DOI: 10.17587/mau.23.414-419

I. N. Agliukov, ildariwe@gmail.com,
National Research University Higher School of Economics, Moscow, 101000, Russian Federation,
K. V. Sviatov, k.svyatov@ulstu.ru,
Ulyanovsk State Technical University, Ulyanovsk, 432027, Russian Federation,
S. V. Sukhov, ssukhov@ulireran.ru,
Ulyanovsk Branch of Kotelnikov Institute of Radioengineering and Electronics of Russian Academy of Sciences, Ulyanovsk, 432071, Russian Federation,

Corresponding author: **Sukhov S. V.,** Cand. Sc., Ulyanovsk Branch of Kotelnikov Institute of Radioengineering and Electronics of Russian Academy of Sciences, Ulyanovsk, 432071, Russian Federation, e-mail: *ssukhov@ulireran.ru*

Accepted on May 16, 2022

A Method for Catastrophic Forgetting Prevention during Multitasking Reinforcement Learning*

Abstract

Reinforcement learning is based on a principle of an agent interacting with an environment in order to maximize the amount of reward. Reinforcement learning shows amazing results in solving various control problems. However, the attempts to train a multitasking agent suffer from the problem of so-called "catastrophic forgetting": the knowledge gained by the agent about one task is erased during developing the correct strategy to solve another task. One of the methods to fight catastrophic forgetting during multitask learning assumes storing previously encountered states in, the so-called, experience replay buffer. We developed the method allowing a student agent to exchange an experience with teacher agents using an experience replay buffer. The procedure of experience exchange allowed the student to behave effectively in several environments simultaneously. The experience exchange was based on knowledge distillation that allowed to reduce the off-policy reinforcement learning problem to the supervised learning task. We tested several combinations of loss functions and output transforming functions. Distillation of knowledge requires a massive experience replay buffer. Several solutions to the problems of optimizing the size of the experience replay buffer are suggested. The first approach is based on the use of a subset of the whole buffer; the second approach uses the autoencoder as a tool to convert states to the latent space. Although our methods can be applied to a wide range of problems, we use Atari games as a testing environment to demonstrate the methods.

Keywords*: reinforcement learning, offline reinforcement learning, multitask learning, experience exchange, experience replay buffer, policy distillation, behavior cloning, imitation learning, catastrophic forgetting, continuous learning*

Acknowledgements: The study was financially supported by the Russian Foundation for Basic Research and the Government of the Ulyanovsk Region (Project No. 18-47-732006).

For citation:

Agliukov I. N., Sviatov K. V., Sukhov S. V. A Method for Catastrophic Forgetting Prevention during Multitasking Reinforcement Learning, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2022, vol. 23, no. 8, pp. 414—419.

DOI: 10.17587/mau.23.414-419

УДК 519.711

DOI: 10.17587/mau.23.414-419

И. Н. Аглюков, студент, ildariwe@gmail.com,
Высшая школа экономики, Москва,
К. В. Святов, канд. техн. наук, k.svyatov@ulstu.ru,
Ульяновский государственный технический университет, г. Ульяновск,
С. В. Сухов, канд. физ.-мат. наук, ssukhov@ulireran.ru,
УФИРЭ им. В. А. Котельникова РАН, г. Ульяновск

Метод преодоления катастрофического забывания при мультизадачном обучении с подкреплением

Принцип обучения с подкреплением основан на взаимодействии агента с окружением в целях максимизации своей награды. Обучение с подкреплением показывает очень хорошие результаты в решении различных задач управления. Тем не менее, попытки обучить интеллектуального агента эффективно решать несколько задач страдают от проблемы так называемого "катастрофического забывания". Полученные агентом знания об одной задаче вытесняются информацией в ходе выработки правильной стратегии для другой. Одним из методов предотвращения катастрофического

*Исследование выполнено при финансовой поддержке РФФИ и Правительства Ульяновской области (проект № 18-47-732006).

забывания при многозадачном обучении является обучение агента на сохраненных в буфере опыта ранее встреченных состояниях. Разработанный нами метод позволяет обучить агента тому, как эффективно вести себя в нескольких средах одновременно на основе обмена опытом с агентами-учителями, используя буфер опыта. Обмен опытом основан на распространенном в глубоком обучении подходе, называемом дистилляцией знаний. Дистилляция знаний позволила свести задачу с подкреплением к задаче обучения с учителем. В ходе исследований были протестированы и выбраны максимально успешные сочетания различных функций потерь и способов преобразования выходных слоев нейросетей. Метод дистилляции знаний требует хранения огромного буфера состояний. Предложены несколько методик оптимизации хранения буфера: использование части буфера и сжатие состояний во внутреннее представление нейросети с помощью автокодировщика. В качестве тестового окружения для экспериментов использовались игры Atari.

***Ключевые слова**: обучение с подкреплением, автономное обучение с подкреплением, мультизадачное обучение, обмен опытом, буфер опыта, дистилляция стратегии, клонирование поведения, имитационное обучение, катастрофическое забывание, непрерывное обучение*

Introduction

General-purpose robotic systems should possess a wide variety of skills. The ability to acquire different skills is an essential feature of intelligent agents (IAs) to ensure their intelligent behavior. The principle of reinforcement learning ensures a general approach for robotic systems to acquire new skills. During reinforcement learning, intelligent agents interact with their surroundings and try to maximize their reward [1]. Current reinforcement learning methods are rather inefficient and assume the learning to occur by trial-and-error method during millions of attempts. Such inefficiency is not critical in a case of an agent learning a single task in a virtual environment. However, training becomes very time-consuming for a robotic platform multitask training in a real-world environment [2]. A large amount of time required to learn a single skill makes the multitask robotic platform very difficult to implement [3]. Hence, interest exists in the development of methods to speed up the learning process of IAs. One of the possible methods for accelerated training of an agent is the borrowing of experience from other agents [4]. To store and exchange experiences more efficiently, one needs to create a compact representation of knowledge. Thus, further research is needed in multitask reinforcement learning and knowledge exchange between AIs [5] and robotic systems [6].

The peculiarities of multitasking reinforcement learning

Many multitask reinforcement learning approaches assume simultaneous availability and accessibility of the data for all the tasks at an arbitrary instant of time. Such an approach can be unrealistic for robots learning a single problem at any instant of time. Multitask learning can be organized by training several agents each on a separate task with a consequent merging of the acquired experiences inside one agent (experience exchange). Currently, several groups suggested several methods for the implementation of experience exchange between IAs during reinforcement learning. One of these methods is imitation learning (also known as 'learning from demonstration') [7]. During imitation learning, the student agent is learning by repeating the actions of an expert agent. Another method of experience exchange between IAs is policy distillation (or behavior cloning). The term 'distillation' usually assumes that the exchange of policies occurs as a result of supervised learning of the student agent on a dataset (set of states and corresponding actions) provided by the expert agent [8—10]. One another method of experience exchange between IAs is offline reinforcement learning [11]. This method is based on the postulate that online interaction with an environment is impractical in a majority of cases: data collection can be a very slow and laborious procedure (for example, in robotics) or it can be very dangerous (for example, during autonomous driving or in medicine). Offline learning reuses previously collected data stored in, the so-called, experience replay buffer without acquiring additional data from the environment [11]. However, during offline learning, an agent can get into the state not stored in the buffer. Thus, to improve the training process, the agent still needs periodic interaction with an environment to get additional data.

In the case of partially observed states, the process of decision-making during reinforcement learning is not Markovian anymore; in this case, one needs to consider previous states encountered by the agent. Storing the sequential states in a buffer could require additional computer memory. The excessive use of computer memory is even more probable during multitask learning. The size of the stored experience replay buffer can be decreased if one uses compressed representations of states. One can use autoencoders to translate the states into compressed latent representations [12, 13].

An additional problem of multitask learning is the retaining of the skills acquired by an agent at

earlier stages. This can be a nontrivial task for IA because of the problem of catastrophic forgetting well known in machine learning [14]. There are several approaches to fight catastrophic forgetting [15]. The simplest one assumes joint training on all the data previously encountered by an agent. The experience replay bufferes suit well for this purpose.

In this paper, we investigate the mechanisms of experience exchange between intelligent agents in reinforcement learning tasks. The agents are located in partially observable states. To implement the experience exchange, we use the distillation method reducing the reinforcement learning to a supervised learning problem. We tested several methods for downsizing the stored experience replay buffer. We demonstrate the possibility of the prevention of catastrophic forgetting when an agent uses its own previously stored buffer.

Experience exchange between intelligent agents on the base of compressed representations of the states of an environment

In this paper, we present an approach that allows to merge the experiences acquired by several agents into a single multitask agent. The problem statement is in the following. Two intelligent agents are trained each on its own task. These two pretrained agents serve as teachers. The student agent is a copy of one of the teacher agents. We search for a way to teach the student agent the skills of the second teacher. During learning the skills of the second teacher, it should be ensured that the student does not forget previous skills.

At the first stage, we train the teacher agents with a deep Q-learning (DQN) algorithm [16]. In this approach, IA learns the value $Q(s, a)$ of certain action a in a specific state s. In each step, an agent chooses an action with the largest value. In this approach, after every interaction with an environment, the agent stores into an experience replay buffer the current state s, the action taken a, the reward for this particular action r, the next state s', and the variable indicating the terminal state. The buffer allows for efficient use of previous experience through its continuous rehearsing. The variant of Q-learning used in the current paper is called Dueling-DQN [17]. In this approach, the neural network consists of two evaluating networks. One network predicts the weighted average of Q-function over all the actions $V(s)$. Another network predicts the advantage of every action:

$$A(s,a) = Q(s,a) - V(s).$$

These two estimates are combined inside a separate layer to get the current value of the Q-function.

At the second stage, we transfer the knowledge to the student agent using distillation [18]. Within this approach, the student receives a batch of states out of the experience replay buffer of the teacher and outputs state-action Q-values ($Q_{student}$). We compare student's Q-values to the ones of teacher agents ($Q_{teacher1}$, $Q_{teacher2}$) for the same states and correct the weights of student's neural network. Using this method, we reduce the reinforcement learning problem down to supervised learning. To prevent catastrophic forgetting, one needs to satisfy an additional condition: the Q-values of student agent with new knowledge should not differ from the student's previous Q-values. Accordingly, we use the following loss function:

$$Loss = \\ = loss_1(Q_{teacher1}, Q_{student}) + loss_2(Q_{teacher2}, Q_{student}),$$

where $loss_1$ and $loss_2$ show the divergence between Q-functions of the student and the teacher (Fig. 1). Our approach reminds behavior cloning [8], however, it is applied not to the behavioral policy, but to the Q-function.

Experience replay buffers can occupy an enormous amount of computer memory. We tested several approaches to decrease the buffer size. One of the approaches is based on selective storage of the encountered states. The first option is to use only a certain amount of the latest states. The second option is to use the subsampling of the states encountered by an agent. We use the following algorithm for the subsampling. On arrival into some state, we generate a random key from the uniform distribution $U[0, 1]$. The current state is added into the experience replay buffer if the key is larger than B/t, where B is the current buffer size, t is the state's order number. The disadvantage of this option is that stored states are

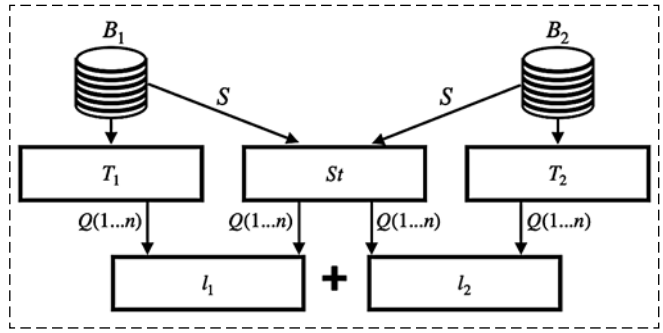

Fig. 1. The scheme of experience exchange between IAs. B_i is the experience replay buffer of i-th environment, S is the batch of states, T_i is the teacher agent for the i-th environment, St is the student agent, Q are the state-action values for the student and teachers, l_i is the loss function for the i-th environment

not consecutive, which is a problem in the case of partially observable states. The solution could be the simultaneous storage of several consecutive states at once. However, storing the series of states can lead to the excessive use of computer memory. One another approach to decrease the size of the buffer is to compress the representation of stored states. To perform the compression, we use an autoencoder [13]. Autoencoders are standard instruments of machine learning for dimensionality reduction of data. The buffer compressed by an encoder can be further restored by the decoder and used for experience exchange between the agents.

Computational test

As an example of the application of the developed multitask learning method, we used IAs trained to play Atari games in OpenAI environment [19]. We should note that the developed algorithm is general and can be used in other environments and robotic systems.

For experience exchange experiments, we trained five agents on different Atari games. Image frames received from the environments were stacked into four frames per state following the original paper of DeepMind [16]. This stacking is related to the fact that each frame represents the partially observable state (for example, the frame does not contain information about the speed of objects). Among five environments, we chose the agents having the same number of actions: two agents (Atlantis and Breakout) had 4 action options; three agents (Demon Attack, Pong, and Up and Down) had 6 options for action. As DQN learning is very time-consuming, we trained the agents only to the level comparable to human abilities to play these games. Maximum points scored by the agents in different games are shown in Table 1.

In the process of agents training, we stored the replay buffer (10^5 states for Atlantis and Breakout games, $2 \cdot 10^5$ states for Pong, Demon Attack, and Up and Down games). These buffers served as the base for later experience exchange. Different environments require different times to achieve an optimal score; this fact determines the different sizes of stored buffer (Table 2).

Table 1

Scored points for the environments used in the paper

Environment	Random behavior	Human	DQN agent
Atlantis	~13 000	~29 000	~3 000 000
Breakout	~2	~30	~350
Demon Attack	~150	~3400	~30 000
Pong	~−21	~9	~20
Up And Down	~533	~9 000	~16 000

Table 2

Dimensions of the replay buffers

Environment	Atlantis	Breakout	Demon Attack	Pong	Up And Down
Amount of the states in the buffer, 10^6	4	4	21,4	0,8	14,6
The dimension of the total buffer, Gb	26,29	26,29	140,63	5,26	95,94
The dimension of the compressed buffer, Gb	0,95	0,95	5,1	0,2	3,48

During training, we used the batch consisting of 16 states for every environment. To optimize the gradient descent, we used the Adam optimizer [7] with a learning rate $6{,}25 \cdot 10^{-4}$. The number of iterations of the algorithm was 10^6.

The evaluation of the quality of the models was performed as follows. In every environment, the trained agent was tested ten times during 12 500 consecutive states. After that, we calculated the mean and standard deviation for the points scored during every test.

As an alternative to our method of experience exchange, we tested offline learning. In this approach, the student agent learns the effective behavior in two environments by training with the DQN algorithm on a combined buffer of teacher agents. This approach resulted in a fast loss of performance in the previously learned environment and in the absence of the learning progress in the new environment (Table 3).

At the next stage, we tested the methods of experience exchange between IAs on the base of distillation with additional buffer compression methods.

As the first option, we used an approach based on storing only the part of the total buffer obtained dur-

Table 3

Comparison of the loss functions for Pong and Up-and-Down games. The mean and standard deviation of the scored points are provided

Environment	Offline learning	Mean square loss	Kullback-Leibler loss	Binary cross-entropy loss
Pong	−19,9 ± 5,05	15,50 ± 1,59	15,67 ± 0,93	15,54 ± 0,86
Up and Down	3878 ± 186	5111 ± 1256	4347 ± 943	6336 ± 886

ing initial training. A large number of stored states requires enormous computer memory to accommodate the whole buffer (Table 2). For example, the size of the buffer for the Demon Attack game amounted to 140 Gb. To decrease the size of the buffer, we used $8 \cdot 10^5$ latest states for every game. The output values of Q-functions were modified with the argmax function (to single out the only action) or with the softmax function (to take into account the probability of every action); the modified values were supplied into the loss function. The best results were achieved with the softmax function combined with binary cross-entropy as a loss function. This combination allowed the student agent to achieve scores comparable with that of teacher agents (Table 3).

The use of the argmax function demonstrated learning instability. During the initial epochs of training, the agent adopted the ability to adequately behave in two environments. However, during further training, there occurred a surge in the loss function with the agent's sudden loss of the ability to have productive interaction with the environments.

Besides binary cross-entropy, we used mean-square and Kullback-Leibler losses. As one can see from Table 3, all these loss functions adequately solve the problem.

The next option to decrease the buffer size is selective storage of encountered states as was described in the previous section. Each entry in the buffer consisted of four consecutive states determined by their partial observability. We collected two buffers 20 000 states each for games Pong and Up-and-Down. When training on this subsampled buffer, the student agent did not show high scores compared to training on the final part of the total buffer. The following scores were achieved: 13,66 ± 0,86 for the Pong game and 2804 ± 255 for Up-and-Down.

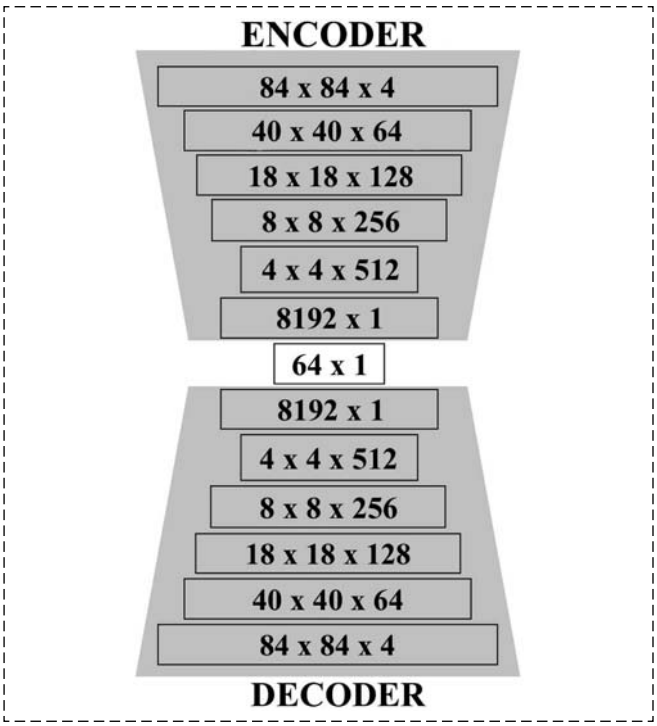

Fig. 2. The architecture of the autoencoder used for the compression of the states of the game environments

The next method of decreasing the size of the buffer was based on the transformation of the frames of the games into a latent space of autoencoder. For every environment, the autoencoder with latent space 64 was trained on the total buffer (Fig. 2). Autoencoder was trained on batches of 25 states. We used the Adam optimizer with a learning rate 0,0005. All the states were compressed with a coder and stored on a computer drive that allowed to reduce the representation of the state of the environment from 84 (int) × 84 (int) × 4 = = 28 224 bytes down to 64 (float32) × 4 = 1024 bytes. The autoencoder allowed to reduce the buffer size by 27 times (Table 2); that way the buffer fully fitted into computer memory. The process of experience exchange had just a small modification: during the training of the student agent, the stored states should be decompressed with a decoder. After the training of student agent on a compressed buffer, the efficiency of its behavior became comparable to that of teacher agents (Table 4). Table 4 shows the results for the softmax function at the output of the network combined with the binary cross-entropy loss function. The only problem with the proposed method of buffer compression is the limitations of the autoencoder: the peculiarities of the autoencoder's compression could make it difficult to encode some essential parts

Table 4

The outcome of experience exchange

Teacher 1 Teacher 2	Teacher's score	Student's score (training on a part of the buffer)	Student's score (training on latent representations)
Atlantis	271856 ± 132507	233824 ± 178721	180000 ± 17200
Breakout	212,3 ± 45,3	199,9 ± 52,8	8,4 ± 2
Demon Attack	9251 ± 3301	6169 ± 2599	9284 ± 2987
Up and Down	6608 ± 918	4526 ± 329	4892 ± 470
Pong	15,9 ± 1,43	15,54 ± 0,86	15,43 ± 0,78
Up and Down	6607 ± 918	6336 ± 886	6397 ± 686

of an image. For example, for the Breakout game, the autoencoder could not encode the image of the ball. As a result, experience exchange did not work for this particular case.

Conclusion

We proposed and tested the variant of behavior cloning where Q-function is used instead of policy. The behavior cloning in our case had the additional feature: with our method, the student agent learned the behavior of a teacher agent without losing the ability to perform previously learned tasks. This was achieved by utilizing both the teacher's and its own experience replay buffers. The outcomes of experience exchange are shown in Table 4. One can see that the performance of student agents is comparable to that of teachers. One can conclude that our method successfully handles catastrophic forgetting and can be used in multitask reinforcement learning.

The methods of decreasing the dimensions of experience replay buffer (using the latest encountered states or using the latent representations) allowed to significantly reduce the usage of computer memory without the significant deterioration of the accuracy of experience exchange. The use of latent features allowed to decrease the amount of memory for storing the buffer. However, it increased the time of agent's training as the states should have been first decompressed to be used in knowledge distillation. This method had additional limitations due to the peculiarities of the encoding-decoding procedure. To be used in full, the autoencoder's compression quality needs further improvements.

References

1. **Shmygun A. A, Ermolaeva L. V., Zakharov N. V.** Obuchenie s podkrepleniem, *Novaya Nauka: sovremennoe sostoyanie i puti razvitiya*, 2016, no. 12-3, pp. 189—191, available at: https://elibrary.ru/download/elibrary_27724493_81982095.pdf (in Russian).

2. **Ecoffet A., Huizinga J., Lehman J. J., Stanley K. O., Clune J.** First return, then explore, *Nature,* 2021, vol. 590, no. 7847, pp. 580—586, DOI: 10.1038/s41586-020-03157-9.

3. **Kalashnikov D., Irpan A., Pastor P., Ibarz J., Herzog A., Jang E., Quillen D., Holly E., Kalakrishnan M., Vanhoucke V., Levine S.** Qt-opt: Scalable deep reinforcement learning for vision based robotic manipulation, *arXiv preprint* arXiv:1806.10293, 2018, available at: https://arxiv.org/abs/1806.10293.

4. **Da Silva F. L., Taylor M. E., Costa A. H. R.** Autonomously reusing knowledge in multiagent reinforcement learning, *Proc. 27th Int. Joint Conf. on Artificial Intelligence*, 2018, pp. 5487—5493, available at: https://www.ijcai.org/proceedings/2018/0774.pdf.

5. **Koroteev M. V.** Obzor nekotoryh sovremennyh tendentsyj v tehnologiyah mashinnogo obucheniya, *E-Management*, 2018, pp. 30—31 (in Russian).

6. **Lesort T., Lomonaco V., Stoian A., Maltoni D., Filliat D., Díaz-Rodríguez N.** Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges, *Information Fusion*, 2020, vol. 58, pp. 52—68, DOI: 10.1016/j.inffus.2019.12.004.

7. **Ross S., Gordon G., Bagnell A.** A reduction of imitation learning and structured prediction to no-regret online learning, *Journal of Machine Learning Research*, 2011, vol. 15, pp. 627—635, available at: http://proceedings.mlr.press/v15/ross11a.

8. **Parisotto E., Lei Ba J., Salakhutdinov R.** Actor-mimic: Deep multitask and transfer reinforcement learning, *arXiv preprint* arXiv:1511.06342. 2015, available at: https://arxiv.org/abs/1511.06342.

9. **Teh Y. W., Bapst V., Czarnecki W. M., Quan J., Kirkpatrick J., Hadsell R., Heess N., Pascanu R.** Distral: Robust multitask reinforcement learning, *Proc. 31st Int. Conf. on Neural Information Processing Systems,* 2017, pp. 4499—4509, available at: https://proceedings.neurips.cc/paper/2017/hash/0abdc563a06105aee3c6136871c9f4d1-Abstract.html.

10. **Rusu A. A., Colmenarejo S. G., Gulcehre C., Desjardins G., Kirkpatrick J., Pascanu R., Mnih V., Kavukcuoglu K., Hadsell R.** Policy distillation, *arXiv preprint* arXiv:1511.06295. 2015. URL: https://arxiv.org/abs/1511.06295.

11. **Levine S., Kumar A., Tucker G., Fu J.** Offline reinforcement learning: Tutorial, review, and perspectives on open problems, *arXiv preprint* arXiv:2005.01643. 2020. URL: https://arxiv.org/abs/2005.01643.

12. **Ballard D.** Modular learning in neural networks, *Proc. 6th National Conf. on Artificial Intelligence,* 1987, vol. 1, pp. 279—284, available at: https://www.aaai.org/Library/AAAI/1987/aaai87-050.php.

13. **Akinina N. V., Akinin M. V., Sokolova A. V., Nikiforov M. B., Taganov A. I.** Avoenkoder: podhod k ponizheniyu razmernosti vektornogo prostranstva s kontroliruemoj poterej informatsii // Izvestiya Tul'skogo gosudarstvennogo universiteta. Tehnicheskie nauki. 2016. no. 9. pp. 3—12, available at: https://elibrary.ru/download/elibrary_27277969_89179513.pdf (in Russian).

14. **McCloskey M., Cohen N. J.** Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem, *Psychology of learning and motivation*, 1989, vol. 24, pp. 109—165, DOI: 10.1016/S0079-7421(08)60536-8.

15. **Sukhov S., Leontev M., Miheev A., Sviatov K.** Prevention of catastrophic interference and imposing active forgetting with generative methods, *Neurocomputing,* 2020, vol. 400, pp.73—85, DOI: 10.1016/j.neucom.2020.03.024.

16. **Mnih V., Kavukcuoglu K., Silver D., Rusu A. A., Veness J., Bellemare M. G., Graves A., Riedmiller M., Fidjeland A. K., Ostrovski G., Petersen S., Beattie C., Sadik A., Antonoglou I., King H., Kumaran D., Wierstra D., Legg S., Hassabis D.** Human-level control through deep reinforcement learning, *Nature*, 2015, vol. 518, no. 7540, pp. 529—533, DOI: 10.1038/nature14236.

17. **Wang Z., de Freitas N., Lanctot M.** Dueling Network Architectures for Deep Reinforcement Learning, *Int. Conf. on Machine Learning*, 2015, pp. 1995—2003, available at: http://proceedings.mlr.press/v48/wangf16.html.

18. **Biryukova V. A.** Tehnologiya distillyatsii znanij dlya obucheniya nejronnyh setej na primere zadachi binarnoj klassifikacii, *Intellektual'nye sistemy. Teoriya i prilozheniya,* 2020, vol. 24, no. 2, pp. 23—52, available at: http://intsysjournal.ru/pdfs/24-2/Birukova.pdf (in Russian).

19. **Brockman G., Cheung V., Pettersson L., Schneider J., Schulman J., Tang J., Zaremba W.** Openai gym, *arXiv preprint* arXiv:1606.01540, 2016, available at: https://arxiv.org/abs/1606.01540.