Journal of Engineering and Applied Sciences 14 (13): 4491-4500, 2019
ISSN: 1816-949X
© Medwell Journals, 2019

Neural Networks in Business Applications

Mohammed Khawwam Ahmed
Department of Shariaa Computer Science Information Systems, College of Islamic Sciences,
University of Samarra, Samarra, Salahuldeen, Iraq

Abstract: Neural networks originally inspired from neuroscience provide powerful models for statistical data analysis. Their most major feature is their ability to "learn" dependencies based on a finite number of observations. In the context of neural networks the term "learning" means that the knowledge acquired from the samples can be generalized to as yet, sense observation. In this sense, a neural network is often called a learning machine. As such, neural networks might be considered as a symbol for an agent who learns dependencies of his environment and thus, infers strategies of behavior based on al limited number of observations. In this contribution, however, the researcher does not want to abstract from the biological origins of neural network technique but present it as a purely mathematical model and also its statistical applications.

Key words: Artificial Neural Network (ANN), neuron, transfer functions, hidden lopper supervised training, momentum factor, training tolerance, backdrop, galion, cross-validation, jackknifing and bootstrapping

INTRODUCTION

Artificial ventral Network (ANN) is Artificial Intelligence (AI) methods structure according to human brain. From the beginning of their presence in science ANNs are being investigated according to two different approaches. First, the biological aspect explores ANNs as simplied simulations of human brain and uses them to test the hypothesis about human brain functioning. The second approach treats ANNs as technological systems for complex information processing (Zahedi, 1993).

The reason is why ANNs often outperform classical statistical methods lies in their barites to analyze incomplete, noisy data, to deal with problems that have to clear cut solution and to learn on historical data. Because of those advantages, ANNs have shown success in predictions of financial data series that have high degree of instability and fluctuations. Among the disadvantages of ANNs, it is necessary to mention the lack of tests of statistical magnificence of the ANN Models and parameters estimated (Refenes *et al*., 1997). In despite of those disadvantages many research results show that neural networks can solve almost all problems more efficiently than traditional modeling and statistical methods. It is mathematically proven that three layer neural networks having randomly transfer function are capable to approximate any nonlinear function (Masters, 1995).

History of neural networks: The first major research conducted on neural networks was done by McCulloch and Pitts (1943) by creating a learning algorithm for a model known as the perception. Inters in neural network research increased more and more until 1969 when a book by Minsk and Paper indicated that neural networks could be used to judge only a limited variety of functions. This facer creates no practical difficulty but an increase of interest resulted only after the development of the Hopfield Model, the back propagation algorithm and the exponential advances in computation technology (Ahmadian, 1995).

MATERIALS AND METHODS

Basic principles of neural networks: ANNs consist of two or more layers or groups of processing elements called neurons. The term neuron denotes a basic unit of a neural network model attended for data processing. Neurons are connected into a network in the way that the output of each neuron represents the input for one or more other neurons, According to its direction, the connection between neurons can be either one-directional or bi-directional and due to its intensity the connection can be active or inactive, neurons are grouped into layers. There are three main types of layers: input, hidden and output. The input layer receives input data from external environment and sends it to the hidden layer (s). In the hidden layer the information is processed and sent to the output layer neurons where the information is processed and sent to the output layer neurons where the network output is compared to the desired (actual) output and the network error is computed. The error information then flows backward through the network and the values of connection weights between

the neurons are adjusted using the error term. The process is repeated in the network for a number of iterations that is necessary to achieve the output closest to the desired (actual) output. Connection weight is the strength of connection between two neurons. If, for example, neuron j is connected to neuron i, W_{ji} denotes the connection weight from neuron j to neuron i(W_{ij} is the weight of reverse connection from neuron i to neuron j). If neuron i is connected to neurons called 1, 2, ..., N. Their weights are saved in the variables W_{i1}-W_{ni}. A neuron receives as many inputs as there are input connections to that neuron and produces a single output to other neurons according to transfer function. The process of neural network design consists of four steps (Zahedi, 1993):

- Display neurons in various layers
- Determining the type of connections between neurons (inter-layer) and intra-layer connection
- Determining the way neurons receives input and produces output
- Determining the learning rule for adjusting the connection weights

The requirements to construct ANN are as follows:

- Data sets
- Type of connection between neurons
- Connection between input and output data
- Input and transfer functions
- Type of learning
- Learning parameters

Data sets: In ANN methodology, the sample is often subdivided into "training" "validation" and "test" sets. The terms "validation" and "test" sets are often confused. The following definitions are taken from Ripley (1996).

Training set: The training data set is a set of examples (observations) can be used to fit the parameters (i.e., weights) of the classfier.

Validation set: A set of examples (observations) used to tune the parameters (i.e., architecture, not weights) of a classifier, for example, to choose the number of hidden units in a neural network.

Test set: A set of examples used only to assess the performance (generalization) of a fully-specified classifier.

Bishop (1995), another essential reference on neural networks, provides the following explanation: since, our goal is to find the network having the best performance on new data, the simplest approach to the comparison of different networks is to evaluate the error function using data which is independent of that used for training. Various networks are trained by minimization of an appropriate error function defined with respect to a training data set. The performance of the networks is then compared by evaluating the error function using an independent validation set and the network having the smallest error with respect to the validation set is selected. This approach is called the called the hold out method. Since, this procedure can itself lead to some over fitting to the validation set, the performance of the selected network should be confirmed by measuring its performance on a third independent set of data called a test set.

By Bishop (1995), the test set is never used to choose among two or more networks, so that, the error on the test set provides an unbiased estimate of the generalization error. Any data set that is used to choose the best of two or more networks is a validation set and the error of the chosen network on the validation set is biased.

There is a problem with the usual distinction between training and validation sets. Some training approaches, such as early stopping, require a validation set, so, the validation set is used for training. Other approaches such as maximum likelihood, do not require a validation set. So, the "training" set for maximum likelihood might merge both the "training" and "validation" sets for early stopping. Cherkassky *et al.* (2001) has suggested the term "design" set can be used for cases that are used to adjust the weights in a network.

Type of connection between neurons: Connections in the network can be realized between two layers (inter-layer connections) and between neurons in one layer (intra-layer connections).

Inter-layer connections: There are various types of inter-layer connections (Zahedi, 1993) which can be as follows:

Fully connected: Each neuron in the first layer is connected to each neuron in the second layer.

Partially connected: A neuron in the first layer does not have to be connected to all neurons in the second layer.

Feed-forward: Connection between neurons is one-directional, neurons in the first layer send their output to the neurons in the second layer but they do not receive any feedback.

Bi-directional: There is a feedback when the neurons from the second layer send their output back to the neurons in the first layer.

Hierarchical: The neurons of a lower layer may only communicate with neurons of the next level of layers.

RESULTS AND DISCUSSION

Resonance: Two-directional connection where neurons continue to send information between layers until a certain condition is satisfied.

Intra-layer connections: Connections between neurons in one layer (intra-layer) connected.

Recurrent: Neurons in one layer fully or partially connected. The connection is realized in a way that neurons send their outputs to each other neuron after they receive their inputs from another layer. The communication continues until neurons reach a stable condition. When the stable condition is reached, neurons are allowed to send their output to the next layer.

On-center/off-surround: In this connection a neuron in one layer has an active connection toward itself and toward the neighbor neurons but an inactive connection toward other neurons in the other layers.

Connection between input and output data: ANNs can also be distinguished according to the connection between input and output which can be:

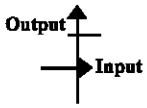
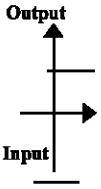
Auto-associative: Input vector is the same as output vector.

Hetero-associative: Output vector differs from the input vector. Auto associative networks are used in signal processing, noise filtering and to discover the patterns of input data.

Input and transfer functions: The basic principles of ANN functioning are the input function and transfer (output) functions.

Input (summation) functions: When a neuron receives the input from the previous layer, the value of its input is computed according to an input function, usually called a "summation" function. The simplest summation function for the neuron I is determined by multiplying the output sent by the neuron j to the neuron I (denoted as output j) with the connection weight between neurons I and j, then calculate the summation of those multiplications for all j neurons connected to neuron I or:

Table 1: Most frequently used transfer functions and their graphs

Functions	Output	Graphs
Step function	$Output_I = \begin{cases} 0 & \text{when } input_I \le T \\ 1 & \text{when } input_I > T \end{cases}$	
Sign function	$Output_I = \begin{cases} 1 & \text{when } input_I > 0 \\ 0 & \text{when } input_I = 0 \\ -1 & \text{when } input_I < 0 \\ 1 \\ 1 \end{cases}$	
Sigmoid	$Output_I = 1/1+e^{input_I}$	
Hyperbolic tangent function	$Output_I = E^{input_I} - e^{-input_I} / E^{input_I} + e^{-input_I}$	
Linear function	$Output_I = g*input_I$	

$$Input_I = \sum \left(W_{ji} * output_j \right)$$

where, n is the number of neurons in the layer that sends its output received by the neuron i. In other words, $input_I$ of a neuron I is the sum of all weighted outputs that arrive into that neuron. Input values can be normalized to an interval (usually (0, 1) or (-1.1) to avoid the extreme influence of high-valued inputs. Therefore, normalization is recommended in most of neural networks (Cross *et al*., 1995).

Output (transfer) functions: After receiving the input according to the summation function the output of a neuron is computed and sent to the next layer neurons it is connected to. The output of neurons is computed according to the transfer function. The most frequently used transfer functions are:

- Step function
- Sign function
- Sigmoid function
- Hyperbolic-tangent function
- Linear function

Table 1 shows the most frequently used transfer functions, the formula for each and the corresponding graph:

- The output in the step function is computed according to the above formula where T is the threshold and T∈R
- A sign function is a special form of the step function, when the Threshold T = 0. The sign function was used in the first neural network which is called perception by Rosenblatt (1958)

- A sigmoid (or logistic) function is one of the mostly used transfer functions in ANNs. The function results are continuous values in (0, 1)
- The hyperbolic tangent function is a special form of a sigmoid function. The graph of a hyperbolic tangent function is similar to the graph of the sigmoid function, only the value interval is her (-1, 1)
- A linear function has the form:

$$Output_1 = g*unput_1$$

where, g is the inverse of the threshold g = 1/T.

Type of learning: "Leaning" is the process of calculating the weights among neurons in a network (Zahedi, 1993). Weights are an important factor that determines the vale of a neuron input and indirectly affects a neuron output. There are two main types of learning in a network:

- Supervised
- Unsupervised

The difference between those two types of learning is in availability of known output in the training sample. In supervised leaning the set of training data consists of previous cases with known input and output values. The neural network system receives the actual output, computes the error and adjusts the weights according to the error.

On the other hand, the actual outputs are not known in unsupervised learning. Inputs are available to the network and the weights cannot be adjusted based on the actual output. This type of learning is commonly used for pattern recognition problems and clustering. Every ANN goes through three stages:

Learning (training) stage: Network learns on the training sample, the weights are being adjusted in order to minimize the objective function (for example, RMS-Root Mean Square error).

Testing stage: Network is tested on the testing sample while the weights are fixed.

Operator (recall) stage: ANN is applied to the new cases with unknown results (weights are also fixed).

Learning rules: A learning rule represents the formula that is used in ANN to adjust the connection weights among neurons. Among various learning rules developed, so far, four of them are most commonly used:

- Hebb's rule
- Hopfield rule
- Delta rule

Hebb's rule: The first and the best known learning rule was introduced by Hebb. The description appeared in his book the Organization of Behavior (1949).

His basic rule is: if a neuron receives an input from another neuron and of both are highly active (mathematically have the same sign), the weight between the neurons should be increased and vice versa.

Hopfield rule: It is similar to Hebb's rule with the exception that it specifies the magnitude of the strengthening or weakening. It states, "If the desired output and the input are either active or inactive, increment the connection weight by the learning rate" (Hegde *et al*., 1988).

The delta rule: "Standard back prop" is known as the generalized delta rule. This training algorithm was developed by Rummelhart (1986). It remains the most widely sed supervised training method for neural nets. The generalized delta rule (including momentum factor) is called the "heavy ball method" in the numerical analysis literature (Polyak, 1987). It is based on the simple idea of continuously modifying the strengths of the input connections to reduce the difference (the delta) between the actual output value and the predicted output of a processing element. This rule changes the weights in the way that minimizes the mean squared error of the network. This rule is also referred to as the Least Mean Square (LMS) learning rule.

The way that the delta rule works is that the delta error n the output layer s transformed by the derivative of the transfer function and is then used in the previous neural layer to adjust input connection weights. In other words, this error is back-propagated into previous layers one layer at a time. The process of back-propagating the network errors continues until the first layer is reached.

Learning parameters: Training a neural network is fine tuned with the following three training parameters:

- Learning rate
- Momentum factor
- Training tolerance

Learning rate: Learning rate limits or expands the level of weight adjustments in a training loop. A high learning rate reacts can make networks unstable. If the learning rate is too high the network's prediction ability will be poor.

However, if the learning rate is too low, the network training time is increased (Garson, 1991). A high learning rate is useful to accelerate learning until the weight adjustments be available. However, the higher learning rate increases the risk that the weight search comes over a minimum error condition which could case ANN learning procedures to fail.

The agreement among researchers that adaptive learning rates steady risk of failure and accelerate the training process. A constant learning rate is inefficient because ANN needs a gradually declining rate to converge on a solution and a low rate from the beginning is time consuming.

Momentum factor: Momentum factor describes the proportion of the weight change that I added to each subsequent weight change. Low momentum causes weight fluctuation and instability, preventing the network from learning. High momentum makes neural network adaptability poor. For table neural network, the momentum factor should be kept <1 than one (unity). Momentum factors close to unity are needed to smooth fluctuations when they occur (Garson, 1991).

Training tolerance: Training tolerance is the margin of error acceptable when training target values are compared with the vales generated by the network during the training. A training tolerance factor of zero indicates that network values must exactly match target vales. Zero training tolerance is a reasonable requirement during network training because a high degree of accuracy should be required with the training data. The higher the training tolerance factor is, the more inaccurate the neural network will be (Garson, 1991). Balance is necessary when applying training parameters to a neural network. The training parameters are specific to each network and are determined primarily by trial and error.

Back propagation algorithm-a mathematical approach: Back propagation algorithm was the one that made neural networks a widely used and popular method in various areas of application. Originally developed by Rummelhart (1986), this was the first network with more than one hidden layer. The aim of this algorithm is computing the error at the output layer and back propagating it to each hidden layer such that weights of connections are being adjusted until the input layer is reached (Dhar and Stein, 2012). A single artificial neuron which learns using the back propagation learning algorithm.

Back propagation learning algorithm: The back propagation algorithm seeks to minimize the error term between the output of the neural net and the actual output value. The error term is calculated by comparing the net output to the actual output and is then fed back through the network to adjust the weights then to minimize error (Fahlman, 1988). The process is repeated until the error reaches a minimum value. The network uses Eq. 1 to update the weight W_{ij} from a given Neuron N_1 to the current Neuron N_j:

$$W_{lj}.(1+1) = W_{ij.2} + (\lambda)(\varepsilon W_{ij})(N_1) \quad (1)$$

Where:

t = The number of times the network has been updated
λ = The learning parameter or learning rate
N_1 = The sensitivity of Neuron
W_{ij} = The Weight is represented by εW_{ij}

The total input to a neuron s described n Eq. 2 as follows:

$$S_1 \sum N_1 W_{ij} \quad (2)$$

Where:

S_j = The Sum of all inputs to a neuron
N_1 = The output of the previous Neuron
W_{ij} = The Weight connection between the Ith neuron of the previous layer to the jth neuron in the current layer

This output is then transformed using the logistic activation function which can be expressed as follows:

$$N_j = \frac{1}{1+e^{-sj}} \quad (3)$$

where, N is the total output of neuron j. The overall error for a single pass of the neural network is represented as follows:

$$E = \frac{1}{2}\sum\left[N_j - D_j\right] \quad (4)$$

where, D_j is the desired output of the output neuron j. The error term for the entire network has been calculated, this information is fed back through the network to reduce error. This procedure can be done through 4 steps which can be as follows.

The first step: The error term for each output neuron O_j must be calculated and identify how much the error term changes with respect to a change in each output neuron. This calculation can be expressed as follows:

$$\varepsilon O_j = (N_1 - D_1) \quad (5)$$

The second step: The amount the error term changes as the input is varied to a given output neuron must be calculated. This is done by determining how much Eq. 4 changes when the total input to the neuron (Eq. 2) S changed. This calculation is simplified in Eq. 6:

$$\varepsilon S_i = \varepsilon O_i N_i \left(1 - N_j\right) \tag{6}$$

The third step: The weight adjustment needed for W_{ij} is calculated from a layer below the current layer N_i to the current neuron N_j. The calculation is simplified in Eq. 7:

$$\varepsilon W_{ij} = \varepsilon S_i N_j \tag{7}$$

The fourth step: This operation is continued on neurons in lower layers by allowing neurons in the lower hidden layers to play the role of the output neuron. All errors form all inputs to the hidden layers to play the role of the output neuron. All errors from all inputs to the hidden layer must be summed. Additionally, the error in the hidden neuron is calculated by examining how the error of neurons above the hidden neuron change with respect to changes in the hidden neuron. The simplified equation for calculating the weight update s provided in Eq. 8 where the subscript j represents neurons in the layer above the hidden layer:

$$\varepsilon H_i = \sum \varepsilon S_j W_{ij} \tag{8}$$

where, H_i is the difference between the weight of an iteration and the weight of the next iteration:

$$H_i = W_{ij(1)} - W_{ij(2)}$$

Dhar and Stein (2012) states "In this manner the error of the network is propagated in a reverse repeating methods through the entire network and all of the weights are adjusted to minimize the overall network error".

Overtraining is the universal problem for all types of ANN algorithms. It occurs when the network learns the training sample perfectly but is not able to generalize on the test sample. To come over this problem the following techniques can be used during the training process of a neural network:

- Cross-validation technique
- Jackknife technique
- Bootstrap technique

Cross-validation technique: Cross-validation using the validation sample to determine when to stop learning. The training will continue as long as the error on the validation sample improves. When it does not improve, the training will stop. Such iterative procedure is usually called the "Save best" procedure which alternatively trains and tests the network until the performance of the network does not improve for n number of iterations. After the best network is selected, it is tested on a new test sample to determine its generalization ability.

In k-fold cross-validation, the data can be divided into k subsets of (approximately) equal size. The net is trained k times, each time leaving out one of the subsets from training but using only the omitted subset to computer the error. If k equals the sample size, this is called "leave-one-out" cross-validation. "Leave-v-out" is a more complex version of cross validation that involves leaving but all possible subsets of V cases. The cross validation is quite different from the "split-sample" or "hold-out" method that is commonly used for early stopping in ANNs. N the split sample method. only a single subset (the validation set) is used to estimate the generalization error, instead of k different subsets; i.e., there S no "crossing". The distinction between cross validation and split-sample validation is extremely important because cross-validation is noticeably superior for small data sets as demonstrated by Goutte (1997) in a reply to Zhu and Rohwer (1996), who stated that cross-validation an be applied only for large data sets and it needs more assumptions for small data sets in their study "No free lunch for cross-validation" published in 1996.

Jackknife technique: Leave-one-out cross-validation technique is also easily confused with jackknife technique. Both involve omitting each training case in turn and retraining the network on the remaining subset. But cross-validation is used to estimate generalization error, while, the jackknife is used to estimate the bias of a statistic. In the jackknife, some statistical methods of interest are computed in each subset of the data. The average of these subset statistics is compared with the corresponding statistic computed from the entire sample in order to estimate the bias of the statistic computed from the entire sample. Also jackknife estimate of the standard error of a statistic can be produced. Jackknife technique can be used to estimate the bias of the training error and hence, to estimate the generalization error but this process is more complicated than leave-one-out cross-validation (Ripley, 1993).

Bootstrap technique: Bootstrap technique seems to work better than cross-validation in many case (Efron, 1983). In the simplest form of bootstrapping, instead of repeatedly analyzing subsets of the data, sub samples of the data are repeatedly analyzed. Each sub sample is a random sample

with replacement from the full sample. Depending on what is needed, anywhere from 50-2000 sub samples might be used. There are many more sophisticated bootstrap methods that can be used not only for estimating generalization error but also for estimating confidence bounds for network outputs (Efrom and Tibshirani, 2011). Use of bootstrapping for ANNs is described by Baxt and White (1995) and Tibshirani (1996).

Advantages and limitations of neural
Networks: Neural networks are nonlinear and adaptable and thus can deal very effectively with data sets demonstrating not easily recognized or unknown nonlinearity more effectively than multiple linear regression (Gorr *et al*., 1994; Marquez *et al*., 1991).

When performing ANN regression analysis, the model under examination need not be specified in advance. Thus, neural networks do not require explicit relationships between inputs and outputs to be defined in the data set. This feature eliminates uncertainty and load of model specification in conventional regression methods (Garson, 2012). In a neural network simulation conducted by Marquez *et al*. (1991) the neural network closely approximated the true model when tested on an unspecified model and was superior to linear regression. Similarly, Sarle (1994) found that neural networks produced better results than regression did with a noisy unspecified model.

Neural-network based regression is unconstrained by the typical assumptions of regression analysis. Neural network based analysis methods require no assumptions that independent variables are not correlated, this solving the multi non linearity problem found in multiple linear regression (Cheng and Titterington, 1994). Additionally, neural networks do not assume that the outputs of independent variables are necessarily independent of each other and they do not assume the normal distribution of residuals with constant variance and zero mean. Thus, the flexibility of neural network function allows ANN analysis to overcome many of the limitations caused by the assumption multiple linear regression.

Neural networks are considerably more robust than conventional methods of statistical analysis because of their ability to learn from experience and adjust themselves according to new data examples (Rummelhart, 1986). As new information is added to the network, its effect is balanced proportionally with the quantity of old information affecting the training of the network. Furthermore, data patterns that repeat often are enforced while those that do not are weakened to improve network adaptability.

ANN is also able to model data with outliers because they draw out only the essential features of the data set to make predictions. This is because the activation function limits each neuron's output to a certain range. Marquez *et al*. (1991) found that at the high levels of noise and with a small sample, the neural network model performed better than the noiseless model at dentifying the true model.

Limitations of neural networks: Neural networks are often referred to as "black boxes" because the parameters and coefficients they derive to approximate patterns cannot be easily analyzed or interpreted. Unlike, convent ional regression techniques, ANN lack marginal evaluators which unclear the network from revealing the functional relationships among the variables (Gorr *et al*., 1994). Although, networks often provide better results than multiple linear regression, the method is difficult to interpret. These disadvantages are most evident in explanation research where the underlying relationship among the variables is to be understood.

Another limitation that must be acknowledged is that multiple linear regression is superior when all the assumptions are met, the model is correctly specified and the functional relationship is known (Warner and Misra, 1996). Under these circumstances a well prepared regression model will perform better than a neural network. Model fitting through linear regression is less abstract and computationally intense than neural network model training. Additionally, linear regression performs better with very small samples and is superior at decomposition (Gorr *et al*., 1994).

The relationship between ANN and statistical methods: There is considerable overlap between the fields of neural networks and statistics. Statistics is concerned with data analysis. In eural network terminology, statistical inference means learning to generalize from noisy data. Most neural networks can learn to generalize effectively from noisy data are similar or identical to statistical methods. For example:

Feed forward nets with no hidden layer (including functional-link neural nets and higher-order neural nets) are basically generalized linear models:

- Feed forward nets with one hidden layer are closely related to projection pursuit regression
- Projection pursuit regression
- Projection pursuit regression
- Probabilistic neural nets are identical to kernel discriminant analysis
- Kohonen nets for adaptive vector quantization are very similar to k-means cluster analysis

- Hebbian learning is closely related to principal component analysis

Neural networks applications: There is a growing body of literature based on the compoarison of neural network computing to traditional statistical methods of anlysis. Hertz *et al.* (1991) offer a comprehensive view of neural networks and issues of their comparison to statistics. Hinton (1992) investigates the statistical aspects of neural networks. Weiss and Kulikowski (2007) offer an account of the classification methods of many different neural and statistical models.

The main focus for the artificial neural network technology, n application to the financial and economic fields has so far been data involving variables in non-linear relation. Many economists supported the application of neural networks to different fields in economics (Kuan and White, 1994; Bierens, 2006). Several researchers have examined the application of neural networks to financial markets where the non-linear properties of financial data provide many difficulties for traditional methods of analysis (Ormerod *et al.*, 1991; Altman *et al.*, 1994; Kaastra and Boyd, 1995; Wikowska, 1995). Also, neural networks have been touted as all powerful tools in stock-market prediction (Salchenberger *et al.*, 1992). The additional neural network applications in the financial world may be as follows (Verkooijen, 1996):

- Currency predction
- Bond ratings
- Business failure prediction
- Debt risk assessment
- Bank theft
- Bank failure

The ANN is most commonly used to do one or more of the following statistical techniques:

Classification: Discriminating between two things, based on similarities, e.g., loan applications as good or bad risks (Farago and Lugosi, 1993).

Clustering: Classifying things into groups with similar important attributes, e.g., working out groups of customers who buy the same thing (market segmentation) (Wong and Lane, 1983).

Modeling: People can learn to model relationships from only a few examples, we interpolate between them (in the model) to generalize to new cades or problems (Rissanen, 1978).

Forecasting and prediction: In time-series forecasting, look at patterns for some period back through time to work out the future (Humpier, 1999; Wong, 1991).

Constraint satisfaction and optimization: ANN can be used to realize an object with the satisfaction of some conflicting constraints which is known as operations research (Pearl, 1988; Ripley, 1993).

CONCLUSION

Artificial neural networks present a number of advantages over traditional methods of analysis these advantages can be listed as follows:

Artificial neural networks make no assumptions about the nature of the distribution of the data and are not, therefore, biased in their analysis. Instead of making assumptions about the underlying population, neural networks with at least one hidden layer use the data to develop an internal representation of the relationship between the variables (White, 1988).

Since, time-series data are dynamic in nature, it is necessary to have non-linear tools in order to discern relationships among time-series data. Neural networks are best at discovering non-linear relationships (Hoptroff, 1993; Moshiri *et al.*, 1999; Shtub and Versano, 1999; Garcia and Gencay, 2000; Hamm and Brorsen, 2000).

Neural networks perform well with missing or incomplete data (Kou and Reitsch, 1995). It is relatively easy to obtain a forecast n a short period of time as compared with an econometric model. Neural networks can overcome many of the shortcomings of traditional regression techniques, analyzing noisy data and data with outliers. From this discussion one can see that the capabilities of neural networks fall into three kinds of applications:

- Function fitting or prediction
- Noise reduction
- Pattern recognition
- Classification or placing into types

RECOMMENDATIONS

When results are required without an extraordinary need to understand the parameters behind them, then neural networks are most beneficial. When researchers need both an accurate prediction and an understanding of the predictive parameters both ANN and Regression should be used in combination. If the regression assumptions are not satisfied the ANN is more suitable

for analysis but if al regression assumption are completely satisfied then the regression analysis is more recommended.

SUGGESTIONS

Evaluate the performance of ANN using different paradigms, architectures and training schemes. Understand how the quality and quantity of training data can affect the performance. Investigate how the nature of the data and the domain of the application affect the performance of ANN in comparison with other methods. Develop general guidelines for the design of neural network structures. Develop an explicit set of rules to determine whether a given learning algorithm is suitable for particular applications.

REFERENCES

Ahmadian, M.H., 1995. Artificial neural networks-A new tool in technology. Tech Directions, 54: 33-36.

Altman, E.I., G. Marco and F. Varetto, 1994. Corporate distress diagnosis: Comparisons using linear discriminant analysis and neural networks (The Italian Experience). J. Banking Finance, 18: 505-529.

Baxt, W.G. and H. White, 1995. Bootstrapping confidence intervals for clinical input variable effects in a network trained to identify the presence of acute myocardial infarction. Neural Comput., 7: 624-638.

Bierens, H.J., 2006. Comment on artificial neural networks: An econometric perspective by Chung-Ming Kuan and Halbert White. Econom. Rev., 13: 93-97.

Bishop, C.M., 1995. Neural Networks for Pattern Recognition. Oxford University Press, Oxford, UK., Pages: 477.

Cheng, B. and D.M. Titterington, 1994. Neural networks: A review from a statistical perspective. Stat. Sci., 9: 2-30.

Cherkassky, V., J.H. Friedman and H. Wechsler, 2001. From Statistics to Neural Networks: Theory and Pattern Recognition Applications. Vol. 136, Springer, Berlin, Germany.

Cross, S.S., R.F. Harrison and R.L. Kennedy, 1995. Introduction o neural networks. Lancet, 346: 1075-1079.

Dhar, V. and R. Stein, 2012. Intelligent Decision Support Methods: The Science of Knowledge Work. Prentice Hall, Upper Saddle River, New Jersey, USA.

Efrom, B. and R.J. Tibshirani, 2011. An Introduction to the Bootstrap. Chapman & Hall, Boca Raton, Florida.

Efron, B., 1983. Estimating the error rate of a prediction rule: Improvement on cross-validation. J. Am. Stat. Assoc., 78: 316-331.

Fahlman, S.E., 1988. An empirical study of learning speed in back-propagation networks. Master Thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Farago, A. and G. Lugosi, 1993. Strong universal consistency of neural network classifiers. IEEE. Trans. Inf. Theor., 39: 1146-1151.

Garcia, R. and R. Gencay, 2000. Pricing and hedging derivative securities with neural networks and a homogeneity hint. J. Econom., 94: 93-115.

Garson, G.D., 1991. A comparison of neural network and expert systems algorithms with common multivariate procedures for analysis of social science data. Soc. Sci. Comput. Rev., 9: 399-434.

Garson, G.D., 2012. Neural Networks: An Introductory Guide for Social Cientists Thousand. Sage Group, Newcastle upon Tyne, UK.

Gorr, W.L., D. Nagin and J. Szczypula, 1994. Comparative study of artificial neural network and statistical models for predicting student grade point averages. Intl. J. Forecasting, 10: 17-34.

Goutte, C., 1997. Note on free lunches and cross-validation. Neural Comput., 9: 1245-1249.

Hamm, L. and B. Wade Brorsen, 2000. Trading futures markets based on signals from a neural network. Appl. Econ. Lett., 7: 137-140.

Hegde, S.U., J.L. Sweet and W.B. Levy, 1988. Determination of parameters in a Hopfieldnank computational network. Proceedings of the IEEE 1988 International Conference on Neural Networks, July 24-27, 1988, IEEE, San Diego, California, USA., pp: 291-298.

Hertz, J., A. Krogh and R.G. Palmer, 1991. Introduction to the Theory of Neural Computation. Addison-Wesley, Boston, Massachusetts, USA., ISBN:9780201503951, Pages: 327.

Hinton, G.E., 1992. How neural networks learn from experience. Sci. Am., 267: 144-151.

Hoptroff, R.G., 1993. The principles and practice of time series forecasting and business modelling using neural nets. Neural Comput. Appli., 1: 59-66.

Humpier, D., 1999. Neural Networks for Conditional Probability Estimation: Forecasting Beyond Point Predictions. Springer, Berlin, Germany, ISBN:9781447108481, Pages: 275.

Kaastra, I. and M.S. Boyd, 1995. Forecasting futures trading volume using neural networks. J. Future Markets, 15: 953-970.

Kou, C. and A. Reitsch, 1995. Neural networks vs. conventional methods of forecasting. J. Bus. Forecasting, 14: 17-22.

Kuan, C.M. and H. White, 1994. Artificial neural networks: An econometric perspective. Econ. Rev., 13: 1-91.

Marquez, L., T. Hill, R. Worthley and W. Remus, 1991. Neural network models as an alternative to regression. Proceedings of the 24th Annual Hawaii International Conference on System Sciences, January 8-11, 1991, IEEE, Kauai, Hawaii, USA., pp: 129-135.

Masters, T., 1995. Neural Novel and Hybrid Algorithms for Time Series Prediction. John wiley & Sons, Hoboken, New Jersey, USA., ISBN:9780471130413, Pages: 514.

McCulloch, W.S. and W. Pitts, 1943. A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys., 5: 115-133.

Moshiri, S., N.E. Cameron and D. Scuse, 1999. Static, dynamic and hybrid neural networks in forecasting inflation. Comput. Econ., 14: 219-235.

Ormerod, P., J.C. Taylor and T. Walker, 1991. Neural Networks in Economics. In: Money and Financial Markets, Taylor, M.P. (Ed.). Blackwell, Oxford, UK., pp: 341-53.

Pearl, J., 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Mateo, USA., ISBN: 9781558604797, Pages: 552.

Polyak, B.T., 1987. Introduction to Optimization. Optimization Software, New York, USA.

Refenes, A.P., A.N. Burgess and Y. Bentz, 1997. Neural networks in financial engineering: A study in methodology. IEEE. Trans. Neural Networks, 8: 1222-1267.

Ripley, B.D., 1993. Statistical Aspects of Neural Networks. In: Networks and Chaos Statistical and Probabilistic Aspects, Barndorff-Nielsen, O.E., J.L. Jensen and W.S. Kendall, (Eds.). Chapman & Hall, Boca Raton, Florida, USA., pp: 40-123.

Ripley, B.D., 1996. Pattern Recognition and Neural Networks. Cambridge University Press, Cambridge.

Rissanen, J., 1978. Modeling by shortest data description. Automatica, 14: 465-471.

Rosenblatt, F., 1958. The perceptron: A probabilistic model for information storage and organization in the brain. Psych. Rev., 65: 386-408.

Rummelhart, D.E., 1986. Learning Internal Representations by Error Propagation. In: Parallel Distributed Processing: I. Foundations, David, E., R. James and L. McClelland (Eds.). MIT Press, Cambridge, Massachusetts, USA., ISBN:9780262680530, pp: 318-362.

Salchenberger, L.M., E.M. Cinar and N.A. Lash, 1992. Neural networks: A new tool for predicting thrift failures. Decis. Sci., 23: 899-916.

Sarle, W.S., 1994. Neural networks and statistical models. Proceedings of the 19th Annual International Conference on SAS Users Group Cary, April 10-13, 1994, SAS Institute, Cary, North Carolina, USA., pp: 1538-1550.

Shtub, A. and R. Versano, 1999. Estimating the cost of steel pipe bending, a comparison between neural networks and regression analysis. Intl. J. Prod. Econ., 62: 201-207.

Tibshirani, R., 1996. A comparison of some error estimates for neural network models. Neural Comput., 8: 152-163.

Verkooijen, W., 1996. A neural network approach to long-run exchange rate prediction. Comput. Econ., 9: 51-65.

Warner, B. and M. Misra, 1996. Understanding neural networks as statistical tools. Am. Stat., 50: 284-293.

Weiss, M. and C.A. Kulikowski, 2007. Computer Systems that Lleam an Mateo. Morgan Kaufmann Publishers, Burlington, Massachusetts, USA.

White, H., 1988. Economic prediction using neural networks: the case of IBM daily stock returns. Proceedings of the International Conference on Neural Networks, July 24-27, 1988, IEEE Press, California, pp: 451-459.

Wikowska, D., 1995. Neural networks as a forecasting instrument for the polish stock exchange. Int. Adv. Econ. Res., 1: 232-242.

Wong, F.S., 1991. Time series forecasting using backpropagation neural networks. Neurocomputing, 2: 147-159.

Wong, M.A. and T. Lane, 1983. A Kth nearest neighbour clustering procedure. J. Royal Stat. Soc. Ser. B. Methodol., 45: 362-368.

Zahedi, F., 1993. Intelligent Systems for Business: Expert Systems with Neural Networks. Wadsworth Publishing Company, Belmont, California, ISBN:9780534188887, Pages: 658.

Zhu, H. and R. Rohwer, 1996. No free lunch for cross-validation. Neural Comput., 8: 1421-1426.