JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering, Vol. 16, No. 4, December 2022, pp. 222-232

A Study of Job Failure Prediction on Supercomputers with Application Semantic Enhancement

Haotong Zhang
College of Information Science and Engineering, Chongqing Jiaotong University, Chongqing, China
tongx2 zh@163.com

Gang Xian[†], Wenxiang Yang[†]
Computational Aerodynamics Institute, China Aerodynamics Research and Development Center, Mianyang, China
gang xn@foxmail.com, yangwenxiang10@nudt.edu.cn

Jie Yu*
Computational Aerodynamics Institute, China Aerodynamics Research and Development Center, Mianyang, China
State Key Laboratory of Aerodynamics, China Aerodynamics Research and Development Center, Mianyang, China
yujie@nudt.edu.cn

Abstract

The powerful computing capabilities of supercomputers play an important role in today's scientific computing. A large number of high performance computing jobs are submitted and executed concurrently in the system. Job failure will cause a waste of system resources and impact the efficiency of the system and user jobs. Job failure prediction can support fault-tolerant technology to alleviate this phenomenon in supercomputers. At present, the related work mainly predicts job failure by collecting the real-time performance attributes of jobs, but it is difficult to be applied in the real environment because of the high cost of collecting job attributes. In addition to analyzing the time and resource attributes in the job logs, this study also explores the semantic information of jobs. We mine job application semantic information from job names and job paths, where job path is collected by additional monitoring of the job submitting process. A prediction method based on job application semantic enhancement is proposed, and the prediction results of the non-ensemble learning algorithm and the ensemble learning algorithm are compared under each evaluation indicator. This prediction method requires more miniature feature collection and computation overhead and is easy to apply. The experimental results show that the prediction effect is promisingly improved with job application semantic enhancement, and the final evaluation indicator S score is improved by 5%-6%, of which is 88.16% accuracy with 95.23% specificity and 88.24% sensitivity.

Category: Smart and Intelligent Computing

Keywords: Execution Efficiency; Job Failure Prediction; Application Semantic Enhancement; Machine Learning

Open Access **http://dx.doi.org/10.5626/JCSE.2022.16.4.222** **http://jcse.kiise.org**

Received 21 August 2022; **Accepted** 14 December 2022

*Corresponding Author

[†]These authors contributed equally to this work and should be considered co-first authors.

I. INTRODUCTION

High Performance Computing (HPC) is a branch of computer science, which mainly refers to the research and development of high performance computer technology from the aspects of architecture, parallel algorithms and software development, and mainly refers to a class of computers with supercomputing capabilities. It is committed to developing high performance computers, mainly using simulation methods to calculate today's ultra-large, ultra-high and ultra-complex computing tasks, research parallel algorithms, and develop related software. High performance computing has been recognized as the third scientific research method for human beings to understand the world and transform the world after theoretical science and experimental science.

Supercomputers achieve higher-performance com- puting capabilities than current mainstream comput- ers through high-speed interconnection and paral- lelization of multiple compute nodes. Each compute node is similar to an independent computer with its CPU, memory and operating system. Generally, the data is accessed in a shared storage way in the super- computer, which can ensure the correctness of the data. In addition, a job scheduling manager is a unique software in supercomputers, which mainly responds to the jobs submitted by the user, and can also schedule and manage the jobs. Meanwhile, the job scheduling manager will record each job submit- ted by the user. In this paper, the system job log will be used as the study object.

Users mainly complete some advanced computa- tional science and engineering tasks [1, 2] (such as long-term weather forecasting, oil exploration, and overall aircraft power) by submitting high perfor- mance computing jobs. Users submit a large number of jobs to be executed in supercomputers every day, and these jobs occupy many system resources and en- ergy [3]. Significantly, the failure of large-scale and complex jobs will cause waste of system resources, affect the overall execution efficiency of the system, prolong the waiting time of subsequent queuing jobs and reduce the effectual output of users. For example, there are more than 670,000 jobs in the resources load log of the Google data center, and more than 40% of the jobs fail [4, 5]. Therefore, ensuring efficient uti- lization of system resources and improving execution efficiency is a major issue in supercomputers.

Job failure prediction is a solution to solve the waste of system resources. It mainly predicts which jobs will fail and feeds them back to users to help users make decisions. Meanwhile, job failure prediction can also be used to support fault- tolerant technology [6-8]. In the early days, people mainly used some statistical learning methods [9, 10] and used the resource and performance attributes of the job to predict job failure, but there were disadvantages of large sample size and complicated computation. In this paper, we study the job logs of a real production supercomputer, which holds many job attributes. In addition to exploring the resources and other attributes of the job, it is also find that there are a large number of jobs with similar job names in the user's jobs, and the job names contain rich semantic information, which may have a certain indicating effect in predicting job failures. Adding additional monitoring information, i.e., the submission path of the job, with the hierarchical information of the job source and other relevant semantic information in the path, can make the job more fine-grained. This study proposes the definition of job application and the job failure prediction method based on job application semantic enhancement. In addition, the prediction effects under the non-ensemble learning algorithm and the ensemble learning algorithm are compared.

The rest of this paper is organized as follows. Sec- tion II gives an overview of the related works. Section III introduces relevant background knowledge. Sec- tion IV introduces the job logs analysis. Section V introduces application semantic enhancement. Sec- tion VI shows the experimental details. Section VII introduces the discussion. Finally, we conclude with a summary of our work in Section VIII.

II. RELATED WORK

Due to certain instability in high performance computing systems, various failures will occur, which is very detrimental to the execution efficiency of the system and the effectual output of users. Job failure prediction can be used as a supporting means of fault- tolerant technology in high performance computing. Much related research work can be roughly divided into two levels: system level failure prediction and job level failure prediction. The study in this paper is at the job level.

A. System Level

Morais et al. [11] developed an automatically scalable service development framework based on various CPUs, using Linear Regression (LR), Auto Correlation (AC) and Auto Regressive Integrated Moving Average (ARIMA) algorithm. Zhao et al. [12] studied the problem of disk fault prediction. They used the different features of continuous time interval measurement of the disk drive as time series and used Hidden Markov Model (HMM) and Hidden Semi- Markov Model (HSMM) to model these time series to identify disk failure. Gentner et al. [13] used the data got from the cycle to predict integrated circuit (IC) failures. In addition to hardware failure prediction, Jayanthi et al. [14] and Kumaresan et al. [15]

focused on software reliability modeling with software defect prediction using the neural network classifier method.

These research works are all proposed for specific system failure problems with certain limitations.

B. Job Level

Samak et al. [16] used the Naive Bayesian (NB) classification algorithm to predict the failure of tasks in the process of scientific workflows. El-Sayed et al.

[17] developed a work failure prediction model based on Random Forest (RF) classifier. Yoo et al. [18] used the RF algorithm to extract and characterize the patterns of failed job statues, but the cost of feature data collection is very high and may increase the system load. Banjongkan et al. [19] compared the prediction effect of using job queue attribute with time attributes under the Decision Tree algorithm to predict job failure at job submit-state and job start- state.

In this paper, the use of job application semantic enhancement to predict job failure can enrich the job working mode and improve the effect in the prediction model, with minor features collection overhead and more robust applicability.

III. BACKGROUND

A. High Performance Computing Job

High performance computing jobs come from users' research work or projects, and their job application directions are different. In this paper, the application of jobs is defined as follows according to the semantic attributes of jobs:

DEFINITION 1 (Application). Jobs with similar job path and similar job name belong to the same application.

The different states during the life cycle of HPC jobs are shown in Fig. 1. After a user submits a job, the job is queued in the waiting queue to be scheduled, and the job can have many termination states, such as canceled, failed or completed. When the termination status of a job is abnormally completed, the job may be resubmitted and executed by the user. Multiple submitted jobs may come from the same application, and these jobs have only minor differences in semantics. We will elaborate on the methods to enhance the semantics of job application in Section V. Meanwhile, this paper's job failure prediction method is suitable for the prediction after job submission, which can give users more response time to job failure.

B. Machine Learning Algorithm

The machine learning classification mechanism is a type of supervised learning, which is the process of learning from a training set of existing class labels to predict the class labels of anonymous data with input features associated with the class labels.

Decision trees (non-ensemble algorithm) and Random Forest (ensemble algorithm) are used as modeling methods in this study. Decision Tree (DT) [20, 21] is a non-parametric supervised learning method that can summarize decision rules from a series of data with features and labels, and present these rules in a dendrogram structure to solve classification problems. The decision tree algorithm has the advantages of simple working process, easy explanation and strong applicability. Fig. 2 illustrates the process of decision tree classification. The place where the initial question is located is called the root node, each question before the conclusion is reached is an intermediate node, and each conclusion obtained is called a leaf node.

Impurity is used as an indicator of branch generation in decision tree. In this paper, the Gini impurity is used as the calculation formula of impurity. The calculation formula is as follows:

$$Gini(t) = 1 - \sum_{i=0}^{c-1} p(i|t)^2 \tag{1}$$

where t represents the specified node, c represents the number of sample categories, and $p(i|t)$ represents the proportion of the classification label i on the node t. Generally, the lower the impurity, the better the decision

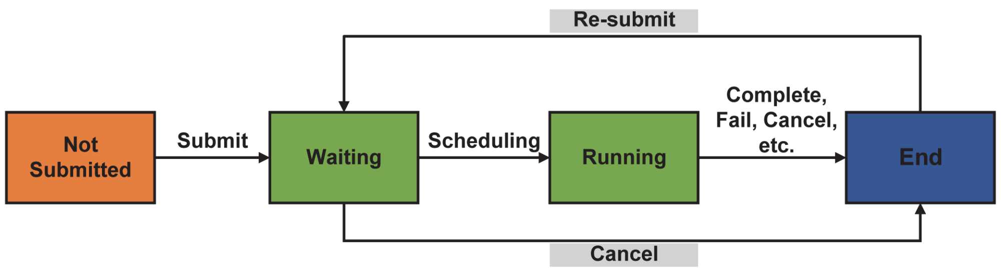


Fig. 1. The life cycle of HPC jobs.

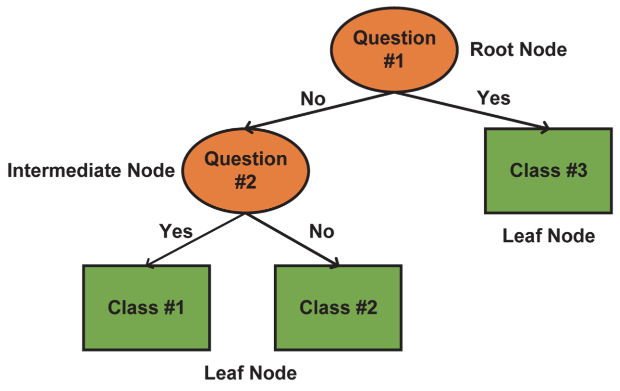


Fig. 2. The process of decision tree classification.

tree fits the training set.

Ensemble learning [22] is a very popular machine learning method. It is not a separate machine learning algorithm, but builds multiple models on the data to synthesize the modeling results of all models. Random Forest (RF) [23, 24] is composed of multiple decision trees, each decision tree is constructed by randomly selecting a predefined number of features, and the mode of the results of multiple decision trees is used as the prediction result, which can prevent overfitting to a certain extent.

IV. JOB LOGS ANALYSIS

Slurm (simple Linux utility for resource management) is a highly scalable and fault-tolerant cluster manager and job scheduling system that can be used for large-scale compute node clusters and is widely used by supercomputers and high performance computing clusters around the world [25]. Slurm maintains a queue of pending jobs and manages the overall resource utilization of this job. It manages the available compute nodes (depending on resource requirements) in a shared or non-shared manner for users to execute jobs. Slurm appropriately allocates resources to job queues and monitors jobs until they are complete. Today, slurm has become the leading resource manager used on many of the most powerful supercomputers.

In this study, the data studied are from a produc- tion supercomputer at the China Aerodynamics Re- search and Development Center (CARDC), whose job scheduler is slurm and has 11,392 compute nodes [26]. The institute is mainly devoted to computational fluid dynamics (CFD) research tasks. We collected the log records of 472,251 jobs for around one year. In order to protect the privacy of users, the data shown in this paper are desensitized.

The relevant attributes of each job will be recorded in the job log, and each attribute will be stored separately in the form of a field. Table 1 shows the relevant attributes of the job recorded in the job log. The **State** field contains various types, such as CANCELLED, COMPLETED and FAILED. In this paper, the jobs marked as COMPLETED are regarded as successful jobs, and the jobs marked as FAILED are regarded as failed jobs. The jobs in these two termination states are our research targets.

Table 1. Job log field of slurm record

Category	Log Fields
ID	JobID, UID
Name	JobName, User
Resource	ReqCPUS, AllocCPUS, NNodes, NodeList
Time	Submit, Start, End
Job State	State

Meanwhile, we analyze the distribution of job failures in terms of time and resources, and refine the job submission time. Fig. 3 illustrates the distribution of jobs from Monday to Sunday. It can be seen that the total number of jobs submitted by users on working days (Monday to Friday) is much higher than that on rest days (Saturday and Sunday). In addition, the number of failed jobs on working days is maintained at a stable level, and

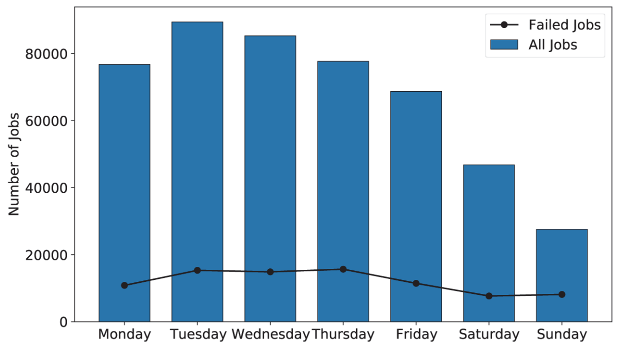


Fig. 3. The distribution of jobs from Monday to Sunday.

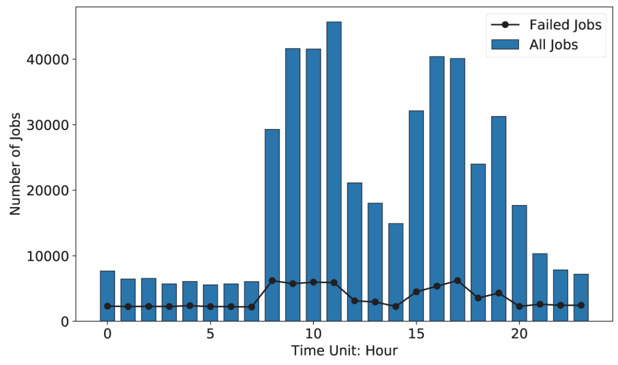


Fig. 4. The distribution of jobs in each hour of 24 hours.

Table 2. The distribution of jobs in single node and multiple nodes

Node Attribute	Job State	Number of Jobs	Failure Rate
single-node	COMPLETED	270,513	**12.48%**
	FAILED	38,579	
multi-node	COMPLETED	117,763	**27.82%**
	FAILED	45,396	

there is a significant downward trend in failed jobs adjacent to rest days. The number of failed jobs on rest days is the least. In the full job logs, the jobs submitted by users on the rest day tend to be more stable.

Fig. 4 illustrates the distribution of jobs per hour in 24 hours. The total number of jobs is less and changes steadily from 20:00 to 8:00 of the next day (non-working hours). The total number of jobs is more from 8:00 to 12:00 and from 15:00 to 20:00 respectively. The changing trend of failed jobs is similar to that of the total number of jobs, and the number of failed jobs is less at night and during work breaks (12:00 to 14:00). During this period, users may prefer to submit jobs that are not easy to fail and take a long time to execute.

To sum up, there are different patterns in the job distribution under different time units. Therefore, the submission time of the job will be used as an input feature of the prediction model.

In the resource attributes of jobs, the number of compute nodes reflects the scale of jobs and the complexity of parallel jobs. We divide the job categories into single-node jobs and multi-node jobs. Table 2 shows the job distribution under single-node and multi-node jobs. The job failure rate in single- node jobs is 12.48%, and the job failure rate in multi- node jobs is 27.82%. Most jobs in this institution are single-node jobs. The failure rate of multi-node jobs is higher than that of single-node jobs because jobs are executed in parallel on multiple computing nodes, and the communication and job behavior between nodes is more complex.

In the analysis phase, the attributes of Submit, NNodes, ReqCPUS and UID are retained as the input features of the prediction model. However, relying on these features alone is not enough. These attributes cannot more accurately describe the mode of job failure, therefore, we try to predict job failure from the semantic attributes of jobs.

V. SEMANTIC ENHANCEMENT

Users submit a large number of jobs in the supercomputer for their scientific research or project engineering. There are many of jobs with similar job names in the job logs. These jobs may come from the same research work of users. There are still jobs with similar job names among different users, and these users may belong to the same research group. Since the research work of users belongs to their intellectual property, we cannot accurately understand the research content of users' submitted jobs. Job name contains much potential information, such as data scale and project source. In this study, the job name is one of the semantic features of the job application.

In addition, we collect the job submission path (named Path) from the additional monitoring information, i.e., the jobs submitted by the user are executed in these directories, which contain more affluent job semantic information. It is found that the same job names in the job logs can come from different paths, i.e., the job path can enhance the application semantic information and subdivide the job application. Therefore, the path is enhanced semantic information to describe the job application.

Table 3. Constituent elements of job name and path

Element Category	Detail
Letter	A-Z, a-z
Special Character	-, , +, =, (,), ., /, >, < ...
Number	0-9

Table 4. Examples of similar job names and job paths

Job Name	Path
Web3 Http2	/home/Canyon/Tim/Vis-labor+2021/boot t/c62=d0
WEB3 HTTP2	/home/Canyon/Tim/Vis-labor+2018/boot t/c6=D0
WeB2 Http92	/home/Canyon/Tim/Vis-labor+19/boot t/C10=D0
WEB23 Http2	/home/Canyon/Tim/Vis-labor+20/boot t/c83=D0

Table 3 shows the constituent elements of job name and job path, which are composed of letters (L), special characters (C) and numbers (N). Table 4 shows similar job names and job paths. We need to cluster these job names and paths.

A. Job Name Clustering

As shown in Table 5, the composition mode of job names is mainly divided into three cases: letter mode,

Table 5. Composition mode of job names

Composition Mode	Detail
letter mode	L, L+C, L+N, L+C+N
special character mode	C,C+N
number mode	N

character mode and number mode. Generally, the priority of the semantic composition of strings is letters > numbers > special characters. However, when letters and numbers exist in the job name simultaneously, the number has less impact on the semantic value of the job name and is not as good as the spacing semantics of special characters. Therefore, to improve the clustering efficiency of job names and reduce the computational cost. We remove the redundant information of the job name according to the character priority: letters > special characters > numbers.

Considering that the case of letters does not have much difference in the semantics of the job name, all letters are converted to lowercase. For the job name composed of pure numbers, convert the job name to the length of the job name character. The retained information of the job name is shown in Table 6. Finally, the job names with the same information are divided into a group, and 4,309 job names are clustered into 771 job name groups.

B. Path Clustering

Although the constituent elements of the job path and job name are the same, and the composition mode only has the letter mode, the difference is that the job path has hierarchical information, and the composition length of the job path is much longer than that of job name, which means that job path contains more redundant semantic information and is difficult to cluster. Therefore, we only retain the letters and special characters in the job path and convert all letters to lowercase, which retains the job path's hierarchical information and retains the job path's primary semantic information. The retained information is shown in Table 7. Finally, job paths with the same retained information are divided into the same group, and 304,839 paths are clustered into 7,969 path groups.

According to the definition of job application in this paper 771 job name groups and 7,969 path groups get 8,773 job applications. Note that we use Python's built-in hash function to encode each job name group and path group as a feature of job failure prediction.

VI. EXPERIMENTATION AND RESULTS

A. Experiment

This paper simulates job failure prediction in a real production environment, i.e., continuously predict job failures according to time windows and observe the prediction effect. First, the collected job logs are divided into the initial training set and the total test set, and the relevant information of the initial training set and the total test set is shown in Table 8. Since it is a continuous job failure prediction, the initial training set is the total job data set of the first month, and the initial test set is the

Table 6. Example of job names after retaining information

Composition Mode	Original Job Name	Retained Information	Retained Job Name
letter mode	R0817-S25	L+C	r-s
special character mode	2021+23=2044	C	+=
number mode	19971231	N	8

Table 7. Example of paths after retaining information

Original Path	Retained Path
/home/Canyon/COMPuter Related W/Django012-S	/home/canyon/computer related w/django-s
/home/Clid/Tim/optimistic-things_23upper t	/home/clid/tim/optimistic-things upper t
/home/Eddie/First01-come/Running-here27	/home/eddie/first-come/running-here
/home/Han/Why?This/Pro20+Boy22	/home/han/why?this/pro+boy

Table 8. Statistics of initial training set and total test set

Initial Training Set	Total Test Set	Number of Failed Jobs	Failure Rate of Total Test Set
28,566	443,682	78,334	17.66%

first-day job data of the total test set. Then, keep the training set for one month (training time window) jobs and the test set for one day (testing time window) jobs, update the prediction training set and test set with this rule. Finally, comprehensively evaluate the prediction effect.

In the prediction process, we use six prediction features, i.e., Submit, NNodes, UID, ReqCPUS, JobName and Path, where JobName and Path belong to job application semantic enhancement features. Due to the imbalance between successful jobs and failed jobs, the model is more likely to predict jobs as successful jobs in the process of building the model. Therefore, the training set is processed by over sampling method to reduce the impact of the imbalance. In addition, the decision tree algorithm and random forest algorithm will be used to build models respectively, and the effects of job data under the non-ensemble algorithm and the ensemble algorithm will be compared.

B. Evaluation Indicator

Taking the failed jobs as the positive samples (P) of the prediction target and the successful jobs as the negative samples (N), we can get the following four situations as shown in Fig. 5, where TP indicates that the job will be correctly predicted as a failed job, TN indicates that the job will be correctly predicted as a successful job, FP indicates that the job is incorrectly predicted as a failed job, and FN indicates that the job is incorrectly predicted as a successful job.

In data science, the accuracy is usually used to evaluate the prediction effect, but the proportion of successful jobs in the test set reaches around 82.34% in the job failure prediction. Even if all predictions are successful, promising prediction effect can still be achieved. Therefore, we cannot only look at the accuracy of the research, but also introduce specificity and sensitivity (recall) to evaluate the ability to predict successful and failed jobs. The formulas of the three evaluation indicators are as follows:

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2}$$

$$specificity = \frac{TN}{TN+FP} = \frac{TN}{N} \tag{3}$$

$$sensitivity = recall = \frac{TP}{TP+FN} = \frac{TP}{P} \tag{4}$$

In order to evaluate the effect of continuous prediction as a whole, we use the weighted method of job proportion to sum the above three indicators. Their formulas are defined in (5), (6), (7), (8):

$$weight_{day} = \frac{job_{day}}{job_{total}}$$

$$N\_weight_{day} = \frac{N\_job_{day}}{N\_job_{total}} \tag{5}$$

$$P\_weight_{day} = \frac{P\_job_{day}}{P\_job_{total}}$$

$$accuracy_{final} = \sum_{i=1}^{n} accuracy_i \times weight_i \tag{6}$$

$$specificity_{final} = \sum_{i=1}^{n} specificity_i \times N\_weight_i \tag{7}$$

$$sensitivity_{final} = \sum_{i=1}^{n} sensitivity_i \times P\_weight_i \tag{8}$$

where $weight_{day}$ represents the quantity weight of the day's jobs in the total test set, $N\ weight_{day}$ represents the quantity weight of negative samples in the total test set, and $P\ weight_{day}$ represents the quantity weight of positive samples in the total test set.

Finally, we combine specificity and sensitivity, and use $S\ score$ as the final evaluation indicator, which is equivalent to the harmonic average of specificity and sensitivity. The formula is defined in (9).

$$S\_score_{\beta} = (1+\beta^2) \times \frac{specificity_{final} \times sensitivity_{final}}{(\beta^2 \times specificity_{final}) + sensitivity_{final}} \tag{9}$$

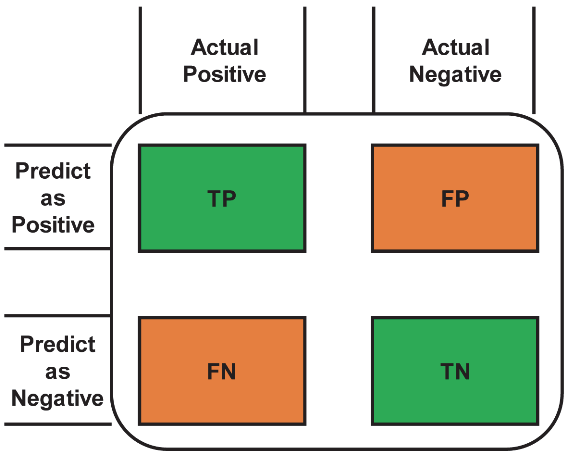


Fig. 5. Classification of prediction results.

When β is 1, the user thinks that the prediction ability of the model for successful jobs and failed jobs is equally important. When β is 2, the user is more likely to identify more failed jobs and find more possibilities of job failure.

C. Results

1) Before Semantic Enhancement:

We first predict job failure without applying semantic enhancement, i.e., only use the four prediction features of Submit,UID, NNodes and ReqCPUS. The prediction results under the two algorithms are shown in Table 9. The accuracy, specificity and sensitivity of DT are 80.45%, 88.2% and 80.96% respectively, and the accuracy, specificity and sensitivity of RF are 81.58%, 89.44% and 82.08% respectively.

Fig. 6 illustrates the comparison results under the *S score*. It can be seen that the prediction effect under the random forest is better than that of the decision tree, the *S* $score_1$ is increased by 1.17%, and the *S* $score_2$ is increased by 1.14%.

2) After Semantic Enhancement:

We apply the condition of semantic enhancement to predict job failure. Firstly, the JobName is added into the model, and the prediction effect is shown in Table 10. The accuracy, specificity and sensitivity of DT are 81.07%, 88.74% and 81.08% respectively, and the accuracy, specificity and sensitivity of RF are 82.91%, 90.7% and 82.91% respectively.

Fig. 7 illustrates the comparison results under the *S score*. Compare with the *S score* before semantic enhancement, the *S* $score_1$ of DT and RF are increased by 0.31% and 1.03% respectively, and the *S* $score_2$ of DT and RF are increased by 0.19% and 0.91% respectively.

On the basis of the JobName, then add the Path feature into the model. The prediction effect is shown in Table 11. The accuracy, specificity and sensitivity of DT are 86.37%, 93.59% and 86.45% respectively, and the accuracy, specificity and sensitivity of RF are 88.16%, 95.23% and 88.24% respectively.

Fig. 8 illustrates the comparison results under the *S score*. Compared with before semantic enhancement, the *S* $score_1$ of DT and RF are increased by 5.45% and 6% respectively, and the *S* $score_2$ of DT and RF are increased by 5.48% and 6.1% respectively.

From the experimental results, the effect of job application semantic enhancement on job failure prediction is a promising improvement.

Table 9. The prediction results of the two algorithms before semantic enhancement

Algorithm	Accuracy	Specificity	Sensitivity
DT	80.45%	88.2%	80.96%
RF	81.58%	89.44%	82.08%

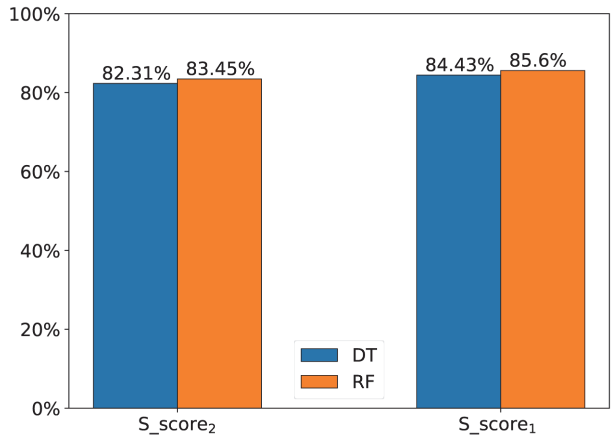


Fig. 6. Results of each algorithm under two S scores (no semantic enhancement).

Table 10. The prediction results of the two algorithms after adding job name semantics

Algorithm	Accuracy	Specificity	Sensitivity
DT	81.07%	88.74%	81.08%
RF	82.91%	90.7%	82.91%

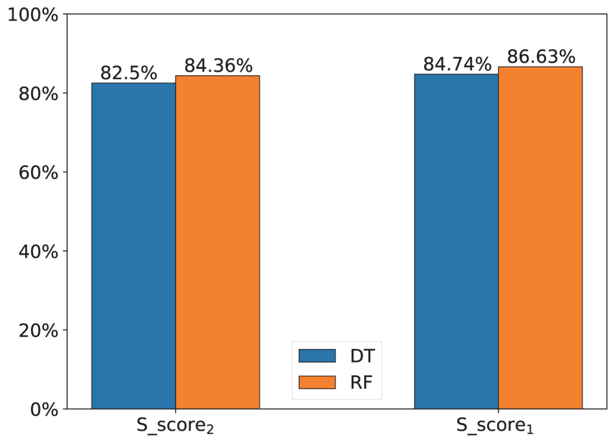


Fig. 7. Results of each algorithm under two S scores (semantic enhancement with JobName).

Table 11. The prediction results of the two algorithms after the job application semantic enhancement

Algorithm	Accuracy	Specificity	Sensitivity
DT	86.37%	93.59%	86.45%
RF	88.16%	95.23%	88.24%

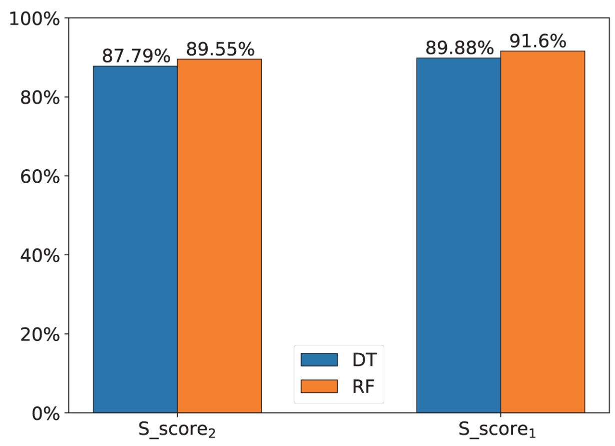


Fig. 8. Results of each algorithm under two S scores (semantic enhancement with JobName and Path).

VII. DISCUSSION

In the related work of job failure prediction, dynamic collection of job resource characteristics is mainly used to predict job failure. There are certain differences in the resource characteristics of each supercomputer system. These characteristics change greatly, can not effectively reflect the application characteristics of the job, and have low expansibility. From the perspective of job semantic information, this study uses multiple innovative feature job names and job paths to enhance the semantic information of jobs, and verifies the effectiveness of prediction through experiments. Semantic enhancement can indeed improve the prediction effect, especially the use of job paths. Meanwhile, RF ensemble learning method is used to further improve the prediction effect. Compared with using resource characteristics, the job failure prediction method using semantic information has less computational overhead and is easy to be applied to various supercomputers.

VIII. CONCLUSION

Job failure prediction is an important means to improve the efficiency of a supercomputer. It can help users make decisions on jobs and reduce the waste of system resources as much as possible. This paper proposes a job failure prediction method based on job application semantic enhancement, which uses *S score* as the final comprehensive evaluation indicator. In addition to using the traditional characteristics of job time and resources, this method can also predict job failure from semantic enhancement to effectively improve the prediction effect. The final *S score* can be improved by about 5%-6% compared with before semantic enhancement. Meanwhile, the prediction effects of the non-ensemble learning algorithm and ensemble learning algorithm are compared. The experimental results show that the ensemble learning algorithm can improve the prediction effect and finally achieve 88.16% accuracy with 95.23% specificity and 88.24% sensitivity.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (Grant No. 62202471), and Sichuan Science and Technology Program, No: 2022YFG0040.

CONFLICTS OF INTEREST

The authors declare that there are no conflicts of interest.

REFERENCES

1. P. Uthayopas, T. Angskun, and J. Maneesilp, “Build- ing a parallel computer from cheap pcs: Smile clus- ter experiences,” in *Proceedings of the Second Annual National Symposium on Computational Science and Engineering, Bangkok, Thailand*. Citeseer, 1998.
2. X. Yang, X. Liao, K. Lu, Q. Hu, J. Song, and J. Su, “The tianhe-1a supercomputer: its hardware and software,” *Journal of computer science and technology*, vol. 26, no. 3, pp. 344-351, 2011.
3. J. Kunkel and M. F. Dolz, “Understanding hardware and software metrics with respect to power consumption,” *Sustainable Computing: Informatics and Systems*, vol. 17, pp. 43-54, 2018.
4. M. Soualhia, F. Khomh, and S. Tahar, “Predicting scheduling failures in the cloud: A case study with google clusters and hadoop on amazon emr,” IEEE. 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems, 2015, pp. 58-65.
5. D. Grzonka, A. Jakobik, J. Ko-lodziej, and S. Pllana, “Using a multi-agent system and artificial intelligence for monitoring and improving the cloud performance and security,” *Future Generation Computer Systems*, vol. 86, pp. 1106-1117, 2018.
6. J. Elliott, K. Kharbas, D. Fiala, F. Mueller, K. Ferreira, and C. Engelmann, “Combining partial redundancy and checkpointing for hpc,” in *2012 IEEE 32nd International Conference on Distributed Computing Systems*. IEEE, 2012, pp. 615-626.
7. P. Garraghan, P. Townend, and J. Xu, “An empirical failure-analysis of a large-scale cloud computing environment,” in *2014 IEEE 15th International Symposium on High-*

Assurance Systems Engineering. IEEE, 2014, pp. 113-120.

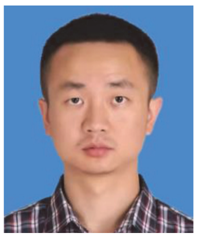
8. B. Mohammed, M. Kiran, K. M. Maiyama, M. M. Kamala, and I.-U. Awan, "Failover strategy for fault tolerance in cloud computing environment," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1243-1274, 2017.
9. X. Chen, C.-D. Lu, and K. Pattabiraman, "Failure prediction of jobs in compute clouds: A google cluster case study," in *2014 IEEE International Symposium on Software Reliability Engineering Workshops*. IEEE, 2014, pp. 341-346.
10. A. Rosa, L. Y. Chen, and W. Binder, "Predicting and mitigating jobs failures in big data clusters," in *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 2015, pp. 221-230.
11. F. J. A. Morais, F. V. Brasileiro, R. V. Lopes, R. A. Santos, W. Satterfield, and L. Rosa, "Autoflex: Service agnostic auto-scaling framework for iaas deployment models," in *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*. IEEE, 2013, pp. 42-49.
12. Y. Zhao, X. Liu, S. Gan, and W. Zheng, "Predicting disk failures with hmm-and hsmm- based approaches," in *Industrial Conference on Data Mining*. Springer, 2010, pp. 390-404.
13. T. Gentner, K. P. Gungl, and G. Hutzl, "Integrated circuit failure prediction using clock duty cycle recording and analysis," Apr. 19 2016, uS Patent 9,319,030.
14. R. Jayanthi and L. Florence, "Software defect prediction techniques using metrics based on neural network classifier," *Cluster Computing*, vol. 22, no. 1, pp. 77-88, 2019.
15. K. Kumaresan and P. Ganeshkumar, "Software reliability modeling using increased failure interval with ann," *Cluster Computing*, vol. 22, no. 2, pp. 3095-3102, 2019.
16. T. Samak, D. Gunter, M. Goode, E. Deelman, G. Juve, F. Silva, and K. Vahi, "Failure analysis of distributed scientific workflows executing in the cloud," in *2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)*. IEEE, 2012, pp. 46-54.
17. N. El-Sayed, H. Zhu, and B. Schroeder, "Learning from failure across multiple clusters: A trace- driven approach to understanding, predicting, and mitigating job terminations," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 1333-1344.
18. W. Yoo, A. Sim, and K. Wu, "Machine learning based job status prediction in scientific clusters," in *2016 SAI Computing Conference (SAI)*. IEEE, 2016, pp. 44-53.
19. A. Banjongkan, W. Pongsena, N. Kerdprasop, and K. Kerdprasop, "A study of job failure prediction at job submit-state and job start-state in high- performance computing system: Using decision tree algorithms," *Journal of Advances in Information Technology*, vol. 12, no. 2, 2021.
20. S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660-674, 1991.
21. B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *Journal of Applied Science and Technology Trends*, vol. 2, no. 01, pp. 20-28, 2021.
22. Z. Zhou, "Ensemble learning," in *Machine learning*. Springer, 2021, pp. 181-210.
23. G. Louppe, "Understanding random forests: From theory to practice," *arXiv preprint arXiv:1407.7502*, 2014.
24. A. Antoniadis, S. Lambert-Lacroix, and J.-M. Poggi, "Random forests for global sensitivity analysis: A selective review," *Reliability Engineering & System Safety*, vol. 206, p. 107312, 2021.
25. S. Cherala, "Implementing and evaluating multi- resource scheduling in slurm," Ph.D. dissertation, Illinois Institute of technology, 2021.
26. J. Yu, W. Yang, F. Wang, D. Dong, J. Feng, and Y. Li, "Spatially bursty i/o on supercomputers: Causes, impacts and solutions," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 12, pp. 2908-2922, 2020.

Haotong Zhang

Haotong Zhang received his B.S. degree in Shenyang Ligong University, China. He is currently pursuing the M.S. degree in Chongqing Jiaotong University, China. His current research interests include artificial intelligence and computational intelligence.

Gang Xian

Gang Xian received his B.S. degree in Southwest University of Science and Technology, China. He is currently pursuing the M.S. degree in Southwest University of Science and Technology, China. His current research interests include system abnormality detection and artificial intelligence.



Wenxiang Yang

Wenxiang Yang received the B.S., M.S., and Ph.D. degrees from the National University of Defense Technology (NUDT), China. He is currently a research associate at the Computational Aerodynamics Institute, China Aerodynamics Research and Development Center (CARDC). His current research interests are high performance computing, I/O system and storage system.

Jie Yu

Jie Yu received the B.S., M.S., and Ph.D. degrees from the National University of Defense Technology (NUDT), China. He is currently a research associate at the Computational Aerodynamics Institute, China Aerodynamics Research and Development Center (CARDC). His current research interests are high performance I/O, parallel file systems and system monitoring and diagnosis.