Early Software Reliability Prediction with Wavelet Networks Models

Denghua Mei

School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, P.R. China

Abstract

Accurate reliability estimates can be obtained by using software reliability models only in the later phase of software testing. However, for cost effective and timely, management prediction in the early phase is important. Non-homogenerous Poisson process (NHPP) models and Artificial Neural Network (ANN) models are the most important Analytical software reliability growth models. In this paper we study an approach to using past fault-related data with Wavelet Networks model to improve reliability predictions in the early testing phase. A numerical example is shown with both actual and simulated datasets. The analysis with example shows that the proposed approach works effectively in the early phase of software testing.

Keywords: Wavelet Networks Models, Early Software Reliability, Prediction, Accurate

1. Introduction

Data-driven approach and analytical method are used to develop software reliability model. Non-Homogeneous Poisson Process (NHPP) SRGMs is one of the important Analytical software reliability growth models (SRGMs). ANN (Artificial Neural Network) software reliability models are studied more these years[1-6]. Normally，most of models consider fault detection process and data for analysis. In the early phase of testing, it is difficult to obtain accurate predictions. But early software reliability prediction is useful for decision of the software development process. It is necessary to predict accurately earlier.

The first NHPP model, which strongly influences the development of many other models, was proposed by Goel and Okumoto [22]. Huang et al. [11] have discussed a unified scheme of discrete NHPP models by applying the concepts of weighted arithmetic, weighted geometric, or weighted harmonic means. Ohba [9] presented a NHPP model with S-shaped mean value function. Lots of the generalized SRGMs, including the generalized SRGMs mentioned above, have been discussed in terms of continuous-time SRGMs, because the continuous-time SRGM is specifically applicable to the reliability analysis [13]. S. Inoue et al. give a Generalized Discrete Software Reliability Modeling With Effect of Program Size[12]. Okamura et al. [14] have discussed a unified parameter estimation method based on the expectation–maximization (EM) principle and investigated the effectiveness of the estimation method based on the EM algorithm by comparing with Newton's method. Khoshgoftarr et al.[7] introduced the use of the neural networks as a tool for predicting software quality. Their model used domain metrics derived from the complexity metric data. Paper [15-20] have been adapted neural networks to software reliability issues. Emanm and Melo [8] have performed to construct a logistic regression model to predict which classes in a future release of a commercial Java application will be faulty.

2. Early Software Reliability Model with Wavelet Networks

The paper gives the wavelet networks based model to capture the input–output (I/O) relationships of software system to corresponding fault and to improve the accurate of predicting the reliability. The wavelet network tales input and output through evolutionary algorithm adjusted wavelet networks.

With the historical fault-related data of $\{y_1, y_2, \ldots, y_t\}$, the corresponding mapping of wavelet network can be written as $y_{t+1} = g(y_t, y_{t-1}, \ldots, y_{t-k})$. This relationship is illustrated in the framework of Figure 1. Traditional ANN models describe fault detection process and y_t there is a scalar, such as cumulative detected faults number $y_t = d_t$. Figure 2 gives the structure of wavelet networks.

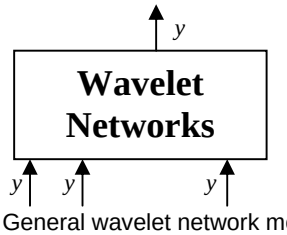

Figure. 1: General wavelet network model framework..

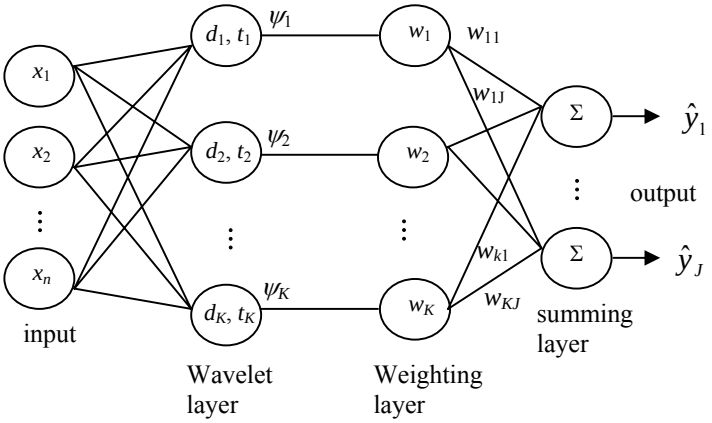

Figure.2: Wavelet networks.

The wavelet networks model for software reliability prediction are designed as a three-layer structure with a wavelet layer (input layer), weighting layer (intermediate layer), and summing layer (output layer). Each layer has one or more nodes. The input data vector is connected to the input nodes of the networks:

$$x = [x_1, x_2, \cdots, x_n]^T \quad (1)$$

n is the input variable number. We can derive the activation functions of the wavelet nodes in the wavelet layer from a mother wavelet $\psi(x) \in L^2(R)$. And

$$\int_{-\infty}^{+\infty} \frac{|\hat{\psi}(w)|^2}{w} dw < \infty \quad (2)$$

$\hat{\psi}$ means the Fourier transform of $\psi(x)$.

$$\psi_{d,t}(x) = 2^{\frac{d}{2}} \psi(2^d x - t) \quad d, t \in Z \quad (3)$$

where Z indicates integer.

The function of $\psi(x) = -xe^{-\frac{1}{2}x^2}$ is the mother wavelet. The activation function of the kth wavelet node k=1, 2, …, K is:

$$\psi_{d_k,t_k}(x) = -2^{\frac{d_k}{2}} (2^{d_k} x - t_k) e^{-\frac{1}{2}(2^{d_k} x - t_k)^2}$$
$$d_k, t_k \in Z \quad (4)$$

The output of the kth wavelet node $\psi_k: R^n \to R$ with n input variables x_i, i=1, 2, …, n, is:

$$\psi_k(x) = \sum_{i=1}^{n} \psi_{d_k,t_k}(x_i) \quad (5)$$

The weights w_{kj}, that connect the kth weighting node and the jth output node are indicated by the weighting vectors w_k=[w_{k1}, w_{k2}, …, w_{kJ}] for k=1, 2, …, K and j=1, 2, …, J; J is the number of the output nodes. The final output of the wavelet networks summing layer is:

$$\hat{y}_j(x) = \sum_{k=1}^{K} w_{kj} \psi_k(x) \quad (6)$$

The parameters of dilation, translation and weighting values of the wavelet networks can be determined by a global-optimal approach[10], [21].

The least-squared error (LSE) function L_i is used to represent the unfitness value of the EWNs associated with the individual $T^i = [t_1^i, t_2^i, \cdots, t_D^i]$ (i=1, 2, …, I), where I is the population size. The elements of t_d, d=1, 2, …, D, are the desired values of the dilation and translation parameters or the weighting values:

$$L_i = \frac{1}{2} \sum_{m=1}^{M} \sum_{j=1}^{J} [y_{jm}(x) - \hat{y}_{jm}(x)]^2 \quad (7)$$

Where $\hat{y}_{jm}(x)$ indicates the jth computed output of the wavelet networks for the nth sample vector. M denotes the total number of sample vectors. $y_{jm}(x)$ is the corresponding actual fault indicator in the jth output node.

Parent T^i creats the offspring vector T^{i+I} by adding a Gaussian random variation with zero mean and a standard deviation proportional to the scaled unfitness value of the parent trial solution:

$$T^{i+I} = T^i + N(0, \delta_i^2) \quad (8)$$

$N(0, \delta_i^2)$ is a vector of Gaussian random variables with mean zero and standard δ_i.

$$\delta_i = sE_i + F \quad (9)$$

Where s is a scaling factor, F indicates an offset.

By the unfitness values, the $2I$ individuals (includes the parent and the created offspring)

compete with the other randomly selected individuals for win:

$$S_i = \sum_{i=1}^{N_C} S_t \tag{10}$$

$$S_i = \begin{cases} 1, v_1 < \dfrac{u_k}{u_k + u_i} \\ 0, otherwise \end{cases} \tag{11}$$

Where u_i is the unfitness value of ith rival. N_c is the number of competitors that are chose randomly. S_i is the score of the ith individuals. $K=\lfloor 2Iv_2+1 \rfloor$. v_1, $v_2 \sim U(0, 1)$ are uniform random real numbers ranging over [0, 1].

All $2I$ individuals are ranked in descending order of their corresponding S_i value when all individuals have experienced competition. The first I individuals and their corresponding unfitness values are selected as new parents of the next generation.

The evolution process stops when the minimum criterion of LSE value or the stopping rule of maximum generation is satisfied. The solution with the lowest unfitness value is regarded as the best wavelet networks for software reliability model.

3. Numerical examples

To illustrate the proposed approach with wavelet networks model, numerical examples are studied in this section.

y_t is a vector composed of cumulative detected number and corrected faults number in the software system at time period t, denoted as $y_t = [d_t, c_t]$.

The example is based on a project for a large Web-based software system. The cumulative detected faults number and corrected faults number.

Week	d_i	c_i	$\hat{d}_i$	$\hat{c}_i$	L_i
1	1.62				
2	2.44				
3	6.47				
4	8.14	6.81	7.19		30.34
5	8.42	5.64	8.22	4.92	19.22
6	9.38	5.81	9.13	5.57	25.27
7	10.34	6.47	9.94	6.34	21.55
8	11.31	7.28	11.89	6.82	17.72
9	12.73	8.39	12.92	8.47	16.85
10	13.15	9.52	12.36	9.57	31.62
11	13.46	10.73	13.1	10.94	27.34
12	13.86	11.11	13.42	118.24	22.9
13	13.97	12.35	14.44	12.48	26.1
14	14.26	13.28	14.52	13.81	10.7
15	14.64	14.62	14.92	14.91	17.3
16	14.70	15.46	15.37	15.83	22.5
17	14.92	16.82	15.77	16.23	38.7
18	15.59	17.92	16.43	17.97	31.7
19	15.92	16.83	16.74	16.35	21.4
20	16.31	17.23	16.92	17.71	27.8
21	16.54	17.42	17.29	17.67	25.2
22	16.76	17.23	17.53	17.42	23.8
23	17.19	17.49	17.82	17.93	20.5
24	17.38	17.83	18.05	17.95	17.4
25	17.78	17.94	18.24	18.46	13.2

Table. 1: Cumulative detected faults prediction.

4. Discussions

In this paper, we have studied an approach of early reliability prediction by wavelet networks. The analysis with example shows that the proposed approach works effectively in the early phase of software testing. However, current analysis needs more practical project datasets are essential for further study.

References

[1] R. Sitte, 1999, Comparison of software-reliabilitygrowth predictions: neural networks vs parametricrecalibration. *IEEE Transactions on Reliability*, 48(3): 285-291.

[2] M.R. Lyu, *Handbook of software reliability engineering,* IEEE Computer Society Press, 1996.

[3] M. Xie, *Software reliability modelling,* World Scientific, Singapore, 1991.

[4] Y. Takada, Matsumoto, K. & Torii, K., A Software-Reliability Prediction Model Using a Neural-Network. *Systems and Computers in Japan*, 25(14): 22-31, 1994.

[5] J.D. Musa, Iannino, A. & Okumoto, K., *Software reliability: measurement, prediction, application,* McGraw-Hill, New York, 1987.

[6] N. Karunanithi, Whitley, D. & Malaiya, Y.K., Prediction of software reliability using connectionist models. *IEEE Transactions on Software Engineering*, 18,(7): 563-574, 1992.

[7] P.M. Khoshgoftaar, E.B. Allen, J.P. Hudepohl and S.J. Aud, Application of neural networks to software quality modeling of a very large telecommunications system. *IEEE Transactions on Neural Network*, 8: 902-909, 1997.

[8] E. Emam, W. Melo, The Prediction of Faulty Classes Using Object-Oriented Design Metrics. *Journal of Systems and Software, Elsevier Science*, 2001. Technical Report, NRCERB-1064, NRC 43609, 1999.

[9] S. Yamada, S. Osaki, S-shaped Software Reliability Growth Model with Four Types of Software Error Data. *Int. J. Systems Science,* 14: 683-692, 1983.

[10] Fogel, *System Identification Through Simulated Evolution: A Machine Learning Approach to Modeling*, Needham, MA: Ginn, 1991.

[11] C.Y. Huang, M. R. Lyu, S. Y. Kuo, A unified scheme of some nonhomogeneous Poisson process models for software reliability estimation. *IEEE Trans. Softw. Eng*., 29(3): 261–269, 2003.

[12] S. Inoue, S. Yamada, Generalized Discrete Software Reliability Modeling With Effect of Program Size. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS*, 37(2): 170-179, 2007.

[13] A. Fries and A. Sen, A survey of discrete reliability-growth models. *IEEE Trans. Rel.*, 45(4): 582–604, 1996.

[14] H. Okamura, A. Murayama, T. Dohi, EM algorithm for discrete software reliability models: A unified parameter estimation method. *In Proc. 8th IEEE Int. Symp. HASE*, pp. 219-228, 2004.

[15] N. Karunanithi, Y. K. Malaiya, D. Whitley, Prediction of Software Reliability Using Neural Networks. *Proc. 1991 IEEE Int, Symp. on Soft. Rel. Eng.*, pp. 124-130, 1991.

[16] N. Karunanithi, D. Whitley, Y. K. Malaiya, Using Neural Networks in Reliability Prediction. *IEEE Software*, 9(4): 53-59, 1992.

[17] J.P. Hudepohl, S. J. Aud, T. M. Khoshgoftaar and E. B. Allen, et.al., EMERALD: Software Metrics and Models on the Desktop. *IEEE Software*, 13(5): 56–60, 1996.

[18] T. Dohi, Y. Nishio, S. Osaki, Optimal Software Release Scheduling Based on Artificial Neural Networks. *Annals of Software Engineering*, 8: 167-185, 1999.

[19] K.Y. Cai, L. Cai, W. D. Wang and Z. Y. Yu, et.al., On the Neural Network Approach in Software Reliability Modeling. *The Journal of Systems and Software*, 47-62, 2001.

[20] S.L. Ho, M. Xie, T.N. Goh, A Study of the Connectionist Models for Software Reliability Prediction. *Computers and Mathematics with Applications*, 46: 1037-1045, 2003.

[21] D.B. Fogel, An introduction to simulated evolutionary optimization. *IEEE Trans. Neural Networks*, 5: 3-14, 1994.

[22] A.L. Goel, K. Okumoto, Time-Dependent error-Detection Rate Model for Software Reliability and Other Performance Measures. *IEEE transactions on Reliability,* R-28(3): 206-211, 1979.