PERCENTAGE ARTIFICIAL NEURAL NETWORK FOR FORECASTING

Erol Egrioglu – Eren Bas

Abstract

In this study, first, a "percentile neuron model" will be defined; subsequently, a hybrid recurrent neural network architecture employing both traditional additive neurons and the proposed percentile neurons will be presented. This recurrent structure, formed by integrating a simple exponential smoothing mechanism into the network architecture, aims to produce more robust and stable forecasts compared to traditional models, especially in time series containing outliers or exhibiting skewed distributions. A training algorithm based on Particle Swarm Optimization (PSO) will be proposed for the new neural network, and the normalization process will also be carried out in a robust form based on percentiles. The performance of the proposed new neural network is investigated using stock market time series from different countries. The results obtained for the stock market time series is compared using the Friedman test.

Key words: forecasting, artificial neural networks, particle swarm optimization, robust learning algorithm

JEL Code: C45, C53

Introduction

In recent years, artificial neural networks data-driven methods have begun to replace linear statistical models used for forecasting. It has been observed that artificial neural networks can produce models capable of better adapting to data through changes in their architectural structure and weights, thereby improving forecasting performance. The most commonly used artificial neural network in the literature, the multilayer perceptron, was introduced in the study by Rumelhart et al. (1986). The multilayer perceptron (MLP) is a multi-layer artificial neural network formed by combining additive neuron models known as perceptron. In the multilayer perceptron, final prediction models are derived from different linear combinations of the explanatory variables in the model and the nonlinear transformations applied to these

combinations. The literature has presented various neuron models, as well as new artificial neural network architectures and models based on these neuron models. Prediction using an artificial neural network based on multiplicative neuron models was addressed in the study by Yadav et al. (2007). In this study, successful prediction results were obtained using a single-neuron model artificial neural network (SMN). An architectural structure that combines both multiplicative and additive neuron models was proposed by Shin and Ghos (1991) under the name "Pi-Sigma artificial neural network." The Pi-Sigma artificial neural network features a hybrid architecture that utilizes both additive and multiplicative combination operations in neurons. Another architectural structure that utilizes both multiplicative and additive neurons is the dendritic neuron model (DNM) artificial neural network, which was proposed by Todo et al. (2014). Although SMN, Pi-Sigma, and DNM artificial neural networks may demonstrate superior performance compared to MLP, their absence from widely used software packages has reduced their frequency of use and led to their being employed in fewer studies in the literature.

An examination of MLP, SMN, Pi-Sigma, and DNM artificial neural networks reveals that they often have limitations, such as being trained with the logistic activation function, being affected by outliers in the training data, and using addition and multiplication operations to combine signals. Although efforts have been made in recent years to address these limitations, the number of such studies remains limited. In recent years, various modifications of shallow artificial neural networks have been proposed. In their study, Egrioglu et al. (2022) proposed a recurrent version of the DNM-ANN. In their study, Egrioglu et al. (2023) made the DNM-ANN more robust by using a winsorized mean. In the study by Egrioglu and Bas (2023a), a new shallow recurrent artificial neural network based on exponential smoothing logic was introduced. In their study, Egrioglu and Bas (2023b) proposed a modified version of the Pi-Sigma artificial neural network.

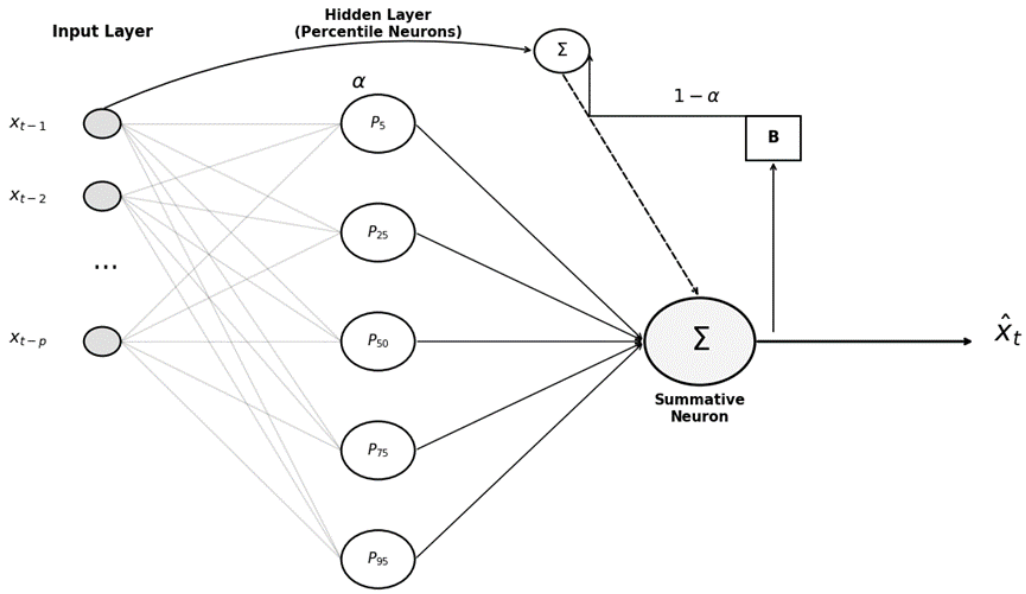
To address the current limitations, this study presents an artificial neural network that operates using a completely different neuron model architecture. The proposed artificial neural network employs neuron models that utilize the percentages suggested for the first time in this study. For this reason, the proposed artificial neural network has been named the Percentage Artificial Neural Network (PANN). In PANN, while considering the percentages of input signals helps reduce the impact of outliers in the signals, the Cauchy activation function was used instead of the logistic function as the activation function. This activation function is specifically designed to eliminate the impact of outliers. Particle swarm optimization was used in the training of PANN, and the decile mean of the squared residuals was used as the fitness function to eliminate the effect of outliers in the target values. In addition, a percentage-based

normalization method was chosen for the data normalization process. In addition to all of this, a feedback structure inspired by the simple exponential smoothing method has been incorporated into the architecture to enhance the performance of the artificial neural network. In this respect, the proposed PANN is a feedback-based hybrid network architecture. The first section of the study presents the PANN components, its architectural structure, forward and backward calculation formulas, and the training algorithm. The second chapter presents the results of the study, and the final chapter discusses the findings.

1 Percentage Artificial Neural Network

PANN is an artificial neural network that uses spiking neural models. It has an architectural structure that includes an input layer, a hidden layer, an output layer, and a feedback connection. The architectural structure of PANN is shown in Figure 1.

Fig. 1: The PANN Architecture



Source: own computation

The percentage parameters used for the neurons in the hidden layer are 5, 25, 50, 75, and 95. Thus, the distributional characteristics of the incoming weighted signals (the left and

right extremes of the distribution at the 5th and 95th percentiles, the quartiles i.e., the measures of variability at the 25th and 75th percentiles, and the centre of the distribution at the 50th percentile) form the outputs of the hidden layer neurons. This allows for the extraction of more meaningful information from the incoming signals, going beyond simply processing different linear combinations of those signals. Additionally, the fact that the number of neurons is fixed at five simplifies the hyperparameter selection process. The Cauchy activation function given in Equation (1) was used to generate the hidden layer outputs of the PANN.

$$f(x) = \frac{1}{\pi}\arctan(x) + \frac{1}{2} \qquad (1)$$

This activation function has a heavy-tailed distribution, making it more suitable for modelling outliers. Another key feature of PANN is that its architectural structure incorporates a feedback mechanism. The feedback is the weighted average of the variable's value one step behind and the network's previous output. This structure was added to the architecture, inspired by the simple exponential smoothing method. The aim of this addition is to improve the network's performance.

The equations provided below are used to calculate PANN's outputs. The incoming signals to the hidden neurons are computed by using Equation (2).

$$net_j = Percintile\left(\left\{w_{1j}x_{t-1}, w_{2j}x_{t-2}, \dots, w_{pj}x_{t-p}, b_j\right\}, P_j\right), j = 1,2,3,4,5 \qquad (2)$$

Where $P_1 = 5, P_2 = 25, P_3 = 50, P_4 = 75, P_5 = 95$ and $Percintile(A, P)$ shows P^{th} quantile of A data. The outputs of the hidden neurons are calculated by using Equation (3).

$$h_j = \frac{1}{\pi}\arctan\left(net_j\right) + \frac{1}{2}\ , j = 1,2,3,4,5 \qquad (3)$$

The incoming signal to the output neuron is calculated by using Equation (4).

$$net_{output} = \sum_{j=1}^{5} v_j + b_o \qquad (4)$$

The output of the network is calculated by using Equation (5).

$$\widehat{x_t} = \alpha\left(\frac{1}{\pi}\arctan\left(net_{output}\right) + \frac{1}{2}\right) + (1-\alpha)x_{t-1} \qquad (5)$$

In conclusion, the parameters that need to be estimated for the network are connection weights between input and hidden units $\{w_{ij}, i = 1,2,\dots,p,\ j = 1,2,3,4,5\}$, biases of hidden units $\{b_j, j = 1,2,3,4,5\}$, connection weights between hidden units and output unit $\{v_j\ j = 1,2,3,4,5\}$, bias of the output unit b_0 and smoothing parameter α. The total number of the parameters that need to be estimated for the network is $5p + 12$.

The training of the proposed method is performed by estimating the parameters provided above. Particle swarm optimization was used as the training algorithm, and the fitness function is given below. The positions of the particles are parameters that need to be estimated for the network.

$$fitness = DM(\{e_t^2, t = 1,2,\dots,n\} \qquad (6)$$

Where DM is decile mean and it is calculated by using Equation (7).

$$DM(X) = \frac{\sum_{i=1}^{9} Percintile(X, i\times 10)}{9} \qquad (7)$$

DM is a robust central tendency measure and it is not badly affected by outliers. Because of the using DM as a fitness function, the outliers in the target are not badly affect the training performance.

2 Applications

The performance of the proposed new artificial neural network was evaluated using time series of S&P 500 closing prices. Two time series were obtained from the S&P 500 closing prices. The first series is a daily time series covering the period from February 20, 2024, to February 14, 2025, while the second series is a daily time series covering the period from August 5, 2024, to July 31, 2025. While both time series exhibit a trend, the second time series shows a consecutive decline in closing prices during a specific period.

In the analysis of time series, the SMN, DNM, MLP, PS, GRU, LSTM, and proposed PANN methods were used. To ensure fairness, all methods were applied using the same analytical approach. In the implementation, the time series data was divided into three blocks: training, validation, and test data. The validation and test sets each contain the same number of 20-day observations. Artificial neural networks were trained using the training data, and the validation set performance was used to select hyperparameters. The hyperparameters are the number of lagged variables (p), the number of units in the hidden layer (m), and the number of

hidden layers (h). Not all methods have all hyperparameters. For example, in the proposed method, only the number of delayed variables is a hyperparameter for PANN and SMN, whereas for LSTM, all three hyperparameters are determined based on validation set performance. After hyperparameter selection, the training and validation sets are combined to obtain a new training set. The architecture, determined based on optimal hyperparameters, is retrained using this expanded training set with 30 different random initial weights, and the mean, standard deviation, minimum, and maximum values of the root mean square error (RMSE) obtained for the test set are reported. The methods are compared based on these statistics. In addition to the original forms of both time series, contaminated time series were created by injecting three outliers into the training sets. Analyses were also performed on the contaminated series.

Table 1 summarize the results obtained for the original first series. As can be seen, PANN has the lowest average RMSE value. Furthermore, both the best and worst results are lower than those obtained with other methods. The method's results exhibit very low variability. Table 2 present the results for the contaminated instances of the first dataset. The proposed PANN method shows a clear advantage over other methods. The results appear to be quite close to those obtained for the original dataset.

Table 3 present the results for the original version of the second series. It can be seen that the proposed PANN method is the third-best method, yielding the lowest result after MLP and SMN based on the average RMSE value. In this series, the methods produce results that are close to one another and competitive. PANN's standard deviation is at reasonable levels, and it ranks in the middle among the other methods in terms of both the best and worst results. Table 4 present the results for the contaminated version of the second series. It can be seen that the results obtained using the proposed PANN method are quite similar to those obtained for the original series. It is to be expected that PANN would yield successful prediction results for data containing outliers compared to other methods.

Table 1. Results obtained in the application of the methods for in Original Series 1

Methods	Best Config (p) or (p,m) or (p,m,h)	Mean RMSE	Std RMSE	Min RMSE	Max RMSE
SMN	(1)	52,24372	0,0046	52,2355	52,2549
DNM	(1, 1)	56,2348	3,5363	51,8803	62,6646
MLP	(1, 3)	53,8718	4,2640	52,1432	70,5638

PS	(1, 1)	52,2439	0,0006	52,2425	52,2461
GRU	(3, 3, 1)	56,3085	2,4043	51,2978	64,2597
LSTM	(3, 3, 1)	55,8830	1,7229	52,4300	59,9031
PANN	(1)	50,1610	0,0820	49,9537	50,2920

Source: own computation

Table 2. Results obtained in the application of the methods for in Contaminated Series 1

Methods	Best Config (p) or (p,m) or (p,m,h)	Mean RMSE	Std RMSE	Min RMSE	Max RMSE
SMN	(1)	212,7345	6,8289	189,7260	222,1200
DNM	(2, 2)	704,9432	305,5321	61,0105	1213,4080
MLP	(3, 3)	708,1282	337,7178	116,7177	1403,9820
PS	(3, 1)	176,6134	119,5051	69,8844	486,08020
GRU	(3, 1, 1)	441,5320	306,3290	65,5365	1276,0760
LSTM	(3, 1, 1)	361,1384	315,4223	60,6057	1483,5980
PANN	(3)	51,68793	0,0525	51,4723	51,7651

Source: own computation

Table 3. Results obtained in the application of the methods for in Original Series 2

Methods	Best Config (p) or (p,m) or (p,m,h)	Mean RMSE	Std RMSE	Min RMSE	Max RMSE
SMN	(2)	36,8103	0,1289	36,4637	37,1459
DNM	(2, 3)	37,8483	0,6505	36,1266	39,4199
MLP	(3, 3)	35,7889	1,1863	33,4487	38,7298
PS	(2, 1)	38,3011	0,0550	38,2050	38,4874
GRU	(2, 1, 1)	39,7969	1,0866	34,8939	40,7302
LSTM	(1, 2, 1)	39,7708	2,1201	34,7262	41,9782
PANN	(1)	36,9048	0,4437	36,5292	38,2274

Source: own computation

Table 4. Results obtained in the application of the methods for in Contaminated Series 2

Methods	Best Config (p) or (p,m) or (p,m,h)	Mean RMSE	Std RMSE	Min RMSE	Max RMSE

SMN	(1)	213,1286	11,9738	163,6875	244,0293
DNM	(2, 3)	888,0788	381,8642	185,6516	1562,3130
MLP	(2, 2)	770,6942	378,6966	53,0859	1823,8610
PS	(2, 1)	128,5921	44,8328	54,2236	284,3853
GRU	(3, 1, 1)	344,2371	254,5761	56,2263	992,7193
LSTM	(2, 1, 1)	602,3268	382,3887	52,6010	1437,5430
PANN	(1)	36,1684	0,1659	35,9201	36,6351

Source: own computation

Analyses conducted using two different time series of S&P 500 closing prices clearly demonstrate the contribution of the PANN model to the literature: Original Series 1: The PANN method outperformed all alternative models by achieving the lowest average RMSE value of 50.1610. Furthermore, both the best (Min RMSE: 49.9537) and worst (Max RMSE: 50.2920) results produced by the model outperform the outlier values of all other methods. Box-plot analyses also confirm that the variance (variability) in the PANN model's results is extremely low. Contaminated (Outlier-Added) Series 1: When three artificial outliers were added to the series, dramatic increases in error rates (RMSE) were observed in alternative models (MLP, LSTM, GRU, etc.). For example, the average RMSE value for the MLP jumped from 53.8 to 708.1. In contrast, the PANN model was virtually unaffected by the outliers, maintaining its stable structure with an average RMSE value of 51.6879. Original Series 2: In this series, the PANN method achieved the third-best performance with an average RMSE of 36.9048, following the MLP and SMN models, with all models producing competitive results. Contaminated Series 2: When the second dataset was contaminated, the PANN model replicated its success from the original series with an RMSE of 36.1684 and achieved a decisive advantage over all other methods.

Conclusion

In this study, the performance of the Percentage Artificial Neural Network (PANN) model proposed to overcome the existing limitations of artificial neural networks in the time series forecasting literature (particularly sensitivity to outliers and traditional linear/multiplicative combination functions) was empirically evaluated. The model's performance was compared

with strong alternatives widely used in the forecasting of financial time series, such as MLP, SMN, DNM, Pi-Sigma (PS), GRU, and LSTM.

The PANN model's high and consistent performance both in the original dataset and particularly in datasets containing noise or outliers can be attributed to three fundamental innovative approaches in its architecture: Percentile Neuron Structure: While traditional models process signals using linear combinations, PANN captures the distribution characteristics of signals by using a fixed set of 5 neurons in the hidden layer. This limits the influence of outliers (the 5th and 95th percentiles) and models the central tendency and variability of the data more meaningfully. It also simplifies the hyperparameter selection process. Using the heavy-tailed Cauchy activation function instead of the logistic function in the network's hidden layers dampens the shock introduced by outliers into the system. Similarly, using the decile mean of the squared residuals as the fitness function in the Particle Swarm Optimization (PSO) training process prevents deviations from the target values from negatively affecting the training. The smoothing parameter feedback mechanism, integrated into the architecture and inspired by the simple exponential smoothing method, has enhanced the network's stability and its ability to adapt to financial trends.

In future studies, the performance of the proposed PANN model could be tested on various macroeconomic time series that include high-frequency data, exchange rates, or seasonality. Additionally, the integration of the percentage-based neural architecture into different deep learning architectures emerges as a promising area for further research.

Acknowledgment

This study is funded by the Turkish Scientific and Technological Research Council (TÜBİTAK) 2224-A - Grant Program for Participation in Scientific Meetings Abroad.

References

Hocreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory, *Neural Computation* 9(8) 1735–1780.

Cho, K., Van M. B., Bahdanau, D., Bougares, F., Schwenk H., Bengio Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. Association for Computational Linguistics. arXiv:1406.1078.

Egrioglu, E., Baş, E., & Chen, M. Y. (2022). Recurrent dendritic neuron model artificial neural network for time series forecasting. *Information Sciences*, *607*, 572-584.

Egrioglu, E., Bas, E., & Karahasan, O. (2023). Winsorized dendritic neuron model artificial neural network and a robust training algorithm with Tukey's biweight loss function based on particle swarm optimization. Granular Computing, 8(3), 491-501.

Egrioglu, E., & Bas, E. (2023a). A new hybrid recurrent artificial neural network for time series forecasting. *Neural computing and applications*, *35*(3), 2855-2865.

Egrioglu, E., & Bas, E. (2023b). Modified pi sigma artificial neural networks for forecasting. *Granular Computing*, *8*(1), 131-135.

Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986). Learning internal representations by error propagation in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, *MIT Press* 1(8) 318-362.

Shin, Y., Ghosh, J. (1991). The pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation. *Proc. IJCNN-91-Seattle International Joint Conference on Neural Networks IEEE*, 13–18.

Todo, Y., Tamura, H., Yamashita, K. ve Tang, Z. (2014). Unsupervised learnable neuron model with nonlinear interaction on dendrites. *Neural Networks* 60: 96-103.

Yadav, R. N., Kalra, P. K., John J. (2007). Time series prediction with single multiplicative neuron model. *Applied Soft Computing* 7(3) 1157-1163.

Contact

Eren Bas
Giresun University
Gure Campus, Faculty of Arts and Science, Department of Statistics, Giresun, Türkiye, 28200.
eren.bas@giresun.edu.tr

Erol Egrioglu
Giresun University
Gure Campus, Faculty of Arts and Science, Department of Statistics, Giresun, Türkiye, 28200.
erol.egrioglu@giresun.edu.tr