Proceedings of the Federated Conference on Computer Science and Information Systems pp. 141–145
DOI: 10.15439/2016F183
ACSIS, Vol. 8. ISSN 2300-5963

Deep Evolving GMDH-SVM-Neural Network and its Learning for Data Mining Tasks

Galina Setlak
Rzeszow University of Technology
12 Al. Powstancow Warszawy, 35-959, Rzeszow, Poland
Email: gsetlak@prz.edu.pl

Yevgeniy Bodyanskiy
Kharkiv National Univesity of Radio Electronics
14 Nauky Ave., Kharkiv, 61166, Ukraine
Email: yevgeniy.bodyanskiy@nure.ua

Olena Vynokurova
Kharkiv National Univesity of Radio Electronics
14 Nauky Ave., Kharkiv, 61166, Ukraine
Email: olena.vynokurova@nure.ua

Iryna Pliss
Kharkiv National Univesity of Radio Electronics
14 Nauky Ave., Kharkiv, 61166, Ukraine
Email: iryna.pliss@nure.ua

***Abstract*—In the paper, the deep evolving neural network and its learning algorithms (in batch and on-line mode) are proposed. The deep evolving neural network's architecture is developed based on GMDH approach (in J. Schmidhuber's opinion it is historicaly first system, which realizes deep learning) and least squares support vector machines with fixed number of the synaptic weights, which provide high quality of approximation in addition to the simlicity of implementation of nodes with two inputs. The proposed system is simple in computational implementation, characterized by high learning speed and allows processing of data, which are sequentially fed in on-line mode. The proposed system can be used for solving a wide class of Dynamic Data Mining tasks, which are connected with non-stationary, nonlinear stochastic and chaotic signals. The computational experiments are confirmed the effectiveness of the developed approach.**

I. Introduction

Nowadays, artificial neural networks (ANNs) are widely used for solving a lot of Data Mining tasks. In these tasks initial information is presented in the form of both "object-properties" table and multivariate time series, which are generated by stochastic or chaotic nonstationary nonlinear objects. The advantages of these computational intelligence systems are, first of all, their universal approximation properties and learning abilities using real experimental data [1], [2].

Recent years Computational Intelligence specialists are interested in deep neural networks (DNN) [3], [4], [5], [6], [7]. The deep neural networks comparatively with conventional ANNs, also called shallow neural networks (SNNs), provide much higher quality of information processing. However, these networks are essentially tedious with relation to computational implementation, also are subjected to overfitting in case of short training samples and demand of high operation time and computational resources especially when operated with Big Data [8]. Both the standard neurons (which form set of layers) and shallow neural networks can be used as the basic elements of deep neural networks.

One of the most effective representatives of the shallow neural networks are support vector machines (SVM) [9], [10], [11], [12], [13]. The tuning of support vector machines is provided by using both lazy learning (the activation functions' centers tuning) and optimization procedures (the synaptic weights tuning). However, if the process of the lazy learning is implemented immediately, then optimization tasks solving using support vector machines with big training set is enough complex. In this connection deep neural networks, which are implemented using support vector machines [14], [15], [16], providing high quality of information processing are essentially tedious from the computational point of view.

It can be noticed, that tuning process of support vector machines can be essentially speed up if the least-squares support vector machines (LS-SVM) [17] are used instead of a conventional approach. The learning process of least-squares support vector machines reduces to a solution of the set of Karush-Kuhn-Tucker equations and the result of this learning can be written in an analytical form.

Among a great number of possible deep neural networks' architectures, the deep networks based on GMDH are one of the most effective networks, as it was mentioned in [6]. These networks are based on the group method of data handling [18], [19], [20], which allows automatically increasing a number of layers for information processing to achieve the required accuracy of the results. A combination of the GMDH approach with ANNs have led to synthesis of wide range of computational intelligence systems [21], [22], [23], [24], [25], [26], [27], [28] where different type of artificial neurons are used as nodes.

In this case unlimited increasing of layers in the sistem (using the GMDH paradigm) and simplicity of learning LS-SVM with two inputs (using their universal approximation properties) allow to efficiently information processing in on-line mode of the deep learning.

In the connection with mentioned above, it seems appropriate to develop deep neural networks' architecture based on GMDH and LS-SVM and its learning algorithm. The proposed approach is characterized by simplicity of a computational implementation and high speed learning for the solution of

978-83-60810-90-3/$25.00

wide range of Data Mining tasks, which are described by both short and large volume data set.

II. The architecture of deep evolving GMDH-SVM-neural network

An architecture of the proposed deep evolving GMDH-SVM-neural network is shown in Fig.1.

A $(n \times 1)$-dimensional vector of the input signals $x = (x_1, x_2, \ldots, x_n)^T \in R^n$ is fed to the zero (receptive) layer of the GMDH-SVM-neural network. Further, this input vector is passed to the first layer, which consists of $n_1 = C_n^2 = 0.5n(n-1)$ nodes that are the conventional LS-SVM with two inputs. It is obvious that learning process of the LS-SVM with two inputs has not any problem both in relation of a computational implementation and with regard for requirements to a volume of training set. The output signals $\hat{y}_l^{[1]}$ $(l = 1, 2, \ldots, n_1)$ are formed by the nodes' outputs of the $SVM^{[1]}$ of first hidden layer.

Further these signals are fed to the selection block $SB^{[1]}$ of the first hidden layer. This selection block SB^1 selects $n_1^*(n_1^* \leq n_1)$ signals from a range of signals $\hat{y}_l^{[1]}$. The selected signals are the best in the sense of accepted criterion, in more cases it is the mean square error $\sigma^2_{\hat{y}_l^{[1]}}$, but any other accuracy criterion can be used in relation to reference signal $y(k)$ $(k = 1, 2, \ldots, N$ is an observation number in a training set or a current discrete time, when a learning process and data processing take place in on-line mode).

From the n_1^* best output's signals of the first hidden layer $\hat{y}_l^{[1]*}$ (using the conventional GMDH approach) are formed $n_2(n \leq n_2 \leq 2n)$ pairwise combination of signals $\hat{y}_l^{[1]*}$, $\hat{y}_p^{[1]*}$, which are fed to the inputs of the second hidden layer. The second hidden layer is formed by $SVM^{[2]}$ nodes, which are similar to the elements of the first hidden layer.

From the output's signals $\hat{y}_l^{[2]}$ of this layer, the selection block $SB^{[2]}$ of the second hidden layer selects only that signals $\hat{y}_l^{[2]*}$, whose accuracy is better than the best signal $\hat{y}_l^{[1]*}$ of the first hidden layer. The third hidden layer with selection block $SB^{[3]}$ forms the signals, which have accuracy better than the best signal $\hat{y}_l^{[2]*}$ of the second layer.

In such way, the network's architecture is formed during learning process likewise evolving computational intelligence systems [29], [30].

The architecture's evolution process takes place until the selection block $SB^{[n-1]}$ forms only two signals $\hat{y}_1^{[s-1]*}$ and $\hat{y}_2^{[s-1]*}$ in its output. Just these two signals are fed to the single output nodes of $SVM^{[3]}$ where the output system's signal $\hat{y}^{[s]}$ is computed.

III. The learning of deep evolving GMDH-SVM-neural network

As it was previously noted an each node of the proposed system is the LS-SVM with single output and two inputs. Hence, two-dimensional vector $x_{ij}(k) = (x_i(k), x_j(k))^T$ $(i = 1, 2, \ldots, n; j = 1, 2, \ldots, n; i \neq j)$ is fed to the input of the first hidden layer and output of the each node is a scalar signal $\hat{y}_l^{[1]}$ $(l = 1, 2, \ldots, n_1)$. Therefore, the LS-SVM is the hybrid system, which combines a learning based on both an optimization and a memory [1], [2], [6], [7], [17], and implements minimization of an empirical risk criterion. It is necessary to notice, that the SVMs are the most effective under short data set conditions and are not subject to an overfitting and proved a high quality of approximation.

The mapping, which implements standard LS-SVM SNN, can be written in the form

$$\hat{y}_l^{[1]} = (w_l^{[1]})^T \varphi_l^{[1]}(x) + w_{0l}^{[1]} \tag{1}$$

where $w_l^{[1]} = (w_{1l}^{[1]}, w_{2l}^{[1]}, \ldots, w_{Nl}^{[1]})^T$, $\varphi_l^{[1]}(x) = (\varphi_{1l}^{[1]}, \varphi_{2l}^{[1]}, \ldots, \varphi_{Nl}^{[1]})^T$. The learning process reduces to setting the centers of activation functions (usually Gaussians) in the point, which are determined by a training sample x_{ij} $(k = 1, 2, \ldots, N)$ and minimization of squared criterion simultaneously in the form

$$E_l^{[1]}(N) = \frac{1}{2}\|w_l^{[1]}\|^2 + \frac{\gamma}{2}\sum_{k=1}^{N}(e_l^{[1]}(k))^2 \tag{2}$$

in the presence of N equality-constraints in the form

$$\begin{cases} y(1) &= (w_l^{[1]})^T \varphi_l^{[1]}(x_{ij}(1)) + w_{0l}^{[1]} + e_l^{[1]}(1), \\ \vdots & \\ y(N) &= (w_l^{[1]})^T \varphi_l^{[1]}(x_{ij}(N)) + w_{0l}^{[1]} + e_l^{[1]}(N) \end{cases} \tag{3}$$

where $\gamma > 0$ is a regularization parameter and

$$\begin{aligned} e_l^{[1]}(k) &= y(k) - \hat{y}_l^{[1]}(k) = \\ &= y(k) - (w_l^{[1]})^T \varphi_l^{[1]}(x_{ij}(k)) - w_{0l}^{[1]}. \end{aligned} \tag{4}$$

In this way, LS-SVM learning task is reduced to finding of a saddle point of Lagrange function

$$\begin{aligned} &L_l^{[1]}(w_l^{[1]}, w_{0l}^{[1]}, e_l^{[1]}, \lambda_l^{[1]}(k)) = E_l^{[1]}(N) + \\ &+ \sum_{k=1}^{N} \lambda_l^{[1]}(k)\Big(y(k) - (w_l^{[1]})^T \varphi_l^{[1]}(x_{ij}(k)) - \\ &- w_{0l}^{[1]} - e_l^{[1]}(k)\Big). \end{aligned} \tag{5}$$

This saddle point can be found by solving the Karush-Kuhn-Tucker equations set. In this case, besides $N+1$ synaptic weights $w_l^{[1]}$, $w_{0l}^{[1]}$ the N indefinite Langrange multipliers $\lambda_l^{[1]}(k)$ have to be found.

The main disadvantage of SVM in the system under consideration is necessity of adding new synaptic weights in each nodes with rising of a training set volume. Therefore, if it is necessary to process Big Data than proposed system becomes too tedious. To avoid this problem it is possible by limiting a number of synaptic weights in each node by using, so-called, "sliding window" data processing. Such "sliding window" contains only h last observations.

Introducing the Lagrange function with "sliding window" instead of the expression (5) in the form

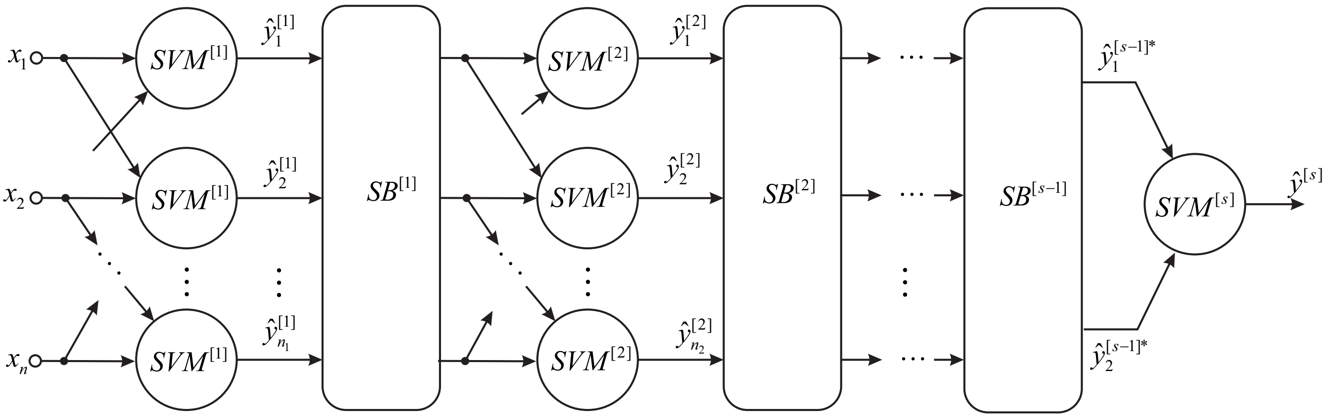

Fig. 1. The architecture of proposed deep evolving GMDH-SVM-neural network

$$\begin{aligned} & L_l^{[1]}(w_l^{[1]}, w_{0l}^{[1]}, e_l^{[1]}, \lambda_l^{[1]}(k), h) = \\ & = \frac{1}{2}\ \|w_l^{[1]}\|^2 + \frac{\gamma}{2} \sum_{\tau=k-h+1}^{k} (e_l^{[1]}(\tau))^2 + \\ & + \sum_{\tau=k-h+1}^{k} \lambda_l^{[1]}(\tau)\Big(y(\tau) - (w_l^{[1]})^T \varphi_l^{[1]}(x_{ij}(\tau)) - \\ & - w_{0l}^{[1]} - e_l^{[1]}(\tau)\Big) \end{aligned} \tag{6}$$

and solving Karush-Kuhn-Tucker equations set, we can write the result in the form

$$\begin{pmatrix} 0 & I_h^T \\ I_h & \Omega(k) + \gamma^{-1} I_{hh} \end{pmatrix} \begin{pmatrix} w_{0l}^{[1]} \\ \Lambda(k) \end{pmatrix} = \begin{pmatrix} 0 \\ Y(k) \end{pmatrix} \tag{7}$$

where $\Lambda(k) = (\lambda(k-h+1), \dots, \lambda(k))^T$, I_h is a $(h \times 1)$ unity vector, I_{hh} is a $(h \times h)$ unity matrix, $Y(k) = (y(k-h+1), \dots, y(k))$, $\Omega(k) = \{\Omega_{\tau l} = \varphi_l^{[1]T}(x_{ij}(\tau))\varphi_l^T(x_{ij}(t)) = K(x_{ij}(\tau), x_{ij}(t))\}$, $\tau = k-h+1, \dots, k$, $t = k-h+1, \dots, k$, $K(x_{ij}(\tau), x_{ij}(t))$ is the kernel function, which is satisfied to the conditions of Mercer theorem [17], and usually it is Gaussian function in the form

$$K(x_{ij}(\tau), x_{ij}(t)) = \exp\left(-\frac{\|x_{ij}(\tau) - x_{ij}(t)\|^2}{2\sigma^2}\right). \tag{8}$$

In this case, the transformation (1), which is implemented by the LS-SVM node, can be rewritten in the form

$$\hat{y}_l^{[1]} = \sum_{\tau=k-h+1}^{k} \lambda_l^{[1]}(\tau) K(x_{ij}(\tau), x_{ij}(t)) + w_{0l}^{[1]} \tag{9}$$

where parameters $\lambda_j^{[1]}(\tau)$, $w_{0l}^{[1]}$ can be defined from the system (7) in the form

$$\begin{aligned} & \begin{pmatrix} w_{0l}^{[1]} \\ \Lambda(k) \end{pmatrix} = \\ & = \begin{pmatrix} 0 & I_h^T \\ I_h & \Omega(k) + \gamma^{-1} I_{hh} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ Y(k) \end{pmatrix}. \end{aligned} \tag{10}$$

As a result, the learning of nodes in the proposed system reduces to a solving of linear equations set with fixed number of variables. It should be noticed, the "sliding window" learning allows managing the deep neural network tuning process, when information is fed to the system's input in online mode in the form of data stream. The nodes of second and next hidden layers are tuned absolutely likewise the expression (10).

IV. Simulation results

A. Identification of the mechanical system signal

Efficiency of proposed deep evolving GMDH-SVM-neural network was examined based on different benchmark data including the identification task using real data from mechanical system. Data is taken from a flexible robot arm. The arm is installed on an electrical motor. [We are grateful to Hendrik Van Brussel and Jan Swevers of the laboratory of Production Manufacturing and Automation of the Katholieke Universiteit Leuven, who provided us with these data, which were obtained in the framework of the Belgian Programme on Interuniversity Attraction Poles (IUAP-nr.50) initiated by the Belgian State - Prime Minister's Office - Science Policy Programming. http://homes.esat.kuleuven.be/ smc/daisy/daisydata.html].

Inputs number of deep evolving GMDH-SVM-neural network were taken as $n = 5$, that for input vector in the form $u(k-2), y(k-2), u(k-1), y(k-1), u(k)$ for identification value $y(k)$ where u is a reaction torque of the structure, y is an accelaration of the flexible arm . Node of deep evolving GMDH-SVM-neural network was training by proposed learning algorithm during 100 iterations. Initial parameters values of kernel functions were taken $\sigma = 0.1$. After 100 iterations the training process was stopped, and the next 50 points for $k = 101 \dots 150$ we have used as the testing data

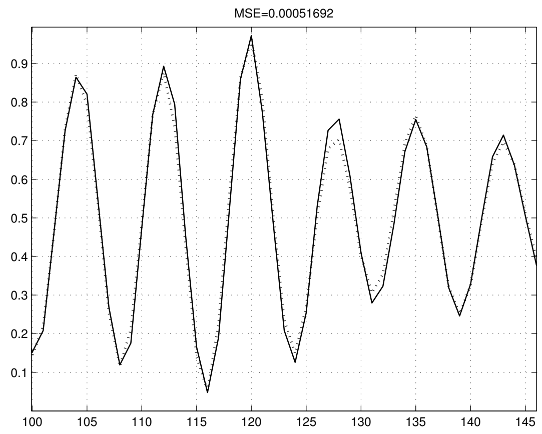

Fig. 2. Results of signal identification

set to compute forecast. Initial values of synaptic weights were taken equal to 0. As the quality criterion of forecasting root mean squared error (MSE) was used. Fig.2 shows the results of signal identification. The two curves, representing the actual (dot line) and identification (solid line) values, are almost indistinguishable.

Table I cotains comparative analysis of the signal identification based on different approaches.

Thus as it can be seen from experimental results the proposed approach provides the best quality of prediction in comparison with conventional GMDH-neural networks.

B. The identification of nonlinear nonstationary signal

Simulation of deep evolving cascaded GMDH-SVM-neural network was performed in the process of identification of nonlinear signal, which is described by equation in the form [31]

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k) \tag{11}$$

where $u(k) = sin(2\pi k/25) + sin(2\pi k/10)$ is control signal.

The inputs number of evolving cascaded GMDH-neural network were taken as $n = 4$, which correspond to the input vector $x(k-3), x(k-2), x(k-1), x(k)$ for the value $x(k+1)$.

LS-SVM-neuron was trained based on proposed procedures for 400 iterations (400 training samples for $k = 1 \ldots 400$). After 400 iterations the training process was stopped, and the next 100 points for $k = 401 \ldots 500$ we have used as the testing data set to compute signal value. Initial values of synaptic weights were taken equal to 0. As the identification quality criterion mean squared error (MSE) was used.

Fig. 3 shows the results of signal identification. The two curves, representing the actual (dot line) and identification (solid line) values, are very close. Table II shows the comparative analysis of nonlinear non-stationary signal identification based on different approaches.

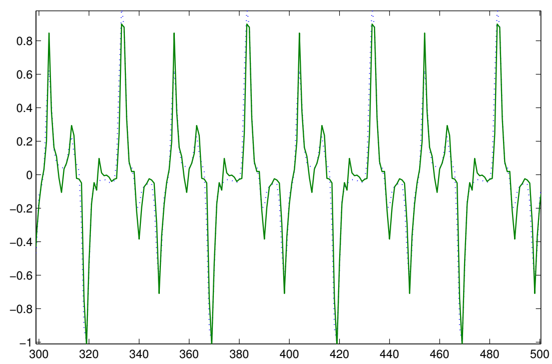

Fig. 3. Results of non-linear non-stationary system identification

V. CONCLUSIONS

In the paper, the deep evolving neural network and its learning algorithms are proposed. The architecture of the deep evolving neural network is developed based on GMDH and least squares support vector machines with fixed number of the synaptic weights are used as nodes. The proposed system is simple in computational implementation, characterized by high learning speed and allows processing of data, which are fed sequentially in on-line mode. The combining, in the context of the common deep learning system, the GMDH paradigm with unlimited increasing of the layers number and LS-SVM nodes with fixed sinaptic weights number allow to predetermine an on-line deep learning in Dynamic Data Mining tasks. The computational experiments are confirmed the effectiveness of developed approach.

REFERENCES

[1] S. Haykin, *Neural Networks and Learning Machines*. Upper Saddle River, New Jersey: Pearson, Prentice Hall, 2009.

[2] K.-L. Du and M. Swamy, *Neural Networks and Statistical Learning*. Springer-Verlag London, 2014. [Online]. Available: http://dx.doi.org/10.1007/978-1-4471-5571-3

[3] G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, May 2006. doi: 10.1162/neco.2006.18.7.1527. [Online]. Available: http://dx.doi.org/10.1162/neco.2006.18.7.1527

[4] I. Arel, D. Rose, and T. Karnowski, "Deep machine learning - a new frontier in artificial intelligence research," *IEEE Computational Intelligence Magazine*, vol. 5, no. 4, pp. 13–18, Nov. 2010. doi: 10.1109/MCI.2010.938364. [Online]. Available: http://dx.doi.org/10.1109/MCI.2010.938364

[5] Y. Bengio, Y. LeCun, and G. Hinton, "Deep learning," *Nature*, no. 521, pp. 436–444, May 2015. doi: 10.1038/nature14539. [Online]. Available: http://dx.doi.org/10.1038/nature14539

[6] J. Schmidhuber, "Deep learning in neural networks: an overview," *Neural Networks*, no. 61, pp. 85–117, Jan. 2015. doi: 10.1016/j.neunet.2014.09.003. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2014.09.003

[7] J.Kacprzyk and W. Pedrycz, Eds., *Springer Handbook of Computational Intelligence*. Springer-Verlag Berlin Heidelberg, 2015. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-43505-2

[8] W.Pedrycz and S.-M. Chen, Eds., *Information Granularity, Big Data, and Computational Intelligence*. Springer International Publishing Switzerland, 2015. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-08254-7

TABLE I
THE COMPARATIVE ANALYSIS OF THE SIGNAL IDENTIFICATION

Neural network /Learning algorithm	Number of layers	Number of inputs into nodes	MSE
Deep evolving GMDH-SVM-neural network/Proposed learning algorithm	3	2	0.00051
Hybrid multilayer GMDH-neural network [28] / Proposed learning algorithm	3	3	0.00098
Hybrid multilayer GMDH-neural network [28] / Proposed learning algorithm	3	2	0.00102
GMDH-neural network / Recurrent least squares method	3	2	0.0563

TABLE II
THE COMPARATIVE ANALYSIS OF SIGNAL IDENTIFICATION RESULTS

Neural network	Number of layers	MSE
Deep evolving GMDH-SVM-neural network	3	0.00018
Hybrid multilayer GMDH-neural network [28]	3	0.00123
Evolving cascaded GMDH neural network based on W-neurons	3	0.0009
Evolving cascaded GMDH neural network based on R-neurons with Gaussian functions	3	0.0023
Evolving cascaded GMDH neural network based on R-neurons with Epanechnikov functions	3	0.0015
Multilayer GMDH-neural network with polynomial functions	4	0.0201

[9] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag New York, 2000. [Online]. Available: http://dx.doi.org/10.1007/978-1-4757-3264-1

[10] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995. doi: 10.1023/A:1022627411411. [Online]. Available: http://dx.doi.org/10.1023/A:1022627411411

[11] M. Grochowina and L. Leniowska, "Comparison of svm and k-nn classifiers in the estimation of the state of the arteriovenous fistula problem," in *Proceedings of the 2015 Federated Conference on Computer Science and Information Systems*, ser. Annals of Computer Science and Information Systems, M. Ganzha, L. Maciaszek, and M. Paprzycki, Eds., vol. 5. IEEE, 2015. doi: 10.15439/2015F194 pp. 249–254. [Online]. Available: http://dx.doi.org/10.15439/2015F194

[12] B. Krawczyk, "Combining one-class support vector machines for microarray classification," in *Proceedings of the 2013 Federated Conference on Computer Science and Information Systems*, M. P. M. Ganzha, L. Maciaszek, Ed. IEEE, 2013, pp. pages 83–89.

[13] M. Ochab and W. Wajs, "Bronchopulmonary dysplasia prediction using support vector machine and libsvm," in *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, ser. Annals of Computer Science and Information Systems, M. P. M. Ganzha, L. Maciaszek, Ed., vol. 2. IEEE, 2014. doi: 10.15439/2014F111 pp. pages 201–208. [Online]. Available: http://dx.doi.org/10.15439/2014F111

[14] S. Kim, , S. Kavuri, and M. Lee, *Neural Information Processing: 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, 2013. Proceedings, Part I*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ch. Deep Network with Support Vector Machines, pp. 458–465. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-42054-2_57

[15] S. Kim, Y. Choi, and M. Lee, "Deep learning with support vector data description," *Neurocomputing*, vol. 165, no. 1-2, pp. 111–117, Oct. 2015. doi: 10.1016/j.neucom.2014.09.086. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2014.09.086

[16] S. Kim, Z. Yu, R. Kil, and M. Lee, "Deep learning of support vector machines with class probability output networks," *Neural Networks*, vol. 64, pp. 19–28, Apr. 2015. doi: 10.1016/j.neunet.2014.09.007. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2014.09.007

[17] J. Suykens, T. Gestel, J. Brabanter, B. Moor, and J. Vanderwalle, *Least Squares Support Vector Machines*. World Scientific, 2002. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-08254-7

[18] A. Ivakhnenko, "The group method of data handling - a rival of the method of stochastic approximation," *Soviet Automatic Control*, vol. 13, no. 3, pp. 43–55, 1968.

[19] ——, "The group method of data handling - a rival of the method of stochastic approximation," *Automatica*, vol. 6, no. 2, pp. 207–219, 1970.

[20] ——, "Polynomial theory of complex systems," *IEEE Transaction on Systems, Man, Cybernetics*, vol. 1, no. 4, pp. 364–378, Oct. 1971. doi: 10.1109/TSMC.1971.4308320. [Online]. Available: http://dx.doi.org/10.1109/TSMC.1971.4308320

[21] A. Ivakhnenko, G. Ivakhnenko, and A. Muller, "Self-organization of the neural networks with active neurons," *Pattern Recognition and Image Analysis*, vol. 4, no. 2, pp. 177–178, 1994.

[22] A. Ivakhnenko, "Self-organization of neuro net with active neurons for effects of nuclear test explosions forecasting," *System Analysis Modeling Simulation*, vol. 20, no. 1-2, pp. 107–116, 1995.

[23] T. Kondo, "Gmdh neural network algorithm using the heuristic self-organization method and its application to the pattern identification problem," in *Proceedings of the 37th SICE Annual Conference, Japan, Tokyo*, Jul. 1998. doi: 10.1109/SICE.1998.742993 pp. 1143–1148. [Online]. Available: http://dx.doi.org/10.1109/SICE.1998.742993

[24] ——, "Identification of radial basis function networks by using revised gmdh-type neural networks with a feedback loop," in *Proceedings of the 41st SICE Annual Conference, Japan, Osaka*, vol. 5, Aug. 2002. doi: 10.1109/SICE.2002.1195514 pp. 2882–2887. [Online]. Available: http://dx.doi.org/10.1109/SICE.2002.1195514

[25] Y. Bodyanskiy, I. Pliss, and O. Vynokurova, "Hybrid gmdh-neural network of computational intelligence," in *Proceedings of 3rd International Workshop on Inductive Modelling, Poland, Krynica*, Sep. 2009, pp. 100–107.

[26] Y. Bodyanskiy, Y. Zaychenko, E. Pavlikovskaya, M. Samarina, and Y. Viktorov, "The neo-fuzzy neural network structure optimization using the gmdh for the solving forecasting and classification problems," in *Proceedings of 3rd International Workshop on Inductive Modelling, Poland, Krynica*, Sep. 2009, pp. 77–89.

[27] Y. Bodyanskiy, O. Vynokurova, and N. Teslenko, "Cascade gmdh-wavelet-neuro-fuzzy network," in *Proceedings of International Workshop Inductive Modelling, Ukraine, Kyiv*, Sep. 2011, pp. 22–30.

[28] Y. Bodyanskiy, O. Vynokurova, A. Dolotov, and O. Kharchenko, "Wavelet-neuro-fuzzy-network structure optimization using gmdh for the solving forecasting tasks," in *Proceedings of International Workshop Inductive Modelling, Ukraine, Kyiv*, Sep. 2013, pp. 61–67.

[29] N. Kasabov, *Evolving Connectionist Systems*. Springer-Verlag London, 2007. [Online]. Available: http://dx.doi.org/10.1007/978-1-84628-347-5

[30] P. Angelov, D. Filev, and N. Kasabov, *Evolving Intelligent Systems: Methodology and Applications*. John Willey and Sons, 2010.

[31] K. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transaction on Neural Networks*, vol. 1, no. 1, pp. 4–26, Mar. 1990. doi: 10.1109/72.80202. [Online]. Available: http://dx.doi.org/10.1109/72.80202