International Journal of Cloud Computing and Database Management

E-ISSN: 2707-5915
P-ISSN: 2707-5907
IJCCDM 2020; 1(2): 17-21
Received: 07-02-2020
Accepted: 10-03-2020

N Subalakshmi
Assistant Professor, Computer Science, Annamalai University Annamalai Nagar, Tamil Nadu, India.

M Jeyakarthic
Assistant Director (Academic), Tamil Virtual University, Chennai. Tamil Nadu, India.

Corresponding Author:
N Subalakshmi
Assistant Professor, Computer Science, Annamalai University Annamalai Nagar, Tamil Nadu, India

Cloud resources analysis and prediction system using Deep Neural network

N Subalakshmi and M Jeyakarthic

DOI: https://doi.org/10.33545/27075907.2020.v1.i2a.15

Abstract
Deep Neural Network classifier is one of the Deep Learning models for categorizing the exactness of systematic scaling orders in the groupings as an Administration (IaaS) layer of cloud computing. The hypothesis in this research is that calculation precision of scaling orders can be improved by demonstrating a reasonable time-arrangement expectation calculation dependent on the presentation plan after some time. In the examination, outstanding burden was considered as the exhibition metric and Deep Neural Network (DNN) were utilized as time-arrangement expectation procedures. The aftereffects of the trial demonstrate that expectation exactness of DNN relies upon there mining task at hand plan of the framework under learning. Precisely, the outcomes demonstrate that DNN has better forecast exactness in the situations with occasional and expanding remaining task at hand plans, while DNN in predicting unexpected workload design. Accurately, this paper proposed a design for a self-versatile expectation suite utilizing an autonomic framework technique. This suite can indicate the maximum appropriate prediction technique based on the performance design, which leads to more exact prediction outcomes.

Keywords: Deep neural network, Support Vector Machine, Neural Networks, Workload pattern, Resource provisioning

1. Introduction
Deep learning is a developing feasible structure model and in the previous time its use has expanded a great deal of acceptance. The National Foundation of Standard and Innovation (NIST) expressed the imperative appearances of distributed computing, as: On-request asset pooling, estimated administrations, fast flexibility, wide system access and self-administration [1]. Versatility illustrative of distributed computing allows clients to obtain and discharge properties on interest, which diminishes their rate by making them pay for the assets they basically have utilized [2, 30]. These administrations can be made accessible to (open cloud), restricted for private use (private cloud), or be presented on a crossover cloud which is a mix of both private and open mists [3]. This paper tended to the IaaS layer of open distributed computing situations. Determining the accurate total of resources for a cloud computing environment is a double-edged sword, which might lead to either over-provisioning or under-provisioning [5, 31]. Over-provisioning and under-provisioning and are results of, correspondingly, immersion or misuse of assets, and are among the most significant difficulties cloud customers are tested with. One technique to overpowering these difficulties is to utilize a scaling framework. Scaling framework comprehends the all-out presentation exchange off by over and again changing application assets dependent on its outstanding task at hand.

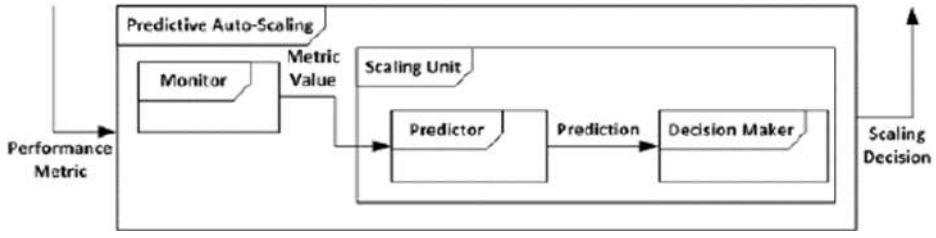

Fig 1: Architectural overview of a predictive auto-scaling system

As exposed in Fig. 1, Monitor, Predictor, and Decision are the primary instruments of a prescient scaling framework. To catch late execution of distributed computing condition, scaling frameworks screen at least one execution metric (s). In this work we thought about outstanding task at hand as the presentation metric. As delineated in Fig. 1, Indicator utilizes execution metric's present an incentive from Monitor to estimate upcoming execution metric worth.

Authors in [5, 7] report that there are three trademark remaining burden designs in distributed computing situations, with each speaks to a commonplace application or situation. These examples are: unpredicted, occasional and developing.

In the previously mentioned experimentation, sliding window procedure was utilized to prepare forecast calculations. In this procedure, window size is one the most significant elements that significantly affects the forecast exactness. Along these lines, to expand prediction results, we have dissected the impact of window size on expectation exactness as a part of our test.

As per [6, 32], Profound Neural Systems (DNN), Neural Systems (NN) and Bolster Vector Machine (SVM) are the best expectation calculations to anticipate future framework attributes. Hence, in this work we utilized DNN, NN and SVM calculations as the Forecast segment. The principle commitments of this work

- Contrasting DNN and NN and SVM expectation exactness with respect to the distinctive remaining task at hand examples.
- Dissecting the effect of sliding window size on DNN with NN and SVM expectation exactness.
- Recommending an abnormal state plan of a self-versatile forecast suite which picks the most reasonable expectation calculation dependent on the approaching remaining task at hand example.

The rest of the examination is arranged as seeks after: Area II talks about the foundation and related work. This is trailed by investigation of DNN in segment III. Area IV is devoted to the examinations and outcomes investigation. Area V end and potential headings for the future research are talked about.

2. Background and Related Work

In this segment basic perceptions used in the paper and correlated work are presented briefly. Section A is an outline of workload concept and its designs. In section B, TPC-W and Amazon EC2 are accessible, and finally, sections C and D provide an overview of the most primary auto-scaling methods in two broad categories: prediction techniques and decision making, respectively.

2.1 Workload

Resource allocation for batch applications is frequently mentioned to as progress which includes meeting a certain job performance deadline [4]. Scheduling has been widely studied in grid situations [4] and also discovered in cloud environments, and it is separate of the possibility of this paper. Similarly, applications with unchanging (or static) workload design do not require a scaling system for resource allocation. Therefore, in this paper we only focus on applications with the following workload patterns

- **Periodic workload:** Represents workloads with seasonal changes. This class of workload covers cyclic/bursting workload of [4] and periodic and once in-a-lifetime workloads of [9].
- **Unpredicted workload:** Represents fluctuating workloads. This class of workload covers unpredicted workload of [9].
- **Growing workload:** Represents workloads with increasing trend. This class of workload covers growing workload of [4] and continuously changing workload of [9].

2.2 Amazon EC2and TPC-W

So as to create previously mentioned remaining task at hand examples, one can utilize either genuine follow records or request standards. A Complete outline of suggestion records and application benchmarks is accounted for in [4]. Essentially, benchmarks incorporate a web application together with an outstanding burden generator that makes session-based solicitations to the application under test. Some normally utilized benchmarks for cloud research are [4]: Rubies, TPC-W, Cloud Stone, and Wiki Bench. In this paper, we have utilized Java usage of TPC-W, because of its straightforwardness and broad online documents. Analysts still use TPC-W to lead auto-scaling tests [6, 17, 28].

2.3 Decision making techniques

The authors in [4, 30] gathering existing scaling approaches into five classifications: lining hypothesis, time-arrangement examination, support learning, control hypothesis, and edge based strategies. Among these classifications, time-arrangement examination centers around the forecast side of the asset provisioning task and isn't a "basic leadership" strategy. Conversely, the edge based strategy is an unadulterated basic leadership system while the remainder of the auto-scaling classifications (control hypothesis, lining hypothesis, and fortification adapting) by one way or another play the Predicator and Decision Maker jobs simultaneously.

2.4 Prediction techniques

The authors in [16] have checked NN and Straight Relapse calculations to anticipate the future estimation of CPU burden and they have presumed that NN outperforms Direct Relapse as far as exactness. Furthermore, they have demonstrated that precision of the two calculations relies upon the information window size. Then again, the creators in [5] have assessed distinctive AI expectation results. The creators have considered SVM, NN and Direct Relapse. They have checked the forecast aftereffects of these calculations utilizing three execution measurements: CPU usage, throughput, and reaction time.

In this paper we utilized SVM and NN strategies for the expectation task. These techniques are two of the most precise AI calculations in the scaling field [5] and have been generally utilized in other designing fields. In section 3 measured basics of these algorithms (i.e., SVM and NN) and their particular arrangement in our trial is displayed.

3. Deep neural network

Deep learning (DL) is a subfield of Machine Learning dependent on learning various levels of representing by making an order of structures where the advanced stages are characterized from the lower levels and a similar lower level highlights can support in characterizing numerous higher level features [11]. Deep Learning structure expands the

neural network (NN) by adding progressively shrouded layers to the system design between the information and yield layers to show increasingly unpredictable and nonlinear relations. This perception enhanced the investigator' consideration in the recent years for its good performance to become the best solution in many problems in medical image analysis applications such as classification, segmentation, registration and image demising [7, 10-13]. Deep neural network (DNN) is another Deep Learning construction that is broadly utilized for order or relapse with achievement in numerous regions.

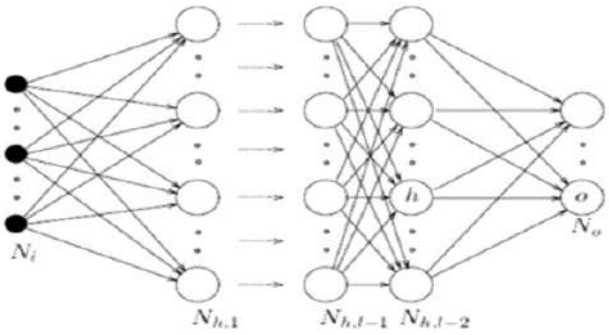

Fig 2: DNN architecture.

Input flows from the input layer to the output layer through number of hidden layers which are more than two layers [13]. Fig. 1 illustrates the typical architecture for DNNs where Ni is the input layer contains of neurons for the input features, No is the output layer contains neurons for the output classes and Nh, l are the hidden layers.

In our experiment we utilized profound neural system calculation characterized in the WEKA apparatus, which is called Multilayer Perceptron. The parameters we utilized in our test are appeared in Table I. Parameter determination is characteristic call originated on heuristics, as there is no technical equation or hypothesis that has been proposed to indicate the best parameters. We have chosen Multilayer Perceptron parameters dependent on the best outcomes after a few preliminaries.

Table 1: Parameters Settings for Deep Nerual Network

Parameter Name	Value
Hidden Layer Sizes	50
Epochs	10
Epsilon	1.0E-8
rho	0.99

4. Experiment and Results

The authoritative objective of this test is to improve forecast exactness of prescient auto-scaling framework. DNN are the most precise machine learning calculations [5] that can be utilized for remaining task at workload expectation. In this analysis we meant to investigate relations between various workload patterns and prediction accuracy of DNN.

4.1 Experimental Environment

To organize the examination condition, we passed on Java execution of Amazon EC2 benchmark on TPC-W structure. Fig. 3 represents to compositional layout of our preliminary course of action. As showed up in Fig. 3, the test course of action includes three virtual machines running on Ubuntu Linux for the client (RBE), web server and database, independently. Table II presents detail of these virtual machines. Note that to lessen test unconventionality we simply observed execution of the web server level in this paper and acknowledged that record isn't a bottleneck. Therefore, an about astounding virtual machine is supposed to be committed to the database level.

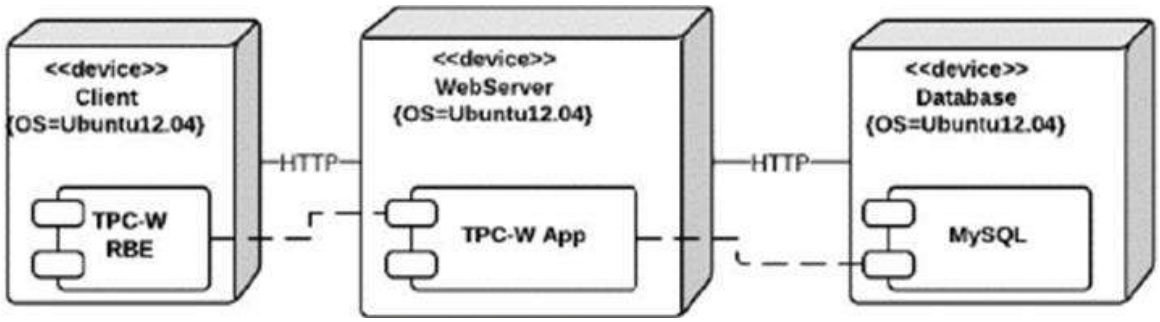

Fig 3: Architectural overview of experiment

Table 2: Hardware Specification of Servers for Experiment

	Memory	Processor	Storage
Client	1GB	4 core	8 GB
Webserver	1GB	4 core	8 GB
Storage	2GB	8 core	20 GB

4.2 Proposed methods of DNN

After producing definite assignments, the precision of DNN is anticipated in "periodic" remaining task at hand examples DNN are test of directed learning class of deep learning procedures. At that point, the made model is assessed against the testing dataset.

Another significant factor in SVM and NN is information highlights. In deep learning an element is an individual quantifiable property of a marvel being watched. Subsequently, in this analysis – notwithstanding our principle objective (for example impact of various outstanding task at hand examples on SVM and NN forecast precision) – we examined impact of window size on the expectation exactness of DNN, SVM and NN, as well.

4.3 Evaluation metrics

Exactness of the outcomes can be assessed dependent on various measurements, for example, Mean Absolute Percentage Error (MAPE), PRED (25) and R2 Prediction Accuracy [21]. Also, R2 Prediction Accuracy is a proportion of integrity of-fit, which it's worth falls inside the range [0, 1] and is normally connected to straight relapse models [16]. Because of the impediments of PRED (25) and R2

Prediction Accuracy, we utilized MAPE measurements in this work. Formal meanings of these measurements are:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|YP_i - Y_i|}{Y_i}$$

Where YPi is the anticipated yield and Yi is the real yield for I-th perception, and n is the quantity of perceptions for which the forecast is made. MAPE typically communicates exactness as a rate and is a famous measurement in insights, particularly in pattern estimation. Smaller MAPE qualities show a progressively scheme.

4.4 Results

Tables 3 represent the accuracy of DNN in predicting "periodic" workload pattern.

Table 3: MAPE Values for Existing and Proposed Methods

Window Size	MAPE Proposed DNN	MAPE NN	MAPE SVM
2	2.1902	4.9108	2.7604
3	2.1890	5.1113	2.7295
4	2.1876	4.8585	2.7356
5	2.1865	4.8553	2.7391
6	2.1843	4.5919	2.7319
7	2.1790	4.6787	2.7389
8	2.1787	4.3998	2.7339
9	2.1760	4.4980	2.6765
10	2.1589	3.3097	2.6742

Fig.4compare MAPE values for proposed DNN and Existing NN and SVM in the workload patterns. We have only presented MAPE values of DNN, NN and SVM in the workload patterns.

In addition, based on Fig.4, for small window sizes NN has somewhat improved prediction accurateness associated to SVM and proposed DNN (for window size = 2, MAPE NN value is 4.91, which is slightly less than MAPE SVM value that is 2.76 and MAPE proposed DNN value that is 2.19). Our results show that for environments with "periodic workload pattern" DNN outperforms SVM and NN. Moreover, increasing window size does not improve prediction accuracy for this workload pattern.

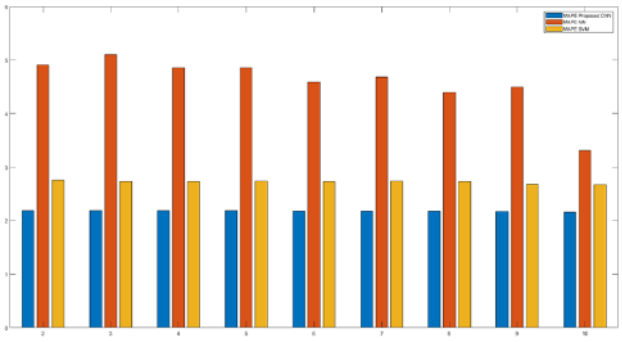

Fig 4: MAPE Values for Existing and Proposed Methods

5. Conclusions and future research

In this paper, we have proposed and demonstrated a hypothesis on expectation exactness of Deep Neural Network System (DNN) to group the prescient scaling frameworks for the IaaS layer of distributed computing. As per the hypothesis, expectation precision of prescient scaling frameworks can be improved by showing a suitable time-arrangement forecast calculation dependent on approaching incoming workload pattern. In the investigation, the impact of remaining burden designs on forecast precision of DNN was contemplated by utilizing MAPE factors as exactness appraisal criteria. Our discoveries demonstrated that in the situations with "developing" or "occasional" remaining task at hand examples DNN has better expectation exactness contrasted with SVM and NN, while in the conditions with "unpredicted" outstanding task at hand examples DNN beats NN and SVM. Our outcomes likewise demonstrated that expanding the window size just has sway on the situations with "unpredicted" remaining task at hand example with increment in forecast precision of DNN. Be that as it may, in different situations (i.e., developing or intermittent remaining burden designs), expanding the window size does not improve the DNN exactness. A further future work will be to contemplate the effect of the database layer and dormancy on the forecast and basic leadership exactness of the various calculations dependent on outstanding burden designs just as the distinctive window sizes.

6. References

1. Mell P, Grance T. The NIST definition of cloud computing, NIST special publication, 2011, 800-145.
2. Nikravesh AY, Ajila SA, Lung CH. Cloud resource autoscaling system based on Hidden Markov Model (HMM), Proc. of the 8th IEEE International Conference on Semantic Computing, 2014.
3. Bankole AA. Cloud client prediction models for cloud resource provisioning in a multitier web application environment, Master of applied science thesis, electrical and computer engineering Department, Carleton University, 2013.
4. Lorido-Botran T, Miguel-Alonso J, Lozano JA, A review of auto-scaling techniques for elastic applications in cloud environments, Journal of Grid Computing. 2014; 12:4.
5. Nikravesh AY, Ajila SA, Lung CH. Measuring prediction sensitivity of a cloud auto-scaling system, Proc. of the 7th IEEE International workshop on service science and systems, in collaboration with the 38th International Computers, Software & Applications Conference, 2014.
6. Ajila SA, Bankole AA. Cloud client prediction models using machine learning techniques, Proc. of the IEEE 37th Computer Software and Application Conference, 2013.
7. Workload patterns for cloud computing, [Online], Available: http://watdenkt.veenhof.nu/2010/07/13/workloadpatterns-for-cloudcomputing/, 2010.
8. Williams JR, Burton FR, Paige RF, Polak FC. Sensitivity analysis in model-driven engineering," Proc. Of the 15th International Conference on Model Driven Engineering Languages and Systems, 2012.
9. HW Cain, R Rajwar, M Marden, MH Lipasti. "An architectural evaluation of Java TPC-W," Proc. of the 7th International Symposium on high performance computer architecture, 2001.
10. Fehling C, Leymann F, Retter R, Schupeck W, Arbitter P. Cloud Computing Patterns: Fundamentals to Design, Build, and manage cloud applications, Springer, 2014.

11. Mohsen H, El-Dahshan ESA, El-Horbaty ESM, Salem ABM. Classification using deep learning neural networks for brain tumors. Future Computing and Informatics Journal. 2018; 3(1):68-71.
12. Amazon Elastic Compute Cloud (Amazon EC2), [Online], Available: http://aws.amazon.com/ec2, 2013.
13. Rack Space, the Open Cloud Company, [Online], Available: http://rackspace.com, 2012.
14. Jhade Sunil, SS Patil. Efficient scheme of classifier on land use pattern. Int J Stat Appl Math 2019;4(2):38-43.
15. Hasan MZ, Magana E, Clemm A, Tucker L, Gudreddi SLD. Integrated and autonomic cloud resource scaling, Proc. of the network operations and management Symposium, 2012.
16. Kupferman J, Silverman J, Jara P, Browne J. Scaling into the cloud, Technical Report, Computer Science Department, University of California, Santa Barbara, 2009.
17. Roy N, Dubey A, Gokhale A. Efficient autoscaling in the cloud using predictive models for workload forecasting, Proc. of the 4th IEEE International Conference on Cloud Computing, 2011.
18. Islam S, Keung J, Lee K, Liu A. Empirical prediction models for adaptive resource provisioning in the cloud", Future generation computer systems. 2012; 28:155-165.
19. Nicholas I, Sapankevych R Sankar. Time Series Prediction using support vector machines: A Survey, IEEE Computational Intelligence Magazine. 2009; 4:2.
20. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten H *et al*. The WEKA data mining software: an update, SIGKDD Explorations. 2009; 11:1.
21. Trevor H, Tibshirani R, Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, New York: Springer, February, 2009.
22. Arlot S, Celisse A. A survey of cross-validation procedures for model selection", Journal of Statistics Surveys. 2010; 41:40-79.
23. Witten I, Frank E. Data mining practical machine learning tools and techniques with Java implementations," Academic Press, San Diego, 2000.
24. Garlan D, Schmerl B. Model-based adaptation for self-healing systems, Proc. of the 1st Workshop on Self-Healing Systems, 2002.
25. Sterritt R, Smyth B, Bradley M. PACT: personal autonomic computing tools", Proc. of the 12th IEEE International conference and workshops on the engineering of computer based systems, 2005, 519-527.
26. Bigus JP, Schlosnagle DA, Pilgrim JR, Mills III WN, Diao Y. ABLE: a toolkit for building multiagent autonomic systems IBM Syst Journal. 2002; 41(3):350–371,
27. Littman ML, Ravi N, Fenson E, Howard R. Reinforcement learning for autonomic network repair, Proc. of the 1st International conference on autonomic Computing, 2004, 284-285.
28. Dowling J, Curran E, Cunningham R, Cahill V, Building autonomic systems using collaborative reinforcement learning, Knowledge Eng Rev. 2006; 21:231–238.
29. Hwang K, Bai X, Shi Y, Li M, Chen W, Wu Y. Cloud Performance Modeling and Benchmark Evaluation of Elastic Scaling Strategies", IEEE transactions on parallel and distributed systems, 2015, 99.
30. Jeyakarthic M, Subalaskhmi N. Soft Computing Approaches with Neural Networks – An Evaluation, Journal of advanced research in dynamical & control systems (JARDCS), ISSN: 1943–023X. 2018; 10(14):1001-1005.
31. Thirumalairaj A, Jeyakarthic M. An Intelligent Feature Selection with Optimal Neural Network Based Network Intrusion detection system for cloud environment, International journal of engineering and advanced technology (IJEAT), ISSN: 2249–8958, 2020; 9(3):3560-3569.
32. Jeyakarthic M, Venkatesh S. An Effective customer churn prediction model using adaptive gain with back propagation neural network in cloud computing environment, The Journal of Research on the Lepidoptera, ISSN: 0022-4324 (print), ISSN: 2156-5457 (online).
DOI: 10.36872/LEPI/V51I1/301034, 2020; 51(1):386-399.
33. Right Scale Cloud management, [Online], Available: http://rightscale.com, 2012.