AIMS Mathematics, 8(10): 24254–24273.
DOI: 10.3934/math.20231237
Received: 01 July 2023
Revised: 21 July 2023
Accepted: 28 July 2023
Published: 14 August 2023

http://www.aimspress.com/journal/Math

Research article

Forecasting the gross domestic product using a weight direct determination neural network

Spyridon D. Mourtas[1,2,*], **Emmanouil Drakonakis**[1] **and Zacharias Bragoudakis**[3,4]

[1] Department of Economics, Mathematics-Informatics and Statistics-Econometrics, National and Kapodistrian University of Athens, Sofokleous 1 Street, 10559 Athens, Greece
[2] Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University, Prosp. Svobodny 79, 660041 Krasnoyarsk, Russia
[3] Bank of Greece, 10250 Athens, Greece
[4] National and Kapodistrian University of Athens, Greece

* **Correspondence:** Email: spirmour@econ.uoa.gr.

Abstract: One of the most often used data science techniques in business, finance, supply chain management, production, and inventory planning is time-series forecasting. Due to the dearth of studies in the literature that propose unique weights and structure (WASD) based models for regression issues, the goal of this research is to examine the creation of such a model for time-series forecasting. Given that WASD neural networks have been shown to overcome limitations of traditional back-propagation neural networks, including slow training speed and local minima, a multi-function activated WASD for time-series (MWASDT) model that uses numerous activation functions, a new auto cross-validation method and a new prediction mechanism are proposed. The MWASDT model was used in forecasting the gross domestic product (GDP) for numerous nations to show off its exceptional capacity for learning and predicting. Compared to previous WASD-based models for time-series forecasting and traditional machine learning models that MATLAB has to offer, the new model has produced noticeably better forecasting results, especially on unseen data.

Keywords: neural networks; time-series forecasting; machine learning; gross domestic product; weights and structure determination
Mathematics Subject Classification: 68T10, 65F20, 91B40

1. Introduction

As information technology has advanced, new sophisticated computer approaches have been adopted in order to implement the best practices in public administration, financial management, and

planning. Several studies have demonstrated the necessity of implementing new information and communication technology (ICT)-based techniques for the reformation and enhancement of public and financial management strategies [1]. In numerous scientific fields, the use of new machine learning (ML) techniques has resulted in the acceptance of new, improved intelligence methodologies for time-series forecasting issues [2]. Methodologies for artificial intelligence have been used in a variety of fields, including but not limited to engineering, medicine, economics and finance, and social science research. They are frequently employed in the field of engineering for feedback control systems stabilization [3, 4], solar systems measurements [5], wind speed forecasting [6] and alloy behavior analysis [7]. Also, they are often employed in the field of medicine for diagnosing diabetic retinopathy [8], flat foot [9] and several types of cancer, including lung cancer [10] and breast cancer [11], whereas they are usually employed in the field of economics and finance for portfolio optimization [12], time-series forecasting [13, 14], and macroeconomic factors prediction [15, 16]. Additionally, methodologies for artificial intelligence have been successfully applied in social science research usually for multiclass classification tasks, such as characterizing occupational mobility [17], evaluating jobs' potential for teleworking [18], and classifying occupations [19].

Gross domestic product (GDP) forecasts are increasingly valuable for financial management and planning. Several studies have examined how various forecasting approaches can be used to predict the GDP, such as incorporating economic signals from the domestic economy's main trading partners [20], incorporating stochastic volatility to improve both point and density forecast accuracy [21], establishing a grey forecasting model with time power term [22], using gradient boosting and random forest models [23], and utilizing mixed-frequency factor-mixed-data sampling models [24]. ML algorithms, on the other hand, are prospective substitutes for the time-series regression models that central banks generally employ for forecasting important macroeconomic indicators [25]. For managing data sets with a large number of potential regressors, ML models are especially well suited. In this paper, we investigate the performance of different ML algorithms in obtaining accurate forecasts of GDP for the United States (U.S.), the United Kingdom (U.K.), Italy, France, Greece and India.

The main objective of this research is to develop a model for GDP forecast using innovative neural networks enriched with cutting-edge methods. To achieve this, we will employ a 3-layer feed-forward neural network that is able to handle regression tasks. As an alternative to the well-known back-propagation algorithm that is used to train feed-forward neural networks, a weights and structure determination (WASD) training algorithm will be utilized. The WASD algorithm provides the following advantages when training a neural network [26]:

- It computes the ideal set of weights directly by using the weights direct determination (WDD) process as opposed to the back-propagation algorithm, which iteratively modifies the network's structure.
- It avoids getting stuck in local minima.
- It ultimately contributes to lower computational complexity.

In this paper, a novel multi-function activated WASD for time-series (MWASDT) model is introduced. It takes into account the unique features of the WASD based model for binary classification presented in [27], which uses multiple activation functions to reduce the training error even more than the single activation function models. It also takes into account the unique features of the WASD based model for time-series forecast presented in [13], which finds the optimal number of lagged observations to reduce the training error. In order to further improve the structure and functionality of the WASD based neural networks in the case of time-series forecast, the MWASDT model specifically makes use of various

activation functions, optimizes the ratio between the fitting and validation sets (i.e., cross-validation auto-adjustment) and employs a new prediction mechanism. Results from six experiments reveal that, when compared to some of the most cutting-edge ML regression models available through MATLAB's regression learner app, the MWASDT model performed better on all measures.

The following can be used to summarize the main concepts of this work:

- A novel 3-layer feed-forward WASD neural network for time-series forecast, termed MWASDT, is presented.
- The MWASDT makes use of various activation functions, optimizes the ratio between the fitting and validation sets to reduce bias and prevent becoming caught in local optima during the training phase and employs a new prediction mechanism.
- Using the new prediction mechanism, the MWASDT model's predictions on unseen data can be kept within a user-specified reasonable range.
- In six experiments on GDP forecast, the MWASDT model is contrasted with some of the most advanced regression models accessible through MATLAB's regression learner app.
- Six experiments on GDP forecast demonstrate that the MWASDT model offers superior prediction abilities compared to other WASD models, including the WASDP model developed in [28].

The following provides a description of the paper's structure. An overview of the multi-function activated WDD process for time-series process is given in Section 2. Section 3 presents the 3-layer feed-forward MWASDT neural network structure. Section 4 presents the MWASDT algorithm as well as the whole training and forecasting processes for the MWASDT neural network model. Section 5 shows and discusses the findings of six experiments on GDP forecast using the MWASDT model, the WASDP model developed in [28] and some of the most cutting-edge regression models available through MATLAB's regression learner app. In Section 6, concluding observations are given.

2. The multi-function activated WDD process for time-series

This section describes the multi-function activated WDD process for time-series. The WDD process is an essential part of any WASD method since it does away with the requirement for labor-intensive, usually unreliable iterative computations to obtain the appropriate weights matching the current hidden layer layout. The WDD procedure reportedly allows for both speed and lower computational complexity whilst avoiding some of the accompanying challenges as opposed to traditional weight determination methods [26].

Here, comprehensive justifications of key theoretical underpinnings and studies are provided for the creation of the MWASDT neural network. First, it is important to mention a few of the main symbols used in this work: $a!$ denotes the factorial of a; $()^{\mathrm{T}}$ denotes transposition; $()^{\dagger}$ denotes pseudoinversion; sign$(\cdot)$ denotes a sign function; round$(\cdot)$ denotes a round function. $()^{\odot}$ denotes the elementwise exponential.

The Taylor polynomial approximation (TPLA) theorem is restated from [29].

Theorem 2.1. *The following holds when a target function, $f(\cdot)$, has the $(H+1)$-order continuous derivative on the range $[r_1, r_2]$, and H is a nonnegative integer:*

$$f(a) = B_H(a) + C_H(a), \quad a \in [r_1, r_2], \tag{2.1}$$

where $C_K(a)$ and $B_K(a)$, respectively, imply the error term and the H-order TPLA of $f(a)$.

Let $f^{(h)}(z)$ be the value at the point z of the h-order derivative of $f(x)$. The approximate representation of $f(a)$ is shown below:

$$f(a) \approx B_H(a) = \sum_{h=0}^{H} \frac{f^{(h)}(z)}{h!}(a-z)^h, \quad z \in [r_1, r_2]. \tag{2.2}$$

Proposition 2.1. *Theorem 2.1 can be used to approximate multivariable functions. Let the target function with v variables and $(H+1)$-order continuous partial derivatives in an origin's neighborhood $(0, \ldots, 0)$ be $f(a_1, a_2, \ldots, a_v)$. The H-order TPLA $B_H(a_1, a_2, \ldots, a_v)$ about the origin is shown below:*

$$B_H(a_1, a_2, \ldots, a_v) = \sum_{h=0}^{H} \sum_{h_1+\cdots+h_v=h} \frac{a_1 \cdots a_v}{h_1 \cdots h_v} \left(\frac{\partial^{h_1+\cdots+h_v} f(0, \cdots, 0)}{\partial a_1^{h_1} \cdots \partial a_v^{h_v}} \right), \tag{2.3}$$

where $h_1, h_2, \ldots, h_v$ are nonnegative integers.

According to [27, 13], the data also requires normalization to a range of $[-0.5, -0.25]$ before their input in the neural network model because it enhances the accuracy of the WDD method. We achieve that by using a linear transformation [30] as below:

$$A_{nor} = \frac{A - A_{min}}{4(A_{max} - A_{min})} - \frac{1}{2}, \tag{2.4}$$

where A_{max} and A_{min} are the maximum and minimum values of the time-series data $A = [A_{t-1}, A_{t-2}, \ldots, A_{t-m}] \in \mathbb{R}^{1\times m}$, respectively, with $t > m$ denoting the time. It is worth noting that the neural network can deal with over-fitting in this way. As a consequence, the normalized input A and the target vector $D = A_t \in \mathbb{R}$ are considered.

According to the power activated multi-input neural networks in [26], the nonlinear function given below may be used to express the relationship between the input variables $A_{t-1}, A_{t-2}, \ldots, A_{t-m}$ and the output target D of the neural network:

$$f(A_{t-1}, A_{t-2}, \ldots, A_{t-m}) = D. \tag{2.5}$$

Thereafter, based on Proposition 2.1, the H-order TPLA $B_H(A_{t-1}, A_{t-2}, \ldots, A_{t-m})$ can map (2.5) as shown below:

$$B_H(A_{t-1}, A_{t-2}, \ldots, A_{t-m}) = \sum_{h=0}^{n-1} k_h w_h, \tag{2.6}$$

where $k_h = G_h(A_{t-1}, A_{t-2}, \ldots, A_{t-m}) \in \mathbb{R}^{1\times mn}$ signifies a power activation function, $w_h \in \mathbb{R}^{mn}$ signifies the weight that corresponds to k_h, and h is both the number of the hidden layer neurons and the power value. Additionally, the four power elementwise activation functions (AFs) presented in Table 1 are recommended when dealing with regression tasks [13].

Table 1. Power activation functions.

Name	$G_h(X)$	Range	Reference
Power	X^h	$(-\infty, \infty)$	AF 1
Power sigmoid	$\frac{e^{X^h}}{e^{X^h}+1}$	$[\frac{1}{2}, 1)$	AF 2
Power inverse exponential	e^{-X^h}	$(0, 1)$	AF 3
Power softplus	$\ln(1 + e^{X^h})$	$(0, \infty)$	AF 4

For a given number of $mr \in \mathbb{N}$ observations, the input matrix A and the target vector D become:

$$A = \begin{bmatrix} A_{t-m} & A_{t-m+1} & \dots & A_{t-1} \\ A_{t-m-1} & A_{t-m} & \dots & A_{t-2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{t-m-r} & A_{t-m-r+1} & \dots & A_{t-r} \end{bmatrix} \in \mathbb{R}^{r\times m}, \quad D = \begin{bmatrix} A_t \\ A_{t-1} \\ \vdots \\ A_{t-r+1} \end{bmatrix} \in \mathbb{R}^r. \tag{2.7}$$

Thereafter, setting $k_{r,h} = G_h(C_1, C_2, \dots, C_m) \in \mathbb{R}^{r\times mn}$, where $C_i \in \mathbb{R}^r$ denotes the ith column of the input matrix A in (2.7), the input-activation matrix is shown below:

$$K = \begin{bmatrix} k_{1,0} & k_{1,1} & \dots & k_{1,n-1} \\ k_{2,0} & k_{2,1} & \dots & k_{2,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ k_{r,0} & k_{r,1} & \dots & k_{r,n-1} \end{bmatrix} \in \mathbb{R}^{r\times mn}, \tag{2.8}$$

and the weight vector is $W = [w_0, w_1, \dots, w_{n-1}]^{\mathrm{T}} \in \mathbb{R}^{mn}$. As opposed to the iterative weight training used in traditional neural networks, the weights of the H-order TPLA neural network are then calculated directly using the WDD process outlined below [29]:

$$W = K^{\dagger} D. \tag{2.9}$$

3. The MWASDT neural network structure

The 3-layer feed-forward neural network structure is shown in Figure 1. Particularly, the neural network receives the normalized input values $A_{t-1}, A_{t-2}, \dots, A_{t-m}$ based on (2.4) from Layer 1 (i.e., input layer) and allocates them to the relevant neuron of Layer 2 with equal weight 1. Notice that Layer 2 has a maximum number n of activated neurons. Further, the neurons that connect Layer 2 and Layer 3 (i.e., output layer) are acquired using the WDD procedure and have weights W_j, $j = 1, 2, \dots, n-1$. To compute the prediction $\check{D}$ of A_t, the formula shown below is used:

$$\check{D} = KW. \tag{3.1}$$

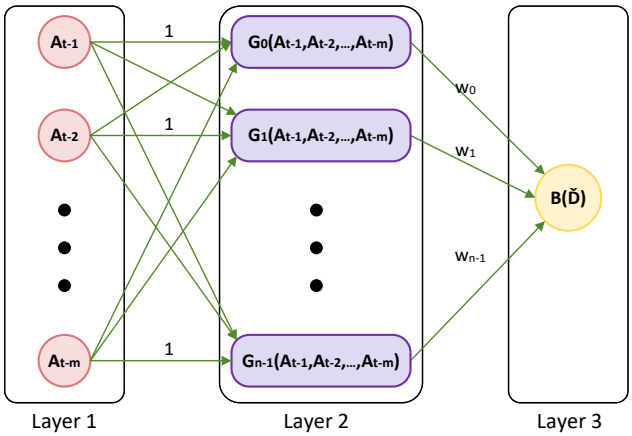


Figure 1. Structure of the MWASDT neural network.

Last, Layer 3 has one activated neuron, which uses the function shown below:

$$B(\check{D}) = \begin{cases} A_{t-1} + \text{Var}(A) \cdot \gamma & , \|\check{D}| - |A_{t-1}\| > \text{Var}(A) \cdot \gamma \ \& \ \check{D} > A_{t-1} \\ A_{t-1} - \text{Var}(A) \cdot \gamma & , \|\check{D}| - |A_{t-1}\| > \text{Var}(A) \cdot \gamma \ \& \ \check{D} < A_{t-1} \\ \check{D} & , \text{otherwise} \end{cases} \tag{3.2}$$

where $A = [A_{t-1}, A_{t-2}, \ldots, A_{t-m}]$, $\text{Var}(A)$ denotes the variance of A, and the parameter $\gamma \geq 0$ imposes a bound on in the predicted value of the neural network model. Given that the parameter γ is specified by the user, the model's predictions on unseen data can be kept within a user-specified reasonable range. Keep in mind that the predicted value of the neural network model remains unbound for $\gamma = +\infty$. It is important to mention that $B(\check{D})$ is the normalized output of the neural network. As a result, it is necessary to denormalize the value of $B(\check{D})$ using the reverse procedure shown in (2.4).

4. The MWASDT algorithm

The MWASDT algorithm is in charge of training the neural network model. That is, it finds the optimal ratio between the fitting and validation input sets (i.e., p^*), the optimal number of inputs m or observation delays (i.e., M^*), the optimal number of hidden layer neurons n, the optimal AF of each hidden layer neuron (i.e., v), and the weights W of the neural network. Consider the time-series $A_t, A_{t-1}, \ldots, A_{t-g}$, the maximum number of hidden layer units $n_{\max}$ specified by the user and, according to [13], the maximum number of inputs $M_{\max} = \text{round}(g/3)$. Also, consider the parameter $p \in [0, 1]$ which is the ratio between the fitting and validation input sets (i.e., cross-validation split). Since a typical split is above 50% for the fitting set and less than 50% for the validation set, the following iterative procedure (*(1)-(10)* steps) is used in particular for $p = 0.55 : 0.01 : 0.85$, where p takes prices from 0.55 to 0.85 with step 0.01.

(1) For $M = 1 : M_{\max}$, where M takes prices from 1 to $M_{\max}$ with step 1, repeat the following *(2)-(9)* steps.

(2) We create the input matrix A and the target matrix D according to (2.7) for $r = g$ and $m = M$, and we set $G_1 = \text{round}(pg)$, the number of the fitting data, and $G_2 = g - G_1$, the number of the validation data. That is, the training set of the neural network comprises the matrices A and D.

Algorithm 1 Matrix K calculation.

Require: The data A, the vector V that contains the optimal powers of every hidden-layer neuron checked so far, the $h = [AF\ 1, AF\ 2, AF\ 3, AF\ 4]$ that contains the optimal AFs of V, the delays number M.

$j \leftarrow 1$
while $j \leq length(V)$ **do**
  $Q \leftarrow A^{\odot V(j)}$
  if $h(j) == AF\ 1$ **then**
    $K(:, M(j-1)+1 : Mj) \leftarrow Q$
  else if $h(j) == AF\ 2$ **then**
    $K(:, M(j-1)+1 : Mj) \leftarrow e^{Q} \odot (e^{Q}+1)^{-1}$
  else if $h(j) == AF\ 3$ **then**
    $K(:, M(j-1)+1 : Mj) \leftarrow e^{-Q}$
  else if $h(j) == AF\ 4$ **then**
    $K(:, M(j-1)+1 : Mj) \leftarrow \ln(1+e^{Q})$
  end if
  $j \leftarrow j+1$
end while

Ensure: The matrix K.

(3) For $h = 0 : n_{\max} - 1$, where h takes prices from 0 to $n_{\max} - 1$ with step 1, repeat the following *(4)-(8)* steps.

(4) For $v = 1 : 4$, where v takes prices from 1 to 4 with step 1, repeat the following *(5)-(7)* steps.

(5) Create K_v (i.e., one matrix K for each of the four AF of (1)) for the input A in line with Alg. 1.

(6) Calculate the weights W_v for the first G_1 observations (fitting set) of K_v, i.e., $K_v(1 : G_1, :)$, in line with the WDD process of (2.9). That is, we set $W_v = K_v(1 : G_1, :)^{\dagger} D(1 : G_1)$.

(7) Based on W_v of the previous step and the last G_2 observations (validation set) of K_v, i.e., $K_v(G_1+1 : G_2, :)$, their predictions' mean absolute percentage error (MAPE) over the target value $D(G_1+1 : G_2)$ is measured. That is, we set $\check{D}(G_1+1 : G_2) = K_v(G_1+1 : G_2, :)W_v$, where $\check{D}(G_1+1 : G_2)$ denotes the predictions on the validation set. Note that the MAPE is a well-known statistic tool that measures the accuracy of a forecasting approach and is commonly used in ML as a loss function for regression problems. In addition, MAPE values that are closer to zero are preferable, and is calculated as follows:

$$\text{MAPE} = \frac{100\%}{g} \sum_{i=1}^{g} \left| \frac{\hat{D}_i - D_i}{D_i} \right|, \tag{4.1}$$

where $\hat{D}$ and D, respectively, are the forecasted and the target prices.

(8) Based on the MAPE, the best performing AF (i.e., v^*) of each neuron h is chosen iteratively. If the MAPE of v^* is lower than the previous best MAPE, then a new hidden-layer neuron is created under that AF (i.e., we set the AF $V = [V, v^*]$ and the hidden-layer neuron power $N = [N, h]$), otherwise this v^* is bypassed and not included in the hidden-layer neurons. As a result, we keep at minimum the hidden-layer neurons while reducing the overall MAPE of the neural network model.

(9) Compare the best MAPE of the current M to the minimum MAPE of the optimal delays number so far (i.e., M^*). If the best MAPE of the current M is lower than M^*, it becomes the minimum MAPE, and the current M becomes the optimal delays number.

(10) Compare the best MAPE of the current p to the minimum MAPE of the optimal ratio between the fitting and validation input sets so far (i.e., p^*). If the best MAPE of the current p is lower than p^*, it becomes the minimum MAPE, and the current p becomes the optimal ratio between the fitting and validation input sets.

In this way, the MWASDT algorithm can keep the lowest number of hidden-layer neurons in the neural network while optimizing the ratio between the fitting and validation input sets through the parameter p. At the same time, it finds the optimal number of inputs and lowers the MAPE of the neural network as a whole. The full workflow of the MWASDT algorithm is depicted in the diagram in Figure 2.

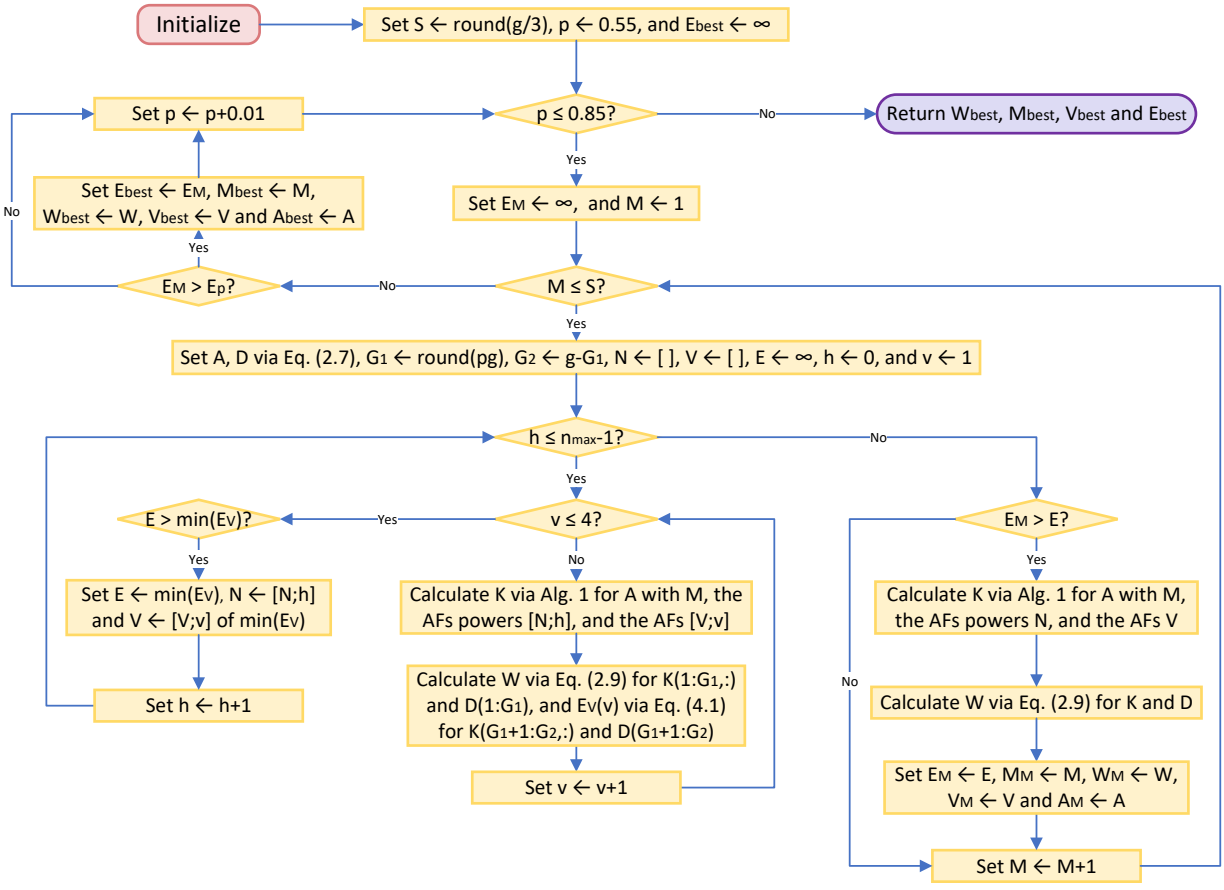


Figure 2. The MWASDT algorithm.

After finding the optimal structure of MWASDT neural network model of Figure 1, we use the last M observations of the time-series, i.e., $A_t, A_{t-2}, \ldots, A_{t-M-1}$, to create the test set of the neural network. The forecast for the next $Z \in \mathbb{N}$ in number time instances can therefore be obtained through (3.2). Particularly, for $z = 1 : Z$, where z takes prices from 1 to $Z \in \mathbb{N}$ with step 1, we repeat the following *(a)*-*(b)* steps.

(a) By using N, V and the observations $A_{t+z-1}, A_{t+z-2}, \ldots, A_{t+z-M}$, create K.
(b) Forecast the price of A_{t+z} through (3.2).

As a result, we are able to obtain the neural network forecast for the next Z in number time instances. The detailed procedure for training and forecasting with the MWASDT neural network model is depicted in the diagram in Figure 3.

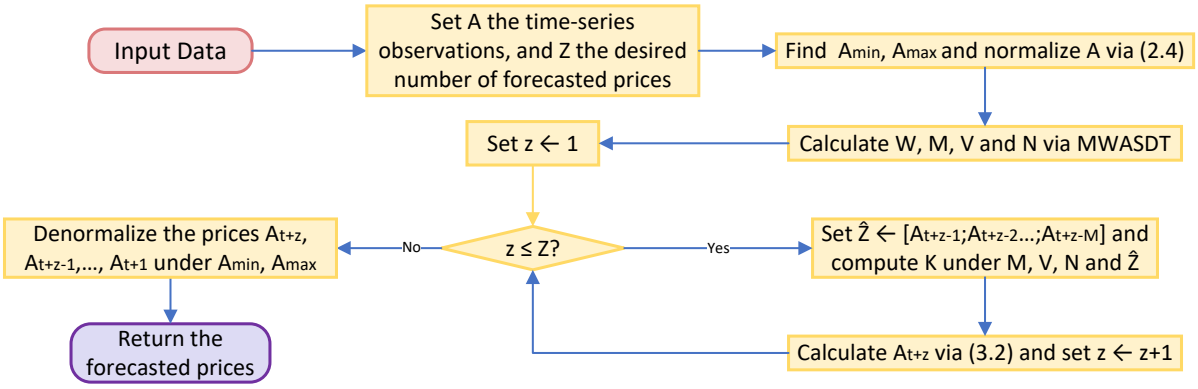


Figure 3. Procedure for training and forecasting with the MWASDT neural network.

5. Experiments on GDP forecast

This section looks at the MWASDT model's efficiency and forecasting capacity when used with real-world data. Particularly, the performance of the MWASDT neural network is investigated and compared to some of the best-performing regression models accessible in the MATLAB classification learner app during six experiments on GDP forecast. The exponential Gaussian process regression (EGPR), linear support vector machine (LSVM), and ensemble bagged trees (EBT) are these regression models. The WASDP neural network model developed in [28] is also compared. With the following link, you may get the whole creation and deployment of the concepts and computational methods featured in Sections 2–4 from GitHub:

`https://github.com/SDMourtas/MWASDT`.

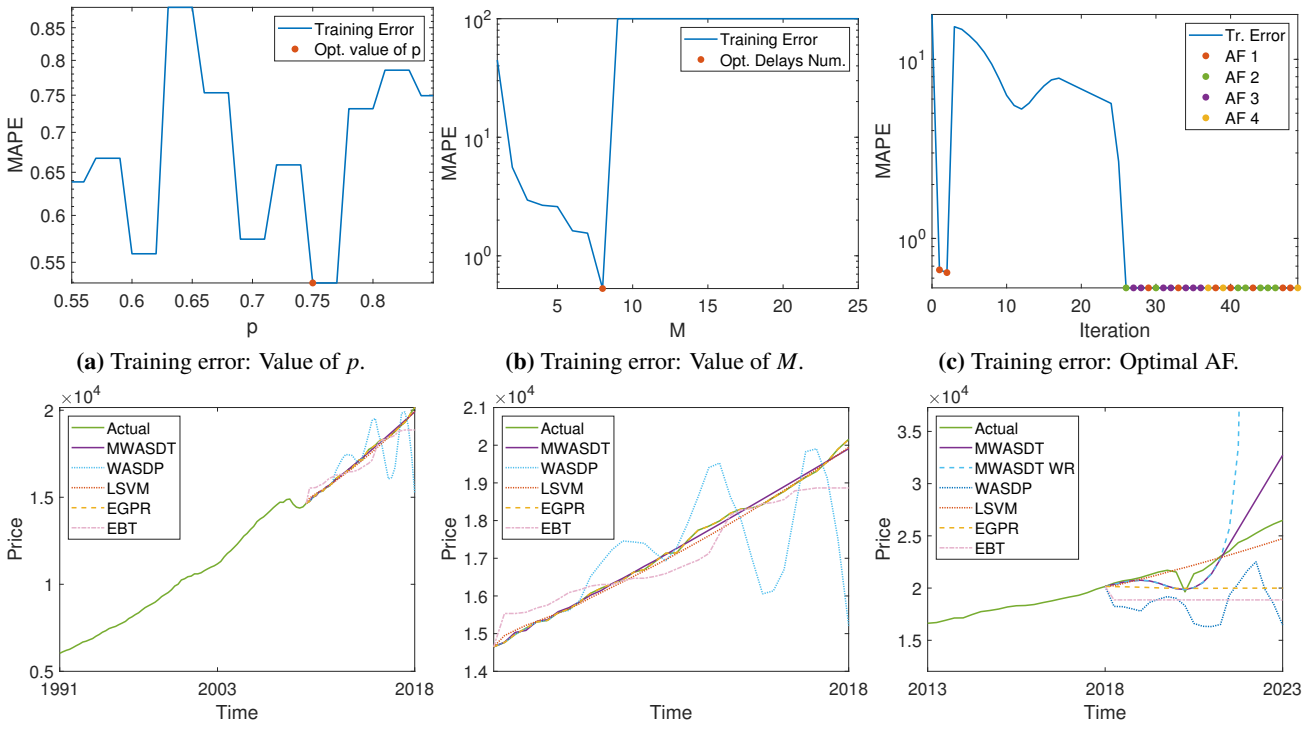
The MATLAB package provides full implementation and extensive installation instructions.

5.1. Experiments description

The datasets used in the experiments of this section are taken from the Federal Reserve Economic Data (FRED) at `https://fred.stlouisfed.org/`, which contains frequently updated U.S. macro and regional economic time-series at annual, quarterly, monthly, weekly, and daily frequencies. For our experiments, we employed the quarterly GDP time-series for the U.S., U.K, Italy, France, Greece and India. The time frames for training the neural network models are from 1/1991 to 10/2017 for the U.S., U.K and France, from 1/1997 to 10/2017 for Italy and Greece, and from 4/2004 to 10/2017 for India. These time frames include 109, 93 and 56 observations, respectively. Therefore, we set $g = 109$, $g = 93$ and $g = 56$, respectively, in the MWASDT algorithm described in the diagram of Figure 2.

Additionally, we set $n_{\max} = 50$ for all the time frames in the MWASDT algorithm. The time frame for testing the neural network models is from 1/2018 to 1/2023 for all the aforementioned nations and includes 20 observations. As a result, we set $Z = 20$ in the process described in the diagram of Figure 3. It is important to note that we set $\gamma = 10$ in (3.2) for all the time frames in the MWASDT algorithm with the exception of Italy's GDP, where we have set $\gamma = 1$. We have also set $\gamma = +\infty$ for all the time frames in the MWASDT algorithm, and in the figures and table of this section, we designate it as MWASDT WR (i.e., MWASDT without restrictions).

In the case of the U.S.'s GDP, the neural network training and test results are presented in Figure 4. Particularly, the MAPE of the validation set during training is shown in Figure 4a–4c for various ratios of the fitting and validation input sets p, various delays M for the optimal value of p, and various AFs for the optimal values of p and M, respectively. More particularly, Figure 4a shows that the optimal value of p is 0.75, Figure 4b shows that the optimal value of M is 8, and Figure 4c shows that the optimal number of hidden-layer neurons is 26. In Figure 4d and 4e, which depict the predicted prices on the validation set during training, we can see that the MWASDT, LSVM, and EGPR models have a better match with the actual prices than the WASDP and EBT models. As shown in Figure 4f, the MWASDT and LSVM models are more accurate at forecasting the actual prices than the MWASDT WR, WASDP, EGPR and EBT models.



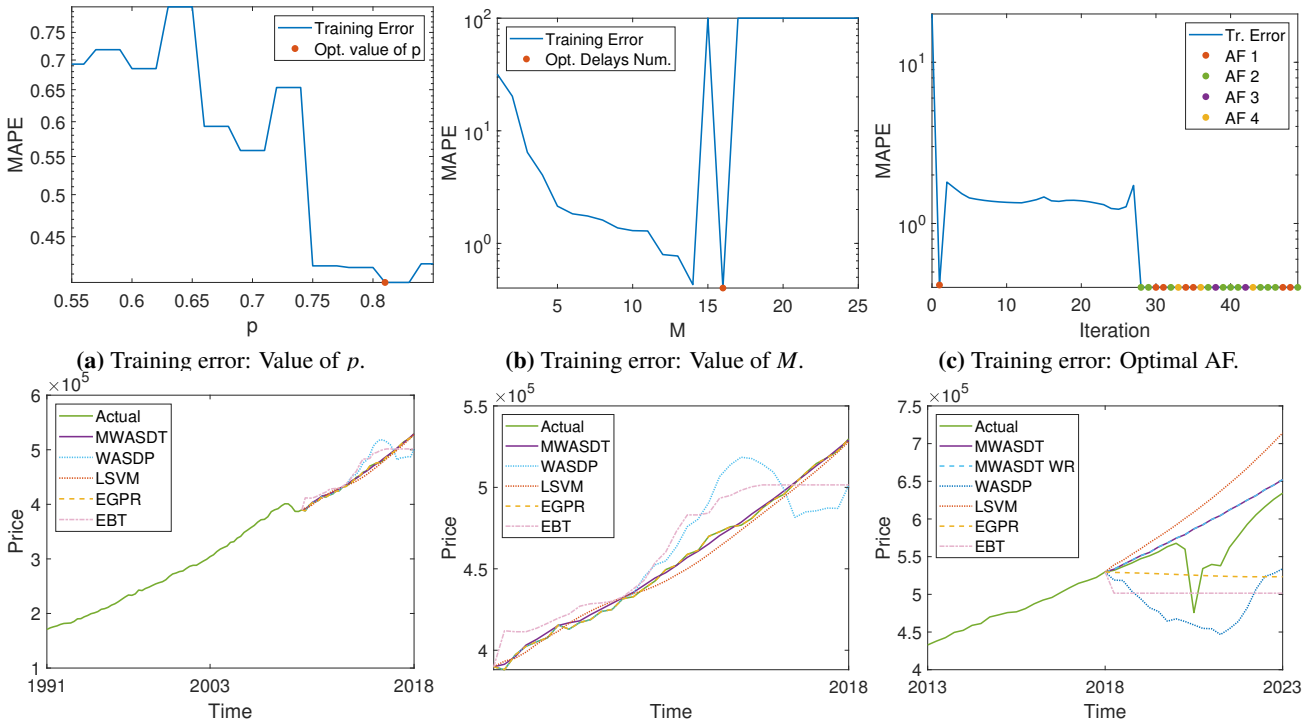
(a) Training error: Value of p. **(b)** Training error: Value of M. **(c)** Training error: Optimal AF.

(d) Fitting and validation set. **(e)** Validation set. **(f)** Test set (forecast).

Figure 4. Training and testing the neural network models on the GDP of U.S..

When it comes to the U.K.'s GDP, the neural network training and test results are presented in Figure 5. Particularly, the MAPE of the validation set during training is shown in Figure 5a–5c for various ratios of the fitting and validation input sets p, various delays M for the optimal value of p and

various AFs for the optimal values of p and M, respectively. More particularly, Figure 5a shows that the optimal value of p is 0.81, Figure 5b shows that the optimal value of M is 16, and Figure 5c shows that the optimal number of hidden-layer neurons is 23. In Figure 5d and 5e, which depict the predicted prices on the validation set during training, we can see that the MWASDT, LSVM, and EGPR models have a better match with the actual prices than the WASDP and EBT models. The MWASDT and MWASDT WR models are more accurate at forecasting the actual prices than the other models, as shown in Figure 5f.



(a) Training error: Value of p. **(b)** Training error: Value of M. **(c)** Training error: Optimal AF.

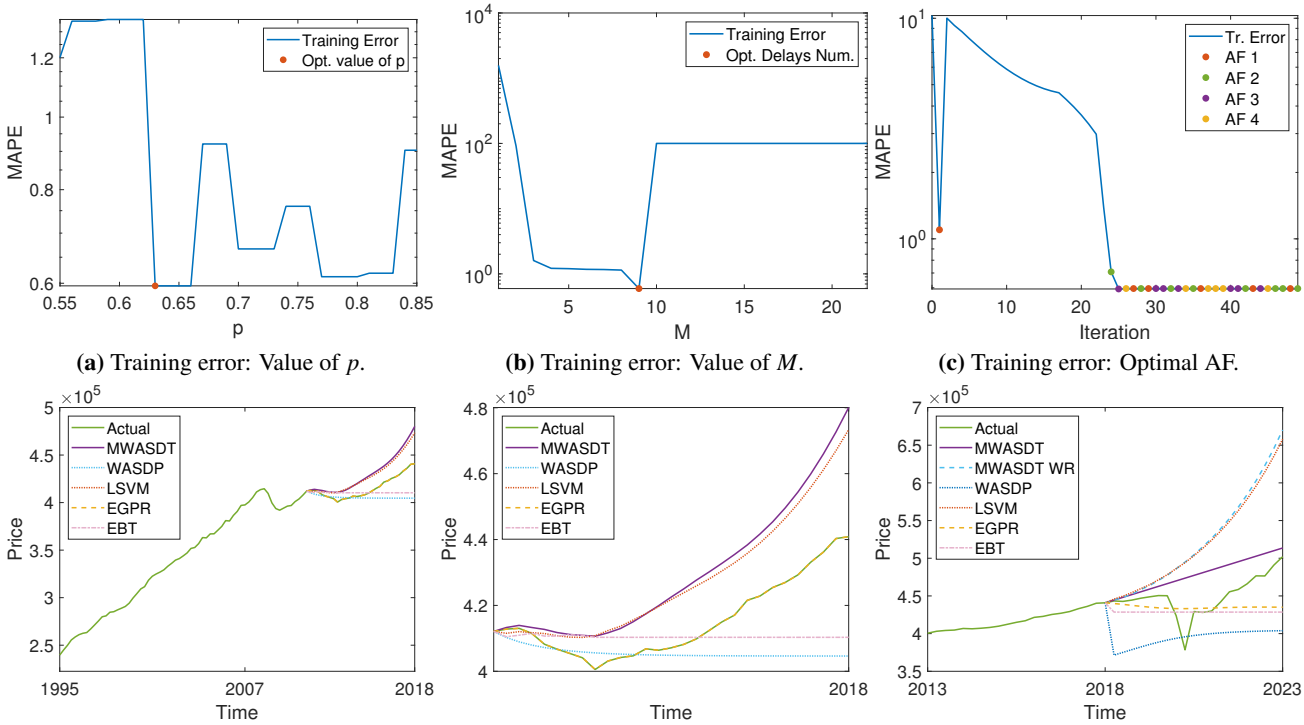
(d) Fitting and validation set. **(e)** Validation set. **(f)** Test set (forecast).

Figure 5. Training and testing the neural network models on the GDP of U.K..

Regarding Italy's GDP, the neural network training and test results are presented in Figure 6. Particularly, the MAPE of the validation set during training is shown in Figure 6a–6c for various ratios of the fitting and validation input sets p, various delays M for the optimal value of p, and various AFs for the optimal values of p and M, respectively. More particularly, Figure 6a shows that the optimal value of p is 0.63, Figure 6b shows that the optimal value of M is 9, and Figure 6c shows that the optimal number of hidden-layer neurons is 27. In Figure 6d and 6e, which depict the predicted prices on the validation set during training, we can see that the EGPR model has a better match with the actual prices compared to the other models. As shown in Figure 6f, the WASDP model is less accurate at forecasting the actual prices than the rest of the models.

In the case of the France's GDP, the neural network training and test results are presented in Figure 7. Particularly, the MAPE of the validation set during training is shown in Figure 7a–7c for various ratios of the fitting and validation input sets p, various delays M for the optimal value of p, and various AFs for the optimal values of p and M, respectively. More particularly, Figure 7a shows that the

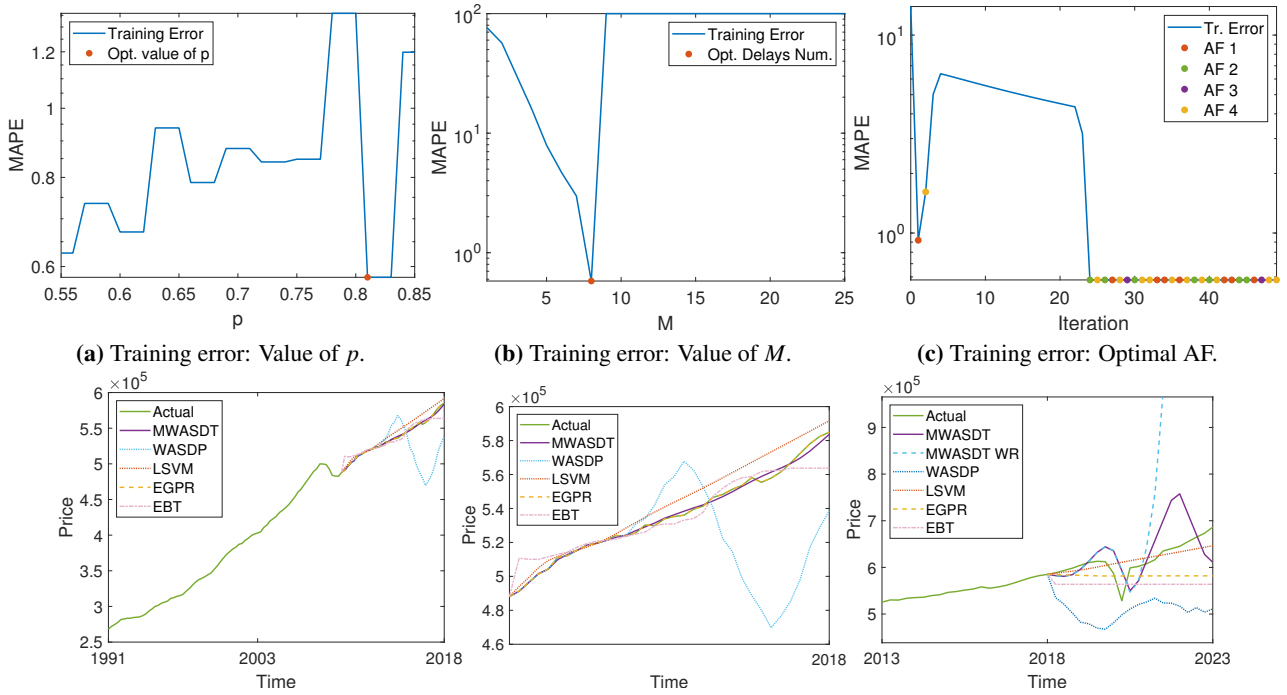
optimal value of p is 0.81, Figure 7b shows that the optimal value of M is 8, and Figure 7c shows that the optimal number of hidden-layer neurons is 28. In Figure 7d and 7e, which depict the predicted prices on the validation set during training, we can see that the MWASDT and EGPR models have a better match with the actual prices compared to the rest of the models. The MWASDT and LSVM models are more accurate at forecasting the actual prices than the MWASDT WR, WASDP, EGPR and EBT models, as shown in Figure 7f. When it comes to Greece's GDP, the neural network training and test results are presented in Figure 8. Particularly, the MAPE of the validation set during training is shown in Figure 8a–8c for various ratios of the fitting and validation input sets p, various delays M for the optimal value of p, and various AFs for the optimal values of p and M, respectively. More particularly, Figure 8a shows that the optimal value of p is 0.81, Figure 8b shows that the optimal value of M is 9, and Figure 8c shows that the optimal number of hidden-layer neurons is 20. In Figure 8d and 8e, which depict the predicted prices on the validation set during training, we can see that the WASDP model is less accurate at matching the actual prices than the other models. The MWASDT and MWASDT WR models are more accurate at forecasting the actual prices than the rest of the models, as depicted in Figure 8f.



(a) Training error: Value of p. **(b)** Training error: Value of M. **(c)** Training error: Optimal AF.

(d) Fitting and validation set. **(e)** Validation set. **(f)** Test set (forecast).

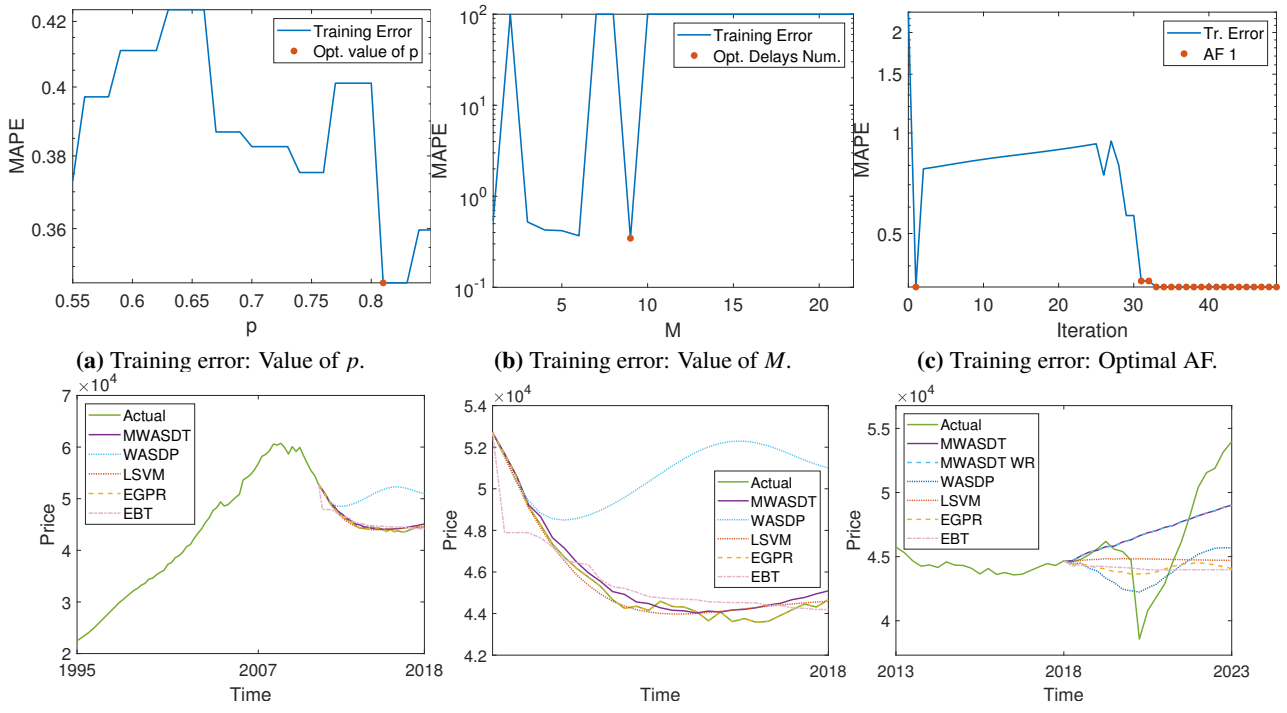
Figure 6. Training and testing the neural network models on the GDP of Italy.



(a) Training error: Value of p. **(b)** Training error: Value of M. **(c)** Training error: Optimal AF.

(d) Fitting and validation set. **(e)** Validation set. **(f)** Test set (forecast).

Figure 7. Training and testing the neural network models on the GDP of France.

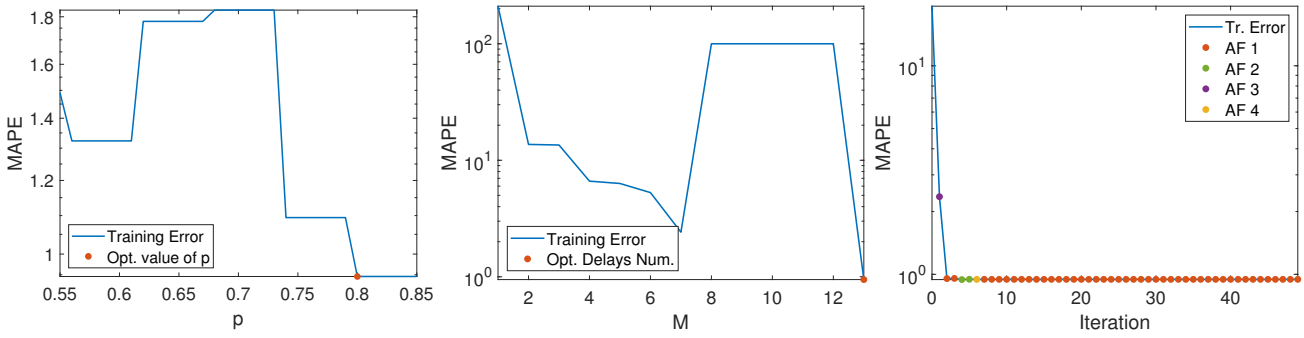


(a) Training error: Value of p. **(b)** Training error: Value of M. **(c)** Training error: Optimal AF.

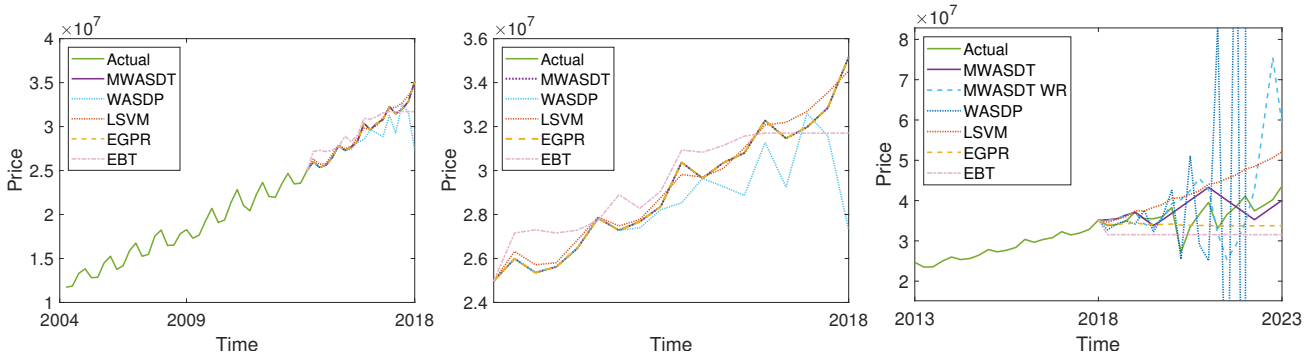
(d) Fitting and validation set. **(e)** Validation set. **(f)** Test set (forecast).

Figure 8. Training and testing the neural network models on the GDP of Greece.

Regarding India's GDP, the neural network training and test results are presented in Figure 9. Particularly, the MAPE of the validation set during training is shown in Figure 9a–9c for various ratios of the fitting and validation input sets p, various delays M for the optimal value of p, and various AFs for the optimal values of p and M, respectively. More particularly, Figure 9a shows that the optimal value of p is 0.8, Figure 9b shows that the optimal value of M is 13, and Figure 9c shows that the optimal number of hidden-layer neurons is 49. In Figure 9d and 9e, which depict the predicted prices on the validation set during training, we can see that the EBT and WASDP models are less accurate at matching the actual prices than the other models. As depicted in Figure 9f, the MWASDT model is more accurate at forecasting the actual prices than the other models.



(a) Training error: Value of p. **(b)** Training error: Value of M. **(c)** Training error: Optimal AF.



(d) Fitting and validation set. **(e)** Validation set. **(f)** Test set (forecast).

Figure 9. Training and testing the neural network models on the GDP of India.

5.2. Results discussion

The models statistics on the training and test sets for the GDPs of the U.S., U.K., Italy, France, Greece and India are shown in Table 2. It is crucial to note that the statistics presented in this table were generated using MATLAB and then double-checked using SPSS. The coefficient of determination (R^2), MAPE, symmetric MAPE (SMAPE), mean absolute error (MAE), root-mean-square error (RMSE), mean absolute scaled error (MASE) and mean directional accuracy (MDA) are the performance measures considered in our analysis. Particularly, R^2 is the proportion of the variation in the dependent variable that is predictable from the independent variable(s), SMAPE is an accuracy measure based on percentage errors, MAE is a measure of errors between paired observations expressing the same phenomenon, RMSE is a measure of the differences between values predicted by a model

and the values observed, MASE is a measure of the accuracy of forecasts, and MDA is a measure of prediction accuracy of a forecasting method and compares the forecast direction to the actual realized direction. See [31] for more details and in-depth analysis of these measures.

In the case of the U.S.'s GDP, the neural network models' statistics on the training and test sets are presented in Table 2. In the training set, EGPR has the best R^2, MWASDT has the second best, and WASDP has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, EGPR has the lowest values, MWASDT has the second lowest values, and WASDP has the highest values. Also, EGPR and MWASDT have the best MDA, and WASDP has the worst. In the test set, MWASDT has the best R^2, and EBT and MWASDT WR have the worst. On MAPE, SMAPE, MAE, RMSE and MASE, MWASDT has the lowest values, and MWASDT WR has the highest values. Also, MWASDT has the best MDA, and EBT and MWASDT have the worst. Overall, the MWASDT has the second best statistics on the training set and the best statistics on the test set.

When it comes to the U.K.'s GDP, the neural network models' statistics on the training and test sets are presented in Table 2. In the training set, EGPR has the best R^2, MWASDT has the second best and WASDP has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, EGPR has the lowest values, MWASDT has the second lowest values, and WASDP has the highest values. Also, EGPR has the best MDA, MWASDT has the second best, and LSVM has the worst. In the test set, all the statistics of MWASDT and MWASDT WR are identical. MWASDT has the best R^2, and EBT has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, MWASDT has the lowest values, and WASDP has the highest values. Also, MWASDT and LSVM have the best MDA, and WASDP has the worst. Overall, the MWASDT has the second best statistics on the training set and the best statistics on the test set.

Regarding Italy's GDP, the neural network models' statistics on the training and test sets are presented in Table 2. In the training set, EGPR has the best R^2, MWASDT has the third best, and WASDP has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, EGPR has the lowest values, and MWASDT has the highest values. Also, EGPR has the best MDA, MWASDT has the third best, and WASDP has the worst. In the test set, MWASDT WR has the best R^2, MWASDT has the third best, and EGPR has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, EGPR has the lowest values, MWASDT has the third lowest values, and MWASDT WR has the highest values. Also, MWASDT, MWASDT WR and LSVM have the best MDA, and WASDP, EGPR and EBT have the worst. Overall, the MWASDT has the third best statistics on the training set and the second best statistics on the test set.

In the case of France's GDP, the neural network models' statistics on the training and test sets are presented in Table 2. In the training set, EGPR has the best R^2, MWASDT has the second best, and WASDP has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, EGPR has the lowest values, MWASDT has the second lowest values, and WASDP has the highest values. Also, EGPR has the best MDA, MWASDT and LSVM have the second best, and WASDP has the worst. In the test set, MWASDT has the best R^2, and EBT and MWASDT WR have the worst. On MAPE, SMAPE, MAE, RMSE and MASE, MWASDT has the lowest values, and MWASDT WR has the highest values. Also, MWASDT has the best MDA, and WASDP has the worst. Overall, the MWASDT has the third best statistics on the training set and the best statistics on the test set.

Table 2. Neural network models' statistics on the GDPs of the U.S., U.K., Italy, France, Greece and India.

		Neural Network Models									
		Training Set					Test Set					
GDP	**Statistic**	**MWASDT**	**WASDP**	**LSVM**	**EGPR**	**EBT**	**MWASDT**	**WASDP**	**LSVM**	**EGPR**	**EBT**	**MWASDT WR**
U.S.	R^2	0.9963	0.1661	0.9919	0.9999	0.8870	0.6754	-6.78	0.4858	-4.0×10^{3}	-Inf	-Inf
	MAPE	0.4070	4.6594	0.5897	0.0052	2.1425	6.63	17.77	3.1653	11.17	15.34	145.6
	SMAPE	0.2035	2.4111	0.2959	0.0025	1.0642	3.1771	9.97	1.5867	6.08	8.48	Inf
	MAE	73.2	858.1	102.2	0.8837	369.9	1.6×10^{3}	4.1×10^{3}	736.7	2.6×10^{3}	3.6×10^{3}	Inf
	RMSE	91.7	1.3×10^{3}	133.3	1.1084	443.4	2.3×10^{3}	4.6×10^{3}	981.1	3.3×10^{3}	4.1×10^{3}	Inf
	MASE	0.4342	5.0934	0.6066	0.0052	2.1960	3.0237	7.71	1.3783	5.02	6.78	Inf
	MDA	1	0.7187	0.7812	1	0.8437	0.6842	0.1052	0.4210	0.1578	0.1052	0.5263
U.K.	R^2	0.9973	0.7661	0.9854	0.9999	0.8365	-0.2121	-8.1325	-0.5773	-671.2	-Inf	-0.2121
	MAPE	0.3824	2.8499	0.8684	0.0051	2.7464	5.1247	13.64	9.90	7.25	10.76	5.1247
	SMAPE	0.1909	1.4128	0.4358	0.0026	1.3569	2.4277	7.41	4.61	3.81	5.74	2.4277
	MAE	1.7×10^{3}	1.4×10^{4}	3.9×10^{3}	23.32	1.2×10^{4}	2.7×10^{4}	7.8×10^{4}	5.5×10^{4}	4.2×10^{4}	6.2×10^{4}	2.7×10^{4}
	RMSE	2.1×10^{3}	2.0×10^{4}	4.6×10^{3}	28.64	1.4×10^{4}	3.9×10^{4}	8.4×10^{4}	6.6×10^{4}	5.3×10^{4}	7.0×10^{4}	3.9×10^{4}
	MASE	0.3692	3.0271	0.8574	0.0051	2.7	1.8282	5.11	3.6377	2.7708	4.08	1.8282
	MDA	0.9688	0.8125	0.7187	1	0.8437	0.8421	0.1579	0.8421	0.2105	0.21	0.8421
Italy	R^2	0.2337	-123.5	0.3038	0.9999	-2.9825	-2.1692	-34.90	-0.984	-273.7	-8.14	-0.9225
	MAPE	3.5837	2.5196	3.2356	0.0014	2.7343	6.8643	12.47	16.37	4.7947	5.7314	17.01
	SMAPE	1.7491	1.2938	1.5837	0.0007	1.3438	3.2469	6.6946	7.3437	2.4505	2.9573	7.5803
	MAE	1.5×10^{4}	1.1×10^{4}	1.3×10^{4}	5.8581	1.1×10^{4}	2.9×10^{4}	5.6×10^{4}	7.4×10^{4}	2.1×10^{4}	2.6×10^{4}	7.7×10^{4}
	RMSE	1.7×10^{4}	1.5×10^{4}	1.6×10^{4}	7.3976	1.3×10^{4}	3.7×10^{4}	6.0×10^{4}	8.9×10^{4}	2.9×10^{4}	3.3×10^{4}	9.3×10^{4}
	MASE	7.56	5.3944	6.8173	0.0029	5.6350	2.7428	5.2340	6.8190	2.0116	2.4153	7.09
	MDA	0.7407	0.3333	0.7777	1	0.5925	0.6842	0.3684	0.6842	0.3684	0.3684	0.6842
France	R^2	0.9947	-1.623	0.9124	0.9999	0.8533	0.1795	-33.58	-0.812	-3.5×10^{3}	-Inf	-Inf
	MAPE	0.2369	4.8101	1.2691	0.0031	0.9838	6.4375	17.84	2.9038	6.4967	9.0490	41.16
	SMAPE	0.1185	2.5287	0.6290	0.0015	0.4915	3.1308	9.8938	1.4324	3.3835	4.7753	Inf
	MAE	1.3×10^{3}	2.6×10^{4}	6.9×10^{3}	17.10	5.3×10^{3}	4.0×10^{4}	1.1×10^{5}	1.8×10^{4}	4.1×10^{4}	5.7×10^{4}	Inf
	RMSE	1.7×10^{3}	4.1×10^{4}	8.2×10^{3}	22.39	7.7×10^{3}	5.1×10^{4}	1.2×10^{5}	2.5×10^{4}	5.0×10^{4}	6.4×10^{4}	Inf
	MASE	0.4162	8.5665	2.2087	0.0054	1.6924	2.8574	7.9594	1.2532	2.9316	4.0702	Inf
	MDA	0.9062	0.5937	0.9062	1	0.7812	0.5263	0.1578	0.4210	0.2105	0.2105	0.4210
Greece	R^2	0.9609	-20.06	0.9718	0.9999	0.4854	-5.94	-10.10	-5.8×10^{3}	-223.5	-2.2×10^{3}	-5.94
	MAPE	0.7804	12.01	0.6098	0.0051	1.4565	5.54	6.24	6.44	6.79	7.09	5.54
	SMAPE	0.3882	5.5702	0.3044	0.0025	0.7333	2.6897	3.2493	3.3208	3.5406	3.6995	2.6897
	MAE	348.26	5.1×10^{3}	272.65	2.2732	683.24	2.5×10^{3}	3.0×10^{3}	3.1×10^{3}	3.3×10^{3}	3.4×10^{3}	2.5×10^{3}
	RMSE	407.17	6.1×10^{3}	338.40	3.4009	1.0×10^{3}	3.4×10^{3}	3.8×10^{3}	4.3×10^{3}	4.4×10^{3}	4.6×10^{3}	3.4×10^{3}
	MASE	0.8034	12.26	0.6290	0.0052	1.5763	1.9282	2.3339	2.3869	2.5383	2.6474	1.9282
	MDA	0.7777	0.4444	0.8148	1	0.5185	0.6315	0.4210	0.4736	0.4210	0.4210	0.6315
India	R^2	1	-0.119	0.9771	0.9999	0.4075	-1.715	-0.111	-0.965	-397.3	-Inf	0.0334
	MAPE	2.1×10^{-11}	3.2801	1.1351	0.0029	3.6391	8.89	Inf	17.29	9.15	13.89	19.86
	SMAPE	1.1×10^{-11}	1.7469	0.5653	0.0014	1.8082	4.1371	38.76	7.6581	4.7875	7.5073	8.7166
	MAE	6.3×10^{-6}	1.1×10^{6}	3.4×10^{5}	918.2	1.1×10^{6}	3.0×10^{6}	Inf	6.2×10^{6}	3.4×10^{6}	5.2×10^{6}	7.4×10^{6}
	RMSE	7.9×10^{-6}	2.1×10^{6}	4.1×10^{5}	1.2×10^{3}	1.3×10^{6}	4.2×10^{6}	Inf	7.5×10^{6}	4.4×10^{6}	5.9×10^{6}	1.1×10^{7}
	MASE	6.9×10^{-12}	1.1765	0.3741	0.0010	1.1627	1.0454	Inf	2.1319	1.1759	1.7906	2.5326
	MDA	1	0.75	1	1	0.6875	0.6315	0.5789	0.7368	0.5263	0.3157	0.7368

When it comes to Greece's GDP, the neural network models' statistics on the training and test sets are presented in Table 2. In the training set, EGPR has the best R^2, MWASDT has the third best, and WASDP has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, EGPR has the lowest values, MWASDT has the third lowest values, and WASDP has the highest values. Also, EGPR has the best MDA, MWASDT has the third best, and WASDP has the worst. In the test set, all the statistics of MWASDT and MWASDT WR are identical. MWASDT has the best R^2, and LSVM has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, MWASDT has the lowest values, and EBT has the highest values. Also, MWASDT has the best MDA, and WASDP, EGPR and EBT have the worst. Overall, the MWASDT has the third best statistics on the training set and the best statistics on the test set.

Regarding India's GDP, the neural network models' statistics on the training and test sets are pre-

sented in Table 2. In the training set, MWASDT has the best R^2, and WASDP has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, MWASDT has the lowest values, and EBT has the highest values. Also, EGPR, MWASDT and LSVM have the best MDA, and EBT has the worst. In the test set, MWASDT WR has the best R^2, MWASDT has the fourth best, and EBT has the worst. On MAPE, SMAPE, MAE, RMSE and MASE, MWASDT has the lowest values, and WASDP has the highest values. Also, LSVM and MWASDT WR have the best MDA, MWASDT has the second best, and EBT has the worst. Overall, the MWASDT has the best statistics on the training and test sets.

In the situations of the U.S., U.K. and France, we can see that these nations have steadily rising GDPs and have fully recovered from the 2007–2008 and COVID–19 financial crises. We can observe that whereas Italy and Greece have entirely recovered from the COVID-19 financial crisis, they have not yet fully recovered from the 2007–2008 financial crisis. India's GDP is extremely volatile, and while it entirely recovered from the financial crisis of 2007–2008, it has not yet fully recovered from the COVID–19 financial crisis. We are able to forecast the rising trend in these countries' GDPs using neural networks, but we are unable to predict with high accuracy how the COVID-19 financial crisis will impact GDP. The tests' execution aids in these participating nations' economic forecasts. In order to predict a country's future economic activity, economists use the most recent data available. Although the specifics of these reports vary, their fundamental methodology is the same: They forecast an economy's growth using economic indicators and models. Most central banks, global rating agencies, and organizations like the International Monetary Fund (IMF) carry out this kind of examination. However, they are crucial for investors because they aid in their decision as to whether or not to invest in a particular nation.

Altogether, the MWASDT model did a great job resolving forecasting issues, which is consistent with the information shown in Table 2. The GDPs of the U.S., Italy, France, and India, where the accuracy of the MWASDT on the test set grew considerably in comparison to MWASDT WR, clearly demonstrated the effectiveness of (3.2). The performance of the MWASDT compared to traditional neural network approaches is quite competitive or even better when forecasting tasks are considered as regression problems.

6. Conclusions

This paper introduced a novel 3-layer feed-forward WASD neural network for time-series forecast, termed MWASDT. The findings of six experiments on GDP forecast show that the MWASDT model outperforms the WASDP model and some of the most cutting-edge regression models available through MATLAB's regression learner app. As a result, it has been demonstrated that the MWASDT model is an excellent substitute to MATLAB's regression neural network models for forecasting the GDP. It is important to mention that due to limitations placed by the WDD procedure, which is used by the MWASDT algorithm, only time-series data can be used as input to train and test the MWASDT neural network model. Another limitation of the MWASDT model is that it was only designed to be used for time-series forecasting tasks. Its proper adjustment and application to diverse time-series forecasting issues across many scientific domains will therefore be the subject of future research.

Use of AI tools declaration

The authors declare they have not used artificial intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by the Ministry of Science and Higher Education of the Russian Federation (Grant No. 075-15-2022-1121).

Conflict of interest

The authors declare no conflict of interest.

References

1. G. N. Kouziokas, A new W-SVM kernel combining PSO-neural network transformed vector and Bayesian optimized SVM in GDP forecasting, *Eng. Appl. Artif. Intel.*, **92** (2020), 103650. https://doi.org/10.1016/j.engappai.2020.103650
2. O. Cepni, I. E. Güney, N. R. Swanson, Nowcasting and forecasting GDP in emerging markets using global financial and macroeconomic diffusion indexes, *Int. J. Forecasting*, **35** (2019), 555–572. https://doi.org/10.1016/j.ijforecast.2018.10.008
3. T. E. Simos, V. N. Katsikis, S. D. Mourtas, P. S. Stanimirović, Unique non-negative definite solution of the time-varying algebraic Riccati equations with applications to stabilization of LTV systems, *Math. Comput. Simulat.*, **202** (2022), 164–180. https://doi.org/10.1016/j.matcom.2022.05.033
4. S. D. Mourtas, V. N. Katsikis, C. Kasimis, Feedback control systems stabilization using a bio-inspired neural network, *EAI Endorsed Trans. AI Robotics*, **1** (2022), 1–13. https://doi.org/10.4108/airo.v1i.17
5. N. Premalatha, A. V. Arasu, Prediction of solar radiation for solar systems by using ANN models with different back propagation algorithms, *J. Appl. Res. Technol.*, **14** (2016), 206–214. https://doi.org/10.1016/j.jart.2016.05.001
6. S. X. Lv, L. Wang, Multivariate wind speed forecasting based on multi-objective feature selection approach and hybrid deep learning model, *Energy*, **263** (2023), 126100. https://doi.org/10.1016/j.energy.2022.126100
7. C. Huang, X. Jia, Z. Zhang, A modified back propagation artificial neural network model based on genetic algorithm to predict the flow behavior of 5754 aluminum alloy, *Materials*, **11** (2018), 855. https://doi.org/10.3390/ma11050855
8. S. Gayathri, A. K. Krishna, V. P. Gopi, P. Palanisamy, Automated binary and multiclass classification of diabetic retinopathy using Haralick and multiresolution features, *IEEE Access*, **8** (2020), 57497–57504. https://doi.org/10.1109/ACCESS.2020.2979753
9. L. Chen, Z. Huang, Y. Li, N. Zeng, M. Liu, A. Peng, et al., Weight and structure determination neural network aided with double pseudoinversion for diagnosis of flat foot, *IEEE Access*, **7** (2019), 33001–33008. https://doi.org/10.1109/ACCESS.2019.2903634

10. M. R. Daliri, A hybrid automatic system for the diagnosis of lung cancer based on genetic algorithm and fuzzy extreme learning machines, *J. Medical Syst.*, **36** (2012), 1001–1005. https://doi.org/10.1007/s10916-011-9806-y

11. R. J. S. Raj, S. J. Shobana, I. V. Pustokhina, D. A. Pustokhin, D. Gupta, K. Shankar, Optimal feature selection-based medical image classification using deep learning model in internet of medical things, *IEEE Access*, **8** (2020), 58006–58017. https://doi.org/10.1109/ACCESS.2020.2981337

12. S. D. Mourtas, V. N. Katsikis, Exploiting the Black-Litterman framework through error-correction neural networks, *Neurocomputing*, **498** (2022), 43–58. https://doi.org/10.1016/j.neucom.2022.05.036

13. S. D. Mourtas, A weights direct determination neuronet for time-series with applications in the industrial indices of the federal reserve bank of St. Louis, *J. Forecasting*, **14** (2022), 1512–1524. https://doi.org/10.1002/for.2874

14. S. X. Lv, L. Peng, H. Hu, L. Wang, Effective machine learning model combination based on selective ensemble strategy for time series forecasting, *Inf. Sci.*, **612** (2022), 994–1023. https://doi.org/10.1016/j.ins.2022.09.002

15. Y. Zhang, D. Guo, Z. Luo, K. Zhai, H. Tan, CP-activated WASD neuronet approach to Asian population prediction with abundant experimental verification, *Neurocomputing*, **198** (2016), 48–57. https://doi.org/10.1016/j.neucom.2015.12.111

16. Y. Zhang, Z. Xue, M. Xiao, Y. Ling, C. Ye, Ten-Quarter Projection for Spanish Central Government Debt via WASD Neuronet, In: *International Conference on Neural Information Processing*, Springer, 2017. 893–902.

17. F. Groes, P. Kircher, I. Manovskii, The U-shapes of occupational mobility, *Rev. Econ. Stud.*, **82** (2015), 659–692. https://doi.org/10.1093/restud/rdu037

18. I. N. Generalao, Measuring the telework potential of jobs: Evidence from the international standard classification of occupations, *Philippine Rev. Econ.*, **58** (2021), 92–127. `https://doi.org/10.37907/5erp1202jd`

19. D. Lagios, S. D. Mourtas, P. Zervas, G. Tzimas, A weights direct determination neural network for international standard classification of occupations, *Mathematics*, **11** (2023), 629. https://doi.org/10.3390/math11030629

20. J. Garnitz, R. Lehmann, K. Wohlrabe, Forecasting GDP all over the world using leading indicators based on comprehensive survey data, *Appl. Econ.*, **51** (2019), 5802–5816. https://doi.org/10.1080/00036846.2019.1624915

21. M. Marcellino, M. Porqueddu, F. Venditti, Short-term GDP forecasting with a mixed-frequency dynamic factor model with stochastic volatility, *J. Bus. Econ. Stat.*, **34** (2016), 118–127. https://doi.org/10.1080/07350015.2015.1006773

22. C. Liu, W. Xie, T. Lao, Y. Yao, J. Zhang, Application of a novel grey forecasting model with time power term to predict China's GDP, *Grey Syst. Theory Appl.*, **11** (2021), 343–357. https://doi.org/10.1108/GS-05-2020-0065

23. J. Yoon, Forecasting of real GDP growth using machine learning models: Gradient boosting and random forest approach, *Comput. Econ.*, **57** (2021), 247–265. https://doi.org/10.1007/s10614-020-10054-w

24. H. H. Kim, N. R. Swanson, Methods for backcasting, nowcasting and forecasting using factor-MIDAS: With an application to Korean GDP, *J. Forecasting*, **37** (2018), 281–302. https://doi.org/10.1002/for.2499

25. A. Richardson, T. van Florenstein Mulder, T. Vehbi, Nowcasting GDP using machine-learning algorithms: A real-time assessment, *Int. J. Forecasting*, **37** (2021), 941–948. https://doi.org/10.1016/j.ijforecast.2020.10.005

26. Y. Zhang, D. Chen, C. Ye, *Deep Neural Networks: WASD Neuronet Models, Algorithms, and Applications*, CRC Press: Boca Raton, FL,USA, 2019.

27. T. E. Simos, V. N. Katsikis, S. D. Mourtas, A multi-input with multi-function activated weights and structure determination neuronet for classification problems and applications in firm fraud and loan approval, *Appl. Soft Comput.*, **127** (2022), 109351. https://doi.org/10.1016/j.asoc.2022.109351

28. T. E. Simos, S. D. Mourtas, V. N. Katsikis, Time-varying Black-Litterman portfolio optimization using a bio-inspired approach and neuronets, *Appl. Soft Comput.*, **112** (2021), 107767. https://doi.org/10.1016/j.asoc.2021.107767

29. Y. Zhang, X. Yu, L. Xiao, W. Li, Z. Fan, W. Zhang, Weights and structure determination of articial neuronets, In: *Self-Organization: Theories and Methods*, New York, NY, USA: Nova Science, 2013.

30. G. P. Zhang, D. M. Kline, Quarterly time-series forecasting with neural networks, *IEEE T. Neur. Network.*, **18** (2007), 1800–1814. https://doi.org/10.1109/TNN.2007.896859

31. A. Tharwat, Classification assessment methods, *Appl. Comput. Inf.*, **17** (2020), 168–192. https://doi.org/10.1016/j.aci.2018.08.003

AIMS Press

© 2023 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0)