Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Performance comparison of second order conjugate algorithms in neural networks for predictive data mining

Parveen Sehgal [1, *], Sangeeta Gupta [2], Dharminder Kumar [3]

[1]*Department of Computer Science and Engineering, NIMS University, Jaipur, Rajasthan-303121, India*
[2]*Guru Nanak Institute of Management, Guru Gobind Singh Indraprastha University, New Delhi-110026, India*
[3]*Guru Jambheshwar University of Science and Technology, Hisar, Haryana-125001, India*

ARTICLE INFO

Article history:
Received 19 March 2017
Received in revised form
15 May 2017
Accepted 10 June 2017

Keywords:
Artificial neural networks
Conjugate gradient
Line search
Numerical optimization
Prediction modeling

ABSTRACT

In this paper, a performance comparison of several variations of the non-linear conjugate gradient method has been investigated. Neural Network-based prediction models for life insurance sector have been developed and their training has been done with a variety of first and second order algorithms to find an efficient training algorithm, but keeping the focus on conjugate gradient based methods. Traditional second order methods require computation of second order derivatives and need to compute hessian for quadratic termination; which is a tedious and memory consuming task. Here we employ conjugate gradient methods which bypass the computation of hessian, but still achieve quadratic termination and thus prove to be memory efficient.

1. Introduction

Prediction modeling is gaining popularity and plays an important role in all the important areas. It is concerned with the prediction of future probabilities or trends, and to analyze these trends a variety of traditional statistical methods and modern methods are available with their own pros and cons. Applications based on traditional methods like regression techniques, decision trees based prediction, naive bayes classifiers etc. have been developed in the recent years. But due to their limitations to deal with and learn the complex data which is usually present in real life situations; we have a need to develop new methods. Novel techniques like neural networks, genetic algorithms, evolutionary algorithms, fuzzy based techniques, support vector machine, hybrid techniques (Sundarkumar and Ravi, 2015; Keramati et al., 2014) are some of the new upcoming techniques and are in the development stage. In this paper, we develop prediction models based upon artificial neural networks for the insurance sector and compare the convergence behavior of first order and second order algorithms applied for the training of neural networks, especially the conjugate gradient-based techniques.

Neural networks can learn the complex relationships present in the real-life situations and fit well for the development of prediction models. They can be tuned to learn the historical trends present in the large datasets and adapt to new patterns without having any initial hypothesis. There exist a variety of ANN architectures and a number of training techniques of first and second order to train the neural networks. In all the methods, the main idea is to minimize the gradient of error function during training of the network and reach a state of minimum gradient value.

Researchers have developed a variety of gradient-based techniques of first and second order for optimizing network parameters to converge towards the minimum of multi-dimensional error gradient function (Antoniou and Lu, 2007). Major drawback in first order methods like steepest descent is that learning rate has a fixed value and if we keep the learning rate low for the safe convergence then it takes a long time to achieve the minimum of error gradient and if it is kept very high then solution can oscillate near point of convergence and will never converge. Even if the learning rate is kept adaptive as in the case of adaptive learning or adaptive momentum techniques, convergence is slow because of the absence of second order term (Rehman and Nawi, 2012).

On the other hand, second order conjugate gradient technique and its variations use the line

* Corresponding Author.
Email Address: parveensehgal@gmail.com (P. Sehgal)
https://doi.org/10.21833/ijaas.2017.08.010
2313-626X/© 2017 The Authors. Published by IASE.

search method to adapt the learning rate parameter and also avoid the computation of second order hessian matrix; as done in older techniques. Here we present a comparative study of various conjugate gradient techniques and other second order methods. To investigate the performance and efficiency of conjugate techniques, we have developed prediction models in MATLAB Neural Network Toolbox (MathWorks, 2012) and trained the neural networks by applying the variety of training algorithms under consideration. Datasets have been imported from life insurance data warehouse to develop and test the models.

2. Literature review

First order algorithms like gradient descent and its alternatives fail to find the solution even for slightly non-linear cases. These methods take very long time to converge and therefore are not beneficial in such kind of practical situations (Meza, 2010). In addition, when error surface is highly multi-dimensional and irregular, then convergence becomes very difficult and unattainable, as these methods employ a smaller and fixed step size for learning (Rehman and Nawi, 2012).

Instead, second order methods like Newton, quasi–Newton, Levenberg Marquardt, conjugate gradient variations and similar methods are preferred (Slavici et al., 2016), which are more efficient while training in non-linear cases (Fletcher, 2013; Shepherd, 2012).

Second order derivatives enhance the speed of convergence and achieve faster learning in comparison to other methods, which only utilize first order derivatives. Newton's method computes the second order derivatives of Taylor’s approximation as hessian matrix and therefore finds out the point of minimum much faster than first order methods (Shanthi et al., 2009; Yu and Wilamowski, 2012). This utilizes the curvature information to search a more direct route towards the point of minima. But computation of higher order derivatives in hessian consumes more memory space and creates a problem when weight vector and input vector are very large in size (Nocedal and Wright, 2006; Rojas, 2013).

For an improvement, quasi–Newton methods evaluate an approximation for the hessian, but these methods need more computations during each iteration and which in turn may increase the convergence time (Castillo et al., 2006; Robitaille et al., 1993). In Gauss–Newton method sum of squared function values is minimized to avoid the tedious and memory consuming computation of second order derivatives, but the main disadvantage is that estimation results strongly depend on upon the initial selection of the input parameters.

Conjugate gradient methods (CGM) prove superior to quasi–Newton methods in computational terms, when the neural network weight vector is large in size because it bypasses computation of second order derivatives (Haykin, 1994). CGM searches the required solution by moving along successive conjugate non-interfering directions without spoiling minimization during previous steps (Hager and Zhang, 2006a). It utilizes line search technique to calculate the optimal step size for the next iteration and moves along the search direction by taking a jump on the path of descent. Line search eliminates the need to evaluate memory consuming hessian matrix of second derivatives, but the involvement of line search offers a bottleneck in all iterations.

Levenberg Marquardt Algorithm (LMA) denotes an interpolation between Gauss–Newton method and steepest descent method and is considered as a trust region approach for Gauss–Newton method (Pujol, 2007). LMA is more robust than Gauss–Newton method because it converges towards minimum even if its starting point is far away from the point of final convergence. This technique provides a numerical solution to the problems, which are generally non-linear in nature.

As suggested by researchers, the computation of parameter to decide for new orthogonal directions is possible in a number of different ways giving rise to different variations of conjugate methods (Hager and Zhang, 2006a). The scaled conjugate gradient method (SCGM) avoids the complex and time-consuming line search along conjugate directions in each of iteration, by combining the trust region approach from the LMA with the CGM approach resulting in improved convergence. But, estimation of the second order derivatives may be an expensive step in SCGM (Møller, 1993).

3. Materials and methods

3.1. Conjugate gradient methods

Quadratic approximation for the error function $E(W_K)$ in a neighboring point of W_K, till second order term is given by Taylor’s expansion as shown below:

$$E(W_K + z) \approx E(W_K) + E'(W_K)^T z + \frac{1}{2} z^T E"(W_K) z$$

Because of a large number of weights in the weight vector, it is very time-consuming and takes more memory to compute second-order derivatives $E"(W_K)$, the hessian.

Therefore, to avoid computation of hessian, second order conjugate gradient methods perform a line search along conjugate directions in every iteration to search for optimal step size and to minimize the performance function and thus achieve faster convergence than simple gradient descent directions.

Researchers have developed many variations of CGM and to define scalar ‘Z’ (as given in following equations); which decide for the selection of the next search direction (Hager and Zhang, 2006a; Andrei, 2006). Variations like Fletcher-Reeves, Polak-Ribiere, Powell-Beale, Hestenes-Stiefel, Daniel, Dai-

Yuan and some new variations like Hager-Zhang variation are usually applied. CGM consumes comparatively less memory for problems involving large data sets but their convergence is poor in comparison to Newton or quasi–Newton methods due to the involvement of line search in every step.

Methods of conjugate gradients are capable of training any neural network provided that derivative functions exist for weights, inputs, and transfer functions. Back-propagation technique of supervised learning is applied to compute derivatives of performance function with respect to the weights and bias variables X_{k+1} in the successive iteration. New gradient points are computed according to Eq. 1 (MathWorks, 2012; Sandhu and Chhabra, 2011):

$$X_{K+1} = X_K + a^{*} dX_{K+1} \tag{1}$$

Where, the parameter *'a'* is calculated by a suitable line search algorithm to minimize the performance along the new search direction dX_{k+1}. In the first iteration, starting search direction dX_0 is kept as negative of the initial gradient value. But in the succeeding iterations, new search directions are computed from the new gradient values and from the previous search directions, according to Eq. 2 (MathWorks, 2012; Sandhu and Chhabra, 2011):

$$dX_{K+1} = -gX_{K+1} + Zd_k \tag{2}$$

Where *'gX_{k+1}'* is the current gradient and the direction selection parameter *'Z'* can be calculated in several different ways giving rise to a variety of conjugate variations.

3.1.1. Fletcher-Reeves update

For the Fletcher-Reeves variation of the conjugate gradient, computation of *'Z'* is done as shown in Eq. 3 (Hager and Zhang, 2006b; Hagan et al., 1996):

$$Z = \frac{\|g_{k+1}\|^2}{\|g\|^2} \tag{3}$$

3.1.2. Polak-Ribiere update

For the Fletcher-Reeves variation of the conjugate gradient, computation of *'Z'* is done as shown in Eq. 4 (Hager and Zhang, 2006b; Hagan et al., 1996):

$$Z = \frac{(g_{k+1} - g_k)' \, g_k}{\|g\|^2} \tag{4}$$

The memory requirements for Polak-Ribiere Update (it computes four vectors) are slightly more than Fletcher-Reeves (it computes three vectors).

3.1.3. Powell-Beale restarts

In conjugate gradient based methods, to improve the efficiency of the algorithm, the search direction is reset to the negative of current gradient value periodically when the number of network parameters becomes equal to the number of iterations. But researchers have also proposed other reset methods. Powell suggested a modification of the Beale restart and the technique restarts depending on orthogonality left for the new search direction becomes very less. This is implemented using the following inequality (Hager and Zhang, 2006b):

$$g_k g_{k+1}| \geq 0.2\|g_{k+1}\|^2 \tag{5}$$

3.2. Scaled conjugate gradient descent

A computationally expensive line search is required (Tezel and Buyukyildiz, 2016) in all iterations by conjugate gradient methods discussed so far. Møller (1993) and Chel et al. (2011) suggested the solution for avoiding the time-consuming line search in scaled conjugate gradient method (SCGM) by combining the model-trust region approach with the conjugate-gradient approach (Borkar et al., 2016). Memory requirements for SCGM are similar to Fletcher-Reeves variation of CGM. The algorithm trains any neural network as long as derivative functions exist for weight, net input, and transfer functions.

Back-propagation is applied to compute derivatives of performance function with respect to the weights and bias variables X_{k+1} in the successive iteration. SCGM depends on the computation of conjugate directions but avoids time-consuming line search in every iteration. In addition, this method avoids the tedious and memory consuming computation of hessian; done in traditional second order methods (Møller, 1993; Chel et al., 2011).

4. Experimental observations and results

4.1. Training the predictive models based on ANN

To compare the performances of algorithms under consideration, a number of model simulations have been developed and tested with varying parameters in MATLAB Neural Network Toolbox (MathWorks, 2012). To develop and test the prediction models large datasets from life insurance data warehouse have been taken. First order algorithms like *traingd, traingda, traingdm* and second order methods like *tarincgp, traincgf, tarincfb, tarinscg, trainlm* have been tested for their performance.

For experimentation initially, an MLP net object in neural toolbox has been configured with predictor inputs and predicted outputs, hidden layers of neurons, transfer functions, and training method under consideration. Datasets have been prepared for training, validation and testing purpose. During training of network model, performance and error gradient plots have been investigated for optimal results.

4.2. Results and graphs obtained

Best results achieved for each of the conjugate methods have been shown in Table 1.

Training performance for conjugate gradient methods based upon Mean Squared Error (MSE) and error gradient plots have been presented in the following figures. As shown in Figs. 1 and 2, performance and gradient plots have been observed to analyze the respective convergence and behavior of different conjugate methods to achieve the minimum of error gradient. Figs. 1(a-d) demonstrate MSE versus numbers of epochs plots during the training of network with conjugate algorithms.

Table 1: Experiment results of employing different conjugate gradient algorithms

Training Algorithm	Training function	Min. gradient	Neurons in hidden layer	Final epochs	Training time	Training performance	Starting gradient value	Final gradient value
Conjugate gradient (Polak–Ribiere update)	traincgp	1e-05	20	97	0:06:24	.0374	0.514	.01280
Conjugate gradient (Fletcher–Reeves update)	traincgf	1e-05	20	62	0:05:16	.0401	0.634	.00255
Conjugate gradient (Powell-Beale update)	traincgb	1e-05	20	39	0:02:52	.0400	0.433	.00301
Scaled conjugate gradient	trainscg	1e-05	20	517	0:24:21	.0149	.716	9.77e-06

Experiment results with MATLAB (Neural Network Toolbox) software

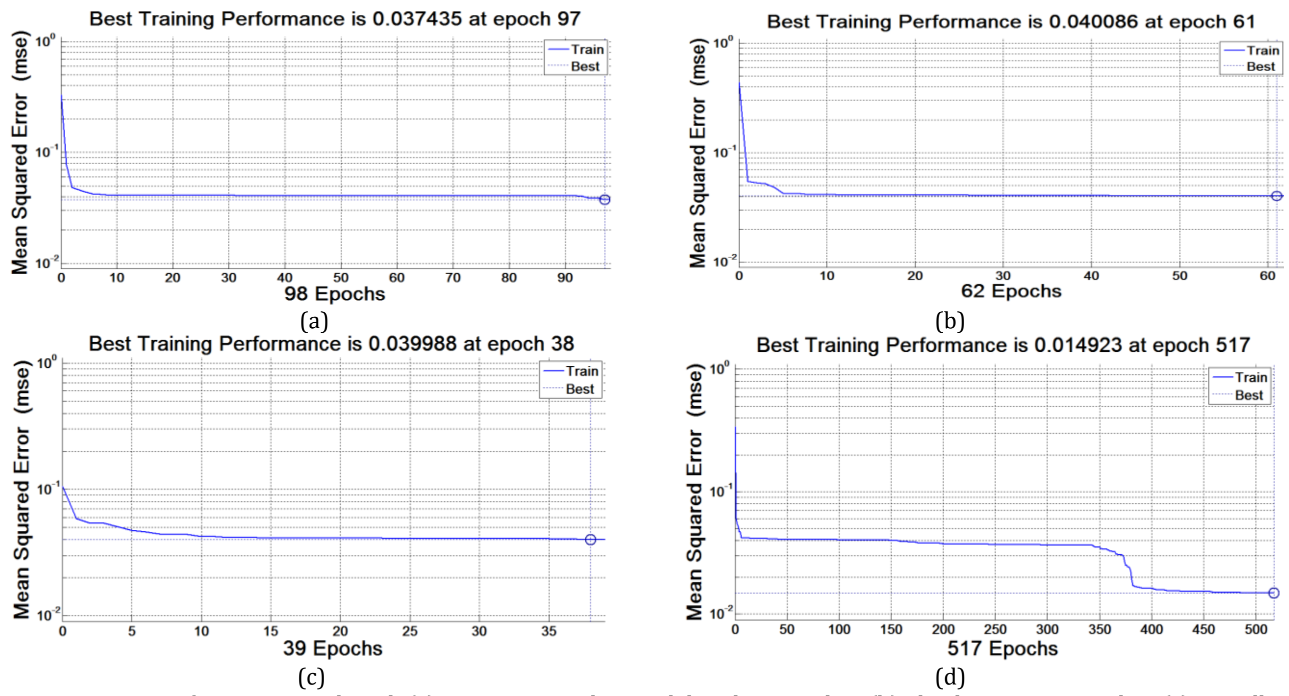

Fig. 1: Training performance graph with (a) Conjugate gradient Polak–Ribiere update (b) Fletcher –Reeves update (c) Powell-Beale update (d) Scaled conjugate gradient learning

Fig. 2(a-d) demonstrate error gradient curves for the training process. A target value of minimum error gradient '1e-05' was set for all the training algorithms. All first order methods failed to achieve the set target of minimum error, second order conjugate algorithms have partially achieved the set target and scaled conjugate gradient was able to converge completely toward the set target. As shown above in Fig. 1(d) for scaled conjugate method has taken 517 iterations to reach the target gradient.

5. Conclusion

In this research, multilayer feed forward neural network have been trained with four different variations of conjugate gradient methods and to evaluate their relative performances. The performance of Scaled Conjugate Gradient method (SCGM) comes out to be the best for the given datasets and it has converged well toward the set target value of minimum gradient. The method has shown a performance value 0.0149 and reached the gradient value of 9.77e-06. On the other hand, it has been observed that first order techniques like steepest descent and its variations are not able to achieve the set target even in 1000 epochs. It has been found that even if conjugate methods using the line search could not completely converge toward the set target of the order of 10^{-5}, but they have reached very near to the set target value. Second best performance has been shown by Fletcher–Reeves update which reached a minimum gradient of 0.00255, but Powell-Beale and Polak–Ribiere updates are also very close. Hence, the models trained with second order conjugate methods can be used effectively for the predictive data mining.

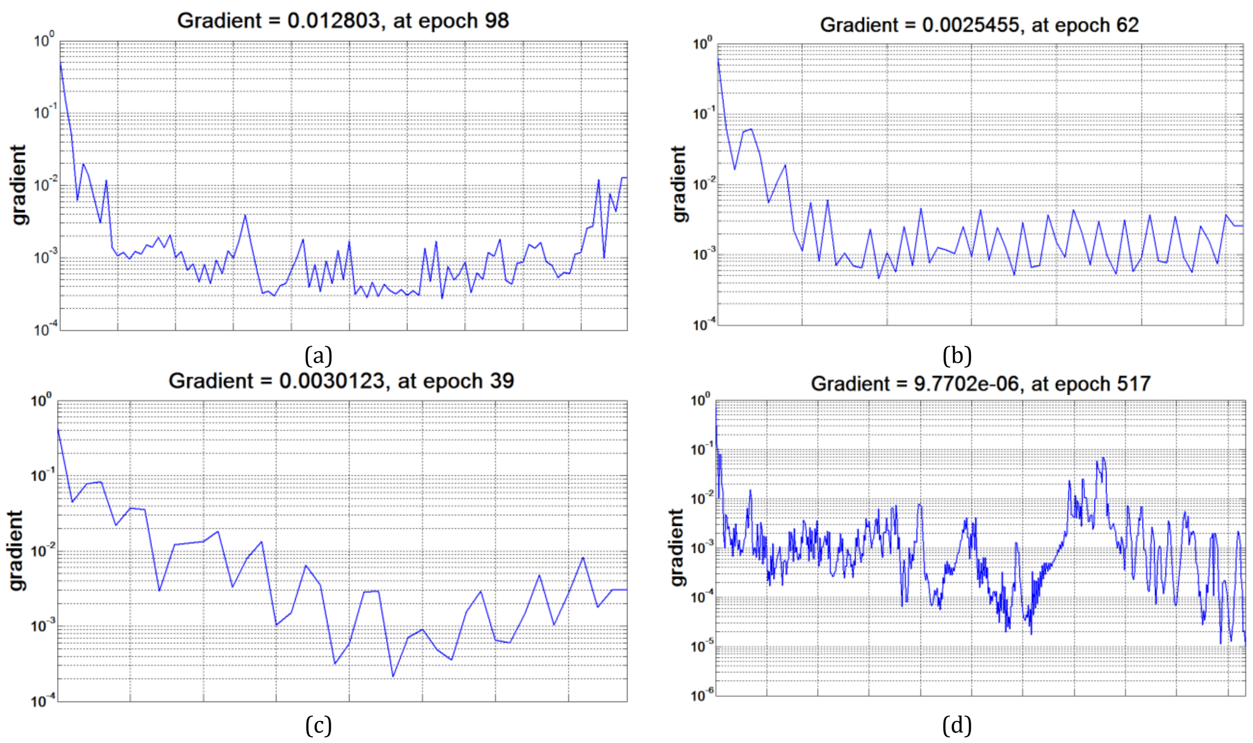

Fig. 2: Error gradient graph with (a) Conjugate gradient Polak–Ribiere update (b) Fletcher –Reeves update (c) Powell-Beale update (d) Scaled conjugate gradient learning (MathWorks, 2012)

Acknowledgement

We are extremely grateful to 'KOTAK Life Insurance' for providing the data sets from the company's live data warehouse and their kind permission to use the data for the research work. We are also thankful for facilitating to explain the meaning of metadata.

References

Andrei N (2006). Conjugate gradient algorithms for molecular formation under pairwise potential minimization. In the 5th Workshop of Mathematical modeling of environmental and life sciences problems, Constanta, Romania: 7–26. Available online at: http://www.csm.ro/home/mmelsp_series/mmelsp_08_papers/nandrei_08.pdf

Antoniou A and Lu WS (2007). Practical optimization: algorithms and engineering applications. Springer Science and Business Media, Berlin, Germany.

Borkar P, Sarode MV, and Malik LG (2016). Employing Speeded Scaled Conjugate Gradient Algorithm for Multiple Contiguous Feature Vector Frames: An Approach for Traffic Density State Estimation. In the International Conference on Information Security and Privacy (ICISP'15), Nagpur, India, 78: 740–747.

Castillo E, Guijarro-Berdiñas B, Fontenla-Romero O, and Alonso-Betanzos A (2006). A very fast learning method for neural networks based on sensitivity analysis. Journal of Machine Learning Research, 7: 1159-1182.

Chel H, Majumder A, and Nandi D (2011). Scaled conjugate gradient algorithm in neural network based approach for handwritten text recognition. In: Nagamalai D, Renault E, and Dhanuskodi M (Eds.), Trends in Computer Science, Engineering and Information Technology: 196-210. Springer Berlin Heidelberg, Heidelberg, Germany.

Fletcher R (2013). Practical methods of optimization. John Wiley and Sons, New Jersey, USA.

Hagan MT, Demuth HB, and Beale MH (1996). Neural network design. PWS Publishing Company, Boston, USA.

Hager WW and Zhang H (2006a). Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent. ACM Transactions on Mathematical Software (TOMS), 32(1): 113-137.

Hager WW and Zhang H (2006b). A survey of nonlinear conjugate gradient methods. Pacific journal of Optimization, 2(1): 35-58.

Haykin S (1994). Neural networks: A comprehensive foundation. Prentice Hall, New Delhi, India.

Keramati A, Jafari-Marandi R, Aliannejadi M, Ahmadian I, Mozaffari M, and Abbasi U (2014). Improved churn prediction in telecommunication industry using data mining techniques. Applied Soft Computing, 24: 994-1012.

MathWorks (2012). Neural network toolbox. Available online at: https://www.mathworks.com/products/neural-network.html

Meza JC (2010). Steepest descent. Wiley Interdisciplinary Reviews: Computational Statistics, 2(6): 719-722.

Møller MF (1993). A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks, 6(4): 525-533.

Nocedal J and Wright SJ (2006). Numerical optimization. Springer Science and Business Media, Berlin, Germany.

Pujol J (2007). The solution of nonlinear inverse problems and the Levenberg-Marquardt method. Geophysics, 72(4): 1-16.

Rehman MZ and Nawi NM (2012). Studying the Effect of adaptive momentum in improving the accuracy of gradient descent back propagation algorithm on classification problems. International Journal of Modern Physics: Conference Series, World Scientific Publishing Company, 9: 432-439.

Robitaille B, Marcos B, Veillette M, and Payre G (1993). Quasi-Newton methods for training neural networks. WIT

Transactions on Information and Communication Technologies, 2: 323-335

Rojas R (2013). Neural networks: A systematic introduction. Springer Science and Business Media, Berlin, Germany.

Sandhu PS and Chhabra S (2011). A comparative analysis of Conjugate Gradient algorithms and PSO based neural network approaches for reusability evaluation of procedure based software systems. Chiang Mai Journal of Science, 38(2): 123-135.

Shanthi D, Sahoo G, and Saravanan N (2009). Comparison of neural network training algorithms for the prediction of the patient's post-operative recovery area. Journal of Convergence Information Technology, 4(1): 24-32.

Shepherd AJ (2012). Second-order methods for neural networks: Fast and reliable training methods for multi-layer perceptrons. Springer Science and Business Media, Berlin, Germany.

Slavici T, Maris S, and Pirtea M (2016). Usage of artificial neural networks for optimal bankruptcy forecasting (Case study: Eastern European small manufacturing enterprises). Quality & Quantity, 50(1): 385-398.

Sundarkumar GG and Ravi V (2015). A novel hybrid undersampling method for mining unbalanced datasets in banking and insurance. Engineering Applications of Artificial Intelligence, 37(1): 368-377.

Tezel G and Buyukyildiz M (2016). Monthly evaporation forecasting using artificial neural networks and support vector machines. Theoretical and Applied Climatology, 124(1-2): 69-80.

Yu H and Wilamowski BM (2012). Neural network training with second order algorithms. In: Hippe ZS, Kulikowski JL, and Mroczek T (Eds.), Human–computer systems interaction: Backgrounds and applications 2: 463-476. Springer Berlin Heidelberg, Heidelberg, Germany.