The prediction dashboard on cluster data with comparative model for failed jobs

Neha V[1], Anupama K C[2], Dr R Nagaraja[3]

[1]M.Tech Student, ISE, Bangalore Institute of Technology-560004
[2]Assistant Professor, ISE, Bangalore Institute of Technology-560004
[3]Professor, PG and Research Coordinator ISE, Bangalore Institute of Technology

Abstract— **Always failure jobs predictions with respect to existing and non-existing classification in service oriented architecture of cloud is always a big challenge. So we introduce a new 2 comparative approaches for predictions with the dataset which collected from Google clusters. The approach named as 2 level comparator (neural association LSTM, probabilistic regression).The goal of the work is to present an application failure prediction model that exactly predicts weather a task or job is successfully finished or failed. Considering the prediction on the traces of Google cluster data, there is a significant consumption of resource due to killed of failed jobs. For this we propose machine learning algorithm called** LSTM (Long-Short- Term-Memory) **network and by serializing the input we are comparing with** PRM (probabilistic Regression Model) **for the accuracy. Prediction takes the priority and task attempts attributes of Google cluster data in order to predict the termination status (e.g., failed or finished). (keywords: long short term memory, serialization , regression prediction, failure jobs)**

I.INTRODUCTION

Failure predictions on the preprocessing and non-preprocessing dataset are complicated as the data contains so many inter dependent and non inter dependent attributes with proper weightages. These weightages will be treated as priorities as these priorities will play major roles in the prediction of failure cases. Generally association algorithms[1] will take lengthy processing time as features with respect to association of the selected seed attributed values , association or collaboration[2] so that more combinations without filtration will frame with respect to priority and task attempts as these are the core attributes for the entire data which were chosen for LSTM technique[3][4] as chosen one and this technique is to frame the threshold and based on the threshold or maximum and minimum so that the user will be selected as the priority based as well as threshold based so that users will be scattered among the neural combinations so that the all the second level of the filtered fine grained as first level[4] will come as the input to next level. So this kind of rules based will allow the frameworks to go to predict the failure of success rates. Online failure prediction based on runtime monitoring is a popular research area. There has been a variety of models and methods that use the current state of a system and, frequently, past experience as well, for example the work by Salfner et al.

Prior failure diagnosis and prediction have been studied in supercomputers and cloud clusters [5]. Liang et al. [6] use tagged logs from the BlueGene machine to discover failures recurrences and correlations between fatal and non-fatal events, and thus predict failures. Using workload traces from The Grid Workload Archive project, Fadishei [7] et al. find correlations between job failures and attributes including CPU intensity, memory usage, CPU utilization, queue utilization, exit hour and migration of jobs. Pan et al. [8] use the differences in the behavior of faulty and normal nodes in a Logical regression model to identify failures. However, problems arise when nodes are heterogeneous or few similar nodes can be treated as references. Williams et al. [9] empirically analyze the fault-free and faulty performance data from a replicated middleware-based system, and find that unstable performance is a precursor of failures. They build a black-box method, and predict failure in a window ahead of impending crash failures. In summary, these works predict system failures, or are confined to particular classes of jobs. In contrast, our work is the first to predict application failures in a diverse workload in the cloud.

Following are the major contributions of the work

i. Failure characterization study is made on the Google cluster data in order to understand the failure responses for the failure, attributes dependency for the failure/ success rate.

ii. Presented a Long-Short-Term-Memory method which is a neural network to detect the failure of jobs/tasks with respect to Google cluster trace in the cloud which we got the results above 85% true positive rate

iii. Serialization technique is used in order to present the same input to the next algorithm for the 2nd level prediction.

iv. Deserialization of the classes is carried out in order to pull the inputs and then presented a probabilistic regression model for the prediction of failure rate by displaying the accuracy and

v. Once the results are obtained the comparison is made between the method used in level one and level two for the results. LSTM achieved highest precision as a result.

The paper organization as follows. Presented some related work on failure prediction and analysis in Section 2. Provided an overview of Google cluster dataset with a brief characterization study in the and the background of the cluster traces and related attributes description followed by preliminaries, proposed model and the conclusion in the order and finally the references are mentioned at the end of the paper .

II RELATED WORK

Possible approaches for failure predictions are presented from the view point of system failure namely SVM, ANN, KNN, Random Forest is popular approaches for failure predictions as these are the machine learning approaches as these selective approaches are not all the fittest models but with respect to the current input model datasets the approaches will vary

Triguero,et al. present various improvements to the well-known data mining technique k-nearest neighbor's algorithm to come up with smart data. k-NN algorithm's weaknesses - noisy data and incomplete data have been addressed using Noise filtering and correction and missing values imputation models. Also through parallelism and data reduction the kNN algorithm has become a core model to detect and correct imperfect data, eliminate noisy and redundant data, as well as correct missing values. They present several case studies that showcase k-NN algorithm as a unique model to obtain smart data from large amounts of potentially imperfect data.

Laloux, J.F., Le-Khac, N.A. and Kechadi, M.T. state that current distributed clustering approaches are predominantly generating global models by aggregating local result, hence losing important knowledge. They present a new distributed data mining approach where local models are not directly merged to build the global ones. Centralize clustering is carried out at each site (node) to build local models. These models are sent to the servers where clusters will be regenerated based on local models features. Considering how many corporations have geographically isolated data centers the authors objective is to reduce data collection expense by minimizing data communication and computational time, while getting accurate global results.

Halkidi, M. and Koutsopoulos, develop a novel approach for online distributed clustering of streaming data using belief propagation techniques. They use a two-level clustering approach to address the problem of clustering distributed streaming data. A set of data arrives at each time period, and the goal is to maintain a set of salient data at each time period, which represents the data received up to that slot. At each epoch, the individual exemplars from distributed system are sent to the central location of the system, which in turn carries a second-level clustering on them to derive a data synopsis global for the whole system. The local exemplars that pop out from the second level clustering procedure are given back to the nodes of the same model with properly changed weights which reflect back their preference in global clustering.

Fernandez, J.R. and El-Sheikh, E.M. state that with today's generation of high speed data streams traditional clustering and/or pattern recognition algorithms are inefficient for clustering data. They define data stream as a dynamic dataset that is characterized by a sequence of data records that evolves over time, has extremely fast arrival rates and is unbounded. In their paper, they present a clustering framework (CluSandra) and algorithm that, combined, address the time constraint and space challenge, and allows end-users to explore and gain knowledge from evolving data streams. They use an integration of open source products that are used to control the data stream and facilitate the harnessing of knowledge from the data stream. The authors highlight that the CluSandra algorithm exhibits the following characteristics: configurable, distributable, elastically scalable, highly available and reliable, and simpler to implement

III BACKGROUND

A. *Google Cluster Trace Overview*

The Google cluster data consisting of a trace of clusters scheduler request and responses. Each trace is consisting of various numbers of jobs submitted by different users. Each task represents a one Linux program; like-wise each job consisting of multiple tasks which is represented with number of attributes namely task hour, cpu mean time, task attempts etc. For each task trace will indicate whether the task is submitted or discarded and the processing time is also noted down if achieves success status.

Table 1: Terminologies of job and task failure

Terminology	Description
Job failure	A job is descheduled due to task failure
Task Failure	A task is descheduled due to task failure (e.g., exceptions, software bugs)

The requested job to submit can have four options usually. As shown in the figure 1 the job can either be submitted successfully, for the machine and starts with the execution and later finishes the task or the second option is that the submitted job/task can be evicted, failed or get killed. For these options the rescheduling of the job is possible which include in the task constraints of the system.

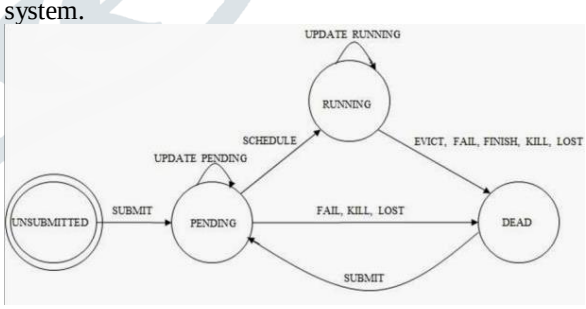

Figure 1. State transition for task and jobs in Google cluster trace

B. *Trace data analysis*

To construct the efficient failure detection system it is important to analyze the trace data set and parameter consideration effects the performance. There are multiple attributes in the system which are considered for the scheduling decision.

Task Resubmission: Once the system get failed or killed the job or task will be keep on submitting each and every time up to some limit value which is called as task attempt. Once it reaches the limit and still does not succeeded then the task or

job is considered to be failed. According to the observation of the dataset there are resubmission of jobs is more for the failed jobs then in the success cases

Resource consumption: Resource consumption is measured for a job or task is the memory consumption and the average cpu time for each job. Resource consumption is noted for both the success and the failure cases.

Scheduling constraints: task priority comes into the picture at this stage. The task priority ranges from0-10, whereas the 0 is the minimal case sometimes it can be considered as un submitted task or job and 10 will be the highest priority of the user and there are also priorities ranges in between.

Termination status: The termination status in the Google cluster data set is the job or the task may evicted, failed, killed or finished. There are only few cases that of obtaining evicted or killed. So the focus mainly on the termination statuses namely finished or failed events.

User-specific-behavior: There are more than 500 users in the trace. The dynamic user selection is chosen as the part of the project. So that the different user's behaviors can be analyzed and the termination status can be predicted. The analysis is the evident that the small number of users can highly effect the scheduling performance.

Table 2: Attributes considered from Google cluster data

Attributes	Functionality description
Users	User name in this trace represents Google engineers and services
Priority	Task priority, small integer value starts from 0 ranges differ with different release.
Task attempts	Number of times the task was run

C. *LSTM Architecture*

The traditional methods of machine learning like FNN and SVM models are not suitable for high dimensional data sets. Meanwhile the RNN fails in dealing with long-term dependencies. To overcome this issue LSTM network model proposed for the current project which is capable of making connections between hidden state and which can give solution for long term dependency problem.

Memory cell is a special feature of LSTM model which is intermediate type of storage as shown in the figure 2. Memory cell in the hidden layer is as common as ordinary node of the hidden layer in the standard RNN. Memory cell will be generally consists of input node, input gate, internal state, forget gate and finally the output gate. Input gate value can be either 1 or 0. If the value is 1 data can pass through, else flow will be cut-off.

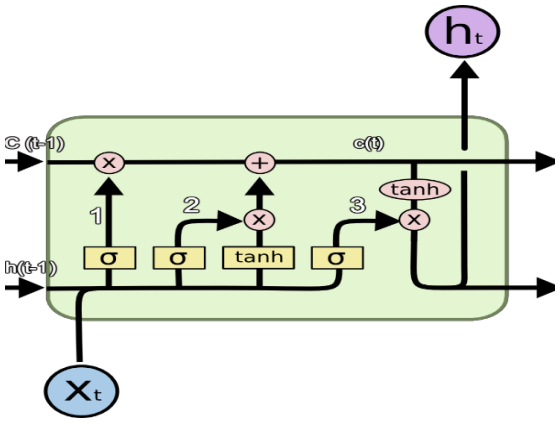

Figure 2. An LSTM Memory Cell

Internal state of LSTM is self-connected recurrent edge which is also called as constant error carousel as because the vanishing and exploding gradient problems will not the effect the error flow in this state. Forget gate is the added feature of LSTM which helps to flush the contents of the previous state called internal state which means it follows the "learning to forget" method. And the final state called output gate will provide the final value of the memory cell by multiplying with the internal state of the same memory cell.

IV.PRELIMINARIES

Association: The model which was implemented as association with neighboring attributed value which is called as neurons as hidden markov models and once the neurons framed. Once the neurons framed and based on the middle of the associations (the maximum associated combinations) where the entire association rules will be scattered for clustering. The clustering is based on the maximum first level of the attribute so that all the maximum association with neighboring comparison the user will be populated for failure only classification.

Clustering: Generally in data mining the clustering is the proper segregation of the data with respect tags, labels, similarities [3] and varieties [5]. The approach gives accurate scope to predict the out result so that all the data will in proper relative places, which can lead to flexible move for later predictions.

Classifications: The classifications are nothing but to classify the data with some label per tuple as success or failure cases [2]. These cases are to be in appropriate models with numeric and non-numerical data as the classification needs proper threshold on linear data. This linear data is in the form of inter dependent data so that the classification process to be ended up smooth.

Predictions: The predictions are always with respect to classification results. But the classifications are totally depends on preprocessing and clustering. The result to end up to view as the tuple frame is predictions over classification methods. The predictions will be with proper accuracies.

V. PROPOSED MODEL

The proposed failure prediction model consisting of multiple steps for the prediction of the termination status in order identify the failure cases prior the model gets the actual status.

The model takes the input as the attributes of the Google cluster data. Data processing step will include the pre processing steps to clean the data set and conversion from the schema.csv file to the excel sheet and get the data to the model acceptable form and then the extraction of the data to the memory is through the POI API packages to the task attributes and the resource attributes as priority, task attempts, CPU mean and the task hour as the input to the model in the attribute selection phase and then applying the LSTM/ PRM algorithms successfully by using the neural association and the threshold with respect to the LSTM model and the threshold and the likelihood calculations are made for the PRM model respectively for the calculation of the termination status. The termination status can be evicted, killed, failed or finished. Most of the cases in the cluster traces are finished and failed. So that the killed, evicted and failed jobs are considered as failed cases of the job submission and the other will be the success cases. The flow is as shown in the Figure 3.

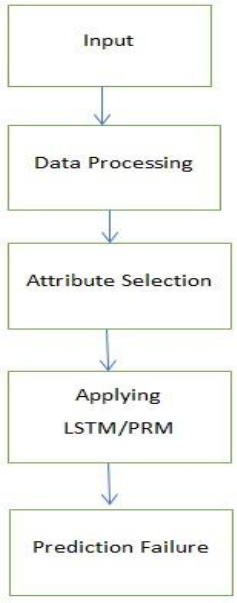

Figure 3: Proposed failure prediction model

VI.PERFORMANCE EVALUATION The implemented system and the performance evaluation using traces of google cluster data is presented in this section

A. *Experimental Setting*

The implemented project is tested in the cloudsim environment and the net beans IDE is used with the java programming language. The traces were in the CSV form and was about the 40 GB data which is condensed and tested with the 80 mb and the test cases are carried out for the same with different number of users for the prediction.

B. *Evaluation Metrics*

To know the quality of the prediction system it is necessary to specify the performance metrics. Following list of metrics are used for the evaluation of performance: Accuracy, True Positive Rate (TPR), False Positive Rate (FPR) described in Table 3

Table 3. Evaluation Metrics with respect to LSTM algorithm

Prediction metrics	Description
TPR	Success rate of the prediction
FPR	Failure rate of the prediction

C. *Job level failure Prediction*

The job level status will be considered in two cases either it finished or failed. Due to the reliability and the severity the three classes evicted, killed and failed considered as one single class.

D. *Performance comparison with PRM*

The accuracy parameter is considered for both the algorithms namely Long-Short-Term-Method and the Probabilistic Regression Method. The LSTM and the PRM comparison is as shown in the Figure 4 with 2D pi-chart representation

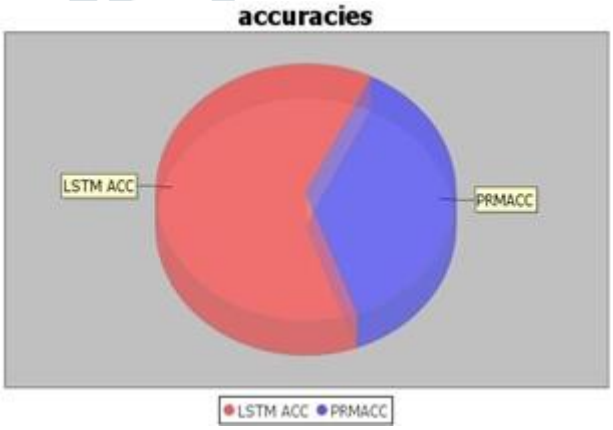

Figure 4: Accuracy prediction comparison for LSTM and PRM for the failed jobs in the cluster trace for the 4000 users.

VII. CONCLUSIONS

With consideration of large volume of Google cluster data traces presented the 2 level comparator approach which consists of LSTM and probabilistic model approach. For predicting failures via various attributes of Google cluster data trace and performance of time series is considered. We successfully predict the termination status of jobs and tasks using 2 different algorithms with descent accuracy. Experiments show the true positive rate more than 80% and false positive rate around 20% in case of LSTM with the accuracy of 90% and probabilistic model achieves accuracy with 80%. So the LSTM dominates the predictive model in terms of accuracy and the predictive model dominates the LSTM at performance time.

.

REFERENCES

[1] Y. Liang, Y. Zhang, M. Jette, A. Sivasubramaniam, and R. Sahoo, "BlueGene/Lfailure analysis and prediction models," in International Conference onDependable Systems and Networks (DSN), 2006, pp. 425 – 434.

[2] Z. Ren, X. Xu, J. Wan, W. Shi, and M. Zhou, "Workload characterization on aproduction hadoop cluster: A case study on taobao," in Workload Characterization (IISWC), International Symposium on. IEEE, 2012, pp. 3–13.

[3] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format+ schema," Google Inc., Mountain View, CA, USA, Technical Report, Nov. 2011,revised 2013.05.06.

[4] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. Kozuch,"Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in Proceedings of the Third ACM Symposium on Cloud Computing. ACM, 2012, p. 7.

[5] X. Pan, J. Tan, S. Kavulya, R. Gandhi, and P. Narasimhan, "Ganesha:Blackbox diagnosis of mapreduce systems," SIGMETRICS Perform. Eval. Rev.,vol. 37, no. 3, pp. 8–13, Jan. 2010.

[6] A. Williams, S. Pertet, and P. Narasimhan, "Tiresias: Black-box failureprediction in distributed systems," in Parallel and Distributed ProcessingSymposium. IEEE International, 2007, pp.1–8.

[7] S. Hochreiter, J. Schmidhuber, "Long short-term memory", Neural Comput., vol. 9, no. 8,pp. 1735-1780,1997.

[8] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representationsby back-propagating errors," Nature, vol. 323, pp. 533–536, 1986.

[9] M. Langkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised ¨ featurelearning and deep learning for time-series modeling," Pattern Recognition Letters,vol. 42, pp. 11–24, 2014.

[10] X. Chen, C.-D. Lu, and K. Pattabiraman, "Failure analysis of jobs incompute clouds: A google cluster case study," in the International Symposium onSoftware Reliability Engineering (ISSRE). IEEE, 2014.

[11] F. Salfner, M. Lenk, and M. Malek, "A survey of online failure prediction methods," ACM Comput. Surv., vol. 42, no. 3, pp. 10:1–10:42, Mar. 2010.

[12] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition, ser. Springerseries in statistics. Springer, 2009.

[13] J. Martens and I. Sutskever, "Learning recurrent neural networks with hessian-free optimization," in Proceedings of the 28th International Conference onMachine Learning, 2011, pp. 1033–1040.

[14] scikit-learn: Machine learning in python. [Online]. Available: http://scikit-learn.org/stable/

[15] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins,J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU mathexpression compiler," in Proceedings of the Python for Scientific ComputingConference (SciPy), Jun. 2010.

[16]. P. Rodriguez, J. Wiles, "Recurrent neural networks can learn to implement symbol-sensitive counting" in Advances in Neural Information Processing Systems, MA,Cambridge:MIT Press, vol. 10, pp. 87-93, 1998.

[17] A. Iosup, H. Li, M. Jan, S. Anoep,C. Dumitrescu, L. Wolters, and D. H. J. Epema, "The grid workloads archive,"2008.

[18] Q. Guan and S. Fu, "Adaptive anomaly identification by exploring metricsubspace in cloud computing infrastructures," in Reliable Distributed Systems(SRDS), International Symposium on. IEEE, 2013, pp. 205– 214.

[19] J. Schmidhuber, "The neural bucket brigade a local learning algorithm for dynamicfeedforward and recurrent networks", Connection Sci., vol. 1, no. 4, pp. 403-412, 1989.