Industrial Engineering Journal
ISSN: 0970-2555
Volume : 50, Issue 9, No. 1, September : 2021

Integrating Trimmed-Winsorized Means with Dempster-Shafer Evidence for Neural Network Time Series Prediction

[1]Dharavath Bhadru, [2]Dr. Udumala Fernandes Dimlo, [3]Gorgie Sangeetha

[1]Associate Professor, Department of CSE, Narsimha Reddy Engineering College, Secunderabad, Telangana
[2] Professor, Department of CSE, Narsimha Reddy Engineering College, Secunderabad, Telangana
[3]Assistant Professor, Department of CSE, Narsimha Reddy Engineering College, Secunderabad, Telangana

Abstract: All tasks involving machine learning and design recognition can be completed using artificial neural networks. In contrast to traditional approaches for time series evaluation, a synthetic neural system needs direction for the time series data and may be used to a broad variety of issues. For the purpose of using a forecasting system, we tested to determine the optimal criterion. The recently created artificial neural networks exhibited increased prediction accuracy under a well-known feature selection method.

1. INTRODUCTION

Frequency range and the automatic coupling of time series often serve as the foundation for the conventional approach's [2] criteria. Artificial neural networks are used for temporal sequence analysis, however it all depends on the data that were found. Given that multiple layer feed forward systems with sufficient amounts of hidden models and one or more hidden layers are capable of approximating function [8, 11] to any significant extent, a *synthetic neural system is powerful enough to represent any kind of time series. In certain APL code outlines, the internal and external products of matrices and vectors may be used to apply the forward and backward routes of the entirely linked feed forward network. We decided to employ a fully connected, layered, feed forward artificial neural network with a single hidden layer and the back-propagation learning method for the application. The definitions and computations that apply are briefly reviewed in the section that follows.

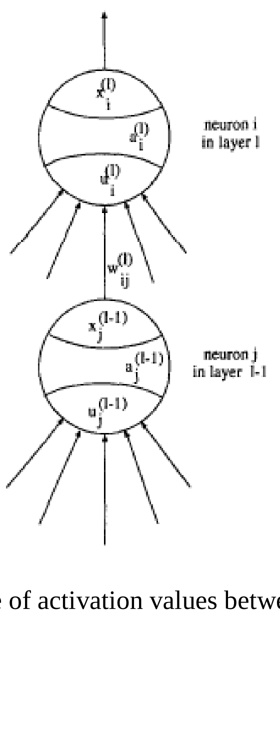


Figure 1: Exchange of activation values between neurons

1.1 A Very Short Introduction to Artificial Neural Networks

Artificial neural nets comprise a large number of little processing units (referred to as processing factors or neurons) organized in layers. These basic principles of the approach for comprehending in neural nets were established by [10]. Each layer is identified by its index, which is l=O,...,M. The following is how the processing components are connected: Processing components of neighbouring tiers are allowed to converse with one another. A level of neurons cannot communicate. There is a certain activation level a for every neuron. The network processes data from the service level exchange between neurons (see image 1):The output value of the i -th neuron in sheet l is denoted by $\chi_i^{(l)}$,It is calculated with the formula

$$\chi_i^{(l)} = g(a_i^{(l)})$$

where $g(\cdot)$ is a monotone growing function. For our examples, we use the function $g(y) = \frac{1}{1+e^{-y}}$ (the “squashing function”). The commencement level $a_i^{(l)}$ of the neuron i in layer l is calculated by

$$a_i^{(l)} = f(u_i^{(l)})$$

Where $f(\cdot)$ is the activation function (in our case the characteristics function is used).

The net input $u_i^{(l)}$ of neuron i in layer l is projected as

$$u_i^{(l)} = (\sum_{j=1}^{n^{(l)}} \omega_{ij}^{(l)} \chi_j^{(l-1)}) - \theta_i^{(l)}$$

Where $\omega_{ij}^{(l)}$ is the weight of neuron j in layer $l-1$ connected to neuron i in layer l, $\chi_j^{(l-1)}$ the output of neuron j in layer $l-1$. $\theta_i^{(l)}$ is a bias charge that is subtracted from the sum of the weighted activations.

The computation of the network position starts at the input layer and ends at the output layer. The input vector I initializes the establishment levels of the neurons in the input layer:

$$a_i^{(0)} = i^{th} \text{ element of } I$$

The identity function is for the input layer. Another layer of the system receives the service level of one coating. Then, the back-propagation learning rule modifies the dumbbells between the nerves. By switching the dumbbells, the artificial neural network determines the enter/output mapping and minimizes the difference between the desired and actual output vector.

When doing network training, the simulator may be divided into two main periods: The input tier of the network is presented with an input/output pair that has been selected at random. The service then spreads to the system's hidden layers and, finally, to the output level. The true output vector for the next operation is different from the intended effect. The incorrect values spread from your output level back to the hidden layers. For a recent demo of the exact same idea, the weights are changed to reduce mistake.

The weight change in layer l at time υ is calculated by

$$\Delta\omega_{ij}^{(l)}(\upsilon) = \eta\delta_i^{(l)}\chi_j^{(l-1)} + \alpha\Delta\omega_{ij}^{(l)}(\upsilon-1)$$

where $\eta \in (0,1)$ is the learning rate and $\alpha \in (0,1)$ is the impetus. Both are kept steady during learning. $\delta_i^{(l)}$ is distinct

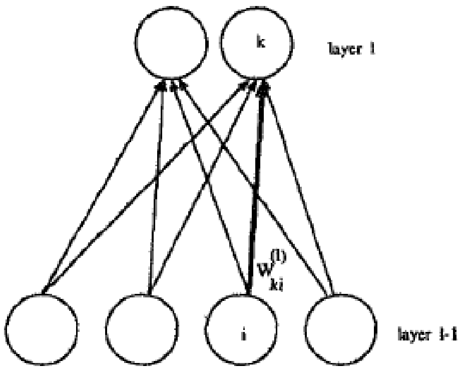


Figure 2: Weight adaptation among two neurons

1. for the output layer $(l = L)\delta_i^{(L)}$ as

$$\delta_i^{(L)} = (d_i - \chi_i^{(L)})g^{'}(u_i^{(L)})$$

Where $g^{'}(u_i^{(L)})$ is the incline of the output function at $u_i^{(L)}$ .The gradient of the output function is always optimistic.

The formula can be elucidated as follows: When the output $\chi_k^{(l)}$ of the neuron i in layer l is too small, $\delta_k^{(l)}$ has a negative assessment. Hence the output of the neuron can be raised by increasing the net input $u_k^{(l)}$ by the following change of the burden values:

If $\chi_i^{(l-1)} > 0,$ then increase $\omega_{ki}^{(l)}$

If $\chi_i^{(l-1)} < 0,$ then decrease $\omega_{ki}^{(l)}$

The regulation applies vice versa for a neuron with an output charge that is too high (see figure 2).

2. for the output layer $(l < L)\delta_i^{(l)}$ is defined by:

$$\delta_i^{(l)} = g^{'}(u_i^{(l)})(\sum_{k=1}^{n^{(l+1)}} \delta_k^{(l+1)}\omega_{ki}^{0l+1})$$

Finally the weights of layer l are adjusted by

$$\omega_{ij}^{(l,new)} = \omega_{ij}^{(l)} + \Delta\omega_{ij}^{(l)}(\upsilon)$$

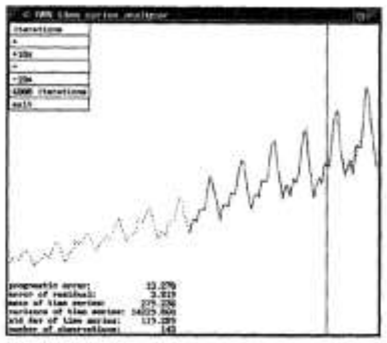
Industrial Engineering Journal

ISSN: 0970-2555

Volume : 50, Issue 9, No. 1, September : 2021

2. IMPLEMENTATION

Core Duo CPU and 4GB of RAM from an additional workstation were used to execute the time series modeling and forecasting categorization. There are two primary parts to the system: a collection of APL routines that limit the replication run results, neural network and log settings, and other data to APL component files. Using a graphical user interface, the user may move about while running simulations and compare the real-world time series to the one generated by the neural network.



```
prognostic error:              13.270
error of residual:              3.319
mean of time series:          279.298
variance of time series:    14225.868
std dev of time series:       119.289
number of observations:           143
```

Figure 3: User interface of the forecasting system

Figure 3 shows a display dump of the interface: the strong line denotes the output of the network, while the damaged line shows the real-time sequence. The vertical bar in the top-right corner of number 3 allows the user to select a perspective of the network's forecasting capability at various countries throughout the entire learning stage. The outlook data is separated from the historic data, the network's training set, by this vertical bar.

The results of the simulator runs may be compared and evaluated by looking at the record files.

By looking at the record files of the simulator runs, current and previous outcomes could be assessed and compared.

3. MODELING

3.1 The Training Sets

We used two well-known time sequences from [2] as a test bed for our forecasting program: the monthly sums of international flight passengers (in thousands) from 1949 to 1960 (see number 4), and the daily closing expenses of share inventories (see picture 5). Some of the two-time series are given characteristics by Stand 1: u g the mean, is the usual deviation, and n the number of finds.

Time Series	σ	μ	n
Share Price	84.11	478.47	369
Grid power unit Price	119.55	280.30	144

Table 1: Properties of time series

The next section is disturbed with the question: How can a neural network learn a time series?

3.2 The Algorithm

The time series Xl,..... Xm are represented by the neural network as many mappings from an input vector to an output value (see figure 6). This method was introduced by [4]. A number of adjacent time series data points are used as service levels for the entry level models (the input window Xt_s, X,-S+,,.., X,) and are scheduled to the interval [0,1]. The value of the result unit and the value of the time sequence at time t 1 are now evaluated to calculate the problem beneficial for that back-propagation learning technique. This error also affects the connections between hidden and output level as well as those between concealed and outcome level. Following the updating of all dumbbells, one demonstration is complete. The back-propagation approach often requires that representations of the input set (referred to as one epoch) be shown several times while training a sensory network.

The representations were presented in a random order to aid in your comprehension of time series data: According to [4], using a random location for each portrayal's input window guarantees improved system performance and avoids local minima.

The selection of the best recommendations for the learning criteria and the best topology for the prediction network are the main topics of the next part.

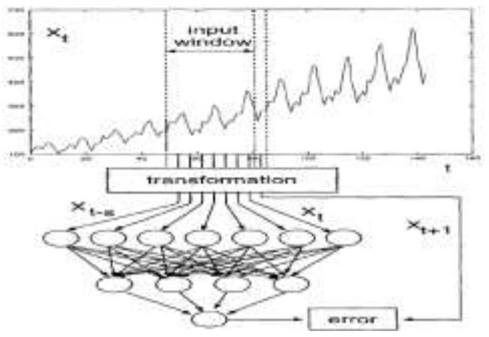


Figure 3: Learning a Time Series

3.3 Network Parameters

The artificial neural network's learning rate (0–1), a scaling factor that instructs the knowledge algorithm how strongly the weights of the connections should be increased for a certain mistake, was chosen so that it could be examined in more detail. The knowledge process may be accelerated by using a higher, but if it is set too high, the algorithm will "step over" the weights that are most beneficial. Throughout all presentations, the learning rate is consistent.

• The dynamism The grade drop of the weights is also impacted by the momentum limitation a (0 1): The impetus term, which maintains the direction of the preceding step [7], is included to stop each association from rapidly responding to any little modification in the solution space, preventing the plunge into local minima. Throughout all presentations, the momentum word remains constant.

• The network topology, which includes the quantity of input and the quantity of hidden units.The number of input units indicates how often the neural network "looks into the past" while making predictions about the future; the number of input units varies depending on the input window's size.

One hidden layer has been shown to be sufficient to approach continuous function [8], although the precise number of hidden units required is "not known in general [7]". Other attempts at using artificial neural networks for time series analysis indicate workable network topologies of 8-8-1, 6-6-1 [CMMR9Z], and 5-5-1[13] (number of neurons in the input-hidden output layer).

We conducted a variety of tests to examine the distribution of these parameters: These parameters were gradually altered in subsequent network runs to see how they affected the network's modeling and forecasting skills.

To evaluate the modeling and forecasting quality of our system, we utilized the following terms: Regarding a time series,

$$s_m = \sqrt{\frac{\sum_{i=1}^{n} (X_i - X_i)^2}{n}}$$

Where r is the number of predicting periods and Xi is the approximate artificial neural network for phase i. The error Sf (equation 2) assesses the neural network's forecasting capabilities over a forecast period of length r. The error sm (equation 1) analyzes the neural network's capacity to imitate the known information set. In our research, we used r = 20.

COMPARISON BETWEEN DIVERGENT FEATURE SELECTION MODELING'S

We compared results gained under feature selection methods called Dempster–Shafer evidence theory and trimmed-Winsorized means. The execution of the procedure uses the applications explained by Jenkins and Box in Component V of their classic [2].The approach is known as an autoregressive integrated moving average method of order (p,d,q). It's described by the equation

$$a(z)\nabla^d X_t = b(z)U_t$$

where X_t stands for in time ordered values of a time series, t =1, . . . , n for n observations. U_t is a sequence of random values called "white noise" process. The backward difference operator V is defined as

$$\nabla X_t = X_t - X_{t-1} = (1-z)X_t$$

We fitted these feature selection models for each time series and let it predict the next 20 observations of the time series. The last 20 observations were dropped from the time series and used to calculate the prediction error of the models.

The following feature selection models were calculated for the power unit price time series (after a logarithmic transformation):

$(1 - z)(1 - z^{12}) X_t = (1 - 0.24169,2 - 0.47962.z^{12}) U_t$ and for the Share time series:

$(1 - z) X_t = (1 - 0.10538z) U_t$

In the forecast errors for the man-made neural network (ANN), the artificial neural network utilizing the logarithmic and change (ANN record,) and the feature selection models opted are compared: The artificial neural network utilizing the logarithmic and changed time series out-performed for both time series, whereas the "simple" artificial neural network expected more precisely just for the Shares time series. This behavior could be described as follows: the bigger data selection of the flight passenger time series results in a reduction in precision for the untransformed in place set. Logarithmic and differencing transformations helped to remove the tendency and planned the time series data in to a smaller variety.

Time series	Dempster–Shafer evidence theory	Trimmed-Winsorized means
power unit prices	20.97	18.72
Share price	10.97	11.35

Table 2: Forecasting errors for ANN under Dempster–Shafer evidence theory and Trimmed-Winsorized means

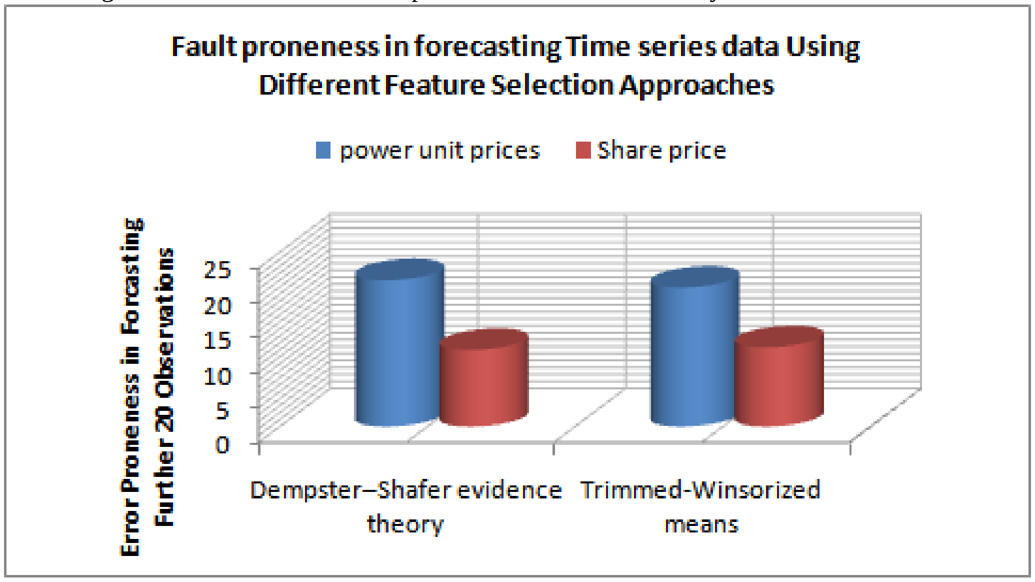


Figure 1: Fault proneness of ANN under Dempster–Shafer evidence theory and Trimmed-Winsorized means

ANN Code

These functions present a complete implementation of the definitions given in the introduction. The main function NET learns the well known XOR (exclusive or) mapping. For reasons of simplicity, the input-output mapping of the XOR function is coded into the function NET.

Conclusion and Further Work

We've offered a predicting program for univariate time sequence that utilizes artificial neural networks. These processing devices proved themselves to be feasible options to classic approaches. The machine may be utilized along with other approaches for time series evaluation or just as a standalone device.

More work may contain the assessment with additional time series analysis approaches, development of hybrid approaches that blend the power of standard methods with artificial neural networks and the use of our program to multivariate time series.

References

[1] M. Alfonseca. Advanced applications of APL: logic programming, neural networks and hypertext. IBM Systems Journal, 30(4):543—553, 1991.
[2] George E. P. Box and Gwilym M. Jenk¬ins. Time Series Analysis - forecasting and control. Series in Time Series Anal-ysis. Holden-Day, 500 Sansome Street, San Franciso, California, 1976.
[3] E. Chatfield. The Analysis of Time Series. Chapman and Hall, New York, fourth edi¬tion, 1991.

[4] Kanad Chakraborty, Kishan Mehrota, Chilukuri K. Mohan, and Sanjay Ranka. Forecasting the Behaviour of Multivariate Time Series Using Neural Networds. Neu¬ral Networks, 5:961-970, 1992.

[5] Dyadic Systems Limited, Riverside View, Basing Road, Old Basing, Basingstoke, Hampshire RG24 OAL, England. Dyalog Apl Users Guide, 1991.

[6] Richard M. Evans and Alvin J. Surkan. Relating Numbers of Processing Elements in a Sparse Distributed Memory Model to Learning Rate and Generalization. ACM APL Quote Quad, 21(4):166-173, 1991.

[7] John Hertz, Anders Krogh, and Richard G. Palmer. Introduction to the Theory od Neu¬ral Computation. Addison Wesley, Red¬wood City, California, 1991.

[8] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer Feedforward Networks are Universal Approximators. Neural Networks, 2:359-366, 1989.

[9] Howard A. Peele. Teaching A Topic in Cy¬bernetics with APL: An Introduction to Neural Net Modelling. ACM APL Quote Quad, 12(l):235-239, 1981.

[10] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning repre¬sentations by back-propagating errors. Na¬ture, 323(9):533-536, October 1986.

[11] Hava Siegelmann and Eduardo D. Sontag. Neural Nets Are Universal Computing De¬vices. Technical Report SYSCON-91-08, Rutgers Center for Systems and Control, May 1991.

[12] Alexei N. Skurihin and Alvin J. Surkan. Identification of Parallelism in Neural Net¬works by Simulation with Language J. ACM APL Quote Quad, 24(l):230-237, 1993.

[13] Halbert White. Economic prediction us¬ing neural networks: the case of ibm daily stock returns. In Proceedings of the IEEE International Conference on Neural Net¬works, pages II—451—11—459, 1988.

[14] G. P. Zhang, B. E. Patuwo, and M. Y. Hu, “Forecasting with artificialneural networks: The state of the art,” Int. J. Forecast., vol. 14, pp.35–62, Mar. 1998.

[15] S. Crone and P. Grafeille, “An evaluation framework for publications onartificial neural networks in sales forecasting,” in Proc. Int. Conf. Artif.Intell., Las Vegas, NV, Jun. 2004, pp. 221–227.

[16] T. Hill, M. O’Connor, and W. Remus, “Neural network models for timesseries forecasting,” Manage. Sci., vol. 42, no. 7, pp. 1082–1092, Jul.1996.

[17] N. R. Swanson and H. White, “Forecasting economic time seriesusing flexible versus fixed specification and linear versus nonlineareconometric models,” Int. J. Forecast., vol. 13, no. 4, pp. 439–461, 1997.

[18] W. Zhang, Q. Cao, and M. J. Schniederjans, “Neural network earningper share forecasting models: A comparative analysis of alternativemethods,” Decision Sci., vol. 35, no. 2, pp. 205–237, 2004.

[19] S. Heravi, D. R. Osborn, and C. R. Birchenhall, “Linear versus neuralnetwork forecasts for European industrial production series,” Int. J.Forecast., vol. 20, no. 3, pp. 435–446, Jul.–Sep. 2004.

[20] L. J. Callen, C. C. Kwan, P. C. Yip, and Y. Yuan, “Neural networkforecasting of quarterly accounting earnings,” Int. J. Forecast., vol. 12,no. 4, pp. 475–482, 1996.

[21] M. Adya and F. Collopy, “How effective are neural networks at forecastingand prediction? A review and evaluation,” Int. J. Forecast., vol.17, nos. 5–6, pp. 481–495, Nov. 1998.

[22] M. Nelson, T. Hill, T. Renas, and M. O’Connor, “Time series forecastingusing NNs: Should the data be deseasonalized first?” J. Forecast., vol.18, no. 5, pp. 359–367, 1999.

[23] G. P. Zhang and D. M. Kline, “Quarterly time series forecasting withneural networks,” IEEE Trans. Neural Netw., vol. 18, no. 6, pp. 1800–1814, Jun. 2007.

[24] S. D. Balkin and J. K. Ord, “Automatic neural network modeling forunivariate time series,” Int. J. Forecast., vol. 16, no. 4, pp. 509–515,2000.