ANN Based Execution Time Prediction Model and Assessment of Input Parameters through ISM

Anju Shukla, Shishir Kumar, and Harikesh Singh
Department of Computer Science and Engineering, Jaypee University of Engineering and Technology, India

Abstract: *Cloud computing is on-demand network access model which provides dynamic resource provisioning, selection and scheduling. The performance of these techniques extensively depends on the prediction of various factors e.g., task execution time, resource trust value etc., As the accuracy of prediction model absolutely depends on the input data that are fed into the network, Selection of suitable inputs also plays vital role in predicting the appropriate value. Based on predicted value, Scheduler can choose the suitable resource and perform scheduling for efficient resource utilization and reduced makespan estimates. However, precise prediction of execution time is difficult in cloud environment due to heterogeneous nature of resources and varying input data. As each task has different characteristic and execution criteria, the environment must be intelligent enough to select the suitable resource. To solve these issues, an Artificial Neural Network (ANN) based prediction model is proposed to predict the execution time of tasks. First, input parameters are identified and selected through Interpretive Structural Modeling (ISM) approach. Second, a prediction model is proposed for predicting the task execution time for varying number of inputs. Third, the proposed model is validated and provides 21.72% reduction in mean relative error compared to other state-of-the-art methods.*

Keywords: *Cloud computing, neural network, Prediction model, Resource selection.*

Received September 20, 2018; accepted January 28, 2020
https://doi.org/10.34028/iajit/17/5/1

1. Introduction

Cloud computing is on demand computing framework that provides numerous facilities including sharing the pool of highly computational resources, economic cost, storage and many more. The cloud facilities are especially available in three forms Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Cloud computing provides various tools to collaborate remote resources that are geographically distributed and eliminates the burden of procuring resources which usually consumes very long time. These resources are managed by effective scheduling and resource provisioning techniques and usually consider execution time as a major metric for selection criteria. But actual execution time is not available before task execution. So predicting precise execution time is necessary.

Another reason for predicting the execution time is that cloud computing offers on demand resource provisioning and allocation based on pay-per-use model [10]. The user uses variety of services and pay per-use basis. Therefore, accurate prediction provides reduction in cost, makespan and other Quality of Service (QoS) performance metrics. To overcome these issues, a prediction technique is required which provides the accurate execution time which is close enough to actual execution time.

The performance of any prediction model solely depends on the input data that is fed into the network. So selection of suitable input also plays important role in predicting the correct value. In this paper, various relevant input parameters are identified through literature and suitable parameters are selected through Interpretive Structural Modeling (ISM) approach. The approach is widely used in identifying the parameters which have strong predicting power. After selecting the suitable input parameters, a Artificial Neural Network (ANN) based prediction model is proposed to predict the task execution time. The proposed prediction model is a function of execution time which includes three parameters- task workload, Central Processing Unit (CPU) speed, and free time of CPU. Various methods are used in prediction models: Linear Regression [17], Support Vector Machine (SVM) [14], genetic algorithms [19] and many other optimization techniques. Here, artificial neural network is used as an optimization method which is inspired by biological neural circuits. The network is trained using back-propagation algorithm in which error is measured and feed back through the neural layers to update the weight coefficients. The GWA-T-13 workload data set [11, 20] is used for training and testing the prediction model. The major contribution of the paper is as follows:

1. Some significant contribution in the field of execution time prediction is analyzed and identified

various parameters that affect the task execution time.
2. From the identified parameters, suitable input parameters are selected for the prediction model.
3. The detailed description of neural based prediction model is presented to predict the execution time for suitable resource selection.
4. The model is simulated and validated using K-Fold Cross Validation approach.
5. The simulation results are compared with Linear Regression (LR) [17] and Chang *et al*. [3] to show the effectiveness of the proposed prediction model.

The rest of the paper is structured as follows. The major contribution in the field of execution time prediction is presented in section 2. In section 3, the methodology and evaluation criteria for predicting the model accuracy are discussed. In section 4, the proposed Multi Layer Perceptron-Artificial Neural Network (MLP-ANN) based prediction model is described. Section 5 present the simulation and validation results of proposed model over other state-of-the-arts methods. Section 5 presents conclusion followed by future directions to enhance the proposed prediction model.

2. Related Work

The study on performance prediction in grid, cloud and distributed environment has involved the concentration of the research community in the previous decades. Some significant contribution is summarized to select the appropriate mechanism to analyze in specific environment and for further optimization.

The prediction techniques can be classified in three categories-Analytical, simulation and emulation, and Empirical assessment [21]. The Analytical methods used mathematical models like machine learning, regression methods to predict the performance measures. Liu *et al*. [13] proposed a finish time prediction method for distributed tasks. The real data is collected by modifying MapReduce and regression is applied to predict the accurate finish time. Fan *et al*. [5] introduced a SVM based prediction model for optimizing the MapReduce performance. They proposed an adaptive algorithm to predict the accurate workload for a specific node. Before giving input, task is merged and re-merges for selecting specific data volume which results in increased makespan. Nemirovsky *et al*. [16] proposed machine learning approach for predicting the performance of threads for various applications. The approach provides improved throughput over traditional scheduling techniques of computer architecture. Significance importance of resource provisioning and scheduling is analyzed by Islam and Manivannan [9] and introduced a neural based model for predicting the failure in cloud environment. Accuracy, precision and recall are the major methods used for evaluation criteria for evaluating the correctness of the model.

The second category is simulation, which is used to determine the performance of the presented model while emulator is more similar to real thing and intended to work smarter. Chang *et al*. [3] presented a resource selection mechanism based on Predicted Execution Time (PET). Before submitting a task to a resource for execution, PET is calculated for each resource. Then, task is submitted to a resource which has least PET for execution. The approach provides reduction in makespan, response time and throughput in grid environment. Nadeem and Fahringer [15] introduced an optimized method for predicting execution time for workflow applications based on templates. Templates are attributes of workflow applications. It is necessary to select the correct template and evaluate them to measure the accuracy. The mechanism provides more accurate result than existing methods. However, work can be extended for simulating real world workflows. Duong *et al* [4] presented a mechanism to predict the simulation running time in cloud environment. By accurate prediction of simulation running time, cost can be effectively optimized. By effective scheduling and resource provisioning techniques, running time can be more optimized to improve the model performance. Li *et al*. [12] proposed an execution time prediction approach for effective CPU provisioning. The presented mechanism considers number of core CPU's and application workload to predict the execution time.

The third category is the empirical assessment based on interpretation taken on a real experiment. Pham *et al*. [21] presented a machine learning based execution time prediction for workflow applications. The model uses the historical data and pre run time parameters to predict the task execution time. Empirical analysis of four workflow applications shows that the model performs better than existing prediction methods but have performance issues with highly dynamic workloads in cloud environment. Islam *et al*. [8] introduced a prediction model for resource provisioning in the cloud environment. They used error correction neural network and linear regression techniques to train the model. The model is validated using cross validation technique [1] to measure the accuracy of the presented model.

The literature shows that lots of research work exists for prediction performance based on analytical, simulation and empirical techniques, but no study emphasized on the parameter selection and interrelationship between input parameters and predicted output value. Therefore, we identified the suitable parameters for prediction model through ISM [6] and proposed a neural based prediction model based on MLP which falls in the first category of prediction techniques i.e., analytical modeling.

3. Identification of Input Parameters Through ISM

The accuracy of any prediction model significantly depends on the input parameters that are fed into the network to produce the output. To solve the issue, firstly, various related parameters are identified which may affect the output of prediction model. Secondly, redundancy is found out between parameters through correlation. Thirdly, some variables are removed that have minor predictive control. Various parameters are identified from literature that are desirable for prediction model and listed in Table 1 [12].

Table 1. Parameters of a prediction model for execution time Prediction.

S. No	Parameter	Notation	Source
1	Task Workload	I_1	[3, 5, 21]
2	CPU Cores	I_2	[12]
3	CPU Capacity	I_3	[12]
4	CPU Speed	I_4	[3]
5	CPU Usage	I_5	[3]
6	Memory Usage	I_6	[3]
7	Disk Throughput	I_7	[12]
8	Network Throughput	I_8	[21]
9	Processing Element	I_9	[3]
10	Cost	I_{10}	[4]
11	Bandwidth	I_{11}	[12]
12	Resource Trust Value	I_{12}	[21]
13	Virtual Machine	I_{13}	[21]
14	Delay	I_{14}	[21]

4. Methodology

Here ISM approach is used to identify the input parameters that are fed into the prediction model for predicting task execution time.

4.1. Interpretive Structural Modeling (ISM) Approach

ISM is an interpretive approach which is used to understand the relationship between various parameters associated with the framework [6]. ISM provides a structured model for classification of various parameters based on four factors:

- Autonomous factors-These factors are considered as disengaged from the system or having no importance for the framework. Parameters having less Driving Power (DrP) and less Dependence Power (DeP) comes in this category.
- Dependent factors- These parameters don't affect others but depends on performance of other parameters. The Parameters having less DrP but high DeP comes in this category.
- Linkage factors- The Parameters having high DrP and high DeP comes in this category.
- Driving factors- The Parameters having high DrP but less DeP comes in this category.

To classify the parameters in these four categories, following are various steps that are followed in ISM approach:

1. Identify various parameters through literature that are relevant to the issue.
2. Establish the relative relationship among the parameters identified in previous step.
3. Develop the Structural Self Interaction Matrix (SSIM) for selected parameter.
4. Develop the reachability matrix and check for the transitive dependency among parameters, and convert into final reachability matrix.
5. Final reachability matrix is partitioned into different levels based on intersection of Reachability set and Antecedent set.
6. Draw the ISM model based on parameters levels and final reachability matrix.
7. Identify the cluster for each parameters based on DrP and DeP.

The detail working of each ISM step is described in the proposed section.

4.2. Execution Time Prediction Model

After selecting the appropriate inputs, a prediction model is proposed for PET of tasks. Various machine learning based prediction techniques are used in cloud computing environment LR [17], SVM [14] and ANN [2]. The neural based machine learning technique, used here, is suitable for unknown pattern of resource usage in cloud environment. We use following model and algorithm for predicting and training the model

- Multi Layer Perceptron-Artificial Neural Network (MLP-ANN) model [18].
- Back-propagation algorithm [7].

Further, the proposed prediction model is validated using K-Fold Cross Validation approach [19] to train and test the model on independent data sets. To validate the prediction model various statistical metrics are used that are described in subsequent section.

4.2.1. Multilayer Perceptron-Artificial Neural Networks (MLP-ANNs) Mode

ANNs are computing systems that are inspired by the biological neural networks that comprise animal brains. The MLP-ANNs are computational models which are used to find patterns between inputs and outputs. MLP is a class of ANN which consists of a series of layers. The MLP-ANNs are computational models which are popular means to model complex relationships between inputs and outputs and to find patterns. In supervised learning the MLP class of neural networks requires a set of training samples which are used to infer a classifier to predict a correct output value. The MLP uses various supervised learning techniques, for example back propagation, Ant Colony Optimization (ACO), Genetic Algorithm (GA) to train the MLP model. These techniques are used to train the MLP model by using various training data set to predict the correct output value.

4.2.2. Back Propagation Algorithm

Back propagation algorithm is used to train the MLP model [22]. Each input data is multiplied with weight to enter into the hidden layer. The output layer produces output through neurons of hidden layer. The output layer uses sigmoid function to calculate the activation function. At last, an error is calculated which is difference between actual and predicted output value. The measured error is propagated back to update the weight values to minimize the future error. The Error Function (*dError*) can be represented as a product of Activation Function (*Acti-fun$_i$*), Weight from Neuron i to j (*Wgt$_{ij}$*), and Sum of weighted weights (*Sum-wgt$_i$*) according to Equation (1)

$$\frac{\mathrm{dError}}{\mathrm{dWgt_{ij}}} = \frac{\mathrm{dError}}{\mathrm{dActi\_fun_i}} * \frac{\mathrm{dActi\_fun_i}}{\mathrm{dSum\_wgt_i}} * \frac{\mathrm{dSum\_wgt_i}}{\mathrm{dWgt_{ij}}} \tag{1}$$

4.3. Evaluation Criteria

We validate the accuracy of the prediction model by using various statistical metrics: Mean Percentage Error (MRE), Root Mean Square Deviation (RMSD), Mean Magnitude Relative Error (MMRE), Mean Magnitude Relative Error to Estimate (MMER), and Mean of Balanced Relative Error (MBRE).

4.3.1. Mean Relative Error (MRE)

The Mean Relative Error is calculated to predict the efficiency of proposed model. The MRE is calculated by using the Equation (2)

$$MRE = \frac{|\mathrm{AET}-\mathrm{PET}|}{\mathrm{AET}} \tag{2}$$

Where AET is the actual execution time and PET is predicted execution time of executed task.

4.3.2. Root Mean Square Deviation (RMSD)

RMSD measures the standard deviation between actual and predicted execution time. RMSD is calculated by using the Equation (3)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(PET-AET)^2}{n}} \tag{3}$$

4.3.3. Mean Magnitude Relative Error (MMRE)

MMRE is calculated by using Equation (4)

$$MMRE = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \frac{|\mathrm{AET}-\mathrm{PET}|}{\mathrm{AET}} * 100 \tag{4}$$

4.3.4. Mean Magnitude Relative Error to Estimate (MMER)

MMER is calculated by using Equation (5)

$$MMER = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \frac{|\mathrm{AET}-\mathrm{PET}|}{\mathrm{PET}} \tag{5}$$

4.3.5. Mean of Balanced Relative Error (MBRE)

MBRE is calculated by using Equation (6)

$$MBRE = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \frac{|\mathrm{AET}-\mathrm{PET}|}{\min(\mathrm{P}-\mathrm{A})} \tag{6}$$

4.4. K-Fold Cross Validation Approach

The cross validation approach is used for determine the accuracy of the trained model. In the proposed model, we applied K-fold cross validation method for estimating the model accuracy by using cross-val-score helper function of the scikit learn package. The sample data set fi (fi ϵ f1, f2,….fk) is divided into two parts. Single partition is used to predict the model accuracy and remaining k-1 partitions are used to train the model. The steps to perform K-Fold cross validation are as follows

1. Divide the original sample data into K equal parts named as (fold1 (f1), f2 ...fk). here we consider k=5
2. For each k:
   a. Keep the f1 as validation data set and remaining k-1 sets as training data set.
   b. Train the model using validation data set (f1) and calculated the accuracy of the prediction model.
3. Calculate the model accuracy by averaging the accuracy of all k-1 folds cross partitions.

5. Proposed ANN Based Prediction Model

5.1. Problem Motivation

Cloud computing allows users to access heterogeneous resources as per task requirement. The user submits tasks to cloud broker for execution. The cloud broker submits tasks to scheduler to select the suitable resource for execution. Before actual execution, task-resource mapping is done based on the availability of resources. The scheduler uses several mechanisms to select appropriate resource for example execution time, cost, budget etc., Here, a prediction model is proposed to predict the task execution time for appropriate resource selection. The accuracy of prediction model majorly depends on the input parameters that are fed into the network to predict the output. Therefore, an ISM approach is used for classification and selection of input parameters instead of random selection.

Let us formalize a set of Tasks T= (T1, T2, ….,Tn) and a set of cloud resources R= {R1, R2,…., Rn). Our aim is to predict the task execution time on each available resource Ri to select suitable resource for actual execution. From related work it is analyzed that prediction of accurate execution time results in improved performance in terms of throughput, response time, makespan and cost. To overcome the issues, an ANN based prediction model is proposed using some prior executions dataset referred as training data. The reasons for selection of ANN are as follows [2]

- Suitable for distributed and parallel architectures.
- Self adaptive and flexible.
- Training large amount of dataset.
- Performance of model depends upon trained dataset.
- Deal with non linear and complex data.

To train the model, we use some of the prediction data for various tasks on a number of available heterogeneous resources. Remaining dataset is used for testing the prediction model. To check the model accuracy various statistical metrics are measured.

5.2. Selection of Appropriate Input Parameters Based On ISM

In this section, ISM approach is used for selecting the appropriate input parameters for prediction model. Various relevant parameters (I_1, I_2….I_{14}) are found from literature as listed in Table 1. To form the SSIM, relative relationships among listed parameters are analyzed based on expert opinion. A questionnaire is designed to collect the feedback from experts and output is based on the maximum response for the parameters pair. Based on the relative relationship between parameters I_i and I_j, various symbols (V, A, X, O) have been placed in each cell.

- V represents that parameter I_i influences Parameter I_j - unidirectional relation.
- V represents that parameter I_j influences Parameter I_i - reverse unidirectional relation X represents that both parameters I_i and I_j, both influences each other-bidirectional relation.
- O represents that both parameters I_i and I_j, have no influence with each other-no relation.

Based on these rules, SSIM is prepared and shown in Table 2.

The SSIM is converted into initial reachability matrix by placing 1 or 0 in each cell based on following rules:

- If (Ii, Ij) contains V, then 1 is placed in (Ii, Ij) cell and 0 is placed in (Ij, Ii) cell.
- If (Ii, Ij) contains A, then 0 is placed in (Ii, Ij) cell and 1 is placed in (Ij, Ii) cell.
- If (Ii, Ij) contains X, then 1 is placed in both cells (Ii, Ij) and (Ij, Ii) respectively.
- If (Ii, Ij) contains O, then 0 is placed in both cells (Ii, Ij) and (Ij, Ii) respectively.

The initial reachability matrix is converted into final reachability matrix by removing the transitive dependencies among parameters. The transitive dependency states that if parameter Ii is related to Ij, and Ij is related to Ik, then parameter Ii must be related to Ik. Table 3 shows the final reachability matrix for the selected parameters.

Further reachability matrix is partitioned into different levels based on intersection of Reachability set (Ri) and Antecedent set (Ai). The Ri and Ai are measured from final reachibilty matrix. The Ri contains the parameters itself and the other parameters that it may affect. The Ai contains the attribute itself and the other parameters that may affect it. The final level partitioning is shown in Table 4. The hierarchal architecture and parameters diagraph are shown in Figures 1 and 2 respectively.

Table 2. Structural Self Interaction Matrix (SSIM).

I_i	**I_{14}**	**I_{13}**	**I_{12}**	**I_{11}**	**I_{10}**	**I_9**	**I_8**	**I_7**	**I_6**	**I_5**	**I_4**	**I_3**	**I_2**	**I_1**
I_1	A	A	O	X	V	X	O	O	O	X	A	A	O	
I_2	V	X	V	V	V	A	O	V	V	V	O	V		
I_3	V	O	X	V	V	A	O	V	A	X	A			
I_4	V	O	O	X	V	O	O	V	O	X				
I_5	V	O	O	O	X	A	O	X	A					
I_6	O	O	O	O	O	X	O	V						
I_7	O	O	O	O	O	O	X							
I_8	O	O	O	O	O	O								
I_9	O	O	O	V	V									
I_{10}	O	A	O	O										
I_{11}	O	A	O											
I_{12}	O	O												
I_{13}	A													
I_{14}														

Table 3. Final reachability matrix.

I_i	**I_1**	**I_2**	**I_3**	**I_4**	**I_5**	**I_6**	**I_7**	**I_8**	**I_9**	**I_{10}**	**I_{11}**	**I_{12}**	**I_{13}**	**I_{14}**	**DrP**
I_1	1	1*	1*	1	1	0	0	1*	1*	1	1*	0	1*	0	10
I_2	1	1	1	1	1	1*	1*	1	1	1	1*	1*	0	1*	13
I_3	1	1*	1*	1	1	1*	1*	1	0	1	0	1	1	1	12
I_4	1	0	0	1	1	1*	1*	1	1*	1	1	1	1*	1	12
I_5	1	0	0	1*	1	0	1*	1	1*	1	1	1	0	1	10
I_6	1*	0	0	1*	0	1	1*	1	1*	1	1*	1	1*	1*	11
I_7	0	0	0	1*	1*	0	1	0	0	0	0	0	0	0	3
I_8	1*	0	0	1*	0	0	0	1	0	1	1*	0	0	0	5
I_9	0	0	0	1	1	0	1*	1*	1	1	0	1	1	1	9
I_{10}	0	0	0	1*	1*	0	0	1*	0	1	0	0	0	0	4
I_{11}	1*	1*	0	1*	1*	0	0	1*	0	1	1	1*	1*	1	10
I_{12}	0	0	0	0	0	0	1*	0	0	0	0	1	0	1*	3
I_{13}	1	0	0	1	1	0	1*	1*	1*	0	0	1*	1	1	9
I_{14}	0	0	0	0	1*	0	0	0	0	0	0	1*	0	1	3
DeP	9	4	3	12	11	4	9	11	7	10	7	10	7	10	

Table 4. Final level partitioning.

Parameter	**Ri**	**Ai**	**Ri∩Ai**	**Level**
I_1	$I_1,I_2,I_3,I_4,I_5,I_6,I_8,I_9,I_{11},I_{13}$	$I_1,I_2,I_3,I_4,I_5,I_6,I_8,I_9,I_{11},I_{13}$	$I_1,I_2,I_3,I_4,I_5,I_6,I_8,I_9,I_{11},I_{13}$	II
I_2	I_2, I_3	I_2, I_3, I_{11}	I_2, I_3	VI
I_3	I_3	I_2, I_3	I_3	VI
I_4	$I_1,I_4,I_5,I_6,I_7,I_8,I_9,I_{11},I_{13}$	$I_1,I_2,I_3,I_4,I_5,I_6,I_7,I_8,I_9,I_{11},I_{13}$	$I_1,I_4,I_5,I_6,I_7,I_8.I_9,I_{11},I_{13}$	II
I_5	I_5, I_9, I_{11}	I_2, I_3, I_5, I_9, I_{11}, I_{13}	I_5, I_9, I_{11}	IV
I_6	I_6, I_{11}	I_6, I_{11}	I_6, I_{11}	IV
I_7	I_4,I_5,I_7	$I_2,I_3,I_4,I_5,I_6,I_7,I_9,I_{12},I_{13}$	I_4,I_5,I_7	II
I_8	I_8,I_{11}	$I_2,I_3,I_5,I_6,I_8,I_9,I_{11},I_{12},I_{13}$	I_8,I_{11}	III
I_9	I_5,I_8,I_9,I_{13}	$I_1,I_2,I_3,I_5,I_6,I_8,I_9,I_{13}$	I_5,I_8,I_9,I_{13}	IV
I_{10}	I_4,I_5,I_8,I_{10}	$I_1,I_2,I_3,I_4,I_5,I_6,I_8,I_9,I_{10},I_{11}$	I_4,I_5,I_8,I_{10}	I
I_{11}	I_2,I_{11}	I_2,I_{11}	I_2,I_{11}	VI
I_{12}	I_{12}	I_2, I_3, I_5, I_6, I_9, I_{11}, I_{12}, I_{13}	I_{12}	III
I_{13}	I_{13}	I_2, I_3, I_6, I_{11}, I_{13}	I_{13}	V
I_{14}	I_5,I_{12},I_{14}	$I_2,I_3,I_4,I_5,I_6,I_9,I_{11},I_{12},I_{13},I_{14}$	I_5,I_{12},I_{14}	I

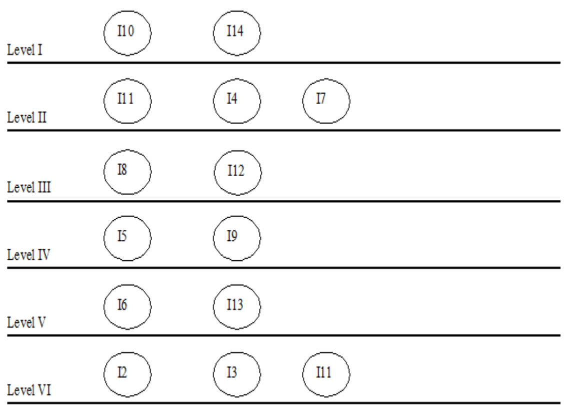

Figure 1. Hierarchical architecture.

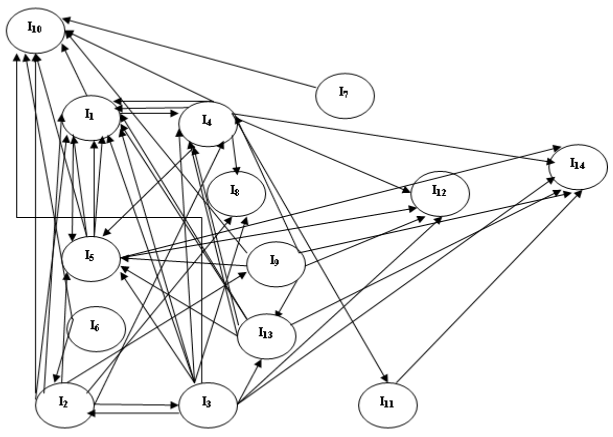

Figure 2. Parameters digraph.

Figure 3 shows the ISM results which indicate that there is no autonomous parameter, it means that all the selected parameters have influenced on the system. The parameters CPU capacity (I3), CPU cores (I2), memory usage (I6), virtual machine (I13), bandwidth (I11), processing element (I9) are having high DrP. Network throughput (I8), cost (I10), disk throughput (I7), resource trust value (I12), and delay (I14) are dependent parameters and affect from the performance of other parameters. CPU speed (I4), task workload (I1), and CPU un-usgae (I5) are linkage parameters and having high DrP and DeP, therefore appropriate selection of these parameters will result in dynamic development of prediction model.

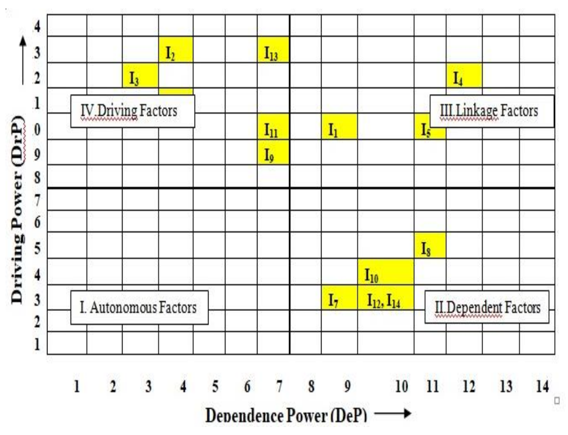

Figure 3. Cluster matrix.

5.3. Model Description

The three parameters task workload (I_1), CPU speed (I_4) and CPU un-usage (I_5) are selected as input parameters due to high predicting power of accurate output as measured through ISM approach. The GWA-T-13 workload data set is used for training and testing the predicting model [11, 20]. The data set contains the values for various fields-Workload, CPU cores, CPU capacity provisioned, CPU speed, CPU usage, Memory provisioned, memory usage, disk throughput, network received and transmitted throughput. The description of selected input parameter is given in Table 5. The data is fed into the prediction model to predict the task execution time as Equation (7)

$$PET = \frac{\text{Workload}}{\text{CPU Speed} * \text{CPU usage}} \tag{7}$$

Table 5. Input data description.

Input Data	Notation	Description
Task Workload	I_1	Number of instructions of task
CPU speed	I_4	Processing capability of CPU
CPU usage	I_5	Percentage use of CPU

The data is normalized and convert into linear form before feed into the network. To normalize and convert the equation into linear form, natural logarithm and coefficients are considered as shown in Equation (8).

$$\begin{gathered}\ln(PET) = \ln A + \ln(I1) - \ln(I4 * I5)\\ \ln(PET) = \ln A + \ln(I1) - \ln(I4) - \ln(I5)\end{gathered} \tag{8}$$

The above equation into the neural network model form is as Equation (9)

$$OPET = W0 + W1*I1' + W2*I2' + W3*I3' \tag{9}$$

Where $O_{PET} = \ln(PET)$
$I_1' = \ln(I_1)$
$I_2' = -\ln(I_4)$
$I_3' = -\ln(I_5)$
$W_0 = \ln A$
$W_1 = W_2 = W_3 = 1$ (initially)

Here, O_{PET} is the PET of the neural model. I_1', I_2', I_3' are input values and W_1, W_2, W_3 are the weight coefficients which are initialized to 1. The derived O_{PET} value is compared with Actual Execution Time (AET) to measure the prediction error. The error should be minimized and propagated to update the weight values using Equations (10), (11), and (12).

$$W_1' = W_1 + e_1 \tag{10}$$

$$W_2' = W_2 + e_1 \tag{11}$$

$$W_3' = W_3 + e_1 \tag{12}$$

Here e_1 is the estimated error of the 1st round prediction. The whole process is repeated until we get the least optimized error for the given inputs. The network is trained using back-propagation algorithm in which error is measured and feed back through the neural layers. Figure 4 shows the proposed ANN based prediction model.

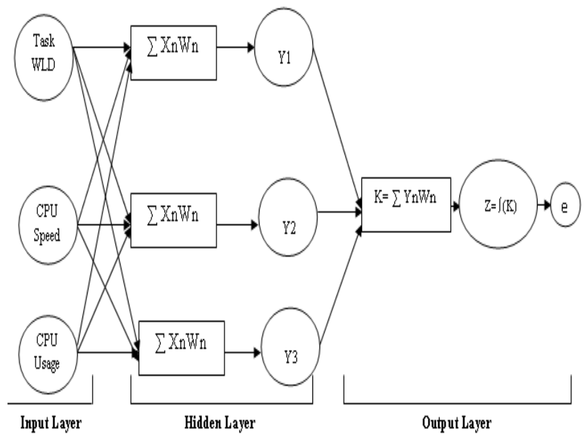

Figure 4. Proposed execution time prediction model.

The model consist four input parameters, two hidden layer and a single output value. All inputs are fed to the hidden layer and interconnected to train the model for given data set. The hidden layer uses a sigmoid function which introduces the nonlinear behaviour of input dataset. The output layer produces a single value as PET on individual resource. The scheduler selects the resource which has least PET value for selected task. After training phase completed, the network is ready to predict the execution time of tasks. Testing data sets are fed to the network to predict the output. The whole procedure continues till the mean square error is close to zero or zero. The training steps are shown in Algorithm I. To measure the accuracy of model, MRE, RMSE, MMRE, MMER and MBRE is calculated.

The results are compared with existing approaches to show the effectiveness of proposed model.

6. Simulation and Results

To verify the effectiveness of the proposed model, work is simulated on Python and results are compared with existing approach. Simulation is performed using windows 7 on Intel Pentium (B940/2 GHz) with 4 GB RAM and 500 MB HDD. For training and testing the prediction model, GWA-T-13 workload data set is used. From the collection of data sets, a subset of data is executed and records the actual execution time for training the model and remaining dataset is used for testing the model. Values and ranges of other parameters that are used in simulation are given in Table 6.

Table 6. Values of various parameters.

Type	Parameter	Range
Machine learning	Number of hidden layers	02
	Learning rate	0.01
	Number of training epochs	25000
	Weight values between connections	Random
	Number of test data set	402
	Number of training data set	598
	Maximum iteration	1000
	Momentum rate	0.3
Tasks	Number of tasks	5000
	Task workload	2-8 (MI)
Resource	Number of resource	70

6.1. Experiment 1-Comparison between Actual and Predicted Execution Time

The 60 percent subset of GWA-T-13 Materna workload data set is used for training the model, and remaining 40 percent data set is used for testing the prediction model. Figure 5 shows the graph between predicted and actual execution time with a step of 250. The graph shows that initially the difference between PET and AET is larger, but after few milliseconds of time, difference between both values is very low. We categorize the data set into five equal training parts and performed K-Fold Cross validation on each data set. The results of cross validation are shown in Table 7.

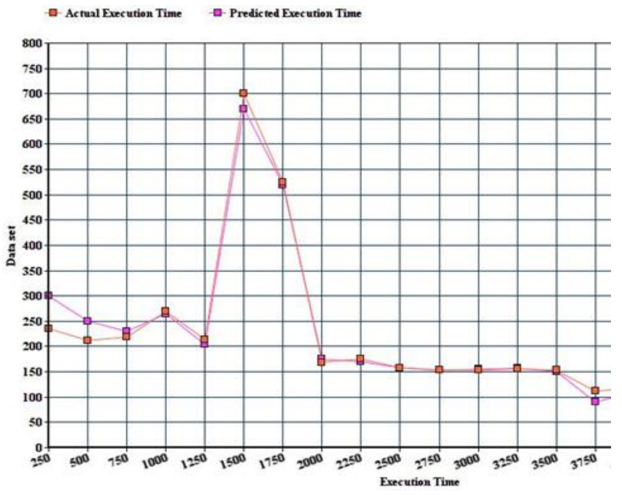

Figure 5. Comparison between actual and predicted execution time.

Table 7. Cross validation results.

Metrics	Training set	Testing set(f1)	Training set	Testing set(f2)	Training set	Testing set(f3)	Training set	Testing set(f4)	Training set	Testing set(f5)
MRE	6.07	13.01	8.3	4.1	8.2	4.33	6.7	10.4	7.9	5.48
RMSE	18.4	11.81	15.15	12.63	6.48	14.79	17	12.16	8.64	14.25
MMRE	0.13	0.06	0.04	0.09	0.043	0.09	0.104	0.07	0.055	0.07
MMER	0.11	0.07	0.04	0.08	0.05	0.09	0.13	0.08	0.05	0.08
MBRE	6.97	0.55	0.55	2.15	0.47	2.17	0.39	2.19	0.79	2.09

6.2. Experiment 2-Comparison between Proposed and Existing approaches

The proposed prediction model is compared with Chang *et al*. [3] and Linear regression [17]. Chang *et al*. [3] presented a prediction based resource selection technique. They achieved 15.49 % and 8.14% MRE while our proposed model reduces up to 7.46 % MRE, which is comparatively better than their results. Comparative analysis of prediction model with existing approaches in terms of MRE, RMSE, MMRE, MMER and MBRE is shown in Table 8.

Table 8. Comparative analysis of prediction model with existing approaches.

Performance Metric	**Proposed Execution Model**	**Chang *et al.* [3]**	**Linear Regression [17]**
MRE	7.46	9.53	8.14
RMSE	12.40	20.03	14.22
MMRE	0.075	0.11	0.081
MMER	0.076	0.126	0.106
MBRE	0.0479	0.273	0.505

Algorithm 1: Training Algorithm

```
Input- Task (T) with Workload (WLD), CPU_Usage (CU), CPU_Speed (CS), and Resource (R)
Output-Predicted Execution Time ($O_{PET}$) for T on R
1. Initialize the weights for all the inputs and set learning rate ($0<\eta<1$).
2. repeat steps 3 to 10
3. for each row of dataset, repeat step 4 to 9
4. Compute the output as-
       $O_{PET} = \ln A + W * \ln(WLD) + (-1)(W * \ln(CS) + W * \ln(1-CU)$
5. Compute Error (E) = $O_{AET1} - O_{PET1}$ where $O_{AET1} = \ln(AET)$ and $O_{PET1} = \ln(PET)$
6. Update weights as follows-
    $dw_1 = \eta * \frac{dE}{dWgt_{ij}}$, where $\frac{dE}{dWgt_{ij}} = E * In\ (WLD)$
    $dw_2 = \eta * \frac{dE}{dWgt_{ij}}$, where $\frac{dE}{dWgt_{ij}} = E * In\ (CS)$
    $dw_3 = \eta * \frac{dE}{dWgt_{ij}}$, where $\frac{dE}{dWgt_{ij}} = E * In\ (1-CU)$
    7. Compute bias as
    $db = \eta * \frac{dE}{dWgt_{ij}}$
8. Update weight as- $w1 = w1 + dw1$; $w2 = w2 + dw2$; $w3 = w3 + dw3$
9. Update bias (b) as- $b = b + db$
10. Repeat the step until get the expected minimum Error
```

7. Conclusions and Future Scope

Most of the time prediction becomes an essential activity for dynamic resource provisioning, selection and scheduling. In this paper, a MLP-ANN based execution time prediction model for resource selection is proposed. The performance of any prediction model greatly depends on the input data that are fed as the input. Therefore, the input parameters are identified and verified by expert judgment and assessment is performed through ISM to provide interrelationship among selected parameters and build a hierarchical model. The parameters task workload (I_1), CPU speed (I_4), and CPU usage (I_5) are found linkage parameters that have high DrP and DeP. It indicates these parameters are highly influenced parameters for the prediction model. The appropriate selection of these parameters will greatly affect the performance of the PET model.

The identified parameters are fed as an input to the prediction model to predict the task execution time. The GWA-T-13 workload data set is used for training and testing the model. The model is compared with existing techniques and provides up to 21.72 % reduction in MRE. Further the model is validated using K-fold cross validation approach to check the effectiveness of proposed model for unknown data sets. In future, the model can be compared with other prediction techniques- recurrent neural network, decision trees, naïve bayes for up gradation.

References

[1] Arlot S. and Celisse A., "A Survey of Cross-Validation Procedures for Model Selection," *Statistics Surveys*, vol. 4, pp. 40-79, 2010.

[2] Bishop C., *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer, 2006.

[3] Chang R., Lin C., and Chen J., "Selecting the Most Fitting Resource for Task Execution," *Future Generation Computer Systems*, vol. 27, no. 2, pp. 227-231, 2011.

[4] Duong T., Zhong J., Cai W., Li Z., and Zhou S., "Ra2: Predicting Simulation Execution Time for Cloud-Based Design Space Explorations," *in Proceedings of Symposium on Distributed Simulation and Real-Time Applications*, London, pp. 120-127, 2016.

[5] Fan Y., Wu W., Xu Y., and Chen H., "Improving MapReduce Performance by Balancing Skewed Loads," *China Communications*, vol. 11, no. 8, pp. 85-108, 2014.

[6] Hasteer N., Bansal A., and Murthy B., "Assessment of Cloud Application Development Attributes Through Interpretive Structural Modelling," *International Journal of System Assurance Engineering and Management*, vol. 8, no. 2, pp. 1069-1078, 2017.

[7] Hecht-Nielsen R., "Theory of the Backpropagation Neural Network," *in Proceedings of International Joint Conference on Neural Networks*, Washington, pp. 593-605, 1989.

[8] Islam S., Keung J., Lee K., and Liu A., "Empirical Prediction Models for Adaptive Resource Provisioning in the Cloud," *Future Generation Computer Systems*, vol. 28, no.1, pp. 155-162, 2012.

[9] Islam T. and Manivannan D., "Predicting Application Failure in Cloud: A Machine Learning Approach," *in Proceedings of International Conference on Cognitive Computing*, Honolulu, pp. 24-31, 2017.

[10] Karamolahy A., Chalechale A., and Ahmadi M., "Energy Consumption Improvement and Cost Saving by Cloud Broker in Cloud Datacenters," *The International Arab Journal of Information Technology*, vol. 15, no. 3, pp. 405-411, 2018.

[11] Kohne A., Spohr M., Nagel L., and Spinczyk O., "FederatedCloudSim: A SLA-aware Federated

Cloud Simulation Framework," *in Proceedings of 2nd International Workshop on CrossCloud Systems*, Bordeaux, pp. 1-5, 2014.

[12] Li H., Wu Y., Chen Y., Wang C., and Huang Y., "Application Execution Time Prediction for Effective CPU Provisioning in Virtualization Environment," *Transactions on Parallel and Distributed Systems*, vol. 28, no. 11, pp. 3074-3088, 2017.

[13] Liu Q., Cai W., Jin D., Shen J., Zhang F., Liu X., and Linge N., "Estimation Accuracy on Execution Time of Run-Time Tasks in a Heterogeneous Distributed Environment," *Sensors*, vol. 16, no. 9, pp. 1-15, 2016.

[14] Meng X., Bradley J., Yuvaz B., Sparks E., Venkataraman S., Liu D., Tsai D., Xin D., Freeman J., Amde M., Owen S., Xin R., Franklin M., Zadeh R., Zaharia M., and Talwalkar A.,"MLlib:Machine Learning in Apache Spark," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1-7, 2016.

[15] Nadeem F. and Fahringer T., "Optimizing Execution Time Predictions of Scientific Workflow Applications in the Grid Through Evolutionary Programming," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 926-935, 2013.

[16] Nemirovsky D., Arkose T., Markovic N., Nemirovsky M., Unsal O., and Cristal A., "A Machine Learning Approach for Performance Prediction and Scheduling on Heterogeneous CPUs," *in Proceedings of 29th International Symposium on Computer Architecture and High Performance Computing*, Campinas, pp. 121-128, 2017.

[17] Oliveira T., Thomas M., and Espadanal M., "Assessing the Determinants of Cloud Computing Adoption: An Analysis of The Manufacturing and Services Sectors," *Information and Management*, vol. 51, no. 5, pp. 497-510, 2014.

[18] Oresko J., Jin Z., Cheng J., Huang S., Sun Y., Duschl H., and Cheng A., "A Wearable Smart Phone-Based Platform for Real-Time Cardiovascular Disease Detection Via Electrocardiogram Processing," *IEEE Information Technology in Biomedicine*, vol. 14, no. 3, pp. 734-740, 2010.

[19] Padhy N., Singh R., and Satapathy S., "Cost Effective and Fault-Resilient Reusability Prediction Model by Using Adaptive Genetic Algorithm Based Neural Network for Web-of Service Applications," *Cluster Computing*, vol. 22, no. 10, pp. 14559-14581, 2018.

[20] Parallel Workloads Archive, http://www.cs.huji.ac.il/labs/parallel/workload, Last Visited, 2018.

[21] Pham T., Durillo J., and Fahringer T., "Predicting Workflow Task Execution Time in the Cloud Using A Two-Stage Machine Learning Approach," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 256-268, 2017.

[22] Ramesh V., Baskaran P., Krishnamoorthy A., Damodaran D., and Sadasivam P., "Back Propagation Neural Network Based Big Data Analytics for A Stock Market Challenge," *Communications in Statistics-Theory and Methods*, vol. 48, no. 14, pp. 3622-3642, 2019.

Anju Shukla is pursuing PhD at Jaypee University of Engineering and Technology, Guna, M.P, India. She has completed B. Tech from Uttar Pradesh Technical University, Lucknow and M.Tech from Shobhit University, Meerut.

Shishir Kumar is working as Professor in the Department of Computer Science and Engineering at Jaypee University of Engineering and Technology, Guna, M.P., India. He has earned PhD in Computer Science in 2005. He has 18 years of teaching and research experience.

Harikesh Singh is working as Assistant Professor in the Department of Computer Science and Engineering at Jaypee University of Engineering and Technology, Guna, M.P., India. He has earned PhD in Computer Science in 2015.