Filter-Wrapper based Feature Ranking Technique for Dynamic Software Quality Attributes

Siti Sakira Kamaruddin[1], Jamaiah Yahaya[2] Aziz Deraman[3], and Ruzita Ahmad[4]

[1]*Universiti Utara Malaysia, Malaysia, sakira@uum.edu.my*
[2]*Universiti Kebangsaan Malaysia, Malaysia, jhy@ftsm.ukm.my*
[3]*Universiti Kebangsaan Malaysia, Malaysia, ad@ftsm.ukm.my*
[4]*Universiti Utara Malaysia, Malaysia, rita_azura@yahoo.com*

ABSTRACT

This article presents a filter-wrapper based feature ranking technique that is able to learn and rank quality attributes according to new cases of software quality assessment data. The proposed feature ranking technique consists of a scoring method named Most Priority of Feature (MPF) and a learning algorithm to learn the software quality attribute weights. The existing ranking techniques do not address the issue of redundancy in ranking the software quality attributes. Our proposed technique resolves the redundancy issue by using classifiers to choose attributes that shows high classification accuracy. Experimental result indicates that our technique outperforms other similar technique and correlates better with human experts.

Keywords: Software quality assessment model; feature ranking technique; Dynamic software quality attribute ranking.

I INTRODUCTION

Software monitoring and assessment is defined as the establishment and operation of appropriate model, methods, systems and procedures necessary to monitor, compile, and analyze data regarding the condition of software being used in the organization. Effective software quality monitoring and assessment is very important to ensure the reliability of the software. International Organization for Standardization (or ISO) defines software as "all or part of the programs, procedures, rules, and associated documentation of information processing system". Software product is defined as "the set of computer programs, procedures, and possibly associated documentation and data designated for delivery to a user" (ISO/IEC9126, 1996). Quality is defined as "the totality of features and characteristics of a product or services that bear on its ability to satisfy stated or implied needs" (Jenner, 1995).

The quality of software can be assessed according to three categories of assessments: internal measures, external measures and quality in use measures (ISO/IEC 9126, 1996). Internal measuring is the evaluation based on internal attributes typically static measures of intermediate products and external measuring is based on external attributes typically measuring the behavior of the code when executed. While the quality in use measures include the basic set of quality in use characteristics that effect the software. These characteristics include effectiveness, productivity, safety and satisfaction. Among these measures, there are several measures that are subjective and unquantifiable to be measured objectively especially software measures that are related to human aspects.

Previous software quality models (Dromey, 1998; Suryn et al., 2002; Suryn et al., 2003) measure the software quality based on technical aspects and theories. The Pragmatic Quality Factor (PQF) model proposed in (Yahaya et al., 2008a) is a comprehensive measurement which consist all the technical aspects with the addition of the human aspects. However, PQF operates on a set of static quality attributes and measures. Thus the model is not flexible and incapable to capture current and future requirements. In this work, we propose a dynamic software quality assessment model using feature ranking technique to enable the model to learn and rank software quality attributes based on previous software quality assessment dataset. In resolving the issue of redundancy and subjectivity of measurement our model focuses on the learning process using classifiers on attribute weights given by quality assessors. Weights are relatively very subjective and difficult to be determined distinctively therefore, the proposed model is designed to identify and recommend to the environment attributes that is given high weights based on previous cases of software quality assessment data.

The rest of the paper is organized as follows. In section II, related work on the application of feature ranking techniques in the assessment of software quality are discussed. Section III presents the proposed filter-wrapper based feature ranking technique. Experimental settings and result are presented in Section IV. Section V presents conclusion of the work.

II FEATURE RANKING TECHNIQUE FOR SOFTWARE QUALITY ASSESSMENT

Feature Selection (FS) is a process of selecting relevant features for building learning models and it is used to remove less important features from the data set. The aim of FS is to improve the classification performance and to offer and improved realization of the fundamental process that created the data (Guyon et al., 2002). FS has been applied to software quality assessment model to identify the most promising

quality attributes that is able to increase the efficiency of software classification (Gao et al., 2009; Huanjing et al., 2010).

FS techniques can be divided into two categories which are feature ranking techniques and feature subset selection techniques. Feature ranking techniques assess attributes individually and rank the attributes according to their individual predictive or classification power. Whilst, feature subset selection techniques select the subset of attributes that collectively have good predictive or classification capability. There are two different approaches to FS which are filter approach and wrapper approach. By using the filter approach, the feature is selected independent of the learning method which means ignoring the induction algorithm to assess the merits of features from data. Whilst, in wrapper approach the features are selected using the same learning algorithm that will be used for learning on domain represented with the selected features.

In Huanjing et al. (2010), an empirical investigation of filter based feature ranking techniques (FRT) for software quality classification was performed. Here, a number of filter based feature ranking techniques were used to predict the quality of software modules as either fault prone or not fault prone. They showed that the performance of the filter based feature ranking techniques varies depending on the classifiers and the performance metrics used to measure the efficiency of the classification results. Gao et al. (2009) applied Filter Attribute Selection (FAS) to perform attribute selection before classification. FAS is a process of selecting a subset of relevant features for building learning models. The idea behind this technique is to remove less important features from the training data set. They tested four FAS techniques and concluded that the technique proposed by (Khosgoftar et al., 2003) i.e.; the Kolmogorov-Smirnov Correlation Based Filter (KSCBF) performed better for the software quality problem. Therefore in this work we compare our proposed technique to KSCBF.

In our work a feature ranking technique is proposed which combines the filter and the wrapper approach. The proposed technique scores each software quality attributes according to a proposed scoring method called Most Priority of Feature (MPF). In the case where there are more than one attribute with same MPF score, the attribute is trained and tested for classification. In our case, the MPF scoring is considered as the filter approach and, we consider the learning process as the wrapper approach where we investigate how well the attribute perform in the classification task. As a contrast to previous work in software quality feature selection methods, our proposed method focuses on learning the weights that is assigned by assessors that indicates the relevancy of a particular software quality attributes.

III FILTER-WRAPPER BASED FEATURE RANKING TECHNIQUE

This section describes our proposed Feature Ranking Technique (FRT). As mentioned earlier our proposed FRT applies both the filter and wrapper approach. We propose a two phase ranking technique. The filter approach is implemented in Phase I where the MPF score is calculated for all attributes and the scores are checked and ranked. If there is more than one highest MPF score, Phase II is executed where the wrapper approach is implemented. The proposed FRT is called FRTMPF for easier referencing. Figure 1 shows the proposed FRTMPF.

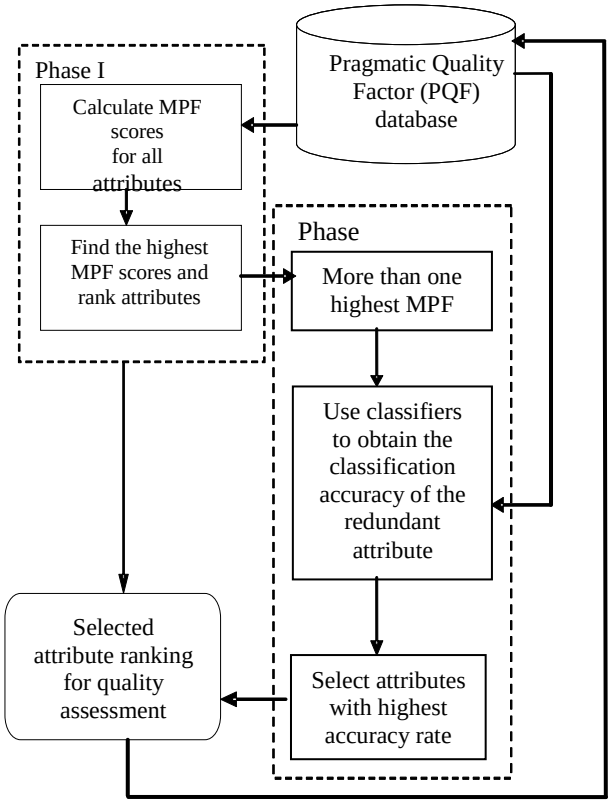

Figure 1. The Proposed FRTMPF.

In phase I, the software quality attribute data are obtained from PQF database which contains over 1000 cases of software quality assessment data collected from previously performed case studies as reported in Yahaya et al. (2008b) and Yahaya et al., (2010). The behavioral attributes collected from the research studies are efficiency, maintainability, functionality, portability, reliability, usability, user factor and integrity. Table 1 shows an example of the software quality attributes and the weights assigned by the assessors (5 is the highest weight score).

Table 1. Example Of Software Quality Attributes With Assigned Weights.

Attribute	*Weights*
Efficiency	4.08
Functionality	3.69
Maintainability	2.66
Portability	3.55
Reliability	3.36
Integrity	3.83
Usability	2.95
User Factor	3.67

The attributes and weights as shown in Table 1 is used to calculate the MPF scores for each attribute. The MPF score is calculated using Eq. 1.

$$MPF = \delta yj \sum (max_{xi} \bullet f max_{xi}) \quad (1)$$

where, *xi* is the weight assigned by assessors to each software quality attributes, δyj is the standard deviation for the selected quality attributes, max_{xi} is the maximum weight of the selected attributes in the database and $f max_{xi}$ is the frequency of maximum weight of the selected attributes.

The next step involves the ranking of the MPF scores. If there are two or more attributes that produces the same MPF scores, the second phase is be implemented. In the second phase, the data from the PQF database corresponding to the attributes which have same MPF scores are obtained to be used for training the classifiers. As pointed out in Huanjing et al. (2010), the choice of classifier may affect the classification accuracy; therefore in this work we have used two classifiers namely Random k-labelsets (Rakel) and k-nearest neighbor (MLkNN) to avoid biasness. The attribute that produces the highest classification accuracy is then chosen to be ranked first. The final ranking of the software quality attributes are stored in the PQF database for future software quality assessments. Table 2 presents the algorithm to accomplish this task.

Table 2. Algorithm of FRTMPF.

Steps	Algorithm
1	Get the software quality attributes and weights from the PQF database
2	Use the weight value to calculate the MPF scores for all attributes
3	Sort and rank the attributes according to the highest MPF scores.
4	If there are more than one highest MPF score For each of the attribute with same MPF score Begin a. Get the corresponding data and weights from the PQF database b. Input the data into two classifiers c. Calculate the average classification accuracy of the two classifiers d. Output the average classification accuracy End
5	Select the attribute with the highest classification accuracy
6	Output the ranked software quality attributes

IV EXPERIMENTAL SETTINGS AND RESULTS

This section explains the settings of our experiment. Its aim is to evaluate the effectiveness of the proposed FRTMPF technique in terms of ranking the software quality attributes. The experiment was conducted using a dataset which were taken from previous cases of software quality assessment data. It contains over 1000 cases of software quality assessment data. To simplify our experiment we have included only the high level attributes as shown in Table 1. As mentioned in the previous section the MPF scores were calculated for each attribute. Table 3 shows the MPF scores for the attributes.

Table 3. MPF Scores For Each Software Quality Attribute.

Attribute Id	*Attribute Name*	*MPF Score*
P008	User Factor	47.34
P004	Maintainability	45.00
P007	Usability	45.00
P002	Functionality	40.63
P005	Portability	34.38
P006	Reliability	26.10
P003	Integrity	22.70
P001	Efficiency	17.81

From the result of the calculated MPF scores, it can be seen that there are two attributes that have the same MPF scores that is *Maintainability* and *Usability* which both MPF scores are 45. Therefore, to resolve this issue, we implement the wrapper approach where the data related to these attributes are trained and tested for classification task with the weights as the target. As was discussed previously, two classifiers namely RAKEL and MLkNN were employed in this task. The area under the curve (AUC) performance metric was used to calculate the classification accuracy. The classification accuracy of the two classifiers is then averaged to obtain the combined classification accuracy. The result of the classification accuracy is shown in Table 4.

Table 4. Classification Accuracy For Attributes With Same MPF Scores.

Classifiers	Maintainability Classification	Usability Classification
Rakel	0.8121	0.8131
MLkNN	0.9171	0.7616
Averaged classification accuracy	0.8646	0.7874

The result in Table 4 shows that the classification accuracy of Rakel is almost identical for both *Maintainability* and *Usability* attributes. The MLkNN classified the *Maintainability* attribute with 92% accuracy while *Usability's* accuracy was 79%. This finding supports the reports in Huanjing et al. (2010), that the accuracy is influenced by the choice of classifier. Therefore we have averaged the classification accuracy to avoid biasness towards a single classifier. The average classification accuracy depicts that the classification accuracy of the *Maintainability* attribute is higher than the *Usability* attribute with a difference of 8%. Therefore the *Maintainability* attribute are chosen to be

ranked higher than the *Usability* attribute. The resulting final ranking of the software quality attributes are reported in a tabular form. The final ranking is shown in Table 5.

Table 5. Final Ranking Of The Software Quality Attributes.

Attribute Id	*Attribute Name*
P008	User Factor
P004	Maintainability
P007	Usability
P002	Functionality
P005	Portability
P006	Reliability
P003	Integrity
P001	Efficiency

To access the effectiveness of the proposed technique we compared the ranking produced by our technique with that of the ranking produced using KSCBF. The result was reported with a line graph. In order to get a baseline for the comparison, we use the expert judgment as a benchmark. To accomplish this, we have collected a real software quality assessment case study using PQF for a health care organization as reported in Yahya et al. (2011). Here, the ranking was produced by experts that were selected through a collaboration process with users, developers and stakeholders of the system. Normally the stakeholders recommend and suggest particular users to become the user experts of the system as well as recommendations from the developers who were directly involved in the system development. The third category of experts are the independent assessor who are free from the control and constraints created by the system's users and developers. The ranking result of expert was averaged and was used as a baseline to compare the performance of the proposed FRTMPF and also the compared technique. Since the scores provided by KSCBF and the experts have a different scale compared to our proposed FRTMPF, all scores were normalized to hold a value between 0 to 1. A comparison graph were plotted to show the results. Figure 2 shows the graph for ranking result of FRTMPF, KSCBF and expert.

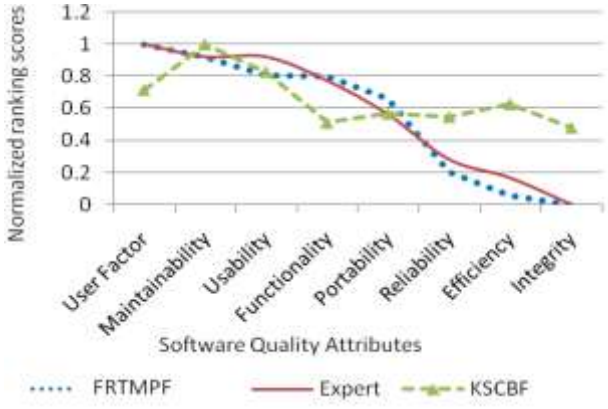

Figure 2. Ranking Results.

As can be seen in Figure 2, the line graph of the FRTMPF ranking is similar to the line graph of the expert as opposed to the line graph of KSCBF. Besides the use of a graph, we also calculated the correlation coefficients of FRTMPF and KSCBF, compared to experts rankings. These scores are shown in a tabular form as in Table 6.

Table 6. Correlation Of Produced Results With Human Ranking.

Compared techniques	Correlation Coefficient
FRTMPF to Expert	0.98
KSCBF to Expert	0.83

As shown in Table 6, the scores produced by FRTMPF associates better to expert ranking compared to KSCBF. The correlation coefficient shows that FRTMPF strongly correlates with the expert judgment with scores of 0.98 or 98% whereas KSCBF correlates 83% to the expert ranking with correlation score of 0.83. The correlation coefficient shows that FRTMPF strongly correlates with the expert ranking compared to KSCBF. The reason for high performance of the proposed FRTMPF is the calculation of MPF scores and the resolution of the redundancy issue in the attribute ranking. The combination of the filter and wrapper approach in the proposed FRTMPF has proved to be very promising.

V CONCLUSION

In this paper, we have presented a Filter-wrapper based Feature Ranking Technique with Most Priority of Feature (MPF) score. We have adopted the human aspect software quality attributes proposed in the PQF model (Yahaya et al., 2008a). One prominent advantage of our technique is the execution of weight learning algorithm using classifiers where the determination of weights are usually very subjective and normally left to the human judgments. The learning concept employed in this work also resolves the redundancy issues in the dataset. The experimental results are very encouraging. The proposed technique has depicted high performance result. With respect to

other ranking technique for software quality attribute, this technique depicted a higher correlation with human experts.

The proposed technique is very beneficial to the software industry and to other researchers as it can be used as a benchmark of a software quality model. It is an enhanced and improved quality model because of the dynamic attributes allowing the model to adapt itself to a particular environment based on previous quality attributes defined in the environment. In the current global and dynamic world, the importance of certain quality attributes can vary depending on the circumstances. Therefore, this model can fulfill the dynamic requirement of software quality model.

Nevertheless the proposed technique can be implemented as part of a more advanced intelligent software quality assessment model. Our future work will focus on the refinement of the proposed technique and evaluating it with larger and different datasets.

ACKNOWLEDGMENT

We would like to thank the Malaysian Ministry of Higher Education for providing the funding for this research through Fundamental Research Grant Scheme (FRGS).

REFERENCES

Dromey, G.R.(1998). Software Product Quality: Theory, Model and Practice. Software Quality Institute. Griffith University, Brisbane, Technical Report. Retrieved from http://www.sqi.gu.edu.au,.

Gao, K., Khoshgoftaar, T.M. & Wang, H.(2009). An Empirical Investigation of Filter Attribute Selection Technique for Software Quality Classification. In: The 2009 IEEE International Conference in Information Reuse and Integration, Las Vegas, Nevada, USA.

Guyon, I, Weston, J. , Barnhill, S. and Vapnik. V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422.

Huanjing Wang, Taghi M. Khoshgoftaar, Jason Van Hulse, (2010). A Comparative Study of Threshold-Based Feature Selection Techniques. GrC : 499-504.

ISO/IEC 9126 (1996): Software Quality Characteristics and Metrics-Part2: External Metrics. *Technical Report, ISO/IEC JTC1/SC7/WG6.*

Jenner, M.G. (1995). Software Quality Management and ISO 9001. New York: A Wiley/QED publication.

Khosgoftaar, T.M., Nguyen, L., Gao, K. & Rajeevalochanam, J. (2003). Application of an attribute selection method to CBR-based software quality classification.

Suryn, W., Abran, A. & April, A. (2003). ISO/IEC SQuaRE: The Second Generation of Standards for Software Product Quality. http://www.lrgl.uqam.ca/publications/pdf/799.pdf.

Suryn, W., Abran, A., Bourque, P. & Laporte, C.(2002). Software Product Quality Practices: Quality Measurement and Evaluation Using TL9000 and ISO/IEC9126. In: The 10th International Workshop, Software Technology and Engineering Practice (STEP).

Yahaya, J.H, Deraman, A. & Hamdan, A.R.. (2008a). Software quality from behavioural and human perspectives. *IJCSNS International Journal of Computer Science and Network Security*, 8(8), August 30, 53-63.

Yahaya, J.H., Deraman, A., & Hamdan, A.R. (2008b). Software Certification Implementation: Case Studies Analysis and Findings. *The 3rd International Symposium on Information Technology (ITSim2008)*, 26-29 August, 2008, Kuala Lumpur, Vol. III, pp. 1541-1548.

Yahaya, J.H., Deraman, A. & Hamdan, A.R. (2010) Continuosly Ensuring Quality Through Software Product Certification: A Case Study. In: The International Conference on Information Society (i-Society 2010), London, UK, 28-30 June ,.

Yahaya, J.H. & Deraman, A (2011). The Architecure of an Integrated Support Tool for Software Product Certification Process. Journal of Information & System Management Vol. 1 No. 1, March , 2011.