Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.031928
Article

Tech Science Press

Failure Prediction for Scientific Workflows Using Nature-Inspired Machine Learning Approach

S. Sridevi* **and Jeevaa Katiravan**

Department of Computer Science and Engineering, Velammal Engineering College, Chennai, 600066, India
*Corresponding Author: S. Sridevi. Email: sridevi5983@gmail.com
Received: 30 April 2022; Accepted: 20 June 2022

Abstract: Scientific workflows have gained the emerging attention in sophisticated large-scale scientific problem-solving environments. The pay-per-use model of cloud, its scalability and dynamic deployment enables it suited for executing scientific workflow applications. Since the cloud is not a utopian environment, failures are inevitable that may result in experiencing fluctuations in the delivered performance. Though a single task failure occurs in workflow based applications, due to its task dependency nature, the reliability of the overall system will be affected drastically. Hence rather than reactive fault-tolerant approaches, proactive measures are vital in scientific workflows. This work puts forth an attempt to concentrate on the exploration issue of structuring a nature inspired metaheuristics-Intelligent Water Drops Algorithm (IWDA) combined with an efficient machine learning approach-Support Vector Regression (SVR) for task failure prognostication which facilitates proactive fault-tolerance in the scheduling of scientific workflow applications. The failure prediction models in this study have been implemented through SVR-based machine learning approaches and the precision accuracy of prediction is optimized by IWDA and several performance metrics were evaluated on various benchmark workflows. The experimental results prove that the proposed proactive fault-tolerant approach performs better compared with the other existing techniques.

Keywords: Failure prediction; intelligent water drops; support vector regression; proactive fault-tolerance; scientific workflows; precision accuracy; resource provisioning

1 Introduction

Cloud is the buzzword in the computational technologies that has brought a paradigm shift in the way data is stored and computing is performed. Cloud computing is a subscription-based service that delivers computation as a utility. The key characteristics of cloud computing are providing elastic, on-demand delivery of services with dynamically configurable resources and innovative pricing models. In the past decade, the concept of scientific workflow has emerged as a booming paradigm for modeling large scale complex data in diverse computing domains. Scientific workflows are abstractions composed of activities and data with intricate dependencies managed by complex engines.

Fault-tolerance is one of the critical factors to guarantee the reliability of cloud services. Fault-tolerance is the ability of the system to keep working in presence of one or more faults but with decaying performance. Fault-tolerance strategies in cloud are classified as two types: reactive and proactive. Reactive fault-tolerance strategies are the techniques used to effectively troubleshoot a system upon occurrence of failure(s). Various reactive fault-tolerance strategies are: Replication, Task resubmission, checkpointing and so on. The execution of such techniques automatically results in the performance degradation of the system. Proactive fault-tolerance makes use of a prediction approach to anticipate the failures in advance thereby reducing the downtime of the system.

Failure prediction influences the delivery of on-demand services in the cloud. Accurate predictions of failure prone machines can help in mitigating the impact of failures in a proactive manner. Hence this work emphasizes on a mathematical prediction-based approach to accurately anticipate the failure prone hosts in the cloud resource pool which facilitates the early migration of the virtual machines to other active hosts with no performance degradation.

2 Literature Review

2.1 Machine Learning Approaches for Failure Prediction

Cagatay Catal (2011) suggested Naive Bayes as the robust machine learning algorithm for supervised software fault prediction [1]. Malhotra and Ankita Jain (2012) observed that the random forest and bagging gave the best results for fault predictions [2]. But they did not consider the effect of size, resource related issues on fault proneness and its severity. Islam, et al. (2012) proved that the prediction accuracy of Error Correction Neural Network (ECNN) demonstrated superior effectiveness for forecasting resource utilization in the cloud [3].

Fronza, et al (2013) introduced a new approach for predicting failures based on Support Vector Machines (SVM) and Random Indexing (RI) [4]. The results of their work proved that weighted SVMs perform better in improving sensitivity. This approach proved reliability in predicting both failures and non-failures. Anju Bala and Inderveer Chana (2015) have designed intelligent task failure prediction models that have been implemented through machine learning approaches like ANN, Naive Bayes, Random Forest, and LR using evaluated performance metrics [5]. The approaches proposed in this paper only predicts whether the task fails or not but does not determine the rate of failure of each task.

Upasna Kothari and Moe Momayez (2018) proposed the use of machine learning to predict the time of failure [6]. The results of the study proved that Machine Learning provided prediction values that are 86% of the time closer to the actual time of failure when compared to the traditional methods. Padmakumari, et al [7] used the ensemble with the combination of machine learning methods in workflow environment to improve accuracy and efficiency of the prediction model. Till now no work has been proposed using the combination of SVR and IWD in the failure prediction of scientific workflows.

2.2 Nature-Inspired Support Vector Regression

Ming, et al. [8] (2017) constructed a PV power prediction model based on EMD and ABC-SVM and proved that the method is superior to other approaches. The model proposed by Sheng, et al. [9] (2015) is a hybrid of LS-SVR and SFA in which SFA integrates an artificial firefly colony algorithm with chaotic map, adaptive inertia weight, and Levy flight. It uses SFA to optimize LS-SVR hyper parameters (i.e., regularization parameter and sigma parameter) and then uses LS-SVR for prediction.

3 Approaches Used for Task Failure Prognostication

3.1 Support Vector Machines for Regression

The concept of Support Vector Machine (SVM) was first introduced by Boser, Guyon and Vapnik. SVM is a supervised machine learning algorithm based on statistical learning theory to fix both non-linear

classification and regression challenges. It has great performance since it can handle a non-linear classification efficiently by mapping samples from low dimensional input space to high dimensional feature space with a non-linear kernel function. The key parameter of SVM is the type of kernel function used. Kernel functions are used for mapping into a higher dimensional feature space. Mercer's theorem is used for the construction of positive definite kernel for SVM regressor.

Support Vector Regression (SVR) has been applied to many actual issues like predicting the performance of compact heat exchangers (Peng and Ling, 2015), modeling of heat transfer in a thermo syphon reboiler (Zaidi, 2015), predicting the sorption capacity of lead (II) ions (Nusrat Parveen et. al, 2016), predicting the price in a car leasing application (Mariana Listiani, 2009), modeling and predicting Turkey's electricity consumption (Kadir Kavaklioglu, 2010), analyzing prognosis of infants with congenital muscular torticollis (Suk-Tae Seo, 2010), forecasting the demand and supply of wood pulp (V. Anandhi, 2013), and so on.

The architecture of Support Vector Regression is given in Fig. 1:

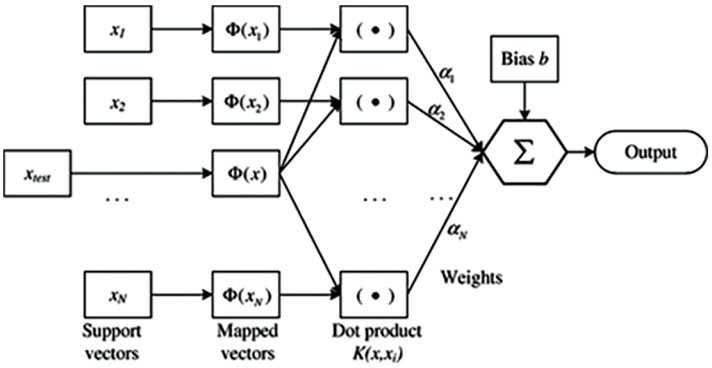

Figure 1: Architecture of a regression machine constructed by the support vector algorithm [10]

The pseudocode for the intelligent water drops algorithm is given in Fig. 2

[Step 1] Initialization of static and dynamic parameters

[Step 2] Random distribution of Intelligent Water Drops (IWDs) along the graph nodes and edges.

[Step 3] Updation of visited node list of each IWD.

[Step 4] Selection of next node.

[Step 5] Updation of local soil and velocity for each moving IWD.

[Step 6] Determination of the local-best solution and updation of soil on its path.

[Step 7] Updation of global soil.

[Step 8] Determination of the Total-best solution and continuation of the process until the completion of all water drops.

Figure 2: Pseudocode for intelligent water drops algorithm

The flowchart for the proposed system is given in Fig. 3

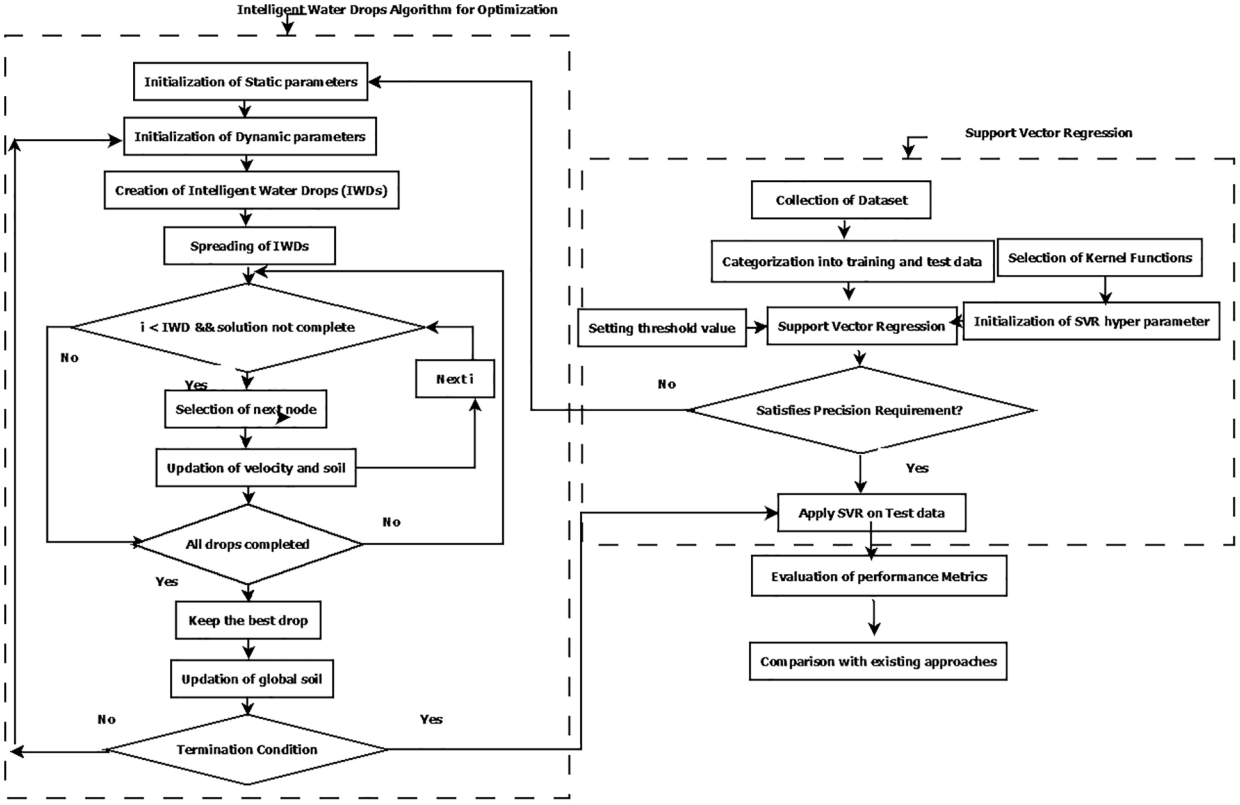

Figure 3: The proposed IWD-SVR based approach for failure prognostication

The steps of IWDA to optimize the SVR parameter is given in Fig. 4

[Step 1] ***Let the input dataset be T. The sample size of dataset be n. Divide the dataset into training set and test set, the sample size for training set is k and the sample size for testing set is n Ök.***

[Step 2] ***Select the kernel function type of SVR as Gaussian Radial Basis Function kernel function.***

[Step 3] ***Set initial global optimal value*** T^{IB}***= min(mid(SVR penalty factor),Gaussian RBF)***

[Step 4] ***If MSE of anticipation function meets accuracy requirement, goto step 10; if do not meet, perform step 5, carry out SVR parameter adjustment with IWD algorithm.***

[Step 5] ***Set*** $N_{iB} = N_1 + N_2$ ***where*** N_1 ***= position of the number of significant digit of SVR penalty factor and*** N_2 ***= position of the width of Gaussian RBF.***

[Step 6] ***Generate parameter adjustment path profile C and α using IWDA.***

[Step 6.1] Initialize static parameters

Water Droplet Quantity = N;
Velocity updating parameters ($a_v = 1$, $b_v = 0.01$, $c_v = 1$);
Soil updating parameters ($a_s = 1$, $b_s = 0.01$, $c_s = 1$)
initial velocity = 0, initial soil =100

Figure 4: (Continued)

[Step 6.2] Initialize dynamic parameters :

$Soil^{IWD}$: Soil of IWD
Vel^{IWD} : Velocity of IWD
Velc: Visited node list
Edgesoil(i,j) : Soil along the edge, while flowing from i^{th} node to j^{th} node.

[Step 6.3] Update the visited node list

[Step 6.4] Update the dynamic parameters of IWD as

$$vel^{IWD}(t+1) = vel^{IWD}(t) + \frac{a_v}{b_v + c_v * soil^2(i,j))}$$

$$soil^{IWD} = soil^{IWD} + \Delta soil(i,j)$$

$$\Delta soil(i,j) = \frac{a_s}{b_s + c_s * time(i,j : vel^{IWD}(t+1))}$$

So that

$$time\big(i,j : vel^{IWD}(t+1)\big) = \frac{HUD(i,j)}{vel^{IWD}(t+1)}$$

Where HUD(j) is the heuristic undesirability defined for the problem. The soil along the path is updated using

$$soil(i,j) = (1-\rho_n).soil\ (i,j) - \rho_n.\Delta soil(i,j)$$

[Step 6.5] Update the soil along the path based on the current iteration best solution.

$$soil(i,j) = (1+\rho_{IWD}).soil(i,j) - \rho_{IWD}.\frac{1}{(N_{IB}-1)} soil_{IB}^{IWD} \forall\ (i,j) \in S^{IB}$$

Where N_{IB}, is the number of nodes in the iteration best solution, and S^{IB} is the iteration best solution.

[Step 7] ***Update the global best solution based on the objective function.***

[Step 8] ***Increment the iteration count, and repeat the steps until the stopping criteria is met. After all IWD iteration complete, N number of water droplet obtains N group parameter C and α.***

[Step 9] ***The parameter C searched for each IWD and α, utilizes study collection to carry out the method for model training according to step 3.***

[Step 10] ***Adopt these values to carry out the determination of failure prognostication; the accuracy of the process is high.***

Figure 4: Steps of intelligent water drops algorithm to optimize SVR parameter

The types of kernel functions for Support Vector Regressor is tabulated in Tab. 1

Table 1: Types of kernel functions for SVR

Types of Kernel	Formula
Linear kernel	$k(x,\ y) = x^T y + c$
Polynomial kernel	$k(x,\ y) = (\alpha x^T y + c)^d$
Gaussian kernel	$k(x,\ y) = \exp(-\gamma \|\|x - y\|\|2)$
Laplace radial basis function	$k(x,\ y) = \exp\left(\frac{-\|\|x-y\|\|}{\sigma}\right)$
Sigmoid kernel	$k(x,\ y) = \tanh(\alpha x^T y + c)$
Anova radial basis	$k(x,\ y) = \sum_{k=1}^{n} \exp(-\sigma(x^k - y^k)^2)^d$
Gaussian RBF	$K(X,\ Y) = e^{\frac{-\|\|X-Y\|\|^2}{2\sigma^2}}$

3.2 Intelligent Water Drops Algorithm for Optimization

The behavior of natural resources acts as a rich source of inspiration for us in tackling numerous real life optimization problems. Intelligent Water Drops Algorithm (IWDA) [11] is one among them in which the flow of water drops along the twist-and-turn path of the natural rivers to determine an optimal path towards its destination helps us to implement it in the form of an algorithm to solve optimization issues. The IWD algorithm is a population based constructive optimization algorithm whose idea depends on the responses of the natural water drops in rivers.

In the IWD algorithm, each intelligent water drop is created with two properties: velocity of the water drop and amount of soil each water drop carries. These properties commence with an initial value and change during the flow of an IWD from a source to its destination. The trip of an IWD begins with an initial velocity and zero soil.

During its journey, the IWD flows in discrete steps from its current location to its next location. Hence the velocity of the IWD increases non-linearly proportional to the inverse of the amount of soil between the two locations. Therefore, a path with less soil lets the IWD move with greater velocity than a path with more soil. An IWD gathers soil during its journey which is non-linearly proportional to the inverse of the time needed to travel from the current location to the next. The time taken is proportional to the velocity of the IWD.

The IWD uses a mechanism to select its path to the next location. According to this mechanism, it prefers to travel in the path with low soil so that it can move with greater velocities. The same mechanism can be adopted to find an optimal solution in several real-time problems. Some of the popular applications of IWDA are Vehicle routing problem, Multiple Knapsack problem, Job Shop scheduling, Travelling Salesman problem and so on. The pseudo-code of an IWD-based algorithm might be indicated in eight stages:

4 The Proposed IWD-SVR Approach

The proposed model seeks to adopt intelligent water drops algorithm to optimize the support vector regression parameter which involves the following steps:

5 Evaluation Criteria

This work correctly identifies which tasks could be failed due to resource overutilization. For predicting its performance, the data were collected from different scientific workflows with 25, 50 and 100 and 1000 tasks at fixed intervals using the CloudSim [12] and WorkflowSim [13] tools. The task failure prediction accuracy has been evaluated using some evaluation metrics [14]. Pegasus [15] takes care of abstract mapping to concrete workflows.

5.1 Sensitivity (Recall)

Sensitivity is a measure of the proportion of actual positive cases that got predicted as positive (or true positive).

$$\boldsymbol{Sensitivity} = (\boldsymbol{TP})/(\boldsymbol{TP} + \boldsymbol{FN}) \tag{1}$$

- TP-True Positive = Tasks that are labeled as failure and also evaluated as failed tasks.
- FN-False Negative = Tasks that are labeled as non-failure but evaluated as failed tasks.

5.2 Specificity

Specificity is defined as the proportion of actual negatives, which got predicted as the negative.

$$\textit{Specificity} = (TN)/(TN + FP) \tag{2}$$

- TN–True Negative = Tasks labeled as non-failure and also evaluated as non-failed tasks.
- FP-False Positive = Tasks labeled as failure but evaluated as non-failed tasks.

5.3 Precision

Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives.

$$\textit{Precision} = (TP)/(TP + FP) \tag{3}$$

5.4 F1 Score

F1 Score is the Harmonic Mean between Precision and Recall. The range for F1 Score is [0, 1]. The greater the F1 Score, the better is the performance of our model.

$$\mathbf{F1} = 2* \frac{1}{\frac{1}{\mathbf{precision}} + \frac{1}{\mathbf{recall}}} \tag{4}$$

5.5 Classification Accuracy

Classification Accuracy is the ratio of number of correct predictions to the total number of input samples.

$$\textit{Accuracy} = \frac{\textit{Number of correct predictions}}{\textit{Total number of predictions made}} \tag{5}$$

6 Experimentation Results and Discussions

Our experimental setup has four steps:

- Data collection
- Evaluation of performance metrics using percentage splits
- Comparison of proposed approach with existing models for various scientific workflows.

6.1 Data Collection

CloudSim [12] and WorkflowSim [13] classes are used to collect the dataset for failure prognostication. For predicting its performance, the data were collected from different scientific workflows with 25, 50, 100 and 1000 tasks at fixed intervals using the CloudSim [12] and WorkflowSim [13] classes. The attributes considered in the failure prediction process is listed in the Tab. 2

The maximum dynamic threshold is set using the utilization parameters such as CPU, Bandwidth, RAM, Memory, Disk and EET based on the current values and the historical data. If the current utilization value is greater than the maximum threshold value, the task is categorized as failure.

Table 2: Attributes considered for dataset collection

Attribute name	Description
T_{id}	Task id
VM_{id}	Virtual machine id
D_{id}	Data Center id
U.BW	Utilization of bandwidth in percentage
U.CPU	Utilization of CPU in percentage
U.RAM	Utilization of RAM in percentage
U.Mem	Utilization of memory in percentage
$Time_S$	Start time of execution
$Time_F$	Finish time of execution
Status	Success or Failure
EET	Estimated execution time

T_{id} and VM_{id} denote the id of the failed task and its corresponding Virtual Machine. D_{id} denote the id of the datacenter where the task failure occurs. Percentages split used for training and testing data are 66% and 34% respectively.

6.2 Evaluation of Performance Metrics Using Percentage Split

The main contribution of this work is a SVR-IWD based approach in which the accuracy of the failure prognostication is high compared with the existing approaches. Figs. 5–7 illustrates the comparison of various performance metrics such as sensitivity, specificity, precision, recall and F1 score.

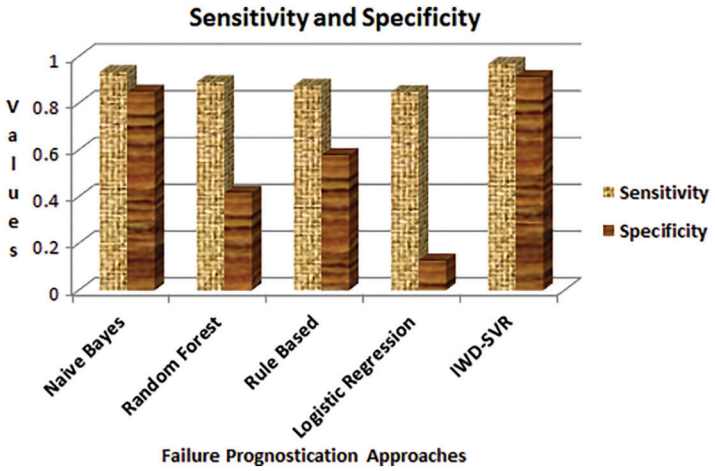

Figure 5: Comparison of sensitivity and specificity

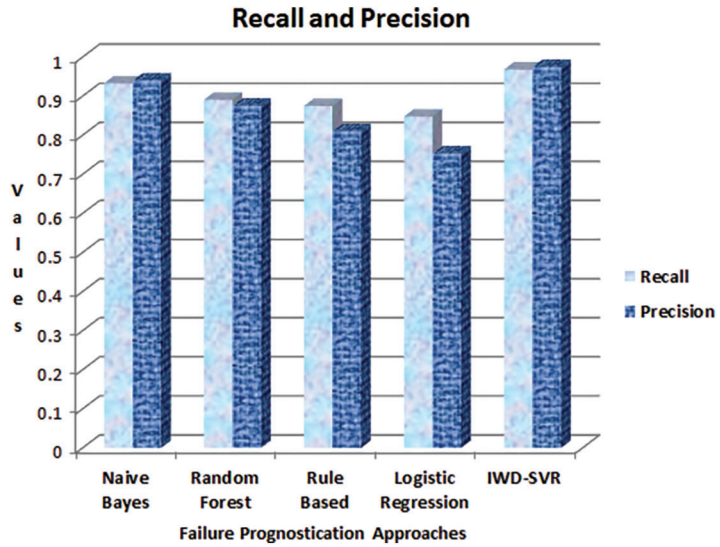

Figure 6: Comparison of recall and precision

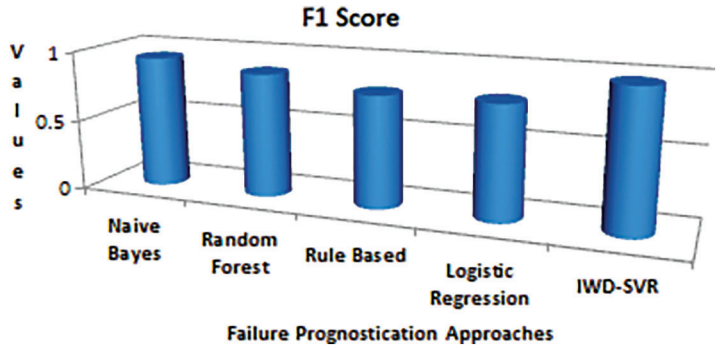

Figure 7: Comparison of F1 Score

6.3 Comparison Evaluation of Various Scientific Workflows with Existing Approaches

The proposed IWD-SVR based approach is evaluated using various scientific workflows such as Epigenomics, Montage, CyberShake, Inspiral and Sipht. To assess the algorithm, synthetic workflows in the scientific community like Montage (astronomy), Epigenomics (biology), LIGO (gravitational physics) and SIPHT (biology), CyberShake are utilized. These are accessible as abstract workflow in XML format (DAX). All the algorithms are assessed using four different sizes of workflows. We have executed our IWD-SVR in the WorkflowSim simulation tool. Each workflow differs according to the characteristics and tasks.

The Montage workflow stitches multiple input images together to create custom mosaics of the sky. The Cybershake workflow is used to characterize earthquake hazards in a region. The LIGO workflow is used to analyze gravitational waveforms. The SIPHT workflow is used to automate the searching process of untranslated RNAs in the NCBI database. The Epigenomics workflow is used to automate various operations in genome sequence processing.

Fig. 8 illustrates the accuracy comparison of proposed approach with existing Naive Bayes, Random Forest, Rule Based, and Logistic Regression. The overall accuracy of the proposed approach is 98.2% which is 2.18% higher than the existing Naive Bayes approach.

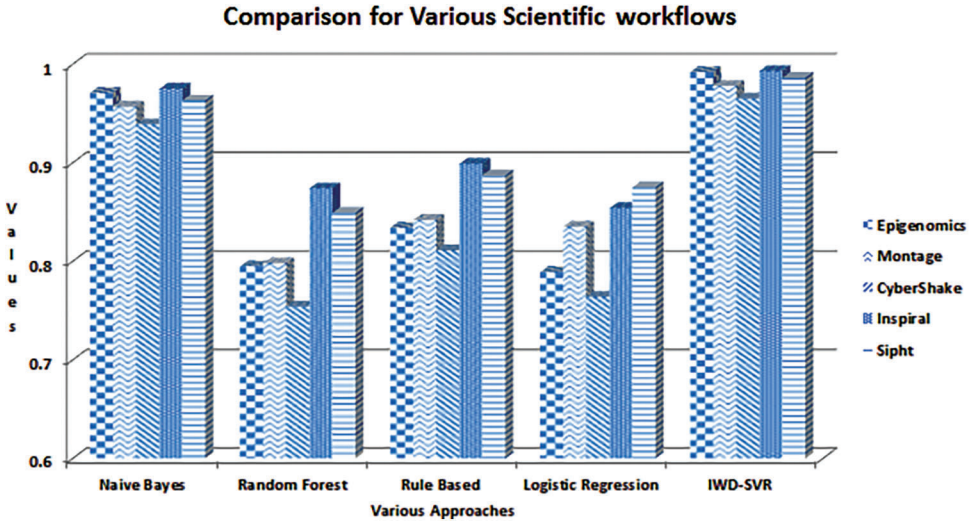

Figure 8: Accuracy comparison using various scientific workflows

7 Conclusion

Proactive fault-tolerance mechanism is preferred in scientific workflows in the cloud environment because they can anticipate the failure well ahead so that there is sufficient time for migration and mitigation of failures. Proactive mechanisms are mainly based on the historical data to predict the future faults. The efficiency of such mechanisms is based on several factors of which the prediction accuracy plays the vital role. Hence the focus of my work is to enhance the prediction accuracy of the proactive fault-tolerance mechanism. In our proposed IWD-SVR based approach, the prediction accuracy can be achieved better by nearly 2.18% higher compared to the existing approaches. The future work of the proposed system is to implement it in real-time to support current working environment.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] C. Catal, "Software fault prediction: A literature review and current trends," *Expert Systems with Applications*, vol. 38, no. 1, pp. 4626–4636, 2011.

[2] R. Malhotra and A. Jain, "Fault prediction using statistical and machine learning methods for improving software quality," *Journal of Information Processing Systems*, vol. 8, no. 2, pp. 241–262, 2012.

[3] S. Islam, J. Keung, K. Lee and A. Liua, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Generation Computer Systems*, vol. 28, pp. 155–162, 2012.

[4] I. Fronzaa, A. Sillitti, G. Terho and J. Vlasenkoa, "Failure prediction based on log files using random indexing and support vector machines," *The Journal of Systems and Software*, vol. 86, no. 12, pp. 2–11, 2013.

[5] A. Bala and I. Chana, "Intelligent failure prediction models for scientific workflows," *Expert Systems with Applications*, vol. 40, pp. 980–989, 2015.

[6] U. Chandarana, K. Kothari and M. Momayez, "Machine learning: A novel approach to predicting slope instabilities," *International Journal of Geophysics*, vol. 9, pp. 152–172, 2018.

[7] P. Padmakumari and A. Umamakeswari, "Task failure prediction using combine bagging ensemble (CBE) classification in cloud workflow," *Wireless Personal Communications*, vol. 107, pp. 23–40, 2019.

[8] X. Ming, S. Yang and S. Pan, "Optimal parameter selection for support vector machine based on artificial bee colony algorithm: A case study of grid-connected pvsystem power prediction," *Hindawi Computational Intelligence and Neuroscience*, vol. 14, no. 12, pp. 156–171, 2020.

[9] J. Sheng, S. Chou, N. Ngo and A. Pham, "Shear strength prediction in reinforced concrete deep beams using nature-inspired metaheuristic support vector regression," *American Society of Civil Engineers*, vol. 14, no. 2, pp. 156–172, 2015.

[10] N. Parveen, S. Zaidi and M. Danish, "Support vector regression model for predicting the sorption capacity of lead (II)," *Perspectives in Science*, vol. 8, pp. 629–631, 2016.

[11] H. Hosseini, "The intelligent water drops algorithm: A nature-inspired swarm-based optimization algorithm," *International Journal Bio-Inspired Computation*, vol. 1, no. 2, pp. 42–49, 2009.

[12] R. Calheiros. N. Ranjan, R. Beloglazov and A. Rose, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, pp. 23–50, 2011.

[13] W. Chen and E. Deelman, "WorkflowSim: A toolkit for simulating scientific workflows in distributed environments," in *IEEE 8th Int. Conf. on E-Science*, China, vol. 10, pp. 1–8, 2012.

[14] A. Mishra, "Metrics to evaluate your machine learning algorithm," *Towards Data Science*, vol. 12, no. 4, pp. 1224–1230, 2015.

[15] E. Deelman, "Pegasus: A framework for mapping complex scientific workflows onto distributed systems," *Scientific Programming*, vol. 13, pp. 219–237, 2005.