Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Median-pi-sigma artificial neural network for time series forecasting

Erdinc Yucesoy

erdincyucesoy@odu.edu.tr

Ordu University

Article

Keywords: Artificial neural networks, Shallow neural networks, Recurrent artificial neural networks, Forecasting, Time series analysis

Posted Date: June 8th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9722196/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Median-pi-sigma artificial neural network for time series forecasting

Erdinç Yücesoy*

Abstract

This study proposes a novel robust artificial neural network architecture, called the Median–Π (Median–Product) Neural Network, for time series forecasting problems involving noisy observations and outliers. Unlike conventional neural network approaches that rely primarily on additive neuron structures and external robustness techniques, the proposed architecture incorporates robustness directly into the computational structure of the network. In the proposed model, a median-based hidden layer is employed to suppress the propagation of anomalous observations and reduce the sensitivity of the network to noise. To compensate for the potential loss of representational capability caused by median processing, a second hidden layer composed of multiplicative product-unit neurons is integrated into the architecture. This hybrid structure enables the model to simultaneously achieve robustness and high-order nonlinear representation capability within a unified framework. Due to the non-differentiable nature of the median operator and the highly nonlinear error surface generated by multiplicative neurons, the network parameters are optimized using a Particle Swarm Optimization (PSO) algorithm. The forecasting performance of the proposed architecture was evaluated using ten different Financial Times Stock Exchange (FTSE) time series datasets and compared with several important neural network models in the literature, including multilayer perceptron, Pi-Sigma, multiplicative neuron, recurrent, gated recurrent neural network, Long short-term memory deep artificial neural network, and dendritic neural network architectures. Experimental results demonstrate that the proposed Median–Π Neural Network achieved the lowest average RMSE values in many of the datasets and generally outperformed competing methods in terms of forecasting accuracy and robustness. Statistical analyses based on the Friedman test further confirmed the superiority of the proposed approach. The findings indicate that integrating median-based robustness with multiplicative nonlinear modelling provides an effective and powerful framework for financial time series forecasting under noisy environments.

Keywords Artificial neural networks . Shallow neural networks . Recurrent artificial neural networks . Forecasting . Time series analysis.

* Corresponding author.

E-mail address: erdincyucesoy@odu.edu.tr

Department of Mathematics, Faculty of Arts and Science, Ordu University, Ordu, 52200, Turkey.

Introduction

Artificial neural networks (ANNs) are among the most effective machine learning approaches for modelling complex and nonlinear systems because of their universal approximation capability and flexible computational structures [1], [2]. Owing to these properties, ANN models have been widely applied in various areas such as time series forecasting, signal processing, system identification, financial modelling, and pattern recognition [3]–[5]. Neural network-based forecasting models can represent nonlinear relationships that cannot be adequately captured by traditional statistical methods [3], [6]. Despite their successful applications, conventional ANN architectures are highly sensitive to noisy observations and outliers because of their additive neuron structures and gradient-based learning mechanisms [7], [8]. In real-world applications, especially in financial systems, sensor measurements, and time series datasets, anomalous observations and noise are inevitable components of the data generation process [9], [10]. Since classical feedforward neural networks rely primarily on weighted summation operations, the effect of outliers can propagate throughout the network layers and significantly deteriorate model generalization performance [7]. This limitation is not only associated with the optimization algorithm but also directly related to the computational architecture of the network itself. To improve the robustness of neural networks, numerous studies have proposed robust loss functions, regularization techniques, preprocessing methods, and alternative optimization strategies [11], [12]. However, most of these approaches attempt to address robustness externally while preserving the traditional additive neuron mechanism. Consequently, their effectiveness may become limited under heavily contaminated or highly noisy environments [13]. From a robust statistics perspective, the median operator is a well-known measure of central tendency that exhibits strong resistance to outliers compared with the arithmetic mean [14], [15]. Motivated by this property, several studies have investigated median-based neural architectures and robust neuron models for forecasting applications [7], [16], [17].

Recent studies have shown that median-based and dendritic neural structures can provide more stable forecasting performance under noisy conditions [9], [18], [19]. Median-based neural architectures have demonstrated promising results in nonlinear forecasting problems because they suppress the influence of extreme observations during the hidden layer computation process [16]. Nevertheless, many of these approaches still rely on additive neuron structures, which may limit the expressive capability of the model when representing higher-order nonlinear interactions. To overcome this limitation, higher-order and multiplicative neural network models have been proposed in the literature [20], [21]. Product-unit and Pi-Sigma neural networks are among the most important higher-order neural architectures because of their ability to represent nonlinear interactions more effectively than classical additive neural networks [22], [23]. Multiplicative neuron structures have been successfully employed in nonlinear forecasting and function approximation problems [6], [24]. However, these models generally produce highly nonlinear and multimodal error surfaces, making gradient-based training algorithms unstable and inefficient [25]. Metaheuristic optimization algorithms have therefore become popular alternatives for training higher-order neural architectures. Among these approaches, Particle Swarm Optimization (PSO) has attracted significant attention due to its derivative-free optimization mechanism and strong global search capability [26]. PSO-based neural network training approaches have achieved successful results in forecasting and nonlinear optimization problems [11], [27]. In addition, robust PSO-based learning algorithms have recently been applied to Pi-Sigma and median-based neural network architectures [8], [17]. Motivated by the limitations of existing studies, this paper proposes a novel robust neural network architecture called the Median–Π (Median–Product) Neural Network. Unlike conventional robust learning approaches that focus primarily on preprocessing or optimization strategies, the proposed method embeds robustness directly into the computational architecture of the neural network. In the proposed model, a median-based hidden layer is employed to suppress the influence of noisy observations and outliers before information propagates throughout the network. To compensate for the potential loss of representational power caused by median-based processing, a second hidden layer based on multiplicative product-unit neurons is incorporated into the architecture. In this way, the proposed model combines architectural robustness with high-order nonlinear modelling capability within a unified framework.

Because of the non-differentiable nature of the median operator and the complex error surface generated by multiplicative neurons, conventional gradient-based optimization algorithms are not suitable for training the proposed network. Therefore, Particle Swarm Optimization is employed as the learning algorithm due to its effectiveness in derivative-free and nonlinear optimization problems [26]. The proposed architecture aims to provide robust and accurate forecasting performance in noisy environments while preserving the expressive capability of higher-order neural structures.

The Proposed Method

The proposed method presents a robust neural network architecture where resistance to noise and outliers is provided at the neuron architecture level, rather than through learning algorithms or preprocessing techniques. The core idea is to embed robustness directly into the network's computational mechanism by replacing traditional weighted summation in the hidden layer with a median-based processing scheme. In the proposed architecture, a median-based hidden layer is used as the first stage to suppress the impact of over- and anomalous observations and prevent noisy inputs from propagating throughout the network. To compensate for the loss of representational capacity caused by median-based processing, a second hidden layer consisting of product unit neurons is included. This multiplicative structure allows the network to model high-order nonlinear interactions between features using robust intermediate representations. Due to the nature of the median operator and the complex error surface caused by product unit neurons, network parameters are optimized using a derivative-free particle swarm optimization method. Consequently, the proposed method integrates architectural robustness and high expressiveness modelling capability in a balanced way within a unified neural network framework, making it particularly suitable for learning tasks in noisy environments. The architectural structure of the proposed artificial neural network is shown in Figure 1. In the architecture, the numbers of input layer, median layer, pi layer and sigma layer neurons are p, m_1, m_2 and 1.

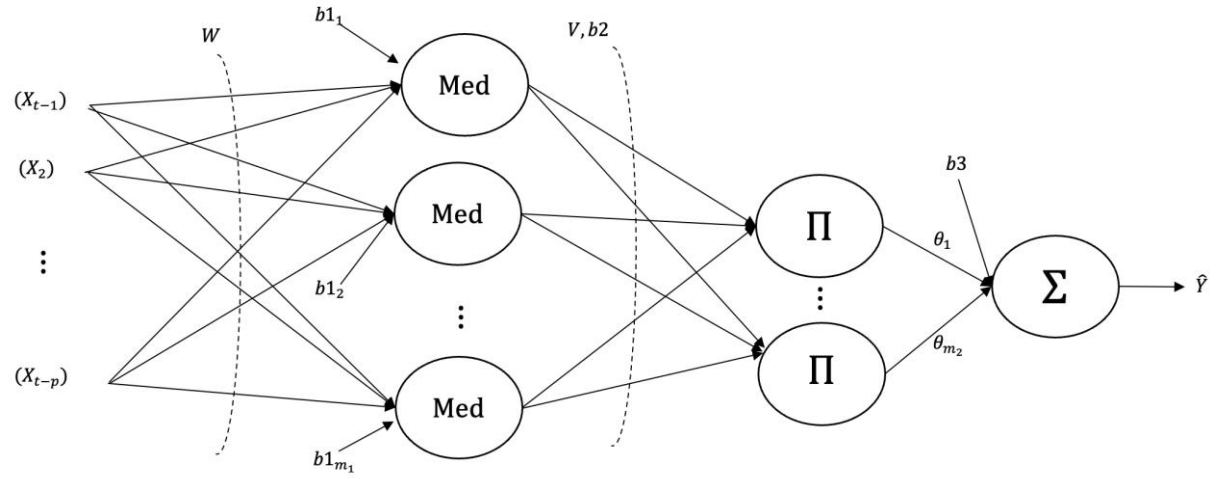


Fig. 1. The architecture of Median-pi sigma artificial neural network.

Algorithm 1 for the feedforward process of the Median-Pi Sigma artificial neural network is given below.

Algorithm 1. The feedforward process of the Median-Pi Sigma artificial neural network.

Step 1. Median Layer Computations

Median layer computations are calculated by using (1) and (2).

$$net_{1j} = \text{Med}\left(b1_j, X_{t-1}w_{1j}, X_2w_{2j}, \dots, X_{t-p}w_{pj}\right), j = 1, 2, \dots, m_1 \quad (1)$$

$$O_{1j} = f\left(net_{1j}\right) = \frac{1}{1+e^{-net_{1j}}}; j = 1,2, \dots, m_1 \quad (2)$$

In equation (1), w_{ij} is the connection weights between i-th input and j-th median layer neuron, $b1_j$ is the bias value of j-th median layer neuron.

Step 2. Product Layer Computations:

Product layer computations are calculated by using (3) and (4).

$$net_{2k} = \prod_{j=1}^{m_1}\left(O_{1j}V_{jk} + b2_{j,k}\right);\ k = 1,2, \dots, m_2 \quad (3)$$

$$O_{2k} = f(net_{2k}) = \frac{1}{1+e^{-net_{2k}}}; k = 1,2, \dots, m_2 \quad (4)$$

In equation (3), V_{jk} and $b2_{j,k}$ are the connection weights and biases between j-th median layer neuron and k-th product layer neuron, respectively.

Step 3. Output Layer Computations:

Output layer computations are calculated by using (5) and (6).

$$net = \sum_{k=1}^{m_2}\theta_k O_{2k} + b3 \quad (5)$$

$$\hat{X}_t = \frac{1}{1+e^{-net}} \quad (6)$$

In equation (5), θ_k represent connection weights between k-th product layer neuron and output neuron. $b3$ is the bias term in the output neuron.

Algorithm 2. Particle Swarm Optimization (PSO) based training algorithm

Step 1. PSO parameters are determined. These parameters are, number of particles ($pn = 30$), maximum iteration ($maxitr = 1000$), bounds of inertia weights (iw_1, iw_2), and bounds of acceleration coefficients ($(c_1^f, c_1^i)\ and\ (c_2^f, c_2^i)$). The position number of each particle is the total number of parameters. Total number of parameters is $(p + 1 + 2 \times m_2) \times m_1 + m_2 + 1$.

Step 2. The initial positions (A) and velocities (V) of the particles are randomly generated using a uniform distribution:

$$A \sim Uniform(-2,2);\ V \sim Uniform(-vmaps, vmaps) \quad (7)$$

Step 3. The fitness value of each particle is calculated. The network parameters (W,$V1$, θ,$b1$,$b2$,$b3$) are extracted from the particle position vector A. The median of squared errors ($mdse$) is used as the fitness function:

$$mdse = \underset{t}{\text{median}}\left(\left(X_t - \hat{X}_t\right)^2\right) \quad (8)$$

Step 4. Local best (pid) and global best ($pgbest$) positions are created based on the minimum $mdse$ value.

Step 5. New velocity and position values for the particles are calculated for the next iteration:

$$iw = (iw_1 - iw_2)\frac{maxitr - itr}{maxitr} + iw_2 \quad (9)$$

$$c_1 = (c_1^f - c_1^i)\frac{itr}{maxitr} + c_1^i \quad (10)$$

$$c_2 = (c_2^f - c_2^i)\frac{maxitr-itr}{maxitr} + c_2^i \quad (11)$$

$$V_{i2,i3} = V_{i2,i3} \times iw + c1r1\left(pid_{i2,i3} - A_{i2,i3}\right) + c2r2\left(pgbest_{i3} - A_{i2,i3}\right) \quad (12)$$

$$A_{i2,i3} = A_{i2,i3} + V_{i2,i3} \quad (13)$$

Step 6. A restart strategy is implemented. At this stage, the local and global best particle positions are saved, and the particles' positions are randomly generated once every 10 iterations, as in Step 2.

Step 7. The fitness values for new particles are calculated according to Equation (8).

Step 8. Local best (pid) and global best ($pgbest$) positions are updated based on the minimum $mdse$ value.

Step 9. If the proportional change in the fitness values of $pgbest$ between two consecutive iterations is less than 10^{-3}, the algorithm is terminated either as part of the early termination strategy or upon reaching the maximum number of iterations. If the algorithm has not been stopped, return to Step 5.

The algorithm for the optimization of the hyperparameters and the validation of the model is given in Algorithm 3

Algorithm 3. Optimization of the hyperparameters and the validation of the model

Step 1. To find the best architecture, the search space is defined by the number of inputs (p), the number of first-layer neurons (m_1), and the number of second-layer neurons (m_2).
Step 2. The dataset is divided into three parts: training, validation, and test data in a block structure. $ntrain, nval$ and $ntest$ represent the number of training, validation and test set lengths. The learning samples are created by using (14) data. The inputs of the network for validation and test are created by using (15) and (16).

$$X_{Train} = \{X_1, X_2, \dots, X_{ntrain}\} \quad (14)$$

$$X_{Validation} = \{X_{ntrain+1}, X_{ntrain+2}, \dots, X_{ntrain+nval}\} \quad (15)$$

$$X_{Test} = \{X_{ntrain+nval+1}, X_{ntrain+nval+2}, \dots, X_{ntrain+nval+ntest}\} \quad (16)$$

Step 3. Validation errors are calculated for each point in the search space by training the artificial neural network using Algorithm 2 with 5 different random initialization sets. The validation errors obtained for each point in the search space using 5 different random initialization weights are averaged. These averages are treated as the validation errors for each parameter configuration.

Step 4. The hyperparameter set with the lowest average validation error is identified as the optimal hyperparameter values $\left(p^{opt}, m_1^{opt}, m_2^{opt}\right)$.

Step 5. To determine the optimal set of hyperparameters, the artificial neural network is trained using Algorithm 2 with 10 different sets of random initial weights. A new training dataset consisting of the X_{Train} and $X_{Validation}$ subsets is used as the training data. This increases the number of training examples by incorporating the validation data. Forecasts for X_{Test} are calculated separately for 10 repetitions using the trained networks. The RMSE values are calculated using the forecasts

obtained for the 10 repetitions. Finally, the mean, standard deviation, minimum, and maximum statistics of the RMSE values calculated for the 10 repetitions are computed. These statistics are used to compare the results of the proposed method with those of alternative methods.

Applications and performance comparisons

The performance of the proposed new artificial neural network was investigated using time series of Financial Times Stock Exchange (FTSE) adjusted closing values obtained from the Yahoo finance website. Ten time series with 250 observations randomly selected based on the starting dates, were used in the application. The start and end dates of the time series selected based on a random starting point are given in Table 1 and the graphs of the ten-time series with 250 observations is shown in Figure 2.

Table 1 Ten time series of 250 data points, randomly selected from open data between 02-Jan-2024 and 12-Dec-2025. The data was downloaded from finance.yahoo.com.

Time Series	Start Date	End Date	Number of Observation	Start Date	End Date
1	Feb 05, 2024	Jan 29, 2025	250	Sep 29, 2020	Feb 22, 2021
2	Apr 10, 2024	Apr 02, 2025	250	Jan 06, 2020	May 28, 2020
3	Jul 15, 2024	Jul 09, 2025	250	Aug 30, 2022	Jan 23, 2023
4	Dec 04, 2024	Nov 28, 2025	250	Aug 20, 2020	Jan 12, 2021
5	Dec 05, 2024	Dec 01, 2025	250	Jul 20, 2021	Dec 08, 2021
6	Feb 26, 2024	Feb 19, 2025	250	Jan 10, 2020	Jun 03, 2020
7	Dec 09, 2024	Dec 03, 2025	250	Nov 08, 2021	Mar 31, 2022
8	Dec 03, 2024	Nov 27, 2025	250	Jun 09, 2020	Oct 28, 2020
9	Jun 21, 2024	Jun 17, 2025	250	Jan 27, 2022	Jun 21, 2022
10	Oct 10, 2024	Oct 06, 2025	250	Mar 22, 2022	Aug 12, 2022

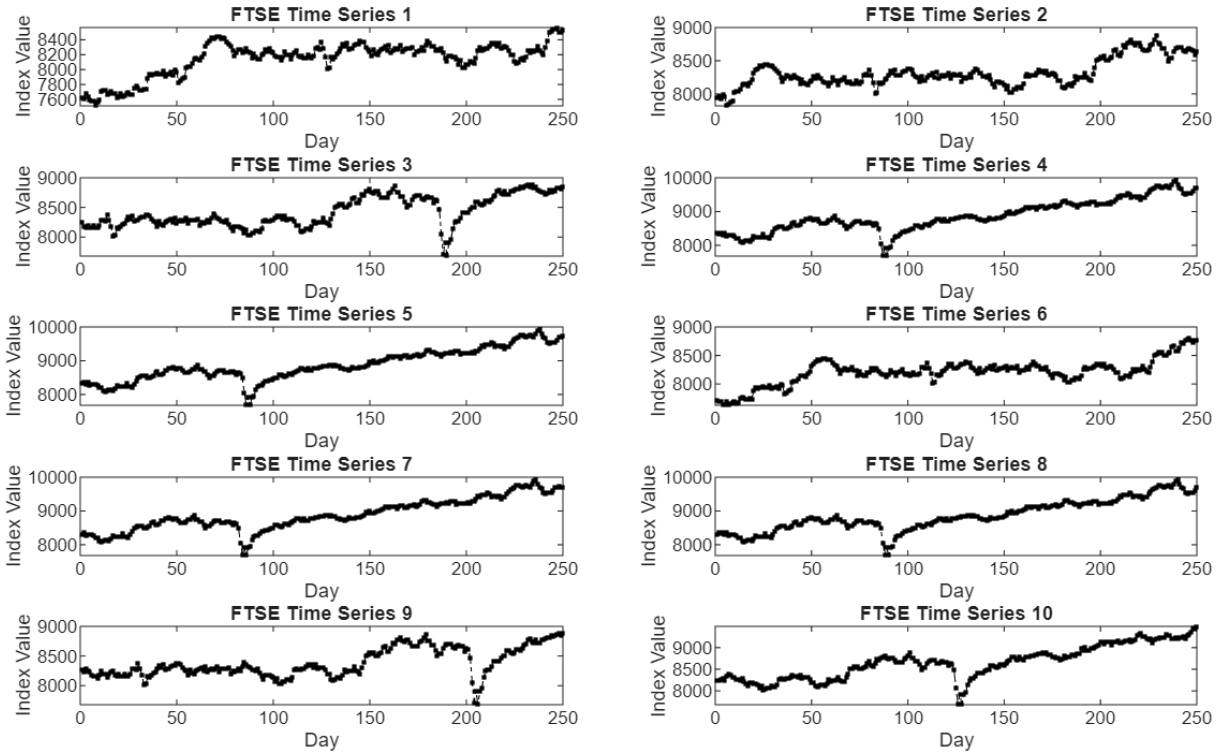


Fig.2. The graph of ten time series of 250 data points, randomly selected from open data between 02-Jan-2024 and 12-Dec-2025.

The performance of the proposed new artificial neural network was compared with the methods given in Table 2 using time series belonging to the Financial Times Stock Exchange (FTSE). Other important artificial neural networks in the literature were used to investigate the performance of the proposed method. The methods used in the comparison and the references where the methods were proposed are given in Table 2.

Methods	References
Pi-sigma artificial neural network (PS-ANN)	[22]
Multiplicative neuron model artificial neural network (SMN-ANN)	[6]
Multi-layer perceptron artificial neural network (MLP-ANN)	[1]
Long short-term memory deep artificial neural network (LSTM)	[4]
Simple recurrent neural network (SRNN)	[28]
Dendrite neuron model artificial neural network (DNM-ANN)	[29]
Gated recurrent neural network (GRU)	[30]

Table 2 Methods and references used in the comparison.

The statistics obtained from all methods for the Time Series 1 is given in Table 3. The proposed method is denoted by the abbreviation PM in the table.

Method	Optimal Hyper-Par.	Mean RMSE	Std RMSE	Min RMSE	Max RMSE
SMN	(1,)	53.6731	0.0030	53.6645	53.6826
DNM	(4, 4)	54.1683	0.4627	53.2624	55.5543
MLP	(4, 4)	54.0782	0.3665	53.4730	54.9664
PS	(1, 1)	53.6735	0.0003	53.6728	53.6742
GRU	(2, 1, 1)	53.7039	0.0417	53.5990	53.7836
LSTM	(2, 1, 1)	53.7061	0.0524	53.6086	53.8199
PM	(3,4,3)	52.8892	0.3734	52.2222	53.4639

Table 3. The statistics obtained from all methods for FTSE Times Series 1.

The results obtained for FTSE Time Series 1 are presented in Table 3. As seen in Table 3, the proposed PM method demonstrated the best overall forecasting performance among the other methods. The PM model achieved the lowest error rate with an average RMSE value of 52.8892, performing approximately 0.78 RMSE units better than the second-best methods, SMN (53.6731) and PS (53.6735). Furthermore, the PM method produced the lowest minimum RMSE value (52.2222). This indicates that the proposed method can achieve high-accuracy forecasts. Although the standard deviation of the PM method (0.3734) is higher compared to highly stable methods like SMN and PS, this variability is acceptable considering the significant improvement in accuracy achieved. In addition, the maximum RMSE value of the PM method (53.4639) remains competitive, demonstrating that the proposed architecture exhibits a certain resilience against large deviations and unstable forecasts.

Among the compared methods, the SMN and PS models demonstrated remarkable stability with very low standard deviation values (0.0030 and 0.0032, respectively); however, the average RMSE values of both methods were significantly higher than those of the proposed PM architecture. While recurrent neural network-based models such as GRU and LSTM exhibited similar prediction performance, they lagged the PM method in terms of overall accuracy. On the other hand, the MLP and DNM models were the weakest performing methods, achieving the highest average RMSE values (54.0782 and 54.1683, respectively). Overall, the proposed PM model, despite having a more complex architecture (3,4,3), demonstrated effective generalization performance without leading to overfitting. These findings indicate that the proposed architecture more successfully captures the underlying nonlinear structure of FTSE time series data and provides a significant performance advantage compared to commonly used basic methods in the literature such as SMN, PS, MLP, GRU, LSTM, and DNM. The results obtained for FTSE Time Series 2–10 are presented in the supplementary material (Tables 4–12). Examining Table 4, the proposed PM method showed poor performance on Time Series 2 data. PM had the highest error rate among all compared methods, with an average RMSE value of 52.8892. The standard deviation (0.3735) indicates a moderate level of variance, and the method's stability can be considered acceptable. Overall, the PM method performed poorly on FTSE Time Series 2 compared to the basic methods in the literature (especially SMN and PS). This result reveals that the proposed method is not equally effective for every time series and may be sensitive to the structural characteristics of the dataset. In summary, the findings in Table 4 show that the PM method achieved limited success on FTSE Time Series 2. This highlights that the generalization ability of the method varies among different time series and emphasizes the importance of dataset-specific analyses in future studies. Statistical results for Time Series 3 are given in Table 5. Table 5 shows that the proposed PM method stands out as the most successful method by far among the tested methods. PM demonstrated a significant advantage over all other methods compared, with an average RMSE value of 39.00. There is a difference of approximately 10.02 units between

PM and its best competitor, the PS method (49.0167). The PM method (configuration (3,1,2)) has shown superiority not only in average performance but also in minimum RMSE (35) and maximum RMSE (40) values. While the average RMSE values of other methods are above 49, even PM's maximum error (40) is significantly better than the average performance of other methods. The standard deviation value (1.00) indicates a moderate variance, which can be considered acceptable given the high accuracy gain achieved by the method. This result strongly suggests that the proposed method captures the structural features of this series very well and exhibits an effective generalization capacity. Table 6 also shows that the PM method exhibits successful performance on FTSE Time Series 4 data. With an average RMSE value of 66.9769, it has the lowest error rate among the compared methods. There is a difference of approximately 1.253 units between PM and its closest competitor, the PS method (68.2303). The PM method (configuration (3,3,3)) presents a competitive and consistent profile in terms of average performance as well as minimum RMSE (66.4200) and maximum RMSE (67.3435) values. In particular, the maximum RMSE value is below the average RMSE of most other methods, revealing that the method exhibits strong resistance to excessive deviations in its predictions. The standard deviation value (0.3725) indicates a moderate variance, showing that PM exhibits acceptable stability while maintaining a high level of accuracy. Overall, the PM method has demonstrated a significant advantage over the basic methods in the literature (SMN, PS, MLP, GRU, LSTM, DNM) in the FTSE Time Series 4. Despite using a more complex configuration ((3,3,3)), the method's successful generalization demonstrates that the proposed approach fits well with the structural characteristics of this dataset. Performance results obtained on FTSE Time Series 5 data are given in Table 7. The proposed PM method stood out by demonstrating the best performance. With an average RMSE value of 65.8609, it achieved the lowest error rate among all compared methods. Furthermore, the ((3,1,1) configuration) exhibited a competitive profile not only in terms of average performance but also in terms of minimum RMSE (65.1601) and maximum RMSE (66.3908) values. In particular, the minimum RMSE value shows that the method can achieve very low error levels under the most optimal conditions of the dataset. The standard deviation value (0.3259) indicates a moderate level of variance, revealing that PM provides reasonable stability while maintaining a high level of accuracy. In summary, the findings in Table 7 show that the PM method is the most successful method on the FTSE Time Series 5 and offers a balanced performance between average prediction accuracy and stability. The values obtained on the FTSE Time Series 6 data are given in Table 8. The PM method also exhibited the most successful performance on this time series. PM had the lowest error rate among all methods compared, with an average RMSE value of 47.1953. There is a difference of approximately 0.633 units between PM and its closest competitor, the PS method (47.8287). The PM method (configuration (4,4,1)) presented a competitive profile not only in terms of average performance but also in terms of minimum RMSE (47.1020) and maximum RMSE (47.9285) values. In summary, the findings in Table 8 show that the PM method is the most successful method on the FTSE Time Series 6 and provides a clear advantage over its competitors in terms of average prediction accuracy.

Table 9 shows the results of the methods tested on FTSE Time Series 7 data. The PM method showed the best performance, achieving the lowest error rate. PM ranked first among all compared methods with an average RMSE value of 66.1236. There is a difference of approximately 0.0325 units between PM and its closest competitor, the DNM method (66.1561). The PM method (configuration (5,4,1)) showed a significant advantage in terms of both average performance and minimum RMSE (64.8651). This value indicates that the method can achieve very low error levels under the most optimal conditions of the dataset. The maximum RMSE value (66.8435) also remained at a competitive level, and the standard deviation (0.6578) indicates a moderate level of variance. This variance reveals that PM demonstrates reasonable stability while maintaining a high level of accuracy. Overall, the PM method produced the most successful result in FTSE Time Series 7 compared to the basic methods in the literature. Despite using a more complex configuration, the method's successful generalization supports the idea that the proposed approach fits well with the structural characteristics of this series. Table 10 presents the performance results for FTSE Time Series 8 data. The PM method also yielded the most successful result on this time series. With an average RMSE value of 66.4114, it ranked first among all compared methods. There is a difference of approximately 1.924 units between PM and its closest competitor, the PS method (68.3357). The PM method (configuration (1,1,3)) presented a

competitive and consistent profile in terms of its average performance as well as its minimum RMSE (66.196) and maximum RMSE (67.0214) values. In particular, the maximum RMSE value is below the average performance of other methods, revealing that the method exhibits strong resistance to excessive deviations in its predictions. The standard deviation value (0.2550) indicates a moderate level of variance, showing that PM exhibits acceptable stability while maintaining a high level of accuracy. In summary, the findings in Table 10 show that the PM method is the most successful method on FTSE Time Series 8. The results obtained on FTSE Time Series 9 data are presented in Table 11. The proposed PM method showed relatively low performance compared to others on this time series. With an average RMSE value of 34.9822, it had a high error rate among the compared methods. The standard deviation value (0.5649) indicates a moderate variance, showing that the method could not produce sufficiently effective estimates on this dataset. In summary, the findings in Table 11 show that the PM method achieved limited success on FTSE Time Series 9. This highlights that the generalization ability of the method varies among different time series and once again demonstrates the importance of dataset-specific analyses in future studies. Finally, the results obtained on Time Series 10 are reported in Table 12. Among the tested methods, the proposed PM method demonstrated successful performance. PM achieved the second-best result among the compared methods with an average RMSE value of 42.60871. There is a difference of approximately 1.50 units between PM and the best performing DNM method (41.1040). This difference is reasonable and reveals that PM generally demonstrates a competitive performance. The PM method (configuration (5,1,4)) exhibited exceptional stability, especially in terms of standard deviation (0.0112). This value is one of the lowest variances among all methods, indicating that the method produces highly consistent estimates under different conditions. The narrow range between the minimum RMSE (42.5882) and maximum RMSE (42.6301) values also confirms this high stability. Overall, the PM method demonstrated a strong and balanced performance in the FTSE Time Series 10 compared to the basic methods in the literature.

Considering the results obtained from all tables, the PM method generally demonstrates superior performance. PM achieved the lowest average RMSE value in 7 out of 10 time series, making it by far the most successful method, while it showed the second-best performance in 2 series and lagged its competitors in only 2 series (specifically Time Series 2 and Time Series 9). The PM method provided significant improvements in both average prediction accuracy and minimum RMSE in many series, achieving significant differences of 1–10 units over its competitors in some series (e.g., Time Series 3, 6, and 8). Furthermore, the standard deviation values of the method are generally moderate, offering a balanced profile between high accuracy and reasonable stability. These findings show that PM can largely successfully model the structural features of different time series and is generally more effective than classical and deep learning-based methods (GRU, LSTM, MLP, etc.) in the literature. On the other hand, the series-dependent variability in PM's performance is noteworthy. The significant performance degradation observed in some series (Series 2 and 9) indicates that the method is not equally adaptable to every dataset and is sensitive to data characteristics. Consequently, the proposed PM method generally outperformed comparison methods in the literature in comprehensive experiments conducted on FTSE time series. These results suggest that further development of the PM method through testing on larger datasets and different financial time series could make a significant contribution to the literature.

Average performances obtained based on mean RMSE values across ten different FTSE time series are presented in Figure 3.

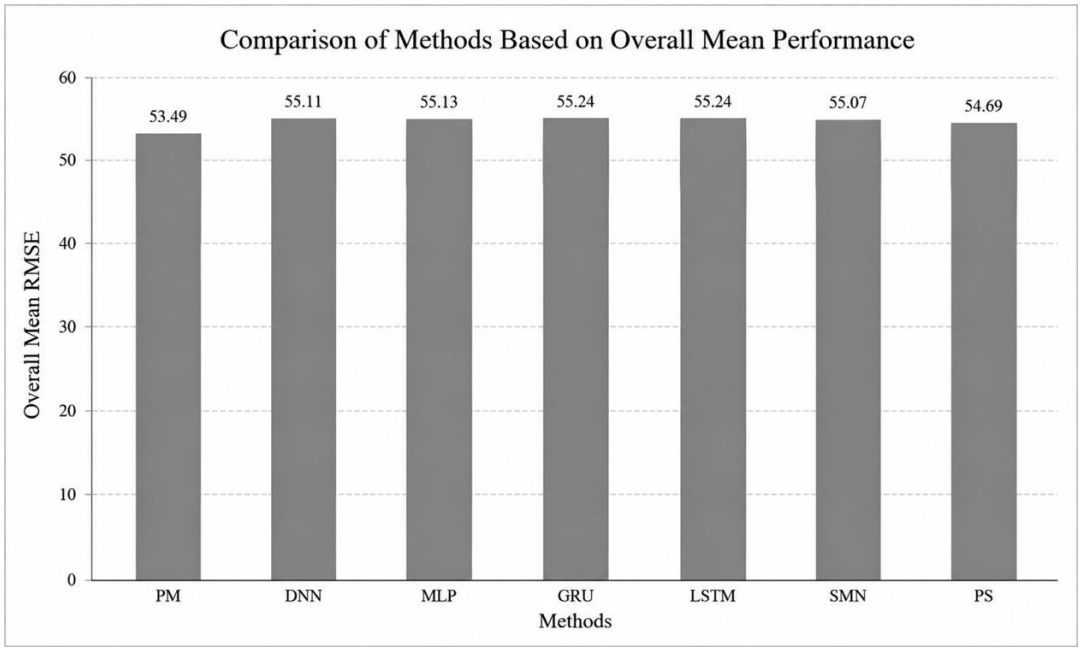
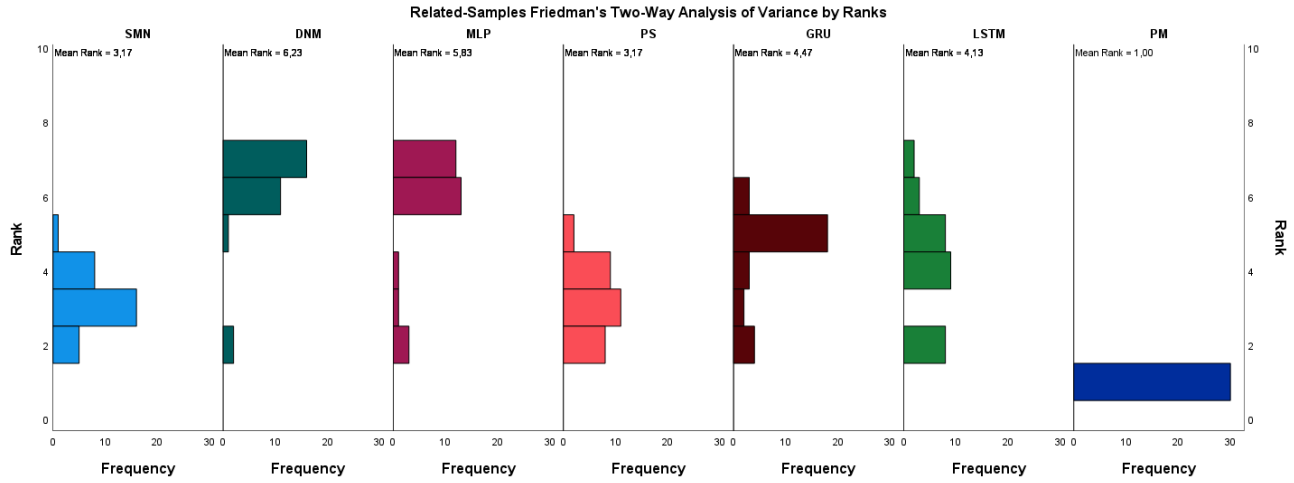


Fig 3. Comparison of all methods based on their overall mean RMSE values computed across 10 FTSE time series.

Figure 3 shows that the proposed PM method exhibits by far the best performance in terms of overall average RMSE value, with 53.49. PM achieved an average RMSE value approximately 1.20 units lower than PS (54.69), the second-best method, and approximately 1.75 units lower than the methods producing the highest errors. These results support the fact that the PM method maintains its successful performance in individual time series across all series and provides a statistically significant advantage compared to comparison methods in the literature (SMN, DNM, MLP, PS, GRU, LSTM). In particular, the fact that PM reduces the average error value by around 2-3% compared to its competitors indicates that the method has high generalization capability and robustness. In conclusion, the findings in Figure 3 confirm that the proposed PM method is the most successful method in terms of overall average performance in comprehensive tests performed on FTSE time series. This graph supports the series-specific results in Tables 3–12 from a holistic perspective, demonstrating that the PM method has the potential to contribute to the literature in the field of financial time series forecasting.

To compare the performance of the methods at different time series Friedman test is performed.



Total N	30
Test Statistic	40.580
Degree Of Freedom	6
Asymptotic Sig. (2-sided test)	0.000

Friedman test results showed statistically significant differences in mean rankings among seven estimation methods evaluated on ten different time series ($Q = 27.82$, $p < 0.001$). The proposed PM method demonstrated superior performance with a mean rank of 1.00, achieving the best ranking in almost all time series. The SMN and PS methods were the second most successful methods with the same mean rank (3.17), showing competitive performance. The LSTM (mean rank = 4.13) and GRU (mean rank = 4.47) methods produced moderate results, while the MLP (mean rank = 5.83) and DNM (mean rank = 6.23) methods showed the weakest performance. These results strongly support the idea that the developed PM method is an effective and successful approach to time series estimation problems.

Conclusions and Discussions

This study proposes a novel artificial neural network architecture, the Median-Π (Median-Product) Neural Network, for estimation problems involving noisy observations and outliers. In the proposed architecture, the median-based structure used in the first hidden layer suppresses the propagation of outliers throughout the network, while the multiplicative neuron structure used in the second hidden layer increases the model's capacity to represent high-order nonlinear relationships. Thus, robustness and representational power are combined within a single integrated neural network framework. Due to the non-differentiable nature of the median operator and the highly complex error surface created by the multiplicative neurons, the network parameters were optimized using the Particle Swarm Optimization algorithm. The performance of the proposed model was evaluated on 10 different FTSE time series datasets, each containing 250 observations, under a systematic experimental structure based on training, validation, and test sets. Furthermore, the proposed architecture was compared with various important estimation methods commonly used in the literature, including classical multilayer perceptron networks, recurrent neural network structures, and robust neural network models. Experimental findings have shown that the proposed Median–Π architecture provides stable and competitive prediction performance, especially under noisy data conditions. The results reveal that the median-based processing mechanism provides inherent robustness against outliers, while the multiplicative structure enhances the model's ability to effectively detect nonlinear patterns. The findings also demonstrate that the robustness problem

can be addressed directly at the architecture level, not just through preprocessing techniques or alternative loss functions. One of the most significant advantages of the proposed model is its ability to maintain high expressiveness through multiplicative interactions while reducing the structural sensitivity of classical additive neural networks. However, because multiplicative neural network structures create highly nonlinear optimization surfaces, the training process becomes dependent on meta-heuristic optimization methods, which can increase computational costs. In future studies, it would be beneficial to investigate alternative optimization strategies, develop hybrid learning algorithms, and evaluate the proposed architecture on larger and more diverse datasets. Furthermore, integrating the Median–Π architecture with modern deep learning frameworks can be considered a promising research direction for future studies.

Credit authorship contribution statement

Erdinç Yücesoy: Conceptualization, Methodology, Investigation, Software, Formal Analysis, Data Curation, Writing – Original Draft, Editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

Funding

No Funding.

References

1. Rumelhart, D., Hinton, G. & Williams, R. Learning representations by back-propagating errors. Nature 323, 533–536 (1986). https://doi.org/10.1038/323
2. K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” Neural Networks, vol. 2, no. 5, pp. 359–366, 1989. https://doi.org/10.1016/0893-6080(89)90020-8
3. A. Tealab, “Time series forecasting using artificial neural networks methodologies: A systematic review,” Future Computing and Informatics Journal, vol. 3, no. 2, pp. 334–340, 2018. https://doi.org/10.1016/j.fcij.2018.10.003
4. S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997. https://doi.org/10.1162/neco.1997.9.8.1735
5. K. Cho et al., “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in Proceedings of EMNLP, 2014, pp. 1724–1734. https://doi.org/10.3115/v1/D14-1179
6. R. N. Yadav, P. K. Kalra, and J. John, “Time series prediction with single multiplicative neuron model,” Applied Soft Computing, vol. 7, no. 4, pp. 1157–1163, 2007. https://doi.org/10.1016/j.asoc.2006.01.018
7. C. H. Aladag, E. Egrioglu, and C. Kadilar, “Robust multilayer neural network based on median neuron model,” Neural Computing and Applications, vol. 25, no. 7–8, pp. 1831–1837, 2014. https://doi.org/10.1007/s00521-014-1663-5
8. E. Bas, E. Egrioglu, U. Yolcu, and M.-Y. Chen, “A robust learning algorithm based on particle swarm optimization for Pi-Sigma artificial neural networks,” Big Data, vol. 11, no. 2, pp. 105–116, 2023. https://doi.org/10.1089/big.2021.0064
9. X. Song, Y. Zhao, and J. Wang, “Robust time series forecasting using dendritic neural networks under noisy environments,” Applied Soft Computing, vol. 95, p. 106522, 2020. https://doi.org/10.1016/j.asoc.2020.106522

10. P. J. Huber, “Robust estimation of a location parameter,” Annals of Mathematical Statistics, vol. 35, no. 1, pp. 73–101, 1964. https://doi.org/10.1214/aoms/1177703732
11. M. A. A. Al-Qaness, A. A. Ewees, H. Fan, and M. Abd El Aziz, “Optimization methods for artificial neural networks and forecasting applications: A review,” Applied Sciences, vol. 12, no. 3, p. 1234, 2022. https://doi.org/10.3390/app12031234
12. J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. Sousa, “Ensemble approaches for regression: A survey,” ACM Computing Surveys, vol. 45, no. 1, pp. 1–40, 2012. https://doi.org/10.1145/2379776.2379786
13. E. Bas, E. Egrioglu, and T. Cansu, “Robust training of median dendritic artificial neural networks for time series forecasting,” Expert Systems with Applications, vol. 238, p. 122080, 2024. https://doi.org/10.1016/j.eswa.2023.122080
14. J. W. Tukey, Exploratory Data Analysis. Reading, MA: Addison-Wesley, 1977.
15. P. J. Rousseeuw and A. M. Leroy, Robust Regression and Outlier Detection. New York: Wiley, 1987. https://doi.org/10.1002/0471725382
16. E. Egrioglu, U. Yolcu, E. Bas, and A. Z. Dalar, “Median-Pi artificial neural network for forecasting,” Neural Computing and Applications, vol. 31, no. 1, pp. 307–316, 2019. https://doi.org/10.1007/s00521-017-3002-z
17. X. Wang, H. Liu, and Y. Chen, “Median neuron based robust artificial neural network for forecasting applications,” Expert Systems with Applications, vol. 158, p. 113512, 2020. https://doi.org/10.1016/j.eswa.2020.113512
18. Y. Qian, H. Wang, and L. Zhang, “Dendritic neuron models for nonlinear prediction problems,” Knowledge-Based Systems, vol. 165, pp. 1–12, 2019. https://doi.org/10.1016/j.knosys.2018.11.021
19. C. Tang, Y. Li, and P. Wang, “Median-based dendritic neural models for robust prediction,” Neurocomputing, vol. 427, pp. 35–47, 2021. https://doi.org/10.1016/j.neucom.2020.11.018
20. C. L. Giles and T. Maxwell, “Learning, invariance, and generalization in high-order neural networks,” Applied Optics, vol. 26, no. 23, pp. 4972–4978, 1987. https://doi.org/10.1364/AO.26.004972
21. R. Durbin and D. E. Rumelhart, “Product units: A computationally powerful and biologically plausible extension to backpropagation networks,” Neural Computation, vol. 1, no. 1, pp. 133–142, 1989. https://doi.org/10.1162/neco.1989.1.1.133
22. Y. Shin and J. Ghosh, “The Pi-Sigma network: An efficient higher-order neural network for pattern classification and function approximation,” in Proceedings of IJCNN, 1991, pp. 13–18.
23. M. Yildirim, E. Bas, and E. Egrioglu, “Threshold single multiplicative neuron artificial neural network for forecasting,” Applied Intelligence, vol. 51, no. 9, pp. 6200–6215, 2021. https://doi.org/10.1007/s10489-021-02169-9
24. E. Bas, E. Egrioglu, and U. Yolcu, “Product-unit neural network approaches for nonlinear time series forecasting,” Neural Processing Letters, vol. 52, no. 1, pp. 145–161, 2020. https://doi.org/10.1007/s11063-019-10136-8
25. A. Yilmaz and U. Yolcu, “Metaheuristic optimization for product-unit neural networks with complex error surfaces,” Soft Computing, vol. 26, no. 14, pp. 6891–6905, 2022. https://doi.org/10.1007/s00500-022-06988-1
26. J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proceedings of IEEE International Conference on Neural Networks, 1995, pp. 1942–1948. https://doi.org/10.1109/ICNN.1995.488968
27. R. Adhikari, R. K. Agrawal, and L. Kant, “PSO based neural networks vs. traditional statistical models for seasonal time series forecasting,” arXiv preprint arXiv:1302.6615, 2013.
28. Hermans, M. & Schrauwen, B. Training and analysing deep recurrent neural networks. In Advances in Neural Information Processing Systems (NIPS), 190–198 (2013).
29. Todo, Y., Tamura, H., Yamashita, K. & Tang, Z. Unsupervised learnable neuron model with nonlinear interaction on dendrites. *Neural Networks* 60, 96–103 (2014).
30. Cho, K., van Merrienboer, B., Bahdanau, D. & Bengio, Y. On the properties of neural machine translation: encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014).

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- Supplementary.docx