A New Methodology for Software Module Characterization

Miyoung Shin and Yunseok Nam

Electronics and Telecommunications Research Institute

Email: shinmy@etri.re.kr, ysnam@etri.re.kr

Abstract

The primary aim of this paper is to introduce and illustrate a radial basis function (RBF) modeling approach for software module characterization, as an alternative to current techniques. The RBF model has been known to provide a rich analytical framework for a broad class of so-called pattern recognition problems. Especially, it features both nonlinearity and linearity which in general are treated separately by its learning algorithm, leading to offer conceptual and computational advantages. Furthermore, our new modeling methodology for determining model parameters has a sound mathematical basis and showed very interesting results in terms of model consistency as well as performance.

1. Introduction

Software developers and managers of large systems are interested in assessing, throughout the life cycle from requirements analysis to system testing, whether the system will be completed within budget, have the desired quality and be ready for delivery as planned. In spite of impressive advances in software engineering technology over the past thirty years, these objectives have been rarely met. Recently there has been a growing realization among software professionals that the use of metrics must become an integral part of software development in order to increase productivity, improve quality, achieve desired reliability and adhere to the project schedule. In fact, a whole field of software metrics [1] has emerged to deal with many of the issues raised in this context. Metrics are now employed by industry and government to evaluate and control software products and processes. The metrics collection, however, is only a part of what is needed. The additional one is appropriate analytical techniques which can be employed for extracting useful information from the metrics data. Traditionally statistical methods such as regression analysis are used. There also have been studies that employ classification trees to generate metrics based rules for identifying potentially faulty models [2].

The primary aim of this paper is to introduce and illustrate a radial basis function (RBF) modeling approach. In Section 2, we discuss some important considerations in software module related data analyses. Section 3 presents the basic structure and parameters of the RBF model, our new modeling methodology and the algorithm. In Section 4 we give a module characterization case study using RBF for a well-known US-NASA metrics database. Finally we make some concluding remarks in Section 5.

2. Considerations in Software Data Analyses

In analyzing software metrics data, two classes of modeling problems often arise: the estimation and classification problems. The estimation model is the one to estimate the number of defects in the module

when given certain characteristics of a module. On the other hand, the classification model is to classify the module into such classes as critical or noncritical, again based on the known values of module characteristics. The modeling process can be summarized as developing an empirical model of an input-output mapping on the basis of limited evidences about the nature of this mapping. In this process it is important to make the following considerations. That is, we should seek a model which would be a good fit, but not an exact fit, to the experimental data. Also, we should build a model that captures the underlying relationship so that it could make good predictions for the unknown output on some future observations of the inputs. The first consideration, which discourages overfit, is captured by a term called training error while the second consideration, which encourages good predictability for future inputs, is quantified via a term called generalization error. Thus, our objective is to develop an empirical characterization which has small generalization error as well as small training error. These considerations are more formally known as the bias-variance dilemma.

Now we state the module characterization problem mathematically. Suppose we are given an experimental data set D, in which both inputs and their corresponding outputs are made available,

$$D = \{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in R^d, y_i \in R, i = 1,..., n\}$$

The problem is to find a mapping function from the d-dimensional input space to the 1-dimensional output space based on the data D. From a characterization perspective, we seek an empirical model that provides the best fit to this training data subject to ensuring the best predictions of y' for future $\mathbf{x}'$ while minimizing model complexity. For the future, only $\mathbf{x}'$ would be given and we would want to seek the value of y' by using the empirical model developed from D.

We now use a concrete example to illustrate these considerations. Suppose we have data about twenty-five modules. The data for each module consist of size, complexity and number of defects. The inputs ($\mathbf{x}$) here are size and complexity and the output (y) is number of defects. The underlying conjecture is that there exists in the real world some relationship between size, complexity and defects. However, we do not know the form of this input-output mapping, known only to mother nature. The objective is to develop an empirical model that captures this relationship based on these twenty-five data points reasonably well. Our aim is to learn from the data about these modules and use what we learn to estimate the number of defects in a yet unseen module for which we would only know the size and complexity. In summary, we want a good enough empirical model for the current data which also provides good estimates for future modules.

3. Radial Basis Function Model

The RBF is a nonlinear model where the output estimate $\hat{y}$ for input vector $\mathbf{x}$ is represented by the following functional form :

$$\hat{y} = f(\mathbf{x}) = \sum_{j=1}^{m} w_j \phi_j(\mathbf{x}) = \sum_{j=1}^{m} w_j \phi(\|\mathbf{x} - \boldsymbol{\mu}_j\| / \sigma_j).$$

Here $\phi(\cdot)$ is called a basis function and popularly taken to be the Gaussian, given by $\phi(r) = \exp(-r^2/2)$. The $\boldsymbol{\mu}_j$ and σ_j are called the center and width of j th basis function, respectively, w_j is the weight associated with the j th basis function output and m is the number of basis functions. Thus, the RBF model is fully determined by the parameters $P = (m, \boldsymbol{\mu}, \sigma, \mathbf{w})$.

For parameter determination, we wish to find a parsimonious model, i.e., one with as few parameters as possible, which also possesses good generalization

ability. Since m is the predominant parameter, we wish to find a model with smallest m that provides a good fit to training data and exhibits good predictions for future inputs. Our methodology for the parameter determination is summarized below; as the SG algorithm; the details are given in [3,4].

[***SG Algorithm***]: For the given data $D = \{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in R^d, y_i \in R, i = 1,..., n\}$,

Step 1: Select a set of values for width σ and a value for the RC measure δ. Heuristically we take $0 \le \sigma \le \sqrt{d/2}$ where d is the number of input variables and δ to be 0.1% to 1.0%.

Step 2: Determine m that satisfies the δ criterion. This step involves singular value decomposition of the interpolation matrix computed from the $n \times d$ input data matrix for the given σ.

Step 3: Determine centers (μ) of the m basis functions which maximize structural stabilization for the selected model complexity, m. This step involves use of QR factorization.

Step 4: Compute weight (w) using the pseudo inverse and estimate output values.

The above algorithm leads to the following design procedure. Select δ and σ values, determine the smallest m for each case. Then obtain the basis function centers and compute the weights. This completes the RBF design. To evaluate each design, next compute the training and validation CE's and choose the model with smallest validation error. The final step is to evaluate test error for each design. Amongst the delta values considered, select the final classifier based on considerations of complexity (m) and the test error.

4. NASA Case Study

In this section we describe how the SG based RBF methodology described above was employed to develop models for characterizing software modules. The data sets are taken from the US-NASA metrics database, a highly respected repository of software metrics data.

Datasets: The database contains software project information such as module level design and code metrics as well as some process metrics. Some results about this case study were reported earlier in [5]. For our study we chose eight software systems and three design related metrics: function calls from module, function calls to module and number of input/output parameters. The dataset used consists of 796 modules. For RBF model development, it was randomly divided into three sets: training (398), validation (199) and test (199). The training set is used to design the RBF classifiers and the validation set to select the best performing classifier as a final model. The performance on future data in real use is estimated by the error on test set.

Design Metrics Based Module Characterization: From a dataset of 398 modules, several RBF models have been developed for specified values of δ and σ. Then we evaluate classification error (CE) which is the ratio of the number of incorrectly classified modules to the total number of modules. For this study, we first take δ=1% and six values of σ=0.2(0.2)1.2. The resulting models, each having minimum complexity satisfying $\delta \le 1\%$ and are listed in Table 1. The training and validation errors for the six models (A to F) are also shown. Using minimum validation error criterion, the best classifier is C with m=4 and σ=0.6, training CE=25.63% and validation CE=20.60%.

Now we compute test error for this model which is found to be 28.14%. In other words, this model is likely to classify future modules correctly about 81.80% of

Table 1: Parameters and error values with minimum m for design metrices based RBF classifiers (δ=1%)

Classifier	σ	m	CE(%)	
			Training	Validation
A	0.2	9	25.88	22.11
B	0.4	5	24.37	22.11
C	0.6	4	25.63	20.60
D	0.8	3	27.14	22.11
E	1.0	2	26.13	22.11
F	1.2	2	26.13	21.61

Table 2: Training, validation and test CE for design metrics based RBF classifiers

δ (%)	σ	m	CE(%)		
			Training	Validation	Test
1.0	0.6	4	25.63	20.60	28.14
0.5	0.4	7	24.37	22.11	28.64
0.1	1.0	4	25.63	21.11	26.13

the time. From the perspective of software engineering and in light of the fact that software data tend to be very nosiy, this result is quite satisfatory. The above process we repeated for δ=0.5% and δ=0.1%. The final models for these cases along with the three CE terms are shown in Table 2. On the basis of these values, the RBF models for δ=0.1% yields the best value of test error, viz. 26.13%. This could thus be considered as our final module classifier model.

5. Concluding Remarks

In this paper we have introduced an innovative new methodology for characterizing fault prone software modules using RBF classifiers. Our methodology yields a systematic, objective and consistent approach. In other words for given data set and specified degree of closeness, represented by the δ measure, the final design is always the same. This is an extremely powerful and innovative feature resulting from the mathematical underpinings of our algorithm. The current methodologies do not possess such properties. We believe that our new approach would provide an easy to use and practical methodology for early detection of troublesome module. We hope that the software engineering community will employ our approach to develop cost effective, high quality software systems.

References

[1] Fenton, N. and Pfleeger, S. Software Metrics: A Rigorous and Practical Approach, PWS Publishing Company, 1997.

[2] Porter, A. and Selby, R. Empirically guided software development using metric-based classification trees. *IEEE Software* 7(3):46-54, March 1990.

[3] Shin, M. Design and Evaluation of Radial Basis Function Model for Function Approximation, Ph.D. Dissertation, Syracuse Univeristy, 1998.

[4] Shin, M. and Goel, A. L. Radial basis function model development and analysis using the SG algorithm. Technical Report, Dept. of EECS, Syracuse Univerisity, June 1998.

[5] Shin, M. and Goel, A. L. A RBF Classifier Based Framework for Software Quality Evaluation. *Proceedings of the International Conference on Computational Intelligence for Modeling, Control and Automation*, Vienna, Austria, Feb. 17-19, 1999.