Meta-data to Guide Retrieval in CBR for Software Cost Prediction

Rahul Premraj, Martin Shepperd and Michelle Cartwright

Bournemouth University,
Empirical Software Engineering Research group,
Royal London House,
Christchurch Road, Bournemouth BH1 3LT,
United Kingdom

{rpremraj,mshepper,mcartwri}@bournemouth.ac.uk

Abstract. In recent years, case-based prediction has become a widely advocated technique for software cost estimation. Typically, such approaches are in essence k nearest neighbour methods supported by a case base of a feature vector per software project. Given that software project data is relatively rare – data sets may contain as few as 20 cases – it is common to find a relatively undiscriminating approach to the projects contained within the case bases. We hypothesise that meta-data generated by monitoring case performance can contribute to identifying misleading cases and improve predictions. This paper reports results of a pilot study in which we enriched our case base with meta-data to record performance behaviour of individual data sets. An external fuzzy model was used to classify individual cases as fit or unfit for future use. Misleading cases i.e. with poor predictive ability were seeded into the data set to assess the potential of the approach. Our results show that the model successfully identified the seed cases and refrained from using them during future retrievals.

Keywords. case-based prediction, software effort prediction, meta-data.

1 Introduction

Cost estimation in the early stages of a software project is recognised as a vital task in the field of software engineering (SE). Costs can be a crucial determinant of project plans and might potentially drive or influence the quality of the end product. Hence, early and accurate estimation is of much interest to all participants such as the project manager, senior management and clients. Over four decades of research have witnessed a variety of algorithmic techniques developed for prediction [11], but unfortunately their performance has been largely disappointing.

However, CBR has emerged as a relatively promising technique for software estimation. More encouraging results have been achieved when using analogy based approaches [17, 2, 7, 8, 12]. But, current levels of prediction accuracy still leave CBR tools generally unfit for commercial use, which calls for further research to improve the quality of results. Out of the various opportunities for refinement, one possibility is to exploit knowledge of the past performance of individual cases in contributing towards effective solutions, that is cost predictions for our problem domain.

Our observation of software engineering data sets indicates the lack of consistency [4] amongst similar cases (software projects). Different cases lying very close in the feature space (i.e. similar cases) may have markedly dissimilar cost values. This can be explained by differences in productivity of projects owing to external variations i.e. factors unaccounted by the feature set characterising the project state. The productivity of a software project can be influenced by a range of factors that together inject an inevitable degree of randomness in project development [6]. For example, a company acquires a project that is very similar to one completed in the past. This may result in substantial amounts of code being reused which will save considerable amount of cost. Hence, in this situation, it may be unwise in the future to use the problem case as a good analogy for similar external projects. We term such cases with unusual productivity levels as 'misleading' cases.

The underlying problem is that we do not have strong models of software development since it is a predominantly a human-centric, design process. Thus we do not have high confidence in

exactly which factors are important, let alone how to capture them as features. The consequence is this external variation. Note that although the focus of this paper is on software project effort prediction the problem of external variation is common to any problem domain in which we do not deep understanding.

Software engineering data sets are predominantly recorded using feature vectors [19] that provide limited flexibility to enable capture of rich information about individual projects that might be valuable during problem solving. Unfortunately, some sources of external variations may not find a place in such data sets [1]. This can be due to numerous reasons such as lack of consensus upon which factors do, or do not, influence productivity [18], unavailability of data, confidentiality, human error, flaws in measurement, unawareness of their existence or importantly, no provision to record them in the data set. These reasons also make *ad hoc* addition of features nearly an impossible task. Even very careful analysis of such data sets may not reveal underlying behaviour or occurrences during the course of the project that cause huge variations in productivity of seemingly very similar projects. As a result, adaptation of such cases is a challenging task since we are aware of the causes of variation in the particular project and this calls for application of alternative measures.

In this paper, we summarise the rationale, methodology and results from a pilot study that was conducted in an attempt to identify such 'misleading' cases by observing individual case performance patterns over a period of time and recording their behaviour as meta data in the case base. Then, such information can be exploited to quarantine cases that may lead to poor solutions in order to avoid their retrieval in the future.

The remainder of this paper is organised as follows. The next section highlights related work. Section 3 elaborates upon the proposed model and its composition. We then discuss the methodology adopted to conduct this study in Section 4. We discuss our results in Section 5 and lastly, indicate directions of future research in Section 6.

2 Related Work

Our approach was partially inspired by Reintartz *et al.* [15] and Leake and Wilson [10] who highlighted the importance of 'meta-data' or quality information for case base maintenance. 'Meta-data' [3] can be described as unindexed features that do not form a part of retrieval. It only provides additional information about other data. The idea behind this pilot study runs in parallel with case-base maintenance, which is a crucial step to continually attempt to preserve the efficiency of the system by examining existing and new content in the case base. Leake and Wilson [10] defined case base maintenance as:

> *"Case base maintenance implements policies for revising the organisation or contents (representation, domain content, accounting information, or implementation) of the case base in order to facilitate future reasoning for a particular set of performance objectives."*

The objective of this pilot study is in tune with the above definition as we aim to identify potentially 'misleading' cases in order to isolate them thereby reducing the chances of poor prediction. Iglezakis's case base maintenance policies [4] further confirm our connection with case base maintenance. These maintenance policies were recommended to give an indication of the quality of cases within the case base. Out of the five policies stated, the one relevant to our study is 'Consistency' that is defined as:

> **Consistency:** A given case is called consistent if and only if there is no other case whose problem description is a subset or the same of the given case's problem description and their solutions are different.

This translates into the fact that it is contradictory (or inconsistent) to have two or more cases with the same or similar problem description but markedly dissimilar solutions. This is analogous to our pilot as similar projects with different productivity levels will result in dissimilar costs leading to inconsistency in the case base.

But Reintartz [15] stressed the fact that currently, collecting maintenance related data is not a priority for CBR systems. With no meta-data available to exploit, it is difficult to reason or justify actions in order to cleanse the data sets and thereafter, measure any enhancements in performance.

Examples of approaches to implicitly filter out 'misleading' cases without the use of any meta-data include attempts by Skalak [16]. Using simple stochastic algorithms (Monte Carlo and Random Mutation Hill Climb), Skalak was able to considerably decrease the volume of cases and features stored without compromising the accuracy of the system. Kirsopp and Shepperd [9] further refined their case subset selection (CSS) technique aimed at finding the best subset of cases which when used as a training set resulted in the lowest aggregate error. Though effective, CSS posed the danger of over-fitting the training set that may prove ineffectual for new target problems. Also, the technique does not provide a fundamental explanation to reason why some cases are unfit for inclusion in the case base.

Leake and Wilson [10] further distinguished between CBR systems as introspective and non-introspective. The former systems are those that have or make available to themselves relevant 'meta-data' that will help them reflect upon their performance and then make appropriate adjustments with the objective of enhancing performance. The latter category comprises of those systems with no automated provisions to check the quality of cases existing in the case base.

By understanding the effects of differences in productivity in software projects, we believe that the above techniques argue in favour of collection of meta-data for case base introspection that can be exploited to isolate potentially 'misleading' cases that may otherwise have a negative impact on the accuracy of our prediction. The rationale behind our technique is further elaborated upon the in the following sections.

3 Meta-data to Guide Retrieval

Retrieval of 'misleading' cases will almost certainly result in poor prediction. This is simply because such cases are associated with unusually high or low cost values in comparison to other similar cases. Thus, predictions are likely to result in large errors which we intend to record to assess the performance of individual cases.

Our proposed model comprises of the combination of two techniques. The first is using ranking schemes (primarily based on the degree of error in prediction) to grade individual cases, while the other is a fuzzy model that will analyse associated ranks of cases and decide if the case is valuable or should be rejected. The following subsections are aimed at elaborating the two techniques used in our approach.

3.1 Ranking Cases

To measure the performance of the individual cases we applied two measures to rank cases. These measures independently grade cases from contrasting view points (explained in the following subsections) which we potentially considered useful in the process of evaluating case performance. The ranking couple for each case serves as a distinct measure of its credibility and consistency.

Mean Ordinal Ranking The degree of penalisation is partially reflected by the Mean Ordinal Rank (MOR) which is calculated as:

$$MOR = \frac{\sum_{i=1}^{m} \frac{rank}{n}}{m} \tag{1}$$

Here, rank is the ordinal rank (in increasing order) of the retrieved case's target value compared to the actual value of the target case. In other words, after the target case has completed, was the retrieved case the most appropriate? This is determined by ranking all cases in terms of the quality of their proposed solutions. The retrieved case will be assigned a *rank* that is equal to its

position on the list of cases sorted by decreasing order of corresponding proposed solution. Also, m is the frequency of retrieval of the particular case and n is the total number of cases in the case base at the point of retrieval.

MOR reflects the relative distance of the retrieved case to the target case as compared to other competing cases in the case base. Over several retrievals, if the ordinal rank of an individual case continues to remain high, MOR will remain at a high value which will confirm that the case under consideration has some unusual behaviour and should be used very cautiously in the future.

However, MOR in itself may be insufficient to render a case ineffectual since it does not account for the degree of error in prediction. For example, if a case may have a high MOR but the proposed solution has very low error, penalising it heavily would be naïve or inappropriate since the case-base contains fairly consistent analogies, but that some cases are more preferred . Hence, another ranking measure is required to complement MOR and be coupled with it in order to justify reluctance in using an individual case.

Balanced Mean Magnitude of Relative Error Balanced Mean Magnitude of Relative Error (BMMRE) [13] measures the degree of inaccuracy in prediction by the source case. BMMRE is given as:

$$BMMRE = \frac{\sum_{i=1}^{m} \frac{|\hat{y}-y|}{max(\hat{y},y)}}{m} \tag{2}$$

In equation 2, y is the actual value of the target case, while $\hat{y}$ is the predicted value. m is the number of times that the case has been retrieved. The error (or absolute residual) $|\hat{y}-y|$ is divided by the greater of the two values in order to normalise it, resulting in a value between 0 and 1. As with MOR, a value close to 0 suggests that the case has proved to be a good match, while a value close to 1 suggests the opposite.

3.2 Fuzzy Model

For deciding upon whether a case should be used for prediction or rejected, a fuzzy model [5] was developed using MATLAB. Fig. 1 depicts the framework of the model. It comprises of three inputs and produces a single output. The model computes the degree of membership for the values of a set of inputs and then analyses them using predetermined fuzzy rules. The output, Decision, is a real number between 0 (reject) and 1 (accept) that is rounded to a whole number. The three input values are MOR, BMMRE and lastly, frequency of retrieval. The following subsections briefly describe the membership functions of the different input and output values.

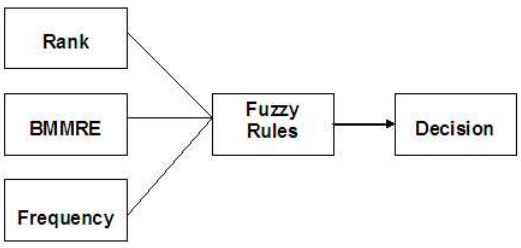

Fig. 1. Pilot Fuzzy Model

MOR Membership Functions MOR has 3 membership functions - Low, Average and High (Fig. 2). A low rank suggests that the corresponding case has proved to be a good match for past target cases. An average value for rank suggests that the results from using the case have acceptable values since they were reasonably close to the actual effort of the target case. Lastly, a high rank suggests that the case was a poor match for previous target cases and suggests caution during future retrievals.

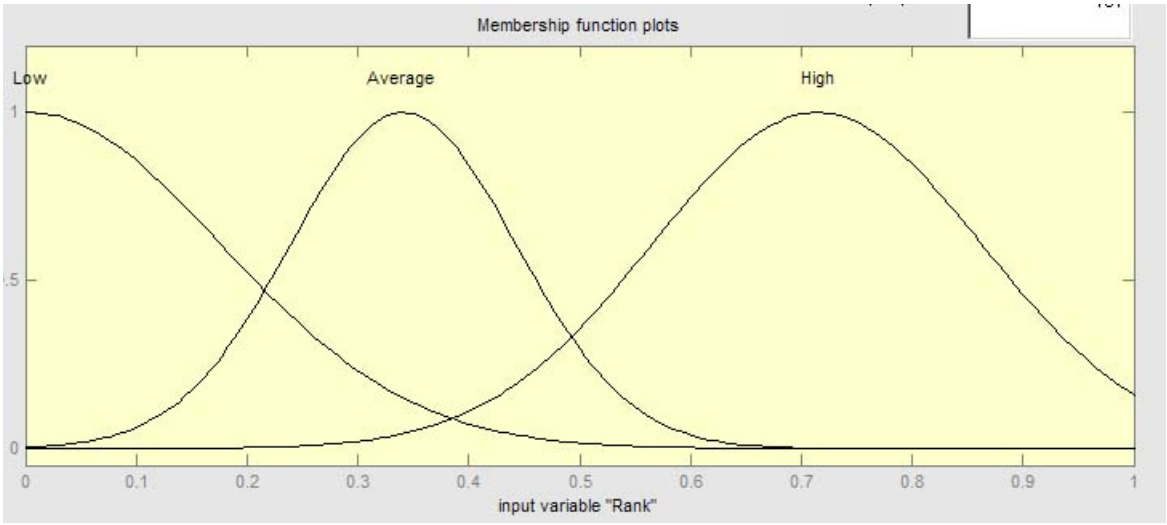

Fig. 2. Rank Membership Function

BMMRE Membership Functions Similar to the membership functions of the Rank input, BMMRE has 3 membership functions too - Low, Average and Rank (Fig. 3). A low value of BMMRE denotes a relatively small difference between the actual and predicted value of effort. An average value represents an acceptable range of error while a high value is not favoured.

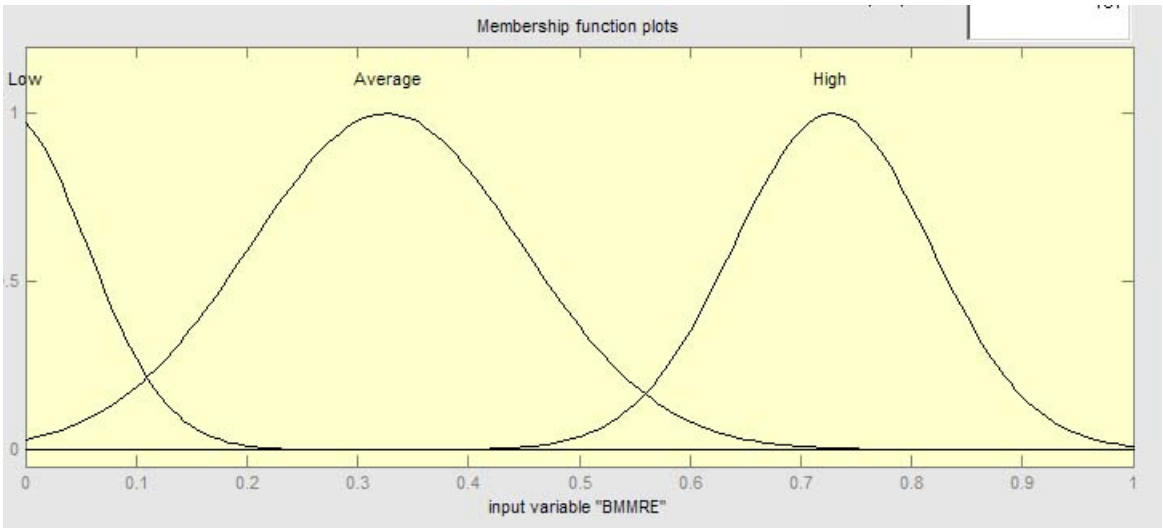

Fig. 3. BMMRE Membership Function

Frequency Membership Functions The third vital input of the fuzzy model is Frequency (Fig. 4). The notion behind including frequency is to be able to confirm if a case is 'misleading' or not since it may not be appropriate to dismiss a case that has produced a poor estimate, but has been used only once. Hence, if a case has been used often and has been repeatedly producing poor estimates, declaring it 'misleading' will be an appropriate action.

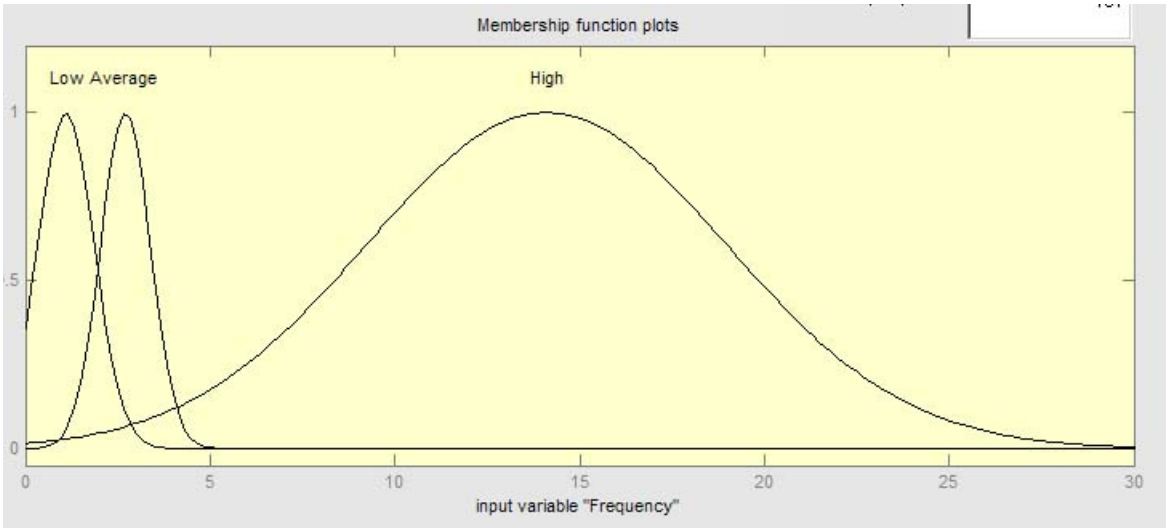

Fig. 4. Frequency Membership Function

Fuzzy Rules A series of fuzzy rules were developed to assess the ranks and frequency of cases in order to judge their suitability for reuse. Cases with very high MOR and BMMRE were immediately discarded irrespective of their frequency of use. On the other hand, cases with low MOR and BMMRE values were preserved and reused irrespective of their frequencies. Other fuzzy rules were relevant to those cases with MOR and BMMRE values lying between the two extremes. These rules maintained a delicate balance to show tolerance for cases leading to poor predictions but seldom used and discarded those which seemed to consistently lead to poor results over frequent use. An example of a fuzzy rule is:

> *If (Rank is Average) and (BMMRE is Low) and (Frequency is Average) then Decision is Use.*

4 Methodology

The methodology used to conduct this pilot study was deliberately simplified since computations were performed manually. Also, at this stage, a simple methodology would help better understand our results derived from each step. This section summarises the data set used for this pilot study, the generated seed cases inserted into the data set and the methodology used to conduct the pilot study.

4.1 Data Set

We used Mendes *et al.*'s data set [12] which comprised of student web development projects recorded as cases. This data set was chosen because of its simplicity and the fact that it contained a manageable number of cases, since most of the computation involved was to be done manually. Since all recorded projects involved web development, the features covered various measures of web architecture, e.g. page count, media count, connectivity, etc. Importantly, the target feature was Total Effort. The data set amounted to a total of 8 features and 34 cases.

4.2 Seed Cases

The purpose of the pilot study was to investigate the ability of the proposed model to identify misleading cases and use them with caution in the future. In order to confirm this, 11 seed cases were constructed and randomly injected into the data set. While the features in the seed cases had values which were in accordance with the composition of the original cases, the target value which is effort had peculiarly high or low values.

Thus, seed cases can be considered as cases with unusual behaviour that has not been captured by the recorded features. As a result, the seed cases have very high or low productivity levels and given the description of the project, they have unusual completion periods.

These seed cases were generated using the following mechanism. The feature values of the cases were allotted such that they are as likely to be retrieved as any of the original cases in the case base. The following was the procedure adhered to for their generation:

1. Choose a random original case from the case base.
2. Assign the first feature, of the seed case, a value equal to the average of the value of the random case's corresponding feature and that of its two nearest values of the same feature from other cases.
3. Similarly, assign remaining features with values with the same random case (from step 1) as a reference.
4. Assign a target value that is 250% higher or lower than that of the random case. This results in inconsistency in the case base.
5. Insert the generated seed case at a random location in the case base.

After inserting our seed cases into the original data set, the case base comprised of a total of 45 cases which included 34 original cases and 11 seed cases. Mendes's data set exclusively consisted of numerical variables. The above procedure would be inapplicable to data sets with categorical variables in them. Methods to generate such seed cases will need to be developed during the course of research to test the model on a variety of data sets.

4.3 Method

The procedure was followed in the order below:

1. To begin with, the case base was randomised to remove any effect that the original order of cases may have.
2. Five random non-seed cases were selected to form the initial case base.
3. Successive cases were used as target cases for which estimates were recorded (using the 2 nearest neighbours i.e. $k = 2$) and the corresponding MORs, BMMREs and frequencies of the source cases were updated.
4. If the case encountered was a seed case, it was simply added to the case base.
5. Meta-data (i.e. MOR, BMMRE and frequency) of every case that was retrieved was made to pass through the fuzzy model for approval. If accepted (output = true), the case was used. Otherwise the next nearest match approved was used for prediction in place of the original.
6. If the retrieved case had no meta-data, i.e. its frequency was 0, then the case was used unconditionally.
7. After prediction, new values of MOR and BMMRE were calculated and meta data for the corresponding cases were updated.
8. The target case (with the correct value of the target feature) was then added to the case base.

5 Results

The results obtained using the model have been very encouraging as the desired objective was achieved. The model was successful at identifying 8 out of the 11 seed cases injected into the case base and prevented their retrieval in the future. This section summarises the results achieved from this pilot study and compares it to the results from another independent run (including seed cases

and using two nearest neighbours for estimation) without the use of our proposed model. The basic method was performed using the same 5 initial cases in the case base and none of the seed cases were used as target cases, but simply added to the case base.

Without the model, the system made no distinction between good and poor cases and always used those with the lowest Euclidean distance for prediction. The resulting mean of the absolute residuals was calculated to be 68.17 hours. However, the experiment using the fuzzy model was able to identify the seed cases (as hypothesised) and avoided their use in 8 different predictions bringing down the residual by 45.2% to 37.33 hours. Fig. 5 compares the absolute residuals from the two independent runs. The boxplots reflect the effectiveness of our model as we can see the residuals from using meta-data fall within a lower and smaller range in comparison to those from the basic method.

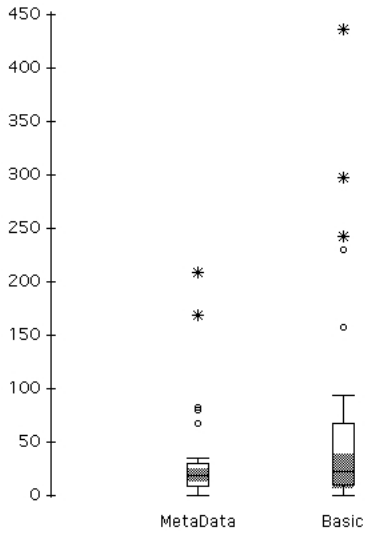

Fig. 5. Boxplot comparing absolute residuals of runs with and without using meta-data.

During the series of predictions, no seed case was reused after being identified as a misleading case, i.e. all 8 of the seed cases retrieved during problem solving were successfully identified. A single use of each seed case was required to enable the model to distinguish between good and misleading cases. Out of the 11 seed cases injected, 8 were used once and quarantined, while the remaining seed cases was never retrieved.

It is important to note that the data used for this pilot study has been artificially generated. Though the results suggest the model to be valuable and effective, it is important to validate it on real data sets to verify its worth.

6 Conclusions and Future Work

We have shown in this simple pilot study that the automatic generation of meta-data can improve prediction accuracy in the problem domain of software project costs. Fuzzy rules were developed that enabled misleading cases to be identified and consequently excluded from the problem solving process. This was demonstrated on a small data set of 34 cases to which 11 "rogue" or misleading cases were introduced. Our results show that given extra knowledge, CBR systems are capable of exploiting it to assess situations and make intelligent decisions accordingly to give better results.

These results are sufficiently encouraging to warrant further research and in particular we plan to address the following issues in the near future:

– In this pilot study, we assumed that the retrieved case is a 'misleading' case and penalise it when it proposes a poor solution. But it is equally plausible that the target case may be a

'misleading' case, which means that the target case may have unusual productivity levels. This calls for a technique to be embedded that will enable the model to make this judgement.
- We plan to extend the currently used fuzzy rules (by including the *4th* input, i.e. *distance*) to make possible exceptional decisions so that a misleading case might be reused if it is substantially more similar to the target than any alternative.
- Further, the fuzzy rules used in this model were hand developed. It will be vital to incorporate a mechanism to automate their generation in the model to make it more robust and able to generalise.
- Importantly, this model needs to be tested upon naturally occurring data sets that are known to contain inconsistencies or 'misleading' cases. Replication of our work on such data sets is vital to confirm its worth.

Acknowledgments

The authors are indebted to Emilia Mendes of the University of Auckland, NZ for making her data set available.

References

1. Delany, S.J.: *The Design of a Case Representation for Early Software Development Cost Estimation.* MSc, Stafford University (1998)
2. Finnie, G.R., G.E. Wittig.: A Comparison of Software Effort Estimation Techniques: Using FPs with Neural Networks, Case Based Reasoning and Regression Models. *Journal of Systems and Software*, Vol. 39,(1997) 281-289
3. Heery, R.: Review of metadata formats, *Program*, Vol. 30, no. 4, (1996) 345-373
4. Iglezakis, I.: The Conflict Graph for Maintaining Case-Based Reasoning Systems. *4th International Conference on Case-Based Reasoning (ICCBR)*, Springer-Verlag (2001) 263-275
5. Jantzen, J.: Tutorial on fuzzy logic. *Technical Report 98-E 868*, Technical University of Denmark (1998)
6. Jørgensen, M., U. Indahl, D. Sjøberg.: Software effort estimation by analogy and regression toward the mean. *Journal of Systems and Software*, Vol. 68, no. 3 (2003) 253-262
7. Kadoda, G., M. Cartwright, L. Chen, M. Shepperd.: Experiences Using Case-Based Reasoning to Predict Software Project Effort. *In Proceedings of the International Conference on Empirical Assessment in Software Engineering*, Keele, UK (2000)
8. Kadoda, G., M. Cartwright, M. Shepperd.: Issues on the Effective Use of CBR Technology for Software Project Prediction, *In Proceedings of the International Conference on Case based Reasoning*, Vancouver (2001)
9. Kirsopp, C., M. J. Shepperd.: Case and Feature Subset Selection in Case-Based Software Project Effort Prediction, *In Proceedings of the 22nd SGAI International Conference on Knowledge-Based Systems and Applied Artificial Intelligence*, Cambridge, UK (2002)
10. Leake, D.B., D.C. Wilson.: Categorizing Case-Base Maintenance: Dimensions and Directions. *In Advances in Case-Based Reasoning: Proceedings of the Fourth European Workshop on Case-Based Reasoning*, Berlin, Germany, (1998) 196-207
11. Leung, H., Fan, Z.: *In Handbook of Software Engineering and Knowledge Engineering*, Vol. 2 (Ed, Chang, S. K.), pp808, World Scientific, (2002)
12. Mendes, E., S. Counsell, N. Mosley.: Investigating the Use of Case-Based Reasoning Adaptation Rules for Web Project Cost Estimation. *The Twelfth International World Wide Web Conference*, ACM Press, Budapest, Hungary (2003)
13. Miyazaki, Y., M. Terakado, K. Ozaki, H. Nozaki.: Robust Regression for Developing Software Estimation Models. *Journal of Systems and Software*, Vol. 27, no. 1 (1994) 3-16
14. Prietula, M.J., S. Vicinanza, T. Mukhopadhyay.: Software-effort estimation with a case-based reasoner. *Journal of Expt. Theor. Artificial. Intelligence.* Vol. 8 (1996) 341-363
15. Reinartz, T., I. Iglezakis, T. Roth-Berghofer.: Review and restore for case base maintenance. *Computational Intelligence*, Vol. 17, no. 2 (2001)
16. Skalak, D.B.: Prototype and feature selection by sampling and random mutation hill climbing algorithms. *In Proceedings of the 11th International Machine Learning Conference*, Morgan Kauffmann (1994) 293-301.

17. Shepperd, M., C. Schofield.: Estimating software project effort using analogies. *IEEE Transactions on Software Engineering*, Vol. 23 (1997) 736-743
18. Smith, R., J.E. Hale, A.S. Parrish.: An Empirical Study Using Task Assignment Patterns to Improve the Accuracy of Software Effort Estimation. *IEEE Transactions on Software Engineering*, Vol. 27, no. 3 (2001) 264-271
19. Spalazzi, L.: A Survey on Case-Based Planning. *Artificial Intelligence Review* Vol. 16, no. 1 (2001) 3-36.