EURASIP Journal on Image and Video Processing

RESEARCH Open Access

Buffer evaluation model and scheduling strategy for video streaming services in 5G-powered drone using machine learning

Yu Su, Shuijie Wang*, Qianqian Cheng and Yuhe Qiu

*Correspondence: wangshuijie@outlook.com
R&D Department II, China Mobile Chengdu Institute of Research and Development, Chengdu 610000, Sichuan, China

Abstract
With regard to video streaming services under wireless networks, how to improve the quality of experience (QoE) has always been a challenging task. Especially after the arrival of the 5G era, more attention has been paid to analyze the experience quality of video streaming in more complex network scenarios (such as 5G-powered drone video transmission). Insufficient buffer in the video stream transmission process will cause the playback to freeze [1]. In order to cope with this defect, this paper proposes a buffer starvation evaluation model based on deep learning and a video stream scheduling model based on reinforcement learning. This approach uses the method of machine learning to extract the correlation between the buffer starvation probability distribution and the traffic load, thereby obtaining the explicit evaluation results of buffer starvation events and a series of resource allocation strategies that optimize long-term QoE. In order to deal with the noise problem caused by the random environment, the model introduces an internal reward mechanism in the scheduling process, so that the agent can fully explore the environment. Experiments have proved that our framework can effectively evaluate and improve the video service quality of 5G-powered UAV.

Keywords: Buffer starvation, Video streaming, 5G-powered drone, Deep learning, Reinforcement learning

1 Introduction

In recent years, the popularity of 5G cellular networks has allowed a variety of mobile devices to communicate with each other and provide different services [1–3]. Among them, the development of 5G-powered UAV is particularly rapid, and real-time video backhaul is an important service that this equipment can provide. Compared to other communication modes, the transmission of video streams does not require minimal end-to-end delay [4–6]. Instead, some key performance indicators used to evaluate the quality of user experience are indispensable. Throughout these indicators, buffer starvation probability and buffer cumulative duration deserve more attention. These two pieces of data quantify the event that the buffer is empty. Insufficient buffer will cause the image to no longer change, that is, the video freezes. Therefore, this defect will have a non-negligible impact on service quality and user behavior in the process of real-time video streaming.

In order to reduce the negative impact of buffer starvation, many experts and scholars are conducting research to evaluate this event and try to find a reasonable start-up delay configuration strategy for the transmission of video streams [7–11]. Nevertheless, there is an unavoidable difficulty in this problem, that is, wireless network video transmission is a random process that controls the arrival of data packets. Thus, it is not comprehensive enough to focus only on video streams of limited size and length. In addition, since the network states of different transmission processes are not geographically homogenous, the fixed start-up delay configuration will no longer be suitable for changing network environments [11–14]. This shows that the prefetch threshold calculation model designed under a given traffic intensity and file size distribution has very large limitations, because this method ignores the uncontrollable impact of network status fluctuations [15–19].

The first purpose of this article is to design an evaluation model based on deep learning to calculate the buffer starvation probability under different transmission scenarios. The deep neural network can extract the characteristics of deep-level correlation, which enables the model to accurately return the buffer starvation distribution through channel information. Thus, the starvation behavior can be effectively evaluated [20, 21]. After obtaining the specific distribution through the evaluation algorithm, we adopt the reinforcement learning method to dynamically configure the start-up delay during the transmission process and the data packet prefetching strategy to achieve the purpose of intelligently scheduling the video stream transmission process [22]. The input of the model is the encoded state, and the output is an actual value of each possible action that is taken. The agent starts to execute the strategy from a given initial state. The initial state can choose the maximum operation or take random exploration operations [23, 24]. The model is trained based on the channel parameter datasets collected by a 5G-powered drone, and is executed on different threads independent of each other according to specific strategies, which can not only effectively solve the problem of insufficient buffer in this scenario, but also improve the quality of video service under different wireless network environments to a certain extent [25].

The main contributions of this paper are:

- We propose a deep neural network that can accurately return the probability distribution of buffer starvation in the video stream transmission process. The calculation results are used in the subsequent video stream scheduling process.
- We propose a reinforcement learning scheduling model that can dynamically allocate start-up delays and calculate packet prefetching strategies. This method can greatly improve the quality of video transmission service.
- Based on the scheduling model, we propose an internal reward mechanism to deal with random environmental noise, which can also reduce the difficulty of training when the model rewards are sparse.
- The robustness of the model has been verified in the actual 5G-powered UAV real-time video transmission process. Experiments have proved that the method performs very well in complex network environments.

The structure of this article is as follows. The second section introduces the related research. The third section describes the regression model based on deep learning in detail. Section 4 presents the internal-driven reinforcement learning model for video streaming scheduling. Section 5 shows the experimental process and results. Section 6 summarizes this work.

2 Related work

2.1 Buffer starvation evaluation

There are existing studies [26] related to the present work. About the evaluation of buffer starvation events, Y. D. Xu and E. Altman et al. uses a method based on Ballot theorem to obtain the buffer starvation probability during fixed-size video transmission [9]. This article proposes a clear solution and summarizes it as the M/D/1 queue. In addition, the authors also derived a recursive method to calculate the distribution of starvation, and extended it to the ON/OFF burst arrival process. For a given start-up threshold, the method provides a fluid model to calculate the starvation probability, and analyzes how the prefetching threshold affects the starvation behavior.

2.2 Video streaming scheduling

As for the transmission scheduling problem, the research on the trade-off between buffer starvation and start-up delay is mainly divided into bandwidth change model and mathematical method [27]. Xu et al. modeled the buffer starvation probability and the starvation event generation function in [10], then dynamically analyzed the starvation behavior at the file level. When the distribution of traffic intensity and file size can be determined, this method can calculate the relationship between the starvation probability and the packet prefetching threshold. Despite the merits of this work, it only considers a single video stream of fixed size and length, so it has great limitation and is not practical. In [5], the authors take into account the needs of network operators and solve three problems: measuring traffic patterns, modeling the probability of buffer starvation and using calculation results for resource allocation. This article takes a method which uses short-term and long-term QoE to balance the overall QoE. At the same time, the authors also introduced a Bayesian inference algorithm, which can make the model have the ability to infer whether the input stream is short-view or long-view.

3 Methods

3.1 Buffer starvation evaluation model

3.1.1 Overview

We propose a packet-level deep learning model to calculate the starvation probability and the distribution of starvation behaviors during video streaming.

Our model is based on a recurrent neural network (RNN) framework, such as the gate recurrent unit (GRU) structure, to extract the correlation between different time series [28, 29]. In the feature selection stage, we use spatial attention mechanism combined with channel attention mechanism. This approach makes the model tend to focus on the information that plays a key role in achieving the goal. That is, we only consider about the state of the network transmission channel that assists in the judgment. In this way, certain parts of the final input are more helpful to decision-making than irrelevant

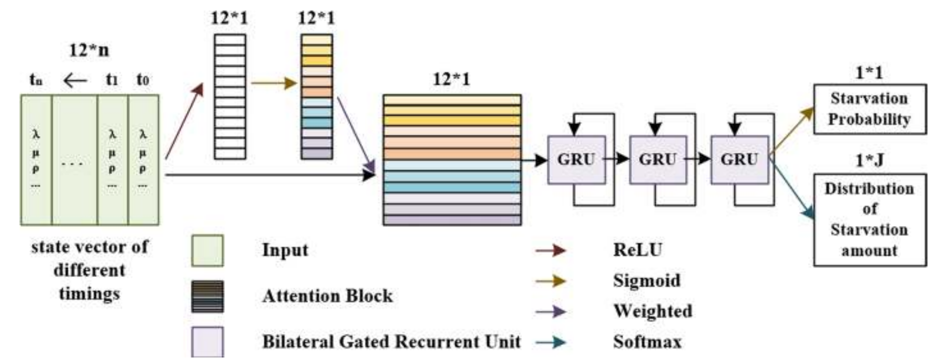

Fig. 1 Structure of the buffer starvation behavior evaluation model

Table 1. The elements contained in the state vector

Notation	Description	Notation	Description
λ	Packet arrival rate	T1	Start-up delay
μ	Packet service rate	d	Duration of a service slot
ρ	Traffic intensity	N	File size in packets
p	Packet arrival probability	Np	Total number of packets
q	Packet departure probability	Nm	Minimum file size
× 1	Start-up threshold in packets	θ	Mean of exponential file size

information that is discarded [30, 31]. In addition, the model uses a multi-task learning structure to share a large part of the weights for prediction, which can effectively reduce the scale of parameters and make the prediction more effective. The network architecture is shown in Fig. 1, where we adopt our backbone with three bilateral gate recurrent unit (BiGRU) blocks [32]. More specific descriptions about this network can be found in the next section.

3.2 System description

The input of the network is the state vector of different timings in the process of data packet transmission. The elements contained in the vector are summarized in Table 1, including Poisson arrival rate, Poisson service rate, traffic intensity, etc.

The model consists of three important parts: attention mechanism module, bilateral gate recurrent unit module (BiGRU) and multi-task learning module, which is shown in Fig. 1. The combination of these three structures can effectively extract the correlation between different elements in the state vector at the same time, and obtain the correlation between the state features of different time series. Meantime, it can also assign greater weight to certain elements that have a greater impact on the final buffer hungry probability, and make full use of different levels of correlation to obtain better results. Finally, the model can accurately predict the starvation probability and the distribution of starvation behaviors.

Because certain parts of the channel state information play a more critical role in the final calculation result, the model uses a spatial attention mechanism. And this attention mechanism is only for the state information at a single moment, so it is also called channel sequence attention. We use the structure of Squeeze-and-Excitation (SE) block to accomplish this task. The first stage of SE block is the compression operation. It achieves feature compression along the dimension of the state sequence, and a real number can be obtained, which has a global receptive field to some extent. The output dimension matches the feature channel and sequence number. This number characterizes the global distribution of responses on characteristic channels and sequences, and enables layers close to the input to obtain global receptive fields. The second stage is the excitation operation, which uses parameters to assign weights for each feature sequence and channel.

The recurrent neural network part uses a three-layer bilateral GRU structure. GRU is a variant of LSTM, but it has only two gates (update and reset). The GRU block can make the training phase easier to converge due to its simple structure, and it can avoid the problem of gradient disappearance to a certain extent. The bilateral GRU block combines the forward GRU and the backward GRU to solve the problem that the one-way structure cannot encode backward-to-forward sequence information, so that it can capture more comprehensive semantic dependence between different channel state sequences.

This module is mainly divided into three parts, the first is the channel information extraction layer, and the second is the channel information representation layer, and the last one is called the buffer starvation behavior prediction layer.

- The channel information extraction layer. After adding the attention mechanism, the information matrix is used as the input for the next step.
- The channel information presentation layer. Considering that both the front and back directions during the video stream transmission process may contain timing information, the bilateral GRU encoding structure is used as the representation of channel information.
- The buffer starvation behavior prediction layer is completed by a fully connected layer of perceptron structure as a prediction of starvation behavior at a certain moment.

After the recurrent network, the model adopts a multi-task structure. That is, for a channel state input, the network simultaneously outputs the buffer starvation probability value and the starvation event distribution. When multiple tasks are forecasting at the same time, a large part of the weight is shared, which reduces the scale of the overall model parameters and makes the forecast more efficient. In addition, the two tasks are highly correlated, which has been verified by experiments to stably improve the accuracy of buffer starvation behavior prediction.

3.3 Buffer starvation behavior loss

The loss function is represented by the sum of two parts

$$\text{Loss} = \alpha \cdot \text{meansquareerror} + \beta \cdot \text{cross} - \text{entropy}, \tag{1}$$

where α, β denote discount factors.

The first part of the formula is the starvation probability loss. The specific calculation is given by

$$\{\backslash text\{mean\}\}\backslash,\{\backslash text\{square\}\}\backslash,\{\backslash text\{error\}\} = \backslash,\backslash left\backslash\left|\left\{Y_{\{S\}} - P_{\{S\}}\right\}\backslash right\backslash\right|_{\{2\}}. \tag{2}$$

In this part, Y_S represents the output value of the model which fits the buffer starvation probability, and P_S is given by the following formula based on the famous Ballot theorem [18]:

$$P_S = \sum\nolimits_{k=x1}^{N-1} \frac{x1}{2k - x1} \binom{2k - x1}{k - x1} p^{k-x1}(1-p)^k. \tag{3}$$

The detailed proof can be found in [10].

During the file transfer process, starvation events may occur multiple times. Given a fixed file size N, the maximum number of starvation events is $J = \lfloor N/x1 \rfloor$, where $\lfloor\cdot\rfloor$ is the lower limit of real numbers. $P_S\left(j\right)_i$ represents the probability of meeting j starvations. Therefore, the second part is the distribution loss of the starvation events, using cross-entropy loss:

$$Cross - entropy = -\sum\nolimits_{i=1}^{n} P_S\left(j\right)_i \cdot \log(\mathrm{Y}_{Di}). \tag{4}$$

The vector $\boldsymbol{Y}_D = (P_S(0), P_S(1), \ldots, P_S(J))$. We let $P_{\varepsilon(k_l)}$, $P_{S_l(k_l)}$,$P_{U_j(k_j)}$ be the probabilities of events 'the buffer becoming empty for the first time in the entire path', 'the empty buffer after the service of packet given that the previous empty buffer happens at the departure of packet k_l and 'the last empty buffer observed after the departure of packet k_j. The calculation process of $P_S\left(j\right)$ is given by

$$P_S\left(j\right) = \sum_{k_1=1}^{N}\sum_{k_2=1}^{N}\cdots\sum_{k_{j-1}=1}^{N}\sum_{k_{j=1}}^{N} P_{\varepsilon(k_1)} \bullet P_{S_1(k_1,k_2)} P_{S_{j-1}(k_{j-1},k_j)} \bullet P_{U_j(k_j)} = P_\varepsilon\left(\prod_{l=1}^{j-1} P_{S_l}\right) P_{U_j}^T, \tag{5}$$

where T denotes the transpose. The detailed analysis can be found in Ref. [10].

3.4 Video streaming scheduling model

3.4.1 Reinforcement learning model environment settings

The purpose of reinforcement learning (RL) algorithm is to train the model so that the agent can complete specific tasks. In order to achieve this goal, it is necessary to abstract the video streaming scheduling problem as an RL problem, which requires the definition of the environment of the RL model. The environment describes the state of a certain task within a specified period of time, a series of actions that can be taken, and the final impact of these actions [33].

The state in the environment is represented by a vector, which mainly describes the network state during video streaming in the wireless network. The elements in the vector include: packet arrival rate, packet service rate, duration of service time slot, start delay, traffic intensity, file size in the packet, packet arrival probability, total number of packets, packet departure probability, The minimum file size, the start threshold in packages, and the average value of the index file size.

The action of the agent is to reconfigure the packet prefetching strategy in each state and reconfigure the start-up delay when buffer starvation occurs. Every action will result in a change of the state. The state is not only determined by these actions, but also related to the current network state. Therefore, the environment has a certain degree of randomness, and the model is a model-free RL approach.

The reward function in the environment assigns an actual value to each possible pair of state and action. In the video stream scheduling problem, we define the reward function to be composed of two parts, which, respectively, represent the current state buffer starvation probability and the expected interval between two adjacent buffer starvations. This reward function R^e is also a quantitative form of QoE:

$$R^e = -(P_s + \gamma\left(g(E(T))\right). \tag{6}$$

P_s is the buffer starvation probability. The calculation of it is based on Ballot theorem.

Ballot theorem: In a ballot, candidate A gets $\mathrm{N_A}$ votes, candidate B gets $\mathrm{N_B}$ votes, where $\mathrm{N_A} > \mathrm{N_B}$. Assuming that all orders are the same when counting votes, the probability that A will always lead in the number of votes during the whole counting process is $(\mathrm{N_A} — \mathrm{N_B})/(\mathrm{N_A} + \mathrm{N_B})$.

After the start of a transmission service, for a given initial queue length $x1$ and total size N, the starvation probability is given by:

$$P_S = \sum_{k=x1}^{N-1} \frac{x1}{2k - x1} \binom{2k - x1}{k - x1} p^{k-x1} (1-p)^k. \tag{7}$$

The detailed proof process can refer to [9].

$E(T)$ is the expected time interval between two starvations. We let $g(\cdot)$ be a strictly mono-increasing and convex function of the expected start-up delay:

$$E(T) = \frac{x_1}{\lambda(1-\rho)}. \tag{8}$$

3.4.2 Internal reward mechanism

In the actual video stream transmission scheduling process, due to the network may produce unpredictable fluctuations at any time, its status cannot be determined [34, 35]. Therefore, a problem arises, that is, random environment and sparse reward (or even almost no reward) may cause the agent to be unable to effectively explore the environment. In response to this problem, we propose an internal reward mechanism. Independent of external reward signals, the mechanism is modeled as the difference between the predicted state and the actual state in the feature space [36]. Meanwhile, a self-supervised inverse dynamic model is used to extract state features from the feature space. When the environment changes, the model can have strong adaptability. The structure of this mechanism is shown in Fig. 2.

Different from the traditional reinforcement learning reward form, the reward signal is divided into two parts. It means the function is rewritten into $R = R^i + R^e$, where R^i represents the internal reward due to the mechanism we proposed, and R^e represents the reward inherent in the environment, which is calculated by Eq. (6). Then we use the strategy learning method to find the corresponding strategy by optimizing accumulated rewards.

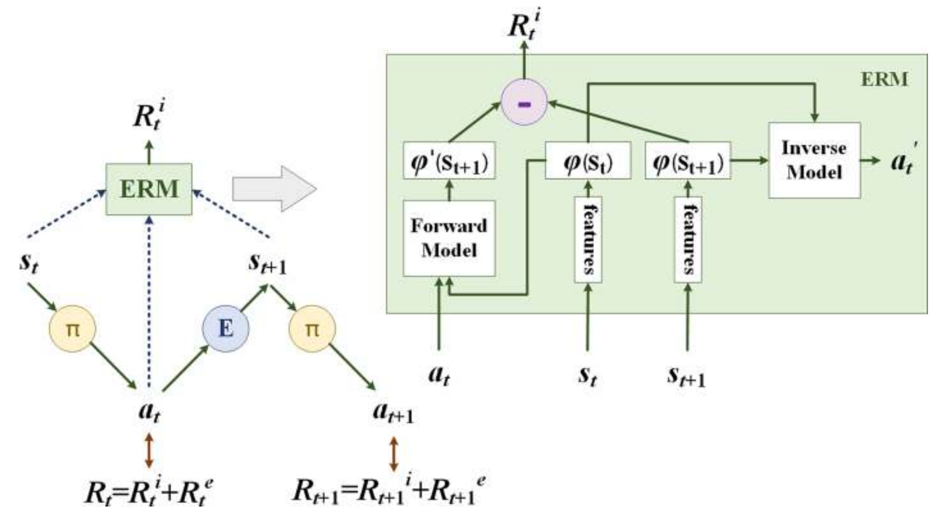

Fig. 2 Structure of the internal reward mechanism (ERM)

The essence of this mechanism is to learn the effective information that actually affects the agent. We use a deep neural network to extract the feature from the state s. Through which we can obtain $\varphi(s)$. Then we use the feature extraction $\varphi(s^{'})$ of the next state to predict the action between these two states:

$$a_{\text{prediction}} = DNN\left(s, s^{'}; \theta_I\right). \tag{9}$$

By minimizing the error between the predicted $a_{\text{prediction}}$ and the actually adopted action a, back propagation is used to allow the neural network to extract the features that are truly influenced by the action. Considering that the action here is discrete, we can take SoftMax function on the predicted action, and then set the corresponding loss function through maximum likelihood estimation, that is

$$\min_{\theta_I} L_1\left(a_{\text{prediction}}, a\right). \tag{10}$$

After obtaining the feature $\varphi(s)$ from the current state, we also use a neural network to predict the feature of the next state $s^{'}$:

$$\backslash varphi\left(\backslash hat\left\{s^{\{\backslash prime\}}\right\}\right)\{\backslash text\{=f\}\}\backslash varphi(\{\backslash text\{s\}\}), \{\backslash text\{a\}\}; \backslash theta\left(_{\{\{\backslash text\{F\}\}\}}\right). \tag{11}$$

Since the predicted feature is a vector, the L_2 norm is used as the loss:

$$L_F\left(\varphi\left(s^{'}\right), (\backslash hat\{s^{\{}\backslash prime\}\})\right) = \frac{1}{2}\left|\varphi\left(s^{'}\right) - \varphi\left(\widehat{s^{'}}\right)\right|_2^2. \tag{12}$$

At the same time, we use the above loss $L_F\left(\varphi\left(s^{'}\right), (\backslash hat\{s^{\{}\backslash prime\}\})\right)$ to calculate the internal reward:

$$R^i = \eta L_F\left(\varphi\left(s^{'}\right), (\backslash hat\{s^{\{}\backslash prime\}\})\right). \tag{13}$$

The learning goal of this model is given by:

$$\mathrm{Min}_{\theta_P\theta_I\theta_F}\left[-\alpha E_{\pi(s_t;\theta_P)}\left[\sum\nolimits_t R_t^e\right]+(1-\beta)L_1+\beta L_F\right], \quad (14)$$

where $\alpha > 1, 0 \leq \beta \leq 1$ is only a measure of the scale of the corresponding item.

In the training phase, after the initial part of the prediction is accurate, in order to obtain more internal rewards, this mechanism will actively explore more unknown states.

3.4.3 Optimal strategy learning

Our model uses a Deep Q network (DQN) method to learn the best strategy. DQN and Q-learning are similar to algorithms based on value iteration [37]. In the ordinary Q-learning method, if the state and action space are high-dimensional continuous, it is hard to use the Q table. Therefore, we convert the Q table update into a function fitting problem and use a deep neural network instead of the Q table fitting function to generate the Q value. DQN uses a neural network to approximate the value function. After obtaining the value function, DQN takes the $\epsilon -$ greedy strategy to select action. The structure of the approach is shown in Fig. 3.

The algorithm has two main structures:

- The experience replay (experience pool) method is used to solve the problem of correlation and non-static distribution.
- A MainNet is introduced to obtain the real-time Q value, and the target Q value is obtained through another TargetNet.

The memory mechanism in the experience pool is used to learn from previous experiences. For the reason that Q-learning is an off-policy learning method, it can learn from the current experience as well as the past experience. Thus, randomly adding previous experience during the learning process will make the deep neural network more efficient. The experience pool stores the transfer samples (s_t, s_t, r_t, s_{t+1}) obtained by the interaction between the agent and the environment at each time step into the playback

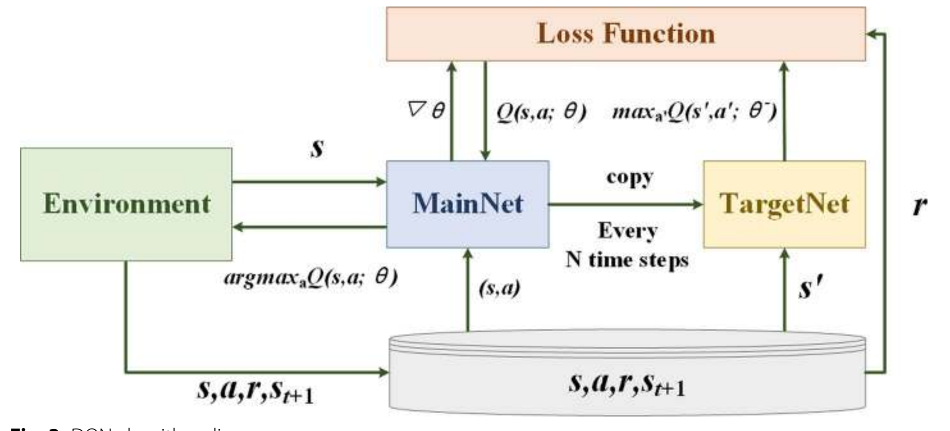

Fig. 3 DQN algorithm diagram

memory network, and randomly takes out some batches to disrupting the correlation while training.

Q-targets is actually an approach to disrupt correlation. It will build two networks with the identical structure. However, they have completely different parameters. The network for predicting Q estimation, MainNet, uses the latest parameters, while the TargetNet parameters of the network that predicts Q reality use a long time ago. $Q(s, a; \theta_i)$ represents the output of the current network MainNet. They were used to measure the value function of the current state action pair. $Q(s, a; \theta_i^-)$ represents the result of TargetNet, which can be used to calculate the target Q and update the MainNet parameters based on the loss function. They can also copy the MainNet parameters to TargetNet after a certain number of iterations.

In the value function network training phase, the environment will give an observation at first; then, the agent will get all the Q values about this observation calculating by the value function network, and use $\epsilon -$ greedy method to select the action and make a decision. After the environment receives this action, it will feedback a reward and the next observation. This is a complete step. At this time, we update the parameters of the value function network according to the reward, then enter the next step. This cycle continues until we have trained a satisfying value function network.

The update of the value function in this algorithm is given by:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[R + \gamma max_{a'} Q\left(s^{'}, a^{'}\right) - Q(s,a)\right]. \tag{15}$$

The loss function of DQN is as follows, θ indicates that the network parameter is the mean square error loss:

$$L(\theta) = E[\left(R + \gamma max_{a'} Q\left(s^{'}, a^{'}; \theta\right) - Q(s,a;\theta)\right)^2]. \tag{16}$$

There are two structures which have nearly the same design but different parameters in DQN. The network MainNet predicts Q estimation value by latest parameters, while the TargetNet predicts Q reality value by parameters of a long time ago. It means when the agent in the model takes action *a* in the RL environment, Q can be calculated according to the above formula and the MainNet parameters can be updated. They will be copied to TargetNet after a certain number of iterations. Thereupon, a learning process is completed.

4 Results and discussion

4.1 Experiment settings

In the experiment, in order to verify that the model is effective and robust in complex 5G NAS network, a 5G-powered drone (DJI M210 UAV) equipped with a communication module (Hubble I) provided by China Mobile was used as the video streaming transmission equipment (as shown in Fig. 4). The trace-driven simulation proves the accuracy of our method.

On the issue of experimental settings, we once considered pure simulation experiments, which generate the required data and variables through random distribution. However, this has another very big flaw. The pure simulation method cannot represent

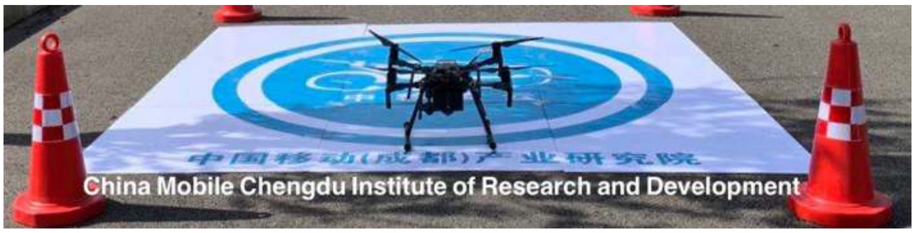

Fig. 4 DJI M210 UAV equipped with a 5G communication module provided by China Mobile Chengdu Institute of Research and Development

the fluctuating network environment, nor can it simulate the noise problem in the random environment. At the same time, when there are enough random events, this approach will converge to a specific mathematical model. However, this runs counter to our purpose. Because even if the pure simulation method can verify the correctness of the model, it is not enough to evaluate the accuracy of the model. In combination with the above considerations, our experiment finally adopts a tracking-driven simulation method, and randomly selects requests in the 5G-powered UAV video streaming service in a real wireless network environment. Such an experimental method can not only effectively test the performance and robustness of the model in actual scenarios, but also artificially control the parameter range to a certain extent for targeted measurement.

The video resolution used in the experiment includes 1080p, 2 k and 4 k, and the video frame rate includes 24fps and 30fps. Since 5G base stations cover 100–300 m, the drone's flying range is within 100 m of the base station. The terrain for the experiment includes above the lake, above the woods, and above the buildings.

For different network environments, the accuracy of the buffer starvation probability evaluation model can reach about 96.3%. Using the channel data generated during the transmission of videos with a total duration of about 100 h for training, the model converges within 22 min. Under the same experimental conditions, the cumulative QoE finally achieved by the reinforcement learning scheduling model is more than 15% higher than that of the existing methods, and the training process converges after about 130,000 episodes.

4.2 Starvation of transmission process

4.2.1 Starvation behavior prediction

For a comprehensive evaluation, video streams of different lengths have been tested up to 8000 times. We use four different parameter settings: $\rho = 0.95$ or 1.25, $\times 1 = 40$ or 60 packets. If not specifically mentioned, the departure rate μ is normalized to 1. The video size in the experiment is between 300 and 9000 packets. Figure 5 shows the probability of 0–4 starvations with the parameters $\rho = 0.95$ and $\times 1 = 40$. As the file size increases, the none starvation probability decreases. We can learn that the probability of multiple starvations first increases and then decreases. Figure 5 also shows that our analysis results are in great agreement with the simulation results. When the starting threshold is 50 packets, Fig. 6 shows similar results. The none starvation probability decreases as the video is longer. However, the probability of multiple starvations increases at first and then decreases. Figure 7 verifies the asymptotic none starvation probability with the

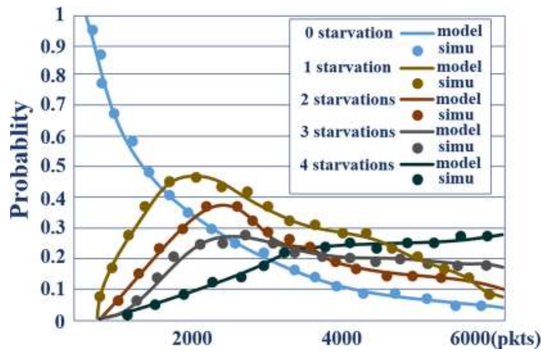

Fig. 5 Probability of 0–4 starvations with × 1 = 40

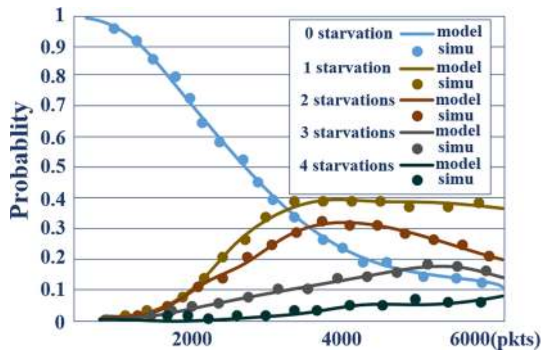

Fig. 6 Probability of 0–4 starvations with × 1 = 60

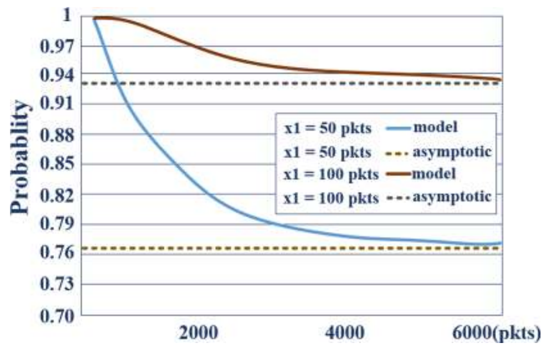

Fig. 7 Asymptotic probability of no starvation

traffic intensity $\rho = 1.25, \times 1 = 50$ and 100. When the video is short and the data packets are small, the total transmission time is relatively short. Thus the network status has not changed much during the whole process, and the probability of buffer starvation is relatively small. That is, there is no starvation most of the time. This can explain why the model line is farther away from the asymptote when the data packets is smaller. Figure 8 plots the asymptotic probability of a single starvation event with the same settings. The probability of one starvation occurring increases as the video file is larger. More prefetching packets will cause smaller starvation probability.

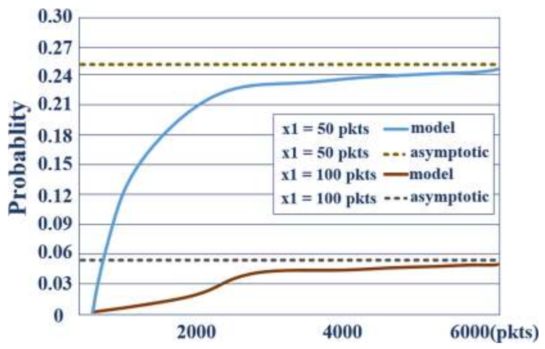

Fig. 8 Asymptotic probability of 1 starvation

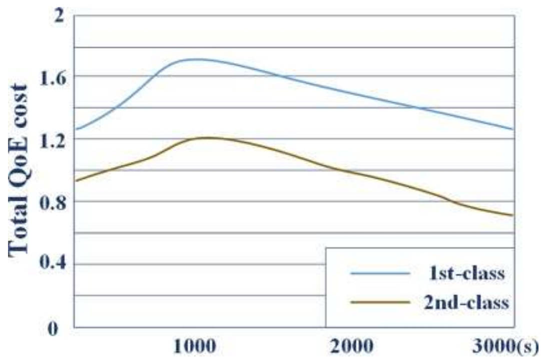

Fig. 9 QoE vs transmission duration (ours)

4.3 Reward metrics and analysis

4.3.1 Cumulative QoE

In order to make the base station under a heavy load but not exceed the capacity area, we finally set the traffic intensity $\rho = 0.98$, while the video request intensity $\lambda = 0.009$. This can also make the attainment rate of the video request within a stable range. In specific experiments, we will evaluate the overall objective quality of experience and the performance of the model in different start-up states of the video transmission process.

First, we divide the video stream into two categories to measure the model, respectively. The Poisson arrival rate of the k-th video request is λ_k. The total arrival rate is $\lambda = \lambda_1 + \lambda_2$. The service time of a video stream in a state is equal to the video request size in bits divided by its throughput. Since viewing time follows a super-exponential distribution, the service time of each category is also exponentially distributed. Here, we use the state pair (m, n) to indicate that there are currently m first-class flows and n second-class flows passing through the bottleneck. As more video streams share bottlenecks, the probability of buffer starvation during the transmission process will increase.

Figure 9 illustrates the change in the objective QoE of the first-class stream and the second-class stream scheduling by our model when the video transmission duration increases from 0 to 3000 s. Figure 10 shows the results of scheduling in the same experimental environment through the method in [5]. It can be found that although our scheduling model is unable to make the QoE index higher than the result obtained by the

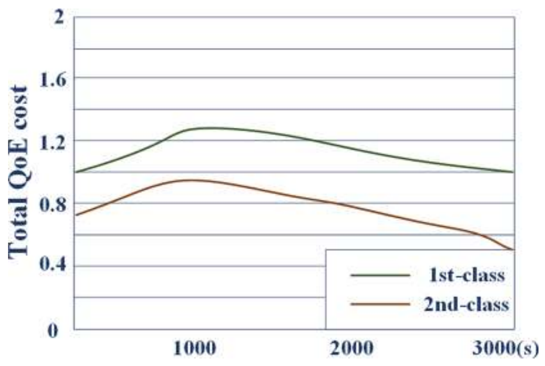

Fig. 10 QoE vs transmission duration ([5])

method in [5] at each moment of the transmission process, it can effectively improve the long-term cumulative reward during the process.

When the video streaming transmission process started in different states, we also evaluated the fluctuation of the objective QoE, as shown in Figs. 11 and 12. We used the state pairs of (0, 6), (2, 6), (4, 8) to conduct experiments. In addition, Figs. 13 and 14 compare the long-term cumulative QoE of first-class and second-class flows with different maximum numbers of coexisting flows, respectively. As the maximum number of streams increases, more video streams may coexist in the base station, thus causing buffer starvation to occur more often. Comparing the cases where the

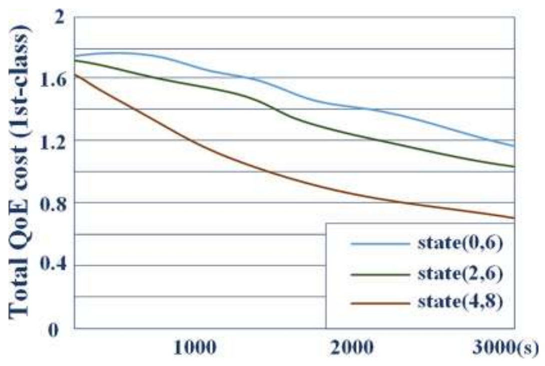

Fig. 11 QoE (class-1) under different initial states

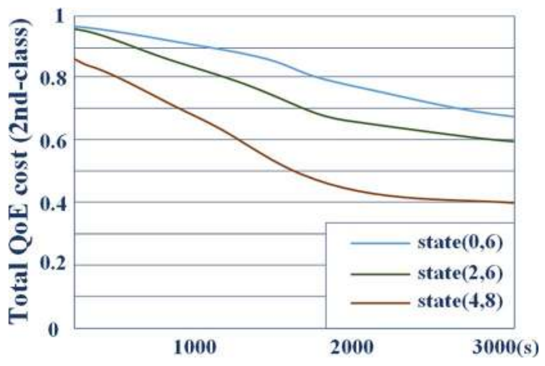

Fig. 12 QoE (class-2) under different initial states

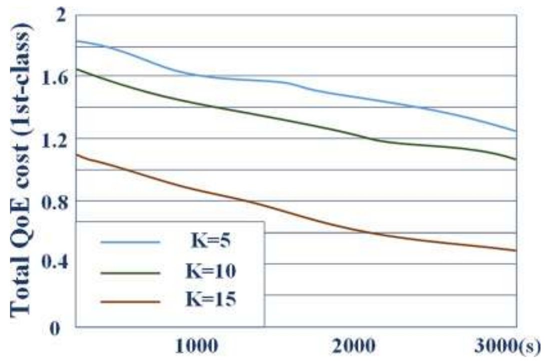

Fig. 13 QoE (c1) under different flow amount

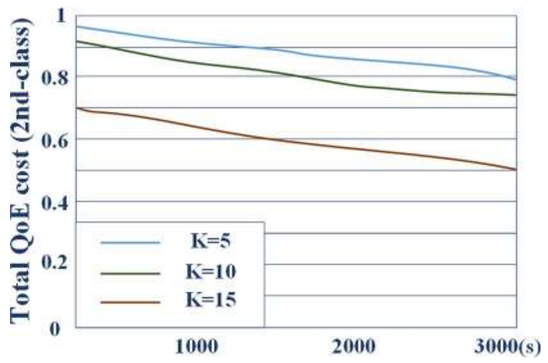

Fig. 14 QoE (c2) under different flow amount

maximum flow number is 5, 10 and 15, it can be found that the long-term cumulative QoE has a strong correlation with the maximum flow amount. Therefore, our model can be used to design a video stream admission control strategy that can tolerate a certain degree of starvation probability.

4.4 Mean DT/VT ratio

The DT/DV ratio is an important indicator that reflects the best buffer ratio during the entire video streaming period. In Figs. 15 and 16, we plot the average DT/DV ratio when the maximum number of streams increases from 5 to 15. It can be seen from the curve that the increase of the number of flows leads to a higher average DT/DV ratio. However, our scheduling strategy can effectively reduce the video buffering time compared to the method in [5] and transmission without scheduling.

Our reinforcement learning model provides an effective scheduling strategy for the QoE trade-off of heterogeneous video streams. When different types of video streams have different perceptions of buffer starvation, our algorithm can empower the base station, so that it has the ability to intelligently schedule different flows, which can optimize the long-term cumulative QoE. For example, if a flow is more sensitive to buffer starvation behavior at a certain moment, the scheduling strategy can provide it with higher priority.

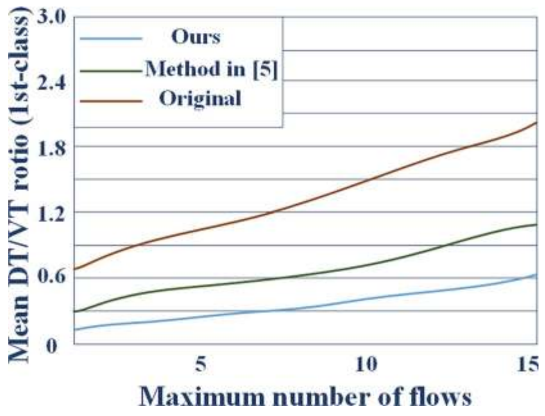

Fig. 15 Mean DT/DV ratio vs max flow amount

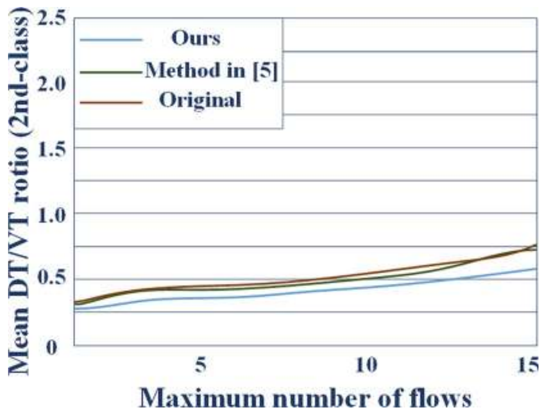

Fig. 16 Mean DT/DV ratio vs max flow amount

5 Conclusion

In this article, we first propose a regression model which combines the recurrent neural network and attention mechanism. This model can accurately calculate the buffer starvation probability and the specific distribution of starvation events in any state during the video streaming transmission process, that is, the buffer starvation behavior is precisely evaluated and analyzed at the packet level. After obtaining the starvation probability distribution, we propose a reinforcement learning model which introduces an intrinsic reward mechanism to intelligently schedule the transmission of video streams. This method can not only maximize the long-term cumulative QoE by dynamically adjusting the start-up delay and data packet prefetching strategy regardless of random noise, but also is highly adaptable to different network environments.

The effectiveness of the approach proposed in this paper has been verified in the 5G-powered UAV video streaming transmission scenario. It is found that our model can stably improve the quality of video transmission service in a complex wireless network environment, and thus provide broader ideas for 5G low-latency research topics.

Abbreviations

5G: 5Th generation mobile networks; UAV: Unmanned aerial vehicle; qoE: Quality of experience; DL: Deep learning; RL: Reinforcement learning; RNN: Recurrent neural network; LSTM: Long short-term memory; GRU: Gated recurrent unit; DQN: Deep Q-learning; DT/VT: Delay time/video time.

Acknowledgements
This work is supported by CMLC team of China Mobile Chengdu Institute of Research and Development.

Authors' information

Yu Su received Ph.D degree in Information and Communication Engineering from Northwestern Polytechnical University. He is currently the senior engineer and the vice president of China Mobile Chengdu Institute of Research and Development, Chengdu, China. He received the second prize of National Technology Invention Awards in 2017. He is the winner of the Special Allowance sponsored by State Council of China.

Shuijie Wang received the B.S. and M.S. degrees in computer science and technology from Harbin Institute of Technology, Harbin, China in 2018 and 2020, respectively. He is currently a researcher of R&D Department II, China Mobile Chengdu Institute of Research and Development, Chengdu, China. Prior to that, he was an assistant researcher at the state key laboratory Peng Cheng Lab, Shenzhen, China. His research interests focus on machine learning, reinforcement learning and intelligent signal processing.

Qianqian Cheng received the B.S. degree in electronic information engineering from the Tianjin University of Science and Technology, Tianjin, China, in 2016, and the M.S. degree in electronic and communication engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2020. She is currently a researcher of R&D Department II, China Mobile Chengdu Institute of Research and Development, Chengdu, China.

Yuhe Qiu received the B.S. degree in communication engineering from Guilin University of Electronic Technology in 2004, and the M.S. degree in communication and electronic engineering from Tsinghua University in 2011. He is currently the technical director of R&D Department II, China Mobile Chengdu Institute of Research and Development, Chengdu, China. His main research interests include machine learning, communication technology and 5G-powered unmanned aerial vehicle.

Authors' contributions

SW designed the overall idea and proposed an experimental plan for verification, and then completed the writing of the first draft. YS conducted overall supervision of the work and determined the final version. QC and YQ participated in some experiments and assisted in revising final manuscript. All authors read and approved the final manuscript.

Funding
This work was supported by the national project of China-Application demonstration of 5G key technologies in the field of people's livelihood under grant G19391310010000.

Availability of data and materials

The video and channel parameter datasets required for this experiment can be collected by 5G-powered drone. They can also be obtained from the corresponding author of this paper.

Declarations

Competing interests

The authors declare that they have no competing interests.

Received: 23 February 2021 Accepted: 10 August 2021
Published online: 23 August 2021

References

1. R. Singh, P. R. Kumar, Optimal decentralized dynamic policies for video streaming over wireless channels. 2019
2. P.C. Hsieh, I.H. Hou, Heavy-traffic analysis of QoE optimality for on-demand video streams over fading channels. IEEE/ACM Trans. Netw. **26**(4), 1768–1781 (2018)
3. Y.H. Ezzeldin, A. Sengupta, C. Fragouli, Wireless network simplification: the performance of routing. IEEE Trans. Inf. Theory **66**(9), 5703–5711 (2020)
4. M. Wu, Y. Zhong, G. Wang, et al. URLLC in Large-Scale Wireless Networks with Time and Frequency Diversities[C]// 2019 IEEE Global Communications Conference (GLOBECOM). IEEE, 2020
5. Y. Xu, Z. Xiao, H. Feng et al., Modeling buffer starvations of video streaming in cellular networks with large-scale measurement of user behavior. IEEE Trans. Mobile Comput. **1**, 1–1 (2017)
6. Qiu H, Psounis K, Caire G, et al. High-rate WiFi broadcasting in crowded scenarios via lightweight coordination of multiple access points. Acm International Symposium on Mobile Ad Hoc Networking & Computing. ACM, 2016
7. Zhang Y, Psounis K. Efficient MU-MIMO via Switched-beam Antennas[C]// Acm International Symposium on Mobile Ad Hoc Networking & Computing. ACM, 2017
8. H. Hu, H. Shan, C. Wang et al., Video surveillance on mobile edge networks—a reinforcement-learning-based approach. IEEE Internet Things J. **7**(6), 4746–4760 (2020)
9. Y. D. Xu and E. Altman et al., Probabilistic analysis of buffer starvation in Markovian queues. In: Proc. IEEE Infocom 2012, tech. rep.

10. Analysis of buffer starvation with application to objective QoE optimization of streaming services. IEEE Trans Multim, 2014, 16(3):813–827.
11. W. Sun, J. Koo, S. Byeon, et al. BlueCoDE: Bluetooth coordination in dense environment for better coexistence[C]// IEEE International Conference on Network Protocols. IEEE, 2017
12. A. Gupta, R.K. Jha, A survey of 5G network: architecture and emerging technologies. IEEE Access **3**, 1206–1232 (2015)
13. Sidi, Habib, B, et al. Multipath Streaming: Fundamental Limits and Efficient Algorithms. IEEE J. Select. Areas Commun. 2017.
14. Y. Zhou, X. Gu, D. Wu et al., Statistical study of view preferences for online videos with cross-platform information. IEEE Trans. Multimedia **20**(6), 1512–1524 (2018)
15. A. Bader, S. Member et al., Mobile Ad Hoc networks in bandwidth-demanding mission-critical applications: practical implementation insights. IEEE Access **99**, 1–1 (2017)
16. I. H. Hou, P. C. Hsieh. QoE-Optimal Scheduling for On-Demand Video Streams over Unreliable Wireless Networks[C]// the 16th ACM International Symposium. ACM, 2015
17. Nikaein, Navid, Schiller, et al. Network store: Exploring slicing in future 5G networks[C]// 2015
18. L. Takács, A generalization of the ballot problem and its application in the theory of queues. J. Am. Stat. Assoc. **57**(298), 327–337 (1962)
19. R. El-Azouzi, K. V. Acharya, S. Poojary, et al. Analysis of QoE for Adaptive Video Streaming over Wireless Networks with User Abandonment Behavior[C]// IEEE Wireless Communications and Networking Conference. IEEE, 2019.
20. H. Poor, Vincent, et al. Application of Non-Orthogonal Multiple Access in LTE and 5G Networks. IEEE Communications Magazine Articles News & Events of Interest to Communications Engineers, 2017.
21. S. Poojary, R. El-Azouzi, E. Altman, et al. Analysis of QoE for adaptive video streaming over wireless networks[C]// 2018 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt). 2018
22. M. Noura, R. Nordin, A survey on interference management for Device-to-Device (D2D) communication and its challenges in 5G networks. J. Netw. Comput. Appl. **1**, 130–150 (2016)
23. J. Park, K. Chung, Queueing theoretic approach to playout buffer model for HTTP Adaptive Streaming. KSII Trans. Internet Inf. Syst. **12**(8), 3856–3872 (2018)
24. T.H. Luan, L.X. Cai, X.S. Shen, Impact of Network dynamics on user's video quality: analytical framework and QoS provision. IEEE Trans. Multimedia **12**(1), 64–78 (2009)
25. N. Bhushan, J. Li, D. Malladi et al., Network densification: the dominant theme for wireless evolution into 5G. IEEE Commun. Mag. **52**(2), 82–89 (2014)
26. P. Yang, N. Zhang, S. Zhang et al., Content popularity prediction towards location-aware mobile edge caching. IEEE Trans. Multimedia **21**(4), 915–929 (2019)
27. Z. Zhang, X. Chai, K. Long et al., Full duplex techniques for 5G networks: self-interference cancellation, protocol design, and relay selection. IEEE Commun. Mag. **53**(5), 128–137 (2015)
28. K. Cho, B. Van Merrienboer, C. Gulcehre, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. Computerence, 2014
29. J. Donahue, L. A. Hendricks, M. Rohrbach, et al. Long-term Recurrent Convolutional Networks for Visual Recognition and Description[C]// 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2017:677–691
30. M. G. Plessen. Automating vehicles by deep reinforcement learning using task separation with hill climbing. 2017.
31. V. Miagkikh, W. F. Punch. An Approach to Solving Combinatorial Optimization Problems Using a Population of Reinforcement Learning Agents GECCO'99 Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation 529: 484489
32. S. Forestier, P. Y. Oudeyer. Towards hierarchical curiosity-driven exploration of sensorimotor models[C]// Joint IEEE International Conference on Development & Learning & Epigenetic Robotics. IEEE, 2015
33. S.J. Nasuto, Y. Hayashi, Anticipation: Beyond synthetic biology and cognitive robotics. Biosystems **148**, 22–31 (2016)
34. Authors A. Large-Scale Study of Curiosity-Driven Learning. 2018.
35. M. Ivo, H. Cornelia, B. Christoph et al., The impact of financial reward contingencies on cognitive function profiles in adult ADHD. PLoS ONE **8**(6), e67002 (2013)
36. K. Wang, Q. Liu, L. Chen, On optimality of greedy policy for a class of standard reward function of restless multi-armed bandit problem. IET Signal Proc. **6**(6), 584–593 (2012)
37. M.J. Kearns, M.L. Littman, S.P. Singh et al., Optimizing dialogue management with reinforcement learning: experiments with the NJFun System. J. Artif. Intell. Res. **6**, 105–133 (2000)

Publisher's Note