# The Potential Benefit of Relevance Vector Machine to Software Effort Estimation

Liyan Song, Leandro L. Minku, Xin Yao
CERCIA, School of Computer Science
The University of Birmingham
Edgbaston, Birmingham B15 2TT, UK
{lxs189,l.l.Minku,x.yao}@cs.bham.ac.uk

## ABSTRACT

Three key challenges faced by the task of software effort estimation (SEE) when using predictive models are: (1) in order to support decision-making, software managers should have access not only to the effort estimation given by the predictive model, but also how confident this model is in estimating a given project and how likely other effort values could be the real efforts required to develop this project, (2) SEE data is likely to contain noise, due to the participation of humans in the data collection, and this noise can hinder predictions if not catered, and (3) data collection is an expensive task, and guidelines on when new data need to be collected would be helpful for reducing the cost associated with data collection. However, even though SEE has been studied for decades and many predictors have been proposed, few methods focus on these issues. In this work, we show that relevance vector machine (RVM) is a promising predictive method for addressing these three challenges. More specifically, it explicitly handles noise, it provides probabilistic predictions of effort, and can be used to identify when the required efforts of new projects should be collected for using them as training examples. With that in mind, this work provides the first step in exploiting RVM's potential for SEE by validating both its point prediction and prediction intervals. It then explains in detail future directions in terms of how RVMs can be further exploited for addressing the above mentioned challenges. Our systematic experiments show that RVM is very competitive compared with state-of-the-art SEE approaches, being usually ranked the first or second in 7 across 11 data sets in terms of mean absolute error. We also demonstrate how RVM can be used to judge the amount of noise present in the data. In summary, we show that RVM is a very promising predictor for SEE and should be further exploited.

## Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management—*Cost estimation*; I.2.6 [**Artificial Intelligence**]: Learning


*PROMISE* '14, Sep 17-17 2014, Torino, Italy
ACM 978-1-4503-2898-2/14/09.
http://dx.doi.org/10.1145/2639490.2639510

## General Terms

Experimentation, Algorithms

## Keywords

Software Effort Estimation, Machine Learning, Prediction Interval, Relevance Vector Machine, Data Collection Guidance, Effort Noise

## 1. INTRODUCTION

Software effort estimation (SEE) is the process of predicting the effort (e.g., in person-month or person-hour) required to develop a software system. It often takes place in the very early stage of software development and, based on it, project managers make important decisions such as the budget, the schedule of the project, and the bidding price [4]. Though being investigated for decades, there are still several problems hindering the practical use of SEE models. Among all of them, we focus on three challenging problems, as explained next.

First, most existing SEE methods produce merely point estimations, i.e., they provide a single estimation of the effort required to develop a given project, rather than providing prediction intervals or the level of confidence associated with the estimations. However, there are several sources of uncertainty in the context of software estimation task [37, 18, 16]. Simply relying on a point estimation may ignore the uncertain factors and may lead project managers to wrong decision-making. Therefore, it is safer to produce interval predictions along with a most likely point estimation for practical purpose [37]. Moreover, interval predictions can provide more flexibility to project managers. For instance, in bidding process, if the competition is very fierce the project manager can report a lower price within the interval to enhance the chances of winning; while, if the competition is less fierce, he/she can propose a higher price for getting more profit to the organization. In summary, it would be good to have predictions intervals associated with confidence levels in the context of SEE [13, 17].

Second, data quality is considered to be an important factor of prediction accuracy [10], and noise in effort is an important aspect of data quality. Effort noise is defined as the difference between the real and the collected required effort. It is usually immeasurable, as we usually do not know the real required effort, only the collected one[1]. Unfortunately,

[1]Note that effort noise is different from *absolute residual*, which is related to a specific SEE approach, and is defined as the difference between the predicted and the collected required efforts.

the efforts collected by project managers are very likely to contain noise due to the participation of humans. For instance, some employees may work in more than one project at the same time, and it may be difficult for the project manager to clarify how much effort exactly they have devoted to a specific project. So, SEE faces the challenge of achieving good performance in the presence of noise. As noise is very likely to hinder the performance of classical predictive models which rely on perfectly collected outputs (efforts), a model able to deal with effort noise more effectively is necessary. In addition, it would be good to have a method able to detect the level of noise of a data set.

A third challenge of SEE task is the high cost associated with data collection [10]. Though the collection of independent features for completed projects is not very costly with the assistance of automated metric extraction tools, the collection of actual effort values is far more costly and may require considerable amount of time and workload [20, 21]. Rather than collecting as many completed efforts as possible, it is advised to collect the most 'relevant' data, i.e., the ones which are representative of the range of different projects and have potential to improve the generalization performance. Therefore, it is of great practical use to propose guidelines to support project managers in deciding whether or not to collect the effort associated with a new project.

In this work, we aim to introduce a promising method, Relevance Vector Machine (RVM) [39], which has the potential to address these three challenges. We will show that RVM explicitly models effort noise, and that the estimated level of effort noise can be used to analyze the data set. Following the Bayesian framework, RVM conducts a probabilistic prediction for a new project, where each estimated effort is associated with a probability. We will show that it is possible to derive a prediction interval associated with a confidence level based on this probabilistic prediction. Moreover, Automatic Relevance Determination (ARD) [30] used by RVM can help us to decide whether the required effort of a new project should be collected.

Before the attempts to address the three problems, the first step is to investigate whether RVM can perform well in comparison with state-of-the-art SEE approaches when used as a point estimator. This is because if RVM performs very badly for SEE, none of the conclusions based on it are reliable. On the other hand, if RVM is competitive against other SEE approaches, its conclusions are more reliable. Moreover, this would also mean that modifying RVM to take into account specific features of SEE, such as non-normal distribution of noise, is a promising area of research which could improve SEE and its associated problems further.

Our experiments show that RVM is very competitive compared with state-of-the-art SEE approaches, being usually ranked top two in 7 out of 11 data sets in terms of mean absolute error, and behaving statistically similar to approaches that have been showing to perform well in SEE. So, given the potential of RVM to address the key challenges related to SEE and its competitive performance, we encourage future research in exploiting and improving RVM further in order to fully address the three challenges described above.

The main aims of this paper are to validate RVM in terms of its point estimation and prediction intervals in the context of SEE, and to provide further detailed directions on how to use RVM for addressing the challenges described above. In summary, this paper investigates the following points:

P1 How well can RVM perform compared with other well-established SEE predictors when used as a point estimator? This, together with the potential of RVM in dealing with the three challenges described above, allows us to know how promising RVM is in the context of SEE. It is worth noting here that our aim is not to show that RVM is a supreme predictor which outperforms all other existing SEE approaches, but to investigate whether it presents competitive performance and should be investigated further, given its potential to address the three challenges described above.

P2 How to provide prediction intervals associated with confidence levels based on RVM? Does the proposed method provide reasonable prediction intervals? This allows us to know how well RVMs deal with the first challenge described above.

P3 How to use RVM to detect the level of noise of SEE data, which is one of the factors associated with data quality in SEE? This provides detailed directions on how to address the second challenge described above.

P4 How to use RVM to decide whether the required effort of a new project should be collected, i.e., to guide data collection? This provides detailed directions on how to address the third challenge described above.

The remainder of this paper is organized as follows. Section 2 presents related work on machine learning for SEE. In section 3, we briefly introduce RVM and its usage in SEE. After the experiment design in section 4, the performance of RVM as a point estimator will be evaluated in section 5 (P1). In section 6, we propose a method for predicting an interval associated with a confidence level, and its validation (P2). Section 7 shows our original solutions on how to address (P3) and (P4). Threats of validity are discussed in section 8. The paper is concluded in section 9.

# 2. RELATED WORK

In this section, we will discuss some related literature in three subtopics as follows. It is worth noting that none of the existing approaches is able to solve all the three problems explained in section 1 simultaneously.

## 2.1 Machine Learning Techniques in SEE

Many machine learning (ML) algorithms have been proposed for SEE in the past two decades. Among them, $k$-Nearest Neighbor ($k$-NN), artificial neural networks (ANN), and regression tree (RT) were three most frequently used ones [40, 10].

Shepperd and Schofield (1997) [35]'s work was a landmark study in SEE. They used $k$-NN, also known as analogy-based estimation (ABE) or case-based reasoning (CBR), based on normalized attributes and Euclidean distance as similarity measure. Despite its simplicity, this method has remarkably been shown to be competitive in terms of how frequently it obtained the best Mean Absolute Error (MAE) compared with other approaches. Nevertheless, when it is not among the best approaches for a certain data set, it can perform considerably worse than the best in terms of MAE [27].

Regression tree is another popularly used SEE predictor [8]. It is equipped with easy-to-understand structures such as if-then rules to separate examples based on their feature values. It was reported to perform well in comparison to several other learning machines and can be a good choice of base learner for ensemble approaches [27].

MultiLayer Perceeptron (MLP) is another widely used machine learner in the scope of ANN techniques. It is composed of at least three layers of neurons and the neurons of a certain layer are connected to all neurons of the next layer. Existing studies have been contradictory in their conclusions on the usefulness of MLPs in SEE. A recent work [36] revealed that MLPs can achieve competitive performance, but only if their parameters are very well tuned. The high sensitiveness of MLPs to parameter choices can be the reason for the contradictory results in the literature.

Besides single SEE approaches, recent works have been emphasizing the relatively good performance achieved by ensemble techniques such as Bagging ensembles of RTs (Bagging+RTs) and Bagging ensembles of MLPs (Bagging+MLPs) [23, 25, 27, 28]. Ensembles are groups of learning machines combined together with the aim of improving predictive performance. For instance, it was reported that Bagging+RTs were the most frequently ranked the best and when they were not the best, they rarely performed considerable worse than the best approach in a study involving three ensemble and three single learning machines [27].

## 2.2 Automatic Interval Prediction for SEE

So far, only a few studies considered the development of automatic models providing uncertain predictions in SEE.

The work of Angelis and Stamelos in 2000 [2] was considered as the first attempt to suggest effort prediction intervals. They presented a method to compute an interval prediction by the use of the bootstrap mechanism with (parametric bootstrap) and without (non-parametric bootstrap based on $k$-NN) the theoretical assumption on the population distribution of the data set. As pointed out by the authors, this method was not without problems. The quality of interval prediction relied on the good quality of training projects. In 2001, the same authors proposed similar interval predictions but in project portfolio cost estimation problem based on a bootstrap model.

Later in 2003, Jorgensen and Sjobergis presented and evaluated simple effort prediction interval approaches [17], based on the assumption that the empirical distribution of estimation accuracy was consistent between the historical and the predicted data. Human judgement or more formal cluster analysis may be required for identifying the projects with the same expected degree of estimation uncertainty as the predicting project.

In 2011, Klas et al. proposed the integration of bootstrapping in the context of hybrid software estimation [19]. Their empirical results showed improved and more realistic uncertainty estimates.

Another work related to interval prediction was conducted in 2005 by employing ordinal regression to classify a new project to a predefined effort category (e.g., Low, Nominal, High or Very high) [33]. The historical completed projects with actual efforts were required to predefine the cost categories and build the model. The point estimation can be obtained by using the mean or median value of the category the predicted project falls in.

There were some approaches based upon Bayes' Theorems [9, 38, 31, 26] to infer the uncertain prediction for software effort. For instance, in [9], human experts defined the structure of the model (COCOMO II) and the prior joint distribution of the unknown parameters. Then, the posterior of these parameters were calculated by Bayes' theorem. Other Bayesian-based SEE models [38, 31, 26] were represented by Bayesian belief networks (BBNs), where the prediction models were designed as a structural causal model by humans. One of the limitations of these Bayesian models is that they require intensive expert participation to build a predefined structural model for each organization or data set.

## 2.3 Data Analysis via Learning Machines

To the best of our knowledge, there has been no work investigating the distribution of label noise or providing guidance for label collection via predictive models. Thus, in this section, we review some of the works investigating the characteristics of SEE data via effort estimators.

In 2001, Shepperd and Kadoda explored the relationship between the characteristics of data sets and the performance of three SEE approaches: stepwise regression (SWR), rule induction (RI) and cased-based reasoning (CBR) in simulated data sets (not real SEE data sets). It is believed that this exploration between prediction accuracy of the estimators and the characteristics of data sets can lead to a better understanding of in which circumstances a particular technique is better.

Kocaguneli et.al [22] conducted instance selection to automatically prune irrelevant instances by a $k$-NN based algorithm. Though their initial aim was to improve the prediction performance using Cross-Company (CC) data, their mechanism disclosed the relevant SEE data for the used approach indirectly. Another work of the same authors also contribute to this issue. In [24], they employed a $k$-NN based method, called QUICK, to characterize the essential content of SEE data. QUICK calculates the Euclidean distance between rows (instances) and columns (features) of SEE data and prunes similar features and distant instances (outliers). The remaining content is considered as essential. Their results showed that the essential content of SEE data was usually small.

Minku and Yao [27] analyzed regression trees to determine the most important input features of SEE data sets. In another study [29], they presented a ML-based data analysis to provide insight into the behaviour of a company in comparison with other companies over time, helping project managers to (1) decide if and when new policies to improve a given company's productivity are necessary, (2) monitor the results of adopting new policies, and (3) decide how to improve productivity.

# 3. RELEVANCE VECTOR MACHINE

In this section, we will briefly introduce Relevance Vector Machine (RVM) (Tipping,2001) [39]. For simplicity and better understanding, we deliberately omit many details. Please refer to Chapter 7.2 of [5] and papers [39, 11] for further information.

RVM is a typical linear model, and can be represented for output $y$ given input vector $X$ as

$$y = \boldsymbol{\theta}^T \Phi(X) + \epsilon \tag{1}$$

$\boldsymbol{\theta}$ are the model parameters, $\epsilon$ represents the probability dis-

tribution of effort noise, $\Phi$ are known as the basis functions that give linear regression substantial flexibility for modeling cases where $y$ and $X$ are non-linearly related. In the case of RVM, each basis function associates with a separate training sample, measuring the distance of this training sample to the predicted project. There are several possible choices for the basis functions. In this work, we employ non-normalized Gaussian kernel $\phi_j(\mathbf{x}) = exp\{-\frac{(\mathbf{x}-\boldsymbol{\mu_j})^2}{2s^2}\}$ as our basis function, where $\boldsymbol{\mu_j}$ is the $j$-th training sample, and $s$ controls their spatial scale. This function was chosen for its simplicity and localization [27]. It is also noteworthy that $s$ is the only tuned parameter.

From the linear model, we can see that RVM explicitly encodes the effort noise ($\epsilon$). Ideally, the distribution of effort noise should be proposed by carefully investigating the residues between the real and the collected required efforts reported from the training samples. However, this is usually impossible since the real required efforts exempted from noise are not known in reality. Among a range of probability distributions, Gaussian distribution can often match the actual distribution of the noise in real-world processes reasonably well, and it is easy to deal with due to the well-developed mathematical theory behind it [1]. Thus, in this paper, we assume that label noise is normally distributed, i.e., $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The linear model Eq. 1 will contribute to a Gaussian *likelihood* with respect to model parameters $\boldsymbol{\theta}$.

Following the Bayesian framework, by introducing a proper (Gaussian) *prior* to the model parameters $\boldsymbol{\theta}$, we can obtain the *posterior* of $\boldsymbol{\theta}$ as a Gaussian distribution proportional to the product of the Gaussian *prior* of the parameters $\boldsymbol{\theta}$ and the Gaussian *likelihood* of all training samples, according to Bayes' Rules $p(x|y) = p(x)p(y|x)/p(y)$. Therefore, the probabilistic prediction of a new project is again Gaussian, and can be derived by the convolution of the (Gaussian) *posterior* of $\boldsymbol{\theta}$ and Gaussian distribution of linear model Eq. 1 (i.e., the *prior* of a project effort). As pointed out in Sec. 6, for a new project, we use the mean as a point estimation because it is the most likely value the predicted effort can achieve.

Here, the probabilistic effort prediction is caused by the Gaussian assumption of effort noise, and requires no further assumption upon itself. It is also noteworthy that the Gaussian effort prediction relates to but does not equal to Gaussian effort noise. In Bayesian terms, the assumption of Gaussian effort noise provides a *prior* knowledge for a project effort before any training sample is observed, which is the case that project effort following $\sigma^2$-Gaussian distribution with its mean value defined by (1); in contrast, the Gaussian prediction of a new project is a *posterior* estimation after training samples are observed and used to train the model.

A key characteristic of RVM is that people introduce a separate *hyperparameter* for each of the model parameters $\theta_i$ instead of a single shared *hyperparamter* as in classical Bayesian linear regression [5]. This mechanism results in *sparsity*, for which a large subset of model parameters $\boldsymbol{\theta}$ will be driven to zero with the corresponding training samples pruned [11]. This formulation of prior is a type of *automatic relevance determination* (ARD) prior [39, 30]. It is worthwhile to emphasize that with the mechanism ARD, RVM can automatically choose 'relevant' projects, namely *relevance vectors*, from training samples, which can capture the major structure of the training space [39]. And, based on this fact, we can introduce our guidance on whether or not the effort of a new project should be collected, given that RVM is applied. For instance, if a new project falls in a region of the input space where several training projects were pruned in the process of ARD, this indicates that some relevance vectors outside this region are capable of capturing the structure of this region. Thus it is suggested not to collect the required effort of this project for reducing the cost associated with data collection. More guidance will be discussed in Sec. 7.2.

**Table 1: Data Sets**

| Repository | Name | #(Project) | #(Feature) |
|---|---|---|---|
| PROMISE | Maxwell | 62 | 23 |
| | Kitchenham | 145 | 3 |
| | Cocomo81 | 63 | 17 |
| | Nasa93 | 93 | 17 |
| ISBSG | Org1 | 76 | 3 |
| | Org2 | 32 | 3 |
| | Org3 | 162 | 3 |
| | Org4 | 122 | 3 |
| | Org5 | 21 | 3 |
| | Org6 | 22 | 3 |
| | Org7 | 21 | 3 |

# 4. EXPERIMENT DESIGN

## 4.1 Data Sets and Performance Measurement

The discussions presented in this paper are based on several data sets from the PRedictOr Models In Software Engineering Software (PROMISE) Repository [32] and from the International Software Benchmarking Standards Group (ISBSG) Repository [15] Release 10. Table 1 contains a basic description of the used data sets. For further preprocessing procedures and project IDs used in this paper, please refer to [27, 36] except that outliers are not removed in this work.

We apply 10 times 10-fold Cross-Validation (CV) to validate the performance of the investigated SEE approaches. The procedure is to repeatedly run ten times of 10-fold CV with different sample orders each time, in order to cancel out the impact of project order. We used 10-fold CV because SEE data sets are usually small, which may cause high bias if using small $k$ ($k = 2$ in $k$-fold) due to the lack of training data; whereas large $k$, such as leave-one-out with $k$ equal to the size of the data set, may result in high variance [14]. We performed preliminary experiments using 5 times 2-fold CV, leave-one-out and 10 times 10-fold CV. The results also showed this tendency. Therefore, we consider 10-fold CV as a suitable choice.

The performance was measured by the Mean Absolute Error (MAE) over the prediction defined as $\sum_{i=1}^{N} \frac{|y_i - \widehat{y_i}|}{N}$. MAE was chosen for being a symmetric measure not biased towards under or overestimation [34]. We did not use Mean Magnitude of Relative Error (MMRE) because it was shown to be a biased measurement [12] and we do not want to include a misleading measurement in our results.

## 4.2 Learning Machines Investigated

To evaluate the relative performance of RVM, we compare the point estimations of RVM with the following five approaches: $k$-NN, RTs, MLPs, Bagging+RTs, and Bagging+MLPs. We do not investigate Bagging+$k$-NN because Bagging is known to improve accuracy for unstable pro-

cedures[2] such as MLPs and RTs, whilst it may slight degrade the performance of stable procedures such as $k$-NN [7]. WEKA is used to implement these five approaches: $k$-NN was based on IBK with normalized attributes and Euclidean distance, RTs was based on REPTree without pruning, and the others were based on the corresponding classes with the same name. MLPs were set to automatically normalize dependent and independent variables and to use the nominal to binary filter. RVM was implemented in MATLAB due to its advantage in matrix computation and the absence of the implementation in WEKA.

RTs, Bagging+RTs and Bagging+MLPs were chosen due to their good performance in comparison to several other ML approaches in SEE [27]. $k$-NN is among the simplest SEE approaches, and shown to perform frequently well [35]. MLPs have not been shown to perform well in SEE; however, a recent study showed that after finely tuned they can achieve competitive performance compared with the approaches investigated in [36]. In summary the main reason for including these five approaches is their relatively good performance reported by current literature. To better acknowledge the performance of RVM as a point estimator, we should compare it with the well-performing algorithms.

## 4.3 Parameter Settings Investigated

The parameter values of the approaches investigated in this paper are shown in Table 2. Their default values are emphasized with bold and correspond to the default values of WEKA. For RTs, the maximum depth of $-1$ means unlimited depth. For MLPs, the default value $a$ in #(hidden nodes) represents: $a = [\#(attributes) + 1]/2$. For RVM, $0.1 : 1.0 : 15$ denotes the values counting from 0.1 to 15 with step = 1.0.

All approaches except RVM and $k$-NN have more than one tuned parameter, and parameter settings are consisted by enumerating all values for each parameter with all the others set to the default ones. It is noteworthy that, like $k$-NN, RVM is easy to be tuned for only having one unknown (kernel) parameter.

Our analyses of all approaches are based on the performance with the best parameter settings. The best parameter setting (for each approach independently) is chosen from all the parameter settings shown in table 2, and is the one under which this approach can achieve the best performance by averaging the MAEs over the 10 runs of 10-fold CVs.

When data sets are large enough, it is recommended to use the CV procedure for building models and choosing parameters, and a separate test set not used by the CV procedure to evaluate the ML approaches. By using this scheme, the parameters leading to the model with the best validation error according to the CV procedure should be chosen to be further evaluated on the separate test set for comparing different approaches. However, in SEE, the data sets are too small. Using a separate test set would lead to two serious problems: (1) the number of examples for training and validation would be even smaller and (2) the test set itself would be very small, possibly leading to an invalid model evaluation for not representing the whole space well. So, we have opted not to use a separate test set, and to use CV both for choosing parameters and evaluating approaches. This reveals the potential performance achievable by different approaches given a good parameters choice. A separate analysis of the sensitivity of approaches to different parameter choices can be performed to further reveal the impact of different parameter choices in each approach's performance.

[2]Unstable here means when small changes in the training sample can result in large changes in the model.

**Table 2: Parameter Values**

| Approach | Parameters |
|---|---|
| $k$-NN | $k$(#neighbours)={**1**,3,5,7,9,11,13} |
| RTs | M(mim.#instance/leaf)={1,**2**,3,6,12,20}<br>V(mim.varience for split)=<br>{0.0001,**0.001**,0.01,0.1,10}<br>L(max.tree depth)={**-1**,2,6,10,15,20} |
| MLPs | L(Learning rate)={0.1, 0.2, **0.3**, 0.4, 0.5}<br>M(Momentum)={0.1,**0.2**,0.3,0.4,0.5}<br>N(#epochs)={100,**500**,1000}<br>H(#hidden nodes)={***a***,1,3,5,9} |
| Bagging | I(iteration for Bagging)={5,**10**,25,50,75}<br>All the possible parameters of the adopted base learners, as shown above. |
| RVM | One parameter in basis function = 0.1 : 1.0 : 15 |

# 5. PERFORMANCE COMPARISON

This section mainly aims at accomplishing P1: How well can RVM perform compared with other well-established SEE predictors when used as a point estimator? For a new project, we employ the *mean* of the predicted distribution of RVM as the point estimation, so that RVM can be compared with other state-of-the-art effort predictors. We use the *mean* because it is the most likely estimated effort.

Table 3 shows the performance of all investigated approaches in 11 data sets measured in MAE. The integers in the parentheses are the ranks of the corresponding approach compared with others. We can observe that RVM and $k$-NN rank the second right after Bagging+RTs that is the best estimator on average. This indicates a competitive performance of RVM. Next, we conduct statistical test for a more thorough understanding of the performance comparison.

By Friedman test with significance level at 0.05, we rejected null hypothesis (H0), which states that all algorithms are equivalent. The Friedman value is 4.35 which is much larger compared with critical value of 2.40. Post-hoc tests with Holm-Bonferroni corrections detect statistically significant difference between RVM and MLPs with $p$-value of 0.00142, indicating a better performance of RVMs compared with MLPs in SEE. However, Post-hoc tests with Holm-Bonferroni corrections cannot detect statistically significant differences between RVM and all the other approaches, which have been showing to perform well in SEE [27]. So, besides having potential to deal with the SEE challenges described in section 1, RVM has competitive performance in SEE. Therefore, RVM is a promising SEE estimator and should be further investigated. In particular, RVM may perform even better after adjusting it to adopt assumptions that are more specific to the context of SEE.

# 6. PREDICTION INTERVAL

## 6.1 Prediction Interval of RVM

The uncertainty of effort estimations can be characterized through effort *prediction intervals* (PIs). An effort PI comprises a minimum and maximum effort values associated with a stated *confidence level* (CL). For instance, a project

**Table 3: Performance of all Investigated Approaches. Performance is measured by MAE. The ranks of predictors are in the parentheses. Ave(Rank) denotes the average rank of each approach across all data sets.**

| Data Set | RVM | RTs | Bagging+RTs | MLPs | Bagging+MLPs | $k$-NN |
|---|---|---|---|---|---|---|
| Maxwell | 4192.40(2) | 4881.22(4) | 4181.26(1) | 5057.00(5) | 10284.98(6) | 4501.18(3) |
| Kitchenham | 1701.00(1) | 2202.64(6) | 1963.32(4) | 2182.23(5) | 1899.25(3) | 1804.88(2) |
| Cocomo81 | 591.21(4) | 573.09(1) | 588.29(3) | 654.73(5) | 582.25(2) | 666.27(6) |
| Nasa93 | 359.30(2) | 393.40(3) | 357.04(1) | 664.38(6) | 456.23(5) | 448.87(4) |
| Org1 | 3235.50(2) | 3641.39(5) | 3068.28(1) | 4356.59(6) | 3523.73(4) | 3274.68(3) |
| Org2 | 1829.00(1) | 2130.05(4) | 2112.60(3) | 2637.30(6) | 2150.11(5) | 1983.81(2) |
| Org3 | 1007.29(2) | 1011.94(3) | 1012.59(4) | 1215.63(6) | 1058.60(5) | 986.51(1) |
| Org4 | 3579.10(1) | 4374.10(4) | 4300.92(3) | 4997.01(6) | 4554.39(5) | 4019.28(2) |
| Org5 | 5938.10(6) | 5353.56(5) | 5154.14(2) | 5181.34(3) | 4958.01(1) | 5351.23(4) |
| Org6 | 2811.20(6) | 2806.88(5) | 2615.18(2) | 2707.65(4) | 2640.45(3) | 2432.00(1) |
| Org7 | 4915.00(3) | 4809.73(1) | 5114.43(5) | 5134.44(6) | 4939.80(4) | 4874.32(2) |
| Ave(Rank) | 2.73 | 3.73 | 2.64 | 5.27 | 3.91 | 2.73 |

**Table 4: Prediction Interval Derived by RVM in All Data Sets. The *median* values of actual efforts, PIs with $CL_{0.6827}$ and $CL_{0.9545}$ across each data set, and the estimated standard deviations are listed in the table. The median values of PIs are computed by picking the median lower/upper bound across all lower/upper bounds of the predicted projects.**

| Data Set | Actual effort | PI ($CL_{0.6827}$) | PI ($CL_{0.9545}$) | Estimated std |
|---|---|---|---|---|
| Maxwell | 5190 | [3177, 7339] | [1170, 9379] | 2128 |
| Kitchenham | 1557 | [0, 4107] | [0, 6407] | 1784 |
| Cocomo81 | 98 | [81, 237] | [21, 312] | 70 |
| Nasa93 | 252 | [0, 492] | [0, 755] | 250 |
| Org1 | 1213 | [0, 3536] | [0, 5861] | 2465 |
| Org2 | 2045 | [1185, 2116] | [694, 2564] | 439 |
| Org3 | 1090 | [0, 3278] | [0, 5119] | 1779 |
| Org4 | 3520 | [0, 8382] | [0, 12624] | 4360 |
| Org5 | 5506 | [307, 12257] | [0, 18645] | 5281 |
| Org6 | 2943 | [0, 8024] | [0, 12530] | 4453 |
| Org7 | 4456 | [897, 12664] | [0, 18569] | 5904 |

manager may be 95% certain that the predicted effort of a project falls between 500 and 2500 person-hour with the most likely effort value at 1500. PI should not be confused with another uncertain concept - confidence intervals (CIs). An important difference between them is that CI usually refers to the uncertainty associated with the unknown parameters of models, e.g., the the uncertainty of the mean value of an unknown distribution; whereas PI with a certain CL refers to the case that the real value is within the PI with the level of CL [3, pp.761-824].

In fact, based on the properties of the Gaussian distribution of effort prediction derived by RVM, we can present PI with any confidence level $\alpha$% by use of *cumulative distribution function* (CDF) of (Gaussian) effort estimation, $\mathcal{F}$, as follows:

$$PI_\alpha = [max(0, \mathcal{F}^{-1}(\frac{1-\alpha}{2})), \mathcal{F}^{-1}(\frac{1+\alpha}{2})] \quad (2)$$

$\mathcal{F}^{-1}(\beta)$ represents the effort value located on the $\beta$ percentile of Gaussian CDF. This defined PI is reasonable because according to the symmetry of Gaussian *probability density function* (PDF), the proportion of PDF between $\frac{1-\alpha}{2}$ and $\frac{1+\alpha}{2}$ equals to $\alpha$ which is just the CL, and this can be easily calculated by Gaussian CDF. Moreover, in fact effort is non-negative, so we confine the non-negative value for the left-hand-side. A point estimation can also be easily obtained by being assigned to the mean of the Gaussian predicted distribution, since it is the most likely effort value.

The most commonly used CL for PI in SEE was 0.90 ($CL_{0.9}$) [19]. However, several studies both in industry and academia indicated that there existed strong bias towards over-confidence for the effort PIs; i.e., the estimated effort PIs were too narrow to reflect its corresponding CL [18, 19]. Also, Jorgensen suggested the use of PIs with a lower CI [16]. Thus, we also suggest to use PIs with lower CL, say $CL_{60}$, which allows on average only two projects whose required efforts exceed the upper bound and only two projects whose required efforts fall below the lower bound.

In the rest of this section, we will introduce an easier method to derive the PIs of specific CLs, which are $CL_{0.6827}$ and $CL_{0.9545}$, and present the statistical description (e.g., median) of PIs for all the investigate data sets.

Based on "68-95-99.7" rule of Gaussian distribution [41], the prediction intervals with $CL_{0.6827}$ and $CL_{0.9545}$ can be derived as follows:

$$[max(0, \widehat{t} - i\widehat{\sigma}), \widehat{t} + i\widehat{\sigma}] \quad (3)$$

where $\widehat{t}$ is the predicted mean effort, $\widehat{\sigma}$ is the variance of predicted probability derived by RVM, and $i$ equals to 1 or 2 for $CL_{0.6827}$ or $CL_{0.9545}$, respectively. It is worth noting that it is possible to derive PIs with $CL_{0.6827}$ and $CL_{0.9545}$ according to Eq. 2, but it is easier to derive by Eq. 3.

Besides the fact that the PIs with these two CLs can be obtained easily by adopting Eq. 3 rather than Eq. 2, the reasons why we specialized PIs with $CL_{0.6827}$ and $CL_{0.9545}$ are as follows: (1) A $CL_{0.6827}$ is more or less sufficient in practical usage as usually only three of ten projects may go below the lower bound or exceed the upper bound. (2) The presence of $CL_{0.9545}$ is to have an insight of the higher CL that how well the PIs derived by RVM can achieve when considering a very strict, high CL. The reason with 99.70% excluded is because it is too strict requiring not even a mere violation in 100 PI predictions.

Table 4 lists RVM's the median values of actual efforts, the most likely point efforts and PIs with $CL_{0.6827}$ and $CL_{0.9545}$

**Table 5: $HitRate$ with $CL_{0.6827}$ and $CL_{0.9545}$. HitRate that is much smaller than the corresponding confidence level are in yellow (light grey).**

| Data Set | $HitRate_1(\%)$ | $HitRate_2(\%)$ |
|---|---|---|
| Maxwell | 44.61 | 74.84 |
| Kitchenham | 85.31 | 95.24 |
| Cocomo81 | 30.79 | 53.17 |
| Nasa93 | 68.60 | 82.15 |
| Org1 | 86.84 | 91.97 |
| Org2 | 33.13 | 56.25 |
| Org3 | 83.83 | 95.12 |
| Org4 | 76.07 | 90.16 |
| Org5 | 74.76 | 86.67 |
| Org6 | 90.91 | 95.46 |
| Org7 | 79.05 | 89.52 |

for each data set. The median values of PIs are computed by picking the median of lower / upper bound across all lower / upper bounds of the predicted projects. The aim is to illustrate an ordinary impression of the proposed prediction interval.

As showed in table 4, the actual efforts fall within the PIs with $CL_{0.6827}$ and $CL_{0.9545}$ in all data sets. As these results come from the median values, they indicate the validation of the derived PIs in ordinary cases.

Another observation is that, for the first 7 data sets, both PIs can be considered informative, i.e., the PIs are narrow enough to have practical usage. Taking Maxwell as an example, a project manager would have 68.27% confidence that the actual effort will fall into the interval $[3177, 7339]$. In view of the fact that the best approach among the investigated SEE estimators (Bagging+RTs, see table 3) has MAE around 4000, which approximately equals to the width of PI with $CL_{0.6827}$, we consider that the derived PI is of practical use. But the PIs of Org.4 - Org.7 are very wide and not very informative. Further study found that the predicted standard deviations (stds) of these 4 data sets were much larger than others, all greater than 4000. The large predicted stds directly caused the wide PIs. More investigation on the characteristics of these four data sets will be conducted as a future work.

In the case of wide PIs, we suggest to present a PI with a lower CL, say $CL_{0.60}$, for getting a more informative PI with an acceptable CL. A quick look at table 5 indicates that PIs behave better with $CL_{0.6827}$ than with $CL_{0.9545}$, that has a narrower PI and higher chance to pass the validation.

## 6.2 Validation of Prediction Interval

The most commonly used evaluation method for PIs is the hit rate (HitRate) [19, 16]. The underlying idea is that, if PIs with $CL_\alpha$ are estimated for $n$ software projects, it is expected that around $\alpha$ projects whose actual efforts fall inside the corresponding estimated intervals. HitRate can be calculated by first counting the number of projects whose efforts are within the PIs, and then dividing that by the total number of projects. If the estimated PIs with certain $CL_\alpha$ are realistic, the obtained HitRate should be around the chosen confidence level $CL_\alpha$. If the HitRate is higher, the estimated PIs are too wide; otherwise, the estimated PIs are too narrow.

We can validate the PIs with any CL by the method described in Eq. 2. Since in Sec. 6.1 we focused on PIs with $CL_{0.6827}$ and $CL_{0.9545}$, for consistency we validate the PIs with the same CLs in this section.

**Table 6: Effort Noise. '$\overline{std}_{noise}$' represents the average standard deviation of (Gaussian) effort noise; '$\overline{predEffort}$' denotes the average predicted effort; 'ratio' denotes the ratio between average standard deviation of effort noise and average predicted effort, measuring the level of label noise, and larger value indicates higher level of effort noise.**

| Data Set | $\overline{std}_{noise}$ | $\overline{predEffort}$ | ratio |
|---|---|---|---|
| Maxwell | 1849.84 | 7388.53 | 0.25 |
| Kitchenham | 1683.86 | 2458.87 | 0.68 |
| Cocomo81 | 68.87 | 349.12 | 0.20 |
| Nasa93 | 228.54 | 491.66 | 0.46 |
| Org1 | 2259.19 | 2119.31 | 1.07 |
| Org2 | 505.16 | 2091.64 | 0.24 |
| Org3 | 1690.67 | 1962.29 | 0.86 |
| Org4 | 4256.43 | 5449.54 | 0.78 |
| Org5 | 4498.61 | 6368.12 | 0.71 |
| Org6 | 2167.67 | 3787.19 | 0.57 |
| Org7 | 4571.36 | 6824.44 | 0.67 |

We formalize the validation process of PIs with $CL_{0.6827}$ and $CL_{0.9545}$ as follows:
Suppose $\{t_i\}_{i=1}^N$ are the actual efforts of testing projects, $\{\widehat{t_i}\}_{i=1}^N$ and $\{\widehat{\sigma^2}_i\}_{i=1}^N$ are the predicted means and variances. Set function $f_s$:

$$f_s(t_i) = \begin{cases} 1 & \text{for} \quad |t_i - \widehat{t_i}| < s * \widehat{\sigma}_i \\ 0 & otherwise \end{cases}$$

where $s \in \{1, 2\}$. Then define the ratio $\frac{\sum_{i=1}^N f_s(t_i)}{N}$ as our HitRate with $CL_{0.6827}$ and $CL_{0.9545}$. If $\{HitRate_s\}_{s\in\{1,2\}}$ satisfies the criteria: $HitRate_1 \geq 68.27\%$ or $HitRate_2 \geq 95.45\%$, we conclude that the PIs of RVM pass the PI validation with $CL_{0.6827}$ or $CL_{0.9545}$.

Table 5 lists the HitRate with confidence levels of $CL_{0.6827}$ and $CL_{0.9545}$. As shown in the table, three (in yellow / light grey) PIs of $CL_{0.6827}$ and $CL_{0.9545}$ cannot pass the validation, i.e., it is overconfident to these estimated PIs of CLs, which commonly happens in effort estimated PIs. Nevertheless, in more cases, our estimated effort PIs are not narrow enough for $CL_{0.6827}$. This indicates that there exists improvement space for the PIs derived from RVM.

One possible reason may be that the Gaussian distribution cannot capture the uncertainty in the predicted effort perfectly. If so, it suggests that the Gaussian effort noise assumption may not perfectly match the context of SEE, because the predicted effort distribution is directly caused by the Gaussian noise assumption. Given that RVM was competitive against other approaches and that the derived PIs are, though not perfect, acceptable, we consider that the proposed PIs are of practical use. However, changing the effort noise assumption to a more adequate one in the context of SEE may improve both point estimations and the performance of PI further.

# 7. FURTHER STUDIES OF RVM IN SEE

This section proposes initial answers to P3 (estimating effort noise) and P4 (guidance on whether or not the required effort of a new project should be collected).

## 7.1 Effort Noise Estimation of SEE Data Sets

In the model construction of RVM for SEE, addictive noise inherent in the actual effort is assumed to be Gaussian with

zero mean, because Gaussian distribution often matches the actual distribution of noise in real-world processes reasonably well [1]. The reason for assuming zero mean of Gaussian is that we expect there exist no inherent bias in the effort noise. The variance of effort noise can be derived during the training process along with the unknown parameters of RVM model; i.e., in a data set, once the model of RVM is established, the variance of Gaussian noise within the efforts of this data set can be determined, and therefore the effort noise of this data set is obtained. We can further use the standard deviation (std) of the Gaussian effort noise as an indicator of the quality of this data set: larger std indicates a worse data quality because of a larger level of noise in the efforts, and smaller std suggests a better quality.

In each data set, we use 10 times 10-fold CV to obtain the point prediction and the performance of the derived PIs, and thus we obtain 100 (10 times 10) gauges for effort noise. Each gauge relates with a subset of the entire data set, i.e., 9/10 of the data set is used as the training data and the noise estimation is only related with this subset. For assessing the effort noise of the entire data set, we adopt the mean value of the 100 gauges, and report the mean of the effort noise across 100 subsets in table 6.

Since different data sets have different scales of quantity of the project effort values, the mere value of estimated effort noise cannot reflect the level of effort noise very well, and cannot, therefore, reflect the quality of data set in terms of effort noise very well. For instance, if the predicted efforts in data sets D1 and D2 are around 2000 and 7000 respectively, and the standard deviations of effort noise in D1 and D2 are around 1000, then it is reasonable to consider that D1 is suffering heavier effort noise than D2. Thus, we use the ratios of average effort noise std to the average predicted effort to measure the effort noise of a data set, indicating the level of effort noise per magnitude of effort. The 'average predicted effort' is computed based on the following: for each running of 10-fold CV, all projects will obtain their predicted efforts; after 10 runs we will have ten predictions for each project, and thus the average predicted effort can be computed by averaging ten effort estimations.

We use the average predicted effort rather than the average actual effort for the following reason: like effort noise, predicted efforts are derived from RVM; by calculating the ratio of the two items, the impact of the predictor (e.g., different parameter settings) may be eliminated and the real levels of effort noise may be revealed. By contrast, the actual effort is independent of the predictor, thus the ratio between effort noise and the actual effort is likely to be impacted by different parameter settings of RVM.

As shown in table 6, different data sets have different levels of effort noise. For instance, Maxwell, Cocoma81 and Org2 show lower levels of effort noise in comparison to Org1, Org3, Org4 and Org5 which are much higher.

To some extent, the levels of effort noise can indicate the quality of a data set: the data sets with lower level of effort noise are of higher quality compared with higher level of effort noise with respect to the quality of actual effort collection. This can further impact our choices of data sets in empirical experiments. For instance, it may be preferable to conduct SEE studies, such as the comparison of two non-robust approaches or the analysis of the importance of different SEE attributes, on data sets with low levels of effort noise, where there is less influence from misleading effort noise. Moreover, the quantification of effort noise can provide an extra priority of a data set when researchers aim to improve the quality of data sets.

It is worth noting that effort noise is not the only criteria to justify the quality of SEE data set. Other factors such as highly predictive SEE features, lower level of feature noise, etc., can also contribute to a higher quality. However, we consider the effort noise as an important measurement of quality, which can be dealt with by RVM easily.

## 7.2 Empirical Guidance for Data Collection

As emphasized in section 3, by using ARD, RVM can automatically choose 'relevant' projects from training samples, which can capture the major structure of the training space [39]. Based on this fact, we empirically identify three types of regions in SEE input attribute space, and further introduce three guidances on whether the effort of a new project should be collected given that RVM is applied.

There are three types of local regions: (1) regions with isolated relevance vector (RV); (2) regions with condensed RVs; and (3) regions with few RVs in comparison to the number of samples which are not relevance vectors.

For case (1): if only one isolated RV locates in a region of the input space, it indicates insufficient description for this region (only one) and this point cannot be represented by the relevance vectors in the nearby regions. Therefore, it is suggested to collect more data to further verify whether this case will change to case (2) or case (3).

For case (2): if many RVs squeeze in a region, it indicates a rapid change in this region, and a majority of data have to be RVs to capture the severe change. Therefore, it is suggested to collect more data in this region.

For case (3): if in a region that has had several data points discarded by the RVM, it indicates a gentle change in this region, and it is likely that the chosen RVs are capable of capturing the major structure of this region. Therefore, it is suggested that data collection is not necessary.

Based on the above description, our empirical guidances on whether the required effort should be collected for a new project are summarized as follows: given that RVM is used, if the practitioners find that the new project fell in a region of cases (1) or (2), it is suggested to collect the required effort of this project; in contrast, if a new project fell in a region of case (3), it is suggested not to collect its effort.

For practical use, we present a step-by-step procedure to use this empirical guideline to guide the collection of a new project: (i) Identify its $k$ nearest neighbours with the shortest distance denoted by $d_{min}$. Here, $k$ is a predefined integer depending on the size of data set, and we suggest larger $k$ for bigger data set. The distance can be Euclidean. (ii) $\overline{d}$ is a predefined, upper bound of the distance between this project and its nearest neighbor. $\overline{d}$ depends on the spread of the data set: if the data spread largely across the input space, it is suggested to set a larger $\overline{d}$; if otherwise, better to choose a smaller $\overline{d}$. Verify whether $d_{min} \geq \overline{d}$, if 'yes', we arrive at case (1), and it is suggested to collect this project; if 'no', continue to the third step. (iii) If the majority of the included training samples are chosen as RVs, then we fall into case (2), and it is suggested to collect this project; if only a few included training samples are chosen as RVs, then we reach case (3), and it is suggested not to collect this project; if the portion of RVs is around half, we should go through step (i) - (iii) with smaller/larger $k'$ depending on

the original value of $k$.

As we can see, these guidances are empirical because of the need to set the values $k$, $\overline{d}$ and $k'$. To better define their values, we suggest to use Multidimensional Scaling (MDS) [6] to reduce the input space to a lower dimension, i.e., 3-D or 2-D, and then draw a 3-D/2-D surface with all training samples and the new project. From this approximating visualization of input space for all projects, we can have a gross impression about the local region where this new project locates, and choose these predefined values accordingly.

It is noteworthy that there may be some situations beyond our provided guidance, and further investigation is needed to improve these guidelines. However, as an initial attempt and encouragement for future studies, we believe that, based on ARD, the proposed guidelines address the problem of whether to collect the required effort to a good extent. Our guidelines can also be used together with other general data quality guidelines for SEE.

The essential idea for data collection discussed in this section is easy and also suitable for other SEE predictors, but the (three) proposed rules are specific to RVM due to the usage of RVs rather than the entire training data to guide data collection. One benefit of using RVs instead of the entire data is that RVs can reflect rate of change of a small region; e.g., by analyzing RVs in a region, we can distinguish between cases (2) and (3). For instance, in a small region where there are many training data already, people might suggest not to collect the effort of a new project if they were analysing all training data instead of the RVs, because they may think that there are enough training data already in this region. However if the case is that all the training data in this region are chosen as RVs, this indicates that there are severe changes in this region and that it is better to collect more labelled data.

# 8. THREATS TO VALIDITY AND FUTURE WORK

One potential threat to validity is that effort noise is assumed to be Gaussian in line with the assumptions for noise in many practical situations. However, the real effort noise in SEE may not be normally distributed. Since we do not have access to the noise-free efforts, it is nearly impossible to directly validate whether or not the Gaussian assumption perfectly fits software effort noise issue, or to build up a more suitable model. One possible way to identify a better distribution involves the help from practitioners. For instance, if people tend to report more effort than they actually used, then the model of effort noise may be better assumed as a skewed distribution towards the positive rather than a symmetric Gaussian distribution. However, we do not consider it to be a very serious threat to this study, because RVM's performance as a point estimator was competitive against other approaches that have been doing well for SEE, and its PIs with $CL_{0.6827}$ and $CL_{0.9545}$ were adequate in the majority of the cases. They indicate that the Gaussian assumption can catch the major structure of the real effort noise to some extent, and that the real effort noise distribution is perhaps not too far away from Gaussian distribution. Further investigation of different assumptions of effort noise can be considered as a good research direction.

Another potential threat to validity is the performance measurement. Our analysis were based on MAE, because it is symmetric and unbiased, being recommended in the context of SEE [34]. However, this measure can be influenced by project size. As a future work, we will computer more performance measurements such as Logarithmic Standard Deviation [12] and MAE in the log-scale to further validate RVM based on measures independent of project size.

In our next step, we will also compare RVM with other effort predictions, including support vector regression (SVR), to further verify its good performance. The comparison between our prediction intervals and other approaches will also be conducted in the future work.

# 9. CONCLUSIONS

In this paper, we studied three challenges of software effort estimation problem: (1) uncertain prediction, (2) effort noise, and (3) collection of required effort for a new project. We found that relevance vector machine (RVM) has the potential to address these three challenges simultaneously.

First, when used as a point estimator, our systematic experiments show that RVM is very competitive compared with state-of-the-art effort approaches, being usually ranked top 2 in 7 across 11 data sets, and behaving statistically similar to SEE predictors that have been showing to perform well.

We then presented how to decide the prediction interval (PI) with a certain confidence level (CL) for a new project, and validated PIs under two specific cases with $CL_{0.6827}$ and $CL_{0.9545}$. We found that even though the PIs with $CL_{0.6827}$ and $CL_{0.9545}$ are not perfect in terms of (i) informativeness and (ii) passing the validation, in the majority of the data sets, they are still acceptable and of practical use. And we believe that better performance can be achieved by further investigation on the effort noise assumption.

Furthermore, we present our attempt to address challenges 2 and 3 by using RVM. In summary, by using RVM, we can analyze the quality of data in terms of the level of noise in required efforts based on the Gaussian effort noise assumption. By exploiting the *automatic relevance determination* (ARD) used by RVM, we identified three special cases and proposed guidelines on whether the required effort should be collected for a new project.

Based on the studies above, we showed that RVM is a very promising predictor for SEE and should be further investigated and exploited.

# 10. ACKNOWLEDGMENTS

Liyan Song would like to thank Dr.Yuan Shen for his valuable discussion and comments very much. This work was supported by EPSRC grant EP/J017515/1.

# 11. REFERENCES

[1] Engineering statistics, nist/sematech e-handbook of statistical methods. `http://www.itl.nist.gov/div898/handbook/`, 2012.

[2] L. Angelis and I. Stamelos. A simulation tool for efficient analogy based cost estimation. *Empirical Software Engineering*, 5(1):35–68, 2000.

[3] J. Armstrong. *"The forecasting dictionary," in J. S. Armstrong (Ed.), Principles of forecasting: a handbook for researchers and practitioners*. Kluwer, 2001.

[4] B. Baskeles, B. Turhan, and A. Bener. Software effort estimation using machine learning methods. In *Computer and information sciences, 22nd international symposium on*, pages 1–6, 2007.

[5] C. Bishop. *Pattern Recgonition and Machine Learning*. Springer, 2006.

[6] I. Borg and F. Groenen. *Modern Multidimensional Scaling - Theory and Applications*. Springer, 2005.

[7] L. P. Braga, A. Oliveira, G. Ribeiro, and S. Meira. Bagging predictors for estimation of software project effort. In *International Joint Conference on Neural Networks*, pages 1595–1600, Orlando, 2007.

[8] C. Briand, E. Emam, D. Surmann, I. Wieczorek, and D. Maxwell. An assessment and comparison of common software cost estimation modelling techniques. In *ICSE*, pages 313–322, New York, USA, 1999.

[9] S. Chulani, B. Boehm, and B. Steece. Bayesian analysis of empirical software engineering cost models. *IEEE TSE*, 25(4):573–583, 1999.

[10] K. Dejaeger, W. Verbeke, D. Martens, and B. Baesens. Data mining techniques for software effort estimation: A comparative study. *IEEE, TSE*, 38(2):375–397, 2012.

[11] A. Faul and M. Tipping. Analysis of sparse bayesian learning. In *Advances in Neural Information Processing Systems 14*, pages 383–389. MIT Press, 2001.

[12] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit. A simulation study of the model evaluation criterion mmre. *IEEE TSE*, 29:2003, 2003.

[13] M. Harman, F. Ferruci, and F. Sarro. Search-based software project management. In G. Ruhe and C. Wholin, editors, *Software Project Management in a Changing World*. Springer, 2014.

[14] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*. Springer, 2001.

[15] ISBSG. The international software benchmarking standards group. `http://www.isbsg.org`, 2011.

[16] M. Jorgensen. Evidence-based guidelines for assessment of software development cost uncertainty. *IEEE, TSE*, 32(11):942–954, 2005.

[17] M. Jorgensen and K. Sjoberg. An effort prediction interval approach based on the empirical distribution of previous estimation accuracy. *IST*, 45(3):123 – 136, 2003.

[18] M. Jorgensen, K. H. Teigen, and K. Molokken. Better sure than safe? overconfidence in judgement based software development effort prediction intervals. *Journal of systems and software*, 70:79–93, 2004.

[19] M. Klas, A. Trendowizc, Y. Ishigai, and H. Nakao. Handling estimation uncertainty with bootstrapping: empirical evaluation in the context of hybrid prediction methods. In *ESEM*, pages 245–254, 2011.

[20] E. Kocaguneli, B. Cukic, and H. Lu. Predicting more from less: Synergies of learning. In *RAISE*, San Francisco, CA, USA, 2013.

[21] E. Kocaguneli, B. Cukic, T. Menzies, and H. Lu. Building a second opinion: Learning cross-company data. In *PROMISE*, Baltimore, USA, 2013.

[22] E. Kocaguneli and T. Menzies. How to find relevant data for effort estimation. In *ESEM*, pages 321–324, 2011.

[23] E. Kocaguneli, T. Menzies, and J. Keung. On the value of ensemble effort estimation. *IEEE, TSE*, 38(6):1403–1416, 2012.

[24] E. Kocaguneli, T. Menzies, J. Keung, D. Cok, and R. Madachy. Active learning and effort estimation: Finding the essential content of software effort estimation data. *IEEE TSE*, 39(8), 2012.

[25] Y. Kultur, B. Turhan, and A. Bener. Ensemble of neural networks with associative memory (ENNA) for estimating software development costs. *Knowledge-Based Systems*, 22:395–402, 2009.

[26] E. Mendes and N. Mosley. Bayesian network models for web effort prediction: A comparative study. *IEEE TSE*, 34(6):723–737, 2008.

[27] L. Minku and X. Yao. Ensembles and locality: Insight on improving software effort estimation. *IST*, 55(8):1512–1528, 2012.

[28] L. Minku and X. Yao. Software effort estimation as a multiobjective learning problem. *TOSEM*, 22(4):1–32, 2013.

[29] L. Minku and X. Yao. How to make best use of cross-company data in software effort estimation? In *ICSE*, pages 446–456, 2014.

[30] R. M. Neal. *Bayesian Learning for Neural Networks*. Springer, 1996.

[31] C. Pendharkar, H. Subramanian, and A. Rodger. A probabilistic model for predicting software development effort. *IEEE TSE*, 31(7):615 – 624, 2005.

[32] J. Sayyad Shirabad and T. Menzies. The repository of software engineering databases. School of Information Technology and Engineering, University of Ottawa, Canada, 2005.

[33] P. Sentas, L. Angelis, I. Stamelos, and G. Bleris. Software productivity and effort prediction with ordinal regression. *IST*, 47(1):17 – 29, 2005.

[34] M. Shepperd and S. McDonell. Evaluating prediction systems in software project estimation. *IST*, 54:820–827, 2012.

[35] M. Shepperd and C. Schofield. Estimating software project effort using analogies. *IEEE, TSE*, 23(12):736 – 743, 1997.

[36] L. Song, L. Minku, and X. Yao. The impact of parameter tuning on software effort estimation using learning machines. In *PROMISE*, Baltimore, USA, 2013.

[37] I. Stamelos and L. Angelis. Managing uncertainty in project portfolio cost estimation. *IST*, 43:759–768, 2001.

[38] I. Stamelos, L. Angelis, P. Dimou, and E. Sakellaris. On the use of bayesian belief network for prediction of software productivity. *IST*, 45(1):51–60, 2003.

[39] M. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning*, 1:211–244, 2001.

[40] J. Wei, S. Li, Z. Lin, Y. Hu, and C. Huang. Systematic literature review of machine learning based software development effort estimation models. *IST*, 54(1):41 – 59, 2012.

[41] B. Wlodzimierz. *The Normal Distribution: Characterizations with Applications*. Springer-Verlag, 1995.