Support Vector Regression for Software Reliability Growth Modeling and Prediction

Fei Xing[1] and Ping Guo[2]

[1] Department of Computer Science
Beijing Normal University, Beijing 100875, China
xsoar@163.com
[2] Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences, Beijing 100080, China
pguo@ieee.org

Abstract. In this work, we propose to apply support vector regression (SVR) to build software reliability growth model (SRGM). SRGM is an important aspect in software reliability engineering. Software reliability is the probability that a given software will be functioning without failure during a specified period of time in a specified environment. In order to obtain the better performance of SRGM, practical selection of parameter C for SVR is discussed in the experiments. Experimental results with the classical Sys1 and Sys3 SRGM data set show that the performance of the proposed SVR-based SRGM is better than conventional SRGMs and relative good prediction and generalization ability are achieved.

1 Introduction

Software reliability is one of key factors in software qualities. It is one of the most important problem facing the software industry. There exists an increasing demand of delivering reliable software products. Developing reliable software is a difficult problem because there are many factors impacting development such as resource limitations, unrealistic requirements, *etc.* It is also a hard problem to know whether or not the software being delivered is reliable. To solve the problem, many software reliability models have been proposed over past 30 years, they can provide quantitative measures of the reliability of software systems during software development processes [1–3].

Software reliability growth models (SRGMs) have been proven to be successful in estimating the software reliability and the number of errors remaining in the software [2]. Using SRGMs, people can assess the current reliability and predict the future reliability, and further more, conduct the software testing and debugging process. Now there have already existed many SRGMs such as Goel-Okumoto Model, Yamada Delayed S-Shaped Model, Yamada Weibull-Type Testing-Effort Function Model, *etc.* Most of SRGMs assume that the fault process follows the curve of specific type. Actually, this assumption may not be realistic in practice and these SRGMs are sometimes insufficient and inaccurate to analyze actual software failure data for reliability assessment.

J. Wang, X. Liao, and Z. Yi (Eds.): ISNN 2005, LNCS 3496, pp. 925–930, 2005.

In recent years, support vector machine (SVM) [4] is a new technique for solving pattern classification and universal approximation, it has been demonstrated to be very valuable for several real-world applications [5, 6]. SVM is known to generalize well in most cases and adapts at modeling nonlinear functional relationships which are difficult to model with other techniques. Consequently, we propose to apply support vector regression (SVR) to build SRGM and investigate the conditions which are typically encountered in software reliability engineering. We believe that all these characteristics are appropriate to SRGM.

2 Support Vector Regression

SVM was introduced by Vapnik in the late 1960s on the foundation of statistical learning theory [7]. It has originally been used for classification purposes but its principle can be extended easily to the task of regression by introducing an alternative loss function. The basic idea of SVR is to map the input data $\mathbf{x}$ into a higher dimensional feature space $\mathcal{F}$ via a nonlinear mapping ϕ and then a linear regression problem is obtained and solved in this feature space.

Given a training set of l examples $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_l, y_l)\} \subset \mathbb{R}^n \times \mathbb{R}$, where $\mathbf{x}_i$ is the input vector of dimension n and y_i is the associated target. We want to estimate the following linear regression:

$$f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b, \qquad \mathbf{w} \in \mathbb{R}^n,\ b \in \mathbb{R}, \tag{1}$$

Here we consider the special case of SVR problem with Vapnik's ϵ-insensitive loss function defined as:

$$L_\epsilon(y, f(\mathbf{x})) = \begin{cases} 0 & |y - f(\mathbf{x})| \le \epsilon \\ |y - f(\mathbf{x})| - \epsilon & |y - f(\mathbf{x})| > \epsilon \end{cases} \tag{2}$$

The best line is defined to be that line which minimizes the following cost function:

$$R(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l} L_\epsilon(f(y_i, \mathbf{x}_i)) \tag{3}$$

where C is a constant determining the trade-off between the training errors and the model complexity. By introducing the slack variables ξ_i, ξ_i^*, we can get the equivalent problem of Eq. 3. If the observed point is "above" the tube, ξ_i is the positive difference between the observed value and ϵ. Similar, if the observed point is "below" the tube, ξ_i^* is the negative difference between the observed value and $-\epsilon$. Written as a constrained optimization problem, it amounts to minimizing:

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \tag{4}$$

subject to:

$$\begin{gathered} y_i - (\mathbf{w} \cdot \mathbf{x}_i) - b \le \epsilon + \xi_i \\ (\mathbf{w} \cdot \mathbf{x}_i) + b - y_i \le \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{gathered} \tag{5}$$

To generalize to non-linear regression, we replace the dot product with a kernel function $K(\cdot)$ which is defined as $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$. By introducing Lagrange multipliers α_i, α_i^* which are associated with each training vector to cope with both upper and lower accuracy constraints, respectively, we can obtain the dual problem which maximizes the following function:

$$\sum_{i=1}^{l}(\alpha_i - \alpha_i^*)y_i - \epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) - \frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(\mathbf{x}_i, \mathbf{x}_j) \tag{6}$$

subject to:

$$\begin{aligned} &\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0 \\ &0 \le \alpha_i, \alpha_i^* \le C \qquad i = 1, 2, \ldots, l \end{aligned} \tag{7}$$

Finally, the estimate of the regression function at a given point $\mathbf{x}$ is then:

$$f(\mathbf{x}) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}) + b \tag{8}$$

3 Modeling the Software Reliability Growth

In this section, we present real projects to which we apply SVR for software reliability growth generalization and prediction. The data sets are Sys1 and Sys3 software failure data applied for software reliability growth modeling in [2]. Sys1 data set contains 54 data pairs and Sys3 data set contains 278 data pairs. The data set are normalized to the range of [0,1] first. The normalized successive failure occurrence times is the input of SVR function and the normalized accumulated failure number is the output of SVR function. We denote the SVR-based software reliability growth model as SVRSRG.

Here we list the math expression of three conventional SRGMs refered in the experiments.

- Goel-Okumoto Model:

$$m(t) = a(1 - e^{rt}),\ a > 0,\ r > 0 \tag{9}$$

- Yamada Delayed S-Shaped Model:

$$m(t) = a(1 - (1 + rt)e^{-rt}) \tag{10}$$

- Yamada Weibull-Type Testing-Effort Function Model:

$$m(t) = a[1 - e^{-r\alpha(1-e^{-\beta t^{\gamma}})}] \tag{11}$$

The approach taken to perform the modeling and prediction includes following steps:

1. Modeling the reliability growth based on the raw failure data
2. Estimating the model parameters
3. Reliability prediction based on the established model

Three groups of experiments have been performed. Training error and testing error have been used as evaluation criteria. In the tables presented in this paper, the training error and the testing error are measured by sum-of-square $\sum_{i=1}^{l}(x_i - \hat{x}_i)^2$, where $x_i, \hat{x}_i$ are, respectively, the data set measurements and their prediction. In default case, SVR used in the experiment is ν-SVR and the parameters ν and C are optimized by cross-validation method.

In the experiment of generalization, we partition the data into two parts: training set and test set. Two thirds of the samples are randomly drawn from the original data set as training set and remaining one third of the samples as the testing set. This kind of training is called generalization training [8]. Fig. 1. (a) and (b) show the experimental result for software reliability growth modeling trained by using data set Sys1 and Sys3, respectively. It is obvious that SVRSRG gives a better performance of fitting the original data than the other models. From Table 1 we can find both the training error and the testing error of SVRSRG are smaller than the other classical SRGMs.

In the experiment of prediction, we will simulate the practical process of software reliability growth prediction. It is based on predicting future values by the way of time series prediction methods. Assuming software have been

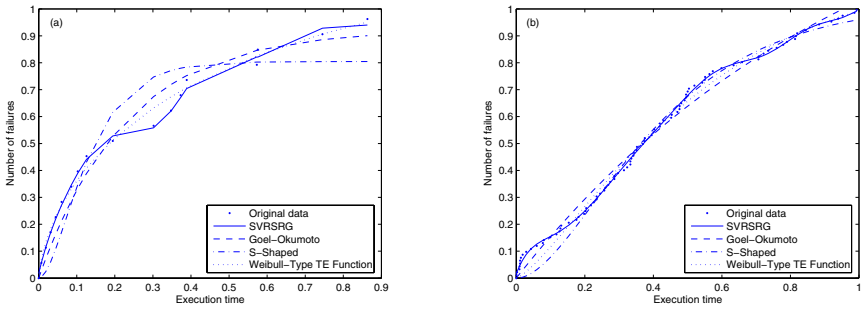

Fig. 1. (a) The generalization curve of four SRGMs trained with Sys1 data set (b) The generalization curve of four SRGMs trained with Sys3 data set

Table 1. The comparison of training error and testing error of four SRGMs for generalization

Data	Sys1		Sys3	
Models	Training error	Test error	Training error	Test error
Goel-Okumoto	0.1098	0.0576	0.1255	0.0672
S-Shaped	0.3527	0.1722	0.1792	0.0916
Weibull-type TE Function	0.0255	0.0137	0.1969	0.1040
SVRSRG	0.0065	0.0048	0.0147	0.0078

executed for time x_i and considering i data pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_i, y_i)$, we calculate predicted number of failures y_{i+1} at time x_{i+1} in the following way. First, use first i data pairs to build model. Second, predict value at time x_{i+1} using current model. Experiment is processed at each time points and mean training error and mean testing error are reported. From Table 2, we can find that in practical process SVRSRG can achieve more remarkable performance than other four SRGMs.

Table 2. The comparison of training error and testing error of four SRGMs for prediction

Data	Sys1		Sys3	
Models	Training error	Test error	Training error	Test error
Goel-Okumoto	0.0259	0.0038	0.0864	0.0011
S-Shaped	0.0855	0.0110	0.1663	0.0015
Weibull-type TE Function	0.0127	0.0012	0.0761	0.0007
SVRSRG	0.0064	0.0021	0.0138	0.0001

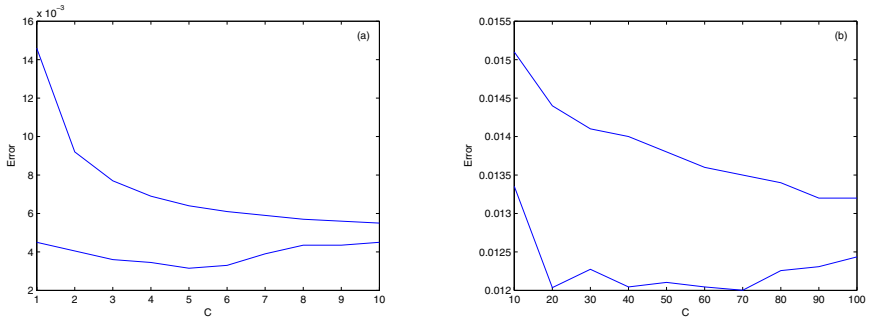

Fig. 2. The influence of parameter C on model generalization ability. Above line is training error, bottom line is testing error. (a) with Sys1 data set (b) with Sys3 data set

Parameter C in SVR is a regularized constant determining the tradeoff between the training error and the model flatness. The following experiment demonstrates the influence of parameter C on model generalization ability. It is conducted as experiment of prediction to simulate the practical process of software reliability growth prediction. The results are shown in Fig. 2. (a) and (b). We can see that with the increase of parameter C, the training error declines gradually, that is to say, the model fits the training data set better and better. However, as for testing, first the testing error declines gradually because the complexity of model is suitable for the need of testing data set more and more. And then the testing error raises because of overfitting problem. The problem of tradeoff between the training error and the model flatness can be solved by

cross-validation technique which divides the training samples into two parts: one for training and another for validation to obtain satisfied generalization ability.

4 Conclusions

A new technique for software reliability growth modeling and prediction is proposed in this paper. SVR is adopted to build SRGM. From the experiments we can see that the proposed SVR-based SRGM has a good performance, no matter generalization ability or predictive ability, the SVRSRG is better than conventional SRGMs. Experimental results show that our approach offers a very promising technique in software reliability growth modeling and prediction.

Acknowledgements

This work was fully supported by a grant from the NSFC (Project No. 60275002) and the Project-sponsored by SRF for ROCS, SEM.

References

1. Musa, J.D.: Software Reliability Engineering: More Reliable Software, Faster Development and Testing. McGraw Hill (1999)
2. Lyu, M.R.: Handbook of Software Reliability Engineering. IEEE Computer Society Press and McGraw-Hill Book Company (1996)
3. Guo, P., Lyu, M.R.: A Pseudoinverse Learning Algorithm for Feedforward Neural Networks with Stacked Generalization Application to Software Reliability Growth Data. Neurocomputing, **56** (2004) 101–121
4. Cortes, C., Vapnik, V.: Support-vector Network. Machine Learning, **20** (1995) 273–297
5. Joachims, T.: Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms. Kluwer (2002)
6. Xing, F., Guo, P.: Classification of Stellar Spectral Data Using SVM. In: International Symposium on Neural Networks (ISNN 2004). Lecture Notes in Computer Science. Vo. 3173. Springer-Verlag (2004) 616–621
7. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag, New York (1995)
8. Karunanithi, N., Whitley, D., Malaiya, Y.: Prediction of Software Reliability Using Connectionist Models. IEEE Transactions on Software Engineering, **18** (1992) 563–574