Stochastic Optimization of Contextual Neural Networks with RMSprop

Maciej Huk

Faculty of Computer Science and Management, Wroclaw University of Science and Technology,
Wroclaw, Poland
maciej.huk@pwr.edu.pl

Abstract. The paper presents modified version of Generalized Error Backpropagation algorithm (GBP) merged with RMSprop optimizer. This solution is compared with analogous method based on Stochastic Gradient Descent. Both algorithms are used to train MLP and CxNN neural networks solving selected benchmark and real–life classification problems. Results indicate that usage of GBP-RMSprop can be beneficial in terms of increasing classification accuracy as well as decreasing activity of neurons' connections and length of training. This suggests that RMSprop can effectively solve optimization problems of variable dimensionality. In the effect, merging GBP with RMSprop as well as with other optimizers such as Adam and AdaGrad can lead to construction of better algorithms for training of contextual neural networks.

Keywords: Classification · Self-consistency · Aggregation functions

1 Introduction

The popularity of data processing methods based on artificial neural networks is related to their interesting properties and proven usefulness in many different applications from science to business and engineering. They are used in medicine to analyze tissue samples and support diagnostic processes [1] as well as in transport to control autonomous vehicles [2]. Artificial neural networks are used for echo cancelling in telecommunication systems [3] and also are crucial in data acquisition and processing during experiments such as ATLAS of the Large Hadron Collider [4]. They can be found as parts of recommender systems for financial institutions [5] as well as for end customers [6]. And currently they find their place in entertainment serving image enhancement in video games [7, 8].

But to solve different types of tasks different kinds and architectures of artificial neural networks are considered and proposed. Various convolutional neural networks (CNN) are developed and used for image processing systems [9]. Recurrent neural networks including Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) networks are very good in recognition of relations within time-series data as well as in analyzing and translation of text [10, 11]. And self organizing maps (SOM) are well known kind of neural systems useful in data clustering for recommender systems and

N. T. Nguyen et al. (Eds.): ACIIDS 2020, LNAI 12034, pp. 343–352, 2020.
https://doi.org/10.1007/978-3-030-42058-1_29

knowledge discovery [12]. Specialized architectures of artificial neural networks such as Generative Adversarial Networks and Variational Autoencoders are used for generation of data of given properties, including images, text and speech [13, 14]. Another example are chaos neural networks can be fast generators of random numbers [15]. Finally, new types of neural networks emerge such as e.g. contextual neural networks (CxNN) with their ability to adjust their internal activity and dimensionality of considered input space to optimize accuracy and cost of processing of given data vectors [16–18].

Contextual neural networks were developed to model situations in which priorities and order of input signals processing can highly influence the results of data analysis [19]. They are using low-level, decentralized selective attention mechanisms in the form of conditional, multi-step aggregation functions [16, 17]. At the same time they can be used easily wherever multilayer perceptron neural networks (MLP) are used. This is because MLP neural networks are just special case of contextual neural networks. Moreover, conditional signals processing allows CxNNs to dynamically limit activity of their internal connections without decrease of data processing accuracy. This makes them a very good replacement for MLP networks in embedded applications with strong limitations of energy and computing power.

Contextual neural networks were used successfully in many practical applications such as e.g. fingerprints classification in crime investigations [20], transmissions prediction in cognitive radio, and classification of cancer gene expression microarray data [18]. Lately contextual neural networks were also implemented in a special version of a very popular H2O machine learning framework. This allows large scale, distributed computation with use of this type of models [21, 22].

In almost all cases mentioned above contextual neural networks were trained with SIF aggregation function [17, 18] and generalized error backpropagation method (GBP) based on stochastic gradient descent approach (SGD, with mini-batch = 1). In this paper we are analyzing properties of contextual neural networks when trained with GBP algorithm modified to use RMSprop [23] stochastic optimization. The tests are performed on three microarray data sets of cancer gene expression, such as: Armstrong (ALL-MLL Leukemia), Golub (ALL-AML Leukemia) and SRBCT (Small Round Blue Cell Tumors) [24–26]. Additionally selected benchmark problems from UCI ML repository were analyzed for comparison with previously reported results [27].

The further parts of the paper are organized as follows. The second section includes description of the GBP algorithm and basic properties of contextual neural networks.

Next, in Sect. 3, the modified GBP method is presented with details related to RMSprop algorithm. Within Sect. 4 the results of experiments with CxNN trained with GBP-SGD are compared with outcomes of GBP-RMSprop. Finally, conclusions are given in Sect. 5 along with planned research.

2 Generalized Error Backpropagation Algorithm

Neurons of contextual neural networks are using multi-step, conditional aggregation functions. Their inputs are clustered in groups of different priorities and in each step of aggregation only one group of inputs is read in and analyzed. The partial activation of the group is calculated and added to the activation of the neuron. This process is repeated for the groups of decreasing priorities till the activation of the neuron is greater than given constant threshold or till all groups are analyzed. Finally, the aggregated activation value is used by activation function (e.g. tanh or linear rectifier) to calculate output value of the neuron. This is generalization of the classical neuron used to build MLP neural networks, in which all inputs belong to only one group.

If the connections which are the most important to solve given problem are assigned to the groups of the highest priorities (one input connection can be assigned only to one group), it can happen that for given input vector not all inputs will be read in and analyzed to calculate the output of the neuron. It can be said that in such case the activity of input connections of the neuron was below 100%. One can also observe that the same neuron with conditional aggregation function can have different activities of inputs for different input vectors. And the lower the activity of input connections of neurons the lower the computational cost of usage of neural network and the lower the influence of low-importance input signals on the output values of the neurons.

It is the role of the training algorithm of neural network to find such grouping of inputs of each neuron to minimize both the output error and average activity of the connections between neurons. To make it possible with gradient-based algorithm and without doubling the number of parameters describing input connection of the neuron, the following assumptions are made:

– the values of number of groups and aggregation threshold parameters are the same for all hidden neurons and are set before the training,
– neurons are using deep coding to store within connections weights both the description of the algorithm of calculation of output values as well as the assignment of inputs to given groups.

In the effect of above assumptions, self-consistence method can be added to the error backpropagation algorithm (BP) to optimize non-continuous, non-differentiable groupings of inputs of neurons by coupling them with continuous and differentiable parameters such as weights of connections [16]. This creates the basic form of the generalized error backpropagation algorithm (GBP) which was shown to effectively train contextual neural networks with different aggregation and activation functions [17, 18] as well as with different schemes of initialization of groupings of neuronal inputs [28]. The schematic block diagram of GBP method is presented at the Fig. 1.

During the training with GBP, weights of connections are updated with usage of generalized delta rule which takes into account the fact that some of input connections for given vectors can have no influence on the output error regardless on the value of the related input signals. At the same time neuron inputs groupings are stored in temporary virtual grouping vectors which are calculated from connections weights vector **w** with use of grouping function $\mho(\mathbf{w})$. Typically, the grouping function $\mho$ forces the relation that for given neuron input connections with higher values of weights belong to groups of higher importance. What is important - update of connections groupings can change the error space of the neural network - and if performed too frequently, can lead to destabilization of the training process.

To overcome this problem, the update of inputs groupings is done only once after each omega training epochs. This controls the coupling between weights vector and grouping

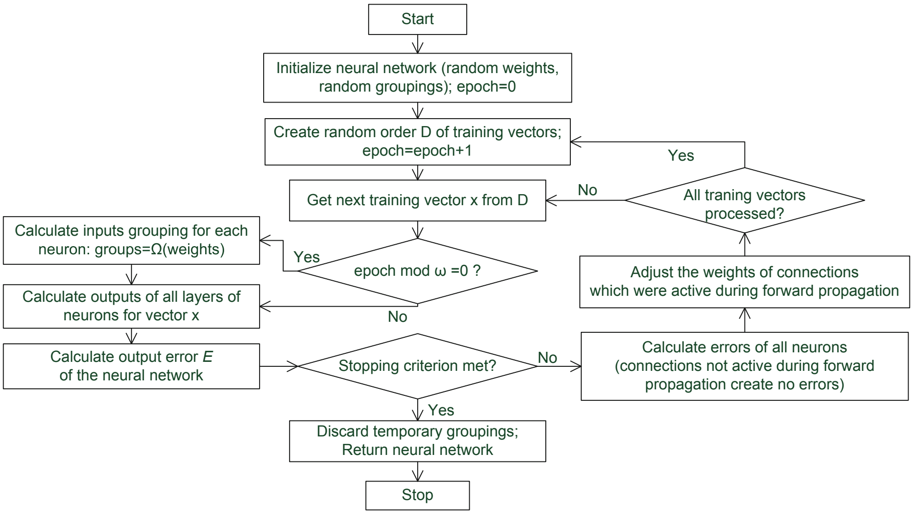
Start
Initialize neural network (random weights, random groupings); epoch=0
Create random order D of training vectors; epoch=epoch+1
Yes
No
All traning vectors processed?
Get next training vector x from D
Calculate inputs grouping for each neuron: groups=Ω(weights)
Yes
epoch mod ω =0 ?
Adjust the weights of connections which were active during forward propagation
Calculate outputs of all layers of neurons for vector x
No
Calculate output error *E* of the neural network
Stopping criterion met?
No
Calculate errors of all neurons (connections not active during forward propagation create no errors)
Yes
Discard temporary groupings; Return neural network
Stop

Fig. 1. Generalized error backpropagation algorithm using stochastic gradient descent

vector - and stabilizes the training. Finally, after the training, virtual grouping vectors can be discarded because they can be later calculated from weights of connections of neurons.

3 Generalized Error Backpropagation with RMSprop

In the previous section as well as in applications reported in the literature, the GBP algorithm is considered with the stochastic gradient descent (SGD) method of updating values of connections weights for each training vector. This is because the simplicity of SGD makes it to be faster than batch gradient descent method and allows straightforward derivation of error and weights update rules for contextual neural networks [16, 23]. But SGD method is not good at avoiding local minima of error functions. It also finds it difficult to guide training of neural networks through saddle points of error spaces. In such points the gradient of error is almost zero in all dimensions [29]. This problem can be especially frequent in low–dimensional error spaces, what is typical case in contextual neural networks. In CxNNs the error space evolves from low-dimensional to high-dimensional during conditional aggregation of inputs in neurons. When the model approaches optimum, this evolution often ends after processing of very few first groups of connections. Thus it seems to be worth checking how GBP will function when it will be extended with gradient descent optimization algorithm such as RMSprop [23].

RMSprop, a gradient descent optimization method proposed by Geoff Hinton is a simplified version of AdaDelta method. It can be expressed with the following formula for update of weight *w* of connection *j* during the training step *t*:

$$w_{t+1}^{j} = w_{t}^{j} - \frac{\alpha}{\sqrt{v_{t}^{j} + \varepsilon}} \left(\frac{\partial E_{j}}{\partial w_{j}} \right)_{t} \tag{1}$$

where

$$v_t^j = \gamma\, v_{t-1}^j + (1-\gamma)\left(\frac{\partial E_j}{\partial w_j}\right)_t^2. \tag{2}$$

In such solution the partial derivative of error E over connection weight w is used to adaptively adjust the length of step of the gradient descent. It is partially controlled by the three constant parameters: learning rate α, fraction of gradient decay γ and computability guard ε. Their typical values are: 0.001, 0.9 and 0.0001, respectively. But the most important element of the RMSprop is its partial memory v of the errors caused by given connection for previous training vectors. It allows to speed up the training while sliding from the saddle points and to make it more precise near points close to optimal solutions. This is why for many problems RMSprop outperforms SGD as well as schedule-based gradient descent methods [30].

But in the case of GBP and contextual neural network for given training vector part of connections between neurons can be not active. Thus the generalized formula for error E of j-th neuron in m-th layer of contextual neurons is

$$E_j^m = F'(\phi_j^m) \sum_{i=1}^{n_{m+1}} E_i^{(m+1)} w_{i,j}^{m+1} H(k_i^{*(m+1)} - \theta_{i,j}^{m+1}), \tag{3}$$

where F is the activation function of the neuron, φ is the activation of the neuron and θ is the number of the group to which is assigned connection between j-th neuron in m-th layer and i-th neuron in layer $m + 1$. At the same time $k*$ is the maximal number of group which was active during aggregation of signals by i-th neuron in layer $m + 1$ and H is the Heaviside function. In the effect, RMSprop formulas to be used with contextual neural network must be rewritten to the following form:

$$w_{t+1}^j = \begin{cases} w_t^j - \frac{\alpha}{\sqrt{v_t^j + \varepsilon}}\left(\frac{\partial E_j}{\partial w_j}\right)_t & : k_j^* \geq \theta_j \\ w_t^j & : k_j^* < \theta_j \end{cases} \tag{4}$$

where

$$v_t^j = \begin{cases} \gamma\, v_{t-1}^j + (1-\gamma)\left(\frac{\partial E_j}{\partial w_j}\right)_t^2 & : k_j^* \geq \theta_j \\ \gamma\, v_{t-1}^j & : k_j^* < \theta_j \end{cases} \tag{5}$$

Thus even if the connection not active for given vector does not contribute to the error of the neuron and its weight is not changed, the related memory of past gradients is modified as given by (5) for $k_j^* < \theta_j$. In the effect, after long inactivity of the connection the actual training step coefficient becomes close to its initial value, what is expected. And finally, by using (3) to calculate right-hand partial derivatives within (4) and (5) one achieves the weights update formula of GBP algorithm combined with RMSprop method.

4 Results of Experiments

To find out how combining GBP with RMSprop modifies the training of contextual neural networks results achieved with GBP-RMSprop algorithm were compared with outcomes of standard version of GBP (further denoted as GBP-SGD). Both methods were used to solve classification problems defined by the three microarray data sets of cancer gene expression, such as: Armstrong (ALL-MLL Leukemia), Golub (ALL-AML Leukemia) and SRBCT (Small Round Blue Cell Tumors) [24–26]. Additionally selected benchmark problems from UCI ML repository were analyzed (Sonar, Heart Cancer and Crx) [27]. During experiments the values of following parameters were examined: number of training epochs, hidden connections activity of resulting models as well as their accuracy of classification for test data. The architectures of trained models were constructed with the use of one-hot encoding of attributes, as well as numbers of neurons and groups of connections as given in the Table 1.

Table 1. Basic properties of data sets and neural networks used within experiments

Data set	Number of classes	Number of attributes	Number of input neurons	Number of hidden neurons	Number of groups	Number of hidden connections	Number of data vectors
Armstrong	2	12582	12582	3	13	33785	72
Golub	2	7129	7129	3	13	21426	72
SRBCT	4	2308	2308	3	13	6963	83
Sonar	2	60	60	30	25	1860	208
Heart C.	5	13	28	10	16	330	303
Crx	2	15	47	20	3	1000	690

Hidden neurons of CxNNs were using SIF aggregation functions and single group initialization of connections grouping [16]. MLP models were trained for reference. In all cases the stopping criterion of training was perfect classification or lack of improvement of accuracy for training data for more than 1200 epochs. For each set of parameters training was performed with the use of repeated 10 times 5–fold cross-validation [31]. During each training models with the lowest test error were stored for analysis.

Values of parameters of considered training methods were: training step $\alpha = 0.01$, mini-batch size = 1, threshold of aggregation function $\varphi^* = 0.6$, groups actualization interval $\omega = 25$, fraction of gradient decay $\gamma = 0.9$, computability guard $\varepsilon = 0.0001$. Activation function of neurons was bipolar sigmoid. Uniform weights initialization was used with range $(-0.2, 0.2)$. All pseudo-random values were generated with the Mersene Twister algorithm (MT19937) [32]. For each type of analyzed training algorithms, aggregation functions and training subsets of data, the same sequences of random values were used during initialization and training of neural networks. Experiments were performed with the use of 3706.6 ± 0.2 MHz 16 core Intel Core i9 9900 K CPU with core-wise separation of simulation processes with appropriate core affinity settings.

The statistical significances of measurements were calculated with the use of two-sample T-Test (confidence above 90%). Shapiro-Wilk normality test was used to analyze the normality of series. Results for MLP neural networks trained with GBP-SGD and GBP-RMSprop are presented in Table 2. Analogous results for CxNNs with SIF aggregation function are given in Table 3.

Table 2. Average results with standard deviations for MLP neural networks trained with GBP-SGD and GBP-RMSprop algorithms. Highlighted values are statistically better.

Data set	Training algorithm	Training epochs [1]	Training time [s]	Test error [%]	Hidden conn. activity [%]
Armstrong	GBP-SGD	**8.2 ± 4.8**	**13.7 ± 8.0**	9.2 ± 8.7	100 ± 0
	GBP-RMSp	56.7 ± 6.1	126.3 ± 148.0	**3.3 ± 4.4**	100 ± 0
Golub	GBP-SGD	**10.8 ± 6.4**	**9.8 ± 5.7**	9.8 ± 8.3	100 ± 0
	GBP-RMSp	30.3 ± 27.2	28.0 ± 25.0	**3.0 ± 4.0**	100 ± 0
SRBCT	GBP-SGD	**11.2 ± 2.9**	**5.6 ± 1.4**	10.9 ± 6.9	100 ± 0
	GBP-RMSp	47.1 ± 21.4	16.2 ± 7.3	**4.2 ± 5.2**	100 ± 0
Sonar	GBP-SGD	280 ± 328	11.1 ± 13.0	13.0 ± 4.5	100 ± 0
	GBP-RMSp	**111 ± 196**	**4.5 ± 7.8**	**10.3 ± 4.1**	100 ± 0
Heart C.	GBP-SGD	528 ± 796	16.4 ± 24.8	12.9 ± 3.8	100 ± 0
	GBP-RMSp	**304 ± 543**	**11.7 ± 20.9**	12.2 ± 3.6	100 ± 0
Crx	GBP-SGD	783 ± 1216	64.9 ± 107.2	11.6 ± 2.2	100 ± 0
	GBP-RMSp	877 ± 928	59.6 ± 63.0	10.9 ± 2.3	100 ± 0

As it can be seen in Table 2 the usage of GBP-RMSprop algorithm allows to generate MLP neural networks which are better than analogous structures trained with usage of GBP-SGD method. The former models for all considered problems have lower average classification error for the test data, and in case of four problems this difference is statistically significant. This is especially evident in the case of neural networks with higher number of connections between neurons (problems Armstrong, Golub, SRBCT). It can be observed that decrease of the average test error in most cases is related with increase of the number of training epochs (except Sonar and Heart Cancer problems). While for all MLP neural networks the activity of hidden connections is by definition equal 100%, changes of number of training epochs are connected with proportional changes of training time. It is also worth to note, that in the case of MLP networks GBP algorithm behaves like standard error backpropagation method (with SGD or RMSprop).

At the same time GBP-RMSprop can train contextual neural networks which produce lower testing error than their counterparts trained with GBP-SGD. And for most considered problems the related change of number of training epochs is lower than for MLP networks. E.g. for Golub data set the number of training epochs increases 3 times when GBP–RMSprop is used for MLP and only by 3% for CxNNs. GBP–RMSprop has also no problems with reduction of activity of hidden connection within CxNNs. For Sonar and Heart Cancer data sets activity of hidden connections is decreased by 20 and 30% points, respectively, while the testing error is lower than results for MLP for both training

Table 3. Average results with standard deviations for CxNNs with SIF aggregation trained with GBP-SGD and GBP-RMSprop algorithms. Highlighted values are statistically better.

Data set	Training algorithm	Training epochs [1]	Training time [s]	Test error [%]	Hidden conn. activity [%]
Armstrong	GBP-SGD	**9.8 ± 5.5**	**14.8 ± 6.6**	6.4 ± 6.6	55.1 ± 24.7
	GBP-RMSp	22.8 ± 17.1	36.3 ± 26.1	**4.3 ± 4.9**	58.7 ± 16.9
Golub	GBP-SGD	15.0 ± 10.8	11.0 ± 9.8	5.9 ± 6.2	50.5 ± 21.6
	GBP-RMSp	15.4 ± 7.5	19.2 ± 9.1	**3.0 ± 4.2**	51.9 ± 13.0
SRBCT	GBP-SGD	**17.1 ± 7.4**	**7.5 ± 2.8**	8.5 ± 8.2	48.3 ± 12.0
	GBP-RMSp	21.4 ± 8.0	10.2 ± 4.6	**6.8 ± 6.3**	72.3 ± 11.3
Sonar	GBP-SGD	1036 ± 755	65.2 ± 47.5	9.7 ± 3.6	46.1 ± 10.3
	GBP-RMSp	**804 ± 836**	**49.5 ± 52.1**	8.9 ± 3.6	**33.1 ± 10.4**
Heart C.	GBP-SGD	1083 ± 1236	71.8 ± 79.9	11.0 ± 3.1	53.6 ± 11.6
	GBP-RMSp	**841 ± 634**	**63.2 ± 48.0**	10.2 ± 3.0	**43.5 ± 10.8**
Crx	GBP-SGD	1150 ± 1407	**116.9 ± 149.1**	10.5 ± 2.2	67.2 ± 7.4
	GBP-RMSp	1168 ± 935	146.5 ± 119.5	10.1 ± 2.3	68.1 ± 6.8

methods. Thus in the case of MLP and CxNNs usage of GBP-RMSprop can be beneficial both in terms of number of training epochs, hidden connections activity and testing error.

5 Conclusions

In this paper a modification of Generalized Error Backpropagation algorithm was presented which includes appropriately adapted RMSprop optimizer. As expected, this allowed statistically significant reduction of test error, activity of hidden connections and number of epochs of training of considered contextual neural networks. Classification results obtained for CxNNs built with GBP-RMSprop in case of four out of six analyzed problems were better than for MLP networks obtained with the same training method. And in all cases models built with GBP-RMSprop were better than those trained with GBP-SGD. This suggests that RMSprop can be effectively adapted and used with contextual optimization models which are operating in error spaces of variable dimensionality which are evolving during processing of given data vectors.

Presented results also open new questions and research directions related with contextual neural networks and optimization algorithms such as GBP-RMSprop. First, it is unknown why for SRBCT data set the measured testing error for models trained with GBP-RMSprop is higher for CxNNs than for MLP. Second, it could be interesting to find out why the usage of GBP-RMSprop decreased number of epochs of training of CxNNs in relation to MLPs in the case of the three biggest of considered neural networks (all for microarray data). Performing analogous analyses for additional benchmark data sets could be very helpful in this task. Finally, presented results of measurements of usage of

GBP–RMSprop indicate that it could be also beneficial to check the results of training of contextual neural networks with other aggregation functions and modifications of GBP method. This would include merging of GBP e.g. with Adam and AdaGrad gradient descent optimizers [23].

References

1. Suleymanova, I., et al.: A deep convolutional neural network approach for astrocyte detection. Sci. Rep. **8**(12878), 1–7 (2018)
2. Chen, S., Zhang, S., Shang, J., Chen, B., Zheng, N.: Brain-inspired cognitive model with attention for self-driving cars. IEEE Trans. Cogn. Dev. Syst. **11**(1), 13–25 (2019)
3. Zhang, S., Zheng, W.X.: Recursive adaptive sparse exponential functional link neural network for nonlinear AEC in impulsive noise environment. IEEE Trans. Neural Netw. Learn. Syst. **29**(9), 4314–4323 (2018)
4. Guest, D., Cranmer, K., Whiteson, D.: Deep learning and its application to LHC physics. Annu. Rev. Nucl. Part. Sci. **68**, 1–22 (2018)
5. Bao, W.N., Yue, J.H., Rao, Y.: A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PloS ONE **12**(7), 1–24 (2017)
6. Tsai, Y.-C., et al.: FineNet: a joint convolutional and recurrent neural network model to forecast and recommend anomalous financial items. In: Proceedings of the 13th ACM Conference on Recommender Systems RecSys 2019, pp. 536–537. ACM, New York (2019)
7. Gao, D., Li, X., Dong, Y., Peers, P., Xu, K., Tong, X.: Deep inverse rendering for high-resolution SVBRDF estimation from an arbitrary number of images. ACM Trans. Graphics (SIGGRAPH) **38**(4), 1–15 (2019). Article no. 134
8. Liu, L., et al.: Automatic skin binding for production characters with deep graph networks. ACM Trans. Graphics (SIGGRAPH) **38**(4), 1–12 (2019). Article no. 114
9. Gong, K., et al.: Iterative PET image reconstruction using convolutional neural network representation. IEEE Trans. Med. Imaging **38**(3), 675–685 (2019)
10. Athiwaratkun, B., Stokes, J.W.: Malware classification with LSTM and GRU language models and a character-level CNN. In: Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), USA, pp. 2482–2486. IEEE (2017)
11. Huang, X., Tan, H., Lin, G., Tian, Y.: A LSTM-based bidirectional translation model for optimizing rare words and terminologies. In: 2018 IEEE International Conference on Artificial Intelligence and Big Data (ICAIBD), China, pp. 5077–5086. IEEE (2018)
12. Dozono, H., Niina, G., Araki, S.: Convolutional self organizing map. In: 2016 IEEE International Conference on Computational Science and Computational Intelligence (CSCI), pp. 767–771. IEEE (2016)
13. Higgins, I., et al.: beta-VAE: learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Represent, ICLR 2017, vol 2, no. 5, pp. 1–22 (2017)
14. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: International Conference on Learning Representations, ICLR 2018, pp. 1–26 (2018)
15. Alcin, M., Koyuncu, I., Tuna, M., Varan, M., Pehlivan, I.: A novel high speed artificial neural network–based chaotic true random number generator on field programmable gate array. Int. J. Circuit Theory Appl. **47**(3), 365–378 (2019)
16. Huk, M.: Backpropagation generalized delta rule for the selective attention Sigma-if artificial neural network. Int. J. Appl. Math. Comput. Sci. **22**, 449–459 (2012)

17. Huk, M.: Notes on the generalized backpropagation algorithm for contextual neural networks with conditional aggregation functions. J. Intell. Fuzzy Syst. **32**, 1365–1376 (2017)
18. Huk, M.: Training contextual neural networks with rectifier activation functions: role and adoption of sorting methods. J. Intell. Fuzzy Syst. **38**, 1–10 (2019)
19. Huk, M.: Learning distributed selective attention strategies with the Sigma-if neural network. In: Akbar, M., Hussain, D. (eds.) Advances in Computer Science and IT, pp. 209–232. InTech, Vukovar (2009)
20. Szczepanik, M., Jóźwiak, I.: Data management for fingerprint recognition algorithm based on characteristic points' groups. In: Pechenizkiy, M., Wojciechowski, M. (eds.) New Trends in Databases and Information Systems. Advances in Intelligent Systems and Computing, vol. 185, pp. 425–432. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-32518-2_40
21. Janusz, B.J., Wołk, K.: Implementing contextual neural networks in distributed machine learning framework. In: Nguyen, N.T., Hoang, D.H., Hong, T.-P., Pham, H., Trawiński, B. (eds.) ACIIDS 2018. LNCS (LNAI), vol. 10752, pp. 212–223. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75420-8_20
22. Wołk, K., Burnell, E.: Implementation and analysis of contextual neural networks in H2O framework. In: Nguyen, N.T., Gaol, F.L., Hong, T.-P., Trawiński, B. (eds.) ACIIDS 2019. LNCS (LNAI), vol. 11432, pp. 429–440. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-14802-7_37
23. Ruder, S.: An overview of gradient descent optimization algorithms, pp. 1–14. eprint arXiv:1609.04747v2 (2017)
24. Armstrong, S.A.: MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. Nat. Genet. **30**, 41–47 (2002)
25. Golub, T.R., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science **286**, 531–537 (1999)
26. Khan, J., et al.: Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nat. Med. **7**(6), 673–679 (2001)
27. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml
28. Huk, M.: Non-uniform initialization of inputs groupings in contextual neural networks. In: Nguyen, N.T., Gaol, F.L., Hong, T.-P., Trawiński, B. (eds.) ACIIDS 2019. LNCS (LNAI), vol. 11432, pp. 420–428. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-14802-7_36
29. Dauphin Y., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., Bengio, Y.: Identifying and attacking the saddle point problem in high dimensional non-convex optimization, pp. 1–14. eprint arXiv:1406.2572 (2014)
30. Darken, C., Chang, J., Moody, J.: Learning rate schedules for faster stochastic gradient search. In: Proceedings of the 1992 IEEE Workshop on Neural Networks for Signal Processing II, September, pp. 1–11 (1992)
31. Bouckaert, R.R., Frank, E.: Evaluating the replicability of significance tests for comparing learning algorithms. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 3–12. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24775-3_3
32. Matsumoto, M., Nishimura, T.: Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. ACM Trans. Model. Comput. Simul. **8**(3), 3–30 (1998)