Theoretical Maximum Prediction Accuracy for Analogy-based Software Cost Estimation

Jacky W. Keung
NICTA Ltd. Sydney Australia
CSE, UNSW, Sydney Australia
Jacky.Keung@nicta.com.au

Abstract

Software cost estimation is an important area of research in software engineering. Various cost estimation model evaluation criteria (such as MMRE, MdMRE etc.) have been developed for comparing prediction accuracy among cost estimation models. All of these metrics capture the residual difference between the predicted value and the actual value in the dataset, but ignore the importance of the dataset quality. What is more, they implicitly assume the prediction model to be able to predict with up to 100% accuracy at its maximum for a given dataset. Given that these prediction models only provide an estimate based on observed historical data, absolute accuracy cannot be possibly achieved.

It is therefore important to realize the theoretical maximum prediction accuracy (TMPA) for the given model with a given dataset. In this paper, we first discuss the practical importance of this notion, and propose a novel method for the determination of TMPA in the application of analogy-based software cost estimation. Specifically, we determine the TMPA of analogy using a unique dynamic K-NN approach to simulate and optimize the prediction system. The results of an empirical experiment show that our method is practical and important for researchers seeking to develop improved prediction models, because it offers an alternative for practical comparison between different prediction models.

Keywords: Software Metrics and Measurement, Software Cost Estimation, Analogy, *K*-NN, MMRE

1. Introduction

In software engineering, software effort estimation is the prediction of the work effort required to complete a software development project. Software effort is usually measured in person-month, and it is generally the largest and least predictable component of software project development costs.

Estimates are probabilistic in nature. They are usually represented as a most likely figure within a range of possible values. If estimates are derived from regression models for example, the upper and lower bounds as an estimate can be calculated based on confidence intervals.

An important research area for predictive software cost estimation models is how to determine model prediction performance accuracy. This is usually achieved by measuring the difference between the predicted value and the actual value in the training dataset, and is commonly carried out by using the *Jackknife* model validation technique.

One of the most common misconceptions about these prediction models is that they are able to achieve up to 100% prediction accuracy. Based on current prediction models and technologies, estimates derived from these models are unlikely to achieve this zero-error objective due to a number of factors other than the robustness of the prediction model itself.

Unfortunately, evaluation criteria such as *MMRE*, *MdMRE* and *Pred*(25) etc. only measure the absolute difference between the predicted value and the actual value based on point-estimate between each predicted value and its actual value in the training set. Given that the dataset quality is one of the important factors influencing the prediction outcome, these accuracy measures overestimate the error given by the predictive cost estimation models, where factors such as dataset quality are not being isolated from the measurement of prediction accuracy. In fact, the attempt to achieve absolute prediction precision is empirical impossible.

In this paper, we propose the notion of Theoretical Maximum Prediction Accuracy (*TMPA*) as a metric and an alternative evaluation criterion for prediction models, where *TMPA* considers the quality of the dataset in the model evaluation. We especially apply *TMPA* to analogy-based software cost estimation, where a technique called dynamic *K*-NN to determine *TMPA* is introduced.

To emphasize the importance of TMPA, we present results from an experiment using the Desharnais dataset. Results show that our method solves some of the fundamental issues of the evaluation criteria of predictive models. Our method can also be used as an alternative evaluation criterion to aid the development of improved prediction models for software cost estimation.

There are two categories of work that may be considered relevant to this study: studies on evaluation metrics, and *K*-NN or Analogy-based studies. Section 2 presents an overview of related work on evaluation metrics used in various software cost estimation studies. Section 3 briefly describes the principle of analogy-based software cost estimation. Section 4 introduces *TMPA*, and in section 5 presents our dynamic *K*-NN approach to determine *TMPA* in the context of analogy. Section 6 provides the datasets and the analysis procedure where results are present in section 7. Section 8 discusses our findings and section 9 concludes the paper.

2. Background and Related Work

MMRE (Mean Magnitude of Relative Error) is the most widely used evaluation criterion for assessing the performance of competing software effort estimation models. It calculates the mean of all *MRE* (Magnitude of Relative Error) values derived from each prediction. Similarly *MdMRE* (Median Magnitude of Relative Error) calculates the median of all *MRE* values from each prediction. *MRE* is the metric component in *MMRE* and *MdMRE* and is defined as follows:

$$MRE = \frac{\left|y - \hat{y}\right|}{y} \qquad (1)$$

where y = actual, $\hat{y}$ = prediction. *MMRE* measures the overall averaged error from each individual case within the training set. The smaller the *MMRE* indicates the better the prediction performance of the model. However it is unrealistic to assume that the *MMRE* would ever be close to zero. Conte et al. [1] consider MMRE $\leq$ 0.25 (within 25%) as acceptable for effort prediction models. It is a common misconception that this barrier threshold value can be applied in every single case as it does not consider other important influential factors which may also contribute to the prediction error.

For example, a prediction model *X* may achieve a better *MMRE* of 0.20 using dataset *A*, but a worse *MMRE* of 0.80 for dataset *B*. The implication in here is that prediction model *X* performed less favorably on dataset *B*, but its prediction ability does not change. What has been really changed is the quality of the dataset. In the above example, the change of dataset has also changed the maximum possible achievable prediction for that dataset, i.e. *MMRE* does not reflect this information.

In the context of software cost estimation research (as opposed to practice), the most commonly applied software effort prediction methods are regression [2] and data-intensive analogy [3] [4]. Previous empirical software cost estimation studies [5] have attempted to determine "which method is the best" using various evaluation criteria such as *MMRE*.

However these studies have produced conflicting results. For example, Shepperd & Schofield [3] claimed analogy provided better prediction accuracy in terms of *MMRE* and *Pred(25)*. This was supported by Angelis & Stamelos [6] who found analogy-based systems were far superior to other methods and by the more recent work of Mendes, Mosley & Counsell [7] and Mendes & Kitchenham [8] on a large heterogeneous data set. In contrast, Myrtvelt & Stensrud [5] replicated previous studies described by Shepperd & Schofield [3], but found analogy was not better than regression, and they also suggested that the results are sensitive to experimental design. Similarly, Briand et al. [9] and Morasca et al. [10] found analogy-based systems were less robust than other methods, particularly when dealing with heterogeneous data sets. Later, Jeffery, Ruhe & Wieczorek [11] also concluded stepwise regression outperformed analogy-based systems with the ISBSG dataset.

Over the past 15 years, the research team led by Martin Shepperd tried to explain why different research teams have reported widely different results by using analogy technology. Most recently, Mair & Shepperd [12] undertook a systematic review to investigate these contradictory results. They review 20 primary studies comparing regression and analogy conducted during the past decade, and concluded that there was no clear indication that regression was better than analogy or vice versa. They concluded that the mixed results are due to the characteristic of the dataset and the individual data points. Shepperd & Kadoda [13] have further studied this issue using simulation and concluded similar findings. The implication is that the resultant prediction is sensitive to the data quality of individual datasets.

Keung and Kitchenham have attempted to identify ways to improve estimation and dataset quality through the use of outlier removal techniques and dataset preprocessing using the Analogy-X method [14][15]. They also used different weights for different project features to achieve better prediction accuracy [16].

All of these aforementioned studies have used *MMRE* and related measures to compare different model performance. A high estimation error is commonly interpreted as a poor estimation model being used. This statement is not entirely a correct interpretation. High estimation errors can be due to other factors such as increased estimation complexity due to the complexity of the system being developed, insufficient cost control of the project and more importantly the dataset quality or the dataset relevance to the prediction problem. Foss et al. in their extensive study [17] on various prediction accuracy metrics also concluded that current measures of performance (such as *MMRE*) are unreliable and may provide misleading information. It is important to provide a realistic and accurate measure of these competing prediction models, which will not only consider the effect of the prediction model itself, but also other factors which may influence the overall prediction.

3. The Principle of Analogy (*K*-NN)

This study focuses only on the analogy-based approach for software cost estimation. It is therefore important to introduce the principle of analogy, and this section provides a brief overview.

Data-intensive analogy for software effort estimation has been proposed as a viable alternative to other prediction methods such as linear regression. In many cases, researchers found analogy outperformed algorithmic methods [3]. Keung et al. [14] defined analogy is based on the following premise:

"*Projects that are similar with respect to a set of project features will also be similar with respect to effort.*"

Based on this principle, analogy utilized the *K*-nearest neighbour algorithm (*K*-NN) to search for projects that are similar with respect to the target project in terms of project features.

The application of analogy to software effort estimation follows a number of steps:

1. When a new estimation problem arises its project characteristics such as size and cost factors are first codified in terms of the feature vector forming the problem description, and becoming the basis for retrieving similar project cases from the project case repository.
2. The effectiveness of similarity measures greatly depends on the overlap of features found in the case repository, because past projects with similar project characteristics (usually measured in Euclidean distance) are likely to have similar project outcomes. The similarity measures between the target project and each of the past projects are used to rank the closeness of the past projects to the target project.
3. A numeric value K is defined, and only the closest K projects are then selected to adapt a prediction solution for the target project. This is a typical example of K-NN for classifying objects based on closest training examples in the feature space.
4. The prediction performance of the model is then evaluated against one of the model evaluation criteria such as *MRE* (Equation 1). The actual effort (y value) for each project case in the training dataset is usually known, based on actual data recorded in the past. The prediction (value $\hat{y}$) of analogy is derived by adapting a solution from K nearest projects that are similar with respect to a set of project features, and we assume these K analogous projects will also be similar with respect to actual effort (value y).
5. Jackknifing is a commonly used model validation approach in software effort estimation. It is also known as the Leave-One-Out approach. A complete Jackknife testing involves training the dataset n times (for a dataset containing n cases), for each time the dataset is trained, one case (i^{th} case) is withheld from the training set and used exclusively to evaluate the performance of the dataset using the prediction model. The withheld case is returned to the dataset, and the $(i + 1)^{th}$ case is then withheld in the following iterations until the last iteration ($i = n$). In this case, a *MRE* value is calculated on each i^{th} case, and subsequently the mean *MRE* values for all n-cases become *MMRE*.

The overall performance of analogy depends on a number of factors. These include the dataset quality, the relevance of the project cases to the target project, the feature subset selection technique, and the distance measures used to assess similarity between the source analogues and the target project [2].

Close examination of the estimation process of analogy revealed that the value of K is critically important as it determines the number of nearest project cases to be used for the solution adaptation. In analogy, a fixed value of K is commonly used to find K similar projects from n projects. The fixed value K is usually determined by experts, or can be optimized using simulation [13].

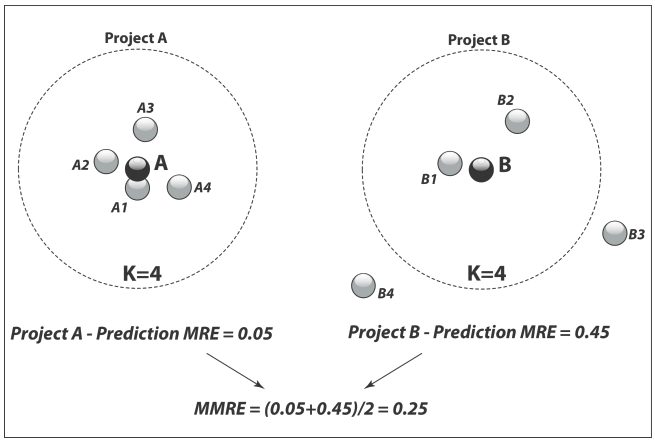

Figure 1 The *K*-NN algorithm: An Example

Figure 1 illustrates an example of the K-NN algorithm using a fixed *K*=4 with 2 project cases in the training dataset. The figure shows that Project *A* has 4 closely related project cases (*A1,A2,A*3 and *A4*) resulting in a *MRE* value of 0.05 (95% accurate compared to the actual effort of Project *A*). In contrast, Project *B* has only two similar project cases (*B1* and *B2*) within the same dimensional space. Nevertheless, as a fixed value *K*=4 has been defined to train the entire dataset, project *B* must also consider projects *B3* and *B4*. Projects *B3* and *B4* are far distant from target project *B*, resulting in a less favourable *MRE* value of 0.45. The combined effect or the overall model prediction accuracy based on *MMRE* is 0.25. It shows that the model performance is heavily influenced by the fixed *K*=4 used in the prediction of project *B*.

The assumption in here is that if a suitable *K* value for each individual project can be identified, the *MRE* value can be substantially improved. This is largely due to the dataset quality and the formation of the data points, as these data points are based on actual events, therefore they are not evenly distributed across the *n*-dimensional feature space.

MRE (see Equation 1) is a measure of "absolute difference" between the prediction and actual. As we can see in the example above the influence of the dataset quality is not included in the metric. Figure 2 illustrates a simple example of measuring software effort prediction accuracy based on *MRE* for 12 software projects using analogy. What is missing in Figure 2 is a realistic measure of the dataset quality margin. This margin represents the maximum possible prediction for case *i*, where the cause of prediction error can be effectively isolated between dataset quality and the prediction model. The question in here is how to determine the value of K_i for each case *i* so that the prediction accuracy can be optimized for analogy using the *K*-NN algorithm.

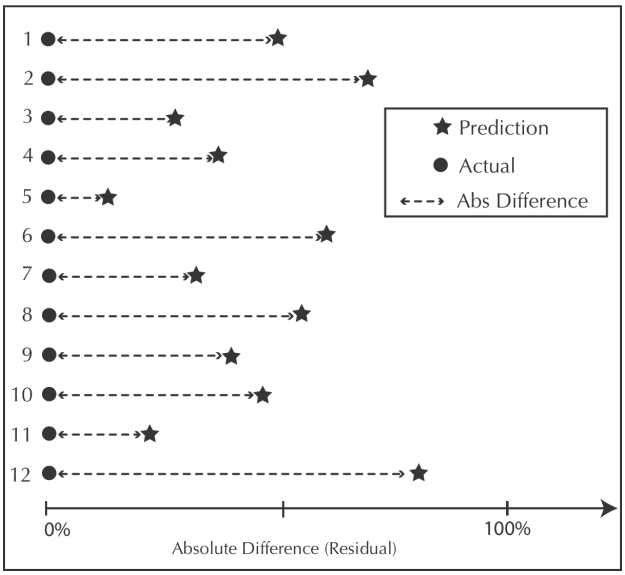

Figure 2 MRE Values of 12 Projects

4. Theoretical Maximum Prediction Accuracy (*TMPA*)

We define the upper limit of the prediction accuracy that can be achieved in the analogy-based framework as its *TMPA*. *TMPA* is important to researchers as well as software engineers. It places a strict barrier that cannot be overcome by any means while remaining empirically compliant for the modelling approach used.

The assumption in here is that, if an optimal K_i value for each project case *i* in the dataset is known, we are able to optimize the prediction accuracy based on *MRE* for the prediction of case *i*. The overall *MMRE* value is also theoretically maximized, which is the *TMPA* of *MMRE*.

Figure 3 illustrates the relationships between values of *Actual*, *TMPA* and *Prediction*.

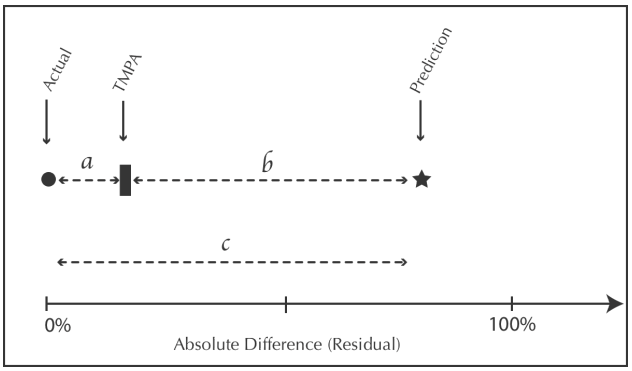

Figure 3 Relationships between values of *Actual*, *TMPA* and *Prediction*.

In Figure 3, *c* is the absolute difference between prediction and actual or overall prediction error, *a* is the prediction error due to the dataset characteristic measured by the *TMPA*, and *b* becomes an effective measure of the error due to the prediction model used.

The theoretical maximized MRE_{MAX} is defined as:

$$MRE_{MAX} = \frac{\left| y_{max} - \hat{y} \right|}{y_{max}} \quad (2)$$

where $y_{max} = TMPA$, $\hat{y}$ = prediction.

The overall theoretical maximized $MMRE_{MAX}$ is defined as :

$$MMRE_{MAX} = \frac{MRE_{MAX}}{n} \quad (3)$$

where *n* is the number of cases in the training set.

5. Method to determine TMPA using Dynamic *K*-NN

In Section 3, we have discussed the model prediction performance implication of the value *K* in the selection of nearest analogous projects. Let's assume that the value of *K* is not fixed but it is dynamically selected in each case of the *Jackknife* simulation as below:

The following steps are used to determine the value *Ki* for each case x_i dynamically.

Given *n* project cases $\{x_1, x_2, x_3, ..., x_n\}$ in the training set, and for each project case x_i in turn:

1. The Euclidean distances between x_i and all other project cases x_{-i} (all but excluding x_i) are computed and ranked by their similarity.
2. The predictions are then simulated with K_i =1, K_i =2,…,K_i =n for case x_i. We use *MRE* in this case.

K	Prediction	Actual	MRE
1	840.000	805.000	0.043
4	1233.750	805.000	0.533
3	1463.000	805.000	0.817
5	1471.400	805.000	0.828
2	1561.000	805.000	0.939
7	1609.000	805.000	0.999
6	1614.667	805.000	1.006
…	…	…	…
n	4886.921	805.000	5.071

Table 1 Simulated *K* values, an example.

In the example shown in Table 1, the simulation result for project case x_i shows that the prediction is at its optimal with 1 nearest project (K_i=1 with an MRE_i =0.043).

3. This completes the simulation for case x_i. Repeat step 1 to 3 for project case x_{i+1}.

The procedure introduced above is called *Dynamic K-NN*, where an optimal K_i value for each project case *i* is simulated and used in the prediction. The aggregated average of all MRE_i above is the overall *TMPA*, which is denoted as $MMRE_{MAX}$. As a measure of prediction robustness of the model and its utilization of the dataset, based on $MMRE_{MAX}$ we define prediction efficiency ε of the model as:

$$\epsilon = MMRE - MMRE_{MAX} \quad (4)$$

For example, if *MMRE* for the prediction model is 0.50 and its related $MMRE_{MAX}$ is 0.20, the prediction model is able to provide 30% prediction accuracy given that the $MMRE_{MAX}$ is 20% towards absolute zero error. This 20% margin is uncontrollable in an experimental environment, primarily due to the quality of the dataset.

6. Dataset and Analysis Procedure

The Desharnais dataset is used in this study. It comprises 77 completed software project data from a Canadian Software house. It was first reported in Desharnais [18] and was used in Shepperd and Schofield [3] to compare regression models and analogy, and in [19] to analyse the impact of project feature weights using an extensive search algorithm. The Desharnais dataset is one of the most well-known and complete datasets publicly available in software effort estimation research. The original version of the dataset had 81 projects, but four of them had missing values and were excluded from our analysis [18]. This dataset has 8 independent variables, which are shown in Table 2. The response variable is *ActualEffort.*

Proj. Feature	Description	Type
Adj.FPs	Adj. Function Points	Continuous
RawFPs	Raw Function Points	Continuous
Transactions	No. of Transactions	Continuous
Entities	No. of Entities	Continuous
Adj.Factor	Technology Adj. Factor	Continuous
ExpProjMan	Exp. of Proj.Mgmt.	Continuous
ExpEquip	Exp. of Equipment	Continuous
Dev.Env	Dev. Environment	Categorical

Table 2 Desharnais Dataset Project Features

For the purpose of project case retrieval, it is important to first identify which features are most influential to the prediction outcome or *ActualEffort*. We identified that the continuous variable *Adj.FPs* is the most influential project feature for the 77 completed software projects based on [3]. It is possible to further improve the prediction accuracy by homogenising the dataset using the categorical variable *Dev.Env*, on the basis of differentiating development environments. In practice, it is unlikely that a company would have access to such large volumes of data. In addition smaller, more homogenous datasets are more useful for effort estimation. This dataset grouping approach is similar to the study of Shepperd and Schofield [3], where the Desharnais dataset has been divided into Desharnais-1(44 cases), Desharnais-2(23 cases) and Desharnais-3(10 cases).

In the following experiments, we will demonstrate the application of *TMPA* on the Desharnais dataset, as well as on its 3 homogenized subsets. To demonstrate what happens when *TMPA* is applied to datasets that show no predictive relationships or whose quality of the dataset is suboptimal, we show the effect on two randomized datasets.

The two randomized datasets were generated to have similar properties to that of the Desharnais dataset in terms of the number of cases and the statistical distribution of variables, but were constrained to have no relationship between the dependent and the independent variables. We generated 10,000 random datasets stratified strictly accordingly to Desharnais dataset's mean and variance and selected two, one has the largest correlation (DeshRand-0) and the other one has the smallest correlation (DeshRand-1).

7. Applying *TMPA* on Desharnais Dataset

In this section we demonstrate our method on the Desharnais dataset and related datasets described in the previous section. Our *Dynamic K-NN* approach will be applied to obtain optimized *K* values for the training set and to derive *TMPA*. To compare our result with the conventional approach using *MMRE*, we simulate different fixed values of *K* on the original Desharnais dataset and the results are shown in Figure 4.

7.1 Fixed *K*-NN

Figure 4 employs box-plots to illustrate the prediction error distribution given by a sequence of fixed *Ks* for nearest neighbour (*NN*) searches, where a sequence of *K* values between 1 and 10 have been selected. The distances of the outlying data points become apparent and increasingly large as the size of *K* increases. Nevertheless, observation shows the optimal *K* value exists between 3 and 4 (see highlighted zone in Figure 4). The *MMRE* values are 0.713 for *K*=3 and 0.666 for *K*=4 (see Table 3). The implication in here is that the optimal *K* value exists in this highlighted zone where the prediction can be highly optimized.

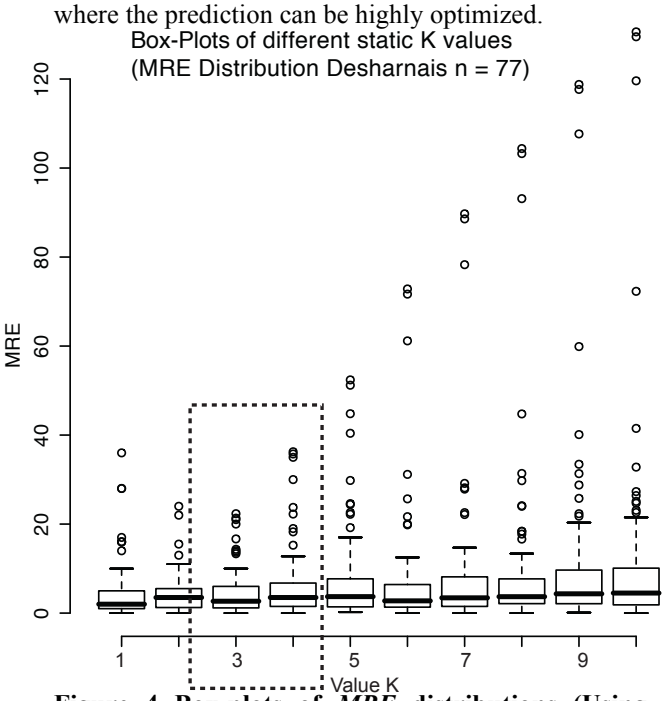

Figure 4 Box-plots of *MRE* distributions (Using Different Fixed *K* values)

7.2 Dynamic *K*-NN

The Dynamic K-NN procedures described in Section 5 has been applied to search for optimal *K* values for each individual case. The measurement error frequency distribution of using a sequence of *K* values for Desharnais dataset is illustrated in Figure 5.

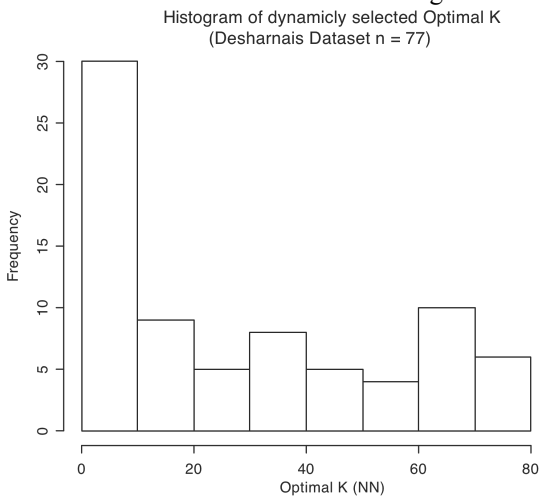

Figure 5 Frequency distribution of dynamically selected optimal *K* values (Desharnais Dataset)

	Cases	$MMRE_{MAX}$	$MMRE_{(K=3)}$	$\varepsilon_{(K=3)}$	$MMRE_{(K=4)}$	$\varepsilon_{(K=4)}$
Desharnais	77	*0.260*	0.713	0.453	0.666	0.406
DeshRand-0	77	0.556	1.474	0.918	1.440	0.885
DeshRand-1	77	0.316	1.050	0.733	0.953	0.637
Homogenized Subsets:						
Desh-Dev1	44	0.127	0.361	0.234	0.391	0.264
Desh-Dev2	23	0.120	0.388	0.268	0.392	0.272
Desh-Dev3	10	0.139	0.252	0.113	0.324	0.186

Table 3 Summary of results using $MMRE_{MAX}$ and ε (Desharnais, DeshRand-0 and DeshRand-1 datasets)

Based on the Desharnais dataset, and given that the K_i values have been optimized using the dynamic *K*-NN approach, we obtained a $MMRE_{MAX}$ of 0.260 based on *TMPA*, which is significantly improved compared to that of using a fixed *K* value (0.713 and 0.666) (see Table 3). The model prediction efficiency ε (see Equation 4) is 0.453 for K=3 and 0.406 for K=4. This also implies that there is room for the development of an improved analogy-based prediction approach.

Based on the two random datasets showing no predictable patterns, our *TMPA* method is able to differentiate the error component from the prediction accuracy measure. For example, the prediction model efficiency ε (based on K=3) for DeshRand-0 is 0.918 and its $MMRE_{MAX}$ value due to dataset errors is 0.556, this implies that the dataset DeshRand-0 is not suitable for the software effort estimation using analogy. DeshRand-1 dataset is subject to similar conclusion.

Using a separate analogy-based approach we homogenize the dataset into three subsets. As shown in Table 3, we observed significant improvement in terms of estimation accuracy. The prediction errors for these three subsets are also smaller, as is in their $MMRE_{MAX}$ values. Their model prediction efficiency ε is exceptionally good compared to the original dataset with 77 cases. Especially for Desh-Dev3, where the model prediction efficiency ε is 0.113 based on K=3, this implies that the prediction model has nearly reached its theoretical maximum (*TMPA*).

8. Discussion

We have successfully employed a novel approach to evaluate the true efficiency of model prediction accuracy based on *TMPA* described in the study.

Our results show that applying *TMPA* is an effective strategy to account for the influence of each project's impact to the overall estimate, precisely the dataset quality and its relevance to the target problem under investigation. This has been clearly demonstrated using the Desharnais dataset.

In theory the maximum achievable prediction accuracy value can be achieved if and only if the prediction model itself is sophisticated and accurate enough. Even if there is an identical project in terms of features and characteristics in the dataset, it is possible that the system is unable to find it because of the influence of other project cases. The prediction power issue is independent to the quality of the dataset in this case as our approach effectively isolates the error terms introduced by the quality of the dataset.

The notion of *TMPA* can be easily adapted to other metrics. We are anticipating further experiments to apply *TMPA* to show its effectiveness with other software cost estimation approaches and models, including regressions models.

9. Conclusion

In this paper we introduced both the notion and the application of the theoretical maximum prediction accuracy (*TMPA*) for software cost estimation using analogy. The introduced approach is a robust software metric in addition to existing model performance criteria such as *MMRE*. *TMPA* utilizes the *Dynamic K-NN* approach to simulate and obtain a maximized possible prediction in each dataset training phase using *Jackknife* validation. *TMPA* captures the influence of the dataset quality and its relevance to the target problem, and effectively isolates the prediction loss due to the dataset quality.

An error-free prediction in software cost estimation remains elusive dream for many, but the truth is, it is both empirically and theoretically impossible. The *TMPA* metric is also not empirically achievable with existing cost estimation models, but it provides a theoretical achievable limit for competing models.

We evaluated our *TMPA* approach using the Desharnais and two random datasets. Our results show that applying *TMPA* is an effective strategy to account for the influence of each project's impact to the overall estimate. And it is proven to provide a measurable and realistic target objective for researchers seeking to develop improved cost estimation models using the *TMPA* measure rather than using the unrealistic absolute zero prediction target. Our method is thus a major improvement to the evaluation of software cost estimation models.

10. Acknowledgements

NICTA (National ICT Australia Ltd.) is funded through the Australian Government's Backing Australia's Ability Initiative, in part through the Australian Research Council.

11. References

[1] S. Conte, H. Dunsmore, and V. Y. Shen, *Software Engineering Metrics and Models*. Menlo Park, Calif: Benjamin/Cummings, 1986.

[2] F. Walkerden and R. Jeffery, "An Empirical Study of Analogy-based Software Effort Estimation," *Empirical Software Engineering,* vol. 4, pp. 135-158, June 1999.

[3] M. J. Shepperd and C. Schofield, "Estimating Software Project Effort Using Analogies," *IEEE Transactions on Software Engineering,* vol. 23, pp. 736-743, 1997.

[4] M. J. Shepperd, C. Schofield, and B. Kitchenham, "Effort estimation using analogy," in *18th Intl. Conf. on Software Engineering*, Berlin, 1996.

[5] I. Myrtvelt and E. Stensrud, "A controlled experiment to assess the benefits of estimating with analogy and regression models," *IEEE Transactions on Software Engineering,* vol. 25, pp. 510-525, 1999.

[6] L. Angelis and I. Stamelos, "Reply to comments by M. Jorgensen, on the paper: 'A Simulation Tool for Efficient Analogy Based Cost Estimation'," *Empirical Software Engineering,* vol. 7, pp. 377-381, 2002.

[7] E. Mendes, N. Mosley, and S. Counsell, "A replicated assessment of the use of adaptation rules to improve Web cost estimation," in *International Symposium on Empirical Software Engineering*, Rome, Italy, 2003, pp. 100-109.

[8] E. Mendes and B. Kitchenham, "Further comparison of cross-company and within-company effort estimation models for Web applications," in *International Symposium on Software Metrics*, Chicago IL, USA, 2004, pp. 348-357.

[9] L. C. Briand, K. E. Eman, and K. D. Maxwell, "An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques," in *International Conference on Software Engineering*, LA, CA, 1999.

[10] S. Morasca, L. C. Briand, V. R. Basili, E. J. Weyuker, M. V. Zelkowitz, B. Kitchenham, S. Lawrence Pfleeger, and N. Fenton, "Comments on "Towards a framework for software measurement validation"," *IEEE Transactions on Software Engineering,* vol. 23, pp. 187-189, 1997.

[11] R. Jeffery, M. Ruhe, and I. Wieczorek, "Using public domain metrics to estimate software development effort," in *International Symposium on Software Metrics*, London, England, 2001, pp. 16-27.

[12] C. Mair and M. Shepperd, "The consistency of empirical comparisons of regression and analogy-based software project cost prediction," in *4th International Symposium on Empirical Software Engineering*, Noosa Heads, Australia, 2005.

[13] M. Shepperd and G. Kadoda, "Comparing software prediction techniques using simulation," *IEEE Transactions on Software Engineering,* vol. 27, pp. 1014-1022, 2001.

[14] J. W. Keung, B. Kitchenham, and D. R. Jeffery, "Analogy-X: Providing Statistical Inferences to Analogy-based Software Cost Estimation," *IEEE Transactions on Software Engineering,* vol. PrePrint:Online (TSE.2008.34), 15 May 2008.

[15] J. W. Keung and B. A. Kitchenham, "Experiments with Analogy-X for Software Cost Estimation," in *Australian Software Engineering Conference* Perth, 2008.

[16] J. W. Keung and B. Kitchenham, "Optimizing Project Feature Weights for Analogy-based Software Cost Estimation using the Mantel Correlation," in *Asia Pacific Software Engineering Conference* Nagoya, Japan, 2007, pp. 222-229.

[17] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A Simulation Study of the Model Evaluation Criterion MMRE," *IEEE Transactions on Software Engineering,* vol. 29, pp. 985-995, 2003.

[18] J. M. Desharnais, "Analyse statistique de la productivitie des projects informatique a partie de la technique des point des function," in *University of Montreal*. vol. Masters thesis: Univ. of Montreal, 1989.

[19] M. Auer, A. Trendowicz, B. Graser, E. Haunschmid, and S. Biffl, "Optimal Project Feature Weights in Analogy-Based Cost Estimation: Improvement and Limitations," *IEEE Transactions on Software Engineering,* vol. 32, pp. 83-92, Feb 2006.