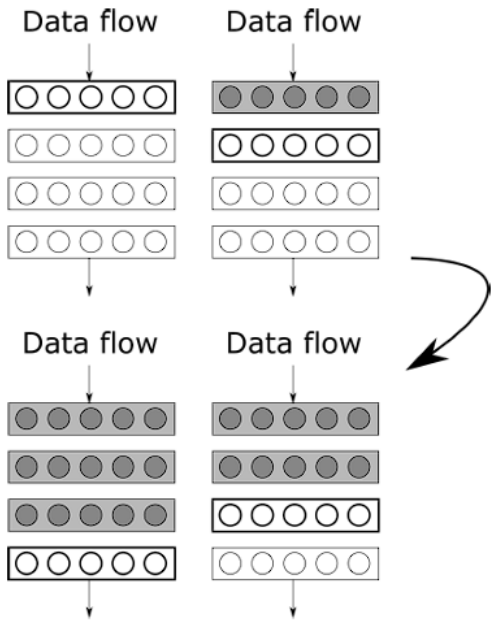

Figure 2. Greedy layer-wise algorithm

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.

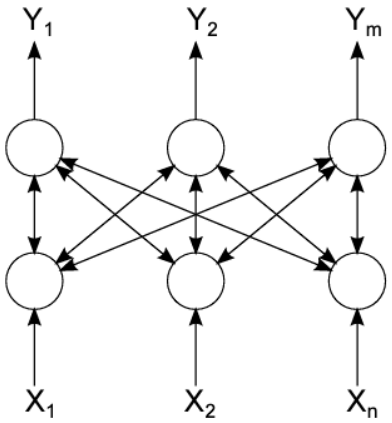

Figure 3. Restricted Boltzmann machine

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

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

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

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

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:

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

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

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

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):

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

where $\triangle y_j^l(k) - y_i^l(0)$,$\triangle x_i^l(k) - x_i^l(0)$,L — a dimension of the training dataset. The RBM online training rules in accordance with the proposed approach for CD-k are as follows:

$\omega_{ij} = \omega_{ij}(t) - \alpha((y_j(k) - y_j(0))F'(S_j(k)x_i$
$+(x_i(k) - x_i(0))F'(S_i(k))u_i(0))$,
$T_i(t+1) = T_i(t) - \alpha(x_i(k) - x_i(0))F'(S_i(k))$,
$T_j(t+1) = T_j(t) - \alpha(y_j(k) - y_j(0))F'(S_j(k))$.

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

$\omega_{ij}(t+1) = \omega_{ij}(t) - \alpha((y_i(k) - y_j(0))F(S_j(k)) \circledast x_i(k) + (x_i(K) - x_i(0))F(S_i(k)) \circledast y_i(0))$,

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

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

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 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

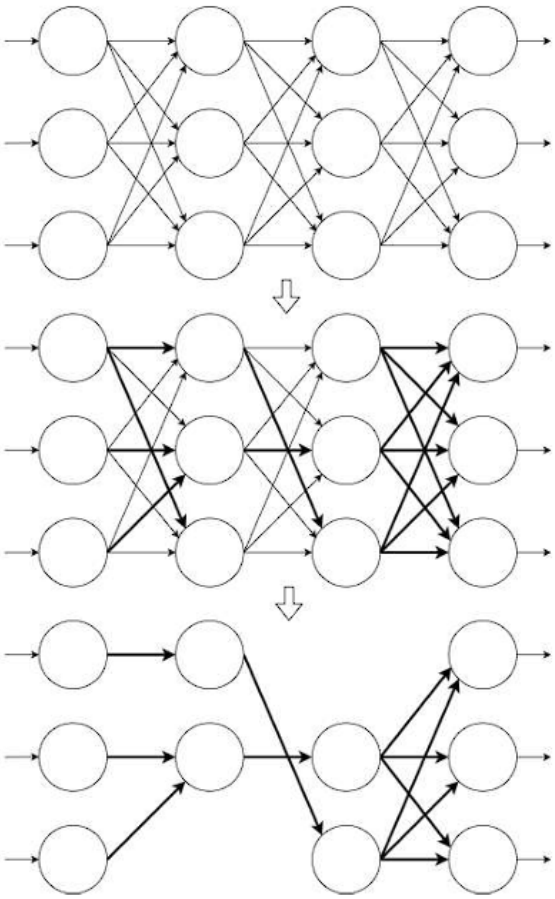

Figure 4. Method of reducing parameters on the example of fully connected layers

compare the effectiveness of different variants of the pre-training procedure.

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
WT	**96.63**/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
WT	**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
WT	**58.56/55**.85	3844682 / 3844682	[t]2=0/0
t=0.2	**58.69/54**.37	409211/**227072**	879.36/**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
WT	**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

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. 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.

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
WT	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

The obtained results substantiate the possibility of pretraining 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. Concluzion

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.