2013 20th Asia-Pacific Software Engineering Conference

Improving Analogy-based Software Cost Estimation through Probabilistic-based Similarity Measures

Passakorn Phannachitta*, Jacky Keung†, Akito Monden*, and Ken-ichi Matsumoto*
*Graduate School of Information Science, Nara Institute of Science and Technology, Japan
†Department of Computer Science, City University of Hong Kong, China
Email: {phannachitta-p, akito-m, matumoto}@is.naist.jp, Jacky.Keung@cityu.edu.hk,

***Abstract*—The performance of software cost estimation based on analogy reasoning depends upon the measures that specifying the similarity between software projects. This paper empirically investigates the use of probabilistic-based distance functions to improve the similarity measurement. The probabilistic-based distance functions are considerably more robust, because they collect the implicit correlation between the occurrences of project feature attributes. This information gain enables the constructed estimation model to be more concise and comprehensible. The study compares 6 probabilistic-based distance functions against the commonly-used Euclidian distance. We empirically evaluate the implemented cost estimation model using 5 real-world datasets collected from the PROMISE repository. The result shows a significant improvement in terms of error reduction, that implies an estimation based on probabilistic-based distance functions achieve higher accuracy on average, and the peak performance significantly outperforms the Euclidian distance based on Wilcoxon signed-rank test.**

***Keywords*-Software Cost Estimation, Analogy, Heterogeneous Distance Function, Machine Learning**

I. Introduction

Accurate software development cost estimation is vital for software development. Unreliable cost model can affect the software product quality or even worse as causing project cancellation. Cost estimation therefore is an important area in software engineering research. For decades, much effort has been given to improve the estimation accuracy. The cost estimation methods most possibly encountered in the literatures are analogy-based reasoning estimation (ABE) and regression-based model construction. They have shared a proportional interest, thus; up to now, numerous attempts were failed to define the best solution, as many research conclusions have produced conflicts [1], [2].

The essential hypothesis of software cost estimation based on ABE reasoning is that a new software project should acquire an approximate cost from similar past projects. ABE estimation accuracy heavily depends on 5 parameters, which are feature subset selection, number of project analogs to build the model, similarity measure, scaling, and case adaption [3]. These parameters require a massive computing power to optimize, which such sufficiently powerful computational architectures were not available in the past. To the best of our knowledge, ABE-CUDA is the first and the only ABE algorithm and framework harnessing the high performance computing (HPC) technology to feasibly optimize ABE in a completed search-space, thus; the estimation model constructed in ABE-CUDA framework can extend the performance of the ABE method [4].

Such available of the high performance architecture enables us to challenge an estimation performance improvement throught sophisticated similarity measures. We explore and adopt the use of probabilistic-based distance functions, namely Value Difference Metric (VDM) [5], as similarity measures in software ABE. The distance functions are commonly discussed in artificial intelligence (AI) area of research [5], where an application requires less computational power than software ABE. Therefore, this study was otherwise impossible in the past. The probabilistic-based distance function is theoretically superior to the Euclidian-based distance that commonly implements in the ABE method. However, literatures in other areas suggested the Euclidian-based distance is too simplistic, and often fails to handle and make use of information provided by nominal attribute type [5].

Our empirical study explores 6 probabilistic-based distance functions, the original and the extension, proposed by Wilson et al. and Blachnik et al. respectively [5], [6]. We compare the 6 functions against the commonly-implemented Euclidian distance. Using leave-one-out validation technique over 5 standard-benchmarking datasets from the PROMISE repository [7], the results shows all the 5 datasets achieve a significant accuracy improvement varying from 6.30% to 48.14% based on RMSE (Root mean squared error) evaluational measure. Further, we apply a statistical test to the results and we conclude the probabilistic-based distance functions can improve the estimation accuracy of software effort estimation based on ABE reasoning, and this improvement is significant.

II. Probabilistic-base Distance Functions

The success of automated learning based systems depend upon good distance function to measure either similarity or dissimilarity specified by its purpose [5]. Well-recognized functions likely applicable to a wide-range of purposes are

1530-1362/13 $31.00 © 2013 IEEE
DOI 10.1109/APSEC.2013.78

the Euclidians (e.g. Minkowsky, Euclidean, and Manhattan) and the Correlations (e.g. Mahalanobis and Kendall's Rank Correlation). The Euclidians have become well known because of their simplicity, however; an estimation model based on Euclidians may not be the best choice, because they ignore a plenty of useful information between features in the similarity calculation [5].

In the conventional software ABE method, Euclidian distance and Hamming distance is combined in order to enable a computation for a software project case consisting of nominal project features (e.g. environment and programming language) and continuous project features (e.g. size and function point). Although this combined distance is mathematically correct and is meaningful for the distance calculation, it is remaining too simplistic [5], and may innately decrease the estimation performance, even so, the correlation-based functions are inapplicable to software ABE as they are good at measuring dissimilarity rather than similarly in which ABE requires. As a result, more sophisticate distance functions are necessarily explored in order to improve software cost estimation based on ABE.

A probabilistic-based distance function is the distance function that estimates the occurrence probability for each project feature attribute. In the overview, it considers a distance between two values to be closer if they imply the same thing (i.e. classification result or group). In real-world example, to estimate the nationality of a student focusing on hair color as the main feature, if the feature set contains only three values, black, blond, and red, where the goal is to find an Asian student. Whatever the other features are, a student who has black hair should be considered higher possibility to be an Asian than the one who has other hair color.

The following in this section, we explain the distance functions that appear in this paper.

1) Euclidian-Hamming Metric: An integration between Euclidian and Hamming distance metric was introduced by Wilson and Martinez [5]. It is the simplest metric that can handle applications consisting of both continuous and nominal feature types. The official name of this metric is Heterogeneous Euclidean-Overlap Metric (HEOM), however; researchers and practitioners in different area may encounter this distance metric with different designation. For example, when Shepperd et al., the pioneer of software cost estimation based on ABE, referred this metric in their work as a Euclidian distance [8].

HEOM calculates the distance between a pair of project cases (X, Y) using the following equation:

$$HEOM(X,Y) = \sqrt{\sum_{i=1}^{F} \delta(X_i, Y_i)} \tag{1}$$

Where F is the number of features, $\delta(X_i, Y_i)$ defines the distance between project case X and case Y scoped only feature i, and is calculated by :

$$\delta(X_i, Y_i) = \begin{cases} \frac{|X_i - Y_i|}{max_i - min_i} & i \text{ is continuous} \\ 0 & i \text{ is nominal and } X_i \neq Y_i \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

For the continuous feature type, $max_i - min_i$ is applied to normalize the feature i into the range of "0 to 1", which allows the distance product combined with nominal features type.

2) Value Different Metric (VDM): The Value Different Metric (VDM) was originally introduced by Stanfill and Waltz to be a reasonable distance function only for nominal attributes [5]. The distance measurement is the difference between posteriori probabilities, The formula is defined as:

$$VDM(X,Y) = \sum_{i=1}^{F} vdm(X_i, Y_i) \tag{3}$$

$$vdm(X_i, Y_i) = \sum_{j=1}^{C} (P(c_j|X_i) - P(c_j|Y_i))^2 \tag{4}$$

Where $p(c_j|X_i)$ is a conditional probability, which is approximated by $\frac{N(X_i,j)}{F}$. $N(X_i, j)$ is a frequency that accumulates instants having attribute value X_i, and implies an output class j. $N(X_i)$ is the total number of attribute having X_i value.

Because it is impossible to apply the VDM-based function on continuous attributes without modify the values, Wilson and Martinez proposed 4 distance functions based on extension of VDM to be applicable with continuous variables [5]. The simplest function namely Heterogeneous Value Different Metric (HVDM) applied a direct use of continuous attributes to VDM by multiplying each attribute value by $\frac{4}{\sigma_i}$, where σ is the standard deviation of feature i. This multiplication transforms continuous attribute values into an approximate range of nominal attribute values produced by VDM. The formal HVDM equation replaces $vdm(X_i, Y_i)$ from Equation 3 as follows:

$$hvdm(X_i, Y_i) = \begin{cases} \frac{|X_i - Y_i|}{4\sigma_i} & i \text{ is continuous} \\ vdm(X_i, Y_i) & \text{otherwise} \end{cases} \tag{5}$$

The rest of VDM-based functions apply discretization technique to transform continuous attribute values into arbitrary ranges. The transformed values become likely nominal values. Ordering by simplicity, Discretized Value Difference Metric (DVDM) ranks the first. DVDM discretizes all the continuous value attributes into nominal and simply let the VDM function calculate dist4ances. DVDM replaces $\frac{|X_i - Y_i|}{4\sigma_i}$ in Equation 5 by $vdm(discretized(X)_i, discretized(Y_i))$. However, this simplicity does not imply substandard as HEOM does. In an empirical benchmark on 48 application conducted by Wilson and Martinez, the estimation based on DVDM shared

39% the best accuracy achievement (8% as the single best distance function) comparing with HEOM and the other VDM members.

The next function is Interpolated Value Difference Metric (IVDM). It is considered the best function that probably produces the highest generalization estimation accuracy among VDMs in the original study [5]. Different from DVDM, which entirely treats continuous features as nominal, IVDM retains continuous interpretation, and applies a linear interpolation technique between midpoints of the discretized valued to find probability for other attribute values. IVDM replace $vdm(X_i, Y_i)$ from Equation 3 as following :

$$ivdm(X_i, Y_i) = \begin{cases} vdm(X_i, Y_i) & i \text{ is nominal} \\ vdm'(X_i, Y_i) & \text{otherwise} \end{cases}$$

$$vdm'(X_i, Y_i) = \sum_{j=1}^{C} (P''(c_j|X_i) - P''(c_j|Y_i))^2$$

$$P''(c_j|X_i) = P(c_j|X_i) + (P(c_j|di(X_i)+1) - P(c_j|X_i)) * \left(\frac{X_i - mid(discretized(X_i), i)}{mid(discretized(X_i)+1, i) - mid(discretized(X_i), i)} \right) \quad (6)$$

Here $mid(discretized(X_i), i)$ is the middle of discretize range of X_i in feature i, $mid(discretized(X_i)+1, i)$ is the middle of discretize range next to $discretized(X_i)$. $P(c_j|X_{i+1})$ and $P(c_j|X_i)$ are posterior probabilities calculate in the middle of $discretized(X_i)$ and the next discretize range $discretized(X_i)+1$.

Windowed Value Difference Metric (WVDM) is the last VDM-based distance function originally proposed by Wilson and Martinez. The extension from IVDM is that IVDM samples probability value at the midpoint of each discretization level and interpolates the value between each point to retain benefice information from continuous features, but WVDM sample the probability with more points. WVDM treats all the X_i values as midpoints, and sample $p(X_i)$ from all the values surrounding X_i in a window size in W, where W is equal to the number of values that are discretized into the same class with X_i.

VDM-based distance function is continually improved. A recent and interesting improvement was proposed by Blachnik et al. [6]. They had a useful detailed study on WVDM and proposed 2 new and interesting distance functions based on VDM namely Gaussian Value Difference Metric (GVDM) and Local Value Difference Metric (LVDM). GVDM use kernel smoothing technique to calculate the posterior probability. This technique flattens each datapoint into normal distribution. The key benefice is it can help reducing noises on the probability density estimation based on it. $GVDM(X, Y)$ is calculated using interpolation technique from $IVDM(X, Y)$ in Equation 6, but the notation of conditional probability $P(c_j|X_i)$ is modified to:

$$P(c_j|X_i) = norm\left(\sum_{k=1}^{N(c_j)} \exp\left(-\frac{X_{ik}}{2\sigma} \right)^2 \right) \quad (7)$$

Where $N(c_j)$ is the number of attribute values having the same output class c_j, X_ik is attribute values having the same value as X_i and having the same output class c_j. $norm$ is the normalization factor that divides $P(c_j|X_i)$ by the summation of all output class c over C.

LVDM boosts probability density for an attribute value X_i by discussing the surround attributes that maintain an approximate to X_i. LVDM is slightly modified Wilson and Martinez's WVDM. The different is LVDM estimates $P(c_j|X_i)$ by sampling $p(X_i)$ based on the value that X_i holds rather than number of attributes surrounding X_i. Here LVDM calculates $p(X_i)$ with the number of points falling in the range limited between $X_i \pm \frac{width_i}{2}$, where $width_i = \frac{max(i) - min(i)}{C}$.

III. Empirical Study

Our empirical study evaluates the benefit of measuring similarity of software project using probabilistic-based distance function in ABE. For this purpose, we firstly build an ABE benchmark framework, which utilize Leave-one-out validation technique (i.e. each project case becomes a test instance of the model built from the remaining data) to evaluate the prediction performance. The Leave-one-out is selected over cross-validation because it generates lower estimation bias and has higher variance [9]. Table I summarizes all the probabilistic-based distance functions explored in this study.

Table I: Summarize 9 probabilistic-based distance functions

Functions	Full Name	Proposed by
HEOM	Heterogenous Euclidean-Overlap Metric	[5]
HVDM	Heterogenous Value Difference Metric	[5]
DVDM	Discretized Value Difference Metric	[5]
IVDM	Interpolate Value Difference Metric	[5]
WVDM	Windowed Value Difference Metric	[5]
GVDM	Gaussian Value Difference Metric	[6]
LVDM	Local Value Difference Metric	[6]

The datasets we use in this study were selected from the PROMISE repository [7], where standard benchmark datasets are available for software cost estimation research. To sum up, this study explores the following research questions:

RQ1: Do the probabilistic-based distances imply higher estimation performance?

RQ2: If they do imply, which distance function would suggest the highest generalization accuracy?

A. Methodology

We construct ABE benchmarking framework utilizing the computational power of GPUs. The core engine and implementation are implemented following the ABE-CUDA algorithm proposed by Phannachitta et al. [4]. The ABE-CUDA optimizes ABE with both best feature subset and best k-NN neighbors selection regardless to the best subset. The original Euclidean distance function presented in the ABE-CUDA is an evolutional baseline distance function. We then replace the Euclidean distance with 6 distance probabilistic-based distance functions presented in Table I.

There is an initial limitation in order to directly apply probabilistic-base distances function among software project cases, which is the classified outputs (C) are not an available feature in software project datasets. To resolve the problem, we borrow an idea from Fuzzy Analogy [10], which reasonably describes an individual attribute with its linguistic definition (i.e. low, medium, and high) rather than its original continuous value type. By the process, we apply discretization technique (e.g linear discretization) to categorize attributes following their linguistic definition.

In the following case studies, we describe attributes in terms of linguistic value where discretization is required in 5 levels as $very\ low, low, medium, high,$ and $very\ high$. Each level equally distributes continuous attributes by frequency.

B. Performance Metrics

We measure the performance using a standard metric RMSE (Root mean squared error) as an evaluation measure. RMSE is selected over another measurement more possibly encounter in literature namely MMRE, because MMRE and its similar metrics measuring relative errors (e.g. MMER and MBRE) are reported by Foss et al. in their extensive study as unreliable metrics and may provide misleading information [11]. RMSE measures errors in absolute, and is defined as:

$$RMSE = \frac{1}{n}\sqrt{\sum_{i=0}^{n}\left(Actual_i - Predicted_i\right)^2} \qquad (8)$$

Where $Actual_i$ is the actual cost of a project case i, $Predicted_i$ is the estimated cost of i, n is the total number of project cases.

Because we compare the approach in the same framework and environment, measuring the performance with absolute error is fair and reasonable. In the following case studies, we build the ABE models (i.e. selecting best subset and best k-NN) based on $RMSE$ metric. Note that RMSE is measuring error, the lower value implies the better.

Further, to justify the error measures we used in this study, we evaluate the error result with a statistical test namely Wilcoxon signed-rank test [12] with 90% confidence level. Wilcoxon signed-rank test is a non-parametric statistical hypothesis. Wilcoxon tests if the two result sets are from the same distribution. The difference form t-test is Wilcoxon ignores the signs, and compares the ranks for the positive and the negative differences, resulting it is more sensible. Also the test is useful when the distributions are unclear.

Wilcoxon win-tie-loss counting is a popular procedure to compare the overall performance of different methods [12]. In case of a comparison between method a and b, if an outcome produced from the two methods fails the predefined confidence of the test (i.e. 90% or 95%), the two methods are "tie". Otherwise, the method that has a higher performance will "win", and the worse will "loss". After the test is done with all datasets, the accumulated score of win, tie, and loss between the two methods will result the overall performance between the two methods.

C. Datasets

Table II: The 11 PROMISE datasets

Dataset	Cases	Features	Included Features	Nominal
Desharnais	77	11	10	1
China	499	19	12	2
Coc81	63	19	18	1
Nasa93	93	24	23	21
Maxwell	62	27	26	22

Table II summarizes the benchmark datasets that were used in our case studies. The 5 datasets were collected form real-world software projects and made available in the PROMISE repository [7]. Because dependent variables may generate bias toward the estimation, the forth column of Table II show the exact number of independent features included in our experiments.

IV. Result

This section demonstrates our empirical study result. Each distance function included in our study was tested on 5 datasets using Leave-one-out validation technique. Table III reports the maximum achievable estimation accuracy in term of RMSE when both best feature subset selection and best k-NN are optimized.

The best improvement for each dataset is highlighted in Table III. In general, larger datasets achieve higher improvement than the smaller. Comparing the percentage difference (i.e. percent_diff $(a,b) = \frac{|a-b|}{Avg(a,b)}$) of RMSE between HEOM and the competing productivity-based functions, Desharnais has the highest improvement at 13.48%, China is roughly 6%, Coc81 improves over 48%, Nasa93 is 16.5%, and Maxwell is 22%. Furthermore, all the 6 probability-based distances improves the estimation accuracy for the Coc81, Nasa93, and Maxwell dataset. Poor improvement has only displayed in the China dataset. This

Table III: Summary of Estimation Accuracy measuring with RMSE metric

Functions	Dataset				
	Desharnais	China	Coc81	Nasa93	Maxwell
HEOM	2719.0	4584.5	1079.7	665.2	5826.8
HVDM	2644.4	4584.0	919.6	640.7	4807.3
DVDM	3028.6	5213.8	**660.8**	**563.7**	5641.8
IVDM	2523.2	4629.7	904.3	630.9	**4643.5**
WVDM	**2375.7**	**4304.5**	1032.9	588.7	5122.4
GVDM	2859.7	4586.3	998.3	617.7	5686.1
LVDM	2741.9	4859.8	928.9	628.5	5662.6

exceptional high error seen in the Chana dataset is an issue commonly discussed [13]. The main problem is China is a diverse dataset, which was collected from different software projects, causing the similar project cases retrieved from the ABE process may not actually similar to the target case.

To verify the improvement of the estimation accuracy, we apply Wilcoxon signed-rank test with 90% confidence. The outcome from Wilcoxon test will tell whether the improvement or degradation is significant. Table IV illustrates win, tie, and loss in the aspect of probability-based distance. In each cell of Table IV, positive value indicates the percentage difference that a distance in that row wins the HEOM, "tie" indicates the difference is insignificant, and negative value indicates the percentage difference that a distance in that row tailing by the HEOM. We summarize total number of win-tie-loss for each function at the column-end, and at the row-end for each dataset.

Table IV: Wilcoxon Signed-Rank Test

Functions	Dataset					W	T	L
	Desharnais	China	Coc81	Nasa93	Maxwell			
HVDM	t	t	31.84	t	t	1	4	0
DVDM	t	t	48.14	16.53	t	2	3	0
IVDM	7.47	-0.98	17.69	5.30	t	3	1	1
WVDM	13.47	6.30	4.43	t	t	3	2	0
GVDM	t	-0.04	7.83	t	t	1	3	1
LVDM	t	-5.82	15.02	t	t	2	2	1
W	2	1	6	2	0			
T	4	2	0	4	6			
L	0	3	0	0	0			

Table IV display the improved performance with statistical verification. We then analyze this result to answer our two main research questions.

RQ1: Do the probabilistic-based distances imply higher estimation performance?

From 6 probabilistic-based functions we have tested, all the distance functions outperform the Euclidian-based HEOM. In Table IV, half of the experimental outcomes tested by Wilcoxon are not significantly different. The reason is our ABE test models the actual cost from the previous software project without a massive value modification. As a result, the estimation value in our test probably remains in the same population as what HEOM does. In such case, if the improvement can pass the significant test, there will be no longer question against the improvement.

To sum up the evidences demonstrated in Table IV, we conclude our RQ1 as the probabilistic-based distances do imply the higher estimation performance, and the improvement is significant.

RQ2: Which distance function would suggest the highest generalization accuracy?

To develop a measure for this question, we score the improvement ranking from each distance functions by $2^{-(i^{th}\text{ rank})}$ in each dataset. The score counts only when a distance function overtakes HEOM, whether or not it "win" by the Wilcoxon signed-rank test. Then, we average sum the score term of each function by number of dataset, resulting this score is range between "0 to 1". Let denote this score as Generalization index (i.e. G-Index).

Table V: G-Index Scores from 9 Probabilistic-based Distance Functions

	HVDM	DVDM	IVDM	WVDM	GVDM	LVDM
G-Index	0.31	0.43	0.45	0.56	0.04	0.05

Table V shows the G-Index of the 6 distance functions. Distance functions having higher G-Index imply higher generalization accuracy. WVDM distinctively outperforms the other distance functions. As a result we conclude WVDM as the chosen one when the computational resource is available for using only one single function. Otherwise, we suggest to optimize the model by exhaustively searching for the best function for an individual dataset.

A. Discussion

From 5 datasets we used in this study, the project features in Desharnais, China, and Coc81 are mostly continuous, and Nasa93 and Maxwell are mostly nominal. As the entire distance function set has improved Coc81 and Nasa93 dataset, we assume both datasets have higher quality than the others. In table III, the result shows the probabilistic-based distance function have done well whether the number of continuous or nominal are higher. The distance functions that have improved all the datasets are HVDM and WVDM. Interestingly, the HVDM hypothesis only extends HEOM for

nominal features (see equation 5). The improvement from the first 3 datasets are small, when applying HVDM as the distance function, because the difference between HEOM and HVDM in such case is only the normalization term (i.e. HEOM normalizes by range, but HVDM normalizes by standard deviation). HVDM improves over HEOM slightly better for the Nasa93 and Maxwell dataset, where P-value was taken into account for the nominal variable. DVDM gives an outstanding accuracy on Coc81 and Nasa93; although, its extension form HVDM is a discretization over the continuous variables. This observation emphasize that the more application of probabilistic, the higher estimation performance can be attained. We can see IVDM and WVDM are complementing from each other in Table III. WVDM performs very well for a case that DVDM is lesser, while either IVDM or WVDM perform poorly when DVDM has an outstanding performance, hence; a future work is required to explain criteria to select a distance function for the others.

V. Conclusion

This paper explores 6 distance functions based on probability estimation in an attempt to improve software projects similarity measure, which directly influence the predictive power of software cost estimation using analogy (ABE). A number of experiments were carried out aiming to address the research inquiry that "An estimation based on probability-based distance functions would significantly improves the estimation accuracy". We selected root mean squared error metric (RMSE) to evaluate their efficacy. The result shows that the estimation accuracy of all 5 real-world project datasets [7] included in this study were improved in different magnitude, and the probabilistic-based function namely Discretized Value Different Metric (DVDM) provided the highest estimation performance overall. DVDM considers a continuous attribute as a discretized nominal, and estimate the probability based on the population in the same discretized class. From this experiment, the best performance was gained at 48.14%.

To conclude if the probability-based distance functions are generally an improvement, and to determine the distance function that giving the highest generalized accuracy, the results were further compared and verified using Wilcoxon signed-rank test for their statistical significant. The verification procedure shows accuracy improvement in all the 5 datasets with different functions utilized. The selection criteria of the different functions are remaining our future work to investigate. Overall the results are promising, and indicating that probabilistic-based similarity measures provides significant improvement over standard Euclidean similarity measures for applications such as Software Cost Estimation in Software Engineering.

References

[1] M. Jorgensen and M. Shepperd, "A systematic review of software development cost estimation studies," *IEEE Trans. Softw. Eng.*, vol. 33, no. 1, 2007.

[2] C. Mair and M. Shepperd, "The consistency of empirical comparisons of regression and analogy-based software project cost prediction," in *Proceedings of 2005 International Symposium on Empirical Software Engineering (ISESE)*, 2005, p. 10.

[3] G. Kadoda, M. Cartwright, L. Chen, and M. Shepperd, "Experiences using case-based reasoning to predict software project effort," in *Proceedings of the 4th International Conference on Evaluation and Assessment in Software Engineering (EASE)*, 2000.

[4] P. Phannachitta, J. Keung, and K. Matsumoto, "An empirical experiment on analogy-based software cost estimation with cuda framework," in *Proceedings of the 22nd Australasian Software Engineering Conference (ASWEC)*, 2013, pp. 165–174.

[5] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *J. Artif. Int. Res.*, vol. 6, no. 1, pp. 1–34, 1997.

[6] M. Blachnik, W. Duch, and T. Wieczorek, "Heterogeneous distance functions for prototype rules: influence of parameters on probability estimation." *Studia Informatica*, vol. 1, no. 2, pp. 19–30, 2006.

[7] T. Menzies, B. Caglayan, E. Kocaguneli, J. Krall, F. Peters, and B. Turhan. (2012, Jun) The promise repository of empirical software engineering data. [Online]. Available: http://promisedata.googlecode.com

[8] M. Shepperd and C. Schofield, "Estimating software project effort using analogies," *Software Engineering, IEEE Transactions on*, vol. 23, no. 11, pp. 736–743, 1997.

[9] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, 1995, pp. 1137–1143.

[10] A. F. Sheta, A. Ayesh, and D. Rine, "Evaluating software cost estimation models using particle swarm optimisation and fuzzy logic for nasa projects: a comparative study," *International Journal of Bio-Inspired Computation*, vol. 2, no. 6, pp. 365–373, 2010.

[11] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A simulation study of the model evaluation criterion mmre," *IEEE Trans. Softw. Eng.*, vol. 29, no. 11, pp. 985–995, 2003.

[12] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[13] J. Keung, E. Kocaguneli, and T. Menzies, "Finding conclusion stability for selecting the best effort predictor in software effort estimation," *Automated Software Engineering*, pp. 1–25, 2012.