▶ KNUV 2018; 2(56): 257-268

Paweł Różycki, Janusz Kolbusz
Wyższa Szkoła Zarządzania i Informatyki w Rzeszowie
Tomasz Bartczak
Akademia Finansów i Biznesu Vistula – Warszawa

EFFICIENT TRAINING OF RBF NEURAL NETWORKS

Summary

RBF networks seem to be an interesting and efficient alternative for traditional sigmoid-based neural networks. More sophisticated activation function makes a network more powerful but requires developing of new training methods. The paper presents a new more efficient training algorithm based on the second-order constructive ErrCor algorithm. The effectiveness of the proposed approach has been confirmed by several experiments with both approximation and classification problems.

Key words: error correction, ErrCor, RBF networks, training algorithms.

JEL codes: L86

Introduction

Our civilization encounters increasingly complex problems that often exceeds human capabilities. Until now, 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 and find effective learning algorithms that allow faster and more effectively teach a network using its properties. Both of problems are strictly connected.

The rapid development of intelligent computational systems allowed to solve thousands of practical problems using neural networks. We can build intelligent systems, 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. Most scientists use the MLP (*Multi-Layer Perceptron*) architecture and the EBP (*Error Back Propagation*) algorithm [1][6]. However, since the EBP algorithm is not efficient, usually using inflated the number of neurons which mean that the network with a high degree of freedom 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.

Most visible progress in this field was develop the LM (*Levenberg-Marquardt*) algorithm to train the neural network. This algorithm is able to teach the network by 100 to 1000 times less iterations than traditional EBP, but its usage to more complex problems is significantly limited, since the size of the Jacobian matrix is proportional to the number of patterns.

In order to solve more and more complex problems with the use of neuron networks we should thoroughly understand the neural network architecture and its impact on the operation of the system and finally develop appropriate processes of learning these networks. Modification of existing algorithms and development of new algorithms for network learning will allow for faster and more effective network teaching.

Often used networks MLP have limited capabilities [1], but new deeper neural network architectures like BMLP (Bridged MLP) [1][2][3] or DNN (*Dual Neutral Networks*) [2] with the same number of neurons can solve problems up to 100 times more complex [2][4]. Therefore, it can be concluded that the way neurons interconnections in the network is fundamental. The use of appropriate architecture has a significant impact on the solution of given problem. An example can be FCC (*Fully Connected Cascade*) network architecture. Such a network with 10 neurons can solve the Parity-1023 problem, while the most widely used the MLP architecture network with 10 neurons in the three-tiered, one hidden layer, architecture, is able to solve Parity-9 problem. Thus, moving away from the commonly used architecture MLP, while maintaining the same number of neurons can increase network capacity, even a hundred times [2][4][5]. However, a problem arises in that the currently known network learning algorithms, such as EBP, or LM do not deal with such network architectures. LM algorithm is not able to teach other architectures than the MLP and limited number of possible patterns restricts LM algorithm for solving a relatively small

problems. The only known algorithm that can learn these new architectures is NBN (*Neuron-by-Neuron*) algorithm [7][8][9]. It is faster than LM and can be used for all architectures, including BMLP, FCC, DNN and MLP, of course, and gives good learning results. However, ISO algorithm [10] and published in 2014 ErrCor (*Error Correction*) algorithm [11] allow to get even better results showing that the shallow architecture can be still a good alternative.

RBF network training

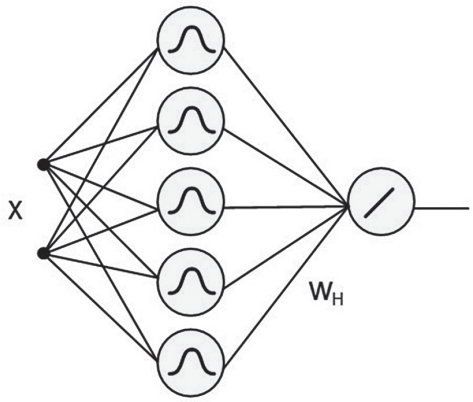
RBF networks fundamentals

RBF network is shown in Fig.1. In such network training units with Gaussian activation function defined by (1) are used.

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

where: and are the center and width of RBF unit *h*, respectively. represents the computation of Euclidean norm.

Figure 1. RBF network architecture



Source: own preparation.

The output of such network is given by:

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

where: presents the weight on the connection between RBF unit *h* and network output. is the bias weight of output unit. Note that the RBF networks can be implemented using neurons with sigmoid activation function in MLP architecture [13].

The most popular shallow technologies such as ELM (*Extreme Learning Machine*) [15] and SVM (*Support Vector Machine*) [16] [17] are adjusting only parameters, which are easy to adjust, like output weights w_h, while other essential parameters such as radiuses of RBF units , 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 experiments shown in [8][10][11] one may notice that the SVM, ELM but also more sophisticated Incremental Extreme Learning Machine (I-ELM) [18], and the Convex I-ELM (CI-ELM) [19] need 10 to 200 times more RBF units than the NBN, the ISO, and the ErrCor 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 a lengthy and tedious trial and error process is needed before optimal training parameters are found.

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, so ErrCor algorithm is not only deterministic but also constructive that allow to achieve a networks with proper number of units. The 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 [11].

As can be found in [11][14] ErrCor algorithm had been successfully used to solve several approximation, classification problems or forecasting applications. The main disadvantage of ErrCor algorithm is long computation time caused mainly by requirement of training of whole network at each iteration.

Enhanced Error Correction algorithm

Long computation time depends on many factors. One of the most important is number of patterns used in training and long training of whole network after adding of next RBF unit. In order of improve this process we suggest the following modifications of ErrCor algorithm [12]: after adding new RBF unit only this new unit is trained using LM-based method used in ErrCor algorithm [11] and after that all output weights are justified using regression and after added N new RBF units whole network is trained using the same LM-based method used in ErrCor algorithm where N is arbitrary assigned value.

Such modification allow to shortened training process because critical whole network training process is limited to cases when N new units are added to network. In the other cases the training is much faster because in fact trained is only one RBF unit and regression is quite small time absorbing process.

Pseudo code of the Enhanced ErrCor algorithm is shown below. Changes to the original ErrCor algorithm [11][12] are bolded.

```
evaluate error of each pattern;
while 1
   C = pattern with biggest error;
   add a new RBF unit with center = C;
   if N new RBF units are added
     train the whole network using LM-based method;
   else
     train only one new added RBF unit using LM-based method;
     adjust output weights for whole network by regression}}
   end
   evaluate error of each pattern;
   calculate SSE = Sum of Squared Errors;
   if SSE < desired SSE
   break;
   end;
  end
```

In the next section some experimental results for this approach is presented.

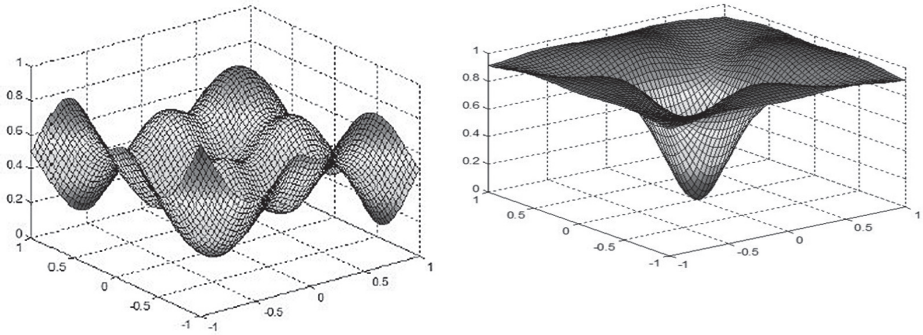
Results of Experiments

To confirm suggested approach several experiments for different approximation benchmark functions and classification problem with selected parameters have been prepared. The following approximation benchmark functions have been selected: Schwefel function and Second Shaffer function. In the next three subsections the ErrCor algorithm and the Enhanced ErrCor algorithm have been used to solve approximation problem of mentioned functions and in the last subsection the classification problem is shown. In all approximation experiments 900 training patterns and 3481 testing patterns have been generated. For such prepared data series of experiments have been done with different values of parameter N and compared to results achieved using original ErrCor algorithm.

Schwefel Function

First experiment was prepared for Schwefel function given by and shown in Fig.2.a

Figure 2. Surface of normalized benchmark functions used in experiments



a) Schwefel function b) Second Shaffer function

Source: as in Figure 1.

Results achieved for Schwefel function are shown in Tab.1. Result for original ErrCor that can be treated as a reference is denoted as OrgErrCor. Parameter N means the number of units that are added to network between full training. The case when training process is done without full network training is denoted as X in column N. The RMSE is Root Mean Square Error given by:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(out_T - out_E)^2}{n}}$$

where out_T is the output of trained network and out_E is expected value and n is the number of patterns.

Table 1. Results for Schwefel function.

N	Training time [s]	Training RMSE	Testing RMSE
OrgErrCor	246.6722	0.0038543	0.0038299
2	215.7920	0.0032388	0.0031717
3	148.7899	**0.0030517**	**0.0029844**
4	88.0451	0.0052154	0.0051204
5	75.5889	0.0053222	0.0051949
6	59.7885	0.0052533	0.0051889
7	51.9710	0.0057257	0.0057030
8	66.9600	0.0079042	0.0077081
9	60.3580	0.0054597	0.0053507
10	51.2228	0.0059679	0.0058742
15	43.1061	0.0105723	0.0104105
30	36.2405	0.0497498	0.0498103
X	21.1715	0.0747471	0.0732735

Source: own preparation.

As shown in Tab.1 training time decreases with increased value of N. This is obvious because frequency of full training, that is the most time consuming part of training process is smaller for higher N. More important is that values of testing and training RMSE for small values of N (2 and 3) are better than these achieved with original ErrCor, and for higher value of N are only slightly worse. Note that results for N=10 are only 53% worse but achieved almost 5 times faster.

Second Shaffer function

The second experiment have been done for Second Shaffer function. This function is given by

$$z(x,y) = 0.5 + \frac{sin^2(x^2 - y^2) - 0.5}{[1 + 0.001\,(x^2 - y^2)]^2}$$

shown in Fig.2.b

Results achieved with Enhanced Error Correction algorithm is shown in Tab.2 Similarly to Schwefel function in this case training time decreases with N while RMSE is relatively are close to or even lower than for original ErrCor.

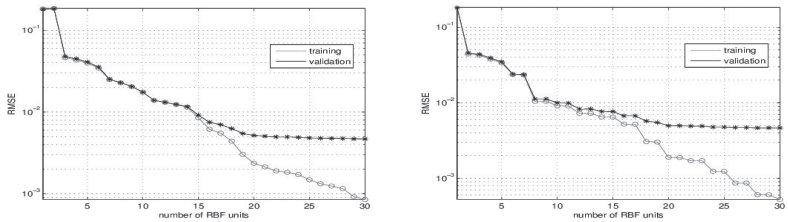
Table 2. Results for Second Shaffer function

N	Training time [s]	Training RMSE	Testing RMSE
OrgErrCor	262.1465	0.0008487	0.0046744
2	160.7057	**0.0005260**	0.0046379
3	140.3324	0.0005305	**0.0046374**
4	123.2476	0.0008320	0.0051382
5	106.8903	0.0014932	0.0047826
6	50.9329	0.0018604	0.0049701
7	44.1457	0.0011056	0.0047119
8	43.9627	0.0010136	0.0046913
9	28.2079	0.0018110	0.0049119
10	57.9185	0.0010418	0.0047068
15	21.7188	0.0017827	0.0049438
30	17.5358	0.0093627	0.0103736
X	14.4285	0.0093627	0.0103736

Source: as in Table 1.

Fig. 3 show the training process that show changes of training and testing RMSE during training process. Training RMSE is drown as blue circles and testing RMSE is drawn as a red stars. As can be observed in the case of Enhanced ErrCor with N = 2 reaches similar result as ErrCor but is able to obtain same results faster and using less neurons.

Fig. 3. Training process for approximation of Second Shaffer function with: (a) original ErrCor algorithm, (b) Enhanced ErrCor (N=2)



Source: as in Figure 1.

California Housing Problem

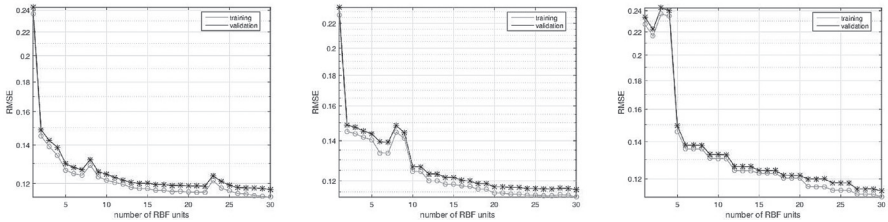
As a classification benchmark the California Housing problem has been selected [21]. This is eight-dimensional dataset that contains 20640 patterns. In the presented experiment patterns have been divided into 14448 training and 6192 testing datasets. The results achieved for the same training parameters as for approximation problems (30 RBFs and selected N values) have been shown in Tab.3. The same as for Schwefel and Second Shaffer functions, both training and testing RMSE with *N*=2 and *N*=3 for Enhanced ErrCor algorithm are better than for original ErrCor, and again much faster. Fig.4 shows training process for original ErrCor (a) and for Enhanced ErrCor with *N*=2 (b) and *N*=3 (c) and Tab.4 shows partial results achieved for selected number of RBFs with *N*=2 and *N*=3. As can be observed results better than original ErrCor is achieved with 20 RBF units for *N*=2 and with 25 RBF units for *N*=3.

Table 3. Results for California Housing dataset

N	Training time [s]	Training RMSE	Testing RMSE
OrgErrCor	3782.5000	0.0129143	0.0137180
2	2057.5655	0.0126966	0.0134522
3	2336,0800	**0.0123965**	**0.0130389**
4	1430,9556	0.0148314	0.0155695
5	491.5510	0.0145035	0.0151981
6	561.1930	0.0147205	0.0153744
7	953.4788	0.0144834	0.0151674
8	474.6803	0.0164396	0.0172553
9	839.1555	0.0156160	0.0161944
10	773.5647	0.0138860	0.0144642
15	995.3262	0.0150111	0.0157218
30	729.7942	0.0163775	0.0169681
X	156.1334	0.0163775	0.0169681

Source: as in Table 1.

Fig. 4. Training process for California Housing benchmark: (a) original ErrCor algorithm, (b) Enhanced ErrCor (N=2), (c) Enhanced ErrCor (N=3)



Source: as in Figure 1.

Table 4. Results for California Housing dataset for different number of RBF units

RBF No.	N	Training time [s]	Training RMSE	Testing RMSE
5	OrgErrCor	190.9289	**0.0160262**	**0.0168709**
	2	107.0947	0.0197249	0.0206772
	3	24.8798	0.0212651	0.0223195
10	OrgErrCor	433.4176	**0.0147809**	**0.0154959**
	2	468.5468	0.0154538	0.0160404
	3	211.5881	0.0170356	0.0176059
15	OrgErrCor	817.5618	**0.0138127**	**0.0144362**
	2	788.9896	0.0140156	0.0147957
	3	387.5943	0.0150842	0.0154397
20	OrgErrCor	1653.0473	0.0134411	0.0141174
	2	1310.7616	**0.0130941**	**0.0137332**
	3	709.6542	0.0144840	0.0148200
25	OrgErrCor	2773.6050	0.0135591	0.0142076
	2	1561.3274	**0.0128821**	**0.0135679**
	3	1615.7546	0.0131249	0.0139395
30	OrgErrCor	3782.5000	0.0129143	0.0137180
	2	2057.5655	0.0126966	0.0134522
	3	2336.0800	**0.0123965**	**0.0130389**

Source: as in Table 1.

Conclusions

Achieved results confirm effectiveness of suggested method for improvement Error Correction algorithm that is currently one of the most powerful for training RBF networks. Proposed modification allows to reduce training time in most cases without losses of low training and testing errors.

Further work will be focused on improvement of proposed algorithm by correction of method for selection of initial localization for new RBF units and on applying described algorithm for wider spectrum of functions and real world classification and regression datasets.

Bibliography

Bengio Y. (20090, *Learning deep architectures for AI*, "Foundations and Trends in Machine Learning", No. 2(1).

Cecati C., Kolbusz J., Różycki P., Siano P., Wilamowski B. (2015), *A Novel RBF Training Algorithm for Short-Term Electric Load Forecasting and Comparative Studies*, "IEEE Transactions on Industrial Electronics".

Ciresan D.C., Meier U., Gambardella L.M., Schmidhuber J. (2010), *Deep big simple neural nets excel on handwritten digit recognition*, CoRR.

Huang G.-B., Chen L., Siew C.-K. (2006), *Universal approximation using incremental constructive feedforward networks with random hidden nodes*, "IEEE Transactions on Neural Network", Vol. 17, No. 4.

Huang G.-B., Chen L. (2007), *Convex incremental extreme learning machine*, "Neurocomputing", Vol. 70, No. 16-18.

Huang G.-B., Zhu Q.-Y., Siew Ch.-K. (2006), *Extreme learning machine: theory and applications*, "Neurocomputing", Vol. 70, Iss. 1-3.

Fahlman S.E., Lebiere C. (1990), *The cascade-correlation learning architecture,* (in:) D.S. Touretzky (ed.), *Advances in Neural Information Processing Systems 2*, Morgan Kaufmann, San Mateo.

Lang K.L., Witbrock M.J. (1988), *Learning to Tell Two Spirals Apart*, Proceedings of the 1988 Connectionists Models Summer School, Morgan Kaufman.

Płaczek S., Adhikari B. (2014), *Analysis of MultiLayer Neural Networks with Direct and Cross Forward Connection*, "Fundamenta Informaticae", Vol. 133, No. 2-3.

Różycki P., Kolbusz J. (2015), *Enhanced Error Correction Algorithm for RBF Neural Networks*, Proceedings of 24th International Workshop Concurrency, Specification & Programming, (CS&P'15), Rzeszów, September 28-30.

Rumelhart D.E., Hinton G.E., Williams R.J. (1986), *Learning representations by back--propagating errors*, "Nature", Vol. 323.

Smola A.J., Schölkopf B. (1998), *A tutorial on support vector regression*, NeuroCOLT2 Technical Report NC2-TR-1998-030.

Vapnik V.N. (1998), *Statistical Learning Theory,* Wiley, New York.

Werbos P.J. (1988), *Back-propagation: Past and Future*, Proceeding of International Conference on Neural Networks, San Diego.

Wilamowski B.M. (2010), *Challenges in Applications of Computational Intelligence in Industrial Electronics*, "IEEE International Symposium on Industrial Electronics", July 04-07.

Wilamowski B.M., Yu H. (2010), *Neural Network Learning Without Backpropagation*, "IEEE Transactions on Neural Networks", Vol. 21, No. 11.

Wu X., Wilamowski B.M. (2013), *Advantage analysis of sigmoid based RBF networks,* Proceedings of the 17th IEEE International Conference on Intelligent Engineering Systems (INES'13).

Yu H., Xie T., Hewlett J., Różycki P., Wilamowski B. (2012), *Fast and Efficient Second Order Method for Training Radial Basis Function Networks*, "IEEE Transactions on Neural Networks", Vol. 24, Iss. 4.

Yu H., Reiner P., Xie T., Bartczak T., Wilamowski B. (2014), *An Incremental Design of Radial Basis Function Networks*, "IEEE Transactions on Neural Networks and Learning Systems", Vol. 25, No. 10.

http://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html [access: 01.10.2017].

Skuteczne szkolenie w zakresie sieci neuronowych radialnych funkcji bazowych (RBF)

Streszczenie

Sieci radialnych funkcji bazowych (RBF) wydają się ciekawą i skuteczną alternatywą dla tradycyjnych sieci neuronowych opartych na sigmoidach. Bardziej zaawansowana funkcja aktywująca czyni sieć potężniejszą, ale wymaga opracowania nowych metod szkolenia. Artykuł przedstawia nowy, bardziej skuteczny algorytm szkolenia oparty na konstruktywnym algorytmie drugiego rzędu ErrCor. Skuteczność proponowanego podejścia została potwierdzona przez kilka eksperymentów zarówno z problemami aproksymacyjnymi, jak i klasyfikacyjnymi.

Słowa kluczowe: korekta błędów, ErrCor, sieci RBF, algorytmy szkolenia.

Kody JEL: L86

This work was partially supported in part by the National Science Centre, Krakow, Poland, under Grant No. 2015/17/B/ST6/01880

Artykuł zaakceptowany do druku w kwietniu 2018 roku.

Afiliacje:
sr inż. Paweł Różycki
dr inż. Janusz Kolbusz
Wyższa Szkoła Informatyki i Zarządzania
Wydział Informatyki Stosowanej
Katedra Elektroniki i Telekomunikacji
ul. Sucharskiego 2
35-225 Rzeszów
e-mail: prozycki@wsiz.rzeszow.pl
e-mail: jkolbusz@wsiz.rzeszow.pl

dr inż. Tomasz Bartczak
Akademia Finansów i Biznesu Vistula
Wydział Informatyki
ul. Stokłosy 3
02-787 Warszawa
e-mail: t.bartczak@vistula.edu.pl