DOI: 10.20998/2079-0023.2025.01.19
UDC 004.8:37.018.43:004.738.5

V. Y. SOKOL, Doctor of Philosophy, Associate Professor, Scientific Researcher – The Learning Technologies Research Group RWTH Aachen University; e-mail: sokol@cs.rwth-aachen.de; ORCID: https://orcid.org/0000-0002-4689-3356
M. D. GODLEVSKYI, Professor, Doctor of Technical Sciences., Director of the Institute of Computer Sciences and Information Technologies Kharkiv, Ukraine; e-mail: Mykhailo.Hodlevskyi@khpi.edu.ua; ORCID: https://orcid.org/0000-0003-2872-0598
M. A. BILOVA, Candidate of Technical Sciences, Docent, Associate Professor of the Department of Software Engineering and Intelligent Technology Management, Kharkiv, Ukraine; e-mail: Mariia.Bilova@khpi.edu.ua; ORCID: https://orcid.org/0000-0001-7002-4698
R. A. TUPKALENKO, Graduate Student, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine; e -mail: Roman.Tupkalenko@cs.khpi.edu.ua; ORCID: https://orcid.org/0009-0008-7268-2393

INTELLIGENT TECHNOLOGY FOR OPTIMIZING THE PROJECT-BASED APPROACH TO TEACHING STUDENTS USING LEARNING MANAGEMENT SYSTEMS

This work is devoted to developing of a recommendation system that enables the effective construction of learning trajectories for students studying in universities using learning management systems. The core of the recommendation system will be an artificial deep neural network of forward propagation, which takes as input information about the student and the subject that he or she should study and produces as output the most effective learning trajectory. The neural network is trained on data prepared using multi-agent modeling. The domain was decomposed into separate components and in the process of multi-agent modeling was represented in the form of agents and the environment in which they communicate with each other. The subject of this research is the modeling of the learning process in learning management systems. The purpose of the study is to optimize the student learning process within learning management systems. The subject area was analyzed and studied, the architecture of the recommendation system was developed, the architecture of the multi-agent system was developed, and a mathematical model of agent interaction was developed. To achieve the goals of the study, it is necessary to solve main tasks, namely: to prepare a training data set using multi-agent modeling and to develop and train a recommendation system that is based on an artificial deep neural network on this data. After completing all the tasks of the work, it is expected that the learning process of students in the learning management system will be optimized in terms of time and resources spent on learning, and the average level of knowledge will be increased.

Keywords: multiagent modeling, learning management system, artificial neural network, backpropagation algorithm, learning process, effective learning trajectory

Introduction. A Learning Management System (LMS) is a software application for administering, documenting, tracking, reporting, automating and delivering of educational courses, curricula, materials, or training and development programs [1]. Learning management systems constitute the largest segment of the learning systems market and continue to grow in popularity. The purpose of learning management systems is to improve the efficiency of the student learning process in terms of time and effort spent on each subject [1]. It is obvious that by improving the quality of such a system, the quality of education received by students can be significantly increased and the financial costs associated with its use can be reduced.

The authors of this article set themselves the main task of developing a recommendation system for LMS, which is based on approaches related to artificial intelligence**.** The core of the recommendation system will be a deep neural network of forward propagation. The developed recommendation system will make it possible to obtain the most effective learning trajectory for an individual student in terms of the quality of education received and financial costs, taking into account his unique characteristics and skills.

The main problem in developing of a recommendation system based on an artificial neural network is the lack of sufficient historical data suitable for use as a training set. That is why it is planned to use mathematical modeling of the learning process to prepare training data. Different methods and algorithms are currently used to model learning systems. Here is a brief description of existing approaches.

1. System Dynamics

System dynamics – this approach is used to model long-term trends in learning by constructing differential equations that describe the system. The method of system dynamics is based on the concepts of feedback, flows and accumulators, which makes it possible to create formal models of the development of students' knowledge over time.

The simulated system consists of interrelated components, such as the level of learning, cognitive load, motivation, level of student participation and external factors (teaching methodology, availability of resources, etc.). Many studies confirm the effectiveness of using system dynamics to analyze educational processes [2].

2. Machine learning methods

Machine learning methods are also widely used to model and analyze learning processes, allowing you to predict student performance, optimize learning materials, and personalize learning approaches [3]. For example, classification algorithms such as logistic regression, decision trees, and gradient boosting (XGBoost) are used to predict training outcomes based on historical data. Clustering methods of *K*-means and DBSCAN make it possible to group students according to similar characteristics, which contributes to the identification of risk groups. Recommendation systems based on collaborative and content filtering help personalize the learning experience by offering customized content [4].

3. Bayesian Networks

Bayesian networks are another effective tool for modeling learning processes. They allow you to create probabilistic models that take into account the uncertainty in students' knowledge and allow you to build personalized educational trajectories [5].

4. Ontological methods

Ontological approaches are also widely used to model educational environments, structure learning materials, and personalize learning content [6].

Statement of the problem and purpose of the study

The subject of the research is models and tools for software development to recommend the most effective learning trajectory for students. The theoretical and methodological basis of the work is the algorithms for training artificial neural networks, multi-agent modeling, the theory of the learning process under the learning management system, and the personal behavior of students in the learning process.

The purpose of the study is to optimize the process of students learning using learning management systems based on the project approach.

To achieve this goal, it is necessary to solve two problems, namely: to prepare a training dataset using mathematical modeling of the training process; to develop and train a recommendation system based on an artificial deep neural network of forward propagation on this dataset.

After completing the tasks of the study it is expected that the learning process of students in the learning management system will be optimized in terms of time and resources spent on learning, and the average level of grades will increase.

According to the authors of this article, the most promising method for modeling of the learning process under learning management systems is multi-agent modeling. It makes it possible to make predictions of scenarios for the behavior of elements within complex non-deterministic systems of various sizes and configurations, modeling the process within the system in the form of interaction of the so-called agents [7–10].

The main elements of agent-based modeling are agents and the space in which they interact and respond to events that occur. Agents are modeled individually. Each agent has sensors with which they can perceive information from the environment, a set of goals that they strive to fulfill, and a memory that serves to store the state. They may have incomplete information, make mistakes, adapt to the situation, and take the initiative. Agent modeling is based on the principles of diversity, interconnectedness, and interaction.

Domain area description

To solve the problem of modeling of the learning process under LMS, we have defined the main terms that describe domain.

A course of study is a set of subjects that consist of separate topics which student has to master during a certain period of study.

Based on the course of study, teachers form a set of educational branches consisting of individual tasks that should cover all the topics of subjects that are chosen for study.

A learning trajectory is a sequential set of learning branches. Each branch consists of certain number of tasks. The tasks are arranged in a certain logical order. Each task represents a set of exercises that must be completed either by an individual student or by a group. For example, a task might be described as "Programming a basic calculator in Python."

The completion of a branch depends on the successful completion of each of the tasks in the prescribed order. The individual branches within the learning trajectory can be conceptualized as nodes that together form a larger "branch graph."

A key principle of this structure is a prerequisite: a branch can only be selected for training if all of its previous branches have been successfully completed.

A general branch graph is a directed acyclic graph.

There are also various obstacles and challenges that may arise during learning (such as health issues or Internet connectivity issues) that affect the ability to successfully complete a particular learning path.

It should also be noted that there are various factors in the learning process that affect the potential grades that students receive. For example, the level of training of a student and professionalism affect the final grades received by students. Also, various non-deterministic things, such as health problems or problems with the Internet connection, can reduce the chances of getting a good grade, since the quality of learning will be much lower for a particular student. Fig. 1 shows an example of how tasks are located within a branch.

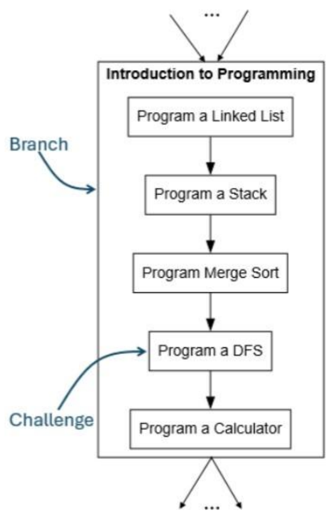

Fig. 1. Example of a training branch and its associated tasks

In order to successfully complete the course, you need to study and cover a certain number of subject topics that make up this course of study. An individual subject can only be counted as completed if at least n % of the subject's topics have been covered as part of the student's completion of a particular learning trajectory. The specific percentage is set by the teacher.

One task can cover many topics from different subjects. Fig. 2 shows an example of such an attitude.

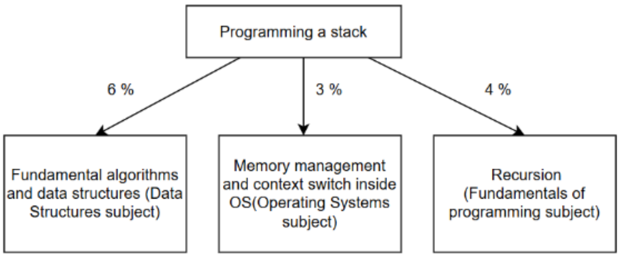

Fig. 2. An example of the connection between the problem and its coverage of certain topics of various subjects

Description of the recommendation system architecture

It should be emphasized once again that the main goal of the study is to create a recommendation information system that will make it possible to optimize the process of teaching students with a project approach in terms of time and effort required for a student to acquire knowledge in a certain set of subjects. The recommendation system will be built using an artificial neural network.

The architecture of any neural network consists of layers and neurons within them. There is an input layer (takes as input a vector containing data that needs to be classified), one or more hidden layers, and an output layer (a vector containing the probabilities of belonging to a particular class). Each individual neuron connects to all (or a specific neuron, depending on the architecture) neurons in the next layer and has its own associated weight and displacement [11–13].

In our case, we plan to use the architecture of a deep fully connected neural network with three hidden layers (fcNN). The quality of any neural network's predictions depends on the quality and volume of training data on which it will be trained. Each training data item is a pair of vectors x_i and y_i, where x_i – vector that is fed to the input of the neural network, and y_i – the vector of expected values to which the vector of activations of output layer a_i^L is being compared.

In our case, the vector x_i will contain a description of the student's characteristics (age, gender, stress resistance, education, etc.), the name of the subject or subjects to be studied, and the grades obtained in the study of related subjects by this student. Vector y_i will contain the probability that a certain learning trajectory is the most effective for a particular student and the set of subjects he must study.

Vector of activations of output layer a_i^L – vector containing the probabilities that a certain learning trajectory will be the most effective. Accordingly, the learning trajectory with the highest probability is considered as the most effective for a particular student. Fig. 3 shows the high-level neural network architecture that will be used to recommend a learning path.

Creating training data using multi-agent modeling

It is planned to use multi-agent modeling to create a training dataset, as there is currently not enough stored data that can be used for this purpose. The data preparation process consists of three stages.

The first step is to prepare static datasets that describe the components of the LMS learning process. These datasets describe (this list is not complete and there are many more items):

- different types of students (different ages, genders, education, IQ, learning ability, etc.);
- different types of teachers (age, education, level of qualification, etc.);
- subjects studied (linear algebra, databases, Java, etc.);
- existing tasks and topics for each of the subjects;
- branches of study and rules describing their location relative to each other.

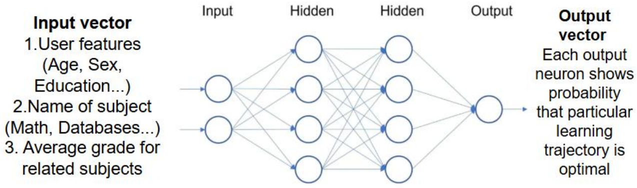

Fig. 3. High-level representation of neural network's architecture

On the second step all potentially possible learning trajectories for certain subjects are formed. This process can be represented as connection of different branches in a certain order, taking into account the rules of their location relative to each other. After the compilation of trajectories from the original branches is completed, these branches are divided into smaller ones (by subtracting a certain number of tasks) and another list of learning trajectories is formed from them. This process continues as long as the coverage of each individual branch of the specified subjects exceeds a certain specified percentage, *n* %.

On the third step a set of all possible triplets will be formed: "student – a group of subjects for study – a learning trajectory". Those triplets will serve as the input of the multi-agent system for modeling of the learning process and assessing the quality of education.

After the data preparation is completed, the process of multi-agent modeling of the training process for each triplet begins. Before we can describe the simulation process, we need to describe the runtime and agents.

Description of the runtime and agents

The agent execution environment is the context within which agents exist and interact with each other. In our case, the runtime can be represented as a 2D space that has a certain set of mechanisms through which agents are managed and interact. For this space, the abscissa axis is the current time inside the simulation, and the ordinate axis is the distance between agents (the smaller the distance, the more likely it is that agents will be able to interact with each other).

The key concept of the modeling process will be the amount of knowledge. This value correlates with the probability that a particular student agent will be able to successfully complete a specific task in a certain period of time. The more knowledge there is at the moment, the higher the probability of successfully completing the task. Each agent can interact with other agents and transfer or receive some amount of knowledge necessary to perform tasks.

The following agents take part in the modeling process:

- A student agent is an agent that simulates the student's behavior in the learning process using an LMS;
- A teacher agent is an agent that simulates the behavior of the teacher in the learning process using the LMS;
- A mentor agent is an agent that simulates behavior of mentors in the learning process under LMS;
- A university agent is an agent who simulates the behavior of the university and can deliver teachers to conduct training;
- An interference agent is and agent with the help of which the learning process is being interfered, it is responsible for generation of various obstacles that may arise during learning.

Description of the multi-agent modeling process

To describe the behavior of agents within a multiagent system, the theory of Markov processes will be used [14] it consists of several components that are needed to be described.

States

The set of states *S*, $s_i \in S$ characterizes the current position of the agent in the learning process. As an example, consider the student agent and its possible states (this list is not complete and only partially describes all possible states): "Ready to perform a task", "Performs a task individually", "Asks for help from other students", "Asks for help from a teacher", "Helps other students", "Performs a task together with other students", "Studies educational literature", "Sick", "Completed the task", "Completed the branch", "Completed the learning trajectory".

Actions

The set of actions *A*, $a_i \in A$, that the agent can execute in the current state. As an example, consider a student agent and its possible actions (this list is not complete and only partially describes all possible actions): "Start a task individually", "Request help from other students", "Request help from a teacher", "Start helping other students", "Start a task with other students", "Start studying lecture material", "Signal the onset of illness", "Signal refusal to continue the task", End task, End branch, and Finish training path.

Transition function $P(s_i', s_i, a_j)$

Describes the probability of transitioning from the state s_i to the state of s_i' when performing an action a_j.

Reward $R(s_i, a_j)$

Reward for completing an action a_j in state s_i. In our case for the student's agent, this value correlates with the amount of knowledge that the student can obtain by performing this or that action.

The behavior of agents can be expressed in the form of a matrix $\boldsymbol{P} = \{P(s_i, a_j) \mid i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n\}$, where m and n are cardinalities of sets *S* and *A*. When modeling the behavior of agents, the method of values iteration will be used to find the most effective action in each of the agent's states. The idea behind this algorithm is to iteratively find the most effective agent policy. Let's introduce a few key concepts used in this method.

Policy $\pi(s_i)$ – it is a function that defines the behavior of an agent starting from each state s_i and in a series of subsequent states. Determines which action he should be chosen in each of the states.

The most effective policy $\pi'(s_i)$ – this is a policy in which the expected total reward is maximum. For example, for a student agent, which moves according to the most effective policy will result in the most amount of knowledge.

State value $V^{\pi}(s_i)$ is determined on the basis of (1) – this is the mathematical expectation $\mathbb{E}$ of the amount $\sigma_{\pi(s_i)}$ rewards received for performing a certain subset of actions of set *A* that corresponds to the policy $\pi(s_i)$ [15].

$$V^{\pi}(s_i) = \mathbb{E}\left[\sigma_{\pi(s_i)}\left(R(s_i, a_j) \times \gamma_{\pi(s_i)}\right)\right], \qquad (1)$$

where $\{\gamma_{\pi(s_i)}\}$ – set of policy-specific weighting ratios related to policy $\pi(s_i)$ that belong to an interval [0, 1).

Initial value for all $V^{\pi}(s_i)$ is set arbitrarily. Then, at each iteration, this value is updated.

Best state value $V^{\pi'}(s_i)$ of state s_i is being determined based on (2). This value represents the maximum expected value of the rewards sum $\sigma'_{\pi(s_i)}$ received for performing a certain subset of actions of the set of all allowed actions *A* [16]:

$$V^{\pi'}(s_i) = \max_{\pi(s_i)}\left\{\mathbb{E}\left[\sigma_{\pi(s_i)}\left(R(s_i, a_j) \times \gamma_{\pi(s_i)}\right)\right]\right\}. \qquad (2)$$

The agent's goal is to determine the most effective policy π' for itself. After the best values are found $V^{\pi'}(s_i)$ for each of the states s_i. The most effective policy $\pi'(s_i)$ will be a set of actions leading to the best values for a particular state s_i.

Description of modeling stages

The first stage of modeling is the initialization of agents and the environment within which they will interact with each other, as well as a timer that determines the current time within the simulation. Each of the agents goes to its initial state and receives its initial task from the environment.

At the second stage each agent, using the method of values iteration, calculates the most effective strategy of behavior for herself and, based on it, chooses the best action a_j', then moves to the next state s_i'. In case the environment generates events that can change the next state of the agent, then the agent can move to this new "unplanned state".

On the third stage all agents enter a new state s_i' and update their internal memory with information about the events that occurred in the previous stage and send message to the surrounding environment about their current state. At this point, each agent must check whether they have finished their current task and whether they should proceed with the next one. If there are no new tasks, the agent goes to its terminal state, which is defined as its final state, after which it stops working.

Further the simulation process is an alternation of stages 2 and 3. Each agent strives to fulfill his goal via calculating the most effective action for himself and moving into a new most effective state. The modeling process will be completed if one of the two conditions is met: all agents have moved to their terminal state; the time allotted for studying under the learning trajectory is over.

At the end of the modeling process for a specific triplet, we save information about how much time the learning process took for a specific trajectory by a specific student, how successful the learning was, and whether it was completed at all, which will be expressed in the final grade that the student will receive. Then the modeling process is repeated for all existing triplets. After that, for each student-subject group pair, only those learning trajectories are selected for which the learning success rate exceeds the specified threshold *N* %.

Those vectors will be stored in the training dataset for training of neural network. A single vector x_i – contains information about the student and subjects, and the vector y_i – contains information about the effectiveness of learning trajectories for a given set of subjects and the student (a separate element describes the probability of how effective a particular trajectory is).

Neural network training

The process of neural network training can be described as minimization of some function *C* of several variables. Where *C* is the cost function of the neural network, and weights and biases are the parameters of the function that are needed to be selected to find its minimum. To find this minimum, a gradient descent algorithm in multidimensional space is used.

The cost function determines the difference between the expected activation values of the last layer and the actual values of the training set. It serves as a measure of the extent to which the neural network qualitatively classifies the input data. And is determined on the basis of (3). In our case, it will show the difference between the expected probabilities of how effective different learning paths will be for a particular vector x_i and actual probabilities y_i

$$C = \frac{1}{K}\sum_{i=1}^{K}\left\| y_i - a_i^{L}(x_i) \right\|^2 \qquad (3)$$

where *K* is the total number of elements in the training set.

In the course of the gradient descent algorithm, the gradient of the cost function is iteratively calculated, which consists of partial derivatives of the value function relative to each of the weights from a certain neuron j to a certain neuron i and the displacements of each of the neurons**.** Relative to this gradient, new values for each of the weights and offsets are being calculated.

Training occurs until the difference in the value of the cost function between individual iterations exceeds a certain value ε or after a certain number of iterations.

Conclusions. Within this scientific work, the general structure of the intellectual technology for optimization of the learning process using learning management systems was developed.

The described technology includes several components.

- A description of the domain area to which the scientific work is devoted, namely the process of teaching students using the project approach within the LMS.
- A description of the architecture of the recommendation system based on an artificial deep neural network of direct propagation, which will accept information about the student and the subjects that he wants to study, and will give the most effective learning trajectory as an output;
- A description of the process of creating training data for a deep neural network based on multiagent modeling.
- A description of the multi-agent model based on Markov processes and the method of value iteration.
- A description of the list of agents and the runtime environment that coordinates their work.

The developed software will help to provide effective recommendations for students, reducing the amount of time and money spent on training, as well as increasing the average level of grades received.

Further work is related to the actual implementation of a deep neural network, the development of a multi-agent model for training a neural network on a training set, and the implementation of a recommendation information system in existing training processes.

References

1. Burke John J., Tumbleson Beth E. *Learning Management Systems: Library Technology Reports.* American Library Assoc., 2016. 42 p.
2. Forrester J. W. *Principles of Systems*. MIT Press, 1968. 391 p.
3. Kotsiantis S. Supervised Machine Learning: A Review of Classification Techniques. *Informatica*. 2007. Vol. 31, iss. 3. P. 249–268.
4. Romero C., Ventura S. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews)*. 2010. Vol. 40, iss. 6. P. 601–618.
5. Conati C., Gertner A. S., VanLehn K. (). Using Bayesian Networks to Manage Uncertainty in Student Modeling. *User Modeling and User-Adapted Interaction*. 2002. Vol. 12(4). P. 371–417.
6. Mizoguchi R., Bourdeau J. Using Ontological Engineering to Overcome Common AI-ED Problems. *International Journal of Artificial Intelligence in Education*. 2000. Vol. 11, P. 107–121.
7. Wooldridge M. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 2009. 484 p.
8. Uhrmacher A. M., Weyns D. *Multi-Agent Systems: Simulation and Applications* (Computational Analysis, Synthesis, and Design of Dynamic Systems). CRC Press; 2009. 582 p.
9. Cervenka R., Trencansky I. *The Agent Modeling Language – AML: A Comprehensive Approach to Modeling Multi-Agent Systems (Whitestein Series in Software Agent Technologies and Autonomic Computing)*. Birkhäuser, 2007. 366 p.
10. Taylor S. *Agent-based Modeling and Simulation*. Palgrave Macmillan, 2014. 327 p.
11. Gacovski Zoran. *Artificial Neural Networks.* Arcler Press LLC, 2017. 224 p.
12. da Silva I. N., Spatti D. H. Andrade R., Liboni L. H. B., dos Reis Alves S. F. *Artificial Neural Networks: A Practical Course*. Springer, 2016. 327 p.
13. Schalkoff R. J. *Artificial Neural Networks.* McGraw-Hill College, 1997. 448 p.

14. *Markov Decision Processes. Course Notes*. Carnegie Mellon University, 2019. ca. 20 p. URL: https://www.cs.cmu.edu/~15281/coursenotes/mdps/index.html (accessed at 22.04.2025)
15. Thomas G. W. *Markov Decision Processes*. Stanford AI Lab Course Notes, ca. 2019. 11 p. URL: https://ai.stanford.edu/~gwthomas/notes/mdps.pdf (accessed at 12.04.2025)
16. Dai P, Goldsmith J. *Topological Value Iteration Algorithm for Markov Decision Processes*. URL: https://www.cs.uky.edu/~goldsmit/papers/tvi.pdf (accessed at 19.04.2025)

References (transliterated)

1. Burke John J., Tumbleson Beth E. *Learning Management Systems: Library Technology Reports.* American Library Assoc., 2016. 42 p.
2. Forrester J. W. *Principles of Systems*. MIT Press, 1968. 391 p.
3. Kotsiantis S. Supervised Machine Learning: A Review of Classification Techniques. *Informatica*. 2007, vol. 31, iss. 3, pp. 249–268.
4. Romero C., Ventura S. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews)*. 2010, vol. 40, iss. 6, pp. 601–618.
5. Conati C., Gertner A. S., VanLehn K. (). Using Bayesian Networks to Manage Uncertainty in Student Modeling. *User Modeling and User-Adapted Interaction*. 2002, vol. 12(4), pp. 371–417.
6. Mizoguchi R., Bourdeau J. Using Ontological Engineering to Overcome Common AI-ED Problems. *International Journal of Artificial Intelligence in Education*. 2000, vol. 11, pp. 107–121.
7. Wooldridge M. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 2009. 484 p.
8. Uhrmacher A. M., Weyns D. *Multi-Agent Systems: Simulation and Applications* (Computational Analysis, Synthesis, and Design of Dynamic Systems). CRC Press; 2009. 582 p.
9. Cervenka R., Trencansky I. *The Agent Modeling Language – AML: A Comprehensive Approach to Modeling Multi-Agent Systems (Whitestein Series in Software Agent Technologies and Autonomic Computing)*. Birkhäuser, 2007. 366 p.
10. Taylor S. *Agent-based Modeling and Simulation*. Palgrave Macmillan, 2014. 327 p.
11. Gacovski Zoran. *Artificial Neural Networks.* Arcler Press LLC, 2017. 224 p.
12. da Silva I. N., Spatti D. H. Andrade R., Liboni L. H. B., dos Reis Alves S. F. *Artificial Neural Networks: A Practical Course*. Springer, 2016. 327 p.
13. Schalkoff R. J. *Artificial Neural Networks.* McGraw-Hill College, 1997. 448 p.
14. *Markov Decision Processes. Course Notes*. Carnegie Mellon University, 2019. ca. 20 p. Available at: https://www.cs.cmu.edu/~15281/coursenotes/mdps/index.html (accessed at 22.04.2025).
15. Thomas G. W. *Markov Decision Processes*. Stanford AI Lab Course Notes, ca. 2019. 11 p. Available at: https://ai.stanford.edu/~gwthomas/notes/mdps.pdf (accessed at 12.04.2025).
16. Dai P, Goldsmith J. *Topological Value Iteration Algorithm for Markov Decision Processes*. Available at: https://www.cs.uky.edu/~goldsmit/papers/tvi.pdf (accessed at 19.04.2025).

Received 15.05.2025

УДК 004.8:37.018.43:004.738.5

В. Є. СОКОЛ, кандидат технічних наук, доцент, науковий співробітник – дослідницька група з технологій навчання Рейнсько-Вестфальського технічного університету, м. Ахен; e-mail: sokol@cs.rwth-aachen.de; ORCID: https://orcid.org/0000-0002-4689-3356

М. Д. ГОДЛЕВСЬКИЙ, доктор технічних наук, професор, Національний технічний університет «Харківський політехнічний інститут», директор інституту комп'ютерних наук та інформаційних технологій; м. Харків, Україна; e-mail: Mykhailo.Hodlevskyi@khpi.edu.ua; ORCID: https://orcid.org/0000-0003-2872-0598

М. О. БІЛОВА, кандидат технічних наук, доцент, Національний технічний університет «Харківський політехнічний інститут», доцент кафедри програмної інженерії та інтелектуальних технологій управління, м. Харків, Україна; e mail: Mariia.Bilova@khpi.edu.ua; ORCID: https://orcid.org/0000-0001-7002-4698

Р. А. ТУПКАЛЕНКО, Національний технічний університет «Харківський політехнічний інститут», аспірант, м. Харків, Україна; e-mail: Roman.Tupkalenko@cs.khpi.edu.ua; ORCID: https://orcid.org/0009-0008-7268-2393

ІНТЕЛЕКТУАЛЬНА ТЕХНОЛОГІЯ ОПТИМІЗАЦІЇ ПРОЄКТНОГО ПІДХОДУ ДО НАВЧАННЯ СТУДЕНТІВ ІЗ ВИКОРИСТАННЯМ СИСТЕМ УПРАВЛІННЯ НАВЧАННЯМ

Дана робота присвячена розробці рекомендаційної системи, яка дасть можливість ефективно будувати траєкторії навчання для студентів, які навчаються у вищих навчальних закладах з використанням систем управління навчанням. Ядром рекомендаційної системи буде штучна глибока нейронна мережа прямого розповсюдження, що приймає на вхід інформацію про студента і предмет, який він повинен вивчити, і пропонує на виході найбільш ефективну для нього траєкторію навчання. Навчання нейронної мережі відбувається на даних підготовлених за допомогою мультиагентного моделювання. Своєю чергою, доменна область декомпонована на окремі складові та в процесі мультиагентного моделювання була представлена у вигляді агентів і середовища, в рамках якого вони комунікують між собою. Предметом дослідження є моделювання процесу навчання в системах управління навчанням. Метою дослідження є оптимізація процесу навчання студентів, які навчаються з використанням систем управління навчанням. Проаналізовано та вивчено предметну область, розроблено архітектуру рекомендаційної системи, розроблено архітектуру мультиагентної системи, розроблено математичну модель взаємодії агентів. Розглянуто метод моделювання процесу навчання з використанням систем управління навчанням. Для досягнення поставлених цілей необхідно розв'язати дві раніше описані задачі, а саме: підготувати набір тренувальних даних за допомогою мультиагентного моделювання, а також розробити й навчити на цьому наборі даних рекомендаційну систему на основі штучної глибокої нейронної мережі. Після виконання всіх поставлених задач дослідження очікується, що процес навчання студентів у системі управління навчанням буде оптимізовано з погляду часу та ресурсів, що витрачаються на навчання, а середній рівень здобутих знань зросте.

Ключові слова: мультиагентне моделювання, система управління навчанням, штучна нейронна мережа, алгоритм зворотного розповсюдження помилки, навчальний процес, ефективна траєкторія навчання

Повні імена авторів / Author's full names

Автор 1 / Author 1: Сокол Володимир Євгенович / Sokol Volodymyr Yevgenovych
Автор 2 / Author 2: Годлевський Михайло Дмитрович / Godlevskyi Mykhaylo Dmytrovych
Автор 3 / Author 3: Білова Марія Олексіївна / Bilova Mariia Oleksiivna
Автор 4 / Author 4: Тупкаленко Роман Андрійович/ Tupkalenko Roman Andriyovichs