Far East Journal of Electronics and Communications

http://www.pphmj.com
http://dx.doi.org/10.17654/0973700622002
Volume 26, 2022, Pages 1-16 **P-ISSN: 0973-7006**

TRAINING PI-SIGMA NEURAL NETWORK USING DOUBLE REGULARIZATION

Khidir Shaib Mohamed[1,2,*], Osman Abdalla Adam Osman[1], Khalid Makin[1], Mohammed Nour A. Rabih[1] and D. S. Muntasir Suhail[1]

[1]Department of Mathematics
College of Sciences and Arts in Uglat Asugour
Qassim University
Buraydah, Kingdom of Saudi Arabia
e-mail: k.idris@qu.edu.sa
o.osman@qu.edu.sa
3953@qu.edu.sa
m.fadlahhah@qu.edu.sa
m.suhail@qu.edu.sa

[2]Department of Mathematics and Computer
College of Science
Dalanj University
Dalang, South Kordofan
Sudan

Received: September 8, 2022; Accepted: October 29, 2022
Keywords and phrases: online gradient method, pi-sigma neural networks, double regularization, L2 regularization.
[*]Corresponding author

How to cite this article: Khidir Shaib Mohamed, Osman Abdalla Adam Osman, Khalid Makin, Mohammed Nour A. Rabih and D. S. Muntasir Suhail, Training pi-sigma neural network using double regularization, Far East Journal of Electronics and Communications 26 (2022), 1-16. http://dx.doi.org/10.17654/0973700622002

Published Online: December 6, 2022

Abstract

Traditional regularization parameters such as L1 and L2 are added to the cost function for neural network learning to improve learning ability and generate sparsity in the solution. L2 regularization adds the squared value of the weights to the cost function, whereas L1 regularization adds the absolute value of the weights. This study proposes an online gradient method with a novel double regularization (OGDr) for enhancing the learning ability of pi-sigma neural networks (PSNNs). The L1 and L2 regularization methods are combined in the double regularization method, which is frequently used in several machine learning frameworks. To improve the suggested method's performance learning ability, we applied the XOR problem, parity problem, Gabor function problem, and sonar benchmark challenge. The numerical examples of cases, OGL1, and OGL2 were compared. The OGDr has a good learning accuracy, according to numerical statistics. In addition, unlike OGL1 and OGL2, the error decreases monotonically, and the gradient of the error function approaches zero throughout learning.

1. Introduction

PSNNs are a form of higher-order neural network (HONN) introduced by Ghosh and Shin in [1] that is effective for pattern categorization and function approximation. Product cells are used as output in this network to incorporate the capabilities of higher-order networks implicitly while utilizing fewer weights and processing units. This network has a periodic structure, learns faster, and can incrementally add units to achieve the necessary level of complexity. Because the weights of the summing nodes to product nodes are fixed at 1, the PSNN performs better than the conventionally feedforward neural networks and even other HONNs such as sigma-pi neural networks [2], functional link neural networks [3], and sigma-pi-sigma neural networks [4]. PSNNs, in particular, have been used in a variety of domains, including nonlinear time series forecasting [5], spotted hyena optimizer [6], temperature forecasting [7], and fuzzy time series forecasting [8]. The learning process entails taking into account sample

observations in order to improve the performance of a network. During learning, the weights in the network are modified to improve the accuracy of the outcome by minimizing observed mistakes. Overfitting is a key issue that impedes the verification of the goals assigned to it throughout the learning process. Cross-validation, regularization, early halting, Bayesian priors, and dropout are some strategies for reducing overfitting. Regularization terms are routinely added into learning techniques to increase generalization performance and prevent overfitting. They can also be used to improve sparse models. Regularization penalty terms appear in neural networks in three ways: weight decay [9], weight elimination [10], and approximate smoother [11]. Weight decay is sometimes known as L2 regularization, because the penalty term is the weights' 2-norm [9, 12, 13]. Regularization terms are frequently expressed as q-norms of weights, resulting in the new error function:

$$E(W) = \tilde{E}(W) + \lambda \| W \|_q^q, \tag{1}$$

where $\tilde{E}(W)$ is a standard error function depending on the weight W of the network, λ denotes regularization parameter, and $\| W \|_q = \left(\sum_{i=1}^{n} | w_i | \right)^{\frac{1}{2}}$ is the q-norm $(0 \le q \le 2)$ of the weights of the network. A sparse optimization solution [14] is a feasible way for L0 regularization. This phrase has been utilized in a variety of contexts, such as blind natural image deblurring [15] and the accelerated iterative hard thresholding technique [16]. The well-known developed model criteria AIC and BIC [17] are particular examples of L0 regularization. L1 regularization is a well-known concept in statistics and machine learning. Reference [18] introduces a common L1 regularization, which is also regarded as a quick optimization method in [19]. L1 regularization is computationally intensive and provides the best convex approximation to L0 regularization. In [20], is proposed a novel L1/2 regularization that is sparser than the L0 and L1 regularizations. Several recent efforts to regularize L1/2 have really been made, including cancer survival analysis [21] and genetic data analysis [22].

In [23], double regularization, also known as elastic net regularization, is proposed for variable selection and is especially useful when the number of predictors exceeds the number of observations. L2 regularization is added to L1 regularization in this phrase. Support vector machines [24], portfolio optimization [25], and cancer diagnosis [26] are all applications of double regularization. In this study, we look at the double regularization term and concentrate on the batch gradient method proposed by

$$E(W) = \tilde{E}(W) + \lambda_1 \| W \|^2 + \lambda_2 \| W \|_1, \tag{2}$$

where λ_1, λ_2 are regularization parameters, and $\| \cdot \|^2$ and $\| \cdot \|_1$ denote the 2-norm and 1-norm, respectively.

This paper offers and investigates an online gradient approach for training PSNN with double regularization. Using double regulation, we confirmed that no earlier research on pi-sigma neural networks had been conducted. We strive to obtain performance capabilities in this work without employing theoretical analysis. We can achieve the following benefits as a result of the proposed strategy and the major contribution of this paper:

- Our numerical simulations show that the double regularization approach may learn as well as compare to the classic and regularization methods.

- It is demonstrated that how double regularization distinguishes between important and unneeded weights and forces extraneous weights to zero, ultimately pruning the network.

- OGDr is more accurate and performs better, as evidenced by numerical findings.

2. Neural Network Structure and Algorithm Description

In this section, we describe the pi-sigma neural network structure considered in this paper, followed by an online gradient method with double regularization.

2.1. Pi-sigma neural network

Consider a PSNN, as shown in Figure 1. The number of the nodes in the input unit, summation unit and output unit are p, n and 1, respectively. Taking $\omega_j = (\omega_{j1}, \omega_{j2}, ..., \omega_{jp})^T \in \mathbb{R}^p$ and $0 \le j \le n$, the weight vector collection of the input nodes is the summation node, where $\omega = (\omega_1^T, \omega_2^T, ..., \omega_n^T) \in \mathbb{R}^{np}$ and $(1 \le i \le p)$. This network notes that the weights of the summing nodes to product nodes are fixed to be one and the activation function is given by $f : \mathbb{R} \to \mathbb{R}$. In particular, for an input vector $x \in \mathbb{R}^p$, the final output of the network is

$$y = f\left[\sum_{j=1}^{n}\left(\prod_{i=1}^{p}\omega_{ji}x_i\right)\right] = f\left(\prod_{j=1}^{n}(\omega_j \cdot x)\right). \tag{3}$$

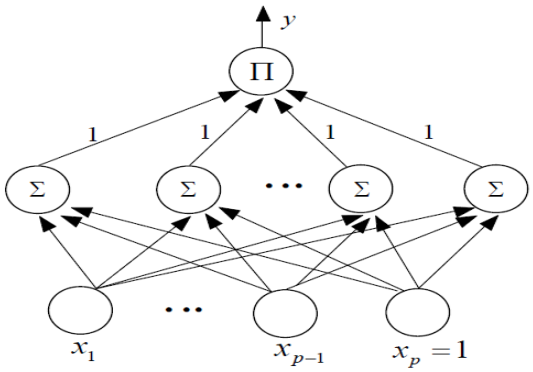

Figure 1. Pi-sigma neural network structure.

2.2. Modified error function with L2 regularization

Let $\{x^l, O^l\}_{l=1}^L \subset \mathbb{R}^p \times \mathbb{R}$ be the set of training samples with O^l as the desired objective output for the input x^l. Then the modified error function is defined as follows by L2 regularization term:

$$E(\omega) = \frac{1}{2}\sum_{l=1}^{L}\left[O^l - f\left(\prod_{j=1}^{n}(\omega_j \cdot x)\right)\right]^2 + \lambda\|\omega\|^2, \tag{4}$$

where $\| \omega \|^2 = \sum_{j=1}^{n} \omega_j^2$, and $\| \cdot \|$ is the usual Euclidean norm. The gradient of the above error function is given by

$$E_{\omega_j}(\omega) = \sum_{l=1}^{L} \delta_l \left(\prod_{j=1}^{n} (\omega_j \cdot x^l) \right) \prod_{\substack{k=1 \\ k \neq j}}^{n} (\omega_k \cdot x^l) x^l + 2\lambda_1 \omega_j, \tag{5}$$

where $\lambda > 0$ is the regularization parameter, $\delta_l(t) = -[O^l - f(t)] f'(t)$, and $E_{\omega_j}(\omega) = \partial E(\omega) / \partial \omega_j$.

Starting from an initial value ω^0, the weights $\{\omega^m\}$ are updated iteratively by

$$\omega_j^{mL+l} = \omega_j^{mL+l-1} + \Delta_j \omega_j^{mL+l-1}, \tag{6}$$

where $j = 1, 2, ..., n$; $k = 1, 2, ..., n$; $l = 1, 2, ..., L$; $m = 0, 1, ...,$ and

$$\Delta_s \omega_j^m = -\eta_m \left[\delta_s \left(\prod_{j=1}^{n} (\omega_j^m \cdot x^s) \right) \prod_{\substack{k=1 \\ k \neq j}}^{n} (\omega_k^m \cdot x^s) x^s + 2\lambda_1 \omega_j^m \right],$$

where $\eta_m > 0$ is the learning.

2.3. Modified error function with double regularization

Let $\{x^l, O^l\}_{l=1}^{L} \subset \mathbb{R}^p \times \mathbb{R}$ be the set of training samples with O^l as the desired objective output for the input x^l. The modified error function is defined as follows by double regularization term:

$$E(\omega) = \frac{1}{2} \sum_{l=1}^{L} \left[O^l - f \left(\prod_{j=1}^{n} (\omega_j \cdot x) \right) \right]^2 + \lambda_1 \| \omega \|^2 + \lambda_2 \| \omega \|_1, \tag{7}$$

where $\| \omega \|_1 = \sum_{j=1}^{n} | \omega_j |$, and $| \cdot |$ denotes the absolute value. The gradient

of the error function is given by

$$E_{\omega_j}(\omega) = \sum_{l=1}^{L} \delta_l \left(\prod_{j=1}^{n} (\omega_j \cdot x^l) \right) \prod_{\substack{k=1 \\ k \neq j}}^{n} (\omega_k \cdot x^l) x^l + 2\lambda_1 \omega_j + \lambda_2 \operatorname{sign}(\omega_j). \quad (8)$$

Starting from an initial value ω^0, the weights $\{\omega^m\}$ are updated iteratively by

$$\omega_j^{mL+l} = \omega_j^{mL+l-1} + \Delta_m \omega_j^{mL+l-1}, \quad (9)$$

where $j = 1, 2, ..., n$; $k = 1, 2, ..., n$; $l = 1, 2, ..., L$; $m = 0, 1, ...,$ and

$$\Delta_s \omega_j^m = -\eta_m \left[\delta_s \left[\prod_{j=1}^{n} (\omega_j^m \cdot x^s) \right] \prod_{\substack{k=1 \\ k \neq j}}^{n} (\omega_k^m \cdot x^s) x^s + 2\lambda_1 \omega_j^m + \lambda_2 \operatorname{sign}(\omega_j^m) \right],$$

where $\eta_m > 0$ is the learning, λ_1, λ_2 are regularization parameters, and the "sign" is signum function of a real number z that can be defined as follows:

$$\operatorname{sign}(x) = \begin{cases} -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ 1 & \text{if } x > 0. \end{cases}$$

3. Numerical Simulations

In order to test the effectiveness of the proposed method (OGDr), we compare its performance with L1 regularization method (OGL1) and with L2 regularization method (OGL2). We use $f(t) = 1/(1 + e^{-t})$ as the activation function with the initial weights randomly selected from $[-0.5, 0.5]$. Comparative results are obtained using the XOR problem, parity problem, Gabor function problem and sonar benchmark problem.

3.1. XOR and parity problems

In this subsection, the XOR problem and a type of parity problem are simulated. Two input units, two hidden units, and one output unit are used for the XOR problem, with learning rate $\eta = 0.07$, and $\lambda_1 = 0.002$ and $\lambda_2 = 0.003$ as the regularization parameters. In the parity problem, four input layers, five summation layers, and one output layer are considered with a learning rate $\eta = 0.042$, and for both the problems, the regularization parameters are $\lambda_1 = 0.0002$ and $\lambda_2 = 0.0003$. For both examples, the maximum iteration number is 1800. The error function, norm of gradient, and the average number of neurons are eliminated (ANE in brief). We performed 10 repetitions each for the XOR and parity problems to obtain average errors and gradient norms.

Through the results obtained from Figures 2 to 5, we see that OGDr has a good mapping capability than OGL1 and OGL2. The error of OGDr decreases monotonically as learning proceeds and its gradient goes to zero. Tables 1 and 2 show the average errors, average gradient norms, and average ANE in brief for three learning methods over the 10 trials.

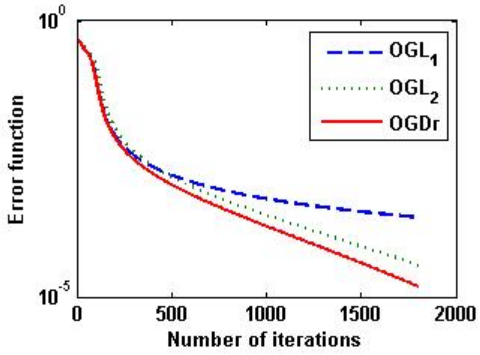

Figure 2. Performance of error function with different algorithms for XOR problem.

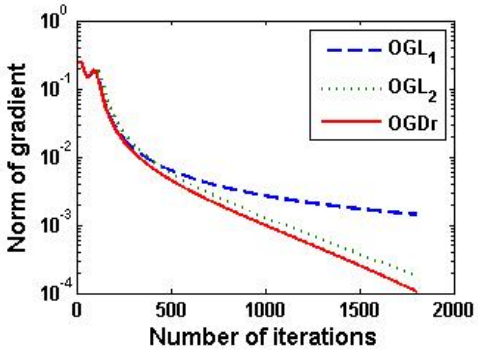

Figure 3. Performance of norm of gradient with different algorithms for XOR problem.

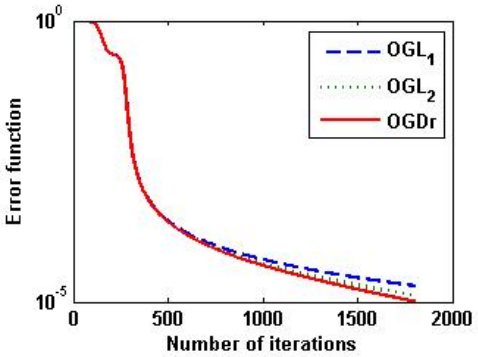

Figure 4. Performance of error function with different algorithms for parity problem.

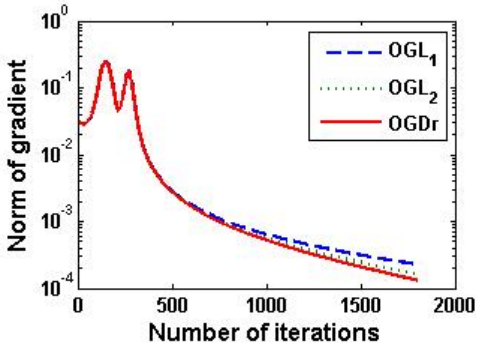

Figure 5. Performance of norm of gradient with different algorithms for parity problem.

Table 1. Comparison average errors of training and norm of gradients for XOR problem

Algorithms	Average training error	Norm of gradient	Training time (s)	ANE
OGL1	2.7371e-04	0.0014	5.330512	3.91
OGL2	3.7026e-05	1.7576e-04	5.311793	4.46
OGDr	1.5360e-05	1.0595e-04	5.215273	5.90

Table 2. Comparison average errors of training and norm of gradients for parity problem

Algorithms	Average training error	Norm of gradient	Training time (s)	ANE
OGL1	1.9265e-05	2.2655e-04	5.561021	5.33
OGL2	1.2907e-05	1.6240e-04	5.557142	7.18
OGDr	1.0160e-05	1.3122e-04	5.535433	10.00

3.2. Function approximation problem

Here, we consider using a neural network to approximate the following Gabor function in two dimensions (see Figure 6) which has the following form:

$$h(x,\ y) = \frac{1}{2\pi(0.5)} \exp\left(\frac{x^2 + y^2}{2(0.5)^2}\right) \cos(2\pi(x + y)).$$

There are 36 input training data sets chosen at random from an evenly spaced 6×6 enable on $-0.5x \le 0.5$ and $-0.5 \le y \le 0.5$. Likewise, the input test samples are 484 points picked from the 22×22 enable on $-0.5x \le 0.5$ and $-0.5 \le y \le 0.5$. The PSNN structure has three input units, five summation units, and one output unit, where $\eta = 0.9$ is the learning rate, $\lambda_1 = 0.0002$ and $\lambda_2 = 0.0003$ are the regularization parameters. The weights update ending up with 10,000 is the maximum number of training epochs.

As shown in Table 3, the OGDr has better generalization ability than the OGL1 and OGL2 algorithms on average, when taking training and testing

errors and ANE in a brief over ten trials. Comparing the results as in Figures 7 to 9, we can see that the OGDr learning curves show that the deterministic result function imitates that OGDr is doing well and approximate performance is better than the OGL1 and OGL2.

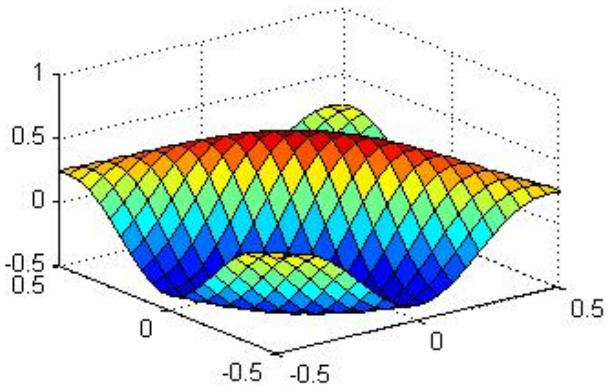

Figure 6. Gabor function.

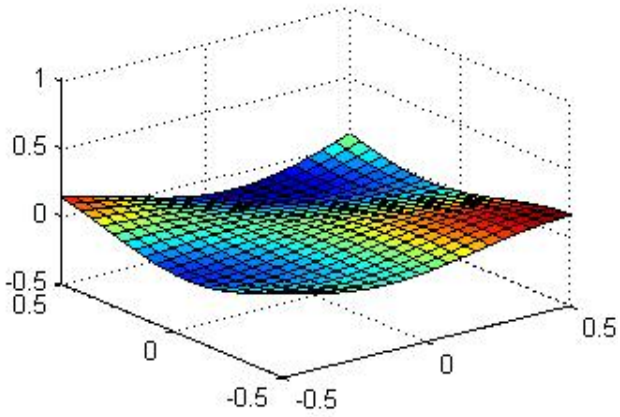

Figure 7. Approximate performance of OGL1.

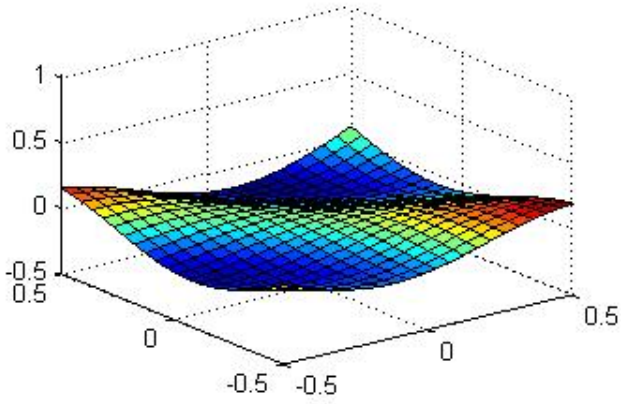

Figure 8. Approximate performance of OGL2.

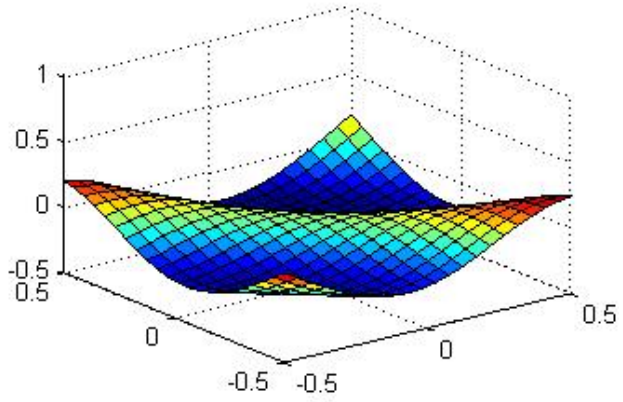

Figure 9. Approximate performance of OGDr.

Table 3. Comparison average errors training/testing for Gabor function

Algorithms	Average training error	Average test error	Training time (s)	ANE
OGL1	0.0610	0.0850	5.700147	6.23
OGL2	0.0530	0.0751	5.688019	7.08
OGDr	0.0409	0.0614	5.683113	8.57

3.3. Sonar benchmark problem

An example of a well-known classification problem is the sonar benchmark. The task is to classify reflected sonar signals into two groups (metal cylinders and rocks). The related data set comprises of 208 input vectors, each with 60 components. There are 60 input layers, 6 summation units, and one output unit. The training constants consist of learning rate $\eta = 0.6$, regularization parameters $\lambda_1 = 0.0001$ and $\lambda_2 = 0.0005$, and the weights update ending up with 5500 to be the maximum number of training epochs. Number of trails performed is 10.

Table 4. Comparison average accuracy training/testing for sonar problem

Algorithms	Accuracy training	Accuracy testing	Training time (s)
OGL1	0.8974	0.8665	15.441670
OGL2	0.9231	0.9103	15.314041
OGDr	0.9759	0.9533	15.011939

In Table 4, the average accuracy was obtained by performing 10 trails. It is seen that OGDr gets the best accuracy than OGL1 and OGL2. This comparison confirms that our new method enhances PSNN training performance.

4. Conclusions

We examine a regular online gradient approach with double regularization for PSNN (OGDr). To improve performance, double regularization is a term proportionate to the magnitude of the weights. It is critical to select the optimal learning rate and regularization parameters to aid in the algorithm's convergence. We show that the OGDr has strong generalization ability and the best accuracy of OGL1 and OGL2 under moderate settings. In the future, we hope to develop a mechanism to demonstrate the theoretical analysis of this study. Because double regularization is not differentiable at the origin, it contains the absolute value.

References

[1] Y. Shin and J. Ghosh, The pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation, IJCNN-91-Seattle International Joint Conference on Neural Networks, IEEE, Vol. 1, 1991, pp. 13-18.

[2] M. Heywood and P. Noakes, A framework for improved training of sigma-Pi networks, IEEE Transactions on Neural Networks 6(4) (1995), 893-903.

[3] S. Dehuri and S. B. Cho, Evolutionarily optimized features in functional link neural network for classification, Expert Systems with Applications 37(6) (2010), 4379-4391. https://doi.org/10.1016/j.eswa.2009.11.090.

[4] C. K. Li, A sigma-pi-sigma neural network (SPSNN), Neural Processing Letters 17(1) (2003), 1-19. https://doi.org/10.1023/A:1022967523886.

[5] E. Akdeniz, E. Egrioglu, E. Bas and U. Yolcu, An ARMA type pi-sigma artificial neural network for nonlinear time series forecasting, Journal of Artificial Intelligence and Soft Computing Research 8 (2018), 121-132. https://doi.org/10.1515/jaiscr-2018-0009.

[6] N. Panda and S. K. Majhi, Improved spotted hyena optimizer with space transformational search for training pi-sigma higher order neural network, Computational Intelligence 36(1) (2020), 320-350. https://doi.org/10.1111/coin.12272.

[7] N. A. Husaini, R. Ghazali, N. M. Nawi and L. H. Ismail, Pi-sigma neural network for temperature forecasting in Batu Pahat, International Conference on Software Engineering and Computer Systems, Springer, Berlin, Heidelberg, 2011. https://doi.org/10.1007/978-3-642-22191-0_46.

[8] E. Bas, E. Egrioglu and E. Kolemen, A novel intuitionistic fuzzy time series method based on bootstrapped combined pi-sigma artificial neural network, Engineering Applications of Artificial Intelligence 114 (2022), 105030. https://doi.org/10.1016/j.engappai.2022.105030.

[9] A. Krogh and J. Hertz, A simple weight decay can improve generalization, Advances in Neural Information Processing Systems, 1991, pp. 950-957.

[10] A. S. Weigend, D. E. Rumelhart and B. A. Huberman, Generalization by weight-elimination applied to currency exchange rate prediction, [Proceedings] 1991 IEEE International Joint Conference on Neural Networks, IEEE, 1991, pp. 2374-2379. doi: 10.1109/IJCNN.1991.155287.

[11] J. Moody and T. Rögnvaldsson, Smoothing regularizers for projective basis function networks, Advances in Neural Information Processing Systems, 1996, pp. 585-591.

[12] E. Phaisangittisagul, An analysis of the regularization between L2 and dropout in single hidden layer neural network, 2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS), IEEE, 2016, pp. 174-179. doi: 10.1109/ISMS.2016.14.

[13] Y. Liu, D. Yang and C. Zhang, Relaxed conditions for convergence analysis of online back-propagation algorithm with L2 regularizer for sigma-Pi-sigma neural network, Neurocomputing 272 (2018), 163-169. https://doi.org/10.1016/j.neucom.2017.06.057.

[14] Z. Wei, Q. Li, J. Wei and W. Bian, Neural network for a class of sparse optimization with L0-regularization, Neural Networks 151 (2022), 211-221. https://doi.org/10.1016/j.neunet.2022.03.033.

[15] Y. Zhang, Y. Shi, L. Ma, J. Wu, L. Wang and H. Hong, Blind natural image deblurring with edge preservation based on L0-regularized gradient prior, Optik 225 (2021), 165735. https://doi.org/10.1016/j.ijleo.2020.165735.

[16] F. Wu and W. Bian, Accelerated iterative hard thresholding algorithm for $$ l_0 $$ l0 regularized regression problem, J. Global Optim. 76(4) (2021), 819-840. https://doi.org/10.1007/s10898-019-00826-6.

[17] J. Kuha, AIC and BIC: comparisons of assumptions and performance, Sociol. Methods Res. 33(2) (2004), 188-229. https://doi.org/10.1177/0049124103262065.

[18] S. Mc Loone and G. Irwin, Improving neural network training solutions using regularization, Neurocomputing 37(1-4) (2001), 71-90. https://doi.org/10.1016/S0925-2312(00)00314-3.

[19] M. Schmidt, G. Fung and R. Rosales, Fast optimization methods for l1 regularization: a comparative study and two new approaches, European Conference on Machine Learning, Springer, Berlin, Heidelberg, 2007. https://doi.org/10.1007/978-3-540-74958-5_28.

[20] Z. Xu, H. Zhang, Y. Wang, X. Chang and Y. Liang, L1/2 regularization, Science China Information Sciences 53(6) (2010), 1159-1169. https://doi.org/10.1007/s11432-010-0090-0.

[21] Y. Liang, H. Chai, X. Y. Liu, Z. B. Xu, H. Zhang and K. S. Leung, Cancer survival analysis using semi-supervised learning method based on Cox and AFT models with L1/2 regularization, BMC Medical Genomics 9(1) (2016), 1-11. https://doi.org/10.1186/s12920-016-0169-6.

[22] H. K. Jiang and Y. Liang, The L1/2 regularization network Cox model for analysis of genomic data, Computers in Biology and Medicine 100 (2018), 203-208. https://doi.org/10.1016/j.compbiomed.2018.07.009.

[23] H. Zou and T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc. Ser. B Stat. Methodol. 67(2) (2005), 301-320. https://www.jstor.org/stable/3647580.

[24] L. Wang, J. Zhu and H. Zou, The doubly regularized support vector machine, Statist. Sinica 16 (2006), 589-615. https://www.jstor.org/stable/24307560.

[25] W. Shen, J. Wang and S. Ma, Doubly regularized portfolio with risk minimization, Twenty-eighth AAAI Conference on Artificial Intelligence, 2014. https://doi.org/10.1609/aaai.v28i1.8906.

[26] P. Milanez-Almeida, A. J. Martins, R. N. Germain and J. S. Tsang, Cancer prognosis with shallow tumor RNA sequencing, Nature Medicine 26(2) (2020), 188-192. https://doi.org/10.1038/s41591-019-0729-3.