Research article
DOI: https://doi.org/10.18721/JCSTCS.17303
UDC 004.8:[62-5+004.382.2]

SUPERCOMPUTER RESOURCES MANAGEMENT USING MACHINE LEARNING METHODS UNDER CONSTRAINTS

V.A. Muliukha ✉ , *A.V. Vostrov*, *D.E. Motorin*, *V.V. Glazunov*

Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russian Federation

✉ vladimir.muliukha@spbstu.ru

Abstract. Artificial intelligence and machine learning technologies are among the most promising in the field of computer science. They make it possible to obtain solutions to problems that until recently were the exclusive prerogative of humans. However, when solving practical problems, it is necessary to implement machine learning models taking into account the restrictions on available resources. Such resources can be both computational and temporary (i.e. the problem must be solved in a certain time and using certain hardware, most often it is about various mobile platforms), and informational, when it comes to small, censored, incomplete or noisy data. The paper examines machine learning methods used to solve practical problems in application areas, such as comparing the shape of three-dimensional objects and intellectualizing resource dispatching, within the framework of the concept of "Supercomputer for AI and AI for a Supercomputer". In the field of solving problems with limited data volume, a method is proposed that allows training a multilayer neural network using an ultra-small training sample to solve the problem of quantitatively assessing the proximity of the shape of arbitrary three-dimensional objects. In the field of applying machine learning models with limited resources, a method has been developed that ensures asynchronous operation of the machine learning model and the executable process, which allows for the effective use of machine learning methods under constraints.

Keywords: machine learning, resource management, supercomputer, constraints, neural networks

Acknowledgements: The research was partially financially supported by the Ministry of Science and Higher Education of the Russian Federation within the framework of the state assignment "Development and research of machine learning models for solving fundamental problems of artificial intelligence in the fuel and energy complex" (FSEG-2024-0027). The results of the work were obtained using the computational resources of the shared-use center "Polytechnic Supercomputer Center" of Peter the Great St. Petersburg Polytechnic University (https://scc.spbstu.ru).

Citation: Muliukha V.A., Vostrov A.V., Motorin D.E., et al. Supercomputer resources management using machine learning methods under constraints. Computing, Telecommunications and Control, 2024, Vol. 17, No. 3, Pp. 32–41. DOI: 10.18721/JCSTCS.17303

Научная статья
DOI: https://doi.org/10.18721/JCSTCS.17303
УДК 004.8:[62-5+004.382.2]

УПРАВЛЕНИЕ ВЫЧИСЛИТЕЛЬНЫМИ РЕСУРСАМИ СУПЕРКОМПЬЮТЕРОВ МЕТОДАМИ МАШИННОГО ОБУЧЕНИЯ В УСЛОВИЯХ ОГРАНИЧЕНИЙ

В.А. Мулюха ✉ , *А.В. Востров*,
Д.Е. Моторин, *В.В. Глазунов*

Санкт-Петербургский политехнический университет Петра Великого,
Санкт-Петербург, Российская Федерация

✉ vladimir.muliukha@spbstu.ru

Аннотация. Технологии искусственного интеллекта и машинного обучения являются одними из самых перспективных в области компьютерных наук. Они позволяют получить решение задач, которые до недавнего времени были исключительно прерогативой человека. Однако при решении практических задач приходится реализовывать модели машинного обучения с учетом ограничений на доступные ресурсы, при этом, ресурсы могут быть как вычислительные и временные (т.е. задача должна быть решена за определенное время и с использованием определенного аппаратного обеспечения, чаще всего речь идет о различных мобильных платформах), так и информационные, когда речь идет о малых, цензурированных, неполных или зашумленных данных. В работе рассматриваются методы машинного обучения, используемые для решения практических задач в прикладных областях, таких как сравнение формы трехмерных объектов и интеллектуализация диспетчеризации ресурсов, в рамках концепции «Суперкомпьютер для ИИ и ИИ для суперкомпьютера». В области решения задач при наличии ограничений на объем данных предложен метод, который позволяет осуществить обучение многослойной нейронной сети с использованием сверхмалой обучающей выборки, для решения задачи количественной оценки близости формы произвольных трехмерных объектов. В области применения моделей машинного обучения при наличии ограничений на используемые ресурсы разработан метод, обеспечивающий асинхронную работу модели машинного обучения и исполняемого процесса, что позволяет эффективно использовать методы машинного обучения в условиях ограничений.

Ключевые слова: машинное обучение, диспетчеризация ресурсов, суперкомпьютер, ограничения, нейронные сети

Финансирование: Исследование выполнено при частичной финансовой поддержке Министерства науки и высшего образования Российской Федерации в рамках государственного задания «Разработка и исследование моделей машинного обучения для решения фундаментальных задач искусственного интеллекта в топливно-энергетическом комплексе» (FSEG-2024-0027). Результаты работы получены с использованием вычислительных ресурсов центра коллективного пользования «Суперкомпьютерный центр “Политехнический”» Санкт-Петербургского политехнического университета Петра Великого (https://scc.spbstu.ru).

Для цитирования: Muliukha V.A., Vostrov A.V., Motorin D.E., et al. Supercomputer resources management using machine learning methods under constraints // Computing, Telecommunications and Control. 2024. Т. 17, № 3. С. 32–41. DOI: 10.18721/JCSTCS.17303

Introduction

Machine learning is one of the fastest growing fields in modern science. The inductive approach in machine learning is a process, in which a computing device analyzes the available data and the desired output, finds correlations between them, and builds a function that relates the input data to the output of the model. Thus, machine learning models are automatically trained to solve classification and regression problems, make predictions and decisions in various fields, such as medicine, finance, manufacturing, etc.

Traditionally, one of the main advantages of machine learning is its ability to process large amounts of data that were previously unavailable for analysis. However, like any other technology, machine learning has its limitations and disadvantages. Some of these include the problem of overfitting, where the model begins to learn from spurious correlations, random noise and errors, rather than from real patterns. In addition, a problem is the availability of small data for training, when training sample sizes are not enough to identify statistically significant patterns. The development of large neural network models, in particular large language models, reveals the problem of performance and cost of running the models.

The aim of the work is to develop methods for applying machine learning models under constraints. The paper considers the results of research projects carried out by the team of the Higher School of Artificial Intelligence Technologies of Peter the Great St. Petersburg Polytechnic University. A distinctive feature of these works is the use of machine learning models under significant constraints. In the first case, the constraint was the small size of the training sample and the requirement to provide the user with the results of the comparison within 15 seconds after the corresponding request was received. In the second case, it was necessary to use predictive analytics models for task execution time on a supercomputer in a mode transparent to the user, taking into account the possibility of batch task launch. To solve this problem, an approach was proposed using simple models based on ensembles of decision trees operating asynchronously in the background.

There are a number of approaches to solving the problem of using machine learning models under resource constraints. In particular, if there are restrictions on the size of the training sample, you can use regularization methods, which can reduce the influence of the noise and increase the accuracy of the model [1]. Another approach is the use of ensemble methods [2], which allow replacing one complex model that has a large number of parameters and requires a large training sample, with many simple models trained on different data or on different combinations of input data. Moreover, each of these simple models has a small number of parameters and can be trained on a relatively small sample. Errors in solving a problem made by individual simple models are leveled out using collegial decision-making methods, thereby increasing the accuracy and reliability of the model [3]. You can also increase the sample size using specialized machine learning models, for example, Siamese neural networks, which learn not from the examples themselves, but from their pairs, while the number of pairs grows in proportion to the square of the sample elements and allows training even on a small number of examples [4]. Another method of increasing the volume of initial data is data augmentation. It is a method of generating new data by combining the original data, as well as by adding various noise. In addition, the approach used in augmentation is used to increase the accuracy of the output of various generative models that generate a response based on the user's request. Adding additional contextual data to a request is called Retrieval Augmented Generation (RAG) [5] and is one of the most widely used methods for improving the performance of the large language models.

Machine learning models, especially those built on the basis of neural networks, are quite demanding in terms of the amount of computational resources required both during the training process and at the stage of model operation. There are several basic methods for reducing the computational complexity of machine learning models. One of the main approaches is to try to replace large and complex models with the simpler ones. In this case, methods, such as knowledge distillation [6] are distinguished, in which the nodes of the original neural network that make a small contribution to the result are reset and excluded from the model. In this way, it is possible to reduce the amount of calculations, while maintaining comparable accuracy of the model. The same method is used in the models of explainable artificial intelligence, when the task is to leave only a small number of nodes that make the greatest contribution to the formation of the result of the model, thus highlighting the key elements in the source data, because of which a specific decision was made.

In addition, one solution is to move to other types of machine learning models that are less computationally intensive, such as forests or deep forests. The Higher School of Artificial Intelligence Technologies researchers proposed methods for combining forests and neural networks [7]. Decision trees carry out the basic classification of the data, spending less resources than the neural networks, since at each step a decision is made based on only one of the attributes. The use of neural networks in the leaves of such trees adds flexibility to the approach and allows us to model complex functions that are not available to classical decision trees.

Further, the article discusses practical problems solved at Peter the Great St. Petersburg Polytechnic University under resource restrictions within the framework of the concept of "Supercomputer for AI and AI for a Supercomputer".

Solving the problem of comparing objects under restrictions on initial data volume

One of the practical problems solved under restrictions on the volume of the training sample was the development of a system for comparing the shape of three-dimensional objects for Rospatent [8]. The task was calculating the semantic proximity function or determining the similarity of digital three-dimensional model using neural networks. Comparison of objects in terms of their semantic proximity in the theory of machine learning refers to a class of tasks that are united under the name of distance metric learning [9]. The main idea underlying the solution to the object comparison problem is that the distance between semantically similar objects should be less than the distance between semantically different objects.

Thus, if there are two training sample vectors $x_i \in \mathbf{R}^m$ and $x_j \in \mathbf{R}^m$, then the distance $d\left(x_i, x_j\right)$ should be minimized, if x_i and x_j are semantically close objects, and this distance should be maximized, if x_i and x_j are semantically distant. The most common and popular distance function is the quadratic Mahalanobis distance $d_M^2\left(x_i, x_j\right)$, which is defined for two vectors as:

$$d_M^2\left(x_i, x_j\right) = \left(x_i - x_j\right)^T M \left(x_i - x_j\right),$$

where $M \in \mathbf{R}^{m \times m}$ is a symmetric positive semidefinite matrix (its eigenvalues are non-negative). The Mahalanobis distance, unlike the Euclidean distance, is scale invariant and allows having correlations between the input data.

The distance matrix is symmetric and positive definite, which allows it to be represented in the form:

$$S^{-1} = W^T W,\ W \in \mathbf{R}^{p \times M},\ p \le M.$$

Then the Mahalanobis distance can be rewritten as:

$$d_M^2\left(x_i, x_j\right) = \left\| W_{x_i} - W_{x_j} \right\|_2 .$$

This equality means that calculating the Mahalanobis distance is equivalent to finding such a linear transformation W that each vector is mapped to a lower-dimensional space, in which the Euclidean distance is equal to the Mahalanobis distance in the original space. Thanks to it, a neural network can be built that functions similarly to the calculation of the Mahalanobis distance, when the activation functions of the neurons are linear. In fact, a neural network allows one to obtain a nonlinear analogue of the Mahalanobis distance by using a combination of linear and nonlinear activation functions.

However, for effective operation, the number of the training sample for neural networks must exceed several times the number of connection weights (training parameters). To solve this problem, the project used a Siamese neural network, which is trained on pairs of data. Thus, having only 600 initial 3D models allowed us to train a Siamese neural network consisting of two six-layer fully connected neural networks containing 96, 150, 120, 80, 40, 32 and 16 neurons, respectively. The second feature of the Siamese neural network is the possibility of using it in so-called one-shot learning (learning from one example) or few-shot learning (learning using several examples). Therefore, it learns in a situation, where there is only one or a few examples in one class and the initial sample is poorly balanced. This is possible, because the Siamese neural network is trained not on individual examples, but on pairs of examples. Hence, it is not the number of objects themselves, but the number of possible pairs of objects that are the elements for learning.

The second challenge addressed by this project was ensuring the required level of performance, when comparing shapes of 3D models. The models themselves came into the system in the form of wireframe models of arbitrary size (up to 100 MB per model). At the same time, the comparison of a newly received model with all existing ones in the database should not, according to customer requirements, last more than 15 seconds. Due to restrictions on the hardware and the requirement to ensure the functioning of the system in the customer's circuit, a transition from a direct comparison of three-dimensional objects to a comparison of their shape descriptors was proposed, which ensures the identification of significant features of the appearance of three-dimensional objects.

To form a descriptor of the shape of a three-dimensional object, a three-dimensional modification of the chord method was used, which made it possible to move from a three-dimensional model consisting of an arbitrary number of points to a diagram of the lengths of segments connecting points on the surface of the model. The descriptors themselves were generated in an asynchronous mode, while the model entered the system, and during the expert's work, only the calculated descriptor of the new model was compared with the previously calculated descriptors of other models in the database. The system provides the ability to change the descriptor for comparison. If a new descriptor appears, for each incoming model, both old and new descriptors are calculated, until all models in the database are recalculated. In this case, before a complete recalculation, the models are compared using old descriptors, about which the relevant information is displayed to the user, and after the recalculation, only new descriptors are used and the old ones are no longer calculated.

The transfer of data processing procedures to the background allowed us to reduce significantly the hardware requirements of the developed system, while also ensuring the required level of performance, which is one of the key elements of the "Supercomputer for AI" approach.

Using machine learning to improve supercomputer performance

As part of the "AI for a supercomputer" approach, Peter the Great St. Petersburg Polytechnic University developed methods for increasing the operating efficiency or performance of a supercomputer, including using artificial intelligence technologies. Research into the prospects for using various methods of managing computational resources has been actively conducted in different countries over the past few years. At the same time, the developed mathematical, heuristic and metaheuristic methods for optimizing task parameters are used to reduce the energy, resource and time costs of performing computational tasks on supercomputers.

A two-criteria hybrid workflow scheduling algorithm in cloud computing is presented in [10]. The proposed Heterogeneous Gravity Search Algorithm (HGSA) is a hybrid of the popular metaheuristic Gravity Search Algorithm (GSA) and the equally popular heuristic Heterogeneous Earliest Finish Time (HEFT) for scheduling workflow applications. The results show that HGSA performs better than Hybrid Genetic Algorithm by 14%, GSA by 20%, and HEFT by 35% in terms of fitness value. The results were obtained using analysis of variance (ANOVA) statistical test. In our work, this algorithm turns out to be ineffective, since it significantly underestimates the expected value of the task execution time.

The problem of optimizing the scheduling of large applications with parallel processes in heterogeneous computing systems, such as hybrid clouds, is NP-complete and is decisive for Quality of Service (QoS). The article [11] considers the formulation of the optimization problem of a large number of homogeneous parallel batches of problems, which are a bottleneck. The planning problem is solved by a cooperative game algorithm with two-criteria optimization in terms of execution time and economic costs and with two restrictions – on network throughput and memory size. The proposed algorithm showed better results in the execution time of computational tasks compared to such greedy approaches as G-Min-Min, G-Max-Min or G-Sufferage, and the efficiency increases with the size of the infrastructure. It should be noted that this algorithm has sufficient accuracy, but at the same time has extremely low performance, because the entire set of tasks in the queue is optimized. In conditions of receipt of tens or even hundreds of new tasks per second during batch setting, this method is not applicable to our problem.

A dynamic game-theoretic approach to solving problems of scheduling scientific computing in cloud systems is considered in [12]. A multi-objective workflow scheduling framework is proposed to reduce cloud creation times and costs, while maximizing load distribution among heterogeneous cloud virtual machines. The performance of the developed system was tested on randomly generated templates of scientific workflows and on data from third-party commercial clouds. This approach, like the previous one, is intended to optimize the distribution of a finite set of tasks and cannot be used effectively in the event of new ones arriving. Similar optimization methods are used in [13, 14].

Evolutionary and swarm algorithms are used in problems of scheduling m jobs for n resources for cloud computing and allow obtaining suboptimal solutions. In [15], the multi-objective bacteria foraging optimization algorithm (MOBFOA), which is a modification of the BFOA algorithm for solving multicriteria planning problems with Pareto optimal front. The modification consists of selecting bacterial positions from dominant and non-dominant fronts to obtain a diversity of solutions, which increases the accuracy and the speed of convergence by introducing an adaptive chemotactic step size. The performance of the proposed algorithm was compared in terms of the speed of convergence to the Pareto optimal front and the distribution of solutions in space, with NSGA-II and OMOPSO. The use of these algorithms requires the presence of software agents on all computing nodes, which significantly complicates its implementation on a functioning supercomputer. A similar approach is used in [16].

The growth in the performance of supercomputer systems is associated with an increase in electricity consumption. Balancing energy consumption and performance of computational resources is discussed in [17]. The authors proposed a new scheduling algorithm based on energy-aware duplication – novel energy-aware duplication-based scheduling (NEADS). During the duplication process, they are counted as favorite predecessor (FP), and the next FP. If the FP does not meet the duplication condition, the predecessor FP will not be checked. The second FP is then evaluated. This allows you to avoid replication of indirect tasks on the same processor. Accordingly, additional communication and computation costs can be reduced. Experimental results show that for synthetic applications, NEADS is effective in reducing energy consumption. This approach can be effectively used in systems that process a large number of small tasks. In case of the SCC Polytechnic, planning is performed at the node level as a whole and we can determine the type of the executed task indirectly only by meta-features without access to the executable code of the task.

Accurately predicting the task execution time in a complex high-performance dynamic computing environment is a complex task, subject to a large number of constraints and parameters. One single strategy is not suitable for solving all types of heterogeneous problems. To solve this problem, a joint forecasting model using multiple strategies multi-strategy collaborative prediction mode (MSCPM) was proposed in [18] to perform online tasks in runtime mode. The accuracy of forecasts was assessed using the concept "Ensuring forecast accuracy" introduced by the authors. Experimental results based on a cluster computing environment show that the predictions of the joint prediction model are consistent with the optimal predictions within the results provided by single strategies, and MSCPM provides improved accuracy in predicting the execution time of online tasks. In our work, we proposed an approach using different prediction models for different classes of computational tasks. At the same time, unlike [18], in our work, in addition to the parameters of the task itself, we also use accumulated statistical data on users and users' groups that run the analyzed task. Using additional data on users allows us to increase the accuracy of task execution time estimation by 10–12%.

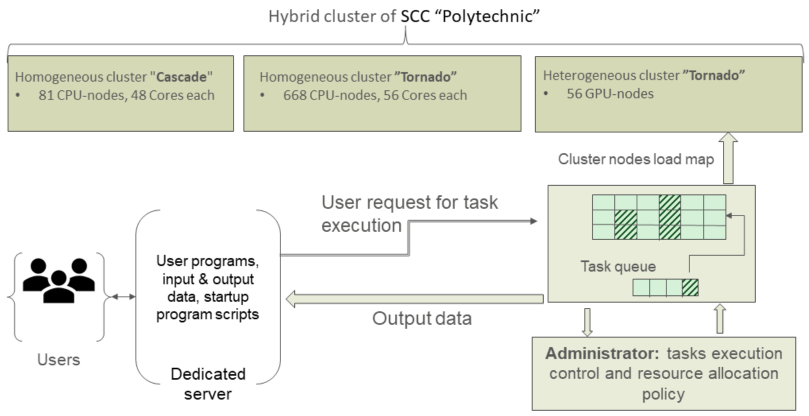
Hybrid cluster of SCC "Polytechnic"
Homogeneous cluster "Cascade"
• 81 CPU-nodes, 48 Cores each
Homogeneous cluster "Tornado"
• 668 CPU-nodes, 56 Cores each
Heterogeneous cluster "Tornado"
• 56 GPU-nodes
Cluster nodes load map
User request for task execution
Task queue
User programs, input & output data, startup program scripts
Dedicated server
Users
Output data
Administrator: tasks execution control and resource allocation policy

Fig. 1. Current scheme for launching a task by a user at the SCC Polytechnic

Our project involved an unbalanced sample of task execution times as part of the task of increasing dispatch efficiency at the SCC Polytechnic. SCC is a hybrid computing system consisting of several computing clusters with different architectures (homogeneous and heterogeneous), located in a single information and computing field, as well as a shared storage Luster, with a capacity of about 1 PB, allowing file exchange between different computing systems.

Registered users gain access to computational resources from the machine login1.hpc.spbstu.ru. Login is carried out using the SSH protocol (terminal client).

Cluster resources are managed using the Slurm software package [19]. The principle of its operation is as follows: the user requests a certain resource (processor cores, memory, etc.), placing his task or tasks in the queue. The system, based on the user's priorities and the current filling of the queue, selects the moment to launch the task. A queue is a sequence of tasks that must be solved on a specific computational resource (group of nodes). Moreover, each node at the current time can be occupied with only one task of one user. Thus, the node is given exclusive use to the task hosted on it, and other tasks on the busy node will not be executed.

The current scheme for launching a task at the SCC Polytechnic SCC is shown in Fig. 1.

The low efficiency of dispatching is determined, firstly, by the low quality of data received by the dispatcher and used by it in the process of its work. Most users either incorrectly indicate the expected task completion time or do not indicate it at all, leaving the default parameter in the metadata.

As part of the project, an intelligent module was developed and implemented that predicts the task execution time using data on previously performed calculations, in the context of the user running this task.

The scheme for launching tasks at the SCC Polytechnic using an intelligent module is shown in Fig. 2.

When developing the architecture of the new task launch system, the following principles were used:

• the implementation of the intelligent module should be carried out in a "transparent" mode for the user, that is, the algorithm and sequence of user actions as a result of the implementation of the intelligent module should not change in any way;

• as a result of system actions, user tasks should not be removed from execution in the event of an erroneously low calculation time prediction;

• when choosing the time to use to schedule a task, Slurm uses the minimum time between predicted and specified by the user.

It was also necessary to take into account the specifics of queuing tasks, namely batch scheduling of tasks for execution. Using machine learning models to analyze all incoming tasks in a batch in real time requires significant computational resources that can be more efficiently used to solve user problems. To implement the principle of transparency for the user, it was decided not to introduce any additional

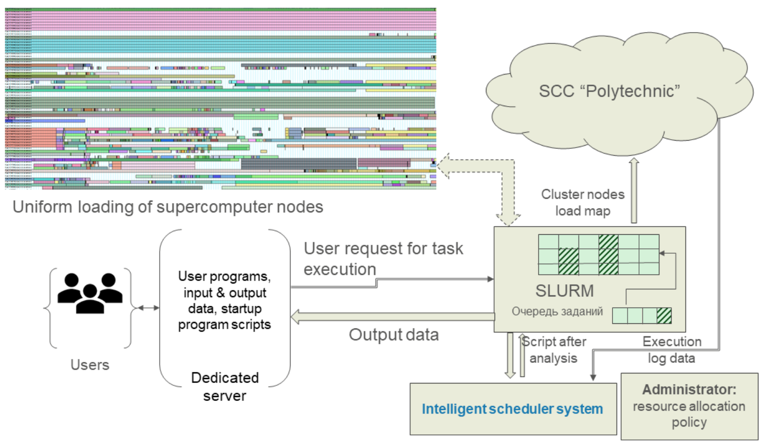
Uniform loading of supercomputer nodes
SCC "Polytechnic"
Cluster nodes load map
User request for task execution
User programs, input & output data, startup program scripts
Dedicated server
Users
SLURM
Очередь заданий
Output data
Script after analysis
Execution log data
Intelligent scheduler system
Administrator: resource allocation policy

Fig. 2. Scheme for launching a task by a user at the SCC Polytechnic using an intelligent module

elements between the user and the dispatcher. Thus, the intelligent module performs asynchronous processing of tasks that are already in the Slurm database, i.e. are waiting in the execution queue. At the same time, by adjusting the execution time of the user's task, the intelligent module changes the time value of tasks already in the queue, and the dispatcher, during a scheduled review of the queue, adjusts it based on the new time value and transfers some of the tasks, filling the free slots in the schedule. In this case, the intelligent module stores the initial time specified by the user, the value of the predicted time, and monitors the actual time for completing this task. If the predicted time is approaching and the task has not completed, then the intelligent module iteratively increases the specified execution time up to the value specified by the user when starting the task.

At the same time, the machine learning model itself, which predicts the task execution time, uses ensemble learning, in which several simple models, for example, small random forests, are trained on random subsamples of the original data sample. The use of ensemble models can significantly reduce the requirements for resources required at the model training stage, as well as speed up the training procedure itself.

Conclusion

Using machine learning in small sample settings is a complex task that requires careful data analysis and selection of optimal methods. However, thanks to the development of new methods and technologies, the possibilities of using machine learning in such cases are constantly expanding.

There are a number of techniques that allow machine learning models to be used even when there are limitations on the resources used, be it the amount of data used to train the model or the resources required to train or operate it. The most typical technique is to use specialized models that support few-shot methods learning or even one-shot learning, when the training procedure takes place using an extremely small amount of data. In addition, if appropriate, you can use preprocessing of the original data or post-processing of the obtained results, often using other machine learning models.

We have proposed a method based on the use of Siamese neural networks for estimating the shape similarity of three-dimensional objects, which allows training a multilayer neural network using an ultra-small training sample of less than 700 objects.

To improve the efficiency of using machine learning models in the presence of limitations on execution time and computational resources, when estimating the duration of solving a computational problem, a method for asynchronously applying a machine learning model and the Slurm dispatcher has been developed at the SCC Polytechnic. According to the results obtained during modeling, the use of machine learning methods allows reducing the average time a task spends in the queue by 10–15% and increasing the ratio of the number of solved problems to a given period of time. At the same time, the module does not affect the operation of the Slurm dispatcher itself.

The use of post-processing methods, in addition to increasing the efficiency of the machine learning model, can be used to implement mechanisms for interpreting (or explaining) the results of the model [20].

REFERENCES

1. **Boyd S.P., Vandenberghe L.** Convex Optimization. UK: Cambridge University Press, 2004.

2. **Domingos P.** A Unified Bias-Variance Decomposition and its Applications. International Conference on Machine Learning, 2000, Pp. 231–238.

3. **Breiman L.** Some infinity theory for predictor ensembles, 2000.

4. **Leal-Taixé L., Canton-Ferrer C., Schindler K.** Learning by tracking: Siamese CNN for robust target association. 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2016, Pp. 418–425. DOI: 10.1109/CVPRW.2016.59

5. **Lewis P., Perez E., Piktus A., Petroni F., Karpukhin V., Goyal N., Küttler H., Lewis M., Yih W.-t., Rocktäschel T., Riedel S., Kiela D.** Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, 2021, arXiv:2005.11401. DOI: 10.48550/arXiv.2005.11401

6. **Hinton G., Vinyals O., Dean J.** Distilling the Knowledge in a Neural Network, 2015, arXiv:1503.02531v1. DOI: 10.48550/arXiv.1503.02531

7. **Konstantinov A., Utkin L., Muliukha V.** LARF: Two-Level Attention-Based Random Forests with a Mixture of Contamination Models. Informatics, 2023, Vol. 10, No. 2, Article no. 40. DOI: 10.3390/informatics10020040

8. **Chernoskulova E.** 3D-sistema dlia Rospatenta [3D system for Rospatent]. Scientific Russia, Available: https://scientificrussia.ru/articles/3d-sistema-dlya-rospatenta (Accessed 01.07.2024). (In Russ.)

9. **Suárez J.L., García S., Herrera F.** A tutorial on distance metric learning: Mathematical foundations, algorithms and software, 2018, arXiv:1812.05944v3. DOI: 10.48550/arXiv.1812.05944

10. **Choudhary A., Gupta I., Singh V., Jana P.K.** A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing. Future Generation Computer Systems, 2018, Vol. 83, Pp. 14–26. DOI: 10.1016/j.future.2018.01.005

11. **Duan R., Prodan R., Li X.** Multi-Objective Game Theoretic Scheduling of Bag-of-Tasks Workflows on Hybrid Clouds. IEEE Transactions on Cloud Computing, 2014, Vol. 2, No. 1, Pp. 29–42. DOI: 10.1109/TCC.2014.2303077

12. **Wu L., Wang Y.** Scheduling Multi-Workflows over Heterogeneous Virtual Machines with a Multi-Stage Dynamic Game-Theoretic Approach. International Journal of Web Services Research (IJWSR), 2018, Vol. 15, No. 4, Pp. 82–96. DOI: 10.4018/IJWSR.2018100105

13. **Coello C.A.C., Pulido G.T., Lechuga M.S.** Handling multiple objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computation, 2004, Vol. 8, No. 3, Pp. 256–279. DOI: 10.1109/TEVC.2004.826067

14. **Wang Y., Liu H., Zheng W., Xia Y., Li Y., Chen P., Guo K., Xie H.** Multi-Objective Workflow Scheduling with Deep-Q-Network-Based Multi-Agent Reinforcement Learning. IEEE Access, 2019, Vol. 7, Pp. 39974–39982. DOI: 10.1109/ACCESS.2019.2902846

15. **Kaur M., Kadam S.** A novel multi-objective bacteria foraging optimization algorithm (MOBFOA) for multi-objective scheduling. Applied Soft Computing, 2018, Vol. 66, Pp. 183–195. DOI: 10.1016/j.asoc.2018.02.011

16. **Zhou X., Zhang G., Sun J., Zhou J., Wei T., Hu S.** Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT. Future Generation Computer Systems, 2019, Vol. 93, Pp. 278–289. DOI: 10.1016/j.future.2018.10.046

17. **Liang A., Pang Y.** A Novel, Energy-Aware Task Duplication-Based Scheduling Algorithm of Parallel Tasks on Clusters. Mathematical and Computational Applications, 2017, Vol. 22, No. 1, Article no. 2. DOI: 10.3390/mca22010002

18. **Tao M., Dong S., Zhang L.** A multi-strategy collaborative prediction model for the runtime of online tasks in computing cluster/grid. Cluster Computing, 2011, Vol. 14, Pp. 199–210. DOI: 10.1007/s10586-010-0145-4

19. **Zaborovsky V.S., Utkin L.V., Muliukha V.A., Lukashin A.A.** Improving Efficiency of Hybrid HPC Systems Using a Multi-agent Scheduler and Machine Learning Methods. Supercomputing Frontiers and Innovations, 2023, Vol. 10, No. 2, Pp. 104–126. DOI: 10.14529/jsfi230207

20. **Kovalev M.S., Utkin L.V., Kasimov E.M.** SurvLIME: A method for explaining machine learning survival models. Knowledge-Based Systems, 2020, Vol. 203, Article no. 106164. DOI: 10.1016/j.knosys.2020.106164

INFORMATION ABOUT AUTHORS / СВЕДЕНИЯ ОБ АВТОРАХ

Muliukha Vladimir A.
Мулюха Владимир Александрович
E-mail: vladimir.muliukha@spbstu.ru

Vostrov Alexey V.
Востров Алексей Владимирович
E-mail: vostrov_av@spbstu.ru

Motorin Dmitrii E.
Моторин Дмитрий Евгеньевич
E-mail: d.e.motorin@gmail.com

Glazunov Vadim V.
Глазунов Вадим Валерьевич
E-mail: neweagle@gmail.com

Submitted: 23.07.2024; Approved: 27.09.2024; Accepted: 04.10.2024.
Поступила: 23.07.2024; Одобрена: 27.09.2024; Принята: 04.10.2024.