Outliers Elimination for Error Correction Algorithm Improvement

Janusz Kolbusz and Pawel Rozycki

University of Information Technology and Management in Rzeszow
jkolbusz@wsiz.rzeszow.pl,prozycki@wsiz.rzeszow.pl
http://wsiz.rzeszow.pl

Abstract. Neural networks are still very important part of artificial intelligence. RBF networks seems to be more powerfull than that based on sigmoid function. Error Correction is second order training algorithm dedicated for RBF networks. The paper proposes method for improvement this algorithm by elimination of inconsistent patterns. The approach is also experimentally confirmed.

Key words: Error Correction, ErrCor, outliers, RBF networks, training algorithms

1 Introduction

Our civilization encounters increasingly complex problems that often exceeds human capabilities. Until recently, the aim was to create artificial intelligence systems so perfect, like a man. Currently, we are able to create intelligent learning systems exceeding the intelligence of the people. For example, we can create a model and predict the behavior of complex natural processes, which cannot be described mathematically. We can also identify economic trends that are invisible to humans. In order to efficiently model complex multidimensional nonlinear systems should be used unconventional methods. For given multidimensionality and nonlinear nature, algorithmic or statistical methods give unsatisfactory solutions. Methods based on computational intelligence allow to more effectively address complex problems such as foreseeing of economic trends, modeling natural phenomena, etc. To a greater extent than now harness the power of this type of network, you must:

- understand the neural network architecture and its impact on the functioning of the system and the learning process.
- find effective learning algorithms that allow faster and more effectively teach a network using its properties.

Both of problems are strictly connected.

The commonly used network MLP(*Multi-Layer Perception*) have relatively limited capacity[1]. It turns out that the new neural networks such as BMLP (*Bridged MLP*)[1,2] or DNN (*Dual Neutral Networks*) [2] with the same number of neurons area to solve problems 10 or 100 times more complex [2,3].

A way of connecting neurons in the network is fundamental. For example, if you combine 10 neurons in the most commonly used three-tiered architecture MLP (with one hidden layer) the biggest problem that can be solved solve with such network is the problem of Parity-9 type. If the same 10 neurons are connected in the FCC architecture (*Fully Connected Cascade*), it is possible to solve the problem of Parity-1023 type. As can be seen a departure from the commonly used MLP architecture, while maintaining the same number of neurons, increases network capacity, even a hundred times [2-4]. The problem is that the commonly known learning algorithms, such as EBP (*Error Back Propagation*) [5], or LM (*Levenberg-Marquardt*), are not able to effective train these new highly efficient architectures. It is important to note that not only architecture, but also the training algorithm is needed to solve given problem. Currently, the only algorithm that is able to teach the new architecture is the NBN (*Neuron by Neuron*) published recently in [6-8]. This algorithm can be used for all architectures with arbitrally connected neurons, including BMLP and DNN. This algorithm works well solving the problems impossible to solve by other algorithms.

Already now we can build intelligent systems, such as artificial neural networks, setting weights with random values initially, and then use an algorithm that will teach this system adjusting these weights in order to solve complex problems. It is interesting that such a system can achieve a higher level of competence than teachers. Such systems can be very useful wherever decisions are taken, even if the man is not able to understand the details of their actions. Neural networks helped solve thousands of practical problems. Most scientists used the MLP and the EBP algorithm. However, since the EBP algorithm is not efficient, usually using inflated the number of neurons which meant that the network with a high degree of freedom to consume their capabilities to learn the noise. Consequently, after the step of learning system was score responsive to the patterns that are not used during the learning, and it resulted in frustration. A new breakthrough in intelligent systems is possible due to new, better architectures and better, more effective learning algorithms.

2 Training Algorithms

Currently, the most effective and commonly known ANN training algorithms are algorithms based on LM[8]. Unfortunately, the LM algorithm is not able to teach other architectures than MLP. Because the size of Jacobian, which must be processed as proportional to the number patterns of learning. It means that LM algorithm may be used only for relatively small problems. Our newly developed second-order learning algorithm NBN [6-8] is even slightly faster than LM and allows to solve problems with a virtually unlimited number of patterns, and it may very effectively teach new powerful architecture of ANN, such as BMLP, FCC, whether DNN [1]. Using the NBN we can solve much more complex problems with more powerful system architectures.

Training of RBF (*Radial Basis Function*) network with the second order algorithm is even more complicated than training sigmoidal networks where are needed only to adjusted weights. Our preliminary research shows that if we can teach widths and locations of RBF centers it is possible to solve many problems in just a few units of RBF instead of hundreds sigmoid neurons.

The discovery of the EBP algorithm [5,9] started a rapid growth of computational intelligent systems. Thousands of practical problems have been solved with the help of neural networks. Although other neural networks are possible, the main accomplishments were noticed using feed forward neural networks using primarily MLP architectures. Although EBP was a real breakthrough, this is not only a very slow algorithm, but also it is not capable of training networks with super compact architectures [1,6]. Many improvements to the EBP were proposed, but most of them did not address the main faults of EBP. The most noticeable progress was done with an adaptation of the LM algorithm to neural network training [3]. The LM algorithm is capable of training networks with 100 to 1000 fewer iterations. The above mentioned LM algorithm [3,10] was adapted only for MLP architectures, and only relatively small problems can be solved with this algorithm because the size of the computed Jacobian is proportional to the number of training patterns multiplied by the number of network outputs. Several years ago adapted the LM algorithm to train arbitrarily connected feed forward.

ANN architectures [11], but still the problem of the number of patternâĂŹs limitations in the LM algorithm remained unsolved until recently when we developed the NBN algorithm [5]. Now we have a tool which is not only very fast, but we can train using second order algorithm problems with basically an unlimited number of patterns. Also NBN algorithm can train compact close to optimal architectures which cannot be trained by the EBP algorithm.

Both technologies (SVM and ELM) are adjusting only parameters, which are easy to adjust, like output weights, while other essential parameters such as radiuses of RBF units σ_h, and the location of centers of the RBF units c_h are either fixed or selected randomly. As a consequence, the SVM and ELM algorithms are producing significantly more networks than needed. From this experiment one may notice that the SVR (Support Vector Regression) [12,13] and the Incremental Extreme Learning Machine (I-ELM) [14], and the Convex I-ELM (CI-ELM) [15] need 30 to 100 more RBF units than the NBN [16], the ISO [17], and the ErrCor [18] algorithms. Another advantage of ErrCor is that there is no randomness in the learning process so only one learning process is needed, while in the case of SVM (or SVR) a lengthy and tedious trial and error process is needed before optimal training parameters are found.

3 Error Correction Algorithm Improvement

3.1 Error Correction Fundamentals

Error Correction (ErrCor) is the second order LM based algorithm that has been designed for RBF networks where as neurons RBF units with Gaussian activation function defined by (1) are used.

$$\varphi_h(x_p) = exp\left(-\frac{\|x_p - c_h\|^2}{\sigma_h}\right) \tag{1}$$

where: c_h and σ_h are the center and width of RBF unit *h*, respectively. $\|\cdot\|$ represents the computation of Euclidean Norm.

The output of such network is given by:

$$O_p = \sum_{h=1}^{H} w_h \varphi_h \left(x_p\right) + w_o \tag{2}$$

where: w_h presents the weight on the connection between RBF unit h and network output. w_0 is the bias weight of output unit. Note that the RBF networks can be implemented using neurons with sigmoid activation function [19,20].

The main idea of the ErrCor algorithm is increasing the number of RBF units one by one and adjusting all RBF units in network by training after adding of each unit. New unit is initially set to compensate largest error in the current error surface and after that all units are trained changing both centers and widths as well as output weights. Details of algorithm can be found in [18]. As can be found in [18] [21] ErrCor algorithm had

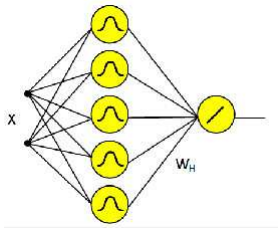

Fig. 1. RBF network architecture

been successfully used to solve several problems like function approximation, classification or forecasting. The main disadvantage of ErrCor algorithm is long computation time caused mainly by requirement of training of whole network at each iteration.

3.2 ErrCor Improvement

Long computation time depends on many factors. One of the most important is number of patterns used in training. We can reduce their number by removing from training dataset outlier patterns that includes data inconsistent with rest of patterns. This approach has been used in [21] to eliminate patterns that contain unusual data like hurricanes, political or criminal events. Such operation allows not only to reduce number of patterns or time of training but also to improve training results achieving lower training error and better generalization. The important issue is how to identify inconsistent patterns (outliers). We suggest to remove patterns for which error has higher value than Outlier Threshold (OT) that can be arbitrary assumed value. In our experiments OT was current MERR (*Mean Error*) dependent value given by:

$$OT = n * MERR \tag{3}$$

where n is typically in range (5-10).

Removing of outliers can be done after adding to network several number of units. Pseudo code of the enhanced ErrCor algorithm is shown below. Changes to original ErrCor algorithm [18] are bolded.

Improved ErrCor pseudo code

```
evaluate error of each pattern;
while 1
    C = pattern with biggest error;
    add a new RBF unit with center = C;
    train the whole network using ISO-based method;
    evaluate error of each pattern;
    calculate SSE = Sum of Squared Errors;
    if SSE < desired SSE
          break;
    end;
    after each N added RBF units remove outliers with error > OT;
end
```

Described mechanism has been successfully used in [21] to improve training process of RBF network for forecasting energy load. It allowed to achieve both better training error and validation error as well as lower training time.

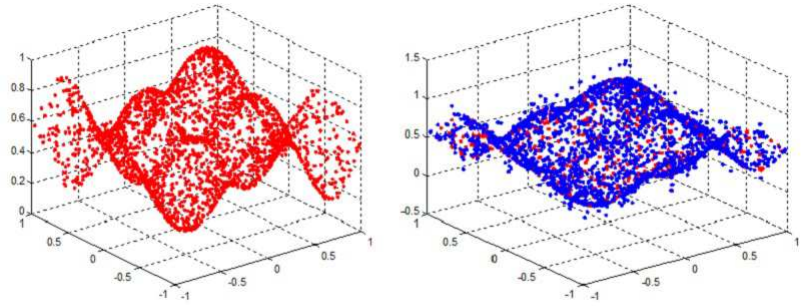

Fig. 2. Data for network learning - a) function Schwefel, b) noised function Schwefel

4 Results of Experiments

To confirm suggested approach several experiments with different dataset and training parameters have been prepared. The first experiment was approximation of noised Schwefel function. Noised function has been built by adding random values to about

20% of randomly selected Shwefel function samples. Original and noised function is shown in Figure 2. In presented experiments number 503 of 2500 samples ware noised. Such created data has been divided into training and testing datasets in the ratio of 4 to 1, to give 2000 training and 500 testing patterns. First, the training process to has been prepared using original ErrCor algorithm and next repeated for different values of parameter OT (from 1.5 to 4.65) and parameter N (5 and 10). In all experiments number of RBF units have been limited to 30. Results contain training MSE (*Mean Square Error*) and testing MSE are shown in Table 1.

Table 1. Results for approximation of Schwefel function with Improved ErrCor algorithm with different values of parameters N and OT

N	***OT***	***Training MSE***	***Testing MSE***
Original ErrCor		0.034665	0.035098
5	1.5	0.000046	0.008399
5	2	0.000129	0.004438
5	2.5	0.000224	0.004411
5	3	0.031460	0.034999
5	3.5	0.000182	0.004331
5	4	0.000619	0.004350
5	4.5	0.000605	0.004470
5	4.6	0.000654	0.004473
5	4.61	0.034665	0.035098
5	4.65	0.034665	0.035098
10	1.5	0.000402	0.005237
10	2	0.001586	0.005475
10	2.5	0.001342	0.004882
10	3	0.005639	0.006439
10	3.5	0.001856	0.004451
10	4	0.002803	0.004791
10	4.5	0.003958	0.004983
10	4.6	0.004075	0.005017
10	4.61	0.034665	0.035098
10	4.65	0.034665	0.035098

Results show that outliers removing allow to achieve better results than original ErrCor algorithm. Best testing MSE for Improved ErrCor have been achieved for OT=3.5 for both N=5 and N=10. Similarly, best training MSE for both N values have been achieved for the same value OT=1.5. This is because for lower value of OT much more patterns are removed during training process that causes better training. Table 2 shows the number of removed patterns during experiments. As can be observed for OT=5 number of removed patterns are higher than number of noised samples. Moreover, for best results with OT=3.5 number of removed patterns are lower than number of noised samples. Note, that for both values of N reaching the same value of OT=4.61 no outliers have been detected and removed that means results the same like for original ErrCor. Figure 3 shows training and testing process for original ErrCor. Training error is assigned by blue stars and testing error is assigned by red circles. It can be observed that best result is reached very quickly on the level of about 0.035 for both training and

Table 2. Number of removed patterns outliers

OT	*Neurons in network*	*Outliers for N=5*	*Outliers for N=10*
1.5	5	450	-
	10	361	450
	15	311	-
	20	202	319
	25	160	-
	Sum	1484	769
2	5	224	-
	10	197	224
	15	188	-
	20	151	199
	25	130	-
	Sum	890	423
2.5	5	71	-
	10	121	103
	15	89	-
	20	123	145
	25	111	-
	Sum	515	248
3	5	19	-
	10	9	19
	15	1	-
	20	0	9
	25	0	-
	Sum	29	28
3.5	5	10	-
	10	41	10
	15	60	-
	20	105	93
	25	93	-
	Sum	309	103
4	5	3	-
	10	19	3
	15	28	-
	20	82	61
	25	79	-
	Sum	211	64
4.5	5	1	-
	10	16	1
	15	10	-
	20	85	24
	25	92	-
	Sum	204	25
4.6	5	1	-
	10	13	1
	15	38	-
	20	55	22
	25	90	-
	Sum	197	23

testing datasets. Figures 4-7 show training process for selected values of OT and both analyzed N. As can be observer the training error is changing abruptly with removing of patterns while testing error is decreasing rather slowly but in the wake of changes of training error. The interesting results have been achieved for OT = 3 where both training and validating errors are relatively high and very close to each other. It means that for some values of OT training process can falls into local minimum and is not able to reach better results. This is especially visible in the case of N=5, where achiever result is not significantly better than for original ErrCor. In this case only 29 outliers have been removed during training process that was too small to eliminate noised patterns. On the second case, for N=10, better results have been obtained only for larger RBF network reaching testing MSE = 0.004294 and training MSE as low as 0.000046.

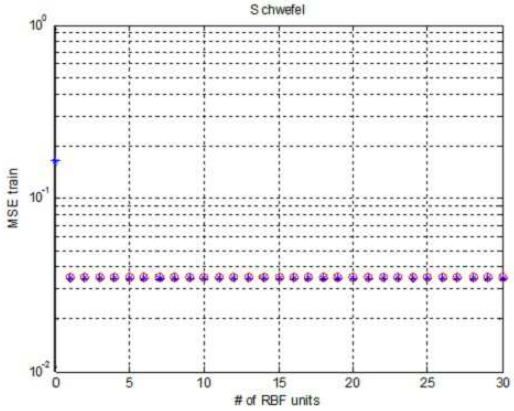

Fig. 3. The process of training with original ErrCor

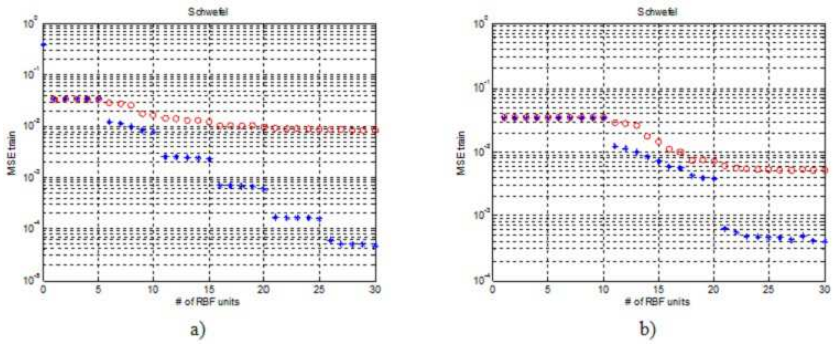

Fig. 4. The learning process modified algorithm ErrCor: a) OT=1.5, N=5, b) OT=1.5, N=10

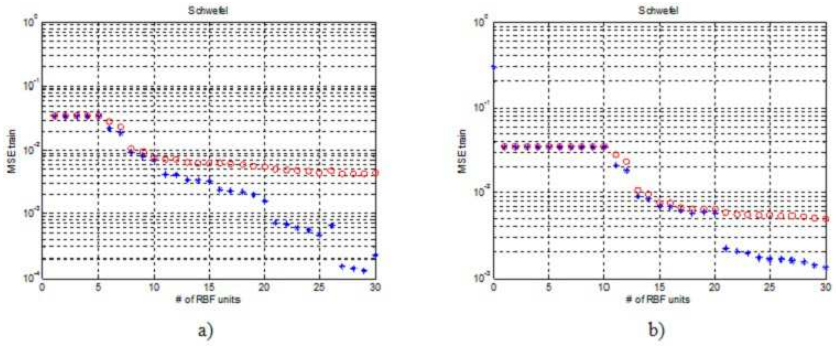

Fig. 5. The learning process modified algorithm ErrCor: a) OT=2.5, N=5, b) OT=2.5, N=10

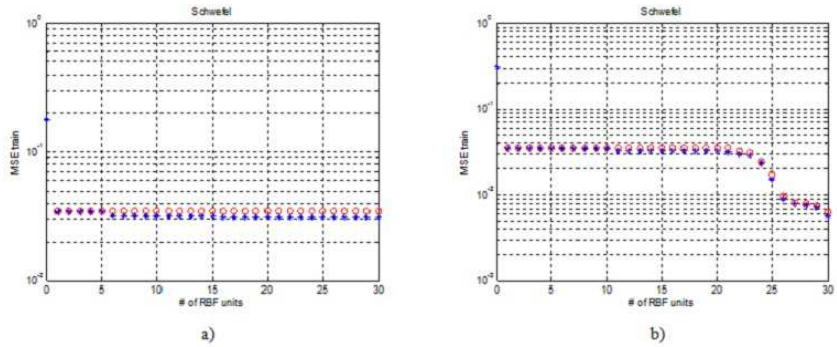

Fig. 6. The learning process modified algorithm ErrCor: a) OT=3, N=5, b) OT=3, N=10

In the second experiment have been used real world datasets from UCI Machine Learning Repository commonly used as a benchmarks, such as Airplane Delay, Machine CPU, Auto Price, California Housing. For each dataset results of original ErrCor algorithm has been compared to discussed modified ErrCor with parameters OT=5 and N=5. Results of these experiments are shown in Figure 8 and Figure 9. Again, blue stars for given number of RBF units are a training MSE, red circles are testing MSE. As can be observed outliers eliminating allows to reach better results for smaller number of units also for real world datasets.

5 Conclusions

The paper presents proposition of improvement for Error Correction algorithm by elimination of inconsistent patterns from training process. Achieved experimental results

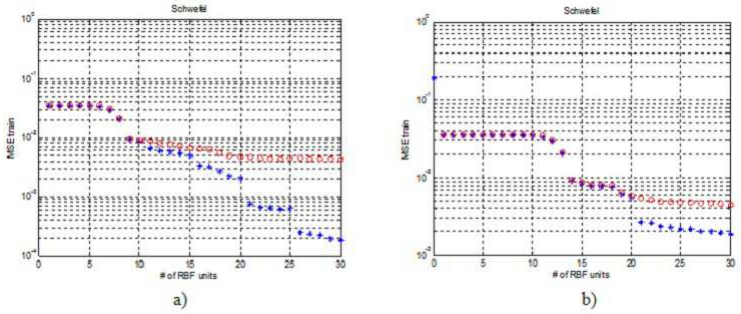

Fig. 7. The learning process modified algorithm ErrCor: a) OT=3.5, N=5, b) OT=3.5, N=10

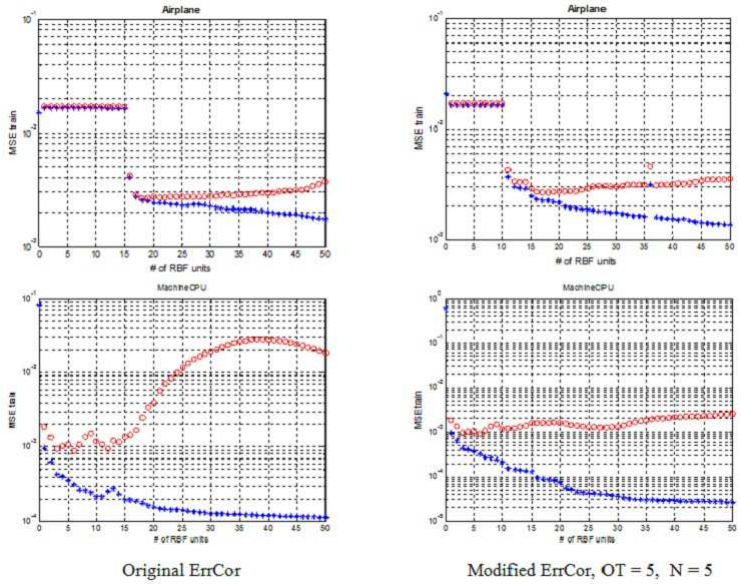

Fig. 8. Results achieved for Airplane Delay and Machine CPU datasets

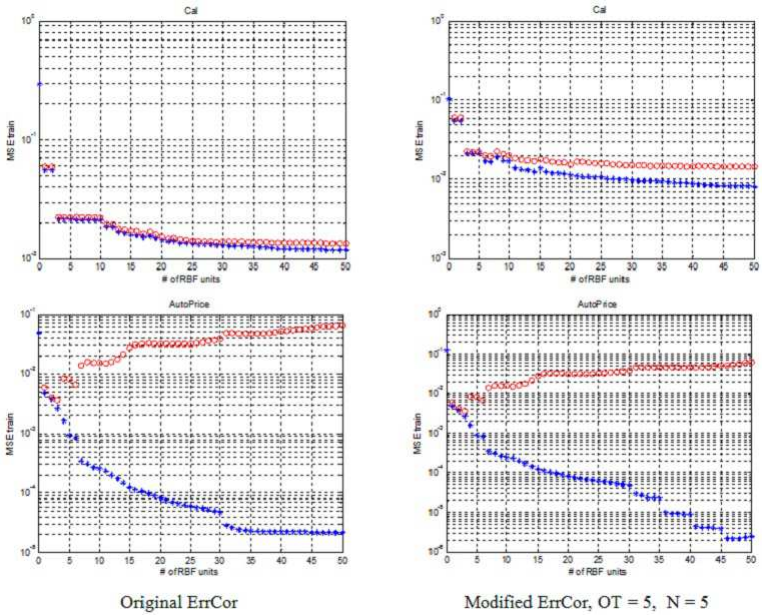

Fig. 9. Results achieved for California Housing and Auto Price datasets

confirm effectiveness of proposed method that was originally suggested in [21]. Mentioned effectiveness depends however on the content of processed dataset and will be higher for more noisy data with more random corrupted data, that will be easy eliminated. Further work in this area will be focused on improvement proposed approach by searching a way that allow to find optimal training parameters for given dataset, as well as applying presented method for other training algorithms such as ELM or NBN.

References

1. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", Nature, vol. 323, pp. 533-536, 1986.
2. G. E. Dahl, M. Ranzato, A. Mohamed, and G. E. Hinton, "Phone recognition with the mean-covariance restricted Boltzmann machine", NIPS, 2010.
3. Y. Bengio, "Learning deep architectures for AI", Foundations and Trends in Machine Learning, 2(1), 1-127. Also published as a book. Now Publishers, 2009.
4. B. M.Wilamowski, "Challenges in Applications of Computational Intelligence in Industrial Electronics", IEEE International Symposium on Industrial Electronics (ISIE 2010), JUL 04-07, 2010, pp. 15-22.
5. B. M. Wilamowski, "Neural Network Architectures and Learning algorithms- How Not to Be Frustrated with Neural Networks", IEEE Industrial Electronics Magazine, vol 3, no 4, pp.56-63, (2009) (best paper award).
6. D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep big simple neural nets excel on handwritten digit recognition", CoRR, 2010.

7. B. M. Wilamowski and H. Yu, "Neural Network Learning Without Backpropagation," IEEE Trans. on Neural Networks, vol. 21, no.11, pp1793-1803, Nov. 2010.
8. P. J. Werbos, "Back-propagation: Past and Future". Proceeding of International Conference on Neural Networks, San Diego, CA, 1, 343-354, 1988.
9. D. Hunter, Hao Yu, M. S. Pukish, J. Kolbusz, and B.M. Wilamowski, "Selection of Proper Neural Network Sizes and Architectures: Comparative Study", IEEE Trans. on Industrial Informatics, vol. 8, May 2012, pp. 228-240.
10. K. L. Lang, M.J. Witbrock, "Learning to Tell Two Spirals Apart" Proceedings of the 1988 Connectionists Models Summer School, Morgan Kaufman.
11. S. E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture", In D. S. Touretzky (ed.) Advances in Neural Information Processing Systems 2 . Morgan Kaufmann, San Mateo, CA, 1990, pp. 524-532.
12. V. N. Vapnik, Statistical Learning Theory. New York: Wiley, 1998.
13. Smola and B. Scholkopf, "A tutorial on support vector regression", NeuroCOLT2 Tech. Rep. NC2-TR-1998-030, 1998.
14. G.-B. Huang; L. Chen; C.-K. Siew , "Universal approximation using incremental constructive feedforward networks with random hidden nodes", IEEE Transactions on Neural Network, vol.17, no.4, pp. 879- 892, July 2006.
15. G. B. Huang and L. Chen, "Convex incremental extreme learning machine", Neurocomputing, vol. 70, no. 16-18, pp. 3056-3062, Oct. 2007.
16. B. M. Wilamowski, H. Yu, "Improved Computation for Levenberg Marquardt Training," IEEE Trans. on Neural Networks, vol. 21, no. 6, pp. 930-937, June 2010.
17. H. Yu, T. Xie, J. Hewlett, P. Rozycki, B. Wilamowski, "Fast and Efficient Second Order Method for Training Radial Basis Function Networks", IEEE Transactions on Neural Networks, 2012, Vol. 24, Issue: 4, pp. 609-619.
18. H. Yu, P. Reiner, T. Xie, T. Bartczak, B. Wilamowski, "An Incremental Design of Radial Basis Function Networks" IEEE Trans. on Neural Networks and Learning Systems, vol 25, No. 10, Oct 2014, pp. 1793- 1803.
19. B. M. Wilamowski, R. C. Jaeger, "Implementation of RBF type networks by MLP networks", 1996 IEEE International Conference on Neural Networks (ICNN 96), pp. 1670-1675.
20. X. Wu, B. M. Wilamowski, "Advantage analysis of sigmoid based RBF networks", Proceedings of the 17th IEEE International Conference on Intelligent Engineering Systems (INES'13). 2013. p. 243-248.
21. C. Cecati, J. Kolbusz, P. Rozycki, P. Siano, B. Wilamowski, "A Novel RBF Training Algorithm for Short-Term Electric Load Forecasting and Comparative Studies", IEEE Trans. on Ind. Electronics, Eearly Access, 2015.