Comparative Analysis of Neural Network Techniques Vs Statistical Methods in Capacity Planning

Nalini Vasudevan
Columbia University
Dept. of Computer Science
New York, USA
naliniv@cs.columbia.edu

Gowri C Parthasarathy
Hewlett Packard
System Performance Group
Bangalore, India
gowri.cp@hp.com

Abstract

Capacity planning is a technique which can be used to predict the computing resource needs of an organization for the future after studying current usage patterns. This is of special import for adaptive enterprises, given the large infrastructure and large number of users. Determining resource needs beforehand can be very beneficial because it is a proactive approach and helps prevent resource crunches and service level violations. Accuracy of the predicted values, however, depends upon the methods used for the forecast and also upon the accuracy of the historical data. Historical data in the capacity planning sense is system performance data. Most of the approaches used for such a prediction make use of statistical methods or are based on queuing theory.

This paper compares the traditional statistical based methods with a method based on neural networks. The training set for the neural network consists of historical values of a metric (for example CPU utilization percentage) for which the prediction is to be done. The advantages of this method over other methods have also been discussed. From the predicted information, we illustrate how capacity planning is done.

1. Introduction

An adaptive enterprise (or adaptive organization)[3] is defined as 'An organization in which the demand and supply of goods or services are matched and synchronized at all times. Such an organization optimizes the use of its resources (including its information technology resources), always using only those it needs and paying only for what it uses, yet ensuring that the supply is adequate to meet the demand'. The success of an adaptive enterprise lies in knowing current resource utilization patterns and being able to predict what future needs will be. Capacity planning serves this purpose by studying trends of and predicting computing resource utilization, over a period. This kind of ability to anticipate future requirements is useful especially when resources have to be procured in advance and service level agreements should never be violated while existing resources should also be efficiently managed.

2 Related Work

Capacity planning and performance monitoring are two fields that have gained importance in IT firms. It is important to decide on the computing resources a firm needs, to plan and organize hardware infrastructure and anticipate hardware needs. Hence, capacity planning is a rapidly developing field of research. Various methods have been formulated or studied to predict future resource needs or computing capacities. [7] illustrates a neural network and data visualization approach to the capacity planning of computer resources. In [10], the authors talk about a back propagation based approach to do capacity planning for a telecom network. [5] compares the accuracy of various neural network and statistical methods to predict cancer.

In this paper we talk about the back propagation based ANN approach to estimate CPU utilization and compare the accuracy of the neural network method with statistical methods.

3 Overview of Capacity Planning

3.1 Importance of Capacity Planning

- Cost: Extra expenditure can be reduced by avoiding costs due to unplanned behavior.
- Distribution: Information can be distributed in advance. For instance, this helps in reducing CPU overuse or CPU idling.

0-7695-2867-8/07 $25.00 © 2007 IEEE
DOI 10.1109/SERA.2007.73

- Migration: Resource migration is possible when the value of a metric hits a threshold. For instance, a process can be migrated in advance to a different system if the expected CPU utilization is more than the maximum threshold.
- Resource Allocation: Resources can be allocated to new applications as well as currently running applications.
- Reduce Delay: The infrastructure can be optimally configured to reduce delay and over time.

3.2 Issues related with Capacity Planning

[14] describes some of the issues related to capacity planning .These are summarized below:

- Lack of time: System and program analysts are so involved in other activities such as installations, maintenance etc. that they do not have time for planning. Very few organizations set aside or assign a group for infrastructure or resource planning.
- Lack of interest: One of the most important reasons for lack of interest is the limited emphasis on resource planning for the organization. More significance is given to current activities and maintenance.
- Inaccurate forecasting: Due to insufficient tools, skills and lack of experience and training, forecasting can be inaccurate and hence could result in incorrect future predictions.
- Use of conventional methods: Analysts may be reluctant to try out and use new methods and may stick to traditional techniques and hence be unaware of more effective and efficient tools.
- Change in corporate model: The strategies within the organization may change every year. These changes are required to be incorporated in the capacity planning tool and therefore the model has to be changed frequently.
- Lack of man-power: Due to cost constraints and limited manpower, organizations tend to give very little importance to capacity planning.

4 Measurement of CPU Utilization

CPU utilization is one of the measures of capacity. It is measured differently for different operating systems. For example, in Linux, the data is obtained from the /proc file system. Data is collected at regular intervals and the final value of utilization is calculated from the measured data and the interval. If the system has more than one CPU then the utilization is a average of the utilizations of the individual CPUs. The CPU Utilization for a single processor system at various times is shown in Table 1.

Sl. No.	Time	CPU Utilization
i 1	9/18/04 4:00PM	17.53
2	9/18/04 5:00PM	6.33
3	9/18/04 6:00PM	6.71
4	9/18/04 7:00PM	3.72
5	9/18/04 8:00PM	3.69
6	9/18/04 9:00PM	3.72
7	9/18/04 10:00PM	3.71
8	9/18/04 11:00PM	7.04
9	9/19/04 12:00AM	3.7
10	9/19/04 1:00AM	4.19
11	9/19/04 2:00AM	4.19
12	9/19/04 3:00AM	3.72
13	9/19/04 4:00AM	3.73
14	9/19/04 5:00AM	4.12
15	9/19/04 6:00AM	3.72
16	9/19/04 7:00AM	3.88
17	9/19/04 8:00AM	13.79
18	9/19/04 9:00AM	23.46
19	9/19/04 10:00AM	24.23
20	9/19/04 11:00AM	24.97
21	9/19/04 12:00PM	10.99
22	9/19/04 1:00PM	23.51
23	9/19/04 2:00PM	25.12
24	9/19/04 3:00PM	51.5
25	9/19/04 4:00PM	21.18
26	9/19/04 5:00PM	19.53
27	9/19/04 6:00PM	4.16
28	9/19/04 7:00PM	3.74
29	9/19/04 8:00PM	3.73
30	9/19/04 9:00PM	3.71
31	9/19/04 10:00PM	3.73
32	9/19/04 11:00PM	7.73
33	9/20/04 12:00AM	3.69
34	9/20/04 1:00AM	4.26
35	9/20/04 2:00AM	4.21
36	9/20/04 3:00AM	3.69
37	9/20/04 4:00AM	3.78
38	9/20/04 5:00AM	4.19
39	9/20/04 6:00AM	3.7
40	9/20/04 7:00AM	4.32

Table 1. CPU Utilization percentage for a single processor system

5 Basic Statistical Methods

The historical data used for prediction is basically time series data and prediction can be done for a specified interval. The data is first processed and then analyzed for trends.

Smoothing is a technique used to find patterns in the data. An appropriate prediction method can then be used. Most prediction techniques make use of statistics. Curve fitting is the process of finding a curve that best fits the data points under consideration. This means that we have to find some curve which passes through most of the data points. In order to do this, one has to understand the nature of data and all the factors which affect it.

In the case of linear regression, this curve is in the form of a straight line of the form,

$$y = mx + c \tag{1}$$

Where m is the slope of the line and c is the y intercept. Here x is the explanatory variable and y is the dependent variable.

Increasing the order of x in the equation means trying to fit more points in the curve.

The slope is generally the correlation coefficient which is denoted by r.

r can take any value between -1 and 1. It is a measure of the degree to which the two variables are related. Thus if a prediction line has a slope of 1, it is a best fit curve for the data points. If the slope of the line is 0 then there is no linear relationship between the two variables.

The intercept is governed by the following equation;

$$Intercept = \frac{\sum y - r \sum x}{n} \tag{2}$$

A related measure is the coefficient of determination denoted by r2. It is a measure of how closely x and y are related and hence how close the points are to the regression line.

Standard error is a measure of how well the regression line fits the data points. Hence it is the distance between the data point and the prediction line at each data point.

If the relationship between the explanatory and the dependent variable is exponential in nature, the following exponential model can be used for prediction:

$$y = exp(mx + c) \tag{3}$$

Where exp is e, the natural logarithm base, also called Eulers e, which is 2.718.

Another model which can be used to predict values for variables whose values are bounded is,

$$y = \frac{y_{max}}{(exp(mx + c) + 1)} \tag{4}$$

Where ymax is the maximum value that y can take.

Several other curve fitting models can also be used. Some of them are: Gaussian, Weibull, Fourier and power series, spline.

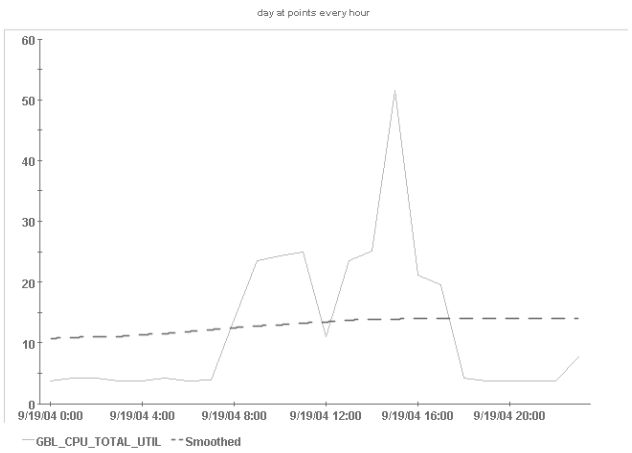

Figure 1. Curve Smoothing

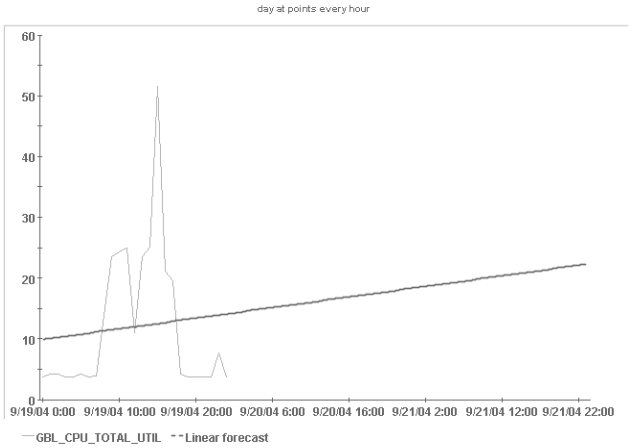

Figure 2. Linear Forecast: Y = .17412 X + 9.91 Correlation Coefficient = .1076 Estimated Standard Error = 10.7267 Approximately 1.16 percent of the variance in GBL_CPU_TOTAL_UTIL can be explained as a function of time.

Sl. No.	Time	Predicted Value	Actual Value	Square Error
1	9/20/04 1:00AM	14.263	4.26	10.003
2	9/20/04 2:00AM	14.4371	4.21	10.2271
3	9/20/04 3:00AM	14.6112	3.69	10.9214
4	9/20/04 4:00AM	14.7853	3.78	11.0053
5	9/20/04 5:00AM	14.9594	4.19	10.7694
6	9/20/04 6:00AM	15.1336	3.7	11.4336
			Mean Error	10.7267

Table 2. Linear Model: Predicted values vs Actual values for different times

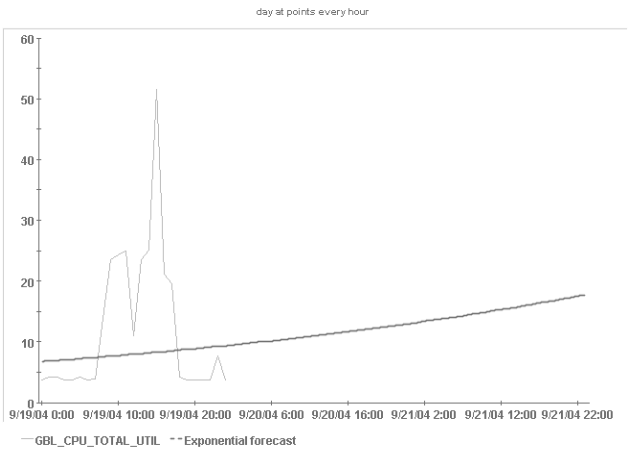

Figure 3. Exponential Forecast: Y = exp(.01360 X + 1.91) Correlation Coefficient = .1117 Estimated Standard Error = 5.8468 Approximately 1.25 percent of the variance in GBL_CPU_TOTAL_UTIL can be explained as a function of time.

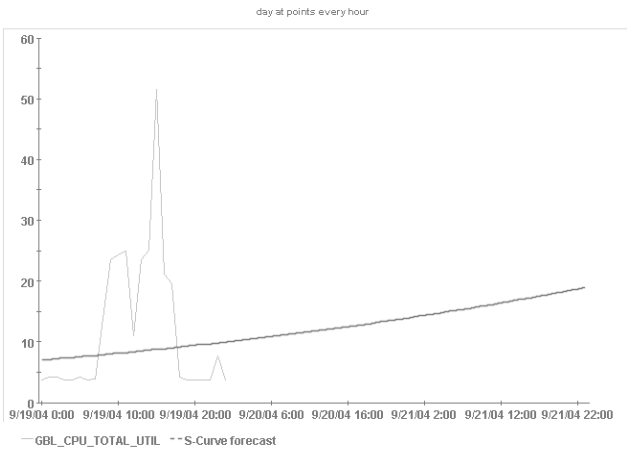

Figure 4. S-Curve Forecast: Y = 100.0 / (exp(-.01589 X + 2.58) + 1.0) Correlation Coefficient = -.1125 Estimated Standard Error = 6.5295 Approximately 1.26 percent of the variance in GBL_CPU_TOTAL_UTIL can be explained as a function of time.

Sl. No.	Time	Predicted Value	Actual Value	Square Error
1	9/20/04 1:00AM	9.4877	4.26	5.2277
2	9/20/04 2:00AM	9.6176	4.21	5.4076
3	9/20/04 3:00AM	9.7493	3.69	6.0593
4	9/20/04 4:00AM	9.8828	3.78	6.1028
5	9/20/04 5:00AM	10.0181	4.19	5.8281
6	9/20/04 6:00AM	10.1553	3.7	6.4553
			Mean Error	5.8468

Table 3. Exponential model: Predicted values vs Actual values for different times

Sl. No.	Time	Predicted Value	Actual Value	Square Error
1	9/20/04 1:00AM	10.1310	4.26	5.8710
2	9/20/04 2:00AM	10.2766	4.21	6.0666
3	9/20/04 3:00AM	10.4240	3.69	6.7340
4	9/20/04 4:00AM	10.5733	3.78	6.7933
5	9/20/04 5:00AM	10.7245	4.19	6.5345
6	9/20/04 6:00AM	10.8776	3.7	7.1776
			Mean Error	6.5295

Table 4. S-curve Model: Predicted values vs Actual values for different times

6 Artificial Neural Networks

In this paper we extend the analysis to use the concept of Neural networks ([4],[6],[9])

A neural network is composed of a number of interconnected units known as artificial neurons or simply neurons([12]).Each unit has input/output(I/O) characteristics and implements a local computation or function within the unit, i.e., the output of any unit is a function of the input. The output of any unit is determined by the I/O characteristics (the function), its interconnection to other units(the input) and also external inputs such as bias. Neural networks have diverse applications from speech recognition to military applications. The various applications of Neural Networks are enumerated in [8] and [15]. In this paper, we have applied the concept of neural networks to estimate the CPU utilization percentage of a system. The neural network model is an efficient and powerful means of computation. Its functionality is modeled on the working of neurons in the human brain and that of the nervous system. Neural networks try to emulate the cognitive activity of the brain but do not totally achieve biological outputs. Neural networks are pattern classifiers. The information and knowledge is distributed throughout the network and is stored in the form of weighted connections or weights . The most valuable characteristics of neural networks are their adaptability and learning process. Thus they are well suited for applications that involve pattern classification of input .

A single neuron by itself cannot be used for pattern recognition. The pattern recognition capability arises when we combine neurons into multilayer structures collectively known as neural networks.

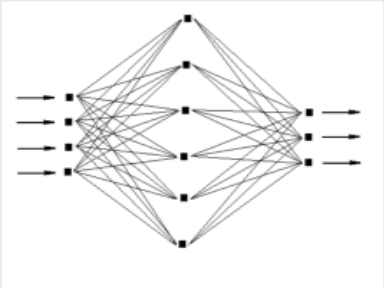

Figure 5. A 'Simple Neural Network'

The components and important terms of a neural network are:

- Neuron: A unit capable of some functionality. In other words neurons are a set of nodes that connect them to inputs, output, or other neurons. They are also termed as synapses.
- Linear Combiner: A function that takes all inputs and produces a single value at the output. The linear combiner functions by adding together the Input multiplied by the corresponding weight. The result is then applied to a function known as the activation function.
- Activation Function: Takes any input from minus infinity to plus infinity and maps it into a range -1 to 1 or into 0 to 1 interval.
- Threshold: Defines the state of a neuron when no input is applied to it. In general, for the neuron to fire or be active, the sum should be greater than the threshold. Here we replace the threshold with an extra input known as bias which is a weight that can change during the learning process : Note that the bias can be positive or negative.
- The layers: In a three layer system, the first layer is known as the input layer, the middle layer is known as hidden layer and the last layer is the output layer.

6.1 Back Propagation

Neural networks are employed for machine learning [11]. Back propagation is one of the algorithms used for self-learning and recognition. The primary objective of this paper is to explain how to use the back propagation training functions to train feed forward neural networks to predict system performance data and plan for resources accordingly and to compare this method with methods based on statistical methods. In most cases, there are four steps in the training process:

1. Aggregate the training(historical) data
2. Structure the network - Choose the learning rate λ and the number of hidden units, input units and output units
3. Train the network with the data.
4. Output the trained network's response to fresh inputs

6.1.1 Feed forward Dynamics

In a back propagation network, the activations of the input units are propagated forward to the output layer through the connecting weights

$$net_j = \sum w_j a_i \tag{5}$$

where a_i is the input activation from unit i and w_{ji} is the weight connecting unit i to unit j. However, instead of calculating a binary output, the net input is added to the unit's bias and the resulting value is passed through a sigmoid function:

$$F(net_j) = \frac{1}{1 + e^{-net_j + j}} \tag{6}$$

The sigmoid function is sometimes called a "squashing" function because it maps its inputs onto a fixed range.

6.1.2 Gradient Descent

Gradient descent is a hill-descending algorithm that finds the minimum of a complicated function. Many algorithms used in training neural networks use gradient descent.

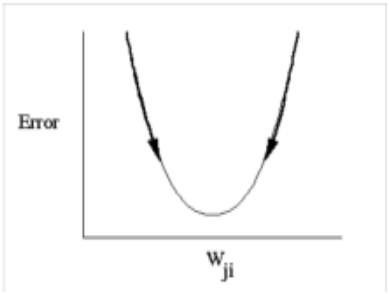

Figure 6. Gradient Descent

6.1.3 The Backpropagation Training Algorithm

The objective of the Backpropagation training algorithm is to minimize the error by adjusting the weights. The algorithm as quoted in [13], can be summarized as follows:

The error signal for unit j at some layer is given by

$$\delta_j = -\frac{\partial E}{\partial net_j}$$

The negative gradient for weight wij:

$$\delta w_{ij} = -\frac{\partial E}{\partial w_{ij}}$$

Using the chain rule, we expand δw_{ij} :

$$\delta w_{ij} = -\frac{\partial E}{\partial net_i}\frac{\partial net_i}{\partial w_{ij}}$$

$$\frac{\partial net_i}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}\sum_k y_i w_{ik} = y_j$$

Combining the above equations, we get

$$\delta w_{ij} = \delta_i y_j$$

The gradient is computed from the activity and error at the corresponding layers in the neural network. The ouput at any unit is given by:

$$y_i = f_i(\sum_j w_{ij} y_j)$$

where f_i is the sigmoidal function. The error at any output unit is given by

$$E = 1/2 \sum o(t_o - y_o)^2$$

On differentiating, we obtain:

$$\delta_o = t_o - y_o$$

The error is propagated back from the output nodes to adjust the weights. The error at the hidden layer is calculated as follows:

$$\delta_j = -\sum_i \frac{\partial E}{\partial net_i}\frac{\partial net_i}{\partial y_j}\frac{\partial y_j}{\partial net_j}$$

$$\frac{\partial net_i}{\partial y_j} = \frac{\partial}{\partial y_j}\sigma_{for-all-k} w_{ik} y_k = w_{ij}$$

$$\frac{\partial y_j}{\partial net_j} = \frac{\partial f_j(net_j)}{\partial net_j} = f_j'(net_j)$$

Combining the above equations, we get,

$$\delta_j = f_j'(net_j)\sum_i \delta_i w_{ij}$$

For any layer l, the error back propagation is as follows:

1. The input layer is initialized as follows where $\vec{x}$ is the input set:

$$\vec{y_0} = \vec{x}$$

2. The activity is propagated forward for l = 1, 2, ..., n, and f_l is the sigmoidal function

$$\vec{y_l} = f_l(\vec{w_l}\vec{y}_{l-1} + \vec{b_l})$$

where b_l is the vector of bias weights.

3. Error is calculated at the output- $\vec{t}$ is the desired output and $\vec{Y_n}$ is the calculated output:

$$\vec{\delta_n} = \vec{t} - \vec{y_n}$$

4. The error is backpropagated: for l = n-1, n-2, ..., 1,

$$\vec{\delta_l} = (\vec{w_{l+1}^T}\vec{\delta}_{l+1}).f_l'(\vec{net}_l)$$

7 Experimental Analysis

Date and time:	9/20/04 2:00AM
Learning Patterns:	Learning....Input: 10.99 23.51 25.12 51.5 21.18 19.53 4.16 Output:3.74 Learning....Input: 23.51 25.12 51.5 21.18 19.53 4.16 3.74 Output:3.73 Learning....Input: 25.12 51.5 21.18 19.53 4.16 3.74 3.73 Output:3.71 Learning....Input: 51.5 21.18 19.53 4.16 3.74 3.73 3.71 Output:3.73 Learning....Input: 21.18 19.53 4.16 3.74 3.73 3.71 3.73 Output:7.73 Learning....Input: 19.53 4.16 3.74 3.73 3.71 3.73 7.73 Output:3.69 Learning....Input: 4.16 3.74 3.73 3.71 3.73 7.73 3.69 Output:4.26
Predicting from:	Input: 3.74 3.73 3.71 3.73 7.73 3.69 4.26 Output=?
Predicted Value:	4.25
Actual Value:	4.21
Error squared:	0.0016

Table 5. Estimation of CPU Utilization on 9/20/04 at 2:00 AM

7.1 Window size

The system performance data is estimated based on the assumption that at any time, the value then will depend on the system performance data at previous n times. In the neural network model that we outline, we take n as 7. We use k such patterns of data to train the network from the historical set (Table 1) and we set k=7; it is not necessary that n and k have to be equal.

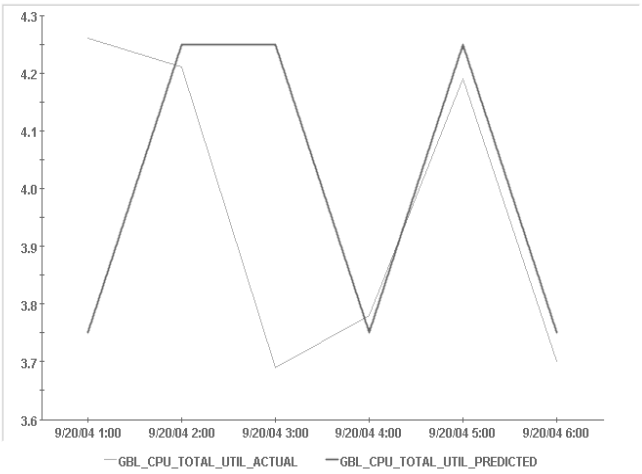

Figure 7. Neural Network Prediction

Sl. No.	Time	Predicted Value	Actual Value	Square Error
1	9/20/04 1:00AM	3.75	4.26	0.2601
2	9/20/04 2:00AM	4.25	4.21	0.0016
3	9/20/04 3:00AM	4.25	3.69	0.3136
4	9/20/04 4:00AM	3.75	3.78	0.0009
5	9/20/04 5:00AM	4.25	4.19	0.0036
6	9/20/04 6:00AM	3.75	3.7	0.0025
			Mean Error	0.09705

Table 6. Neural Network: Predicted values Vs Actual values for different times

7.2 Input, output layers and the hidden layers

Note that the 'n' defined in the previous section is nothing but the size of the input array. The output layer is assigned numbers corresponding to the output discretized with 0.5 (0.5 is the difference between any successive units). The output range varies from 0 to 100 in steps of 0.5 because the CPU utilization cannot drop below 0 or exceed 100. Since the problem is not separable into two or more steps, we choose a neural network with a single hidden layer.

7.3 Selection of λ

The learning rate of λ determines the step size of the algorithm. It is application dependent. If λ is too low, then the network tends to learn slowly. If it is too high, the network will tend to oscillate and will never learn for two or more incompatible sets of data. It is therefore required to choose an optimal lambda. A λ of 0.65 is used in this case.

7.4 Training

The network is trained with historical data. The weights are adjusted such that for a given input, the corresponding weighted output is obtained. For a set of training patterns, the weights remain the same. Once the network is trained, the new data is fed to the input layer and using the trained weights, the output is calculated at each unit of the output layer. The output at each unit is nothing but the probability in terms of other units. Wherever the probability is maximum, the corresponding value at that unit is the predicted value for the given set of data. For e.g., to predict the CPU utilization at 2:00AM on 9/20/04, the network is trained using historical data based on last 7 values as shown in Table 5

8 Results and Comparison

Table 6 shows the predicted values estimated over the first few hours of the day on 9/20/04. The mean square error is calculated as shown. The following results can be drawn from the experiment:

1. Choosing the right historical window: The system performance data for a future time interval may depend on past data collected over varying intervals. The span of the historical data to be chosen depends on the organization, its infrastructure, type of application and the workload. Hence, it becomes quite difficult to choose the right window of information. As discussed before, it is important to choose the right 'n' and 'k' for training the neural network. All models including stochastic techniques suffer from this limitation.

2. Choosing the learning rate λ: It is important to choose the right learning rate. Different servers in the same organization may require different learning rates. The stochastic methods do not require a learning rate equivalent.

3. Kind of data: The linear model is quite useful when the CPU utilization is expected to vary linearly. When the variation is exponential, the exponential model is used. But these two models fail to work when the CPU utilization is irregular. Not only does the neural network work satisfactorily with irregular data, it works well with both linear and exponential data. Table 2, Table 3 and Table 4 show the performance of different models with irregular data

4. Learning time: The stochastic models are very fast compared to the neural network model. The neural net requires continuous forward propagation and back propagation to adjust the weights and hence takes a

longer time. However, the neural network can be programmed in a distributed environment and hence the overhead in time can be reduced.

9 Capacity Planning from predicted values

From the predicted values of CPU utilization, one can calculate the number of processors that will be required in the future, depending on the load that each CPU has to bear. One can also measure the load on individual CPUs instead of a cumulative load and then predict what its utilization will be over a given period of time. This data will in turn be useful for load sharing between CPUs. CPU 'partitions' can be configured and processes migrated. Thus this approach will help in efficient management of existing processing resources. The prediction data can also help determine if additional resources will be necessary in the near future and hence gives sufficient notice to procure those resources.

10 Conclusion

In this paper, we have studied artificial neural networks as a framework for measuring the system performance of an organization. We have compared the neural network behavior with the conventional methods being used by the organization for a typical case. From the results, it can be concluded that for any organization, neural network as a pattern classifier is a better model then statistical methods to predict the CPU utilization. We hope to extend this analysis by using various other neural network architectures as well as other machine learning techniques such as fuzzy logic.

References

[1] Ibm research a statistical approach to capacity planning for on-demand computing services.

[2] Performance by design: Computer capacity planning by example.

[3] Searchsmb definitions, http://searchsmb.techtarget.com.

[4] J. A. Anderson. *An introduction to Neural Networks*. MIT Press, 1 edition, 1995.

[5] R. D. G. P. Burke, H.B. Comparing artificial neural networks to other statistical methods for medical outcome prediction. *Neural Networks, IEEE World Congress on Computational Intelligence*, 4:2213–2216, 1994.

[6] L. V. Fausett. *Fundamentals of Neural Networks : Architectures, Algorithms, and Applications*. Prentice -Hall, Englewood Cliffs, NJ, 1994.

[7] M. T. P. Feng, X. A neural network and data visualization approach to the capacity planning of computer resources. *Journal of Artificial Neural Networks*, 1, Issue 4:501–520, 1994.

[8] S. D. M. Freeman J.A. *Neural networks:algorithms, applications, and programming techiques JA Freeman, DM Skapura*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, 1991.

[9] R. M. Golden. *Mathematical Methods for Neural Network Analysis and Design*. MIT Press, 1 edition, 1996.

[10] L. Lee and H. Chun. An ann appraoch to spare capacity planning. *TENCON 92. Technology Enabling Tomorrow : Computers, Communications and Automation towards the 21st Century, IEEE Region 10 International Conference*, 2:891–895, 1992.

[11] T. M. Mitchell. *Artificial Neural Networks in Machine Learning*. Mc Graw Hill Companies Inc., New York, 1997.

[12] A. R. Nalini Vasudevan, Arjun Jain. A connectionist framework for feature based speech recognition using artificial neural networks. *Student CSI Convention*, Nov 2004.

[13] G. B. Orr. *CS-449: Neural Networks Lecture Notes*.

[14] R. Schiesser. *Why Capacity Planning is Seldom Done Well*. Prentice Hall PTR, Feb 2004.

[15] D. E. R. Widrow, Bernard and M. A. Lehr. neural networks: Applications in industry, business and science. *Communications of the ACM*, pages 93–105, 1994.