Reduction of Neural Network Models in Intelligent Computer Systems of a New Generation

Aliaksandr Kroshchanka
Brest State Technical University
Brest, Belarus
Email: kroschenko@gmail.com

***Abstract*—In the article, an approach to reducing the tuning parameters of deep neural network models, which is based on the use of the pre-training method, is proposed. Examples of using this approach for model reduction are given for MNIST, CIFAR10, CIFAR100 datasets. Recommendations are given on the use of the proposed method in the context of integrating massive neural network models into the intelligent computer systems of a new generation based on the use of the OSTIS Technology.**

***Keywords*—deep neural networks, model reduction, RBM, CRBM, pre-training, transfer learning, hybrid intelligent systems**

I. Introduction

The development of hybrid intelligent systems that combine the use of various models and approaches is associated with integration difficulties. This problem can be successfully solved by using the OSTIS Technology [1], which makes it possible to develop such systems taking into account their semantic compatibility. An equally important factor is the quality of the particular components of the system.

Deep neural network models have recently become one of the most actively used components of hybrid systems. These models show impressive results in solving a wide variety of problems — recognition, detection, and segmentation of objects in photo and video images (for example, [2], [3]), generating annotations for photos, and generating images from a text description [4], generating texts of varying complexity ([5], [6]). Neural networks used to solve such problems contain millions and billions of adjustable parameters, as well as tens and hundreds of layers of neural network elements (Fig. 1).

Although artificial neural networks have recently been constantly expanding the boundaries of their application in various fields, it is the fact of the complexity of these models that gives rise to some conceptual problems and issues that hinder the process of widespread use of truly useful and effective neural networks.

Training remains the only possible way to specialize the model, since no pre-trained neural network can be used to effectively solve a specific problem. The only possibility in this case is additional training of this model on specific data.

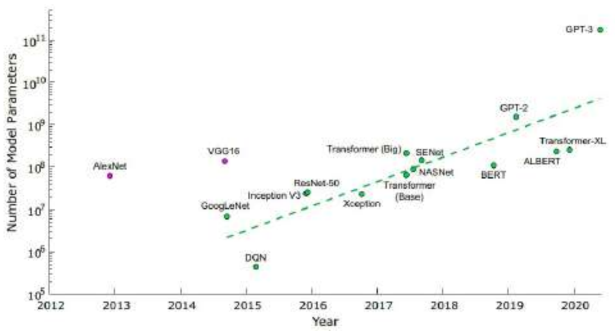

Figure 1. Evolution of parameters amount in deep neural networks [7]

However, training such “heavy” models is not a trivial problem. It is often associated with the risk of overfitting, which results in excellent fit of the model to the training dataset but poor generalization ability. Most often, overfitting occurs when using a small training dataset. Another problem is the size of the models used, which makes the learning process slow and resource intensive even with the use of modern technical means.

Thus, the most lightweight modification of the widely known and used Llama model requires 4–8 video accelerators like NVIDIA A100 with 80 GB of video memory for additional training on user data ([8], [9]). And this, unfortunately, is far from the limit for such models. Thus, the OPT model with 175 billion adjustable parameters already requires 992 video accelerators of the same type for training [10]. Thus, the problem of additional training becomes unattainable for ordinary users and researchers, becoming the prerogative of large laboratories and companies.

Thus, it becomes critical to reduce the number of parameters used without losing the quality of the neural network model. Ideally, it is required to achieve the ability to run models on portable devices with limited computing capabilities.

As possible solutions to both the problems of overfitting and unlimited growth in the number of parameters, the authors see the use of the pre-training procedure.

The following sections are organized as follows: in Section II, the problem of reducing deep neural networks in the context of intelligent computer systems of a new generation is described; in Section III, the proposed approach to reduce the number of parameters of neural network models is considered; in Section IV, the main practical results obtained are shown; finally, in Section V, the main conclusions on the proposed approach are represented and possible options for its development, including in the context of intelligent computer systems of a new generation, are described.

II. Problem formulation

It is known that neural networks have a certain degree of redundancy. Most often, this is conditioned by the use of an inconsistent number of model parameters and the size of the training dataset. If the number of adjustable parameters is greater than the volume of the training dataset, then the problems with the efficiency of training the model, or rather, with its generalizing ability, occur.

In addition, fully connected layers, in comparison with convolutional ones, contain a larger number of adjustable parameters, however, in computer vision problems, convolutional neural networks show significantly better results in terms of generalizing ability than fully connected ones. Thus, it is obvious that in fully connected networks with a larger number of adjustable parameters, they are used less optimally. It can be assumed that the specified "redundant" parameters can be discarded without a significant loss in the efficiency of the network.

Thus, the problem of reducing the neural network model is in identifying parameters that have small effect on the final result of the model and removing them. Moreover, such removal can be performed both logically (zeroing the corresponding weight coefficients and thresholds [11]) and architecturally (complete removal of a neural element if its weight coefficients and a threshold are equal to zero).

Therefore, **performing the reduction of the neural network model leads to its architectural change, reducing the number of neurons used on each layer.**

An important question that arises when performing the reduction concerns the very algorithm for discarding uninformative parameters. For current moment, several works have been proposed in which the authors reduce the dimension of neural network architectures (for example, [12], [13]).

Let us list the main advantages that the reduction of neural network models brings as separate components integrated into intelligent computer ostis-systems of a new generation.

Firstly, it becomes possible to automate the selection of the optimal architecture of the neural network, which prevents the core of the ostis-system from the formation of redundant and confusing rules, most of which are formulated not on the basis of theoretical studies but purely empirically. In the case of reduction, it is sufficient to determine the maximum upper value for the number of neurons of each layer, without the need to select these parameters during a series of experiments.

Secondly, the work of neural network models as components of the ostis-system is accelerated by reducing the number of parameters used. Frequently, deep models can be slow and resource-intensive, which affects the overall system uptime, resulting in serious delays, especially in the absence of powerful computing tools.

Thirdly, the process of additional training of neural network models on user datasets is automated, which could be insufficient in size when training a neural network from scratch, which would lead to the effect of overfitting.

Thus, model reduction expands the possibilities of integrating neural network models as components of ostis-systems, making the models themselves more adaptive, faster, and more efficient, and solving this problem is an urgent problem.

III. Proposed approach

Currently, there are two main approaches to pre-training deep neural networks: **I type** is based on unsupervised training of individual layers of the network, represented, for example, by restricted Boltzmann machines (RBM) [14], **II type** — on the use of special types of activation functions (ReLU), a large available training set, some special regularization techniques (for example, dropout), and special initialization of model parameters.

The selection of one or another approach to pre-train deep neural networks depends on the size of the training dataset. So, if the dataset is large, type II of pre-training is applied. Otherwise, type I of pre-training is used [15].

After performing the pre-training, an acceptable initial initialization of the parameters of the neural network model is achieved, which allows speeding up the additional training process, starting it with a smaller total error. In some cases, the nature of the data for additional training may change, differing from that used for pre-training (transfer learning). The goal of both the first and second types of pre-training is to achieve the ability to additionally train the model on small datasets without the appearance of the overfitting effect [16].

Thus, the process of training a deep neural network model with type I of pre-training consists of the following steps:

1) pre-training of the neural network by greedy layer-wise training, starting from the first layer (Fig. 2). Such training is carried out uncontrollably;
2) fine-tuning the parameters of the entire network using the backpropagation algorithm or the "wake-sleep" algorithm [17].

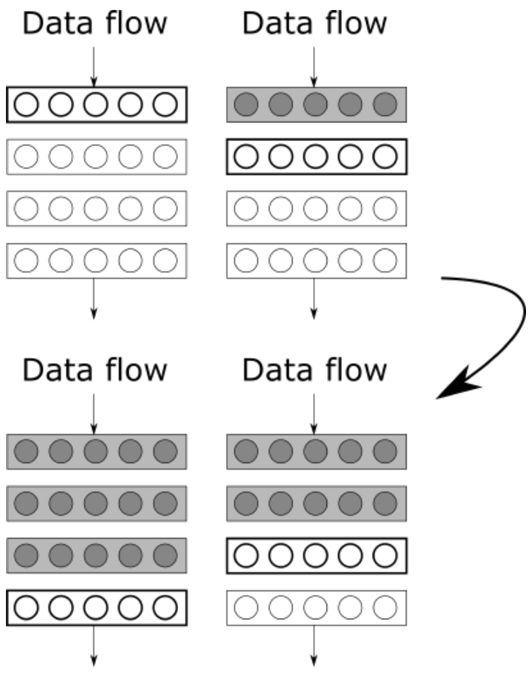

Figure 2. Greedy layer-wise algorithm

The proposed parameter reduction approach is based on the use of type I of pre-training proposed by G. Hinton (hereinafter referred to as the classical method) [18].

Let us give a brief description of this method. To do this, we consider the model of a restricted Boltzmann machine.

This model consists of two layers of stochastic binary neurons, which are interconnected by bidirectional symmetrical connections (Fig. 3). The input layer of neurons is called visible (layer X), and the output layer is called hidden (layer Y). The restricted Boltzmann machine can generate any discrete distribution if enough hidden layer neurons are used [19]. Let the visible layer contain *n* and the hidden layer contain *m* neurons.

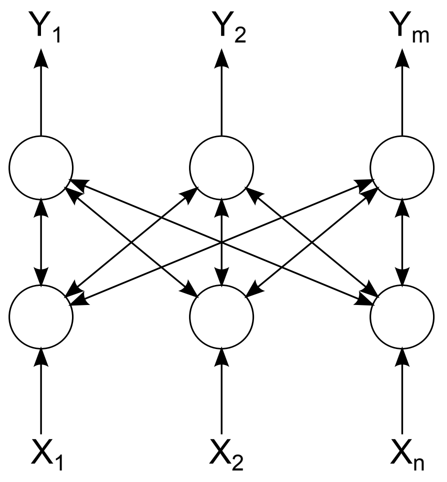

Figure 3. Restricted Boltzmann machine

The rules for online training of a restricted Boltzmann machine proposed in the classical method are as follows:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(x_i(0)y_j(0) - x_i(k)y_j(k))$$

$$T_i(t+1) = T_i(t) + \alpha(x_i(0) - x_i(k))$$

$$T_j(t+1) = T_j(t) + \alpha(y_j(0) - y_j(k))$$

where $x_i(0), x_i(k)$ — the original input data of the visible layer that are restored by the neural network, $y_i(0), y_i(k)$ — the original output data of the hidden layer that are restored by the neural network. Data recovery is performed using the Contrastive Divergence (CD-k) algorithm.

In practice, this algorithm is most often used for *k*=1.

The above rules are relevant for the case of a deep fully connected neural network, however, they can be easily reformulated for the case of a deep convolutional network. In this case, the individual layers of the deep model are treated as convolutional restricted Boltzmann machines (CRBMs) [20].

In this case, the rules will look as follows:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(x_i(0) \circledast y_j(0) - x_i(k) \circledast y_j(k))$$

$$T_i(t+1) = T_i(t) + \alpha(x_i(0) - x_i(k))$$

$$T_j(t+1) = T_j(t) + \alpha(y_j(0) - y_j(k))$$

where $\circledast$ denotes the convolution operation.

Thus, for a deep convolutional neural network, it is possible to combine several training options — with a representation in the form of CRBM (for the first convolutional layers) and in the form of RBM (for finalizing fully connected ones).

Previously, the authors proposed an approach that generalizes the classical approach and demonstrated its effectiveness for some problems (for example, [21]).

In the context of this approach, the learning rules are given, for the derivation of which the authors were guided by the idea of minimizing the total squared error of the network (the case of using CD-k):

$$E_s(k) = \frac{1}{2L}\left(\sum_{l=1}^{L}\sum_{j=1}^{m}(\Delta y_j^l(k))^2 + \sum_{l=1}^{L}\sum_{i=1}^{n}(\Delta x_i^l(k))^2\right)$$

where $\Delta y_j^l(k) = y_j^l(k) - y_j^l(0)$, $\Delta x_i^l(k) = x_i^l(k) - x_i^l(0)$, L — a dimension of the training dataset.

The RBM online training rules in accordance with the proposed approach for CD-k are as follows:

$$w_{ij}(t+1) = w_{ij}(t) - \alpha((y_j(k) - y_j(0))F'(S_j(k))x_i(k) + (x_i(k) - x_i(0))F'(S_i(k))y_j(0)),$$

$$T_i(t+1) = T_i(t) - \alpha(x_i(k) - x_i(0))F'(S_i(k)),$$

$$T_j(t+1) = T_j(t) - \alpha(y_j(k) - y_j(0))F'(S_j(k)).$$

For the CRBM case, the rules will take the form:

$$w_{ij}(t+1) = w_{ij}(t) - \alpha((y_j(k) - y_j(0))F'(S_j(k)) \circledast x_i(k) + (x_i(k) - x_i(0))F'(S_i(k)) \circledast y_j(0)),$$

$$T_i(t+1) = T_i(t) - \alpha(x_i(k) - x_i(0))F'(S_i(k)),$$

$$T_j(t+1) = T_j(t) - \alpha(y_j(k) - y_j(0))F'(S_j(k)).$$

It is possible to prove the identity of these learning rules to the classical ones by using neurons with a linear activation function.

Let us consider an approach for reducing the parameters of a fully connected neural network based on the use of pre-training procedure. The first and fourth stages of this procedure are equal to the stages of performing the first type of pre-training. During the execution of additional stages 2–3, sparse connections are formed between the input and output neurons of the layer and its dimension is reduced by zeroing some of the parameters that are not used in fine-tuning and further using of the neural network model (Fig. 4):

1) Pre-training of a neural network represented as a sequence of restricted Boltzmann machines according to greedy layer-wise algorithm.
2) Zeroing parameters of the neural network that do not exceed some specified threshold $t > 0$. In other words, the parameters falling within the interval $[-t, t]$ are excluded and are not used in further training.
3) Architectural reconfiguration of the neural network, during which the neurons that are not involved in the formation of the output activity of the network (neurons with completely zero weight coefficients) are removed.
4) Fine-tuning of the resulting simplified architecture, for example, by backpropagation algorithm.

At stage 3, zero columns and rows of the layer weight matrix are removed. At the same time, the corresponding elements of the layer threshold vector are deleted and consistent deletion of rows and columns of the next or previous layers is ensured (this is done to avoid violation of consistency between the dimensions of the matrices and vectors of neighboring layers).

IV. Results

Let us demonstrate the effectiveness of the proposed approach using the example of reducing various architectures of fully connected neural networks used to classify images from MNIST [22], CIFAR10 and CIFAR100 [23] datasets. These datasets are classic for testing the performance of machine learning models.

We conducted a series of experiments, including various datasets, architectures, and pre-training options used. Within the same dataset and NN architecture, the current initialization of the parameters was saved to be able to compare the effectiveness of different variants of the pre-training procedure.

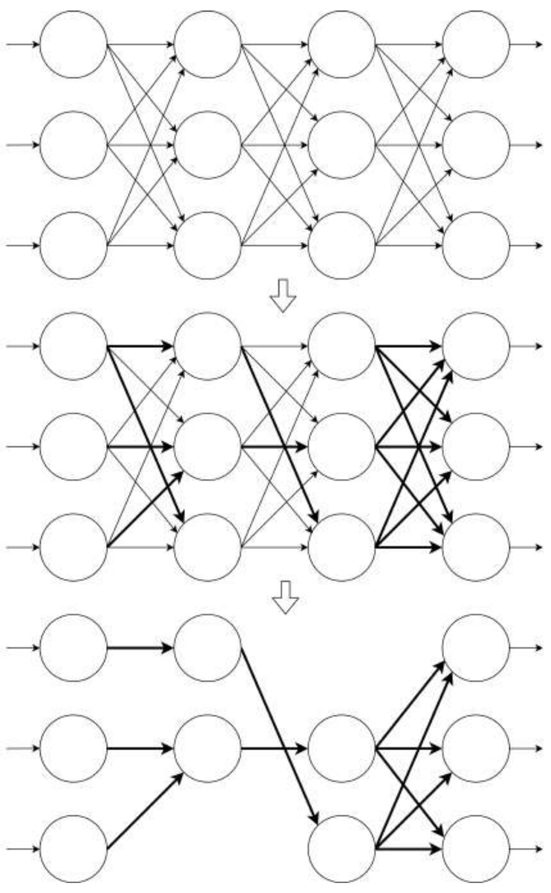
Figure 4. Method of reducing parameters on the example of fully connected layers

Below, for the considered datasets, the main parameters are given, including the learning rate, mini-batch size, momentum parameter, and the number of epochs for pre-training and fine-tuning of models (Table I).

Table I
MAIN TRAINING PARAMETERS

Stage	Parameter	Value
Training	Learning Rate	0.05-0.1
	Mini-batch size	100
	Momentum parameter	0.9
	Number of training epochs	50-100
Pre-training	Learning Rate	0.05-0.2
	Mini-batch size	32-100
	Momentum parameter	[0.5, 0.9]
	Number of training epochs	10

As a result of the computational experiment, results were obtained for various datasets, NN architectures, and values of reduction parameter t (Table II-VI).

As can be seen from the above results, the considered

Table II
RESULTS OF TRAINING – MNIST, 784-800-800-10

Type	**Efficiency, %, Classic / REBA**	**Tunable parameters, Classic / REBA**	**Reduced parameters, %, Classic / REBA**
wr	**98.63** / 98.33	1276810 / 1276810	0/0
t=0.2	**98.61** / 98.27	**233760** / 279635	**81.69** / 78.1
t=0.5	98.03 / **98.05**	**32524** / 32817	**97.45** / 97.43
t=0.8	**97.1** / 96.48	17061 / **12217**	98.66 / **99.04**

Table III
RESULTS OF TRAINING – MNIST, 784-1600-1600-800-800-10

Type	**Efficiency, %, Classic / REBA**	**Tunable parameters, Classic / REBA**	**Reduced parameters, %, Classic / REBA**
wr	**98.76** / 98.37	5747210 / 5747210	0/0
t=0.2	98.51 / **98.55**	**710734** / 781103	**87.63** / 86.41
t=0.5	98.01 / **98.03**	54709 / **43867**	99.05 / **99.24**
t=0.8	**96.9** / 93.08	25385 / **14914**	99.56 / **99.74**

Table IV
RESULTS OF TRAINING – CIFAR10, 3072-1024-512-256-128-64-10

Type	**Efficiency, %, Classic / REBA**	**Tunable parameters, Classic / REBA**	**Reduced parameters, %, Classic / REBA**
wr	**58.56** / 55.85	3844682 / 3844682	0/0
t=0.2	**58.69** / 54.37	409211 / **227072**	89.36 / **94.09**
t=0.5	**42.08** / 41.2	29033 / **11320**	99.24 / **99.71**
t=0.8	**23.02** / 10.0	10058 / **4886**	99.74 / **99.87**

Table V
RESULTS OF TRAINING – CIFAR10, 3072-512-256-128-64-10

Type	**Efficiency, %, Classic / REBA**	**Tunable parameters, Classic / REBA**	**Reduced parameters, %, Classic / REBA**
wr	**57.28** / 53.69	1746506 / 1746506	0/0
t=0.2	**56.83** / 41.72	220037 / **126846**	87.40 / **92.73**
t=0.5	**45.29** / 44.93	20431 / **11383**	98.83 / **99.35**
t=0.8	10.0 / 10.0	8599 / 3797	99.51 / 99.78

Table VI
RESULTS OF TRAINING – CIFAR100, 3072-3072-1024-512-256-128-64-100

Type	**Efficiency, %, Classic / REBA**	**Tunable parameters, Classic / REBA**	**Reduced parameters, %, Classic / REBA**
wr	20.84 / **21.63**	13290788 / 13290788	0/0
t=0.2	20.77 / **21.01**	1304525 / **703319**	90.18 / **94.71**
t=0.5	**13.4** / 1.0	49847 / **24636**	99.62 / **99.81**
t=0.8	**2.67** / 1.0	21329 / **16977**	99.84 / **99.87**

architectures generally retain their generalizing properties, being reduced by more than 80 percent, and even with a greater degree of reduction, they demonstrate good generalizing ability.

It is also possible to notice that the more adjustable parameters in the model, the more efficient the reduction is. However, with an increase in the reduction parameter, the efficiency of the original network gradually decreases, since the reduction begins to concern the parameters that affect the final result.

The obtained results substantiate the possibility of pre-training a deep neural network using an uncontrolled procedure without obtaining the effect of overfitting and reducing the efficiency of the model, since in the process of pre-training, the influence of certain model parameters on the final output activity of the network is actually reduced. Such parameters are “parasitic” in nature and are actually a factor of the model overfitting. At the model additional training stage, they are not modified and can be removed after the pre-training stage.

V. CONCLUSION

In the article, the problem field generated by the use of modern deep neural networks is defined. The main advantages for integrated computer ostis-systems that appear when using reduction as a way to reduce the dimensionality of neural networks are determined.

An approach to the implementation of the method of reducing the parameters of deep neural networks based on the use of pre-training is proposed. A review of the rules for performing unsupervised learning of restricted Boltzmann machines and convolutional restricted Boltzmann machines is given. The obtained theoretical results are used for pre-training of deep neural networks.

Experimental studies of the proposed reduction method were carried out, which confirmed its effectiveness.

The authors see the further development of the proposed approach in obtaining practical results for known deep architectures of models used to solve problems of computer vision and natural language processing.

REFERENCES

[1] V. V. Golenkov, N. A. Gulyakina, and D. V. Shunkevich, *Otkrytaya tekhnologiya ontologicheskogo proektirovaniya, proizvodstva i ekspluatatsii semanticheski sovmestimykh gibridnykh intellektual'nykh komp'yuternykh sistem [Open technology of ontological design, production and operation of semantically compatible hybrid intelligent computer systems]*. Minsk, Bestprint, 2021, (In Russ.).

[2] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” 2022.

[3] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” 2018.

[4] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” 2022.
[5] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, “Llama: Open and efficient foundation language models,” 2023.
[6] OpenAI, “GPT-4 Technical Report,” 2023.
[7] L. Bernstein, A. Sludds, R. Hamerly, V. Sze, J. Emer, and D. Englund, “Freely scalable and reconfigurable optical hardware for deep learning,” *Scientific Reports*, vol. 11, 02 2021.
[8] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, “Alpaca: A strong, replicable instruction-following model,” 2023. [Online]. Available: https://crfm.stanford.edu/2023/03/13/alpaca.html
[9] ——, “Stanford alpaca: An instruction-following llama model,” https://github.com/tatsu-lab/stanford_alpaca, 2023.
[10] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer, “Opt: Open pre-trained transformer language models,” 2022.
[11] A. Kroshchanka and V. Golovko, “The reduction of fully connected neural network parameters using the pre-training technique,” in *2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, vol. 2, 2021, pp. 937–941.
[12] H. Mostafa and X. Wang, “Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization,” 2019.
[13] S. Han, J. Pool, J. Tran, and W. J. Dally, “Learning both weights and connections for efficient neural networks,” 2015.
[14] P. Smolensky, “Information processing in dynamical systems: Foundations of harmony theory,” *Parallel Distributed Process*, vol. 1, 01 1986.
[15] V. Golovko and V. Krasnoproshin, *Neural network data processing technologies*. BSU, 2017. [Online]. Available: https://elib.bsu.by/handle/123456789/193558
[16] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, no. 521 (7553), pp. 436–444, 2015.
[17] G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal, “The “wake-sleep” algorithm for unsupervised neural networks,” *Science*, vol. 268, no. 5214, pp. 1158–1161, 1995. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.7761831
[18] G. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural computation*, vol. 18, pp. 1527–54, 08 2006.
[19] Y. Bengio, “Learning deep architectures for AI,” *Foundations and Trends in Machine Learning*, no. 2(1), pp. 1–127, 2009.
[20] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML ’09. New York, NY, USA: Association for Computing Machinery, 2009, p. 609–616. [Online]. Available: https://doi.org/10.1145/1553374.1553453
[21] V. Golovko, A. Kroshchanka, and E. Mikhno, “Deep Neural Networks: Selected Aspects of Learning and Application,” in *Pattern Recognition and Image Analysis*. Cham: Springer International Publishing, 2021, pp. 132–143.
[22] Y. LeCun and C. Cortes, “MNIST handwritten digit database.” [Online]. Available: http://yann.lecun.com/exdb/mnist/
[23] A. Krizhevsky, “Learning multiple layers of features from tiny images,” pp. 32–33, 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

Редуцирование нейросетевых моделей в интеллектуальных компьютерных системах нового поколения

Крощенко А. А.

Статья посвящена разработке метода редуцирования глубоких нейронных сетей в контексте интеграции подобных моделей в ostis-системы. Предлагается альтернативный подход к обучению глубоких нейронных сетей, базирующийся на использовании RBM и CRBM. Предлагается метод для снижения размерности “тяжелых” моделей. Полученные теоретические результаты подтверждаются вычислительными экспериментами, демонстрирующими эффективность предложенного подхода к редуцированию.

Received 12.03.2023