Sensitivity Analysis for Deep Learning Data Sets Considering the Human Immune System

Haruki Takeda[1,*], Shoichiro Miyamoto[1], Lei Zhou[1], Yoshinobu Tamura[1], and Shigeru Yamada[2]

[1]*Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan*
[2]*Department of Social Management Engineering, Graduate School of Engineering, Tottori University, Tottori, Japan*
E-mail: e038vgv@yamaguchi-u.ac.jp; d009wcu@yamaguchi-u.ac.jp; zhou-lei@yamaguchi-u.ac.jp; tamuray@yamaguchi-u.ac.jp; bex2yama@muse.ocn.ne.jp
[*]*Corresponding Author*

Received 21 June 2024; Accepted 29 September 2024

Abstract

Recently, open source software (OSS) has been widely used in many fields due to the spread and development facilitated by networks. The characteristics of OSS include no cost and high performance, which make it a significant component of modern society. However, the number of reported faults is increasing due to its vulnerabilities. Detecting these faults requires substantial costs, and correcting the growing number of faults necessitates a large workforce. In this paper, we propose a method for Reliability Assessment of OSS using deep learning based on the human immune system. Additionally, we present several numerical examples based on the proposed method.

Keywords: Reliability assessment, deep learning, open source software.

Journal of Graphic Era University, Vol. 12_2, 309–328.
doi: 10.13052/jgeu0975-1416.1228

1 Introduction

Open source software (OSS) is available for anyone to use, modify, and redistribute. Therefore, it can achieve a high level of quality through continuous improvements made by various contributors. Moreover, the software system facilitates easy development with minimal effort due to the use of OSS. As a result, the number of OSS users is increasing in modern society, whether they are individuals or companies. However, it is challenging to modify OSS in response to the growing number of users. Currently, many developers and users are involved in fault modification. Despite these efforts, new faults are continually introduced into OSS. Therefore, optimizing the effort required for fault modification is essential for resolving these issues.

In this paper, we demonstrate numerical examples of fault big data using deep learning to estimate fault modification time, aiming to assess the reliability of OSS. By estimating fault modification time, it becomes possible to optimize the effort required for fault modification. To obtain more accurate results, we adjust the ratio of training data to testing data for deep learning and verify the validity of the proposed method.

2 Previous Research

2.1 OSS

OSS is software that is freely available on the Internet at no cost to anyone. OSS is improved by contributions from many people, and it has evolved into high-performance software. As a result, OSS with a large number of users is modified more frequently, which helps ensure a certain level of reliability. Additionally, the reliability of OSS can be verified by users themselves because the source code is publicly available. For these reasons, OSS has become entrenched in modern society as a cost-effective and high-performance software solution. Representative OSS used in modern society includes Java, Python, OpenStack, Linux, and MySQL, which are employed in various domains such as operating systems, development languages, and databases [1, 2, 3, 4].

However, there are some disadvantages, such as the difficulty in promptly correcting problems when they occur and the vulnerability to attacks by malicious individuals targeting OSS users. Furthermore, the large number of OSS projects and their users makes it challenging to grasp the overall situation.

2.2 Cloud Computing

Cloud computing [5, 6, 7] represents a new paradigm of computing services, including networks and databases, delivered over the Internet. Traditionally, system construction required substantial cost and preparation, but cloud computing enables users to access the necessary environment for system development over the Internet, leading to significant reductions in cost and time. Additionally, since users do not need to own their own servers, they can create the optimal environment for themselves by using only the required resources. Cloud computing is utilized across a wide range of fields by both companies and individuals. It allows necessary information and environments to be stored in the cloud and accessed from any device, facilitating information sharing between companies. On the other hand, cloud computing is a service that has already been completed, which means it may not be suitable for highly detailed or customized settings.

2.3 Reliability

Software may not function properly or may not work at all due to errors during development or unforeseen issues. Software reliability [8, 9] refers to the ability of software to operate correctly in any environment without such defects. Software quality and reliability improve through repeated assessments and modifications. Evaluating the frequency of software faults is known as reliability assessment.

2.4 Software Reliability Growth Model

The software reliability growth model [10, 11, 12] is a stochastic model based on software faults. During software development, faults are detected and corrected through testing. Therefore, testing time is crucial for assessing software reliability. The software reliability growth model can visualize these events and estimate the number of software faults, which is useful for predicting the optimal timing for software release.

2.5 Deep Learning

Deep learning [13, 14] is a type of machine learning method that utilizes deep neural networks. In deep learning, learning is performed by adding multiple intermediate layers to the neural network. In conventional machine learning, feature extraction—which quantifies the characteristics of input data—is

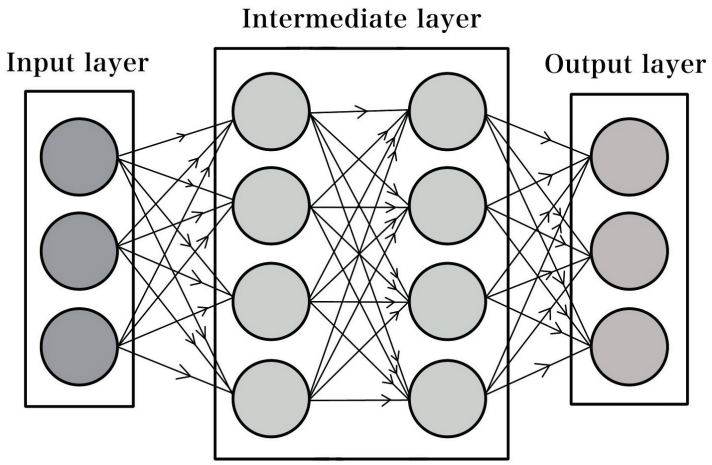

Figure 1 Deep neural network.

performed by humans. In contrast, deep learning automatically extracts and learns features. This process improves learning accuracy because the features are optimized. Additionally, increasing the amount of input data allows for the construction of more accurate models.

2.6 Neural Network

The neural network is a mathematical model that mimics the network structure of the brain. It mainly consists of an input layer, one or more intermediate layers, and an output layer, and it can learn from input data. Conventional neural networks typically have three layers—one for each type—while networks with four or more layers used to suffer from decreased accuracy. However, this issue has been addressed in deep learning, which supports deep neural networks with four or more layers, as shown in Figure 1. Moreover, recent advances in handling big data have made it possible to construct even more powerful deep neural networks.

3 Proposed Method

3.1 Self-Modification Approach Based on Human Immune System

Our research group has been working on cloud edge computing, big data, OSS, deep learning, and stochastic models. Deep learning and stochastic

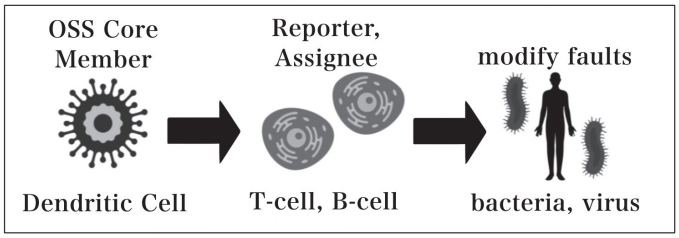

Figure 2 An example of human immune system.

models with statistical independence can address problems that cannot be solved by data-driven approaches alone, as these models often function as black boxes. Specifically, many efforts have been made to mimic mechanisms from the natural world, such as neural networks inspired by the brain's structure, genetic algorithms based on evolutionary principles, and drones modeled after insect functions. These approaches can also solve fundamental issues related to self-modification.

In this method, each cell in the human immune system mimics the actions that individual cells take against pathogenic bacteria and viruses, while the fault modifications and countermeasures are carried out using the same procedures. Figure 2 illustrates a schematic diagram of this research. Most existing studies focus solely on cloud environments, with no methods proposed to assess and optimize operational performance by simultaneously considering edge computing and data structures. By integrating data-driven and biological approaches from different perspectives, as shown in Figure 2, it is possible to predict the reliability of complex network environments in cloud-edge infrastructures and support self-modification through AI.

3.2 Learning Method

In this paper, we focus on fault modification time and fault reporters. We consider a system that predicts changes in fault modification time by observing and learning the conditions under which faults are detected and modified. Specifically, we examine the following weighted modification time.

$$F(i) = (c_i - o_i)\gamma_i, \tag{1}$$

where c_i is the modification completion time for the i-th fault. o_i is the detected time for the i-th fault. In addition, γ_i is the i-th fault severity. In

this case, $(c_i - o_i)$ is the fault modification time. Therefore, $F(i)$ represents the weighted fault modification time for the i-th fault severity. The value of $F(i)$ is large, when the reliability becomes large.

4 Numerical Examples

In this paper, we used the fault big data obtained from the $OpenStack$ Project [15]. We present several numerical examples of two cases. one is which 90% of the fault data was used as training data and the remaining 10% as testing data, and the other is which 80% of the fault data was used as training data and the remaining 20% as testing data. The following parameters were used as the training data of deep learning.

- Fault detection time
- Fault modification completion time
- Fault reporter
- Software product
- Software component
- Fault condition
- Fault resolution Status
- Hardware type
- Operation system type
- Fault severity
- Software version
- Fault Summary

4.1 Percentage of Training Data: 90%

First, we analyzed the data using deep learning with 90% of the data for training and 10% for testing. Figures 3, 4, 5, 6, and 7 present the estimated error for validation and training data, the estimated weighted fault modification time for fault severity, the estimated scatter plot of weighted fault modification time, the estimated weighted cumulative fault modification time, and the estimated scatter plot of weighted cumulative fault modification time.

4.2 Percentage of Training Data: 80%

Next, we conducted an analysis using deep learning with 80% of the data for training and 20% for testing. Figures 8, 9, 10, 11, and 12 show the estimated error for validation and training data, the estimated weighted fault

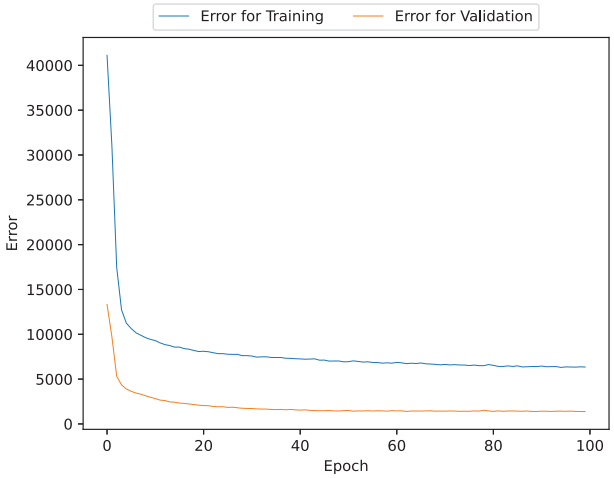

Figure 3 The estimated error for validation and training data (In the case of 90% of training data).

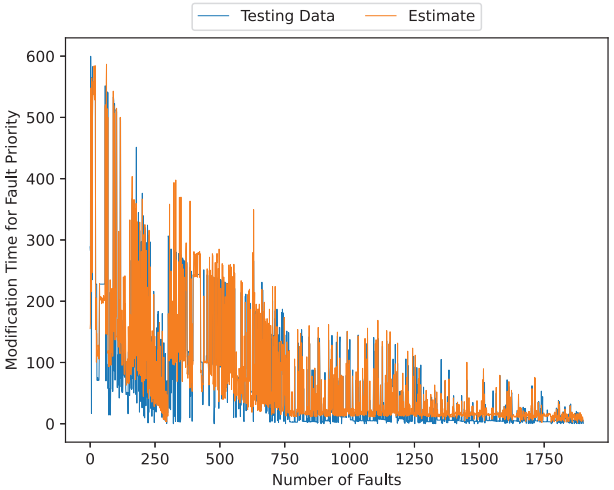

Figure 4 The estimated weighted fault modification time for fault severity (In the case of 90% of training data).

modification time for fault severity, the estimated scatter plot of weighted fault modification time, the estimated weighted cumulative fault modification time, and the estimated scatter plot of weighted cumulative fault modification time.

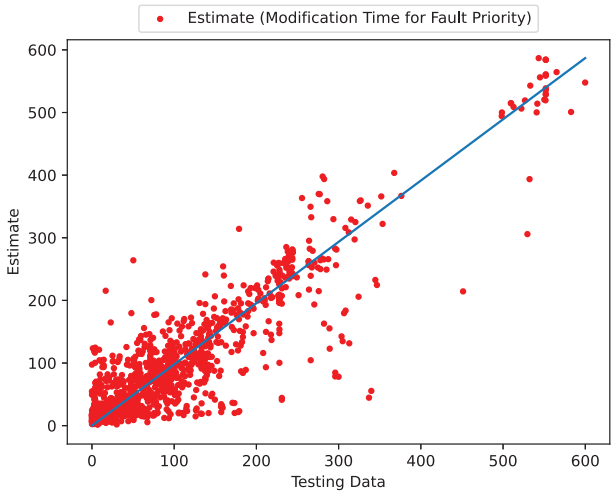

Figure 5 The estimated scatter plot of weighted fault modification time (In the case of 90% of training data).

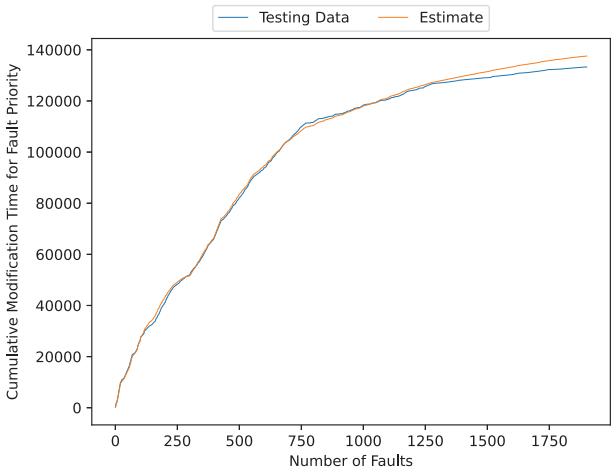

Figure 6 The estimated weighted cumulative fault modification time (In the case of 90% of training data).

4.3 Percentage of Training Data: 70%

Finally, we analyzed the data using deep learning with 70% of the data for training and 30% for testing. Figures 13, 14, 15, 16, and 17 display the estimated error for validation and training data, the estimated weighted fault

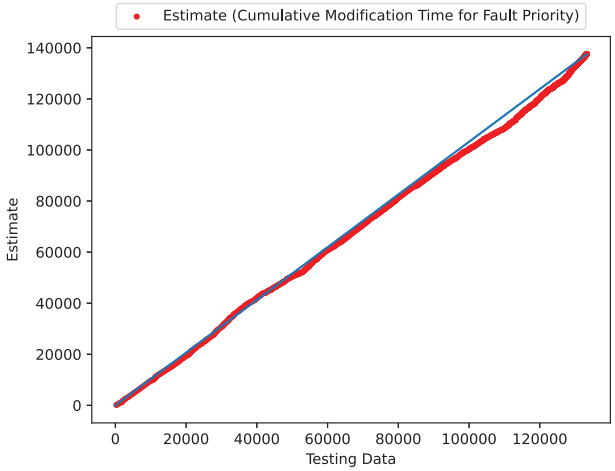

Figure 7 The estimated scatter plot of weighted cumulative fault modification time (In the case of 90% of training data).

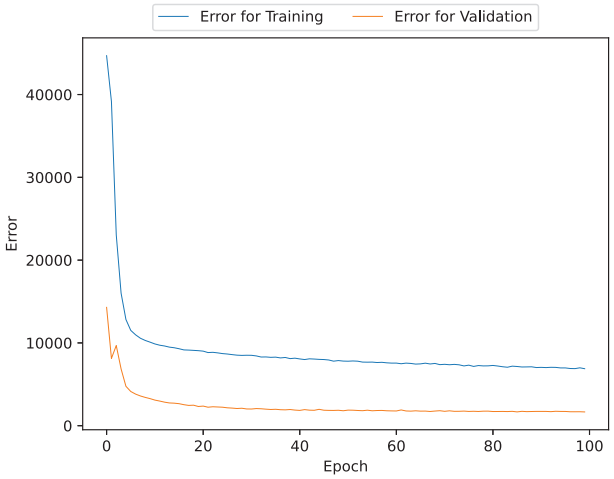

Figure 8 The estimated error for validation and training data (In the case of 80% of training data).

modification time for fault severity, the estimated scatter plot of weighted fault modification time, the estimated weighted cumulative fault modification time, and the estimated scatter plot of weighted cumulative fault modification time.

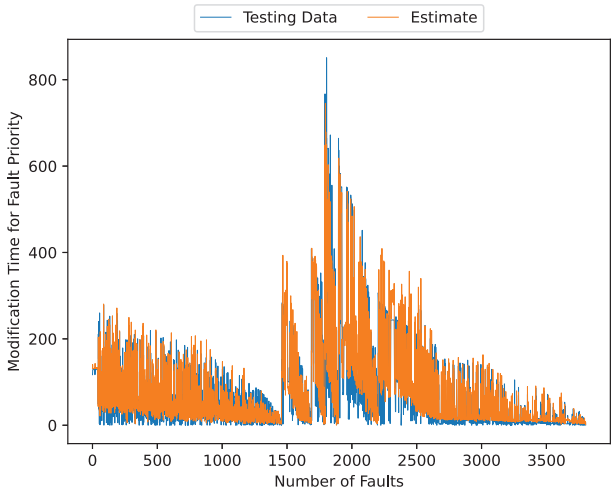

Figure 9 The estimated weighted fault modification time for fault severity (In the case of 80% of training data).

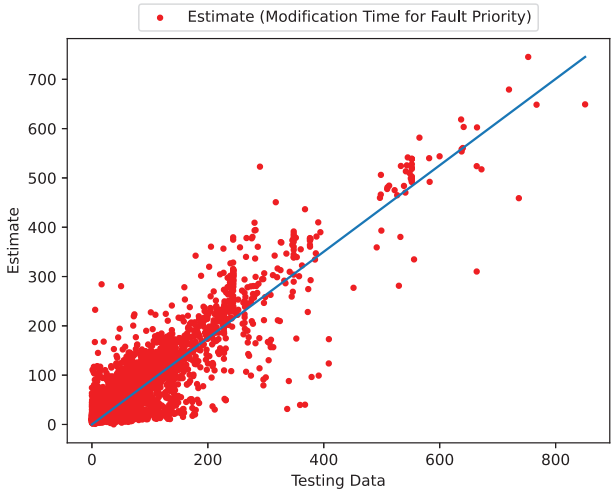

Figure 10 The estimated scatter plot of weighted fault modification time (In the case of 80% of training data).

4.4 Discussion

In the cases of using 90% and 80% of the data for training, the estimated results are favorable. From these results, we have found that the estimates

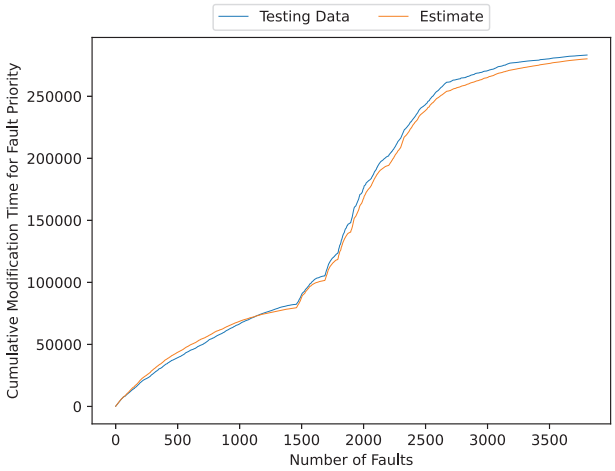

Figure 11 The estimated weighted cumulative fault modification time (In the case of 80% of training data).

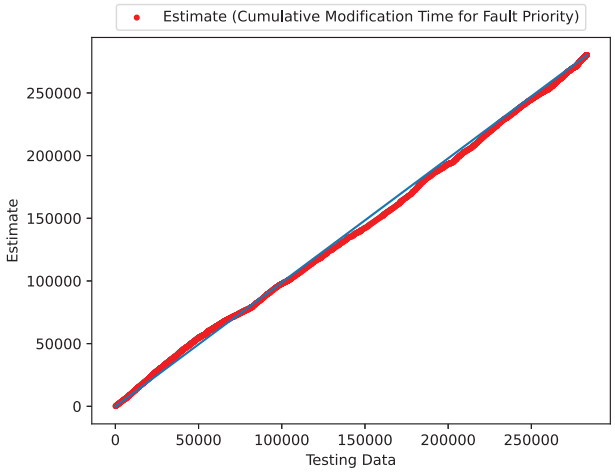

Figure 12 The estimated scatter plot of weighted cumulative fault modification time (In the case of 80% of training data).

improve when the percentage of training data is set at 90% and 80%. In contrast, the estimates with 70% of the training data deviate significantly from the testing data.

We have observed that even over extended periods, using 80% or more of the training data can lead to accurate predictions of the number of faults.

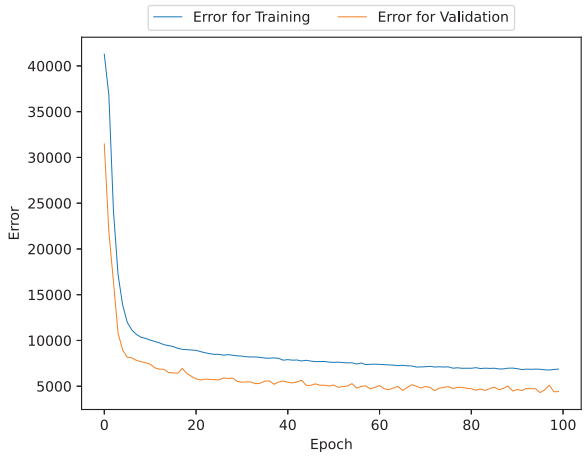

Figure 13 The estimated error for validation and training data (In the case of 70% of training data).

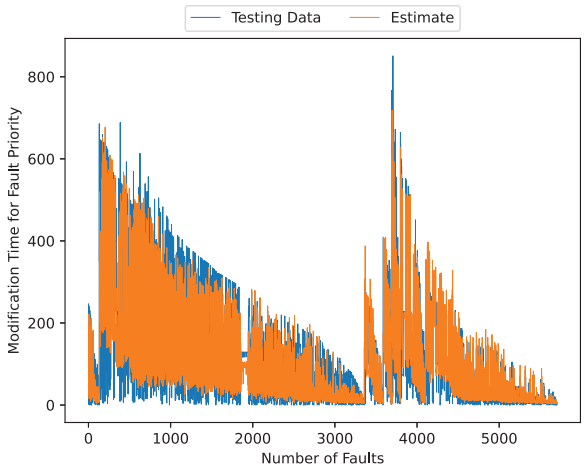

Figure 14 The estimated weighted fault modification time for fault severity (In the case of 70% of training data).

Moreover, Table 1 presents the mean square error (MSE) for the estimated fault modification time. From this table, we found that the MSE value decreases as the percentage of training data increases. Table 2 shows the estimated MSE of existing studies. These results show that the method in this paper is capable of estimating with significantly higher accuracy than the methods in existing studies.

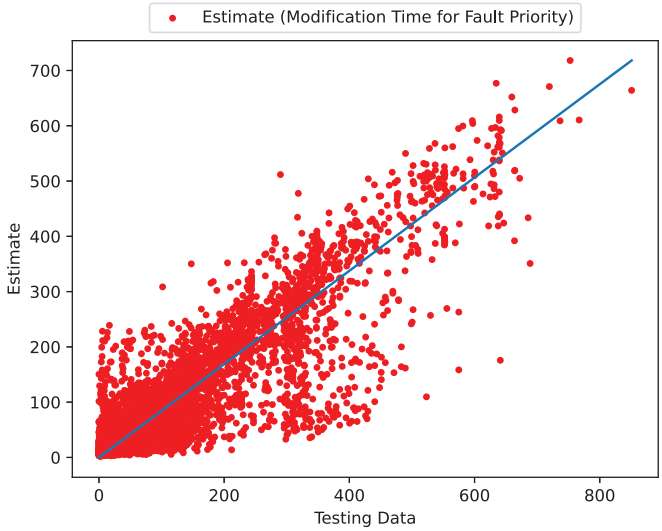

Figure 15 The estimated scatter plot of weighted fault modification time (In the case of 70% of training data).

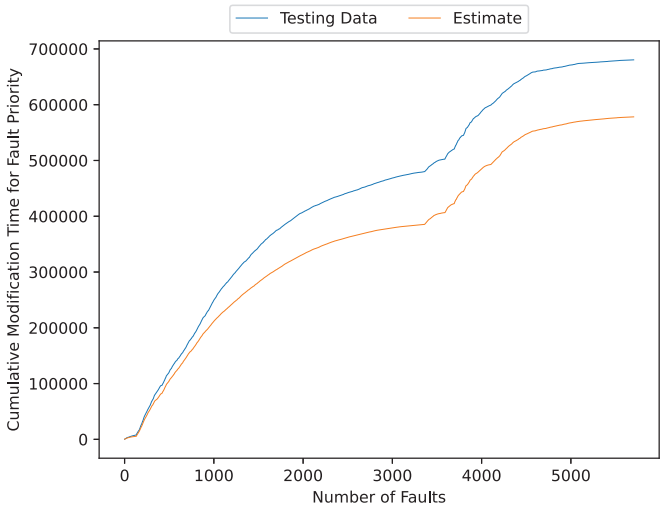

Figure 16 The estimated weighted cumulative fault modification time (In the case of 70% of training data).

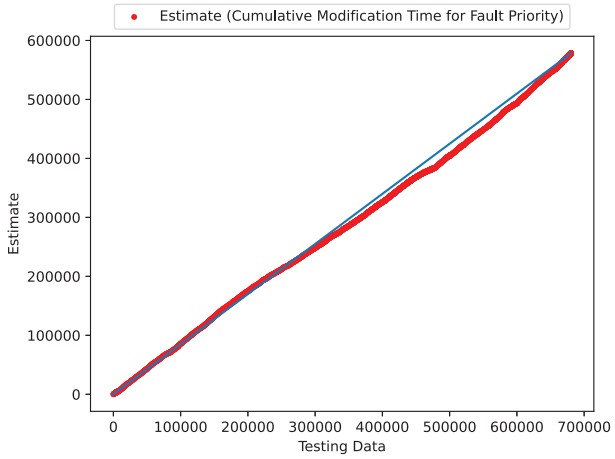

Figure 17 The estimated scatter plot of weighted cumulative fault modification time (In the case of 70% of training data).

Table 1 The estimated MSE for the fault modification time

	MSE
Percentage of Training Data: 90%	2206683.8
Percentage of Training Data: 80%	244221180.0
Percentage of Training Data: 70%	6723100000.0

Table 2 The estimated MSE of existing studies

	MSE
Percentage of Training Data: 90%	2071716900.0
Percentage of Training Data: 80%	4595891700.0
Percentage of Training Data: 70%	134495720000.0

5 Concluding Remarks

In this paper, we proposed a method for reliability assessment using deep learning based on the human immune system. In the proposed model, the fault modification time is estimated and visualized using fault big data as training data. The results indicate that it is possible to estimate the fault modification time with high accuracy, despite some errors. This capability allows for the optimization of personnel required for fault modification, potentially alleviating issues related to effort shortages. By applying this method, it becomes possible to create an application that allows anyone to predict various outcomes using complex big data.

In the future, we aim to use it in fields that require even more accurate predictions, such as the medical field, and to conduct research focused on improving estimation accuracy by optimizing features and variables.

Acknowledgments

This work was supported in part by the JSPS KAKENHI Grant No. 23K11066 in Japan.

References

[1] S. Yamada and Y. Tamura, *OSS reliability measurement and assessment*, Springer International Publishing Switzerland, 2016.

[2] Y. Tamura and S. Yamada, "AIR application for reliability analysis considering debugging process and network traffic in mobile clouds," *Simulation Modelling Practice and Theory*, pp. 165–175, doi: 10.1016/j.simpat.2014.03.010, 2015.

[3] S. Miyamoto, Y. Tamura and S. Yamada, "A Method of Reliability Assessment Based on Trend Analysis for Open Source Software.," *International Journal of Reliability, Quality & Safety Engineering*, Vol. 31, No. 4, pp. 1–15, doi: 10.1142/S0218539324500049, 2024.

[4] S. Miyamoto, Y. Tamura and S. Yamada, "A Method of OSS Reliability Assessment Based on Public Repository Analysis.," *International Journal of Reliability, Quality & Safety Engineering*, Vol. 30, No. 5, pp. 1–12, doi: 10.1142/S0218539323500213, 2023.

[5] M. O. Ozcan, F. Odaci, and I. Ari, "Remote Debugging for Containerized Applications in Edge Computing Environments, " *Proceedings of the 2019 IEEE International Conference on Edge Computing (EDGE)*, Milan, Italy, pp. 30–32, doi: 10.1109/EDGE.2019.00021, 2019.

[6] A. A. Ahmad et al., "Scalability analysis comparisons of cloud-based software services, " *Journal of Cloud Computing: Advances, Systems and Applications*, 10.1186/s13677-019-0134-y, 23 Jul. 2019.

[7] Y. Ngoko and C. Crin, An Edge Computing Platform for the Detection of Acoustic Events, in *Proc. 2017 IEEE Int. Conf. Edge Computing*, Honolulu, HI, pp. 240–243, doi: 10.1109/IEEE.EDGE.2017.44, 2017.

[8] M. R. Lyu, ed., *Handbook of Software Reliability Engineering*, IEEE Computer Society Press, Los Alamitos, CA, 1996.

[9] C. V. Ramamoorthy and F. B. Bastani, "Software Reliability-Status and Perspectives, " *IEEE Transactions on Software Engineering 4*, pp. 354–371, 1982.
[10] M. R. Lyu, ed, *Handbook of Software Reliability Engineering*, IEEE Computer Society Press, Los Alamitos, CA, 1996.
[11] S. Yamada, *Software Reliability Modeling: Fundamentals and Applications*, Springer-Verlag, Tokyo/Heidelberg, 2014.
[12] Y. Tamura and S. Yamada, "Deep Learning Based on Fine Tuning with Application to the Reliability Assessment of Similar Open Source Software, " *International Journal of Mathematical, Engineering and Management Sciences*, Vol. 8, No. 4, pp. 632–639, 2023.
[13] Y. Tamura and S. Yamada, "Prototype of 3D Reliability Assessment Tool Based on Deep Learning for Edge OSS Computing, " *Mathematics*, Vol. 10, No 9, pp. 1–12, doi: 10.3390/math10091572, 2022.
[14] H. P. Martinez, Y. Bengio, and G. N. Yannakakis, "Learning deep physiological models of affect," *IEEE Computational Intelligence Magazine*, Vol. 8, No. 2, pp. 20—33, 2013.
[15] The OpenStack project, Build the future of Open Infrastructure, https://www.openstack.org

Biographies

Haruki Takeda received the B.S.E. degrees from Yamaguchi University in Japan, in 2024. Since 2024, he has been entered at Yamaguchi University Graduate School, Yamaguchi, Japan. His research interests the reliability assessment method of OSS at Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan.

Shoichiro Miyamoto received the B.S.E. and M.S. degrees from Yamaguchi University in Japan, in 2022 and 2023, respectively. Since 2022, he has been entered at Yamaguchi University Graduate School, Yamaguchi, Japan. His research interests the reliability assessment method of OSS at Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan.

Lei Zhou received the M.S. degree from Nanjing Tech University in China, and Ph.D. degree from Tokyo Metropolitan University in 2020, respectively. Since 2021, she has been working as an Assistant Professor at the Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan. Her research interests the design and maintenance policies for linear consecutive-k-out-of-n: G systems at Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan.

Yoshinobu Tamura received the B.S.E., M.S., and Ph.D. degrees from Tottori University in 1998, 2000, and 2003, respectively. From 2003 to 2006, he was a Research Assistant at Tottori University of Environmental Studies. From 2006 to 2009, he was a Lecturer and Associate Professor at Faculty of Applied Information Science of Hiroshima Institute of Technology, Hiroshima, Japan. From 2009 to 2017, he was an Associate Professor at the Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan. From 2017 to 2019, he has been working as a Professor at the Faculty of Knowledge Engineering, Tokyo City University, Tokyo, Japan. Since 2020, he has been working as a Professor at the Faculty of Information Technology, Tokyo City University, Tokyo, Japan. Since 2021, he has been working as a Professor at the Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan. His research interests include reliability assessment for open source software, big data, clouds, reliability. He is a regular member of the Institute of Electronics, the Information and Communication Engineers of Japan, the Operations Research Society of Japan, the Society of Project Management of Japan, the Reliability Engineering Association of Japan, and the IEEE. He has authored the book entitled as OSS Reliability Measurement and Assessment (Springer International Publishing, 2016). Dr. Tamura received the Presentation Award of the Seventh International Conference on Industrial Management in 2004, the IEEE Reliability Society Japan Chapter Awards in 2007, the Research Leadership Award in Area of Reliability from the ICRITO in 2010, the Best Paper Award of the IEEE International Conference on Industrial Engineering and Engineering Management in 2012, the Honorary Professor from Amity University of India in 2017, the Best Paper Award of the 24th ISSAT International Conference on Reliability and Quality in Design in 2018, the Outstanding Paper Award of the IEEE International Conference on Industrial Engineering and Engineering Management in 2022, and the Amity Global Academic Excellence Award of the IEEE 4th International Conference on Intelligent, Engineering & Management in 2023.

Shigeru Yamada was born in Hiroshima Prefecture, Japan, on July 6, 1952. He received the B.S.E., M.S., and Ph.D. degrees from Hiroshima University, Japan, in 1975, 1977, and 1985, respectively. From 1993/10 to 2018/3, he had been working as a professor at the Department of Social Management Engineering, Graduate School of Engineering, Tottori University, Tottori-shi, Japan. He is an Emeritus Professor of Tottori University. He has been also a Honorary Professor at Amity University,India, since 2015.His research interests include software reliability engineering,quality management engineering, and project management. He has published over 600 reviewed technical papers in the area of software reliability engineering, project management, reliability engineering, and quality control. He has authored several books entitled such as Introduction to Software Management Model (Kyoritsu Shuppan,1993), Software Reliability Models: Fundamentals and Applications (JUSE, Tokyo, 1994), Statistical Quality Control for TQM (Corona Publishing, Tokyo, 1998), Software Reliability: Model, Tool, Management (The Society of Project Management, 2004), Quality-Oriented Software Management (Morikita Shuppan, 2007), Elements of Software Reliability–Modeling Approach – (Kyoritsu Shuppan, 2011), Project Management (Kyoritsu Shuppan, 2012), Software Engineering – Fundamentals and Applications – (Science, Tokyo, 2013), Software Reliability Modeling: Fundamentals and Applications (Springer-Verlag, Tokyo/Heidelberg, 2014), and OSS Reliability Measurement and Assessment (Springer International Publishing, Switzerland, 2016). Dr. Yamada received the Best Author Award from the Information Processing Society of Japan in 1992, the TELECOM System Technology Award from the Telecommunications Advancement Foundation in 1993, the Best Paper Award from the Reliability Engineering Association of Japan in 1999, the International Leadership Award in Reliability Engg. Research from the ICQRIT/SREQOM in 2003, the Best Paper Award at the 2004 International Computer Symposium, the Best Paper Award from the Society of Project Management in 2006, the Leadership Award from the ISSAT (International Society of Science and Applied Technologies,

U.S.A.) in 2007, the Outstanding Paper Award at the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM2008) in 2008, the International Leadership and Pioneering Research Award in Software Reliability Engineering from the SREQOM/ICQRIT in 2009, the Exceptional International Leadership and Contribution Award in Software Reliability at the ICRITO'2010, the 2011 Best Paper Award from the IEEE Reliability Society Japan Chapter in 2012, the Leadership Award from the ISSAT in 2014, the Project Management Service Award from the Society of Project Management, "Honorary Canon" Appointment from the Korean Reliability Society in 2014, Title of "Honorary Professor" Recognition from Amity University, India, in 2015, Contribution Award for Promoting OR from the Operations Research Society of Japan in 2017, Research Award for Outstanding Contributions in Software Reliability Engineering from the ISSAT in 2017, Best Paper Award at the ISSAT International Conference on Reliability and Quality in Design in 2018, Society Award from the Society of Project Management in 2020, and IEEE Reliability Society Japan Joint Chapter 2020 Reliability Engineering Award in 2021. He is a life member of the IEICE, a life member of the Information Processing Society of Japan, member of the Operations Research Society of Japan (Fellow Member), the Japanese Society for Quality Control, and the Society of Project Management, and the IEEE Life Member. He is also an Honorary Canon of the Korean Reliability Society.