:

$$\begin{cases} \frac{dv}{dt} = cF \\ F = u - w \\ w = \alpha_0 + \alpha_1 v + \alpha_2 v^2 \end{cases} \quad (1)$$

Among them, v is the running speed of the train; t is the time; c is the acceleration coefficient; F is the resultant force of the train; u is the traction / braking force of the train; w is the resistance to the train; α_0 is the rolling resistance coefficient and the additional resistance ;α_1 is the other mechanical resistance coefficient; α_2 is the external air resistance coefficient.

The train operation model can be represented by the following difference equation:

$$\mathrm{y(k)} = a_1(k) \times y(k-1)^2 + a_2(k) \times y(k-1) + b_1(k) \times u(k-1) + d_0(k) + \delta(k) \quad (2)$$

Among them, y(k) represents the measured value of the running speed of the train; $u(k)$ represents the input of the train running system, that is, the target traction / braking force; $\delta(k)$) is the noise; $a_1(k), a_2(k), b_1(k), d_0(k)$ are the time-varying parameter of train model.

3. Design of train target curve

In order to ensure the smooth operation of the train, we need to design a train performance curve with good performance so that the controlled train can track the target curve accurately and timely.

The target curve generally adopts traction-idler-braking mode conversion mode. In order to ensure the smooth running of the train, the target curve should be as smooth as possible. Therefore, this paper uses the quadratic curve to design the traction curve, braking curve and constant curve.

4. Reinforcement learning and train control principle

The study of introducing reinforcement learning into train control systems relies mainly on the following two points:

A. Train Operation System is a complex time-varying system. Due to the high complexity, non-linearity, basic resistance and additional resistance of train operation, many traditional control methods based on mathematical models of high-precision numerical solutions cannot

work well. Therefore, the current train control is still under the supervision and scheduling of the conductor. The control method based on the experience of the conductor is essentially the thought of fuzzy control.

B. According to the sampling speed and traction / braking force data of Hangzhou Metro Line 4, the tractive / braking force of the train is not infinite accuracy, but distributed in a fixed discrete value. That is, the existing train speed control can rely on increment of tractive/ braking force with limited precision. Therefore, it should be feasible to select the discrete control output (state space) to complete the control of the real train power system.

As summarized in B, a real-world train system outputs discrete control variables by combining the conductor's experience and the real-time status which is the current status and the established target (eg following the target curve, fixed parking and so on). The role played by the conductor is an intelligent control system, and it is also the agent in reinforcement learning that we introduce. Through the train simulator designed in 2 and 3, the established target speed and the reinforcement learning algorithm, we hope that the algorithm we have studied and improved can automatically generate agents that can handle this objective autonomously in many iterations to replace the conductor control of train.

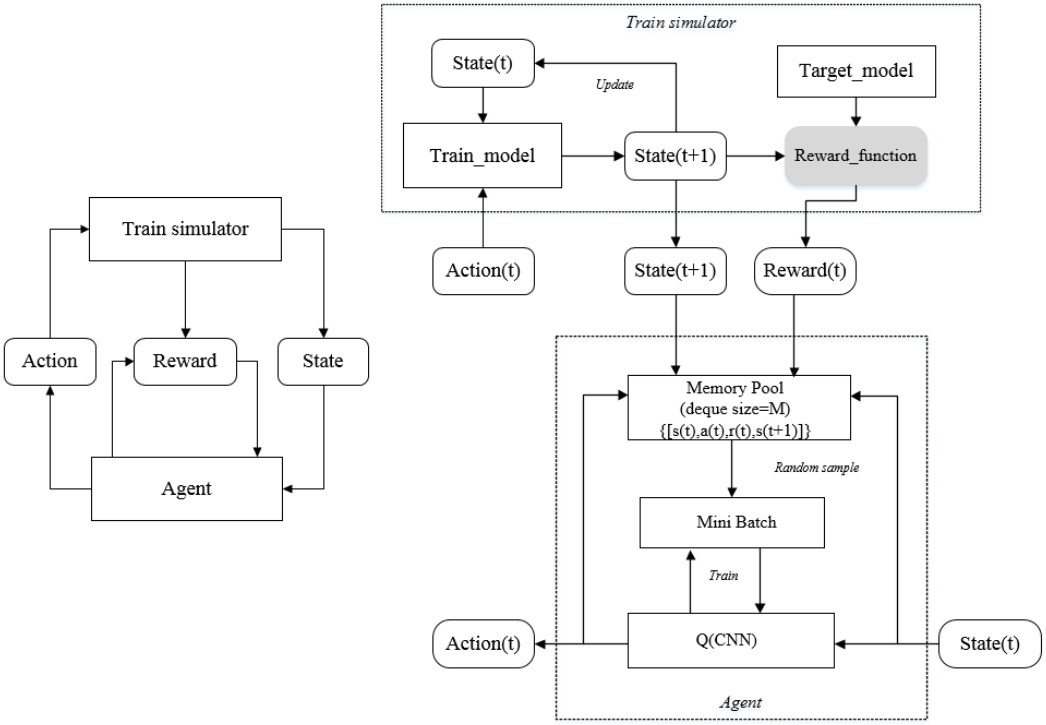

Fig.1: Flow chart of reinforcement learning system

We built the flow chart of reinforcement learning system, as shown in Figure 1. We need to give the system a target and a strategy (target model & reward function), an environment simulator (train model), and finally get an algorithm that can improve its decisions making ability by making our agent interact with the environment and gain rewards. By iteratively iterating the above process, the algorithm allows rewards to reach our setting value [3-4]. In the whole process, the agent doesn't know the specific parameters or mathematical expression of the train model, and the specific reward function. It only interacts with the input and output of the train simulator to complete the learning process, which is the generation process of control algorithms. This means that our algorithm is model free and strategy free, and we can choose the model and strategy for input to generate the control algorithm. This point has great potential,

which means that in the future we can construct more complex, higher-order or fuzzy neural network models that are difficult to control by traditional methods to simulate trains so as to obtain better generalized train model expressions or develop end-to-end strategy based on our needs.

The above part details the flow chart of the whole reinforcement learning algorithm and the construction of the train simulator. In the following, we will introduce how agent can update itself through the sequence of {[state, reward, action]}.

Through the analysis in A and the observation in B, we can abstract the train operation into a Finite-state Markov process. Each step can select an action in a finite set of actions. After the train system accepts the action, the state transition occurs. At the same time, an evaluation R is given and jumps to the state S_{t+1}.

The goal that a reinforcement learning system needs to achieve is to decide an optimal strategy to select the corresponding output state for a given input state so that the total expected value of discount rewards will be maximized in the future when the entire decision-making process is completed.

Under the action of strategy π, the value of state S_t is:

$$Q^{\pi}(S_t) = \mathrm{R}\big(\pi(S_t)\big) + \gamma \sum_{S_{t+1} \in S} P[S_t, A_t, S_{t+1}] Q^{\pi}(S_{t+1}) \tag{3}$$

Where γ is the attenuation coefficient, which indicates the expected return of the current state which discounted by the expected profit of the new state after the completion of the state transition.

The dynamic programming theory guarantees that at least one strategy π^* makes the following formula true[5]:

$$Q^{\pi^*}(S_t) = \max_{A_t \in A} \{ \mathrm{R}\big(\pi(S_t)\big) + \gamma \sum_{S_{t+1} \in S} P[S_t, A_t, S_{t+1}] Q^{\pi^*}(S_{t+1}) \} \tag{4}$$

This lemma makes our selection of the best strategy equivalent to the solution of the best state function $Q^{\pi}(S_t)$. Once we determine the representation of the best state function, the best strategy is $A_t = \operatorname{argmax}_{A_t \in A} \{ Q(S_{t+1} | S_t, A_t) \}$.

Here we introduce the Q-Learning algorithm which is not to estimate the environment model, but directly optimize an iterative Q function. Q-Learning algorithm proposed a method to update the Q value [6]:

$$\mathrm{Q}(S_t, A_t) \leftarrow \mathrm{Q}(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_{A_{t+1}} Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)) \tag{5}$$

The algorithm is proved to converge to the optimal state function we needed– $Q^{\pi}(S_t)$ in an iteration. Therefore, we design the Agent to implement the above algorithm, and in theory we can get the best control strategy in train operation. Among them, $\mathrm{R}\big(\pi(S_t)\big)$ is the reward_function in the flow chart. In simple problems such as finite state problems, for example, maze maneuvers (eg, the input state $S_t$belongs to countable state space), the algorithm uses tables to store each state S_t and the Q value owned by each of the behaviors A_t in this state S_t. The whole algorithm keeps constantly updating the value of Q table, but the running status of the train is often expressed by continuous values (for example, v, u, target (v) Etc.),in which the state space cannot be counted, and a simple method of table updating is difficult to actual operation.

Here we combine the idea of DQN [7] to abstract the train state into the image input, and then introduce the CNN to complete the estimation of Q value of the image input state. Then self-learning and backtracking of the network are realized by memory pool algorithm and experience playback algorithm. Finally, the network can output results close to the best Q value.

5. Experiments

5.1. Design section

5.1.1 Image description

The following three sections describe how we can run an iterative algorithm of how we abstract the train control problem into a flowchart (Figure 1). The strategy we use to abstract the train input state into an image is shown in Figure 2 below.

Image height	*80*
Image width	*80*
Pixel _ t	*0.05*
Pixel _ v	*0.05*
Crash _ limit	*0.3*

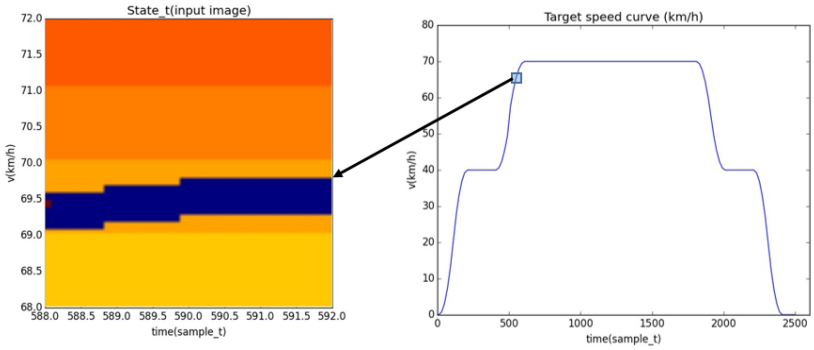

Fig.2: Image of Input State

(1) Parameter list: The image S_t is generated from the parameter list as follows: {height = 80; width = 80; $pixel_t$ = 0.05 (unit: number of samples); $pixel_v$ = 0.05 (unit: number of samples); crash_limit = 0.3km / h}. The image S_t describes the target speed target_v between the sample point t and the sampling point (t + width $\times pixel_t$). The target speed control error interval is 0.3km / h.

(2) Rendering method: The horizontal axis of the image is the sampling point (t ~ t + width $\times pixel_t$) and the vertical axis is the velocity ((v_{min}-1 ~ v_{min} + height $\times pixel_v$). The rendering of upper and lower error interval of target speed- crash_limit/$pixel_v$is 0(pixel value is 0), indicating that the current train speed region, and the rendering of the rest of the image is the speed value of the point. The train particle rendering is 255 (2 * 2 small squares, pixel value is 255).

According to the above description, we will render the target speed and the actual speed of the arbitrary sampling point to the image at any time. The precision is controlled by parameter adjustment of $pixel_t, pixel_v$. Our goal is to get the corresponding output based on the input image, so that the train can always run within the feasible speed area.

5.1.2 Neural network construction

For our output state space, we abstract the output state space into the space described by $\{max_{action}, K_U\}$, as described in Part B of Section 4. Our deep neural network design is shown in Figure 3. For any input abstracted as the state image S_t, we pass three convolutional layers, a pooling layer, two fully connected layers, and finally get an output of length max_{action}, each of which describes the estimated Q value corresponding action A_t in this state $S_t$form the network, and select the maximum Q value corresponding to the action as the output.

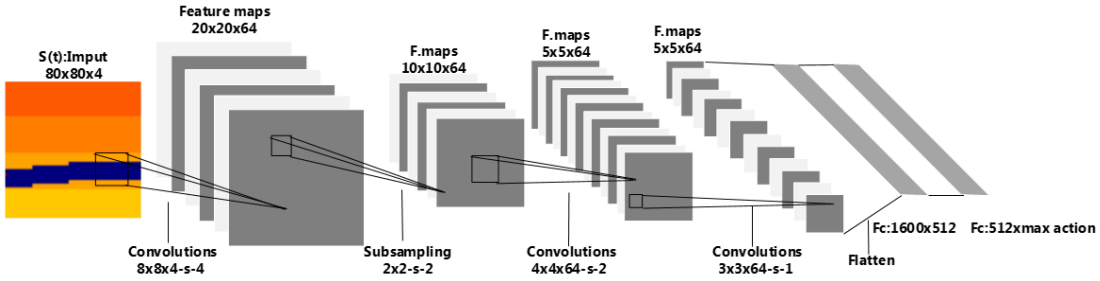

Fig.3:Convolution Neural Network

5.1.3Design of reward system of reinforcement learning

We use reward function to describe the reward system of reinforcement learning. Reward function reflects the mapping of actual demand to control volume. For any input state S_t and action A_t, we can get the next state S_{t+1} from the train simulator. The criteria for judging the behavior A_t is as follows:

a. If the train is still within the target speed limit error in stateS_{t+1}, reward is marked as a_1 (eg, a_1 = 0.2).

b. If the train deviates from the target speed limit error range in the state S_{t+1}, reward is denoted as a_2 (eg, a_2 = -10).

c. If the train is still within the target speed limit error in stateS_{t+1} but its offset is shortened, reward is marked as a_3 (eg, a_3 = 0).

d. If the train is still within the target speed limit error in S_{t+1} and the train speed is getting closer to the target speed curve, reward is written as a_4 (eg, a_4 = 1.2-10 × abs (dv)).

We have formed a variety of reward strategies based on different combinations of parameters $\{a_1, a_2, a_3, a_4\}$. In the following, several representative strategies are selected for comparison.

5.2. Experimental results

According to the description of the above design, we can see that different reward parameter combinations $\{a_1, a_2, a_3, a_4\}$ will form different reward strategies, and we name them strategy A, B, C, D, E, F, etc., as shown in Table 1. Different action parameter combinations $\{max_{action}, K_U\}$ will also form different action strategies, we name them strategy a, b, c, etc., as shown in Table 2. Our strategy is combined with reward strategy and action strategy, as the form of A-a, A-b, B-c, D-a and so on.

Table 1： Reward strategies formed under different combinations of parameters $\{a_1, a_2, a_3, a_4\}$

Parameter	a_1	a_2	a_3	a_4
Strategy A	0.2	-2	0	0.1
Strategy B	0.2	-2	0	1.2-10*abs(dv) (when dv<0.1)
Strategy C	0.2	-5	0	1.2-10*abs(dv) (when dv<0.1)
Strategy D	0.2	-10	0	1.2-10*abs(dv) (when dv<0.1)
Strategy E	0.2	-20	0	1.2-10*abs(dv) (when dv<0.1)
Strategy F	0.2	-5	0	2.2-20*abs(dv) (when dv<0.1)

Table 2： Action strategies formed under the different parameters combination $\{max_{action}, K_U\}$

Parameter	$\max_{action}$	K_U	S
Strategy a	5	5	[-10,-5,0,5,10]
Strategy b	3	10	[-10,0,10]
Strategy c	7	3	[-9,-6,-3,0,3,6,9]

When the train speed is beyond the acceptable control error range, we will restart the train running status and select a certain sampling point in the train model as a new initial point randomly, and compare the target speed and target thrust with a certain offset as the initial state, which is:

$$\begin{cases} v \pm m(m\ represents\ the\ deviation\ from\ the\ target\ speed) \\ u \pm n(n\ represents\ the\ deviation\ from\ the\ target\ thrust) \end{cases}$$

We select reward strategy A and action strategy a, b, c to make three combinations. We get a different average length under different initial conditions for training. The impact of initial conditions on different strategies is shown in Table 3.

Table 3：Compare of the average length on different action strategies and the same reward strategy

the initial state	Strategy A&a	Strategy A&b	Strategy A&c
$\{v \pm 0.3, u \pm 1\}$	186	45	117
$\{v \pm 0.3, u \pm 10\}$	135	35	116
$\{v \pm 0.1, u \pm 1\}$	250	70	119
$\{v \pm 0.1, u \pm 5\}$	204	52	151

It can be seen from the table that the steps obtained by the same strategy are different in different initial states of train, and the steps obtained by different strategies are different in the same initial state. When the speed deviates from a certain distance, the smaller the deviating distance of the thrust, the larger the step size it obtains. However, when the thrust deviates from a certain distance, the smaller the velocity deviation distance, the larger the obtained step size. It can also be observed that the decrease in speed offset distance (eg, from 0.3 to 0.1) increases the step size by a greater amount than the decrease in thrust offset distance (eg, from 10 to 1), so the effect of speed deviation is greater than the thrust deviation.

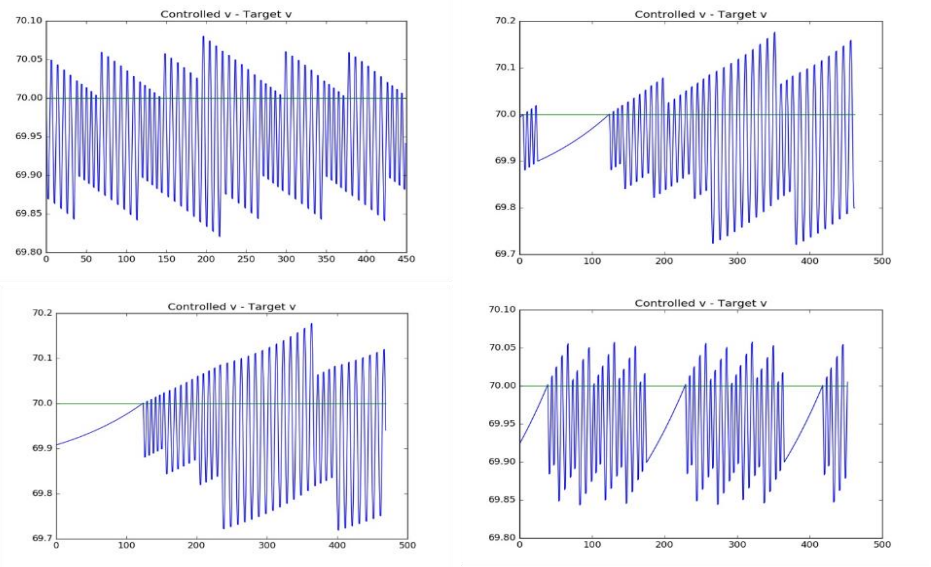

Fig. 4: A group of better performance output state diagram - velocity curves

After our selection of parameters and iterative training, we can choose some better strategies. The velocity curves of controlled by these strategies are shown in Figure 4. In this strategy, the output states perform well. As can be seen from the figure, the speed is different in the initial state, all deviating from the target speed by a certain distance. After multiple operations, the speed returns to the range where is close to the target speed. In the speed section shown in the figure, the maximum fluctuation range is between [69.70km / h, 70.20km / h],and the volatility range is [0,0.43%]. At the same time, we can also see from the figure that in the 500-step range, the speed is controlled by the system within a reasonable error range, and there is no trend of increasing speed error.

To demonstrate the effectiveness of our strategy, we tested the stochastic strategy and the fixed strategy respectively. The experimental results show that the speed of random strategy generation shows the trend of wireless expansion of error within 30 steps, which cannot achieve good speed tracking. We come up with a set of fixed strategy generation maps, as shown in Figure 5. The fixed strategy shown here refers to the traditional expert experience based on the fuzzy control idea. For example, the speed is greater than the target speed, we step on the accelerator, the speed is less than the target curve, and we increase the throttle. It can be seen from the figure below that the speed range remains within the range of [66.0km / h, 71.5km / h] over the 400-step range and the volatility interval is [0,5.7%], which is far beyond the speed fluctuation range under our strategy. At the same time, the speed error shows a divergent trend. It can be predicted that as the step size increases, the speed error will increase.

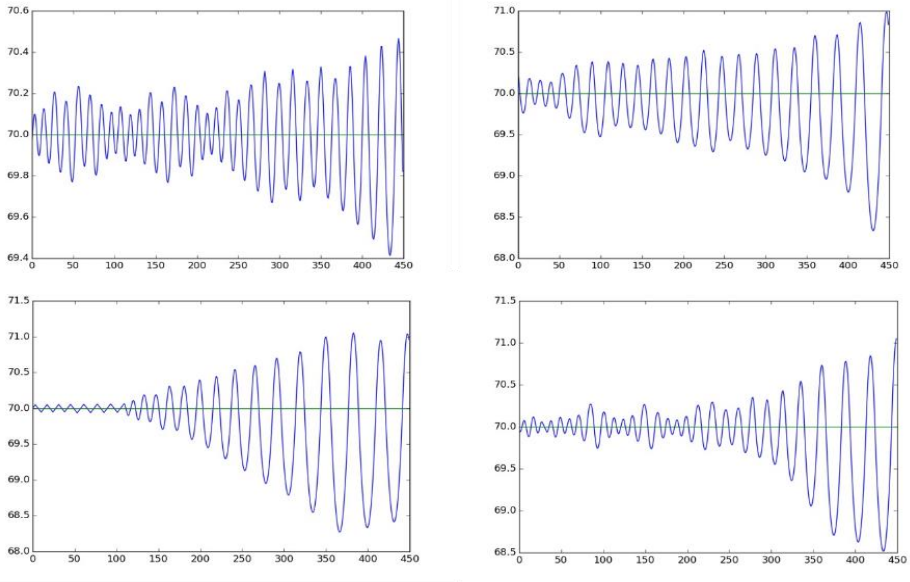

Fig.5: A set of fixed strategy generation graphs - velocity curves

5.3. Experimental Summary

(1)Discrete control is sensitive to the initial conditions, and our strategies work well.

(2) The train model is very unstable. If small deviations cannot be controlled quickly, the model will quickly no longer used or the train operation will collapse.

(3) Our algorithm is still very rough. A lot of parameters and strategies can be adjusted. And we can also choose to test the image of variable speed, which it is not going to be placed in this paper. (Source code can be found in github)

6. Conclusion

In this paper, an automatic train control algorithm based on reinforcement learning is proposed, and a controller with good control effect is designed. The method of this paper gets rid of the traditional train control algorithm based on the precise train model. Combining the idea of fuzzy control with the method of reinforcement learning, the train speed is precisely tracked. At the same time, we also introduced the method of image from the deep learning to describe our state space. Then we use classical CNN algorithm to get the corresponding output state, and strengthen CNN network by reinforcement learning, and finally complete the best Output according to our strategy. As CNN training is very time-consuming, the current network has not yet reached the optimal ideal effect and needs to be improved through continuous training.

7. Acknowledgements

This topic comes from 2015BAG19B03, the 2015 national science and technology support project jointly undertaken by Zhejiang University and Zhejiang Train Intelligent Engineering Technology Research Center Co., Ltd.

8. References

[1] Yang Gang, Liu Mingguang, Yu Le.Nonlinear predictive control of high-speed train operation process [A] .Acta Iron Chrysotment, 2013,35 (8): 16-21

[2] Zhang Kunpeng.Multi-model modeling and predictive control of high-speed EMUs [D] .East China Jiaotong University, 2012.

[3] LI Ning, GAO Yang, LU Xin, et al.A learning agent based on reinforcement learning.Computer Research and Development, 2001,38 (9): 1051 ~ 1056.

[4]L P Kaelbling,M L Littman,A W Moore.Reinforcement Learning:A survey.Journal of Artificial Intelligence Research,1996,4:237~285.

[5] Sutton R S, Barto A G. Reinforcement Learning: An Introduction[J]. Machine Learning, 2005, 16(1):285-286.

[6] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540):529.

[7]Volodymyr Mnih Koray Kavukcuoglu David Silver Alex Graves Ioannis Antonoglou Daan Wierstra Martin Riedmiller. Playing Atari with Deep Reinforcement Learning[J]. Computer Science, 2015.

Authors' background

Your Name	Title*	Research Field	Personal website
Yanmei Guo	Master student	Rail transit control technology	
Ziheng Wu	Master student	Deep learning；machine learning	
Xiangxian Chen	Professor	Rail transit control technology; condition monitoring and signal processing	http://person.zju.edu.cn/xxchen
Zhujun Ling	Project manager	Rail transit technology	