We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

13

An Application of Jordan Pi-Sigma Neural Network for the Prediction of Temperature Time Series Signal

Rozaida Ghazali, Noor Aida Husaini,
Lokman Hakim Ismail and Noor Azah Samsuddin
Universiti Tun Hussein Onn Malaysia
Malaysia

1. Introduction

Temperature forecasting is mainly issued in qualitative terms with the use of conventional methods, assisted by the data projected images taken by meteorological satellites to assess future trends (Paras *et al.*, 2007). Several criteria that need to be considered when choosing a forecasting method include the accuracy, the cost and the properties of the series being forecast. Considering those criteria, it is noted that such empirical approaches that has been conducted for temperature forecasting is intrinsically costlier and only proficient of providing certain information, which is usually generalized over a larger geographical area (Paras *et al.*, 2007). Despite of involving sophisticated mathematical models to justify the use of empirical rules, it also requires a prior knowledge of the characteristics of the input time-series to predict future events. Not only that, most temperature forecasts today have limited information about uncertainty. Yet, meteorologists often find it challenging to communicate uncertainty effectively. Regardless of the extensive use of the numerical weather method, they are still restricted by the availability of numerical weather prediction products, leading to various studies being conducted for temperature forecasting (Barry & Chorley, 1982; Paras *et al.*, 2007)

Due to that inadequacy, Neural Network (NN) has been applied in such temperature forecasting. NN mimic human intelligence in learning from complicated or imprecise data and can be used to extract patterns and detect trends that are too complex to be perceived by humans and other computer techniques (Mielke, 2008). NN which can be described as an adaptive machine that has a natural tendency for storing experiential knowledge, are able to discover complex nonlinear relationships in the meteorological processes by communicating forecast uncertainty that relates the forecast data to the actual weather (Chang *et al.*, 2010). However, when the number of inputs to the model and the number of training examples becomes extremely large, the training procedure for ordinary neural network, especially the Multilayer Perceptron (MLP) becomes tremendously slow and unduly tedious. Indeed, MLP are prone to overfit the data (Radhika & Shashi, 2009) and adopts computationally intensive training algorithms. On the other hand, MLP also suffer long training times and often reach local minima (Ghazali & al-Jumeily, 2009).

Considering the limitations of MLP, therefore in this work, the intention of utilizing the use of higher order neural networks (HONN) which have the ability to expand the input representation space is considered. The Pi-Sigma Neural Network (PSNN) (Shin & Ghosh, 1991-a), a class of HONN, is able to perform high learning capabilities that require less memory in terms of weights and nodes, and at least two orders of magnitude less number of computations when compared to the MLP for similar performance levels, and over a broad class of problems (Ghazali & al-Jumeily, 2009; Shin & Ghosh, 1991-b).

In conjunction with the benefits of PSNN, a new model called Jordan Pi-Sigma Neural Network (JPSN) which posses a Jordan Neural Network architecture (Jordan, 1986) is proposed to perform temperature forecasting. In this regard, the JPSN that managed to incorporates feedback connections in their structure and having the superior properties of PSNN is mapped to function variable and coefficient related to the research area. Consequently, this work is conducted in order to prove that JPSN is suitable for one-step-ahead temperature prediction.

2. Pi-sigma neural network (PSNN)

PSNN is a type of HONN and was first introduced by Shin & Ghosh (1991-a). The basic idea behind the network is due to the fact that a polynomial of input variables is formed by a product ("pi") of several weighted linear combinations ("sigma") of input variables. That is why this network is called pi-sigma instead of sigma-pi. The PSNN exhibits fast learning while greatly reducing network complexity by utilising an efficient form of polynomials for many input variables. This special polynomial form helps the PSNN to dramatically reduce the number of weights in its structure. Figure 1 shows the architecture of PSNN:

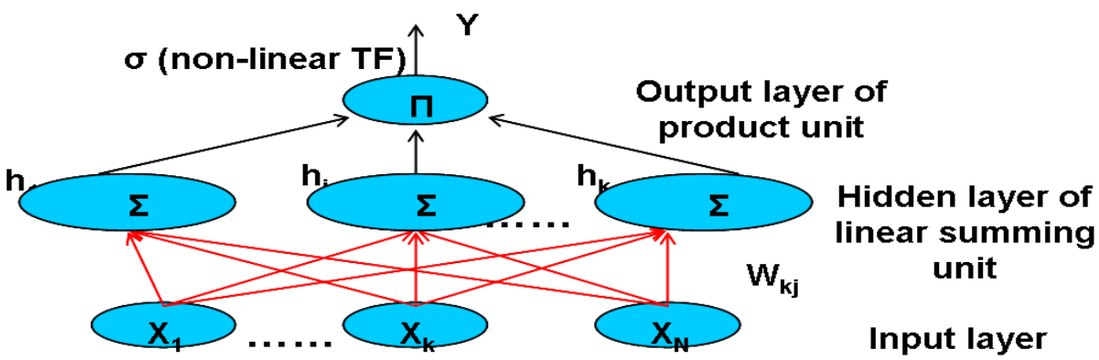


Fig. 1. Structure of K^{th} Order PSNN

Input x is an N dimensional vector and x_k is the k^{th} component of x. The weighted inputs are fed to a layer of K linear summing units; h_{ji} is the output if the j^{th} summing units for the i^{th} output y_i, viz:

$$y_i = \sigma\left(\prod_j \left(\sum_k w_{kji} x_k + \theta_{ji}\right)\right) \tag{1}$$

where w_{kji} and θ_{ji} are adjustable coefficients, and σ is the nonlinear transfer function (Shin & Ghosh, 1991-a). The number of summing units in PSNN reflects the network order. By using an additional summing unit, it will increase the network's order by 1 whilst preserving old connections and maintaining network topology.

In PSNN, weights from summing layer to the output layer are fixed to unity, resulting to a reduction in the number of tuneable weights. Therefore, it can reduce the training time. Sigmoid and linear functions are adopted in the output layer and summing layer, respectively. The use of linear summing units makes the convergence analysis of the learning rules for the PSNN more accurate and tractable (Ghazali & al-Jumeily, 2009; Ghazali *et al.*, 2006). Compared to other HONN models, Shin and Ghosh (1991-b) argued that PSNN can contribute to maintain the high learning capabilities of HONN, needs a much smaller number of weights, with at least two orders of magnitude less number of computations when compared to the MLP for similar performance levels, and over a broad class of problems (Ghazali *et al.*, 2006). Moreover, the PSNN is superior to other HONN in approaching precision computation complexity and has a highly regular structure. Since weights from hidden layer to the output are fixed at 1, the property of PSNN drastically reduces the training time. The applicability of this network was successfully applied for image processing (Hussain and Liatsis, 2002), time series prediction (Knowles, 2005; Ghazali *et al.*, 2011), function approximation (Shin & Ghosh, 1991-a; Shin & Ghosh, 1991-b), pattern recognition (Shin & Ghosh, 1991-a), Cryptography (Song, 2008), and so forth.

3. The properties and structure of Jordan pi-sigma neural network (JPSNN)

The structure of JPSN is quite similar to the ordinary PSNN. The main difference is the architecture of JPSN is constructed by having a recurrent link from output layer back to the input layer. This structure gives the temporal dynamics of the time-series process that allows the network to compute in a more parsimonious way (Hussain & Liatsis, 2002). The architecture of the proposed JPSN is shown in Figure 2 below.

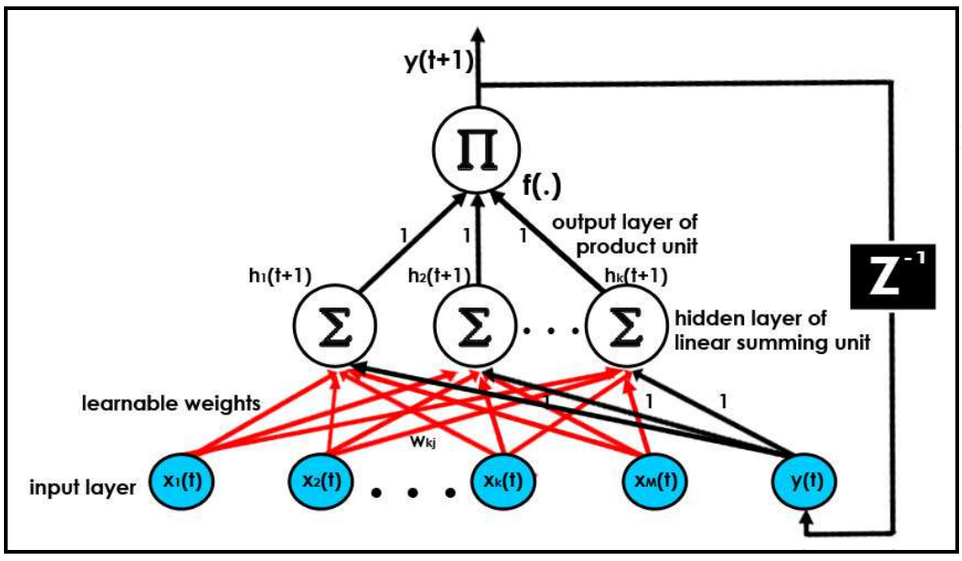


Fig. 2. The architecture of JPSN

where

$x(t)$ - the input nodes at *t-th* time

w_{kj} - the trainable weights

$h_k(t+1)$ - the summing unit

$y(t+1)$ - the output at time $t+1$

$y(t)$ the output at time *t*

$f(\cdot)$ the activation function

Weights from the input layers $x(t)$ to the summing units layer are tunable, while weights between the summing unit layers and the output layer are fixed to 1. The tuned weights are used for network testing to see how well the network model generalizes on unseen data. Z^{-1} denotes time delay operation.

Let the number of external inputs to the network be M and the number of the output be 1. Let $x_m(t)$ be the m *-th* external input to the network at time *t*. The overall input at time *t* is the concatenation of $y(t)$ and $x_k(t), where(k=1,...,M)$, and is referred to $z(t)$ where:

$$z_k(t) = \begin{cases} x_k(t) & if \quad 1 \le k \le M \\ 1 & if \quad k = M+1 \\ y_k(t) & if \quad k = M+2 \end{cases} \tag{2}$$

Meanwhile, weights from $z(t)$ to the summing unit are set to 1 in order to reduce the complexity of the network.

The proposed JPSN combines the properties of both PSNN and Recurrent Neural Network (RNN) so that better performance can be achieved. When utilizing the newly proposed JPSN as predictor for one-step-ahead prediction, the previous input values are used to predict the next elements in the data. Since network with recurrent connection holds several advantages over ordinary feedforward MLP especially in dealing with time-series problems, therefore, by adding the dynamic properties to the PSNN, this network may outperform the ordinary feedforward MLP and also the ordinary PSNN. Additionally, the unique architecture of JPSN may also avoid from the combinatorial explosion of higher-order terms as the network order increases.

3.1 Learning algorithm of JPSN

The supervised learning used in JPSN can be solved with the standard backpropagation (BP) gradient descent algorithm (Rumelhart *et al.*, 1986), with the recurrent link from output layer back to the input layer nodes. Since the same weights are used for all networks, the learning algorithm starts by initialising the weights to a small random value before training the weights. The JPSN is trained adaptively in which the errors produced are calculated and the overall error function *E* of the JPSN is defined as:

$$e_j(t) = d_j(t) - y_j(t) \tag{3}$$

where $d_j(t)$ denotes the target output at time $(t-1)$. At each time $(t-1)$, the output of each $y_j(t)$ is determined and the error $e_j(t)$ is calculated as the difference between the actual value expected from each unit i and the predicted value $y_j(t)$.

Generally, JPSN can be operated in the following steps:

For each training example:

1. Calculate the output.

$$y(t) = f\left(\prod_{L=1}^{k} h_L(t)\right) \tag{4}$$

$h_l(t)$ can be calculated as:

$$h_L(t) = \sum_{m=1}^{m} w_{Lm} x_m(t) + w_{L(m)} + w_{L(m+1)}\, y(t-1) = \sum_{m=1}^{m+1} w_{Lm} z_m(t-1) \tag{5}$$

where $h_L(t)$ represents the activation of the L unit at time t, and $y(t)$ is the previous network output. The unit's transfer function f sigmoid activation function, which bounded of output range of [0,1].

2. Compute the output error at time (t) using standard Mean Squared Error (MSE) by minimising the following index:

$$E_k = \frac{1}{n_{tr}} \sum_{i=1}^{n_{tr}} (y_i - z_{ki})^2 \tag{6}$$

where z_{ik} denotes the output of the k-th node with respect to the i-th data, and n_{tr} is the number of training sets. This step is completed repeatedly for all nodes on the current layer.

3. By adapting the BP gradient descent algorithm, compute the weight changes:

$$\Delta w_j = \eta \left(\prod_{j\neq 1}^{m} h_{ji}\right) x_k \tag{7}$$

where h_{ji} is the output of summing unit and η is the learning rate.

4. Update the weight:

$$w_i = w_i + \Delta w_i \tag{8}$$

5. To accelerate the convergence of the error in the learning process, the momentum term, α is added into Equation 3.6. Then, the values of the weight for the interconnection on neurons are calculated and can be numerically expressed as

$$w_i = w_i + \alpha \Delta w_i \tag{9}$$

where the value of α is a user-selected positive constant $(0 \le \alpha \le 1)$

6. The JPSN algorithm is terminated when all the stopping criteria (training error, maximum epoch and early stopping) are satisfied. If not, repeat step 1)

The utilisation of product units in the output layer indirectly incorporates the capabilities of JPSN while using a small number of weights and processing units. Therefore, the proposed JPSN combines the properties of both PSNN and JNN so that better performance can be achieved. When utilising the newly proposed JPSN as predictor for one-step-ahead, the previous input values are used to predict the next element in the data. Since network with recurrent connection holds several advantages over ordinary feedforward networks especially in dealing with time-series problems, therefore, by adding the dynamic properties to the PSNN, this network may outperformed the MLP and also the ordinary PSNN. Additionally, the unique architecture of JPSN may also avoid from the combinatorial explosion of higher-order terms as the network order increases. The JPSN has a topology of a fully connected two-layered feedforward network. Considering the fixed weights that are not tuneable, it can be said that the summing layer is not "hidden" as in the case of the MLP. This is by means; such a network topology with only one layer of tuneable weights may reduce the training time.

4. Temperature prediction with JPSN

Temperature forecasting is the essence of traceability for weather forecasting. Certainly, temperature is a kind of atmospheric time-series data where the time index takes on a predetermined or unlimited set of values. The temperature can have a greater influence in daily life than any other single element on a routine basis. Therefore, some great observation are needed to obtain accuracies for the temperature measurement (Ibrahim, 2002). Temperature forecasting undoubtedly is the most challenging task in dealing with meteorological parameters. It represents not only a very complex nonlinear problem, but also extremely difficult to model.

A great interest in developing methods for more accurate predictions for temperature forecasting has led to the development of several methods which employ the use of physical methods, statistical-empirical methods and numerical-statistical methods (Barry & Chorley, 1982; Lorenc, 1986). These methods, however, constitutionally complex and are limited and restricted to that of numerical weather prediction products (Paras *et al.*, 2007). Considering the downside of those methods, Neural Networks have placed such sophisticated models within the reach of practitioners, and therefore have been successfully applied in many problems. Therefore, in this work, JPSN is used for temperature perdiction in Batu Pahat.

The forecasting horizon for temperature prediction is a one-step-ahead, whereas the output variable represents the temperature measurement of one-day ahead of temperature data. A univariate data of a 5-years daily temperature measurement in Batu Pahat Malaysia, ranging from 2005 to 2009 was used for the simulation (please refer to Table 1). The data was obtained from the Central Forecast Office, Malaysian Meteorological Department (MMD).

Size	Maximum (^{o}C)	Minimum (^{o}C)	Average (^{o}C)
1826	29.5	23.7	26.75

Table 1. The properties of Batu Pahat Temperature signal

To purify the data for further processing, it is needed to identify and remove the contaminating effects of the outlying objects on the data. Therefore, in this study, a Max-Min Normalization technique was used so that the data can be distributed evenly and scaled into an acceptable range. In order to avoid computational problems, the range was set between the upper and lower bound of the network transfer function, which often to be the monotonically increasing function, $1/\left(1+e^{-x}\right)$ (Cybenko, 1989) between [0, 1]. The Max-Min Normalization can be implemented using the following equation:

$$v' = \frac{v - \min A}{\max A - \min A}(new\_\max A - new\_\min A) + new\_\min A \tag{10}$$

Let A be the temperature data of Batu Pahat region and $min\ A$, $max\ A$ indicate the minimum and maximum values of data A. Max-Min Normalization maps a value v of data A to v' in the range $\{new\_\min A, new\_\max A\}$.

In data normalization, the statistical distribution values for each input and output are roughly uniform. Therefore, removing the outliers should make the data more accurate. Figure 3 shows the daily temperature data of Batu Pahat region before normalization while Figure 4 shows the daily temperature data of Batu Pahat region after normalization.

Meanwhile, Figure 5 shows the frequency of temperature distribution data for 5-years after normalization process. From Figure 5, it can be seen that the histogram of the transformed data is symmetrical. Therefore, it can be said that the temperature data for Batu Pahat (after normalization) is relatively uniform, and closely follow the normal distribution, thus suitable as the network inputs.

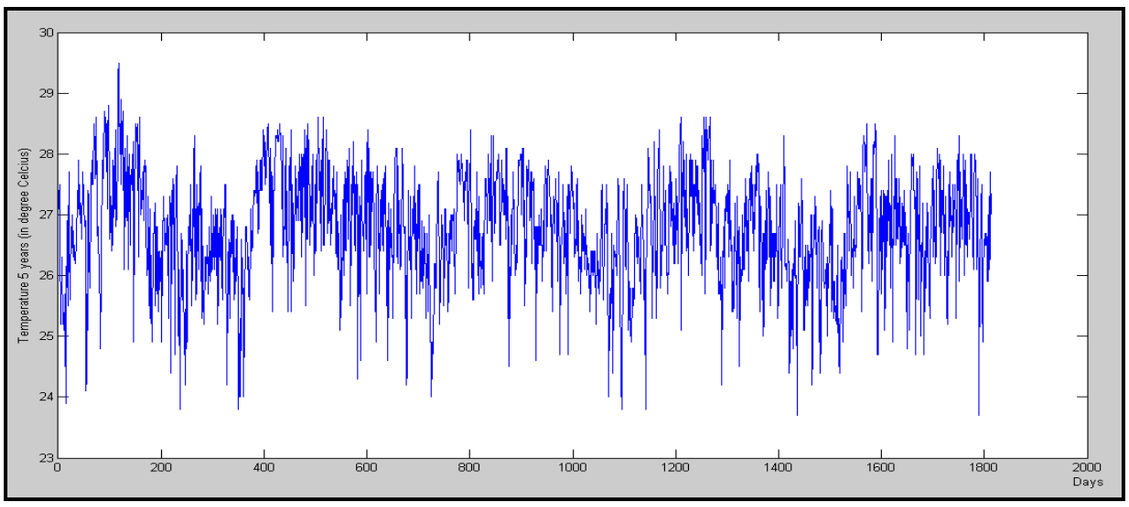


Fig. 3. Daily Temperature Data of Batu Pahat Region (before normalization)

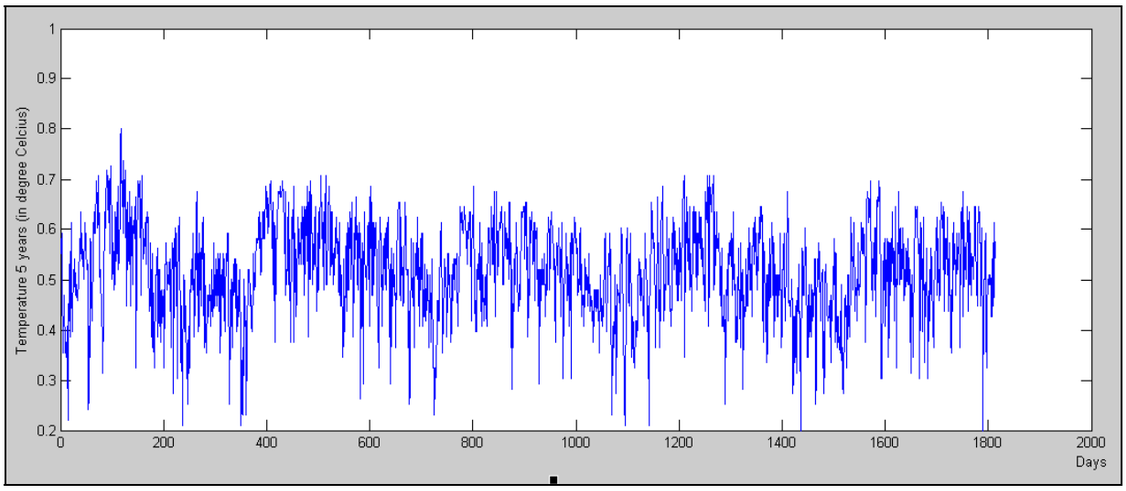


Fig. 4. Daily Temperature Data of Batu Pahat Region (after normalization)

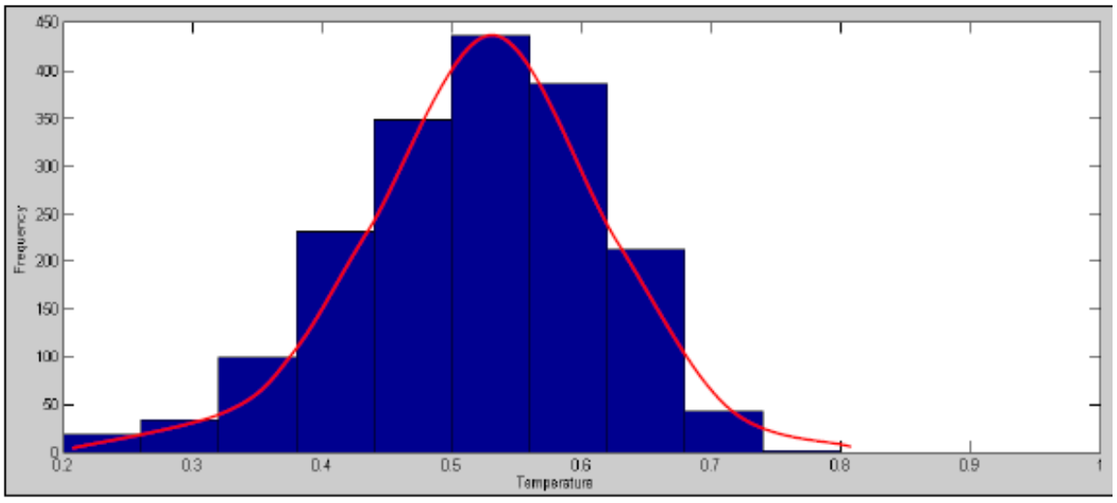


Fig. 5. Normal Distribution of Temperature Data (after normalization)

For simulation purposes, the data was segregated into time order and was divided into three sets; 50% for training, 25% for testing and 25% for validation, as shown in Table 2.

Training	Validation	Testing
914	456	456

Table 2. Summary of Temperature Dataset Segregation

For comparison purposes, the JPSN performances on temperature prdiction will be benchmarked againts that of the ordinary PSNN and the widely known MLP. As there is no rule of thumb for identifying the number of input, a trial-and-error procedure was determined. All networks were built considering 5 different number of input nodes ranging from 4 to 8. A single neuron was considered for the output layer. The number of hidden nodes (for MLP), and the higher order terms (for PSNN and JPSN) were initially started with 2 nodes, and increased by one until a maximum of 5 nodes.

5. Simulation results

The temperature dataset collected from MMD was used to demonstrate the performance of JPSN by considering a few different network parameters. Generally, the factors affecting the network performance include the learning factors, the higher order terms, and the number of neurons in the input layer. Extensive experiments have been conducted for training, testing and validation sets, and average results of 10 simulations/runs have been collected. Two stopping criteria were used during the learning process; the maximum epoch and the minimum error, which were set to 3000 and 0.0001 respectively. In order to assess the performance of all network models, four measurement criteria, namely the number of epoch, Mean Squared Error, Normalized Mean Squared Error, and Signal to Noise Ratio are used. Convergence is achieved when the output of the network meets the earlier mentioned stopping criteria. By considering all in-sample dataset that have been trained, the best value for the momentum term $\alpha = 0.2$ and the learning rate $\eta = 0.1$, were chosen based on extensive simulations done by trial-and-error procedure.

The above discussions have shown that some network parameters may affect the network performances. In conjunction with that, it is necessary to illustrate the robustness of JPSN by comparing its performance with the ordinary PSNN and the MLP. Table 3 to Table 5 show the average results from 10 simulations for the JPSN, the ordinary PSNN and the MLP, respectively.

Input Nodes	Network Order	NMSE on Testing Dataset	MSE on Testing Dataset	SNR	Number of Epoch
4	2	0.7710	0.0065	18.7557	1460.9
	3	0.7928	0.0066	18.6410	1641.1
	4	0.8130	0.0068	18.5389	1209.9
	5	0.8885	0.0074	18.1574	336.8
5	2	0.7837	0.0066	18.6853	193.5
	3	0.8253	0.0069	18.4705	287.6
	4	0.8405	0.0070	18.3910	214.5
	5	0.8632	0.0072	18.2762	117.2
6	2	0.7912	0.0066	18.6504	285.3
	3	0.8329	0.0070	18.4333	292.8
	4	0.8522	0.0071	18.3341	238
	5	0.8850	0.0074	18.1691	97.1
7	2	0.7888	0.0066	18.6626	236.3
	3	0.8310	0.0070	18.4425	178.8
	4	0.8206	0.0069	18.4954	182.2
	5	0.8751	0.0073	18.2213	116.7
8	2	0.8005	0.0067	18.5946	185.9
	3	0.8166	0.0069	18.5106	283.9
	4	0.8468	0.0071	18.3515	149.4
	5	0.8542	0.0072	18.3147	152

Table 3. Average Result of JPSN for One-Step-Ahead Prediction.

Input Nodes	Network Order	NMSE on Testing Dataset	MSE on Testing Dataset	SNR	Number of Epoch
4	2	0.7791	0.0065	18.7104	1211.8
	3	0.7792	0.0065	18.7097	1302.9
	4	0.7797	0.0065	18.7071	1315.3
	5	0.7822	0.0066	18.6935	1201.0
5	2	0.7768	0.0065	18.7234	1222.5
	3	0.7769	0.0065	18.7226	1221.6
	4	0.7775	0.0065	18.7193	1149.9
	5	0.7806	0.0065	18.7023	916.9
6	2	0.7758	0.0065	18.7289	961.3
	3	0.7770	0.0065	18.7222	852.5
	4	0.7775	0.0065	18.7192	1155.6
	5	0.7832	0.0066	18.6880	948.0
7	2	0.7726	0.0065	18.7470	1064.0
	3	0.7733	0.0065	18.7432	911.6
	4	0.7760	0.0065	18.7277	922.1
	5	0.7766	0.0065	18.7244	1072.2
8	2	0.7674	0.0064	18.7688	719.3
	3	0.7677	0.0064	18.7671	877.0
	4	0.7693	0.0065	18.7579	861.4
	5	0.7694	0.0065	18.7574	970.9

Table 4. Average Result of PSNN for One-Step-Ahead Prediction.

Input Nodes	Hidden Nodes	NMSE on Testing Dataset	MSE on Testing Dataset	SNR	Number of Epoch
4	2	0.7815	0.0065	18.6971	2849.9
	3	0.7831	0.0066	18.6881	2468.2
	4	0.7825	0.0066	18.6918	2794.2
	5	0.7827	0.0066	18.6903	2760.9
5	2	0.7803	0.0065	18.7037	2028.6
	3	0.7792	0.0065	18.7097	2451.8
	4	0.7789	0.0065	18.7114	2678.1
	5	0.7792	0.0065	18.7097	2565.8
6	2	0.7750	0.0065	18.7335	2915.1
	3	0.7763	0.0065	18.7261	2837.2
	4	0.7776	0.0065	18.7188	2652.4
	5	0.7783	0.0065	18.7152	2590.8
7	2	0.7742	0.0065	18.7378	2951.2
	3	0.7780	0.0065	18.7164	2566.6
	4	0.7771	0.0065	18.7217	2796.4
	5	0.7786	0.0065	18.7131	2770.0
8	2	0.7734	0.0065	18.7350	2684.8
	3	0.7749	0.0065	18.7268	2647.2
	4	0.7747	0.0065	18.7278	2557.1
	5	0.7753	0.0065	18.7242	2774.6

Table 5. Average Result of MLP for One-Step-Ahead Prediction.

As it can be noticed, Table 3 which shows the results for temperature prediction using JPSN reveals that the 2nd order network, with 4 inputs demonstrates the best results using all measuring criteria except for the number of epochs. Meanwhile, Table 4 shows the results that were produced by PSNN on temperature prediction. It demonstrates that the combination of 8 input nodes and PSNN of Order 2 shows the best performance for all measuring criteria. Same thing goes to the MLP, which signifies that MLP with 2 hidden nodes and 8 input nodes attained the best results for all measuring criteria except for SNR and number of epochs (refer to Table 5).

Network Models	NMSE on Testing Dataset	MSE on Testing Dataset	MAE on Testing Dataset	SNR	Number of Epoch
JPSN	0.771034	0.006462	0.063458	18.7557	1460.9
PSNN	0.779118	0.006529	0.063471	18.71039	1211.8
MLP	0.781514	0.006549	0.063646	18.69706	2849.9

Table 6. Comparison on the Best Single Simulation Results for JPSN, PSNN and MLP.

In order to compare the predictive performance of the three models, Table 6 presents the best simulation results for JPSN, PSNN and MLP. Over all the training process, JPSN obtained the lowest MAE, which is 0.063458; while the MAE for PSNN and MLP were 0.063471 and 0.063646, respectively. By considering the MAE, it shows that JPSN is able to make a very close forecasts to the actual output in analysing the temperature. In this respect, JPSN outperformed PSNN by a ratio of 1.95×10^{-4}, and 2.9×10^{-3} for the MLP. Moreover, it can be seen that JPSN reached higher value of SNR. Therefore, it can be said that the network can track the signal better than PSNN and MLP. Apart from the MAE and SNR, it is verified that JPSN exhibited lower prediction errors, in terms of NMSE and MSE on the out-of-sample dataset. This indicates that the network is capable of representing nonlinear function better than the two benchmarked models. In the case of learning speed, particularly on the number of epoch utilized, PSNN converged much faster than the JPSN and MLP. However, JPSN reached a smaller number of epoch when compared to the MLP. On the whole, the performance of JPSN gives a gigantic comparison when compared to the two benchmarked models.

For demonstration purpose, the models' performance on their NMSE is depicted in Figure 6. It shows that JPSN steadily gives lower NMSE when compared to both PSNN and MLP. This by means shows that the predicted and the actual values which were obtained by the JPSN are better than both comparable network models in terms of bias and scatter. Consequently, it can be inferred that the JPSN yield more accurate results, providing the choice of network parameters are determined properly. The parsimonious representation of higher order terms in JPSN assists the network to model successfully.

The plots depicted in Figures 7 to 9 present the temperature forecast on the out-of-sample dataset for all network models. As shown in the plots, the blue line represents the trend of the actual values, while the red line represents the predicted values. The predicted values of daily temperature measurement made by all network models almost fit the actual values with minimum error forecast. On the whole, JPSN practically beat out PSNN and MLP by 1.038% and 1.341%, respectively. It is verified that JPSN has the ability to perform an input-output mapping of temperature data as well as better performance when compared to

both network models. Besides, the evaluations on MAE, NMSE, MSE, and SNR over the temperature data demonstrated that JPSN were merely improved the performance level compared to the two benchmarked network models, PSNN and MLP. The better performance of temperature forecasting is allocated based on the vigour properties it contains. Hence, it can be seen that the thrifty representation of higher order terms in JPSN assists the network to model effectively.

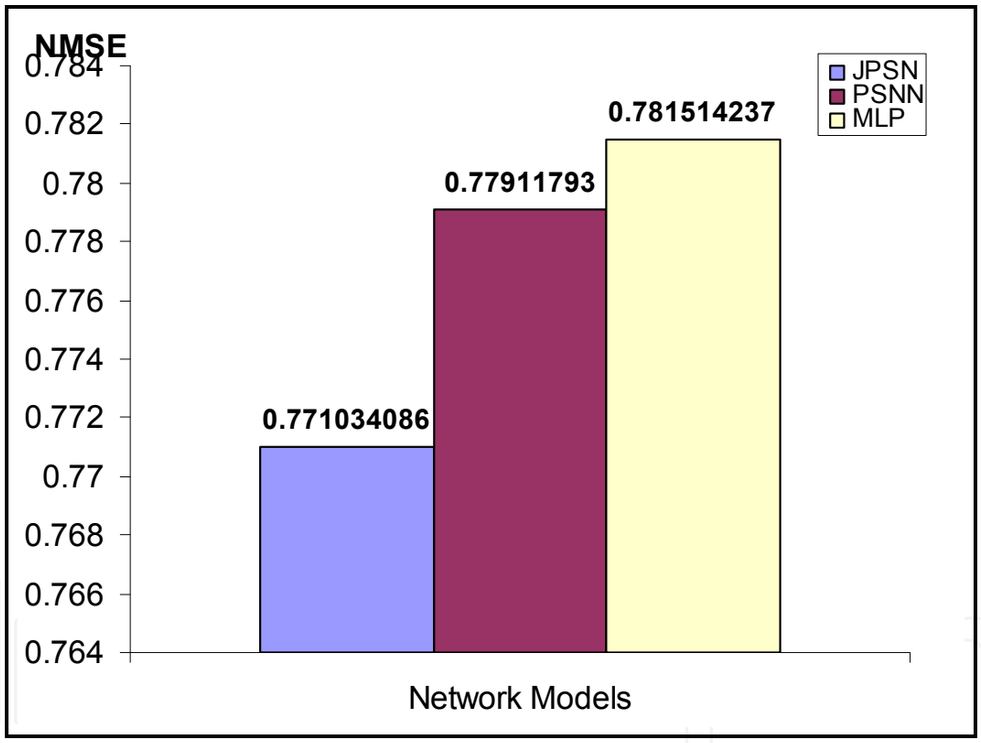


Fig. 6. The NMSE for JPSN, PSNN, and MLP.

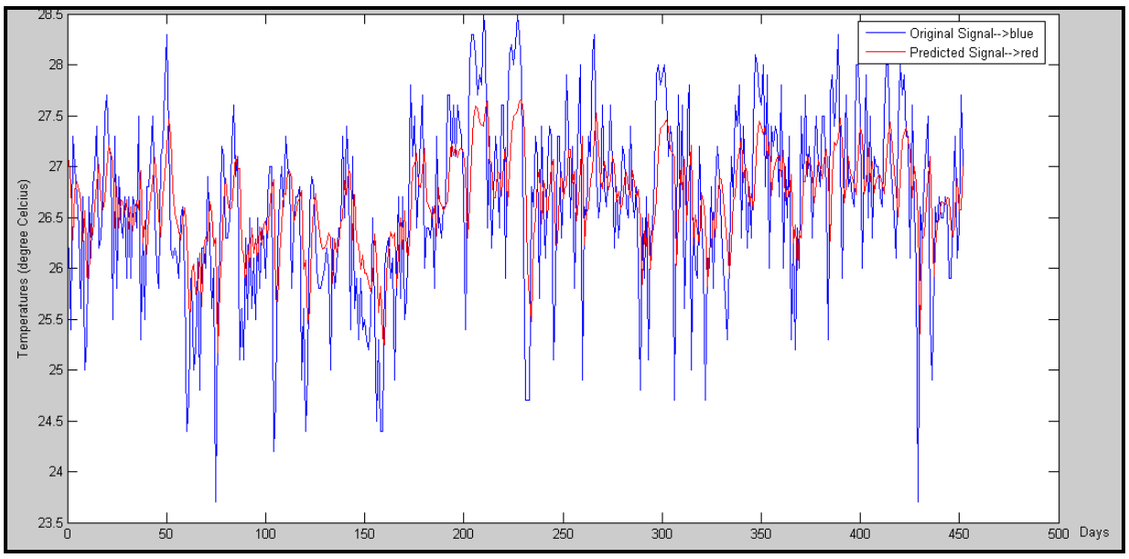


Fig. 7. Temperature Forecast made by JPSN on Out-of sample Dataset.

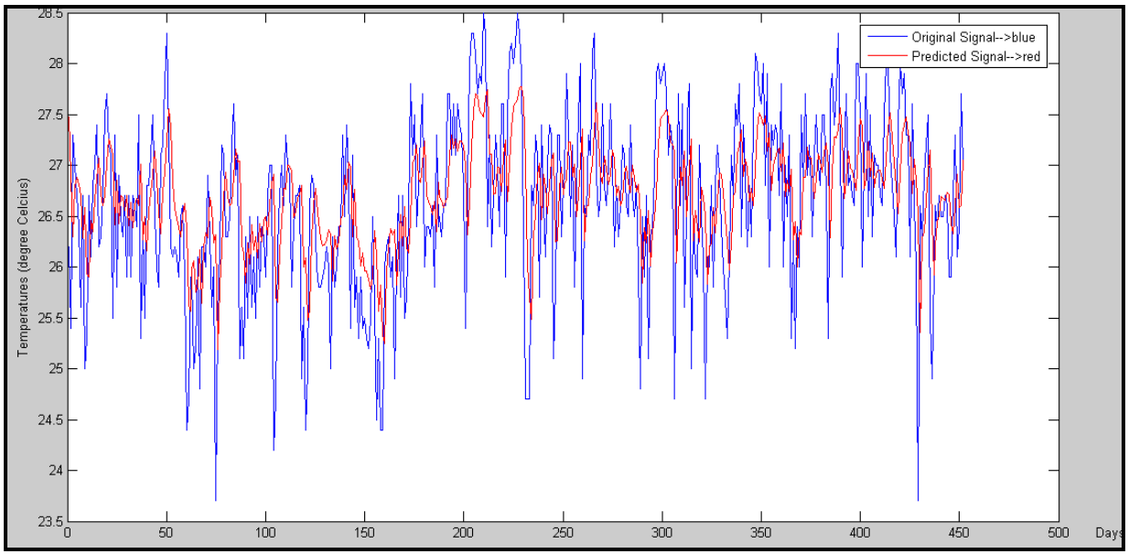


Fig. 8. Temperature Forecast made by PSNN on Out-of sample Dataset.

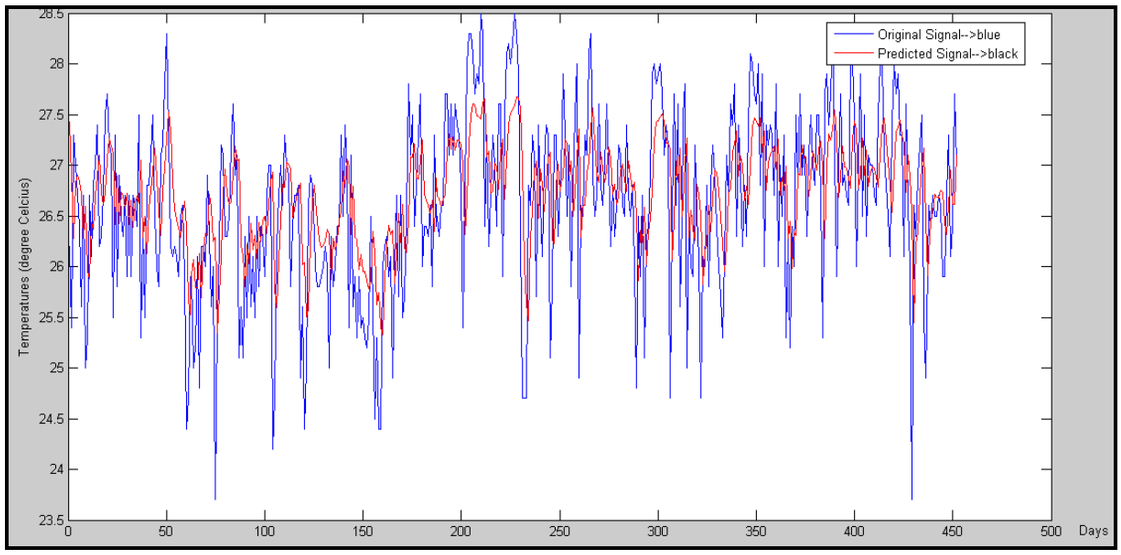


Fig. 9. Temperature Forecast made by MLP on Out-of sample Dataset.

6. Conclusion

There are many applications and techniques on temperature that was developed in the past. However, limitations such as the accuracy and complexity of the models make the existing system less enviable for some applications. Therefore, improvement on temperature forecasting requires continuous efforts in many fields, including NN. Several methods related to NN, particularly have been investigated and carried out. However, the ordinary feedforward NN, the MLP, is prone to overfitting and easily get stuck into local minima. Thus, to overcome the drawbacks, a new model, called JPSN is proposed as an alternative mechanism to predict the temperature event. The JPSN which combines the properties of PSNN and RNN can benefits the temperature prediction event, which may overcome such drawbacks in MLP. In this chapter, JPSN is used to learn the historical temperature data of Batu Pahat, and to predict the temperature measurements for the next-day ahead. Simulations for the comprehensive evaluation of the JPSN were presented, and the evaluation covering several performance criteria: the NMSE, MSE, SNR, and number of epoch were discussed. Experimental results of JPSN were compared with the ordinary PSNN and the MLP. Results obtained from each model were presented, and on the whole, the proposed JPSN has shown to outperform the ordinary PSNN and MLP on the prediction errors and convergence time.

7. Acknowledgment

The work of R. Ghazali is supported by Ministry of Higher Education (MOHE) Malaysia, under the Exploratory Research Grant Scheme (ERGS).

8. References

Baboo, S. S. & Shereef, I. K. (2010). An Efficient Weather Forecasting System using Artificial Neural Network. *International Journal of Environmental Science and Development,* Vol.1, No.4, pp.321-326.

Barry, R. & Chorley, R. (1982). *Atmosphere, Weather, and Climate*: Methuen.

Chang, F.-J.: Chang, L.-C., Kao, H.-S. & Wu, G.-R. (2010). Assessing the effort of meteorological variables for evaporation estimation by self-organizing map neural network. *Journal of Hydrology, 384*(1-2), 118-129.

Cybenko, G. (1989). Approximation by Superpositions of a Sigmoidal Function. *Signals Systems, 2 (303)*, pp. 14.

Ghazali, R. & al-Jumeily, D. (2009). Application of Pi-Sigma Neural Networks and Ridge Polynomial Neural Networks to Financial Time Series Prediction. In *Artificial Higher Order Neural Networks for Economics and Business* (pp. 271-293). Hershey, New York: Information Science Reference.

Ghazali, R.: Hussain, A. & El-Deredy, W. (2006). Application of Ridge Polynomial Neural Networks to Financial Time Series Prediction. *In Proceedings of the International Joint Conference on Neural Networks, IJCNN 2006,* part of the IEEE World Congress on Computational Intelligence, WCCI 2006, Vancouver, BC, Canada, 16-21 July 2006. pp. 913-920, IEEE, 2006.

Ghazali, R.: Hussain, A. J. & Liatsis, P. (2011). Dynamic Ridge Polynomial Neural Network: Forecasting the univariate non-stationary and stationary trading signals. *Elsevier: Expert Systems with Applications.* 38, pp. 3765-3776.

Hussain, A. J. & Liatsis, P. (2002). Recurrent Pi-Sigma Networks for DPCM Image Coding. *Neurocomputing, 55*, pp. 363-382.

Ibrahim, D. (2002). Temperature and its Measurement. In *Microcontroller Based Temperature Monitoring & Control* (pp. 55-61). Oxford: Newnes.

Jordan, M. I. (1986). *Attractor Dynamics and Parallelism in a Connectionist Sequential Machine.* Paper presented at the Proceedings of the Eighth Conference of the Cognitive Science Society, New Jersey, USA.

Lorenc, A. C. (1986). Analysis Methods for Numerical Weather Prediction. *Quarterly Journal of the Royal Meteorological Society, 112*(474), pp. 1177-1194.

Mielke, A. (2008). Possibilities and Limitations of Neural Networks. Retrieved August 24, 2009, from http://www.andreas-mielke.de/nn-en-1.html

Paras, Mathur, S., Kumar, A. & Chandra, M. (2007). *A Feature Based Neural Network Model for Weather Forecasting.* Paper presented at the Proceedings of World Academy of Science, Engineering and Technology.

Radhika, Y. & Shashi, M. (2009). Atmospheric Temperature Prediction using Support Vector Machines. *International Journal of Computer Theory and Engineering, 1*(1), pp. 55-58.

Rumelhart, D. E.: Hinton, G. E. & Williams, R. J. (1986). Learning Representations by Back-Propagating Errors. *Nature, 323 (9)*, pp. 533-536.

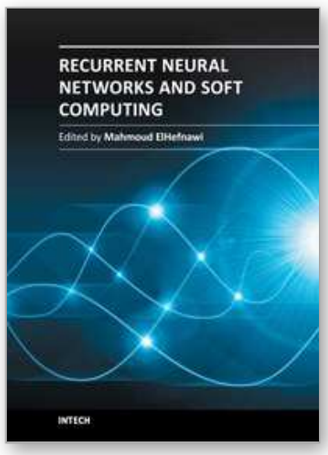
Shin, Y. & Ghosh, J. (1991-a). The Pi-Sigma Networks: An Efficient Higher-order Neural Network for Pattern Classification and Function Approximation. *Proceedings of International Joint Conference on Neural Networks, Vol.1*, pp.13-18.

Shin, Y. & Ghosh, J. (1991-b). Realization of Boolean Functions using Binary Pi-Sigma Networks. In Kumara & Shin, (Ed.), *Intelligent Engineering Systems Through Artificial Neural Networks, Dagli*, (pp. 205-210). ASME Press.

Song, G. (2008). Visual Cryptography Scheme Using Pi-sigma Neural Networks. *Proceedings of the* pp. 679-682.

IntechOpen

IntechOpen



Recurrent Neural Networks and Soft Computing
Edited by Dr. Mahmoud ElHefnawi

ISBN 978-953-51-0409-4
Hard cover, 290 pages
Publisher InTech
Published online 30, March, 2012
Published in print edition March, 2012

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Rozaida Ghazali, Noor Aida Husaini, Lokman Hakim Ismail and Noor Azah Samsuddin (2012). An Application of Jordan Pi-Sigma Neural Network for the Prediction of Temperature Time Series Signal, Recurrent Neural Networks and Soft Computing, Dr. Mahmoud ElHefnawi (Ed.), ISBN: 978-953-51-0409-4, InTech, Available from: http://www.intechopen.com/books/recurrent-neural-networks-and-soft-computing/an-application-of-jordan-pi-sigma-neural-network-for-the-prediction-of-temperature-time-series-signa

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2012 The Author(s). Licensee IntechOpen. This is an open access article distributed under the terms of the Creative Commons Attribution 3.0 License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

IntechOpen

IntechOpen