Implementation of Information Retrieval (IR) in an Electronic Commerce Architecture using Back propagation Network Learning Algorithm

Riktesh Srivastava

***Abstract*—**The present study assesses the technique of back propagation neural networks to appraise the required Information Retrieval (IR) and thereby decreasing the average response time of an Electronic Commerce architecture. In order to delineate the response time, diverse array of user requests were engaged per unit time. Furthermore, engagement of Back Propagation Network Learning (BPNL) algorithm is used to train the requests of the users and summarize the average response time and augment the enactment of the system. The comprehensive study does the comparative investigation to express the average response time for ANN enabled and without-ANN-enabled algorithm. The objective was to plaid whether ANN enabled algorithm had any bearing on the overall performance of the system. For BPNL algorithm, the information retrieval for the user queries were steered for 9 repetitions and then thorough phases were accomplished to assess the response time. After each iterations, error rates were dogged and then feed forward and back propagation algorithm were used to improve the performance. The experimentation will find its prominence in imminent Electronic Commerce system project and employment and will convey the outline for such investigation. Finally, the study expands the meticulous inferences of the study.

***Keywords*—**Electronic Commerce architecture, BPNL Algorithm, ANN.

I. Introduction

THE training of requests for finding the exact content from the list of information stored in an Electronic Commerce setup can be termed as a mathematical problem. Many mathematical equations were defined and implemented to get the fast results and output match the expectation of the end user. The study is an attempt to conduct the experiment of Information Retrieval (IR) using Artificial Neural Networks by implementing Back Propagation Neural Network. Artificial neural networks are biologically stimulated classification algorithms that entails of an input layer of nodes, one or more hidden layers and an output layer. Each node in a layer has one corresponding node in the next layer, thus spawning the stacking effect [1]. Back propagation Network Learning Algorithm (BPNL) is one of the prevalent structures amid artificial neural networks which is extensively used to elucidate complex problems by modeling complex input-output relationships [2].

The preliminary BPNL algorithm was suggested by Rumelhart et al [3] and since then became prominent learning algorithms for ANN. BPNL uses gradient-decent search procedure to alter the connection weights. The structure of a BPNL algorithm is revealed in Figure 2. The output of each neuron is the accumulation of the numbers of neurons of the previous level multiplied by its corresponding weights. The input values are converted into output signals with the calculations of activation functions [4]. BPNL algorithm have been extensively and efficaciously functional in varied applications, such as pattern recognition, location selection and performance evaluations.

Press [5], Yao [6] remarks on the diffusion of electronic commerce architecture with ANN for operative formation of the systems. ANN is the choice for such a diffusion as it does not necessitate any expectations about the distribution of data. Hecht-Nielsen [7] premeditated the mathematical analysis of such a diffusion. Research also exhibited that flexibility and generalization are two most commanding facets of ANN modeling involving BPNL. Sarle [8], Wieland & Leighton [9] directed that if ANN models are instigated appropriately in an Electronic Commerce architecture, they are proficient of modeling complex patterns in data, and they can be pooled with other models to further mend the performance.

Concerning such a diffusion of ANN and Electronic Commerce architecture, experimentation conducted in the study trains the user requests from the web server through 7 iterations to expound the relative stochastic exploration of the Electronic Commerce architecture. BPNL algorithm was developed using Java programming language and employs both feed forward and back propagation approaches to amend the weights accordingly.

Complete paper is alienated into 5 sections. Section 2 interprets the Electronic Commerce architecture used for the study. Section 3 references the mathematical valuations for the BPNL algorithm. Section 4 mentions the result investigation and depicts the actual outcomes of the experiment conducted. Section 5 explicates the supposition and impending work to be steered.

II. Electronic Commerce Architecture

The proposed Electronic Commerce architecture is an

Srivastava, Riktesh, PhD, Associate Professor, Information Systems, Skyline University College, Sharjah. (rsrivastava@skylineuniversity.ac.ae)

extension to the system of Client Server Computing. In the architecture, the chore of Web Server is to yield requests from client and handover it to Application Server. The requests are further conveyed to Database Servers. So the client using the application is not fretful with the complexities of the Business logic and is presented the complete web application with supplementary service. Thus, the architecture offers a momentous workload shift.

In the proposed architecture, the Web server has to no longer do the entire profound lifting when it comes to running applications. The Application Server and Database Server(s), hold the impediments of the architecture. Also, the hardware and software demands on the user's side dwindle and the web server only executes the architectures interface software. The comprehensive architecture is depicted in Fig 1.

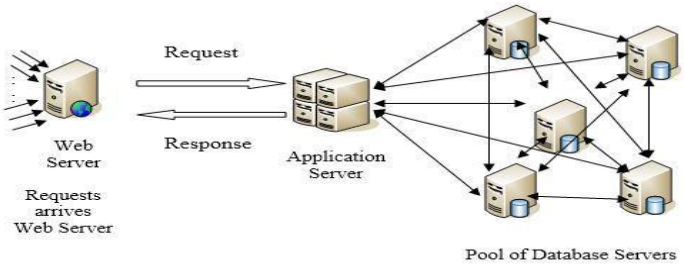

Fig. 1 Electronic Commerce Architecture

As illustrated in Figure 1, all the clients requests is being established at the Web Server. Once the requests are received at Web Server, it gets transported to the Application Server for further dispensation. Application Server, spawns the business logic and then requests gets conveyed to the pool of database servers. At the Database server, the requests gets searched and evolves 90% of the query time. The research is an attempt to implement BPNL algorithm at this level, to train the requests, in order to get the faster response time. Fig 2 given below illustrates the linkage of the Electronic Commerce architecture to the Artificial Intelligence (AI) model. For BPNL algorithm, Hidden Layer is introduced for weight updates.

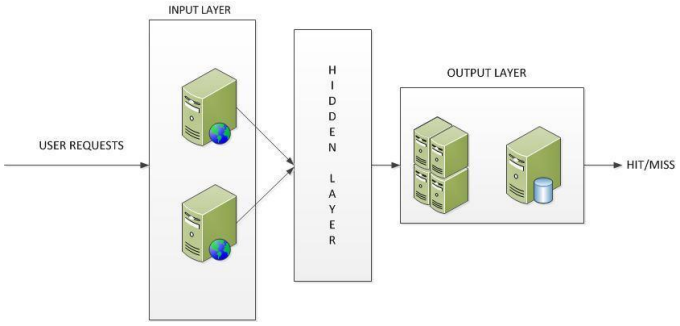

Fig. 2 AI and Electronic Commerce Architecture

III. Back Propagation Request Learning (BPNL) Architecture

Fig 3 represents the architecture of a Simple Neural Networks. As portrayed in the Figure, we have one hidden layer, which is associated to the node in output layer. There is customarily some weights associated with every connection. As depicted in Fig 1, at the input layer, we get the requests from the client, which is usually the raw information. This raw information is fed into the network and gets transferred to the hidden layer, as depicted in Fig 3. Hidden layer accepts data from the input layer. It uses input values and modifies them with some weight value, this new value is then send to the output layer but it will also be adjusted by some weight from connection amongst hidden and output layer. Output layer process information received from the hidden layer and produces the output. This output is then processed by activation function.

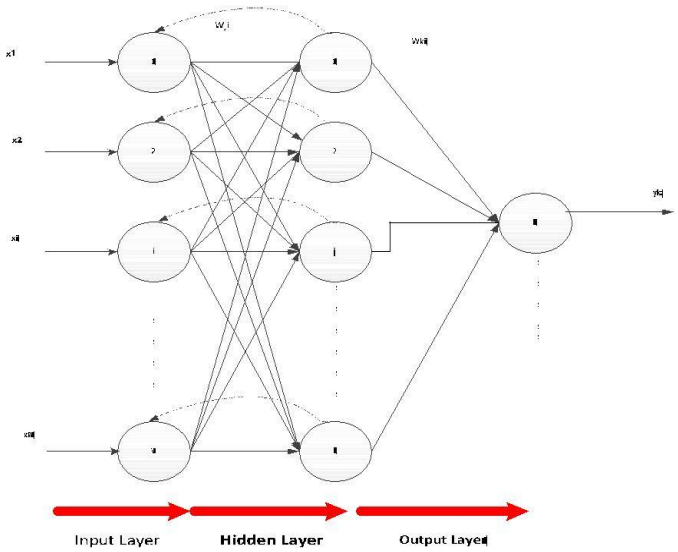

Fig. 3 Neural Network approach for scheduling requests

A. Mathematical Evaluation of BPNL Algorithm

The BPNL founds its base on the study piloted by (Raul Rojas, 2005), which claims that the complete algorithm should be broken into four stages. After selecting the weights of the network randomly, the BPNL is used to compute the necessary corrections. The algorithm can be decomposed in the following four steps:

A. Feed-forward computation
B. Back propagation to the output layer
C. Back propagation to the hidden layer
D. *Weight updates*

The algorithm is clogged when the assessment of the error function has become adequately insignificant. This is very rough and rudimentary assumption for BPNL algorithm. There are some variation but BPNL algorithm based on (Raul Rojas, 2005)] elucidation seems to be fairly precise and easy to follow. The last step, weight updates is happening throughout the algorithm.

BPNL algorithm is being assessed based on the number of requests incoming at the Web Server. The purpose of the algorithm is to accomplish fast response time, after instigating the BPNL algorithm amid Application Server and Database

Server. Employment of BPNL algorithm for Fig 1 is depicted in Fig 3 as given below:

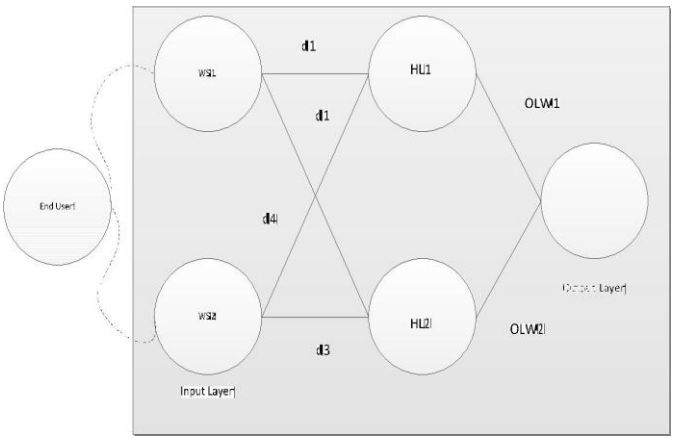

Fig. 4 Implementation of BPNL Algorithm

IV. Generic BPNL Algorithm

The generic BPNL Algorithm used for the experiment is mentioned below. The complete algorithm involves 4 steps with repeated iterations to get the desired output.

1 Initialize weights for the request type (small random numbers)
2 For each training of user requests
3 Repeat until weights convergence or till a required number of epochs are completed
   i. Receive requests as it will be extracted from various queues
   ii. Propagate the error backward from output layer to hidden and input layer.
   iii. Calculate new weights in accordance with BPNL algorithm.
4 Replace old weights new weights as taken from training algorithm
   i. After every ' t' time units
   ii. Measure performance of each requests
   iii. Repeat until performance falls below a threshold level (Δ) else go to Step iv.
   iv. Set activation of input unit. Inputs to input layer will be actual packets that are to be scheduled.
   v. Compute output of hidden and output layer using sigmoid activation function
   vi. Output will be fed to weight decider module which will calculate the required change in weights of the queues.

V. Research Outcomes

The section illustrates the research outcomes of the experiment conducted. 9 repetitions of the experiment was conducted. Fig 5(i) and 5(ii) and Fig 6 describes the outcomes of the results respectively.

```
1000  0  Expected: -1.0  Actual: -0.974
1000  1  Expected: -1.0  Actual: -0.973
1000  2  Expected: 1.0  Actual: 0.970
1000  3  Expected: 1.0  Actual: 0.970
1000  4  Expected: -1.0  Actual: -0.981
1000  5  Expected: 1.0  Actual: 0.970
1000  6  Expected: -1.0  Actual: -0.977
1000  7  Expected: -1.0  Actual: -0.981
Training Error: 0.026

2000  0  Expected: -1.0  Actual: -0.982
2000  1  Expected: -1.0  Actual: -0.981
2000  2  Expected: 1.0  Actual: 0.979
2000  3  Expected: 1.0  Actual: 0.979
2000  4  Expected: -1.0  Actual: -0.986
2000  5  Expected: 1.0  Actual: 0.979
2000  6  Expected: -1.0  Actual: -0.984
2000  7  Expected: -1.0  Actual: -0.987
Training Error: 0.018

3000  0  Expected: -1.0  Actual: -0.985
3000  1  Expected: -1.0  Actual: -0.985
3000  2  Expected: 1.0  Actual: 0.983
3000  3  Expected: 1.0  Actual: 0.983
3000  4  Expected: -1.0  Actual: -0.989
3000  5  Expected: 1.0  Actual: 0.983
3000  6  Expected: -1.0  Actual: -0.987
3000  7  Expected: -1.0  Actual: -0.989
Training Error: 0.015

4000  0  Expected: -1.0  Actual: -0.987
4000  1  Expected: -1.0  Actual: -0.987
4000  2  Expected: 1.0  Actual: 0.985
4000  3  Expected: 1.0  Actual: 0.985
4000  4  Expected: -1.0  Actual: -0.990
4000  5  Expected: 1.0  Actual: 0.985
4000  6  Expected: -1.0  Actual: -0.989
4000  7  Expected: -1.0  Actual: -0.991
Training Error: 0.013
```

Fig 5(i): Experimental Outcomes

```
6000  0  Expected: -1.0  Actual: -0.989
6000  1  Expected: -1.0  Actual: -0.990
6000  2  Expected: 1.0  Actual: 0.988
6000  3  Expected: 1.0  Actual: 0.988
6000  4  Expected: -1.0  Actual: -0.992
6000  5  Expected: 1.0  Actual: 0.988
6000  6  Expected: -1.0  Actual: -0.991
6000  7  Expected: -1.0  Actual: -0.992
Training Error: 0.010

7000  0  Expected: -1.0  Actual: -0.990
7000  1  Expected: -1.0  Actual: -0.990
7000  2  Expected: 1.0  Actual: 0.989
7000  3  Expected: 1.0  Actual: 0.989
7000  4  Expected: -1.0  Actual: -0.993
7000  5  Expected: 1.0  Actual: 0.989
7000  6  Expected: -1.0  Actual: -0.992
7000  7  Expected: -1.0  Actual: -0.993
Training Error: 0.010

8000  0  Expected: -1.0  Actual: -0.991
8000  1  Expected: -1.0  Actual: -0.991
8000  2  Expected: 1.0  Actual: 0.990
8000  3  Expected: 1.0  Actual: 0.990
8000  4  Expected: -1.0  Actual: -0.993
8000  5  Expected: 1.0  Actual: 0.990
8000  6  Expected: -1.0  Actual: -0.992
8000  7  Expected: -1.0  Actual: -0.993
Training Error: 0.009

9000  0  Expected: -1.0  Actual: -0.991
9000  1  Expected: -1.0  Actual: -0.992
9000  2  Expected: 1.0  Actual: 0.990
9000  3  Expected: 1.0  Actual: 0.990
9000  4  Expected: -1.0  Actual: -0.994
9000  5  Expected: 1.0  Actual: 0.990
9000  6  Expected: -1.0  Actual: -0.993
9000  7  Expected: -1.0  Actual: -0.994
Training Error: 0.008
```

Fig 5(ii): Experimental Outcomes

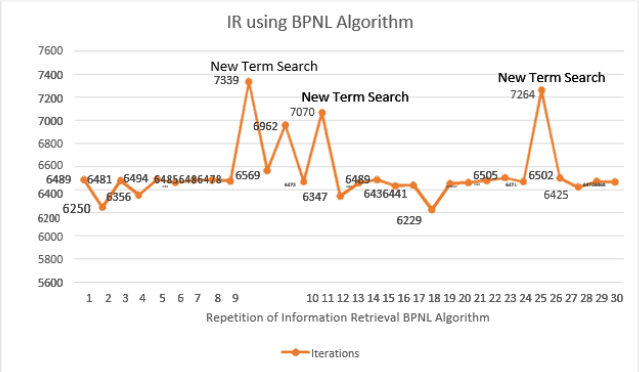

Fig. 6 Information Retrieval performance using BPNL Algorithm

VI. Further Research and Recommendations

As evidently specified in the research that employment of BPNL in Electronic Commerce architecture profoundly upsurges the systems performance. Figure 5 and 6 represents the outcomes of the investigation accompanied without BPNL algorithm and after 9 sets of 30 repetitions training using BPNL Algorithm. Originally, it was pragmatic that BPNL algorithm does not have any influence on the results, however, incessant training has an inclusive influence on the system. The study exhibits that the algorithm gives the enhanced consequences with concentrated 30 repetitions being acknowledged. The recommendations are:

1. Implementation of BPNL Algorithm at the Database server level increases the performance of the system
2. Decreases the Response time of the system

For future, it is planned to evaluate the response time of the system before and after training using various queuing models based on Ergodic condition.

References

[1] S. Shrivastava and M. P. Singh, Performance evaluation of feed-forward neural network with soft computing techniques for hand written English alphabets, Applied Soft Computing, vol.11, no.1, pp.1156-1182, 2011. http://dx.doi.org/10.1016/j.asoc.2010.02.015

[2] B. Karimi, M. B. Menhaj and I. Saboori, Multilayer feed forward neural networks for controlling de-centralized large-scale non-affine nonlinear systems with guaranteed stability, International Journal of Innovative Computing, Information and Control, vol.6, no.11, pp.4825-4841, 2010.

[3] D. E. Rumelhart, G. E. Hinton and R. J. Williams, Learning representations by back-propagating errors, Nature, vol.323, no.6088, pp.533-536, 1986. http://dx.doi.org/10.1038/323533a0

[4] M. N. Hajmeer and I. A. Basheer, A hybrid bayesian-neural network approach for probabilistic modeling of bacterial growth/no-growth interface, International Journal of Food Microbiology, vol.82, no.3, pp.233-243, 2003. http://dx.doi.org/10.1016/S0168-1605(02)00308-2

[5] Press, L., "Tracking the Global Diffusion on the Internet," Communications of the ACM, Vol. 40, No. 11:11-17, 1997. http://dx.doi.org/10.1145/253671.253678

[6] Yao, J.T., "Ecommerce Adoption of Insurance Companies in New Zealand," Journal of Electronic Commerce Research, Vol. 5, No. 1: 54-61, 2004.

[7] Hecht-Nielsen, R., "Neurocomputing," Reading, MA: Addison-Wesley, 1990.

[8] Sarle, W. S., "Stopped Training and Other Remedies for Overfitting," Proceedings of the 27th Symposium on the Interface of Computing Science and Statistics, 1995.

[9] Weiland A. and R. Leighton, "Geometric Analysis of Neural Network Capabilities," Technical Report, Arpanet III 385-392., 1988.

[10] Raul Rojas, "Neural Networks: A Systematic Introduction", Springer, 2005.

[11] www.dspguide.com, "Introduction to Neural Networks", http://www.dspguide.com/CH26.PDF, 201.