SERVER LOAD BALANCING THROUGH QUANTUM RADIAL BASIS FUNCTION NEURAL NETWORK ALGORITHM

Mrs. Ankitha Philip

Assistant Professor

*Dept. of Computer Applications*

*Amal Jyothi College of Engineering*

Kanjirappally,Kerala

Email: ankithaphilip@amaljyothi.ac.in

Jocely Joseph

PG Scholar

*Dept. of Computer Applications*

*Amal Jyothi College of Engineering*

Kanjirappally, Kerala

Email: jocelyjoseph2021@mca.ajce.in

***Abstract*-This paper is all about the implementation of quantum radial basis neural network algorithm to increase the speed of server load balancing. The amount of data reaching the servers or the load of each servers is very much high and uncontrollable by itself. The servers load must be transferred to different servers to reduce the traffic of data in each server. The data load in server is transferred to the other that are in the same cluster. The server cluster is a group of servers connected together to give high availability of data for the users. There should be some kind of algorithm that could identify the increasing loads in each server and transfer the data to different servers. Here I have used Quantum RBF neural network as the load balancer which acts as traffic police in maintaining the traffic, that would transfer server load between different servers within seconds. Since the algorithm used here is a machine learning algorithm the QRBF load balancer is trained for learning from the situation and act accordingly to the specific techniques already implemented into the QRBF load balancer. Normally a neural network may take so much of time for predicting or finding out the result but if the quantum computing technique is used then the process of load balancing done by QRBF load balancer will be done in a fraction of seconds, because of the superposition and entanglement properties of quantum computing. This work will be very useful for the people who work on server load balancing.**

***Keywords-qubits, radial basis function neural network, quantum computing, superposition, entanglement.***

1. **INTRODUCTION**

Let’s begin with some definitions that is very much important in my work.

* Quantum Computing.

Quantum computing is way of computing the data based on Quantum mechanics. Quantum mechanics is the fundamental principle in Physics that is used to describe the properties of different substances at its atomic and subatomic levels. Quantum computing is a technique that uses qubits as its basic structure just like the bits in classical computing. As we already know that 0s and 1s are used to represent bits but the qubit or quantum bit is also very much similar to the classical bits 0s and 1s but in between there is an another bit state that is the situation in which it can be both 0 and 1 at the same time. The qubit state 0 denotes the beginning of a specific activity, both 0 and 1 state determines the intermediate level and the qubit state 1 denotes the final measurement of the activity. The processing of a qubit can be clearly understood by tossing a coin i.e. when we take the coin in our hand it denotes the 0 state, when the coin is tossed up in the air the coin can have any one of the two side that is tails or heads this denotes the intermediate state in the qubit i.e. both 0 and 1 state but when it reaches back to our hand it will be either heads or tails, this measurement denotes the 1 state in qubit. This superficial activity of qubit is due to the superposition and entanglement properties.

* Radial Basis Function Neural Network

Neural network is a part of the artificial intelligence which is developed very much similar to our brain. The neurons in our brain mainly has three parts - i) dendrite that takes information to the neuron, ii) cell body where the processing of all data is done, and iii) axon that transmits the processed data to the specific part of the body. Very much like the working of biological neurons the AI neural network has also three layers that is the input layer that takes in the input, the hidden layer where all the processes happen and the output layer which gives the final output. The hidden layer uses a specific threshold function to activate each neuron in the hidden layer. The threshold function or the activation function determines whether the neurons are active for sending data to the next layer. The RBF network is type of feedforward neural network that uses radius based function that is the Gaussian function as the activation function for the correct prediction of the output.

* Load Balancing

Load balancing refers to the distribution of incoming network traffic across a group of backend servers. The modern servers have a very high traffic that has to answer to hundreds of thousands of simultaneous requests from users or clients and return the correct text, images, video, or application data, all in a fast and reliable manner. To meet the needs of all the users many number of servers are kept connected in clusters. A load balancer acts as the “traffic guard” sitting in front of the servers and routes the client requests across all servers that is able to fulfill those requests in a way that increases the speed and capacity utilization of each server in the cluster and ensures that no one server is overworked, which would reduce the performance. If one server goes down, the load balancer changes the data traffic to the remaining online servers. And when a new server is added in the replacement of the degraded server to the server cluster, the load balancer automatically starts to send requests to it.

The use of QRBF network algorithm as the starting algorithm in the load balancer will allow the complete network of servers and clients to complete the work within few milliseconds without loss of energy and decreases the server utilization.

1. **METHODOLOGY**

The RBF neural network software is added on to the load balancer system that will balance the load in each backend servers. By adding quantum computing capabilities to the RBF algorithm the time consumption of the RBF network is reduced to very few milliseconds. To make the load balancer a Quantum computing capable machine qubits should be added to the system instead of the classical bits. The more number of qubits the system’s entire work will be completed within a few milliseconds. The RBF network has three layers i.e. the input layer where all client request comes and then based on the radial distance of different destined backend servers the client request is given a weighted influence, the farther the backend server of a specific client request is from the load balancer the client request will have lesser weighted influence.

One neuron in the input layer corresponds to each predictor variable. Each neuron in the hidden layer consists of the radial basis function centered on a point with the same dimensions as the predictor variables. And the output layer will be the weighted sum of hidden layer processes.

The RBF neural network is almost similar to KNN models. The basic idea is that the present client request must be moved on to the same server where the some of the previous client requests have been sent. A Euclidean distance is computed from the point of the neuron that contains the specific client request in the input layer to the servers in the hidden layer and based on the weighted influence the result of the request reaches the client.

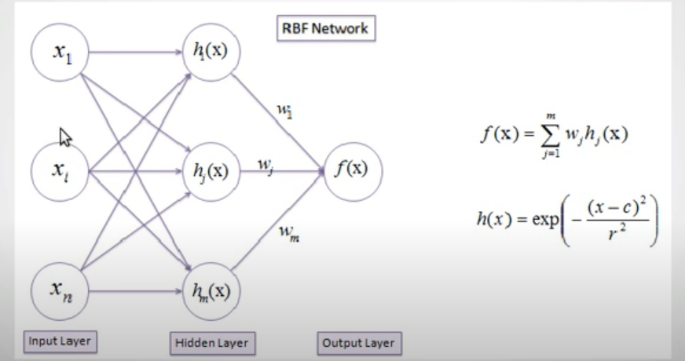


Figure 1: RBF Network

RBF algorithm

* H(x) is the Gaussian activation function with the parameters r (the radius) and c (the average taken from the input space) defined separately at each RBF unit,
* The learning pattern is based on adjusting the parameters of the network to reproduce a set of input-output patterns.
* There are three types of the weight w between the hidden nodes and the output nodes, the center c of each perceptron of the hidden layer and the unit width r.
* The center c can be found out using K-means clustering algorithm.
* The radius r can be found out using K-nearest neighbor algorithm.
* After calculating r and c RBF is added to compute the weighted influence of each request or neuron.

The following code can be used to make the entangled qubit for faster processing of the entire system:

# Import libraries

from qiskit import ClassicalRegister, QuantumRegister, QuantumCircuit

from qiskit import execute

from qiskit import BasicAer

 # import basic plot tools

from qiskit.tools.visualization import plot\_histogram, circuit\_drawer

In [2]:

#Selecting the qasm simulator as backend for executing the circuit using BasicAer

backend = BasicAer.get\_backend('qasm\_simulator')

In [3]:

q = QuantumRegister(2) # A qauntum register of size 2 qubits

c = ClassicalRegister(2) # A classical register of size 2 bits to measure the probability

qc = QuantumCircuit(q, c) # Making a quantum circuit

 # Add a Hadamard Gate at q0 bit

qc.h(q[0])

 # Add a controlled-NOT Gate with control bit at q0 and target bit at q1

qc.cx(q[0], q[1])

 # Measure the circuit

qc.measure(q, c)

 #Execute the circuit

job\_exp = execute(qc, backend=backend, shots=1024)

In [4]:

print('You have made entanglement!')

You have made entanglement!

In [5]:

#Representing the circuit

qc.draw()

Out[5]:

┌───┐ ┌─┐

q0\_0: |0>┤ H ├──■─────┤M├

└───┘┌─┴─┐┌─┐└╥┘

q0\_1: |0>─────┤ X ├┤M├─╫─

└───┘└╥┘ ║

c0\_0: 0 ═══════════╬══╩═

║

c0\_1: 0 ═══════════╩════

1. CONCLUSION

The use of quantum RBF network for server load balancing would be a thriving activity to increase the speed of load balancing in servers as now a very big amount of data is transferred over the Internet.

1. References

\*<http://livescience.com>

\*<http://ngnix.com>

\*<http://visualstudiomagazine.com>

\*Quantum computing course from edx

\*RBF network course from SimpliLearn