

PAPER • OPEN ACCESS

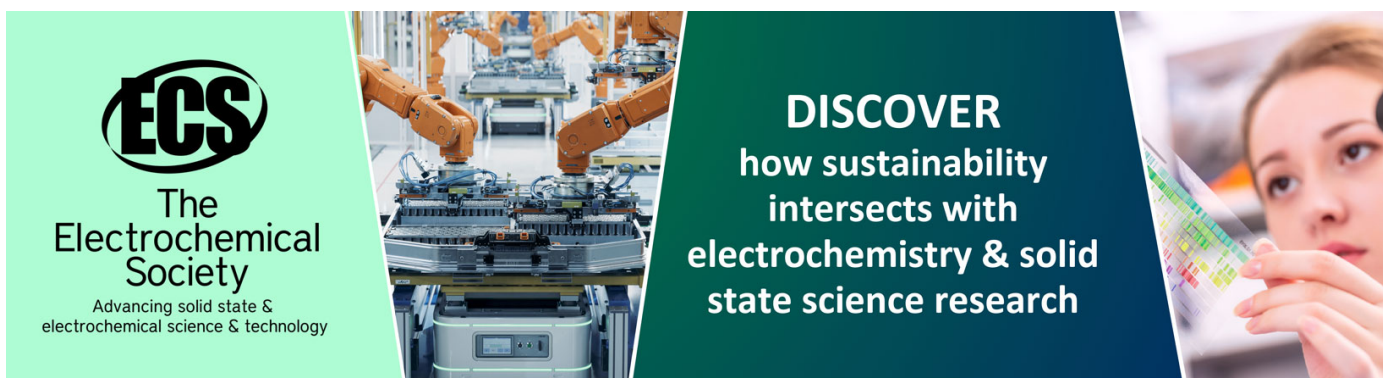
Generalized space time autoregressive (gstar)- artificial neural network (ann) model with multilayer feedforward networks architecture

To cite this article: D R S Saputro *et al* 2019 *IOP Conf. Ser.: Earth Environ. Sci.* **243** 012039

View the [article online](#) for updates and enhancements.

You may also like

- [The correlation of electrical conductivity with the microstructure of \$\text{Ge}_2\text{Sb}_2\text{Te}_5\$ thin films alloyed with Sn](#)
Qixun Yin and Leng Chen
- [A phase-change thin film-tuned photonic crystal device](#)
Longju Liu, Russell Mahmood, Le Wei et al.
- [A Hybrid GSTARX-Jordan RNN Model for Forecasting Space-Time Data with Calendar Variation Effect](#)
F Hikmawati, Suhartono and D D Prastyo



ECS
The
Electrochemical
Society
Advancing solid state &
electrochemical science & technology

DISCOVER
how sustainability
intersects with
electrochemistry & solid
state science research

Generalized space time autoregressive (gstar)-artificial neural network (ann) model with multilayer feedforward networks architecture

D R S Saputro¹, I M Putri², Sutanto³, N H Noor⁴, P Widyaningsih⁵

^{1,2,3,4,5} Universitas Sebelas Maret, Solo, Indonesia

E-mail : dewiretnosaris@staff.uns.ac.id

Abstract. The GSTAR model is one linear space time model used to model time series data with inter-location linkages. One of the weaknesses of the GSTAR model is that the model has not been able to capture any nonlinear patterns that may arise. The ANN is one model of nonlinear artificial intelligence that has a flexible functional form and is a supervised engine learning that provides a good framework to represent a relationship in time series data. Due to the advantages of such ANN, it can be combined with GSTAR model. In this research conducted study of GSTAR-ANN model and its network architecture. The model is constructed in 3 layers namely input, hidden and output. The GSTAR model is used as input in the training process. Each input will receive an input signal and forward the signal to the hidden layer, then to the output layer. The GSTAR-ANN model has network architecture with one neuron unit at the output layer, p input neuron, q neuron in hidden layer (multilayer feedforward networks).

1. Introduction

Time series data is a sequential time-based data [1,2]. The models for the time series data created are Autoregressive (AR) which means that the current state is influenced by the previous state, Moving Average (MA) which means that the current state is affected by errors in the past, and the combination of AR and MA (ARMA). The influence can be linear or nonlinear so that the model also consists of linear and nonlinear models. In some cases, there is data that is not only influenced by time but also influenced by the location conditions around (spatial influences). It called the space-time model (spatio-temporal model). Among the space-time models are the space time autoregressive model (STAR) and the Generalized STAR (GSTAR) model [2]. The GSTAR model was developed from the STAR model. A more flexible model as a generalization of the STAR model is the general space-time autoregressive model (GSTAR). Unlike the STAR model, the GSTAR model does not require that parameter values be the same for all locations. The GSTAR model is more realistic because in reality there are more models with different model parameters for different locations. Theoretical studies related to the asymptotic nature of the GSTAR model parameters and the determination of the weights between locations are given by [3]. The results of a research by [4, 5, 6] show that GSTAR has good performance. Alternatively, there are other models that reinforce time-series or space-time models in increasing interest as a faster and more accurate forecasting tool to predict trends and data patterns. Techniques in predicting trends and data patterns compete amongst linear and nonlinear models [1].



One of the properties of this model is having the ability to do nonlinear mapping. The model is an artificial neural network (ANN) that has two important aspects, namely the architectural aspect (also called taxonomy or structure) and aspects of training algorithms [7], [8], [9]. Forecasting with ANN can be done, given the ability of neural networks to remember and make generalizations of what has been done before [10, 11]. ANN can be classified as a semi-parametric method [12], [13], [14]. In addition, ANN is an extension of nonlinear regression and discriminant models, data reduction models and nonlinear dynamic systems [13]. In its application, ANN contains a limited number of parameters (weight). The problem that still concerns the researchers is how to determine the best NN model (optimal number of parameters) which includes the determination of significant input units and the number of hidden units [15]. There are several methods that have been used such as pruning algorithm, network information criteria (NIC), regulation, and cross-validation. However, these methods have not provided a guarantee of optimal modeling so that the issue remains a topic of continuous review. Meanwhile, the approach based on statistical concepts to obtain optimal ANN model has been introduced by [16], [10], and [12]. Based on these descriptions, compared to other models, ANN has good adaptive capabilities, learning, and nonstationary signal rejection capability [17]. Due to the advantages of such ANN, it can be combined with GSTAR model. Some application of the model has been done by [18, 19] as well as [20]. Related to the improvement of GSTAR model performance combined with ANN, in this study conducted a study of GSTAR-ANN model of network architecture and algorithm.

2. Method

This research is a research of theoretical study in determining network structure of GSTAR-ANN. The basic material on which this research is the scientific work of several experts presented and published in journals, bulletins and books. The research method used is literature study from several literatures, by studying the scientific papers that have been collected. The research step is to derive GSTAR model and prove its parameter estimation, ANN and backpropagation algorithm, network architecture, activation function, training process on ANN and GSTAR-ANN network architecture.

3. Results and Discussion

3.1. GSTAR Model

According to [1], $STAR(p_{\lambda_1, \lambda_2, \dots, \lambda_p})$ model is a special form of the STARMA model whose moving order average is 0 or $q = 0$. Model $STAR(p_{\lambda_1, \lambda_2, \dots, \lambda_p})$ is written as

$$Z_t = \sum_{k=1}^p \sum_{l=0}^{\lambda_k} \phi_{kl} W^{(l)} Z_{t-k} + \varepsilon_t, \quad (1)$$

where k is the autoregressive order, l is the spatial order, ϕ_{kl} is the autoregressive parameter, $W^{(l)}$ is a weighted matrix and ε_t is an error. $GSTAR(p, \lambda_k)$ is the development of $STAR(p_{\lambda_1, \lambda_2, \dots, \lambda_p})$ expressed as

$$Z_{i,t} = \sum_{i=1}^p \left[\sum_{k=1}^p \phi_{k0}^{(i)} Z_{i,t-k} + \sum_{l=1}^{\lambda_k} W_{ij}^{(l)} \phi_{kl}^{(i)} Z_{i,t-k} \right]$$

with $Z_{i,t}$ is the observed value at location i and time t , $W_{ij}^{(l)}$ is the weighting matrix at location i and j and the spatial order l , $\phi_{kl}^{(i)}$ is the diagonal matrix of the autoregressive order parameter spatial l and autoregressive order k at the i -location, and $\varepsilon_{i,t}$ is error, $\phi_{kl}^{(i)}$ is written as

$$\phi_{kl}^{(i)} = \begin{bmatrix} \phi_{kl}^{(1)} & 0 & \dots & 0 \\ 0 & \phi_{kl}^{(2)} & \ddots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \phi_{kl}^{(N)} \end{bmatrix}$$

The value of the autoregressive parameter at each location can be assumed by the least squares method, Modeling with GSTAR (p, λ_k) is done by adopting the Box-Jenkins stage ie model identification, parameter estimation, and model validation. The GSTAR (p, λ_k) model is a special form of the VAR (p) model that has a weighted value for each location. The VAR (p) model is expressed as

$$Z_t = \phi_t Z_{t-1} + \dots + \phi_t Z_{t-k} + \varepsilon_t,$$

The form VAR (p) of GSTAR (p, λ_k) model can also be written as

$$Z_t = (\phi_{k0}^{(i)} + \phi_{k0}^{(i)} W^{(l)}) Z_{t-k} + \varepsilon_t,$$

and can be represented in the VAR (p) model as

$$Z_t = \phi Z_{t-p} + \varepsilon_t \quad (2)$$

with $\phi = (\phi_{k0}^{(i)} + \phi_{kl}^{(i)} W^{(l)})$, i is the location, k is the autoregressive order and l is the spatial order. The autoregressive order (p) of the GSTAR (p, λ_k) model can be obtained from the order of the VAR (p) model having the smallest AIC value, and the spatial order used is order 1.

3.2. Parameter Estimation of GSTAR Model

According to [3] the parameter estimate $\hat{\phi}$ in GSTAR model can be done by the least squares method by minimizing the sum of squares of error. The GSTAR (1) model can be written in the following matrix.

$$\begin{aligned} Z(t) &= \begin{pmatrix} \phi_{k0}^{(1)} & 0 & \dots & 0 \\ 0 & \phi_{k0}^{(2)} & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & \phi_{k0}^{(N)} \end{pmatrix} \begin{pmatrix} Z_1(1) \\ Z_1(2) \\ \vdots \\ Z_N(t-k) \end{pmatrix} \\ &\quad + \begin{pmatrix} \phi_{kl}^{(1)} & 0 & \dots & 0 \\ 0 & \phi_{kl}^{(2)} & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & \phi_{kl}^{(N)} \end{pmatrix} \begin{pmatrix} 0 & W_{12} & \dots & W_{1N} \\ W_{21} & 0 & \dots & W_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ W_{N1} & W_{N2} & \dots & 0 \end{pmatrix} \begin{pmatrix} Z_1(1) \\ Z_1(2) \\ \vdots \\ Z_N(t-k) \end{pmatrix} + \begin{pmatrix} \varepsilon_1(t) \\ \varepsilon_2(t) \\ \vdots \\ \varepsilon_N(t) \end{pmatrix} \\ &= \begin{pmatrix} \phi_{k0}^{(1)} & Z_1(1) \\ \phi_{k0}^{(2)} & Z_1(2) \\ \vdots & \vdots \\ \phi_{k0}^{(N)} & Z_N(t-k) \end{pmatrix} + \begin{pmatrix} \phi_{kl}^{(1)} & 0 & \dots & 0 \\ 0 & \phi_{kl}^{(2)} & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & \phi_{kl}^{(N)} \end{pmatrix} \begin{pmatrix} W_{12}Z_1(2) + \dots + W_{1N}Z_N(t-k) \\ W_{21}Z_1(1) + \dots + W_{2N}Z_N(t-k) \\ \vdots \\ W_{N1}Z_1(1) + \dots + W_{N-1N-1}Z_1(t-k) - 1 \end{pmatrix} \\ &\quad + \begin{pmatrix} \varepsilon_1(t) \\ \varepsilon_2(t) \\ \vdots \\ \varepsilon_N(t) \end{pmatrix} \end{aligned}$$

$$\text{with } V_i(t) = \sum_{j=1}^N W_{ij} Z_j(t), \quad (3)$$

$$Z(t) = \begin{pmatrix} \phi_{k0}^{(1)} & Z_1(1) \\ \phi_{k0}^{(2)} & Z_1(2) \\ \vdots & \vdots \\ \phi_{k0}^{(N)} & Z_N(t-k) \end{pmatrix} + \begin{pmatrix} \phi_{kl}^{(1)} & 0 & \dots & 0 \\ 0 & \phi_{kl}^{(2)} & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & \phi_{kl}^{(N)} \end{pmatrix} \begin{pmatrix} V_1(1) \\ V_1(2) \\ \vdots \\ V_N(t-k) \end{pmatrix} + \begin{pmatrix} \varepsilon_1(t) \\ \varepsilon_2(t) \\ \vdots \\ \varepsilon_N(t) \end{pmatrix} \quad (4)$$

For the overall location, equation (4) can be written in the linear regression model i.e.

$$Z = Z^* \phi + \varepsilon$$

$$\begin{pmatrix} Z_1(1) \\ Z_1(2) \\ \vdots \\ Z_1(t) \\ \vdots \\ Z_N(1) \\ Z_N(2) \\ \vdots \\ Z_N(t) \end{pmatrix} = \begin{pmatrix} Z_1(1) & V_1(1) & \dots & 0 \\ Z_1(2) & V_1(2) & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ Z_1(t-1) & V_1(t-1) & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & Z_N(1) \\ 0 & 0 & \dots & Z_N(2) \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & Z_N(t-1) \end{pmatrix} \begin{pmatrix} \phi_{k0}^{(1)} \\ \phi_{k0}^{(2)} \\ \phi_{k0}^{(3)} \\ \vdots \\ \phi_{k0}^{(N)} \\ \phi_{kl}^{(1)} \\ \phi_{kl}^{(2)} \\ \vdots \\ \phi_{kl}^{(N)} \end{pmatrix} + \begin{pmatrix} \varepsilon_1(t) \\ \varepsilon_2(t) \\ \vdots \\ \varepsilon_N(t) \end{pmatrix}$$

$$\text{with } \mathbf{Z} = \begin{pmatrix} Z_{1,1} \\ Z_{1,2} \\ \vdots \\ Z_{1,t} \\ \vdots \\ Z_{i,1} \\ Z_{i,2} \\ \vdots \\ Z_{N,t} \end{pmatrix}, \mathbf{Z}^* = \begin{pmatrix} Z_{1,1} & V_{1,1} & \dots & 0 \\ Z_{1,2} & V_{1,2} & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ Z_{1,t-1} & V_{1,t-1} & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & V_{i,1} \\ 0 & 0 & \dots & V_{i,2} \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & V_{N,t} \end{pmatrix}, \boldsymbol{\phi} = \begin{pmatrix} \phi_{k0}^{(1)} \\ \phi_{k0}^{(2)} \\ \phi_{k0}^{(3)} \\ \vdots \\ \phi_{k0}^{(N)} \\ \phi_{kl}^{(1)} \\ \phi_{kl}^{(2)} \\ \vdots \\ \phi_{kl}^{(i)} \end{pmatrix}, \text{ dan } \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_{1,t} \\ \varepsilon_{2,t} \\ \vdots \\ \varepsilon_{N,t} \end{pmatrix}$$

Estimation of parameters by least squares method by minimizing the sum of squares of errors. The error in the GSTAR model is written as $\boldsymbol{\varepsilon} = (\mathbf{Z} - \mathbf{Z}^* \boldsymbol{\phi})$ so that the sum of the error quadrants (in the matrix) is

$$\begin{aligned} S_*(\boldsymbol{\phi}^i) &= \sum_{i=1}^N \boldsymbol{\varepsilon}_i \boldsymbol{\varepsilon}_i' = (\mathbf{Z}_i - \mathbf{Z}_i^* \boldsymbol{\phi}^{(i)})' (\mathbf{Z}_i - \mathbf{Z}_i^* \boldsymbol{\phi}^{(i)}) \\ &= \mathbf{Z}_i' \mathbf{Z}_i - 2 \boldsymbol{\phi}^{i'} \mathbf{Z}_i^* \mathbf{Z}_i + \boldsymbol{\phi}^{i'} \mathbf{Z}_i^* \mathbf{Z}_i \boldsymbol{\phi}^i \\ \text{or} \\ \hat{\boldsymbol{\phi}}^i &= (\mathbf{Z}_i^{*'} \mathbf{Z}_i) (\mathbf{Z}_i^{*'} \mathbf{Z}_i^*)^{-1} \end{aligned} \tag{5}$$

3.3. ANN and backpropagation algorithm

In ANN techniques are in trend for fitting the models in numerous sectors these days. Forecasting is one of the places where any technique may not be appeared as a de facto for a particular model ([21]). ANN is one of the artificial representations of the human brain that always tries to simulate the learning process in the human brain that has characteristics similar to the neural network of biology ([5]). In general, biological neural networks can be shown in Figure 1.

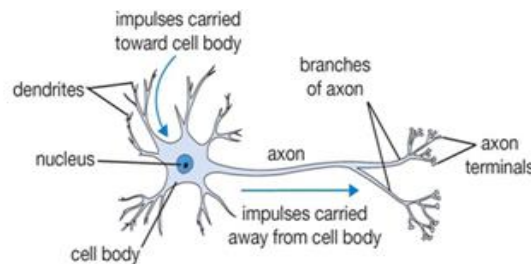


Figure 1. Neural network biology

Artificial neural network (ANN) is an implementation of artificial intelligence technology (artificial intelligence) and developed as generalization mathematical model of the neural network of biology. Artificial neural network architecture is formed as a generalization of the mathematical model of biological neural networks assuming that information processing occurs in many simple elements (neurons), signals are sent between neurons through connectors, interconnected neurons have weights that will amplify or weaken signals, and to determine the output, each neuron uses an activation function including a sigmoid that is imposed on the number of inputs received. The magnitude of this output is then compared to a threshold. ANN is characterized by a pattern of connections between neurons called architecture, a method of weighting each connection (called training or learning, algorithm) and its activation function. In Figure 2 we show the mathematical model with ANN with weights expressed with w_i , bias with b .

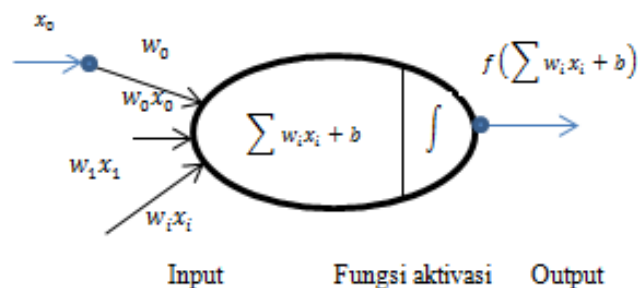


Figure 2. ANN Model

ANN can be classified into various types based on the architecture, the pattern of relationships between neurons, and training algorithms, namely how to determine the value of weight on the liaison. One of the training algorithms is backpropagation. The main purpose of using backpropagation is to get a balance between the recognition of proper training patterns and good responses to similar patterns (test data). Backpropagation algorithm uses output errors to change the value of the weights in the backward direction. To get this error, the forward propagation stage must be done first. At the time of advanced propagation, neurons are activated by using certain activation functions, among them with sigmoid activation function. Early weight selection greatly affects the neural network in reaching the global minimum of error values. By Man-Chung et al. has used multiple linear regression well for initial weighting initialization and accelerates the training process toward convergence. Therefore, in this study used backpropagation variations are used to accelerate training by modifying the weights. In paper [22] was applied backpropagation to neural networks and showed better results than previously known learning algorithms. Thus, it was possible to solve problems that were previously deemed as unsolvable. After that, it has been used to train neural networks to solve a vast number of problems.

3.4. Network Architecture

There are several model structures of artificial neural networks and one of them is a feedforward neural network. Feedforward neural network is one of the neural network models that is widely used in various fields, especially in time series data forecasting. This model is commonly called multilayer perceptron (MLP). The architecture of this model consists of one layer of input, one or more hidden layers, and an output layer. In this study the FFNN model for multivariate time series with one developed response was adapted from GSTAR model. Simply put, the ANN has a feedforward structure with no loop characteristic where the signal moves from the input layer and passes through the hidden layer and then to the output layer. The network architecture used is multilayer feedforward networks consisting of input, hidden and output layers. In the training process, data records are used as training data. Therefore, it is necessary to specify the amount of period with fluctuating data, this period is set intuitively ([13]). For example, artificial neural networks that have feedforward structures are single-layer perceptron, multilayer perceptron, radial-basis function networks and backpropagation neural networks.

3.5. Activation function

There are several model structures of artificial neural networks and one of them is a feedforward neural network. Feedforward neural network is one of the neural network model that is widely used in various fields, especially in time series data forecasting. This model is commonly called multilayer perceptron (MLP). In backpropagation, the activation function used must be eligible: continuous, differentiable and monotonous does not descend ([8, 11]). One of the eligible activation functions is the unipolar sigmoid function (Figure 3). This activation function is used in the hidden layer. Here is a unipolar sigmoid function.

$$y = \left(\frac{1}{1 + e^{-x}} \right) \quad (6)$$

In the output layer is also used the same function as the activation function. The activation function is an activating function that activates each neuron used on a network.

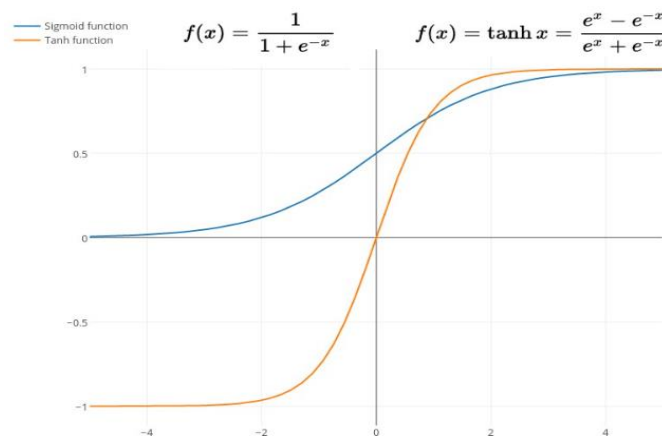


Figure 3. Unipolar sigmoid function

3.6. Training process

It is basically the working principle of ANN by dividing the data to be used for the prediction calculation process, ie some data for the training process and the other for testing. There are several algorithms to conduct the training process on ANN, among them with backpropagation. Backpropagation algorithm consists of three phases. The first phase is feedforward input pattern training. The second phase is the backward phase, the error that occurs is the difference between the network output with the target. The error is propagated backwards, starting from the line corresponding to the unit output of the hidden unit. The third phase is to modify the weight to reduce errors that occur. All three phases are repeated until the termination conditions are met. Generally the terminating conditions used are the number of iterations or errors ([8, 9, 11]). Iterations will stop if the number of iterations performed has exceeded the maximum number of iterations specified, or if the error that occurs is smaller than the specified tolerance limit. The main purpose of backpropagation is to get a balance between training pattern recognition and a good response to other similar patterns.

Backpropagation algorithm uses the minimum point search method to find the weights with minimum errors. The output error is used to change the weight value in the reverse direction. The magnitude of the error at the nth iteration of all the neurons in a single layer is

$$E = 0,5 \sum_{k=1}^m (t_k(n) - y_k(n))^2 \quad (7)$$

$$\frac{\partial E(n)}{\partial w_{kj}(n)} = \frac{\partial}{\partial w_{kj}(n)} 0,5 \sum_{k=1}^m (t_k(n) - y_k(n))^2 \quad (8)$$

Based on equation (8) we get the gradient of the error function to the weights w_{kj} ie

$$\frac{\partial E(n)}{\partial w_{kj}(n)} = \frac{\partial}{\partial w_{kj}(n)} 0,5 \sum_{k=1}^m (t_k(n) - y_k(n))^2$$

$$\begin{aligned}
&= \frac{\partial}{\partial w_{kj}(n)} 0,5 \left(t_k(n) - f(y_{net_k}(n)) \right)^2 \\
&= \frac{\partial}{\partial w_{kj}(n)} 0,5 \left(t_k(n) - f(y_{net_k}(n)) \right)^2 \\
&= \frac{\partial}{\partial w_{kj}(n)} \left(t_k(n) - f(y_{net_k}(n)) \right) f'(y_{net_k}(n)) \\
&= - \frac{\partial}{\partial w_{kj}(n)} \left(t_k(n) - f(y_{net_k}(n)) \right) f'(y_{net_k}(n)) \\
&= \left(t_k(n) - f(y_{net_k}(n)) \right) f'(y_{net_k}(n)) \frac{\partial}{\partial w_{kj}(n)} (y_{net_k}) \\
&= -(t_k(n) - y_k(n)) f'(y_{net_k}) z_j
\end{aligned} \tag{9}$$

We obtain the weight changes from the neurons in the output layer to the neurons in the hidden layer

$$\Delta w_{kj} = -\alpha (t_k(n) - y_k(n)) f'(y_{net_k}) z_j = \alpha \delta_k \tag{10}$$

with

$$\delta_k = (t_k(n) - y_k(n)) f'(y_{net_k}) \tag{11}$$

The weight change of the bias unit in the hidden layer is

$$\Delta w_{k0} = \alpha \delta_k \tag{12}$$

After the weights are modified from the k-neuron in the output layer to the j-neuron in the hidden layer, we then modify the weights of the jth neuron in the hidden layer to the *i*-th neuron of the input layer. We obtained the gradient of the error function against the weights v_{ij} i.e.

$$\begin{aligned}
\frac{\partial E(n)}{\partial v_{ji}(n)} &= \frac{\partial}{\partial v_{ji}(n)} 0,5 \sum_{k=1}^m (t_k(n) - y_k(n))^2 \\
\frac{\partial E(n)}{\partial v_{ji}(n)} &= \frac{\partial}{\partial v_{ji}(n)} 0,5 \sum_{k=1}^m (t_k(n) - f(y_{net_k}(n)))^2 \\
&= (t_k(n) - f(y_{net_k}(n))) f'(y_{net_k}) \frac{\partial}{\partial v_{ji}(n)} (y_{net_k}) \\
\frac{\partial E(n)}{\partial v_{ji}(n)} &= - \sum_{k=1}^m \delta_k \frac{\partial}{\partial v_{ji}(n)} (y_{net_k}) \\
&= - \sum_{k=1}^m \delta_k w_{kj} f'(z_{net_j}) x_i
\end{aligned} \tag{13}$$

We obtained weight changes from neurons in the hidden layer to neurons in the input layer

$$\Delta v_{ji} = \alpha f'(z_{net_j}) x_i \sum_{k=1}^m \delta_k w_{kj} \tag{14}$$

with $\delta_j = -\delta_{net_j} f'(y_{net_j})$ dan $\delta_{net_j} = \sum_{k=1}^m \delta_k w_{kj}$

The weight change of the refractive unit in the output layer is

$$\Delta v_{j0} = \alpha \delta_j \tag{15}$$

Backpropagation optimality is influenced by several parameters, namely the selection of the initial weight, the number of hidden units, the number of training patterns and the duration of the iteration. Initial weights are generated by finding random numbers in intervals [0,1] and [-1,1]. According to [8, 11] a hidden layer is sufficient to recognize any arbitrary between input and target with the specified level of accuracy. The number of patterns required is influenced by the many weights in the network as well as the expected accuracy level. The number of iterations has an effect on the error, for a small enough errors it takes a large number of iterations. The more iterations and smaller the error limit used

will be closer to the actual value. The backpropagation algorithm obtained from equations (7) through (14) is described as follows ([23]).

1. Initialization of weights (initial weights taken with fairly small random values).
2. Defined maximum iteration, error limit and learning rate (α).
3. Initialization, iteration = 0
4. Performed the following steps during (iteration < maximum iteration) and (MSE > error limit)
 - a. Iteration = iteration + 1 Training feedforward input patterns
 - b. Training feedforward input patterns.

(1) Each input unit X_i receives the signal and passes the signal to all units in the hidden layer.

(2) The input signal weight is summed by the hidden unit, therefore

$$z_{netj} = v_{j0} + \sum_{i=1}^5 x_i v_{ji}$$

and the activation function is applied to calculate the output signal

$$z_j = f(z_{netj})$$

and sends the signal to the output unit.

(3) The input signal weight is summed by the output $Y_k(k = 1)$ unit

$$y_{netk} = w_{k0} + \sum_{j=1}^2 z_j w_{kj}$$

and the activation function is applied to calculate the output signal

$$y_k = f(y_{netk})$$

$$y_k = y_{netk}$$

c. Reverse phase or *backpropagation*

Each unit of output Y_k receives a target corresponding to the training input pattern, calculated the error information

$$\delta_k = (t_k - y_k) f'(y_{netj})$$

$$\delta_k = (t_k - y_k)$$

then calculated the weight correction (used to fix w_{kj}) and calculated the weight correction of the bias unit (used to fix w_{k0})

$$\Delta w_{k0} = \alpha \delta_k$$

results of δ_k are sent to the unit in the hidden layer. Delta input from unit output is summed by each hidden unit $Z_j(j = 1, 2)$ dengan demikian

$$\delta_{netj} = \sum_{k=1}^1 \delta_k w_{kj}$$

Furthermore, this value is multiplied by the derivative of the activation function to calculate the error

$$\delta_j = \delta_{netj} f'(y_{netj})$$

$$\delta_j = \delta_{netj} \left[(1 + f(z_{netj}))(1 - f(z_{netj})) \right]$$

then calculated the weight correction (to fix v_{ij})

$$\Delta v_{j1} = \alpha \delta_j x_i$$

and calculated the weight correction of the bias unit (used for repair v_{j0})

$$\Delta v_{j0} = \alpha \delta_j$$

d. Modified weights

Each unit of output $Y_k(k = 1)$ fixes the weights ($j = 0, 1, 2$)

$$w_{kj}(\text{new}) = w_{kj}(\text{old}) + \Delta w_{kj}$$

and each unit hidden $Z_j (j = 1, 2)$ fixes the weights $(i = 0, 1, \dots, 5)$

$$v_{ji}(\text{new}) = v_{ji}(\text{old}) + \Delta v_{ji}$$

5. Test condition stopped.

Mathematically, the basic idea of this backpropagation algorithm is the application of chain rules to calculate the effect of each weight on the error function.

Algorithmic complexities are classified according to the type of function appearing in the big O notation. The complexity of the backpropagation algorithm is with the timing of the linear. Linear is included in the time complexity spectrum of a polynomial algorithm that is a well-performing algorithm.

3.7. Network architecture of GSTAR-ANN

The network architecture used is multilayer feedforward networks consisting of input, hidden and output layers ([8]). In the training process, data records are used as training data. GSTAR model is used as input in the training process so that in the input layer used p neuron, q neuron on hidden layer. As the target is taken the actual data after the period ends so that the neuron in the output layer is only one. Each input will receive an input signal and forward the signal to the hidden layer, then to the output layer. Since the neurons in the expected output layer are only one, $k = 1$ is obtained. From the calculation, we can determine the number of neurons in the hidden layer that is q neuron so that the obtained network architecture is multilayer feedforward networks with p neuron in the input layer, q neuron in hidden layer and 1 neuron in the output layer. Note that with the backpropagation algorithm when a network is assigned an enter pattern as a training pattern then the pattern goes to the units in the hidden layer to be forwarded at the output layer. Furthermore the output layer unit provides a response called a network output. When the output is not equal to the expected output, the output will be backward on the hidden layer forwarded to the unit on the input layer. The GSTAR-ANN model is generally written as

$$\hat{Y}_t = f(x_t, \gamma, \theta) = g_2 \left\{ \sum_{j=0}^q \theta_{0j} g_1 \left[\sum_{i=0}^p \gamma_{ij} x_{it} \right] \right\}$$

$$\hat{Y}_t = \begin{bmatrix} \hat{Y}_{1t} \\ \hat{Y}_{2t} \\ \vdots \\ \hat{Y}_{pt} \end{bmatrix}; Y_{1t-1}^* = \begin{bmatrix} \hat{Y}_{1t} \\ 0 \\ \vdots \\ 0 \end{bmatrix}; Y_{2t-1}^* = \begin{bmatrix} 0 \\ \hat{Y}_{2t} \\ \vdots \\ 0 \end{bmatrix}; Y_{pt-1}^* = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \hat{Y}_{pt} \end{bmatrix}$$

with $x_t = \{Y_{1t-1}^*, Y_{2t-1}^*, \dots, Y_{pt-1}^*, F_{1t-1}, F_{2t-1}, \dots, F_{pt-1}\}$ is the input and $\gamma = \{\gamma_{ij}\}$ is the weight that connects the input layer to the hidden layer with $i = 1, 2, \dots, p$, $j = 1, 2, \dots, q$ and $\theta = \{\theta_{ij}\}$ are the weights connecting from the hidden layer to the output layer and $g_i(.)$ dan $g_2(.)$ are the activation functions (in this case a the sigmoid function is used). Furthermore, this architecture is shown in Figure 4.

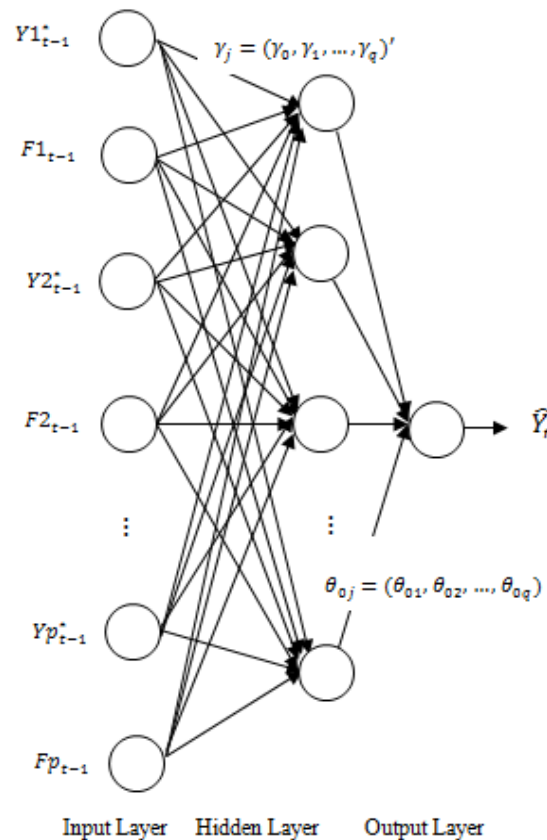


Figure 4. networks architecture

Figure 4 shows the neuron p on the input layer, q neuron in the hidden layer, and 1 neuron at the output layer with the total parameters to be trained as $pq + q$ bias and $qx1 + 1$ bias or $pq + 2q + 1$ or $q(p + 2) + 1$.

4. Conclusion

GSTAR-ANN model is constructed in 3 layers namely input, hidden and output. The GSTAR model is used as input in the training process. Each input will receive an input signal and forward the signal to the hidden layer, then to the output layer (multilayer feedforward networks). GSTAR-ANN model constructed with backpropagation algorithm and network architecture used $p - q - 1$. The complexity of the backpropagation algorithm is $O(n)$ with the timing of the linear. Linear is included in the time complexity spectrum of the polynomial algorithm i.e. the algorithm with good performance. The GSTAR-ANN model has network architecture with one neuron unit at the output layer, p input neuron, q neuron in hidden layer (multilayer feedforward networks).

5. Acknowledgement

In the research, financing and writing of this paper, the authors would like to thank the two institutions : the ministry of research, technology and higher education of the Republic of Indonesia; research institute and community services of Universitas Sebelas Maret through superior basic research grants from universities with contract number 474 / UN27.21 / PP / 2018.

References

- [1] Box G, Jenkins GM and Reinsel G 1970 *Time Series Analysis, forecasting and control, fourth edition* (John Willey and Sons New Jersey).
- [2] Makridakis S and Hibon M 2000 *The M3-Competition: results, conclusions and implications*

- International Journal of Forecasting* **16** 451-476.
- [3] Borovkova S, Ruchjana B N and Lopuhaa H 2008 Consistency and Asymptotic Normality of Least Square Estimators in Generalized STAR Model *Journal Compilation of Statistica Neerlandica* **62** 482-508.
 - [4] Ruchjana B N 2002 Pemodelan Kurva Produksi Minyak Bumi Menggunakan Model Generalisasi S-TAR *Forum Statistika dan Komputasi* IPB Special Edition.
 - [5] Suhartono, Wahyuningrum S R, Setiawan and Akbar M S 2016 GSTARX-GLS model for spatio-temporal data forecasting *Malaysian Journal of Mathematical Sciences* **10**(S) 91-103.
 - [6] Suhartono, Prastyo D D, Kuswanto H and Lee M H 2018 Comparison between VAR, GSTAR, FFNN-VAR and FFNN-GSTAR Models for Forecasting Oil Production *MATEMATIKA* **34** (1) 103–111.
 - [7] Cheng B and Titterton D M 1994 Neural Network : A review from a Statistical Prespective *Statistical Science* **9** 2-54.
 - [8] Fausett L 1994 *Fundamentals of Neural Networks: Architectures, Algorithms and Applications* (Prentice Hall, Inc. New Jersey).
 - [9] Herve A 1994 A Neural Network Primer *Journal of Biological Systems* **02**(03) 247-281
 - [10] Anders U and Korn O 1999 Model Selection in Neural Network *Neural Networks* **12** 309–323.
 - [11] Jek S J 2005 *Jaringan Syaraf Tiruan dan Pemrograman Menggunakan Matlab* (Andi Offset, Yogyakarta, Cetakan pertama).
 - [12] Medeiros M C, Terasvirta T and Rech G 2006 Building Neural network for Time series: A Statistical Approach *Journal of Forecasting* **3** 75-115.
 - [13] Sarle W S 1994 Neural Networks and Statistical Models *Proceedings of the Nineteenth Annual SAS Users Group International Conference SAS Institute Inc., Cary, NC, USA*.
 - [14] Warner B, Misra M 1996 Understanding Neural Network as Statistical Tools *American Statistician* **50** 284-293.
 - [15] Zang G, Eddy P B, and Hu MY 1998 Forecasting with artificial neural networks: The state of the art *International Journal of Forecasting* **14** 35 – 62.
 - [16] White H 1989 Learning in Artificial Neural Networks: A Statistical Perspective *Neural Computation* **1** 425–464.
 - [17] Jingtao Y, Tan C L and Poh H L 1999 Neural Networks for Technical Analysis : A study on KLCI *International Journal of Theoretical and Applied Finance* **2** (2) 221-241.
 - [18] Nurhayati N, Pasaribu U S and Neswan O 2012 Application of generalized space-time autoregressive model on GDP data in West European countries *Journal of Probability and Statistics* 1-16.
 - [19] Saputro D R S, Jati M D W, Widyaningsih P 2016 Vector Autoregressive Generalized Space Time Autoregressive (VARGSTAR) Model with 2-Means Clustering on Rainfall of Central Java *Proceeding of The 2nd International Conference on Applied Statistics 2016* 148-154.
 - [20] Wutsqa D U and Suhartono 2010 Seasonal multivariate time series forecasting on tourism data by using VAR-GSTAR mode *Jurnal ILMU DASAR* **11**(1) 101-109.
 - [21] Jaiswal J K and Das R 2018 Artificial Neural Network Algorithms based Nonlinear Data Analysis for Forecasting in the Finance Sector *International Journal of Engineering & Technology* **7** (4.10) 169-176
 - [22] Bhati R, Shubham S, Chhandak B and Vijayarajan V 2018 Critical Decision Making Using Neural Networks *International Journal of Engineering & Technology* **7** (4.10) 15-18,
 - [23] Yayik A and Kutlu Y 2013 Improving Pseudo random number generator using artificial neural networks *21st Signal Processing and Communications Applications Conference (SIU)* 1-4.