Sensitivity Analysis Considering Wiener Processes and Deep Learning for OSS Reliability Assessment

Seidai Okano[1,*], Yoshinobu Tamura[1]
and Shigeru Yamada[2]

[1]*Yamaguchi University, Japan*
[2]*Tottori University, Japan*
E-mail: e030vgv@yamaguchi-u.ac.jp; tamuray@yamaguchi-u.ac.jp; bex2yama@muse.ocn.ne.jp
[*]*Corresponding Author*

Received 20 August 2025; Accepted 19 September 2025

Abstract

The demand of open source software is increasing because of the low cost, high quality, and short delivery. In particular, open source software is managed by using the bug tracking system. This paper focuses on the method of reliability assessment based on the deep learning. Then, the Wiener process is applied to the output value of objective variables. Moreover, several sensitivity analyses of the parameter of Wiener process are shown as several numerical examples.

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

Journal of Graphic Era University, Vol. 14_1, 35–52.
doi: 10.13052/jgeu0975-1416.1412

1 Introduction

Open Source Software (OSS) is distributed as the free of charge on the Internet. OSS has the advantage that the companies can use the OSS because it is low cost to implement and can be redistributed. However, as the population of OSS users increases, the number of new fault reports is also increasing. Simultaneously, the fault correction time becomes large. Then the fault correction cost is high cost. In this paper, we discuss the application of Wiener process-based data preprocessing to improve the estimation accuracy and estimate the fault correction effort for reliability assessment. We also discuss the machine learning-based software reliability assessment.

OSS software development has no specified testing phase. The OSS faults are eliminated during the operation phase. In this paper, we discuss the reliability evaluation methods for OSS and propose a reliability evaluation method using the fault big data in the development environment. Furthermore, we focus on the characteristics of fault detection events in bug tracking systems. Also, we analyze by using the deep learning based on data preprocessing. Then, we consider the influence from Wiener process.

2 Previous Research

2.1 Open Source Software

OSS is well known as the software whose source code is released by the developer free of charge, and anyone is allowed to freely use, modify, and redistribute it. The advantage of OSS is that it can be used freely by anyone, allowing individuals to add functionality to the source code, report vulnerabilities, and find and fix bugs. Therefore, the improvement and redistributed of OSS is performed according to the OSS development cycle. For this reason, it is currently used by a wide variety of companies and universities. Representative OSS include Linux, MySQL, Java, and Python, which are used in many fields such as operating systems, web browsers, server software, development languages, and databases.

However, OSSRA2021 reports that 84% in 1500 codebases contain OSS, exacerbating the spread of OSS vulnerabilities. Thus, it can be seen that there are many global software failures in Cloud OSS, and once a failure occurs, it is global and the impact surfaces immediately. In addition, OSS software development differs from general commercial software, because there is no explicit testing process, and faults are eliminated as they are used and developed. On the other hand, when we consider the development

environment, we find that such a development environment is no different from the intellectual property activity and identical to the fault occurrence associated with it [1]. In this paper, we discuss the reliability evaluation method of OSS and propose a reliability evaluation method using fault big data in the development environment. In addition, we focus on the characteristics of fault generation events in bug tracking systems and analyze them using deep learning based on data preprocessing. Then, we make a sensitivity analysis in terms of Wiener process parameter.

2.2 Bug Tracking System

A bug tracking system is a system or tool that manages and tracks the process from the discovery and reporting of a fault to the assignment of a person in charge of correcting the fault and the completion of the correction. Functions include problem reporting and tracking, prioritization, communication, and history management.

3 Reliability Evaluation

3.1 Software Reliability

Software is a computer program as a product consisting of programs and documentation created through various development processes. The software reliability is given as the probability that software will not cause the failure of a system for a specified time under specified conditions. Moreover, software reliability can be evaluated by the number of detected faults or the software failure-occurrence time in the testing phase which is the last phase of the development process, and it can be also estimated in the operational phase [2, 3, 4].

3.2 Machine Learning

Machine learning is a data analysis technology that is closely related to artificial intelligence (AI) and deep learning, in which computer systems learn and acquire patterns and knowledge from a variety of data to make predictions. These examples include stock price prediction, medicine, language processing, image processing, and anomaly detection. In addition, the OSS fault data from the determination and correction are ultimately determined by the OSS project manager, automation through machine learning will reduce the development costs, maintenance costs, and time for fault correction. This

also leads to increase the reliability. Various tools have been developed for machine learning, and Python, Matlab, and R are commonly used as machine learning software in programming languages [5].

3.3 Deep Learning

Deep learning is the application of neural networks that reproduce the structure of human brain neurons. Basically, the deep learning is a machine learning technique that consists of a neural network with three or more layers (an input layer, several intermediate layers and an output layer). One of the main differences between deep learning and machine learning is the difference in flexibility for quantified data, a feature that enables deeper learning of complex data by combining multilayer neural networks. At present, the deep learning technique has been applied to automatic driving, optimization of advertising systems, and automatic translation [5].

3.4 Software Reliability Assessment Using Machine Learning

Software reliability evaluation using machine learning is one of the methods to evaluate the probability and quality of software products to work as predicted. Unlike conventional software reliability evaluation methods such as test case design and execution, code review and user feedback, and static analysis tools, this method is highly flexible and enables real-time evaluation. Because of this characteristic, it has been used in the financial engineering and the other fields as a predictor of stock prices, and has also been applied to the prediction of the number of fault findings for software reliability evaluation [6, 7, 8, 9].

However, the predicting stock prices in financial engineering is different from predicting the number of fault findings for the purpose of software reliability assessment. Although the software reliability assessment models can be applied to the prediction of the number of fault detection for software reliability evaluation, it is difficult to obtain the same results as the software reliability growth model because of the problem of estimation accuracy when the prediction is made over a long period of time.

3.5 Estimated Cumulative Number of Detected Faults

Many software reliability growth models have been developed assuming that the process of fault detection is an uncertain event [10, 11]. These stochastic models consist of a continuous physical model with model parameters and

have the advantage of being able to calculate a variety of evaluation measures, allowing for long-term predictions. Therefore, not only probabilistic models but also neural network-based methods have been developed and applied to various software reliability evaluations, such as research applying time series analysis, which is used to predict stock prices, to the fault detection process.

However, applying time series analysis, as typified by stock price prediction, to software reliability evaluation has often been difficult due to differences in stock price trends and fault detection processes. In this paper, we focus on past problems and discuss improvements to software reliability evaluation methods using machine learning.

4 Application of the Wiener Process to Fault Big Data Preprocessing

Considering the optimal fault correction strategies based on deep learning, the software fault data has been used in many software reliability growth models. We apply fault correction time to the objective variable in deep learning. In addition, this paper focuses on the fault correction time and considers a weight parameter as the contribution of the fault corrector in order to add the fault corrector information to the optimal correction strategy. As a result, we define the following equation as the weighted function.

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

c_i is the fault correction time at the ith fault. o_i is the time of fault discovery at the ith fault. Also, γ_i represents the frequency of occurrence of the fault modifier in the ith fault. In this case, $(c_i - o_i)$ represents the fault correction time between the $(i-1)$th and the ith. Therefore, $F(i)$ represents the ith weighted fault correction effort. According to increasing of the value of $F(i)$ increases, the OSS reliability improves.

Next, consider fault big data registered on the OSS bug tracking system. The software fault discovery process is known as an uncertainty event. Traditionally, many software reliability growth models have been used in actual software development. In this case, there are many software reliability growth models that consider incomplete debugging environments [12]. Considering such irregularities in the software fault finding process, we define $W(i)$ as the Wiener process for the white noise $\sigma(i)$ at the ith fault. The Wiener process $W(i)$ then has the following properties.

$$\Pr[W(0) = 0] = 1. \tag{2}$$

$$\mathrm{E}[W(i)] = 0. \tag{3}$$

$$\mathrm{E}[W(i)W(i')] = \mathrm{Min}[i, i']. \tag{4}$$

Here, $\mathrm{E}[\cdot]$ denotes the expected value. In general, Wiener processes are considered as continuous time. In this paper, since we are considering discrete events, we assume $W(i)$ as noise for the discrete ith fault. Continuous-time models based on Wiener processes have also been proposed in conventional software reliability growth models [13, 14, 15]. In general, the horizontal axis is treated as a unit of time, but there are examples where it is constructed as a continuous physical model. In this paper, we assume that the fault correction process follows a Wiener process and define the following objective function as the output value of deep learning.

$$F_\omega(i) = (c_i - o_i)\gamma_i + W(i), \tag{5}$$

where $W(i)$ denotes the Wiener process for the ith fault.

5 Numerical Example

Assuming a real OSS environment, we present a numerical example using fault big data from OpenStack [16], an OSS for cloud computing. Figures 1, 2, 3 and 4 presents the estimated errors for the training and validation data, the estimated weighted fault correction effort, the scatter plots for the estimated weighted fault correction effort, and the estimated weighted cumulative fault correction effort.

Figures 5–9 presents the estimated errors for the training and validation data, the estimated weighted fault correction effort, the scatter plots for the estimated weighted fault correction effort, and the estimated weighted cumulative fault correction effort when the Wiener process is considered. correction effort.

Comparing 4 and 8, it can be seen that the estimated weighted cumulative fault correction effort in the second half of the estimation is markedly different, confirming the better fit of 8, which considers the Wiener process.

Furthermore, the mean absolute error rates shown in Figures 4 and 8 were calculated, yielding 11.33% for Figure 4 and 1.64% for Figure 8. These results confirm the effectiveness of applying the Wiener process in this method.

In addition, 10, 11, 12 and 13 presents the results of the sensitivity analysis to the parameterσ in the Wiener process, the results of the sensitivity

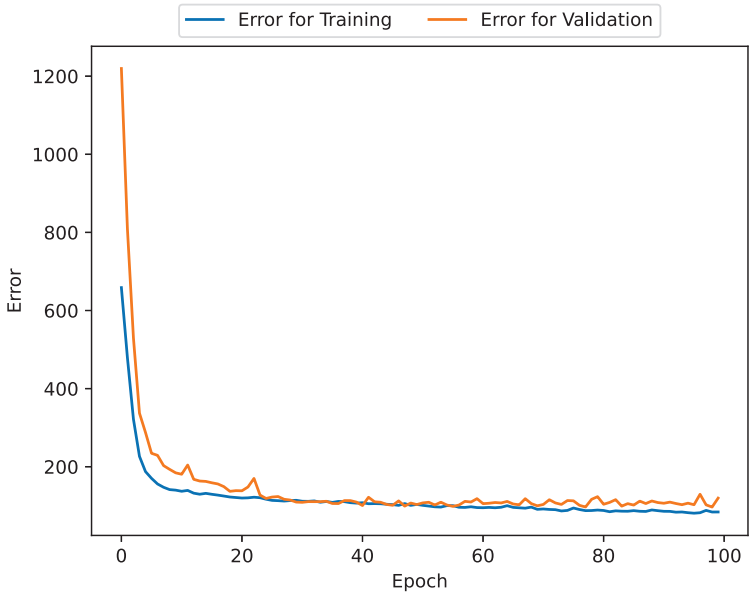

Figure 1 The estimated errors for the training and validation data.

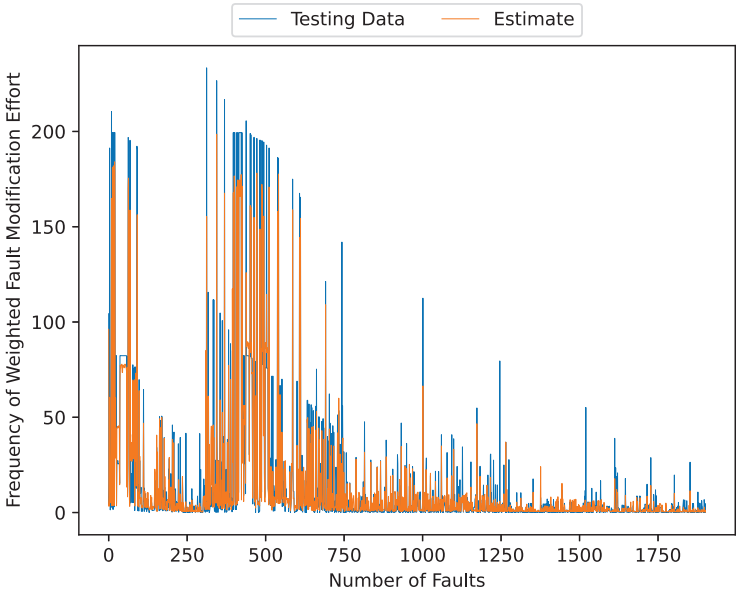

Figure 2 The estimated weighted fault correction effort.

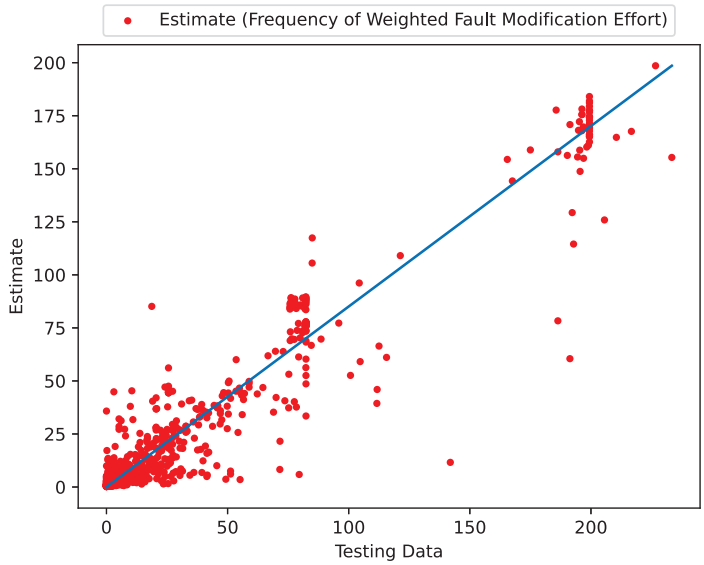

Figure 3 The scatter plots for the estimated weighted fault correction effort.

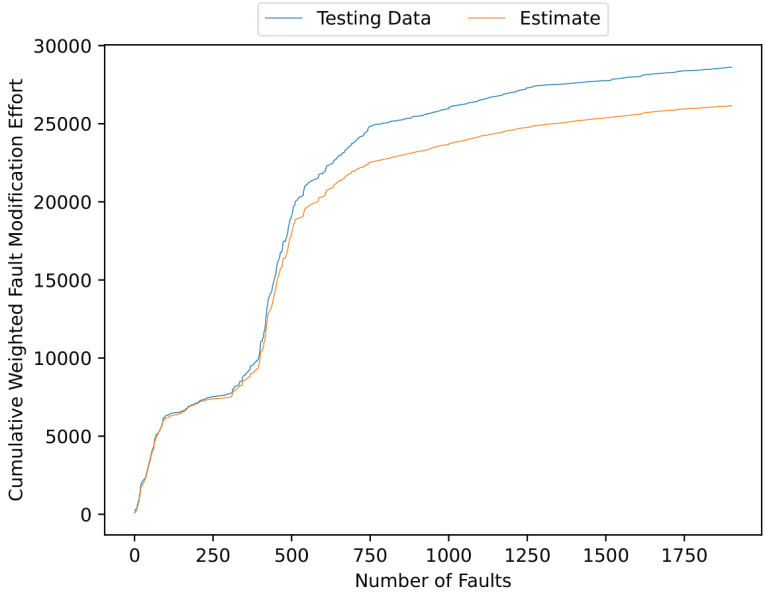

Figure 4 The estimated weighted cumulative fault correction effort.

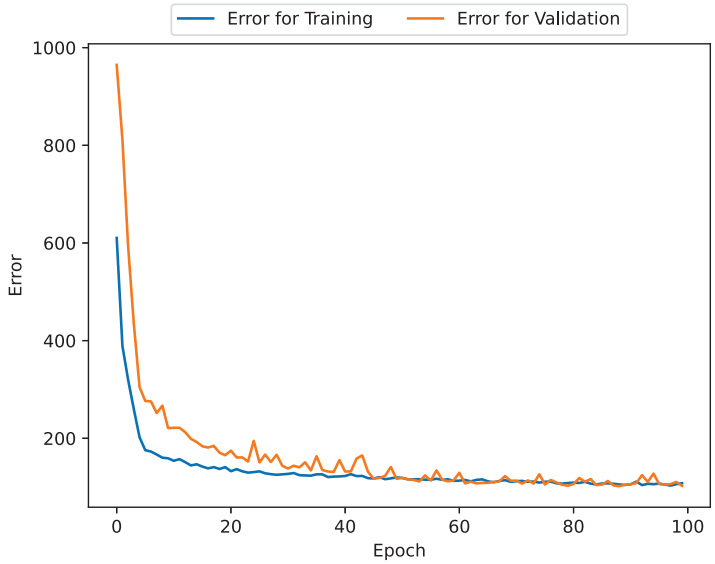

Figure 5 Estimated errors for training and validation data considering Wiener process noise.

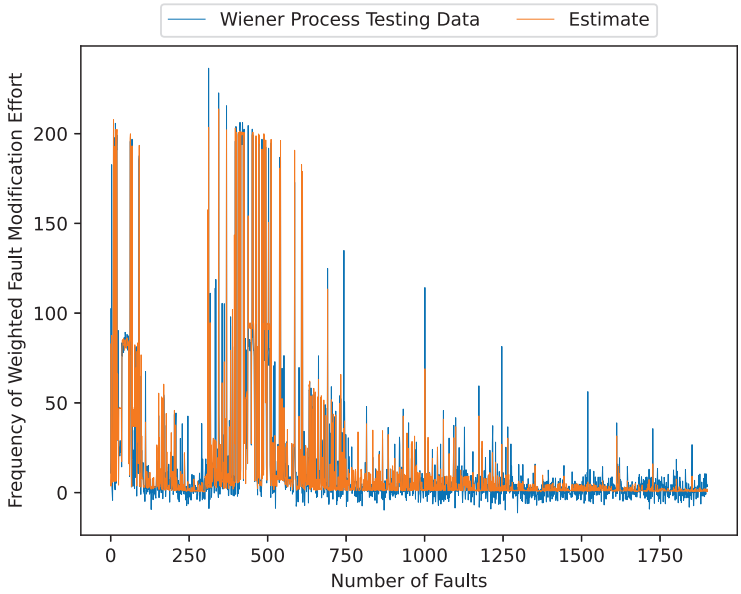

Figure 6 Estimated weighted fault correction effort considering Wiener process noise.

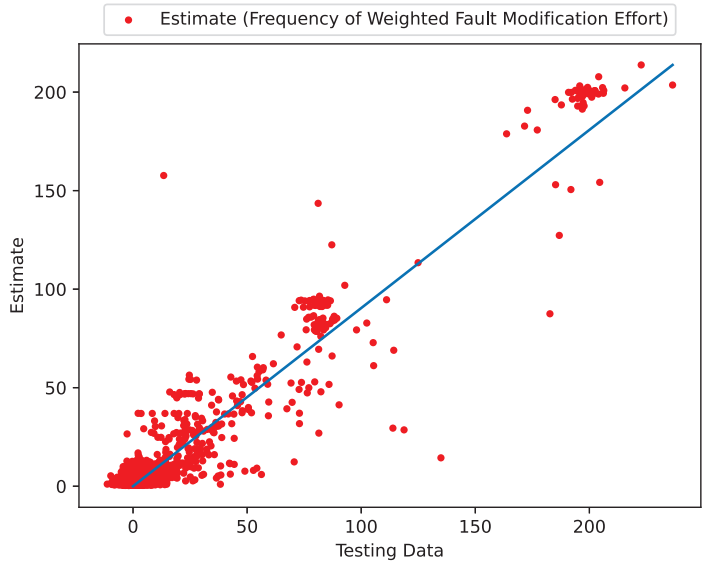

Figure 7 Scatter plots for estimated weighted fault correction effort considering Wiener process noise.

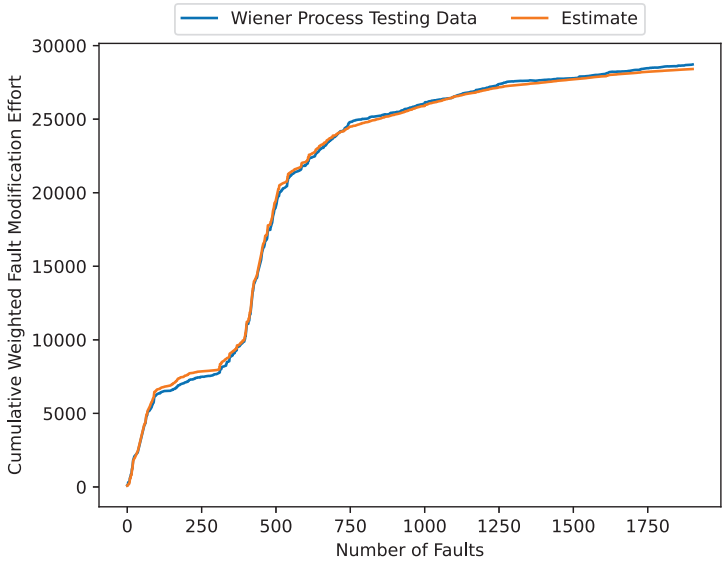

Figure 8 Estimated weighted cumulative fault correction effort accounting for Wiener process noise.

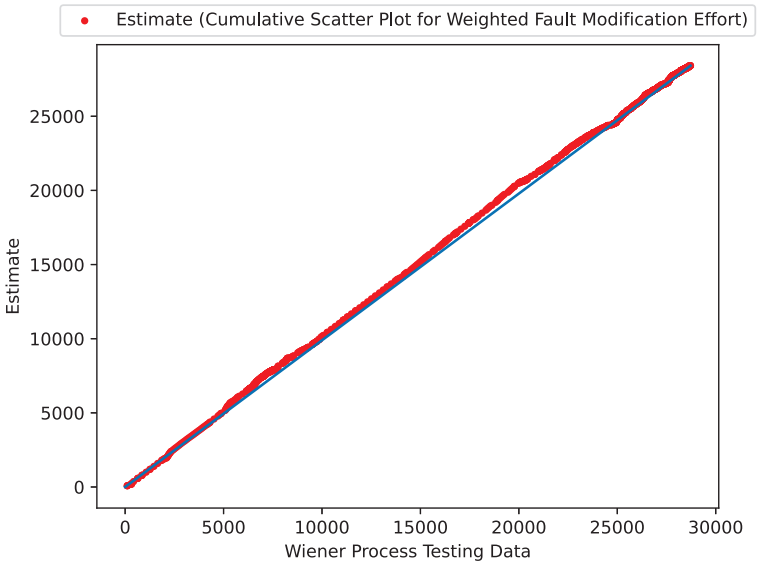

Figure 9 Scatter plots for estimated weighted cumulative fault correction effort accounting for Wiener process noise.

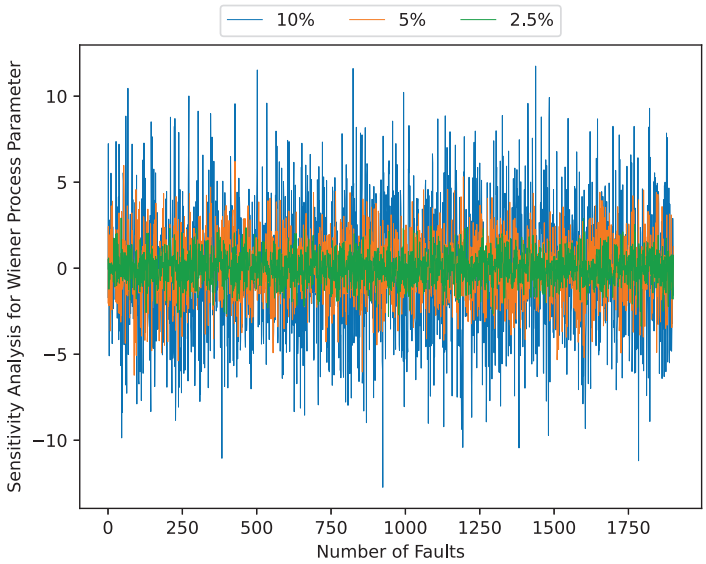

Figure 10 Sensitivity analysis results for parameter σ in the Wiener process.

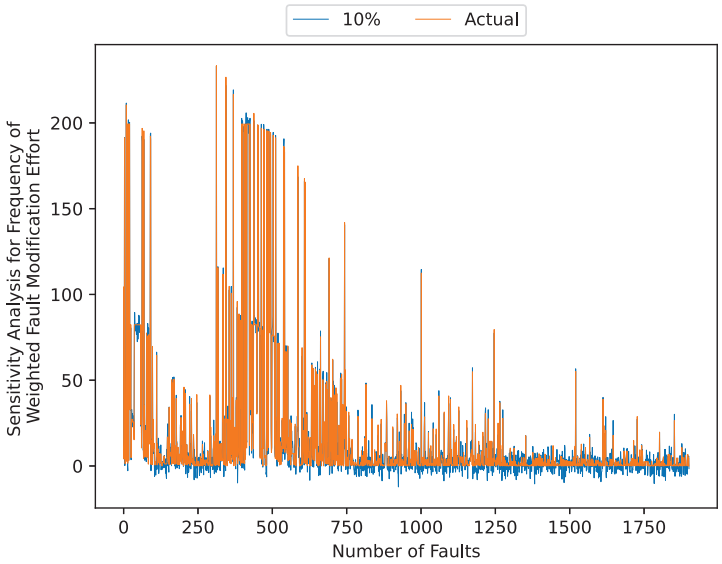

Figure 11 Sensitivity analysis results for weighted fault correction effort when varying the parameter σ in the Wiener process ($\sigma = 0.1$).

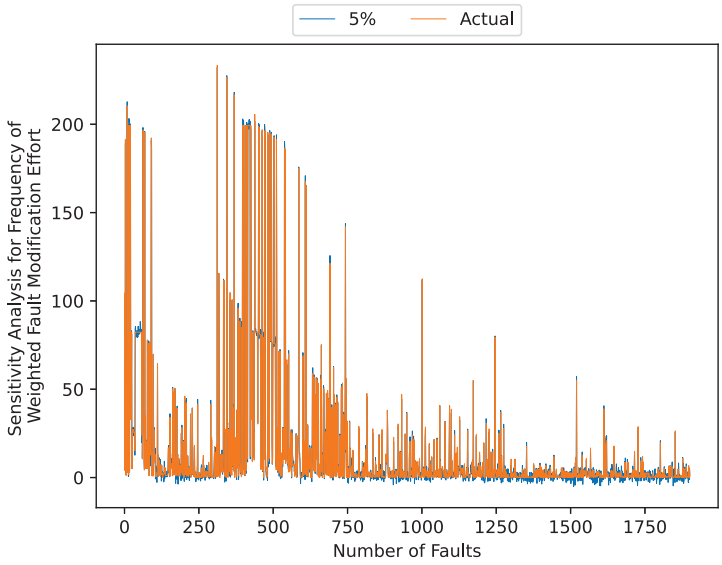

Figure 12 Sensitivity analysis results for weighted fault correction effort when varying the parameter σ in the Wiener process ($\sigma = 0.05$).

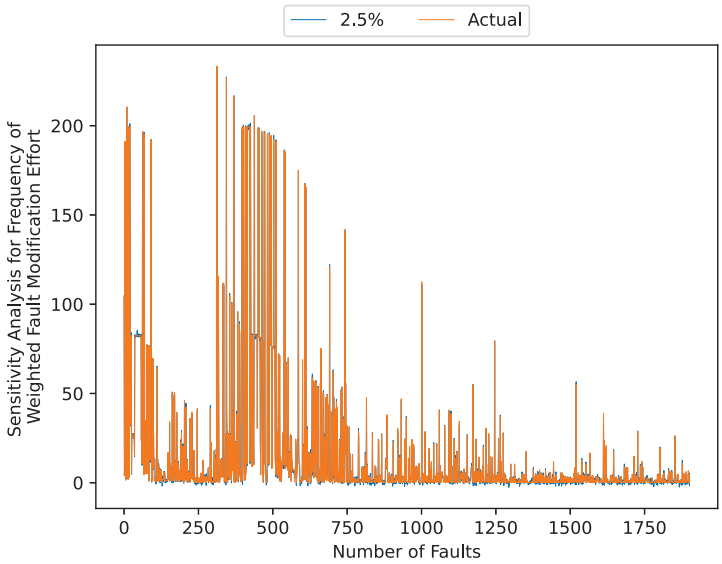

Figure 13 Sensitivity analysis results for weighted fault correction effort when varying the parameter σ in the Wiener process ($\sigma = 0.025$).

analysis to the weighted fault correction effort when varying the parameterσ in the Wiener process ($\sigma = 0.1$). Sensitivity analysis results for weighted fault-corrected effort for varying the parameterσ in the Wiener process ($\sigma = 0.05$), the results of a sensitivity analysis to the weighted fault correction effort for varying the parameterσ in the Wiener process ($\sigma = 0.025$).

These figures show that the noise is small. In particular, he swing width is large when the σ becomes growth.

6 Concluding Remarks

In this paper, we discussed an OSS reliability evaluation method based on deep learning. Specifically, this method sets the functions including development man-hour data and fault correction time as objective variables. Furthermore, by utilizing the weighted correction effort as learning data and prediction data, it is considered possible to quickly assign appropriate correction personnel even in situations where fault data is uncertain.

The method proposed in this study enables the estimation of cumulative fault correction effort, which can be applied to development man-hour estimates. Furthermore, by utilizing the estimated correction effort for each fault,

the workload balance among developers can be optimized. Specifically, it is expected to improve development site efficiency by assigning faults that require a long time to correct to experienced developers and faults that can be addressed in a short time to new developers.

In addition, the effectiveness of the proposed method over existing methods is discussed by presenting an example of reliability evaluation using actual OSS data in a numerical example. The results confirmed that the estimation accuracy is appropriate when the noise based on the Wiener process is applied during the data preprocessing phase.

This paper employed a Wiener process. However, the Wiener processes have the limitation of being able to represent only continuous and stationary fluctuations, and cannot adequately capture large-scale discontinuous jumps. Therefore, a future research topic is to introduce jump-diffusion processes or Lévy processes to enable the representation of non-stationary and realistic fluctuations, aiming to construct a reliability evaluation model more suited to actual operational environments.

The method proposed in this study enables the estimation of cumulative fault correction effort, which can be applied to development man-hour estimates. Furthermore, by utilizing the estimated correction effort for each fault, the workload balance among developers can be optimized. Specifically, it is expected to improve development site efficiency by assigning faults that require a long time to correct to experienced developers and faults that can be addressed in a short time to new developers.

Acknowledgments

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

References

[1] S. Yamada and Y. Tamura, A deep learning approach for OSS reliability assessment based on Wiener process-aware data preprocessing, Technical Report of IEICE, Sapporo Electronics Center, Vol. 123, No. 141, pp. 33–38, 28 July. 2023.

[2] S. Yamada, Software Reliability/Safety Assessment, Special Issue on Safety in Advanced Technology, Vol. 33, No. 6, pp. 432–441, 1994.

[3] S. Yamada, S. Inoue and Y. Tamura, Softfair Reliability Research, The Institute of Electronics, Information and Communication Engineers

Basic and Association Societies Fundamentals Review, Vol. 12, No. 1, pp. 38–50, July, 2018.

[4] T. Chusho, "Reliability on software," Reliability Engineering Association of Japan, Vol. 22, No. 5, pp. 367, 2000.

[5] T. Suzuki, "Machine learning overview," Applied Mathematics, Vol. 28, No. 1, pp. 35, March, 2018.

[6] N. Karunanithi, Y. K. Malaiya, and D. Whitley, "Predictionof software reliability using neural networks," Proceedings. 2nd International. Symposium. Software. Reliability. Engineering., pp. 124–130, IEEE Computer SocietyPress, 1991.

[7] N. Karunanithi, D. Whitley, and Y. K. Malaiya, "Using neural networks in reliability prediction, " IEEE Software, vol. 9, pp. 53–59, 1992.

[8] N. Karunanithi , Y. K. Malaiya, "Neural networks for software reliability engineering, " in Handbook of Software Relia bility Engineering, ed. M. R. Lyu, pp. 699–728, McGraw-Hill, New York, 1996.

[9] Y. Shinohara, M. Imanishi, T. Dohi, and S. Osaki, "Software reliability prediction using neural network technique," in Proceedings of the Second Australia Japan Workshop onStochastic Models in Engineering, Technology & Manage ment, eds. R. J. Wilson, D.N.P. Murthy, and S. Osaki, pp. 564–571, Technology Management Centre, The University of Queensland, Brisbance, 1996.

[10] S. Yamada, *Software Reliability Modeling: Fundamentalsand Applications,* Springer–Verlag, Tokyo/Heidelberg, 2014.

[11] P.K. Kapur, H.Pham, A. Gupta, and P.C. Jha. *Software Re liability Assessment with OR Applications,* Springer-Verlag, London, 2011.

[12] S. Yamada, K. Tokuno, and S. Osaki, "Imperfect debugging models with fault introduction rate for software reliability assessment, " *International Journal of Systems Science,* Vol. 23, Issue 12, pp. 2241–2252, 1992.

[13] L. Arnold, *Stochastic Differential Equations-Theory and Ap plications,* John Wiley & Sons, New York, 19.

[14] E. Wong, *Stochastic Processes in Information and Systems,* McGraw-Hill, New York, 1971.

[15] S. Yamada, M. Kimura, H. Tanaka, and S. Osaki, "Software reliability measurement and assessment with stochastic differential equations, " *IEICE Transactions on Fundamentals,* Vol. E77–A, No. 1, pp. 109–116, 1994.

[16] The OpenStack project, OpenStack, http://www.openstack.org/.

Biographies

Seidai Okano 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.

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, IEEE Reliability Society Japan Joint Chapter 2020 Reliability Engineering Award in 2021, and Lifetime Achievement Award from the ISSAT in 2025. 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.