# Predictive Scaling for Elastic Compute Resources on Public Cloud Utilizing Deep Learning based Long Short-Term Memory

Bharani Dharan
The New College

# Predictive Scaling for Elastic Compute Resources on Public Cloud Utilizing Deep Learning based Long Short-term Memory

Bharanidharan. G[1]
Phd. Research Scholar, Department of Computer Science
VISTAS, Pallavaram, Chennai

Dr. S. Jayalakshmi[2]
Professor, Department of Computer Applications
VISTAS, Pallavaram, Chennai

***Abstract*—The cloud resource usage has been increased exponentially because of adaptation of digitalization in government and corporate organization. This might increase the usage of cloud compute instances, resulting in massive consumption of energy from High performance Public Cloud Data Center servers. In cloud, there are some web applications which may experience diverse workloads at different timestamps that are essential for workload efficiency as well as feasibility of all extent. In cloud application, one of the major features is scalability in which most Cloud Service Providers (CSP) offer Infrastructure as a Service (IaaS) and have implemented auto-scaling on the Virtual Machine (VM) levels. Auto-scaling is a cloud computing feature which has the ability in scaling the resources based on demand and it assists in providing better results for other features like high availability, fault tolerance, energy efficiency, cost management, etc. In the existing approach, the reactive scaling with fixed or smart static threshold do not fulfill the requirement of application to run without hurdles during peak workloads, however this paper focuses on increasing the green tracing over cloud computing through proposed approach using predictive auto-scaling technique for reducing over-provisioning or under-provisioning of instances with history of traces. On the other hand, it offers right sized instances that fit the application to execute in satisfying the users through on-demand with elasticity. This can be done using Deep Learning based Time-Series LSTM Networks, wherein the virtual CPU core instances can be accurately scaled using cool visualization insights after the model has been trained. Moreover, the LSTM accuracy result of prediction is also compared with Gated Recurrent Unit (GRU) to bring business intelligence through analytics with reduced energy, cost and environmental sustainability.**

***Keywords*—Predictive auto-scaling; business intelligence; virtual machines (VM's); deep learning models; analytics; elasticity; high performance public cloud data centre (HP-PCDC); right sizing**

## I. Introduction

Cloud Computing (CC) is a paradigm aimed at retrieving a collection of computer assets such as servers, networks, storage that allow applications to expand resource on demand with agility through virtualization. Cloud computing is the developing model for delivering subscription oriented pay-as-you-go services to access compute resources for deploying applications. One such characteristic, namely, elasticity that enables customers to buy and release the correct quantity of compute resources based on their demands so that more responsive web application developers are continually attracted to use cloud services. Cloud infrastructure and services have become the major aspect [1] [2]. The previously siloed or on-premise data Centre is obsolete immediately, since the client base tends to shift quickly based on business demands. The auto-scale technology from IaaS in cloud enables the dynamic adjustment of the number of VMs, to be added or removed based on the capacity of workloads or user traffic [3]. If a web shop in the cloud has more requests or with seasonal trends, extra VMs can be made available to handle load. Conversely in the case of reduction of traffic, VM instances may also to be removed automatically. The application load balancer supports containerized applications powerfully. It operates like a gateway to receive TCP/HTTP applications received by end-users and disseminate it equally to many clusters with master and slave nodes managed by Kubernetes (K8's) minions to support modern containerized workloads. In this research, we utilize a framework intended for supporting classic and application load balancing, with predictive Auto-scaling approach which is used to forecast the traffic in advance and provision of resources with high-quality services and to minimize the expense of cloud utilization without violating Quality of Service (QOS) and Service Level Agreement (SLA).

Projection of cyclic workload is one of the most crucial and vital aspect in the perfect management of cloud infrastructure. Each request necessitates resources to finish its implementation and such resources are virtually made available through resource pool. The cutting edge CDC consists of different resources such as bandwidth, software, CPU, memory etc., in a virtualized form through Type-1 hypervisors from hardware layer whereas the users are assigned to finish their task performance on request. In accordance to preceding works, it is important to remember that resources are always larger than the real resources needed in order to finish the application [4]. The rationale for the supply of resources is that SLA violations are evaded and QoS satisfaction is attained. The resources are in most cases squandered during the allocation procedure. Although auto-scaling offers extremely excellent advantages, it is a difficult process to execute. Effective auto-scaling involves a new predictive strategy to predict the resources exactly using Machine Learning (ML) or Deep Learning (DL) to handle critical workloads. Auto-scaling shown in Fig. 1 comprises of three techniques, namely:

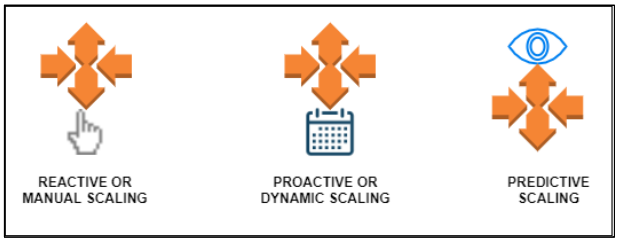

Fig. 1. Types of Scaling Techniques.

The strategy of reactive or manual auto-scaling where the resources has been wasted most of the times when workloads may get exceeded suddenly or decreased due to user specifies the threshold values and the required instances are automatically optimized as per the values hardcoded in the policy. The Proactive Scaling is Reactive with time scheduling to scale up or scale down the resources. In the case of predictive strategy, the cloud application workloads have been predicted through analyzing and visualizing those historical workload traces with minimum 24 hours of metrics history is needed to add or remove the instance capacity in before the first arrival of traffic. Thus, the predictive scaling is more efficient than reactive scaling and even familiar due to its spontaneous nature to allocate instances in advance to maintain optimum performance and the limitation in the predictive scaling is mixed instance policy in Auto-Scaling Groups cannot be supported, supports only CPU Metrics to forecast, and suits only for the applications which undergo periodic traffic spikes.

Precise predictions can be utilized to determine the right quantum of resources required to meet the needs. In order to accurately estimate the future workload, a precise and trustworthy prediction model is necessary. Normally in CDC, the job of the user comes in a pattern with irregular requirements on the resource. This is a big difficulty to anticipate the accurate workload [5]. Researchers have created many models for estimating workload and resource usage, focused mostly on forecasting the memory and CPU [6-8]. Several research projects have solely utilized statistical approaches to estimate load and for huge and diverse data, they cannot anticipate reliable outcomes. By means of machine learning models, numerous study efforts is done to forecast large and changing working loads in cloud. The results of these predictions are shown to be encouraging. Nevertheless, in regulated settings the statistical models can proactively anticipate time burdens. It is thus known that when using heterogeneous data, it will lead to greater predictability by integrating both statistical and machine teaching technologies. However, somewhat few research works were carried out in a field of resource forecasting at task stage [9]. The use of resources at task level helps to characterize activities that have a substantial influence on the capacity planning process, development of VM and assignment apportionment [10].

The CPU usage is one of the key metrics for the performance assessment of the cloud data center host and is utilized by researchers for server performance assessment [11]. CPUs are the utmost challenging resource in cloud data center servers and hence the main reason for not having a resource [12]. For cloud resource usage prediction and capacity planning of CPU, several approaches have been employed namely, Auto-Regressive Moving Average (ARIMA), K-Nearest Neighbor (KNN), Feed-forward artificial Neural Network (FNN), Auto-Regression (AR), Recurrent artificial Neural Network (RNN), Extreme Learning Machine (ELM), Autoregressive Neural Network (AR-NN),and Multi-Layer Perceptron (MLP)[13], [14]. The use of CPU may be regarded as time function such that it could be pronounced as a time series issue. As a result, it becomes a regression problem that may be handled using neural networks or traditional time series.

The aim of the research is to avoid reactive nature of auto-scaling by introducing the predictive nature of auto-scaling solutions with AWS cloud platform which has the capacity to comply with the constantly changing load with QoS standards. The evaluated combinations of Maternal Data Centers in Dortmund and AWS Auto Scaling of Elastic Compute Nodes have been utilized for accessing the predictive auto scaling options. However, the trials are carried out using the LSTM deep learning concept with Explorative Data Analysis (EDA) to forecast the resource needed in advance by means of predictive scaling with visual analytics that has a capacity to auto-scale in and out to the Dev-Ops demand quickly. Hence, this technique is initially presented in the document has utilized to assess auto-scaling performance. Thus, the DCs are one of the biggest responsible for global warming and it our responsible to find some innovative ways for overcome this issues. The initiative of green cloud broker with LSTM based predictive auto-scaling has assisted to reduce CPU utilization, memory usage as well as energy consumption values to maintain in its directory.

The paper discusses the literature of elastic scaling of predictive method using time series forecasting method as well as auto-scaling of resource utilization using Machine Learning (ML) in Section 2. The Section 3 discusses the predictive auto-scaling of Dev-Ops user's resource allocation by EDA and LSTM technique for better and accurate prediction of unprecedented workload. Section 4 discusses experimental AWS platform and visualized real-world workloads to evaluate the predictive scaling approach. Section 5 has concludes that predictive auto-scaling by LSTM has improved the Dev-Ops users resource allocation for developing and running the applications with less compute and memory usage in a lesser time.

## II. Literature Review

The IaaS cloud uses predictive scaling technologies that guarantees cloud energy efficiency with reduced bill cost. Bi Jing et al. suggested a technique to forecast the amount of tasks at a successive interval in data center using ARIMA and hair wavelets, and findings showed that hybrid approaches lead to a greater predictive accuracy. The work does not address computer resources such as CPU, RAM, etc. which play a key role for the distribution of resources [15].

Janardhanan et al., in the Google cluster data utilized ARIMA and LSTM models to predict CPU workloads. The outcomes show that LSTM is 20 percent lower than the ARIMA model and has a higher consistency in its predictions [16]. The experiment on a single computer is implemented and the use of CPU for this machine is projected. Furthermore, Cetinski et al. suggested an Advanced Model for Efficient Workload Prediction in the Cloud (AME-WPC) that utilizes both statistical and machine learning techniques to improve the job prediction precision over time. In aspects of lowering operational costs and handling resource, the proposed strategy is efficient. However, the prediction of certain cloud resources would help to automate and schedule the scale of resources [17].

In a novel method, M. U. Farooq et al. suggested RR. Initially, the most time of explosion is retrieved in RQ. The quantum time value is then derived by computing the 0.8 percent of this BT. If the number of processes in the queue is less than the quantum, they will be given the CPU, while the others will be queued. When all of the processes are completed, a quantum is given a new value equal to the highest BT, and the CPU is freed up for the remaining activities. This new technique worked wonderfully in terms of average waiting times, context switches, and turnaround times [18].

B. Dave et al. recommended a unique method to CPU scheduling the strategy's objective is to compute the best quantum time depending on the left over burst time in ready queue for the said work. Various tests have been carried out to determine the efficacy of this method. In evaluation to DQRRR, SARR, RR, MRR and IRRVQ algorithms, the findings show a reduction in the amount of context switches in queues or resources as well as outperformance [19]. A. Kaushik and D.Khokhar presented a novel approach to solve the RR algorithm's drawbacks. They developed a novel method for determining the optimum time quantum using the mean and median burst time of activities. Experience has proven the effectiveness of a new algorithm in terms of reducing waiting time and turnaround time [20].

Priyanka Singh, Palak Baaga, and Saurabh Gupta offer a detailed overview of the numerous methods that have been published earlier in their paper Assorted Load Balancing Algorithms in CC. Researchers looked at different algorithms and analyzed them based on numerous criteria to find a viable solution for load balancing in a CC environment. The algorithms' benefits and drawbacks are explored. The Minimum Connections Algorithm is one such algorithm. This approach takes the number of active connections that each server will have into consideration. Once a client tries to connect, the load balancer looks for the server with the fewest connections and allocates newer connections to that server. The Least Connections Algorithm is named from the fact that each server's connection is taken into account. It only prevents the server from becoming overburdened [21].

In their paper "Performance Evaluation of Round Robin (RR) Algorithm in Cloud Context," Neethu Myshri, R and Asha, M. L have highlight the RR algorithm's performance in a cloud-based atmosphere. For the goal of simulating and understanding the reaction of cloud computing and its deployment patterns, the use of a cloud analyst toolset has been chosen as a unique method. The proposed method is fairly similar to the Throttled algorithm. Load balancing is achieved in a way of delivering requests in a round-robin way to each server. RR is a typical load balancing scheduling technique for distributing workload amongst servers. This method works well on clusters of servers that have the same specifications. It picks a node at random and distributes the job in a circular pattern [22]. Despite the fact that round robin is the simplest approach for distributing client requests over a set of servers, it suffers from non-uniformity in workload distribution owing to the servers' specs being similar. Furthermore, the Round Robin algorithm ignores priority, resource capabilities and job size. As a result, higher-priority and longer-duration operations have longer response times that can lead to server overloading.

Ming Yan et al. [23] presented a fusion elastic scaling strategy for Kubernetes (k8's) that combined reactive and proactive approaches. The proactive technique uses the Bi-LSTM model to learn the physical host and pod resource consumption history in order to anticipate future workload (Memory usage, CPU utilization). The Bi-LSTM prediction model is used with the online reinforcement learning with reactive model to achieve elastic scaling judgments. It has been shown in experiments that it can help the system achieve micro service SLAs in edge computing environments. The Bi-LSTM model has the least prediction error for the Root Mean Square Error (RMSE) metric when compared to ARIMA, LSTM, and RNN models. Despite this, no strategy for reducing oscillations has been offered.

Machine learning-based auto-scaling architecture is also suggested by Imdoukh et.al. The resource estimator tested the 1998 World Cup website dataset with the help of the LSTM model. They compared the results to those obtained using both the ANN and ARIMA models. The findings showed that while the proposed LSTM model has a little higher prediction error in one-step forecasting than the ARIMA model, it predicts 530 to 600 times faster [24].

Tang et al. proposed a Bi-LSTM-based container load prediction model that forecasts future load based on the container's historical CPU consumption. The recommended model has the lowest prediction error when compared to ARIMA and LSTM models. The authors, on the other hand, give no instructions on how to set up the parameters of the proposed model. Furthermore, the essay focuses only on future load projections and does not address auto-scaling concerns [25].

Mahmoud Imdoukh et al. proposed predictive auto-scaling for running container applications with docker containers for handling dynamic workload characteristics and for timely manner provisioning. The authors used the MAPE (Monitor, Analyzer, Planner and Executor) in auto-scaling architecture connected with Time-Series database. Further LSTM prediction model is used with multi step prediction and it as been compared with ARIMA where LSTM is 130 times faster to predict the real time usage of container auto-scaling. Further they discussed LSTM model performs better in terms of provisioning and elastic agility with auto-scaler metrics [26].

This session has discussed the literature instance of forecasting model of cloud computing resource allocation, provisioning and predictive scaling using Machine Learning ARMA, ARIMA, deep learning models like LSTM, Bi-LSTM that have played a major role in proactive scaling and to overcome the reactive scaling gaps. Similarly, the discussion about load balancing schedule technique by dynamic Round Robin algorithm has utilized for better workload scheduler rather than classic load balancer. However, the support of this literature has addressed the issues of reactive auto-scaling and advantage of predictive auto-scaling by LSTM model. Therefore, this paper has motivated to fill the research gap using Predictive scaling on public cloud infrastructure by LSTM.

## III. Research Methodology

The cloud computing applications are involved with large variation in both development and operation areas that consist of an alternative existence during the solutions of suitable cloud infrastructure. The middleware solutions have involved by accomplishing Dev-Ops requirements with green cloud broker. However, this research has proposed a predictive scaling with green broker management method to maintain the elastic provisioning of VMs based on Public Cloud Service Provider (PCSP) to resolve Dev-Ops requirements without compromising QOS and SLA. Hence, deep learning technique has played a key role in predicting the required resource and allocates it to CSP with Dev-Ops knowledge that currently spread across various VMs or docker containers through virtualization or containerization in the form of elastic cluster management. The allocation of predictive VM resources using dynamic weighted Round Robin (RR) which is applied in scheduling the task based on the CPU time with weights for distribution over the Virtual Machines in a physical host. The RR technique efficiency act as the application load balancer that completely depends upon quantum or traffic. When the application traffic is high or quantum size is very large, then the RR technique may follow the first come first serve method with weights These application load balancers is called as new generation load balancer. It handles multiple applications on a unique physical host that contains VM's. The routing decisions are carried out in layer 7.

The proposed infrastructure of predictive scaling using LSTM technique is shown in Fig. 2. When the Dev-Ops users have cyclic workloads that have been sending through green cloud broker are sequenced as cloudlets by dynamic round robin technique applied in application elastic load balancer for regulating the incoming application traffics that has been distributed equally towards multiple target optimum instances for predictive scaling.

Then the predictive scaling engine utilizes the dataset along with timestamp to forecast the CPU resource utilization to provision the instances or nodes exactly from public Cloud data Centre which has virtualized resources to be provisioned with agility to the dev-Ops users as shown in Table I. The traces are gathered during a three-month period in the dispersed Materna Data Centers in Dortmund [27]. A month's worth of data is shown by each trace. The three dataset consists of the running 527 VMs in trace 1, 527 VM's in trace 2 and 547 VM's in trace 3 respectively of 69 cores and 6780 GB RAM. The workloads in the tracked VMs are mission-critical business applications from well-known organizations throughout the world. It is studied using the LSTM deep learning concept as an Explorative Data Analysis (EDA) to forecast the resource needed by means of predictive scaling, having a minimum optimum capacity to auto-scale in and out to the Dev-Ops demand quickly in advance.

The steps involved in EDA has progressed LSTM algorithm by importing libraries and setting the seeds for generating random numbers by fixing a starting number. The collected raw data is made to data cleansing which removed and imputed the missing data. The data processing is done by min max scalar splitting the cleaned dataset to 60% of train dataset and 40% of validation (test) dataset which can be executed through EDA as shown in Fig. 3. Once the EDA process is done, the dataset is train to fit LSTM model based on the layer of optimizer ADAM (ADaptive Moment estimation), sequential and ReLu. Finally, the output layer of LSTM defined the predicted values through visualization report. The working principle of LSTM model as well as predictive auto-scaling algorithm using LSTM model is discussed.

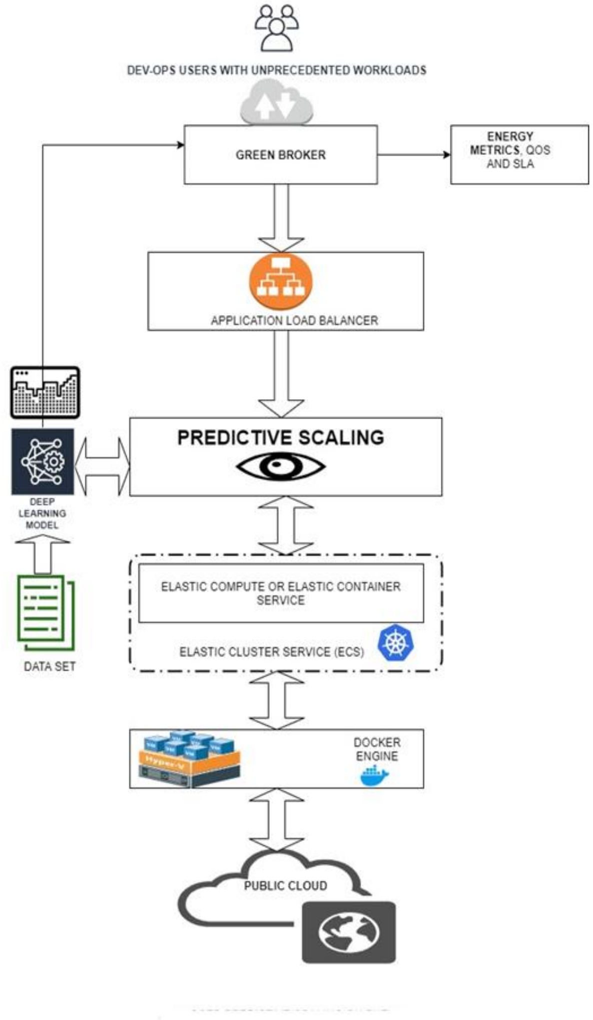
DEV-OPS USERS WITH UNPRECEDENTED WORKLOADS
GREEN BROKER
ENERGY METRICS, QOS AND SLA
APPLICATION LOAD BALANCER
PREDICTIVE SCALING
DEEP LEARNING MODEL
DATA SET
ELASTIC COMPUTE OR ELASTIC CONTAINER SERVICE
ELASTIC CLUSTER SERVICE (ECS)
DOCKER ENGINE
PUBLIC CLOUD

Fig. 2. Proposed Predictive Scaling on Public Cloud Infrastructure.

TABLE I. COLLECTION OF DEV-OPS USERS TRANSACTION AS DATASET

| Timestamp | CPU cores | CPU capacity provisioned [MHZ] | CPU usage [MHZ] | CPU usage [%] | Memory capacity provisioned [KB] | Memory usage [KB] | Memory usage [%] | Disk read throughput [KB/s] | Disk write throughput [KB/s] | Disk size [GB] |
|---|---|---|---|---|---|---|---|---|---|---|
| 11.01.2016 15:00:00 | 8 | 0 | 214 | 1,49 | 25165824 | 17407200 | 69,17 | 734 | 373 | 300 |
| 11.01.2016 17:00:00 | 8 | 0 | 137 | 0,95 | 25165824 | 251658 | 1 | 42 | 9 | 300 |
| 11.01.2016 19:00:00 | 8 | 0 | 45 | 0,31 | 25165824 | 42782 | 0,17 | 0 | 3 | 300 |
| 11.01.2016 21:00:00 | 8 | 0 | 44 | 0,31 | 25165824 | 42782 | 0,17 | 0 | 3 | 300 |
| 11.01.2016 23:00:00 | 8 | 0 | 44 | 0,31 | 25165824 | 108213 | 0,43 | 0 | 3 | 300 |
| 12.01.2016 00:00:00 | 8 | 0 | 43 | 0,3 | 25165824 | 166094 | 0,66 | 0 | 3 | 300 |
| 12.01.2016 00:05:00 | 8 | 0 | 44 | 0,3 | 25165824 | 47815 | 0,19 | 0 | 3 | 300 |
| 12.01.2016 00:10:00 | 8 | 0 | 47 | 0,32 | 25165824 | 148478 | 0,59 | 0 | 3 | 300 |
| 12.01.2016 00:15:00 | 8 | 0 | 47 | 0,33 | 25165824 | 47815 | 0,19 | 0 | 3 | 300 |

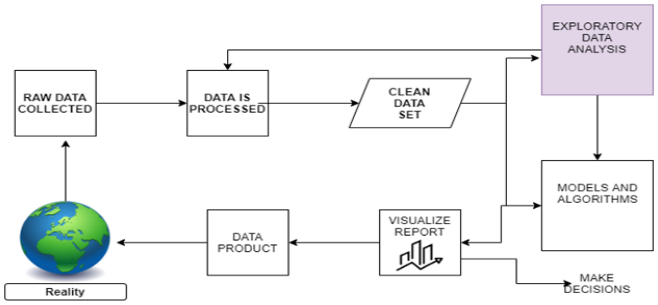

Fig. 3. Work flow of Exploratory Data Analysis (EDA).

## IV. WORKING OF LSTM AND GRU METHOD IN DEEP LEARNING

Deep Learning methods executes well on large and multivariate time-series prediction problems. The performance of learning model is better after training in complex and more non-linear data. LSTM is more accurate for large dataset with long-term predictions and ultimately deep learning has automatic feature extraction on comparing flat machine learning [28].

The working mechanism of the LSTM is illustrated in Fig. 6, and the LSTM is a kind of modern Recurrent Neural Network (RNN) architecture that remembers information at variable intervals. The LSTM algorithm is well-known for classification and forecasting time series requires lagging of timestamps with uncertain periods. One of the major advantages of LSTM is about gap length over relative insensitivity that provides better solution compared to alternative RNNs, Hidden Markov Models (HMM) and various traditional techniques. However, the case of RNN and HMM are completely depend upon the hidden state which initiated before emission and sequence results in vanishing gradient problem. Instead of predicting 10 intervals, the application is predicting 1000 intervals of sequences, the model may forgot the initial points from other technique like RNN and HMM, because the last hidden state doesn't have any information of past to remember, but it can be resolved by LSTM.

GRU (Gated Recurrent Unit) can predict well and performs fast on small datasets. The architecture of GRU which was shown in the Fig. 5 is simpler on comparing LSTM structure. The flow of information in a sequence chain can be regulated by gate structure. The GRU has only two gates known as Reset and Update gates. GRU do not use memory unit like LSTM.GRU is easier to modify. But LSTM and GRU perform moreover equal but LSTM remembers long sequences. The Full GRU Unit has shown in Fig. 4.

**FULL GRU Unit**

$$\tilde{c}_t = \tanh(W_c[G_r * c_{t-1}, x_t] + b_c)$$

$$G_u = \sigma(W_u[c_{t-1}, x_t] + b_u)$$

$$G_r = \sigma(W_r[c_{t-1}, x_t] + b_r)$$

$$c_t = G_u * \tilde{c}_t + (1 - G_u) * c_{t-1}$$

$$a_t = c_t$$

Fig. 4. GRU Unit.

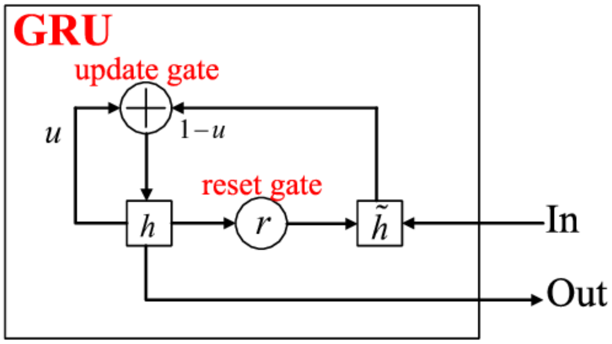

Fig. 5. GRU Working Process.

The cell state that consists of looping arrows that reference the recursive cell nature is believed to constitute the long-term memory in general. In equation 1, the long term memory acknowledges that data from previous intervals that has been stored in LSTM cell shown in Fig. 6.

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \qquad (1)$$

The forget gate, which is below the cell state and adjacent to the input modulation gate has changed the cell state. The previous cell state is forgotten, which is multiplied by the forget gate and the latest information is gathered via the input gates output, according to equation (2).

$$f_t = \sigma\left(W_{f[h_{t-1},x_t]} + b_t\right) \qquad (2)$$

The remember vector is commonly referred to as the forget gate, since its output has shown the cell state, which comprises of information contained in forgot multiplied by 0 to the matrix position. When the forget gate output is 1, the information in the cell state is retained as it is indicated in equation (3). As a result, the input gate is represented as a sigmoid function that is applied to the weighted input for observation as well as the previously concealed state.

$$i_t = \sigma\left(W_{i[h_{t-1},x_t]} + b_i\right) \qquad (3)$$

Furthermore, these gates can determine what data has to be put in the cell state, with the activation functions for each gate accounting for a considerable percentage of the total. As a result, the input gate is modeled as a sigmoid function with a range of [0,1]. If the summing of cell states among the previous cell states is supplied, the sigmoid function can only accumulate memory and not erase forgotten memories. Similarly, the float number is summation among [0, 1] which may never considered to be zero or forget. The input modulation gate in this example has been modeled after the tanh activation function indicated in equation (4) whereas the range of tanh is [-1, 1] which admit the forget memory present in the cell state shown in Fig. 6.

$$\tilde{c}_t = tanh\left(W_{c[h_{t-1},x_t]} + b_c\right) \qquad (4)$$

Thus, the focus vector is generally named as the output gate. There are several potential values in the matrix are made to move forward to the subsequent hidden state is expressed in equation 5.

$$o_t = \sigma\left(W_{o[h_{t-1},x_t]} + b_o\right) \qquad (5)$$

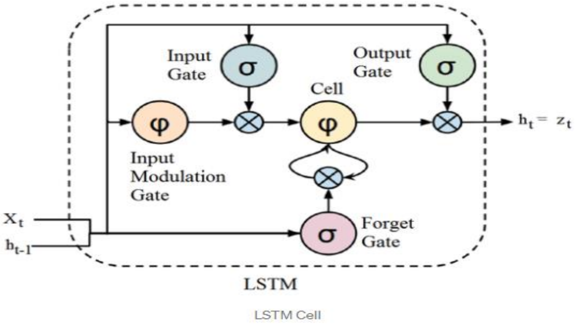

Fig. 6. LSTM Working Process.

The memory usage is generally named as hidden state whereas the data is considered for the next sequence which is illustrated in equation (6) that performs as an analog for the hidden state in RNN.

$$h_t = o_t \circ tanh(c_t) \qquad (6)$$

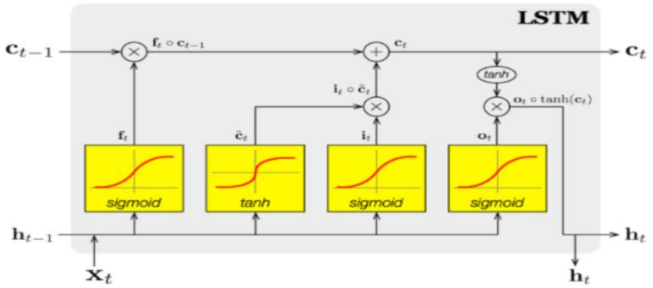

Fig. 7. LSTM Cells Activation Function.

The forget gate that aids in the forgetting of information from a previous cell state, is the sigmoid's initial activation function (Ct-1). Furthermore, the following sigmoid and initial tanh activation functions are used as input gates, and the information is either stored to the cell state or elapsed. As a result, the preceding sigmoid serves as an output gate, determining which information must be sent on to the next hidden state as shown in the Fig. 7.

**Algorithm for Predictive auto-scaling using LSTM**

**Inputs:** Resource utilization from DevOps users, VMs, Load balancer with RR technique, Data Set, CPU load capacity.

**Output:** Predicted CPU Resource utilization for auto-scaling the nodes.

**Step 1:** Request the instances by Dev-ops users for cyclic workloads.

**Step 2:** On user request the cloudlets can be sent through RR technique for further escalation to Predictive Scaling Engine to forecast the future workload demand using LSTM Model.

**Step 3:** Load the dataset for Data preprocessing which can be done through min max scalar through EDA and fit the LSTM model and GRU model for further execution to train and test.

**Step 4:** Based on the CPU workload, the data acknowledged with long term memory from earlier interval is stored in LSTM cells as per equation 1.

**Step 5:** Modification of cell state is done through forget gate as per equation 2 and the recent information gets accumulated in the output of input gate.

**Step 6:** If the forget gate is set to 1, the data is kept on its own as shown in equation 3, else the information present in the forget gate is multiplied by 0 in the matrix position.

**Step 7:** The weighted input has been applied to the optimized ADAM layer as a sigmoid function with the range [0, 1] for observation as well as prior concealed state.

**Step 8:** The forgotten memory existent in the cell state is accepted by the input modulation gate, which is a tanh activation function performed on the ReLu (Rectified Linear Unit) layer as per equation 4 with a range of [-1, 1].

**Step 9:** The hidden state representing memory usage with data is considered for the next sequence as per equation 6 that performs as sequential layer.

**Step 10:** The process of LSTM is executing the current VMs based on VM list provided through proper VM instance type by EDA.

**Step 11:** If CPU actual load_capacity>predictive workload, trigger auto-scale in policy (VM-1) else scale-out (VM+1) with cool down time and also by enabling dynamic scaling for cost optimization.

**Step 12:** Update the provisioning instance count and repeat the steps 3 to step 11

**Step 13:** Terminate VMs and return.

**Step 14:** Visualize the Results for further auto-scaling of right instances to be deployed for running the workloads in public cloud.

Auto-Trigger Policy For Predict and Scale:

```
Cat<<EOF>> predict_sclae_policy_cpu.csv // Load History of traces
{
 Region: me-south-1
 {"Metrics": [CPU], "Instance Type":t2, "Cores":8, "Unit"=MHz}

{
if{"Actual_Target_Value:123>Predicted_Targeted_value=109, VM-1 else
VM+1 "Predefined Metric Type: ASGCPUUtilization"}}
{"Mode": Predict and Scale, "Cool down time":300 seconds}} EOF
```

However, the request from the Dev-Ops user's workload are balanced through dynamic weighted RR technique and analyzed by EDA. The process of EDA workloads are performed with predictive auto scaling as auto-scale in of VMs and auto-scale out of VMs are progressed. The progressed VMs are assigned to the respective CSP users precisely with less usage of CPU capacity as well as minimized memory usage. The proposed predictive scaling by LSTM is evaluated by comparing it with Gated Recurrent Unit (GRU) method.

## V. Result and Discussion

This research experiment has utilized the AWS platform and simulated real-world workloads to evaluate the predictive scaling approach through Amazon Sagemaker or Google Co-lab compatible. The CPU core utilized is 8 with 32 GiB of memory, EBS storage, 2.20GHz Intel(R) Xeon(R) turbo boost and 6 bit platform. The host machines for this experimental instance are t2.large, t2.xlarge and t2.2xlarge of same family with burstable performance instances, all of which feature Intel scalable CPUs with speeds up to 3.0 GHz. In this paper, the implementation of the elastic resource allocation strategy is based on the Quality of Service (QoS) performance criterion. However, the recommended method has the ability to meet an appropriate demand in different kinds of varied workloads. Hence, the proposed approach has considered both reducing the CPU utilization and memory usage which progressively reduced the cost of resources utilization. The optimization outcome of QoS parameters for the proposed predictive auto-scaling by LSTM is shown in Table II reveals that the actual CPU and memory usage is high but the recommended CPU and memory usage is low and hence we can save the bill cost in usage of Compute as well as memory instances.

TABLE II. Performance Results of Predictive Auto-scaling from LSTM Method

| **DateTime** | **CPU usage [MHZ] _Actual** | **Memory usage [KB]_Actual** | **CPU usage [MHZ]_Predicted** | **Memory usage [KB]_Predicted** |
|---|---|---|---|---|
| 2016-02-02 00:00:00 | 58 | 115763 | 81.240875 | 136286.890625 |
| 2016-02-02 00:05:00 | 123 | 332189 | 106.497734 | 405408.000000 |
| 2016-02-02 00:10:00 | 62 | 148478 | 67.060234 | 163492.187500 |
| 2016-02-02 00:15:00 | 49 | 198810 | 66.726730 | 151165.390625 |
| 2016-02-02 00:20:00 | 60 | 0 | 66.162521 | 140813.171875 |

## VI. Performance on CPU Utilization

Fig. 8 and Fig. 9 have illustrated the actual and predicted usage trends of CPU based on the date time through visualization insights. The range obtained in the LSTM CPU usage is from 45 to 125MHZ.

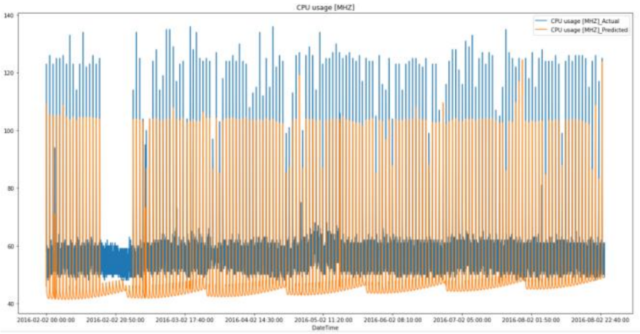

Fig. 8. CPU usage of Actual vs Predicted for LSTM Technique.

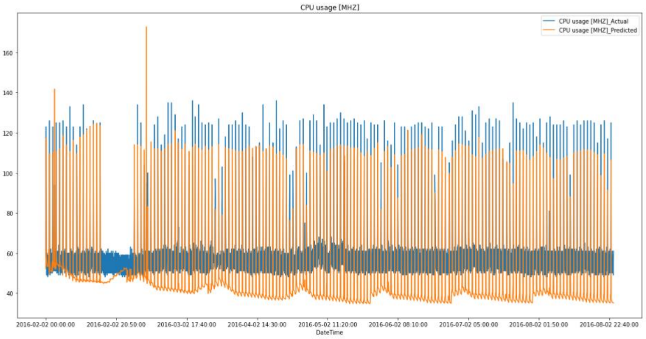

Fig. 9. CPU usage of Actual vs Predicted for GRU Technique.

The maximum CPU resource utilization of actual and predicted for LSTM at 2016-02-02 00:05:00 are 123 MHZ and 109.12 MHZ respectively but in the case of GRU, the CPU utilization at the same datetime is 123 MHZ in actual and 117.18 MHZ in prediction. When comparing the predicted CPU usage of LSTM shows that it is lesser resource utilization than GRU. Therefore, the instance provisioning of LSTM is less while compared to GRU method.

## VII. Performance on Memory Usage

Fig. 10 and Fig. 11 have illustrated the actual and predicted usage of memory based on the datetime. The range obtained in the CPU usage is from 45 to 125MHZ. The maximum memory utilization of actual and predicted for LSTM at 2016-02-02 00:05:00 are 332.19 MB and 422.45 MB respectively but in the case of GRU, the memory utilization at the same datetime is 332.19 MB in actual and 398.35 MB in prediction.

However, when comparing the predicted memory utilization of LSTM is higher in resource utilization than GRU but in the other datetime, predicted memory utilization of LSTM is very less than GRU. Therefore, the instance provisioning of LSTM is less as well as avoiding traffic while compared to GRU method.

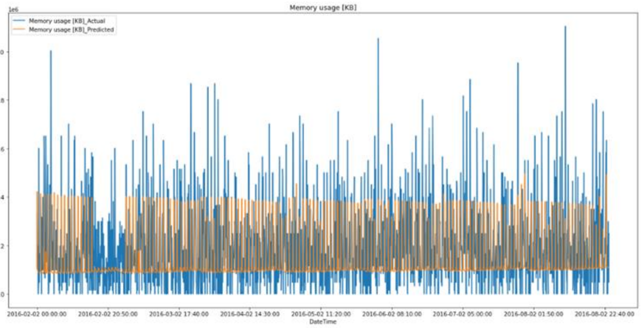

Fig. 10. Memory usage of Actual vs Predicted for LSTM Technique.

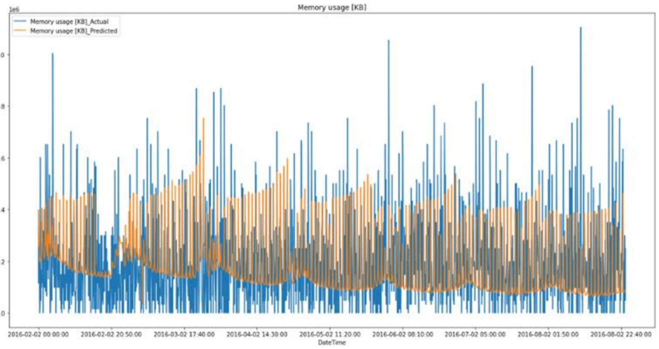

Fig. 11. Memory usage of Actual vs Predicted for GRU Technique.

Moreover, the error rate calculation is done through RMSE value and it is a standard procedure to measure the error of the model in forecasting the quantitative data. It can be measured with the given formula shown in the Fig. 12.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Fig. 12. RMSE SCORE.

The proposed predictive scaling in LSTM technique is 0.0081 whereas in the case of GRU is 0.0095 that affect the accuracy of the predictive scaling of GRU while comparing to LSTM. Thus, the accuracy of predictive scaling in LSTM is higher may assist in reducing the usage of instance provisioning by minimizing the energy consumption and reduction in bill cost.

## VIII. Conclusion

The essential factor required for current situation is green computing which provides the environment with beneficial computing power that believes on energy efficient computing. This paper has presented few basic concepts of predictive auto-scaling with EDA and LSTM algorithm. The working principle of tanh and sigmoid activation function as input has assisted to predict and forecast the predictive scale in and scale out process exactly and accurately. The pitfalls performance has followed the predictive nature to the web application in public cloud. Moreover, the avoidance of these pitfalls can be done through predictive auto-scaling by LSTM technique and validated in a real-time environment. Hence, the result of CPU usage and memory utilization performances defines the performance of ReLu layer and sequential layer of LSTM technique in cloud computing application. Thus, the comparison of LSTM method is compared with GRU technique for evaluating the performance of CPU usage and memory utilization. The evaluation results determined that CPU utilization and memory usage of LSTM is lesser than GRU method and RMSE value of LSTM is 0.0081 but in GRU is 0.0095 illustrated that error rate is higher in GRU. Moreover, the utilization of predictive scaling method has illustrated an improved performance in terms of energy efficiency by determining the better and reduced utilization of CPU as well as less memory usage while compared to GRU in large data set. This presented innovative idea for Dev-Ops users for unprecedented and cyclic workloads with green cloud brokers through predictive auto-scaling by LSTM has provided the CDC to reduce power consumption without violating QOS and SLA.

## References

[1] N. Serrano, G. Gallardo, and J. Hernantes, Infrastructure as a service and cloud technologies. IEEE Software, 32(2):30–36, Mar 2015.

[2] D. Moldovan, H. L. Truong, and S. Dustdar, Cost-awarescalability of applications in public clouds. In 2016 IEEE InternationalConference on Cloud Engineering (IC2E), pages79–88, April 2016.

[3] D. Jayasinghe, S. Malkowski, J. Li, Q. Wang, Z. Wang,and C. Pu. Variations in performance and scalability: Anexperimental study in iaas clouds using multi-tier workloads.IEEE Transactions on Services Computing, 7(2):293–306,April 2014.

[4] Liu, Jinwei, HaiyingShen, and Liuhua Chen, "CORP: Cooperative opportunistic resource provisioning for short-lived jobs in cloud systems," In 2016 IEEE International Conference on Cluster Computing (CLUSTER), pp. 90-99, IEEE, 2016.

[5] Bi, Jing, Libo Zhang, Haitao Yuan, and MengChu Zhou. "Hybrid task prediction based on wavelet decomposition and ARIMA model in cloud data center." In 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), pp. 1-6, IEEE, 2018.

[6] Amiri, Maryam, and Leyli Mohammad-Khanli, "Survey on prediction models of applications for resources provisioning in cloud," Journal of Network and Computer Applications 82 (2017): pp. 93-113.

[7] Yu, Yongjia, Vasu Jindal, FarokhBastani, Fang Li, and I-Ling Yen, "Improving the smartness of cloud management via machine learning based workload prediction," IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol. 2, pp. 38-44, IEEE, 2018.

[8] Kaur, Gurleen, AnjuBala, and Inderveer Chana. "An intelligent regressive ensemble approach for predicting resource usage in cloud computing." Journal of Parallel and Distributed Computing 123 (2019): pp. 1-12.

[9] Borkowski, Michael, Stefan Schulte, and ChristophHochreiner, "Predicting cloud resource utilization," In Proceedings of the 9th International Conference on Utility and Cloud Computing, pp. 37-42. 2016.

[10] Anupama, K. C., R. Nagaraja, and M. Jaiganesh, "A Perspective view of Resource-based Capacity planning in Cloud computing," 1st International Conference on Advances in Information Technology (ICAIT), pp. 358-363, IEEE, 2019.

[11] K. B. Bey, F. Benhammadi, A. Mokhtari, Z. Guessoum, “CPU load predictionmodel for distributed computing," in Proc. 8th Int. Symp. ParallelDistrib. Comput., Jun. 2009, pp. 39-45.

[12] K. Mason, M. Duggan, E. Barrett, J. Duggan, and E. Howley, ``Predictinghost CPU utilization in the cloud using evolutionary neural networks," Future Gener. Comput. Syst., vol. 86, pp. 162_173, Sep. 2018.

[13] Z. Ullah, S. H. Qazi, and G. M. Khan, “Adaptive resource utilization prediction system for infrastructure as a service cloud," Comput. Intell. Neurosci., vol. 2017, Jul. 2017.

[14] S. Ismaeel and A. Miri ``Multivariate time series ELM for cloud data centreworkload prediction," in Proc. Int. Conf. Hum.-Comput. Interact. Cham,Switzerland: Springer, Jul. 2016, pp. 565-576.

[15] Bi, Jing, Libo Zhang, Haitao Yuan, and MengChu Zhou. "Hybrid task prediction based on wavelet decomposition and ARIMA model in cloud data center." In 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), pp. 1-6. IEEE, 2018.

[16] Janardhanan, Deepak, and Enda Barrett. "CPU workload forecasting of machines in data centers using LSTM recurrent neural networks and ARIMA models." In 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST), pp. 55-60. IEEE, 2017.

[17] Cetinski, Katja, and Matjaz B. Juric. "AME-WPC: Advanced model for efficient workload prediction in the cloud." Journal of Network and Computer Applications 55 (2015): pp. 191-201.

[18] M. U. Farooq, A. Shakoor, and A. B. Siddique, ―An Efficient Dynamic Round Robin Algorithm for CPU scheduling,‖ no. March, 2017.

[19] B. Dave, P. S. Yadev, M. Mathuria, and Y. M. Sharma, ―Optimize Task Scheduling Act By Modified Round Robin Scheme with Vigorous Time Quantum,‖ 2017 Int. Conf. Intell. Sustain. Syst., no. Iciss, pp. 905–910, 2017.

[20] D. Khokhar and A. Kaushik, ―BEST TIME QUANTUM ROUND ROBIN CPU,‖ no. 5, pp. 3–7, 2017.

[21] Priyanka Singh, PalakBaaga&Saurabh Gupta (2016), Assorted Load Balancing Algorithm in Cloud Computing: A Survey. International Journal of Computer Applications. 143-No 7, June 2016. pp.34-40.

[22] Asha, M. L. &NeethuMyshri,R. (2014), Performance Evaluation of Round Robin Algorithm in Cloud Environment. International Journal of Computer Applications ( pp.12-16 ). ICICT-2014.

[23] Yan, M.; Liang, X.; Lu, Z.;Wu, J.; Zhang,W. HANSEL: Adaptive horizontal scaling of microservices using Bi-LSTM. Appl. SoftComput. 2021, 105, 107216.

[24] Imdoukh, M.; Ahmad, I.; Alfailakawi, M.G. Machine learning-based auto-scaling for containerized applications. Neural Comput.Appl. 2019, 32, 9745–9760.

[25] Tang, X.; Liu, Q.; Dong, Y.; Han, J.; Zhang, Z. Fisher: An Efficient Container Load Prediction Model with Deep Neural Networkin Clouds. In Proceedings of the 2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous ComputingCommunications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications(ISPA/IUCC/BDCloud/SocialCom/SustainCom), Melbourne, Australia, 11–13 December 2018; pp. 199–206.

[26] Mahmoud Imdoukh, Imtiaz Ahmad, Mohammad Gh. Alfailakawi: Machine Learning –based auto-scaling for containerized applications 2019, Neural Computing and Applications, springer, https://doi.org/10.1007/s00521-019-04507-z.

[27] http://gwa.ewi.tudelft.nl/datasets/gwa-t-13-materna.

[28] Anupama KC, Shivakumar BR, Nagaraja R, “Resource Utilization Prediction in Cloud Computing using Hybrid Model”, International Journal of Advanced Computer Science and Applications (IJACSA), vol.12,No.4,2021.