Applications of Neural Networks for Ranking of Web Services using QoS Metrics

Dr.M.Kamalahhasan
Department of Electronics & Communication Engineering
VLB College of Engineering, Coimbatore, India.

ABSTRACT

Web Service is defined as "a software system designed to support interoperable machine-to-machine interaction over a network". Web service discovery is the process of finding a suitable Web service for given task.. Discovering web services can be based on keyword search or QoS metrics. Different web services may have same functionality but may have different non-functional properties. QoS metrics are useful when a client wants to select a service from a set of available services having the same functional properties. In this paper we report the performance of various Artificial Neural Network (ANN) training algorithms in predicting the ranking of a web service.

1. Introduction

As the web services are becoming increasingly complex systems there is an arising need for the systematic and quantitative quality evaluation of the web services, which is rather a recent and frequently neglected issue.

Even though there exist many detailed guidelines for measuring the web services quality, there is still a debate about what actually constitute a good web services. Furthermore there is a lack of empirical validation for good quality web service guidelines. Thus, we say defining a quality of service metrics is necessary in order to overcome this situation. In this paper we report the obtained results of evaluation through different focus like: Response time, Throughput, Availability, Successability, Reliability, Compliance, Best Practices, Latency and Documentation. We employed ANN technique over the QWS dataset [5, 7]. Finally, the results of this analysis allow us to construct model for predicting if a service will be highly rated in terms of its quality.

This paper is structured as follows. Section 2 describes related work; Section 3 discusses our research background; Section 4 presents the ANN Modeling; Section 5 finally the results of the study in detail, and our conclusion.

2. Related work

Very little research is conducted on investigating web services on the web. In [6], the author collected statistical data using a proposed web service crawler engine (WSCE), which help in conducting investigation of web services on the web and further determining the current status of the web services.

There are several approaches that address how to deal with QoS foe web services.

In [5], the author proposed web service QoS manager, an intelligent system that examines web service's QoS properties to select the best available web service by taking advantage of client QoS preferences

In another effort authors in [7] offers a quality driven discovery of web services and uses a combination of web services attribute for measuring the particular web service based on QoS metrics
Further, incorporating the QoS properties gives confidence about the quality of web services

The work reported in this paper expands on determining the quality of web service using neural networks taking QoS metrics as inputs and finally concludes regarding suitability of neural networks in web engineering.

3. Research background

Neural networks have a close analogy to non-parametric statistical inference and are very powerful in modeling non linear relationships. In contrast to some statistical techniques they do not require to specify the relationships between inputs and outputs. ANNs tend to be useful in quality modeling because, the number of inputs (QoS metrics) is fairly large, many of the metrics are relevant but predictive information lies in a lower dimensional subspace. In this paper we use ANN to predict the rank of a web service .The independent variable used in this study as inputs to the ANN are QoS metrics in the QWS dataset [5, 7].

The following inputs have been used:

1. **Response Time (RT)**: the Time taken to send a request and receive a response (unit: milliseconds).
2. **Throughput (TP)**: the maximum requests that are handled at a given unit in time (unit: requests/min).
3. **Availability (AV)**: a ratio of the time period when a Web service is available (unit: %/3-day period).
4. **Successability (SA)**: Number of response / number of request messages
5. **Reliability (REL)**: Ratio of the number of error messages to total messages.
6. **Compliance (CP)**: the extent to which WEB Service Description Language (WSDL) document follow.
7. **Best Practices (BP)**: the extent to which Web services follow.
8. **Latency (LT)**: Time taken for the server to process a given request
9. **Documentation (DOC)**: measure of documentation (i.e. description tags)

The output of the ANN is **Web Service Relevancy Function (WsRF**), which is used to measure the quality ranking of a Web service based on quality metrics

4. ANN modeling

The most popular ANNs are Multilayered Perceptrons (MLPs). MLPs are multilayered feed forward networks [2]. The network consists of set of sensory units that constitute the input layer, one or more hidden layers of computation nodes, and output layer of computation nodes. The input signal propagates through the network in a forward direction on a layer-by –layer basis.

4.1 Back propagation learning algorithm

Back propagation (BP) algorithm consists of two phases, namely, learning phase and working phase. In learning phase a set of input patterns (training set) are presented to the input layer together with their corresponding desired output patterns small random initial weight value is assigned to each connection between nodes in the input layer and the hidden layer as each input pattern is applied, the actual output is recorded .the weights between the output layer and the previous layer (hidden layer) are recalculated using the generalized delta rule .The adjustment reduces the difference between the network actual outputs and the desired outputs for a given input pattern.

The input to each node for successive layers is the sum of the scalar products of the incoming vector components with their respective weights. Thus the input to a node j (Figure 1) is given by [3, 4]:

$$input_j = \sum_i w_{ji} out_i$$

Where w_{ji} is the weight connecting node i to node and out_i is the output of node i. No calculation is performed at the input layer since it is just feeding the value of input patterns to the network .The output of a node j is, therefore,

$$out_j = f(input_j)$$

The fundamental of back-propagation algorithm employed in training phase is a gradient descent optimization procedure, which minimizes the mean squared error between network output ant the desired output of all input patterns p as follows [4]:

$$E = \frac{1}{2P} \sum_p \sum_k (d_k - out_k)^2$$

The training set is presented iteratively to operate the network, whereby the weights are updated until their values become stabilized according to the following criteria:

- a user defined error tolerance is achieved, or
- a maximum number of iterations is reached.

The neural network was trained using various variants of back propagation algorithm [1, 2]. This algorithm generally works best when the network inputs and targets are normalized all the input metrics and actual outputs were scaled with min-max normalization technique. The network architecture consisted of a single hidden layer. The number of hidden neurons was empirically determined by partitioning the data into training, validation and test subsets in the ratio 3:1:1. A network with five hidden neurons gave the best prediction results. Mu was set equal to 0.001. Table 1 summarizes the architecture as

Table 1: Neural network architecture

Hidden Layer	1
Hidden Neurons	5
Activation function hidden layer	Tansig
Output neuron	1
Activation function output neuron	purelin

Figure 1: Architecture of Neural Network

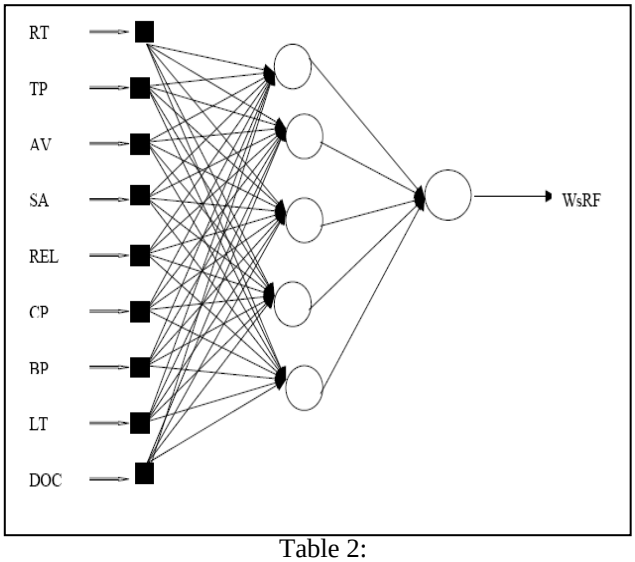

Table 2:

Training Algorithm	Description
trainb	trains a network with weight and bias learning rules with batch updates. The weights and biases are updated at the end of an entire pass through the input data.
trainbfg	updates weight and bias values according to the BFGS quasi-Newton method.
trainbr	updates the weight and bias values according to Levenberg-Marquardt optimization. It minimizes a combination of squared errors and weights, and then determines the correct combination so as to produce a network that generalizes well. The process is called Bayesian regularization.
trainc	Trains a network with weight and bias learning rules with incremental updates after each presentation of an input. Inputs are presented in cyclic order.
traincgb	updates weight and bias values according to the conjugate gradient back propagation with Powell-Beale restarts.
traincgf	updates weight and bias values according to the conjugate gradient back propagation with Fletcher-Reeves updates.
traincgp	updates weight and bias values according to the conjugate gradient back propagation with Polak-Ribiere updates.
traingd	updates weight and bias values according to gradient descent.
traingda	updates weight and bias values according to gradient descent with adaptive learning rate.
traingdm	updates weight and bias values according to gradient descent with momentum.
traingdx	updates weight and bias values according to gradient descent momentum and an adaptive learning rate.
trainlm	updates weight and bias values according to Levenberg-Marquardt optimization.
trainoss	updates weight and bias values according to the one step secant method
trainrp	updates weight and bias values according to the resilient back propagation algorithm (RPROP).
trainscg	updates weight and bias values according to the scaled conjugate gradient method.

5. Conclusion

Table 3:

Training Algorithm	Mean Magnitude of relative error
trainb	0.425
trainbfg	0.056
trainc	0.171
traincgb	0.661
traincgf	0.067
traincgp	0.079
traingd	0.032
traingda	0.246
traingdm	0.283
traingdx	0.202
trainlm	0.008
trainoss	0.083
trainrp	0.079
trainscg	0.083

After the Neural network is trained and the training algorithm with lowest error is determined, the QOS metrics obtained for any new web service can be input to the neural network to determine the WsRF of the web service.

6. Acknowledgements

We would like to thank Mohamed Al-Masri for providing us with his valuable QWS dataset, which we have used for this work.

Reference

1. K.K Aggarwal, Y.singh, P.chandra, M.Puri,"Evaluation of various training Algorithms for software Engineering applications." ACM SIGSOFT Software Engineering Notes, vol 30, no.4 July 2005.
2. S.haykins,"A Comprehensive Foundation on Neural Networks. "Prentice Hall, 1999.
3. www.mathworks.com
4. Kanellopoulos,I.,*http://ams.egeo.sai.jrc.it/eurosta/Lot 16-SUPCOM 95/node7.html*, 1997.
5. Al-Masri, E., and Mahmoud, Q. H., "Discovering the best web service", 16th International Conference on World Wide Web (WWW), 2007.
6. Al-Masri, E., and Mahmoud, Q.H., "Investigating Web Services on the World Wide Web", 17th International Conference on World Wide Web (WWW), Beijing, April 2008.
7. Al-Masri, E., and Mahmoud, Q. H., "QoS-based Discovery and Ranking of Web Services", IEEE 16th International Conference on Computer Communications and Networks (ICCCN), 2007, pp. 529-534.