An Approach of Software Quality Prediction Based on Relationship Analysis and Prediction Model

Wei Peng
Zhongshan College, University of Electronic Science and Technology of China
Zhongshan 528402, China
Email: yuanwei318@126.com

Lan Yao
IBM China Global Delivery Center
Chengdu 610041, China
Qiang Miao
School of Mechatronics Engineering of UESTC
Chengdu 610054, China

Abstract—By predicting the quality of the software that will be formed in the early stage of development, faults brought in at the phase of design will be found out early in order not leave them in the software product. Furthermore, it will be easy for designers to adopt appropriate plans based on specific expectations of the target software. However, the traditional prediction models have following shortages: 1) the relationship between attributes and metrics effectively cannot be expressed; 2) lack of the ability to process data both qualitatively and quantitatively; 3) not appropriate to the case with uncompleted information. In this paper, a model built based on regression analysis and fuzzy neural network is proved to be good at quality prediction of object-oriented software.

Keywords-software quality prediction; regression analysis; fuzzy neural network; quality attributes; CK metrics

I. INTRODUCTION TO SOFTWARE QUALITY PREDICTION METHODS

A. Background of Software Quality Prediction

With the rapid development of computer hardware, requirement of performance of software becomes greater, where high performance is expected. Meanwhile software development and maintenance cost are expected to be reduced and development time becomes shorter.

If we can realize quality prediction in the early stage of software development, this can be more or less an assessment of final design of software product quality, which can optimize its design so as to satisfy expectation. Furthermore, quality of software products can be used to identify the “best” design, the kind that can produce high-quality software design. It is clear that software quality prediction at the early stage of software development is the ultimate way to ensure quality of software products, shorten the development cycle and reduce cost of software development and maintenance, which has very important practical significance and value.

Based on the traditional technologies such as Bayesian Belief Networks[1], Regression Trees[2], Fuzzy Subtractive Clustering[3], and Artificial Neural Network[4], software prediction models can determine the uncertainty relationships between the quality properties of a software and its internal attributes to some extent. However, the accurate and effective description of the causal relationship between software quality properties and internal attributes, as well as processing of incomplete information obtained in the early stage of software development and coexistence of various data patterns are the main challenges in software quality prediction.

B. Description of Object-oriented Software Quality and Internal Attributes

It is well known that the quality of software is an “easy-aware, difficult to define, and can not be measured”[5] concept. In order to explain such a concept, a lot of work has been done recently. ISO/IEC 9126 Standard provides a framework for assessing quality of software and definition of 6 software quality properties, including Functionality, Reliability, Usability, Efficiency, Maintainability, Portability. Different users focus on different software quality properties.

An object-oriented software usually contains a large number of internal attributes, which can provide more accurate and comprehensive descriptions of software’s internal structure and nature. To date, there are large numbers of metrics have been posed, among which Chidamber & Kemerer (C&K) metrics[6] are proved and recognized to be a typical and useful set of Object-oriented (OO) software metrics, including WMC (Weighted Methods for per Class), DIT (Depth of the Inheritance Tree), NOC (Number Of Children), CBO (Coupling Between Object classes), RFC (Response For a Class).

It is not difficult to imagine that there exists certain cause and effect relationship between some of software internal attributes and specified quality properties. Therefore, different software internal attribute sets have different impact on specified software quality property. The establishment of software quality prediction model is to solve following two problems:

(1) To determine cause and effect relationships between software internal attributes and specified software quality properties.

(2) To precisely describe this kind of cause and effect relationships.

The aforementioned two issues can be solved by selection of software internal attributes and construction of effective relationship model for description of selected software internal attributes and software quality properties, as shown in Figure 1.
If software prediction machine which has aforementioned two functions can be established, it is possible for any software entity to determine external quality properties according to its internal attributes information, and realize software quality prediction.

II. OBJECT-ORIENTED SOFTWARE INTERNAL ATTRIBUTES AND RELATIONSHIP ANALYSIS BASED ON LOGISTIC REGRESSION ANALYSIS

To address the first question, a logistic regression model based on maximum likelihood estimation is introduced, which can realize analysis of relationships between independent variables and binary dependent variables. Therefore, it is able to evaluate impact of software internal attributes on objective software quality.

A multivariate logistic regression model is defined as:

$$\pi(x_1, x_2, \ldots, x_n) = \frac{e^{(b_0 + b_1x_1 + \cdots + b_nx_n)}}{1 + e^{(b_0 + b_1x_1 + \cdots + b_nx_n)}}$$

Here, $x_i$, $i = 1, 2, \ldots, n$ are independent variables. $\pi$ is a binary dependent variable determined by $n$ independent variables, which can only be 0 or 1. $b_i$, $i = 0, 1, 2, \ldots, n$ are $n + 1$ coefficients of the model, which is determined in the initial stage of modeling. In order to measure and describe the relationship between each independent variable and specified dependent variable, a number of statistical indicators are defined here: estimation of model coefficient (Coefficient), accuracy of coefficient estimation (p-value), matching level between model and data ($R^2$), impact of independent variable on dependent variable (Odds Ratio).

By using the aforementioned CK metrics as independent variables, and 6 software quality properties as dependent variables, we can establish 6 logistic regression models respectively. In addition, experience data of CK metrics extracted from practical engineering software, also known as descriptive statistics, are treated as independent variables of regression model. Furthermore, convert fuzzy number of software quality properties obtained from expert knowledge to binary variable (e.g., high reliability and low efficiency can be expressed respectively as: $\pi_{Reliability} = 1$, $\pi_{Efficiency} = 0$) and treat it as dependent variable of regression model. Then, statistics can be obtained to describe relationships between CK indicators and specified software quality properties. After that, according to empirical threshold value of the statistic indicators, we can conclude that the specified software quality properties have close cause-and-effect relationship with those CK metrics within the domain of thresholds, while they have little relationship with other CK metrics outside the domain of thresholds.

III. REALIZATION OF OBJECT-ORIENTED SOFTWARE QUALITY PREDICTION BASED ON FUZZY NEURAL NETWORK

A. Fuzzy Neural Network

Since there exists less prior information at the beginning of software development (e.g., only 8 of 10 input variables have prior information), the established software quality prediction model should be able to realize effective prediction of software quality properties under insufficient information. In addition, the acquired data information may have different formats (it may be accurate, such as historical data extracted from practical software engineering, and it may be fuzzy such as expert knowledge and experiences). Therefore, it is necessary to enhance adaptability and compatibility of software quality prediction model to different data formats.

With excellent self-learning capability and adaptability, neural network can describe complicated and uncertain relationships between variables. However, it is hard to deal with fuzzy information and knowledge explanation. On the other hand, fuzzy logic system can express fuzzy information and utilize expert knowledge and experiences, with poor learning capability and adaptability.

B. Realization of Fuzzy Neural Network Based Prediction Model

According to aforementioned theoretical analysis of neural network and fuzzy logic, it is possible to build a collaborative integrated prediction model based on fuzzy neural network, and realize software quality prediction under different data formats and insufficient information.

Figure 2 shows the integrated prediction model based on fuzzy neural network before training. This model utilizes the idea of fuzzy adaptive learning control network and is composed of five layers of neurons from bottom to top. Three software internal attributes including DIT, WMC and RFC are nodes in the first layer, which are selected by logistic
regression models. Software quality properties such as reliability and efficiency are treated as nodes in the fifth layer. Nodes at layer 2 and layer 4 are term nodes acting as membership functions to represent the terms of the respective linguistic variables. The third layer is fuzzy logic rule layer. Links between the third and the second/fourth layers describes the precondition and conclusion of each rule. Each node in layer three represents a fuzzy rule derived from precondition of the second layer and conclusion of the fourth layer before training. Nodes in the third and fourth layers are all connected. Similarly, nodes in the second and third layers are all connected, too. However, there is no more than one link between each rule node and the corresponding term node of the same linguistic node.

The initial set of input and output samples is shown in Tables 1 and 2. Specifically, values of 0 and 1 in the input sample set represents fuzzy terms “low” and “high”, which are given by expert knowledge. Normalized quantitative data in the output set are from historical data of system.
TABLE 1. INPUT SAMPLE SET

<table>
<thead>
<tr>
<th>INPUT</th>
<th>SAMPLE 1</th>
<th>SAMPLE 2</th>
<th>SAMPLE 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIT</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>WMC</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>RFC</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE 2. OUTPUT SAMPLE SET (THE EXPECTED OUTPUT SET)

<table>
<thead>
<tr>
<th>OUTPUT</th>
<th>SAMPLE 1</th>
<th>SAMPLE 2</th>
<th>SAMPLE 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>0.50</td>
<td>0.70</td>
<td>0.90</td>
</tr>
<tr>
<td>Efficiency</td>
<td>0.45</td>
<td>0.88</td>
<td>0.23</td>
</tr>
</tbody>
</table>

Through training of sample data, the structure of fuzzy neural network can be determined in the process of learning. The simulation results show that fuzzy neural network can converge to expected output set quickly as shown in Table 3, given that mean squared error is 0.01.

At the same time, as shown in Figure 3, the structure of fuzzy neural network can be adjusted automatically after training. Each node in the third layer represents a mutually independent fuzzy logic rule. For example,

Rule 1: If DIT is "shallow", WMC is "small", and RFC is "large", then reliability is "high" and efficiency is "high".

Rule 2: If DIT is "deep", WMC is "small", and RFC is "small", then reliability is "high" and efficiency is "low".

Rule 3: If DIT is "deep", WMC is "large", and RFC is "small", then reliability is "low" and efficiency is "low".

TABLE 3. THE ACTUAL OUTPUT SET

<table>
<thead>
<tr>
<th>OUTPUT</th>
<th>SAMPLE 1</th>
<th>SAMPLE 2</th>
<th>SAMPLE 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>0.5513</td>
<td>0.6703</td>
<td>0.8756</td>
</tr>
<tr>
<td>Efficiency</td>
<td>0.4566</td>
<td>0.9046</td>
<td>0.2941</td>
</tr>
</tbody>
</table>

Through training with expert knowledge or experiences, the proposed prediction model got all the existed fuzzy rules corresponding to specified software system. The simulation results show that these rules effectively and accurately describe the potential mapping relationships between software internal attributes and quality properties. So, the impact of different software internal attributes on specified software quality properties can be predicted by analysis of the above relationships.

C. Processing Method on Insufficient Information and Multiple Data Formats

Generally, sample data used for training accurate and quantitative data extracted from experience system. For example, WMC is 20, RFC is 35 and reliability is 80%. They can be used as samples for input and output and are suitable for the proposed prediction model. On the other hand, some sample data are fuzzy and qualitative data from expert knowledge. For example, if WMC is "large", RFC is "small" and DIT is "deep", then reliability is "high" and efficiency is "low". In this situation, it is appropriate to add these fuzzy data as fuzzy rules into prediction model without training. Furthermore, there is another situation that needs to be treated, which is insufficiency in sample data. For example, we only know that if WMC is 20 and RFC is 35, then reliability is "high" and efficiency is "low", and there is no prior information on DIT. In this case, it is necessary to add fuzzy information of DIT to implement training of the proposed neural network. Sample data after adding values can be shown as two rules: if WMC is 20, RFC is 35 and DIT is "deep", then reliability is "high" and efficiency is "low"; if WMC is 20, RFC is 35 and DIT is "shallow", then reliability is "high" and efficiency is "low". It should be noted that DIT will be the input of the second layer. It will enter rule layer together with WMC and RFC that are input of the first layer and are processed by membership functions of the second layer, and will continue following operations.

IV. CONCLUSIONS

By constructing a prediction model based on fuzzy neural network, it not only realizes effective prediction of software quality, but also solve problems exist in the traditional prediction methods. The proposed method can provide comprehensive and effective description on uncertain cause-and-effect relationships between software quality properties and other factors that have impact on them. In addition, it can deal with insufficient data and multiple data formats. Furthermore, to simplify the structure of fuzzy neural network based prediction model and improve training efficiency, logistic regression method is introduced, which has strong capability of relationship analysis and can enhance software prediction capability.

ACKNOWLEDGMENTS

This research was partially supported by the Youth Foundation of Zhongshan College, University of Electronic Science & Technology of China and Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No.20070614023).

REFERENCES
