Clustering and Artificial Neural Network Ensembles Based Effort Estimation

Hamdy Ibrahim[1,2]
[1]Department of Electrical and Computer Engineering,
University of Calgary, Calgary, AB, Canada
[2]Department of Information Systems
Menoufia University, Shebin Elkom, Egypt
hibrahi@ucalgary.ca

Behrouz H. Far
Department of Electrical and Computer Engineering
University of Calgary
Calgary, AB, Canada
far@ucalgary.ca

***Abstract*—Accurate effort estimation of software development projects plays a key role in project success. However, it is still a challenge activity to researchers and practitioners because of the nature of software products and dynamics in software industry and development environment. Artificial neural network (ANN) is as an effective method and has been widely used in various areas of software engineering. This paper proposes a new effort estimation method based on clustering and ANN ensembles. The contribution of the paper is twofold. First, the impact of clustering projects on the estimation accuracy is investigated. Second, the impact of using ANN ensembles instead of a single ANN is also investigated. The proposed method includes three phases called pre-processing, k-means clustering, and ANN ensembles effort estimation. The method starts with exploring the historical projects dataset. Afterward, k-means is used to cluster the projects. Finally, the proposed method as well as two other estimation methods (i.e. a single ANN and expert-based) were applied to the created clusters and results were compared using MMRE and PRED measures. The simulation results show that the proposed method significantly outperforms the two other estimation methods.**

Keywords-component; Artifical Neural Network Ensembles; Clustering; K-means; Effort Estimation; Mean Magnitude Relative Error; Percentage of/Predictions.

I. INTRODUCTION

The estimation of software development resources is the process of predicting the amount of effort, schedule, and cost needed to develop or maintain a software system [1]. It plays an essential role in software project management. It can be used to (1) set the budget and schedule, (2) support build versus buy decision, (3) as a part of tradeoff analysis among cost, schedule, and scope triangle, (4) provide the cost part of cost-benefit analysis, and (5) support the risk analysis of software cost and schedule [2]. Inaccurate estimation leads to software project crisis in terms of delivery delays and cost over-runs although releasing software systems on time and within the budget is a critical need for organizations.

An analysis study of around 50,000 projects (i.e. 60% from USA, 25% from Europe, and 15% from the rest of the world) developed between 2003 and 2012 reports that around 39% (i.e. 19,500) of projects were delivered on time, within the planned budget, and meeting stakeholders' needs; 43% (i.e. 21,500) of them were late, over budget, and/or partially meeting stakeholders' needs; and 18% (i.e. 9,000) of projects totally failed (cancelled prior to completion or delivered and never used) [3].

Underestimating software development effort can lead to understaffing, schedule and cost overruns, and low quality of software systems therefore can have severe impact on business reputation, credibility with customers, competitiveness, and performance. Furthermore, overestimating software development effort can lead to poor allocation and ineffective use of resources (i.e. wasted resources) [4, 5]. Resource estimation in software development project is a challenge task and more difficult than resource estimation in other industries because (1) software requirements and development tasks are ambiguous and not granularly specified, (2) there is dependency between software requirements, (3) fast evolution of technologies, and (4) the information used in the estimation process is uncertain. Addressing these challenges requires the use of a systematic estimation process.

The major role of software resource estimation in the project success and the associated challenges motivates researchers to (1) study software resource estimation problem, (2) propose new estimation techniques or combine existing techniques to better estimate effort, schedule, and cost, and (3) conducting comparative studies of current software resource estimation techniques to determine the most accurate technique. There are a vast number of estimation techniques which was proposed in the literature and can be classified into two main categories [6]:

1) Expert based estimation techniques where experts with relevant experiences in the application domain and in similar projects are consulted to estimate effort, schedule, and cost. This category has the following disadvantages: (1) It lacks the use of analytical methods, (2) Estimates are prone to errors and very subjective particularly when there is a mismatch between expert experience and project characteristics, (3) Sometimes, it is hard if not impossible to find experts with the right experience and If they are found, the cost will be high.
2) Model-based estimation techniques which use some algorithms and models to analyze historical data of past

projects to estimate effort, schedule, and cost of the new projects. This category includes statistical regression based models, machine learning based models (e.g. artificial neural networks (ANN), decision trees, case-based reasoning, Bayesian networks, etc.). It has a number of advantages over expert-based techniques. For example, the estimates are more objective, the same technique with/without adaptation can be used in different projects, and historical data of enormous number of past projects can be used to improve the accuracy of the estimation technique.

For decades, many techniques have been used for estimating software project effort, project duration, and product size. These estimation techniques have been developed using informal models (e.g. expert-judgement based estimation techniques), or formal models such as statistical and machine learning techniques [7-11]. Many researchers used artificial neural networks (ANNs) to estimate the software project effort because of its learning capability and its ability to model complex problems which are characterized by the existence of nonlinear relationship between problem variables. Examples of ANN-based estimation techniques are [12-19]. Feed-forward multilayer perceptron neural networks with back-propagation have been widely used to estimate project effort [12-14]. A number of estimation techniques were developed using radial basis function (RBF) neural networks [15]. Few research attempts [20] investigated the use of clustering and ANN ensembles and their impact on improving the estimation accuracy.

There are numerous software estimation techniques but the accurate effort estimation is still a challenge for the researchers and practitioners particularly project managers. In this paper, clustering and ANN ensembles based estimation method is proposed. The proposed method has three main phases: (1) Pre-processing, (2) K-means clustering, and (3) ANN ensembles-based estimation. The proposed method was evaluated using three datasets [21] and its performance was compared to the performance of a single ANN and expert-judgement based estimation techniques. The simulation results show that the proposed method outperforms other estimation techniques. Specifically, clustering projects improves the estimation accuracy. In addition, Applying ANN ensembles to the clustered projects also significantly improve the estimation accuracy.

The rest of the paper is organized as follows. Section 2 presents the proposed estimation method. Section 3 discusses the two measures used to evaluate the proposed method. In section 4, the proposed method is evaluated as well as the simulation results are presented and discussed. Section 5 presents the conclusion and summarizes the future research directions.

II. The Proposed Effort Estimation Method

The main idea of the proposed technique is using

1. Clustering to categorize the projects according to their similarity. The created clusters will be used to train and validate the neural network component. Clustering is used to improve the reliability of effort estimation because only relevant and homogenous projects are used to estimate project effort.
2. Artificial Neural network (ANN) ensembles where multiple neural network models are created, trained, and validated to estimate the project effort. The estimated efforts are then combined and averaged to represent the predicted project effort. Neural network ensembles are used to improve the accuracy of effort estimation.

The conceptual diagram of the proposed method is shown in Figure 1. The following subsections discuss pre-processing, K-means, and ANN-based Effort Estimation phases.

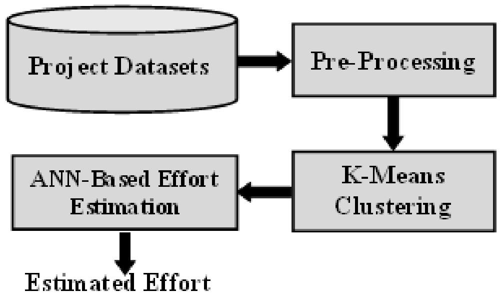


Figure 1. Conceptual Diagram of the Proposed Estimation Technique

A. *Pre-Processing*

In this phase, project datasets are explored to determine (1) number of projects in each dataset, (2) the key features (e.g., application domain) of the projects, (3) effort multipliers that are common between the projects, and (4) data types of the effort multipliers (e.g. numeric, discrete, etc.). In this phase, datasets are also searched for incomplete projects which have missing values for effort multipliers. According to the relative impact of the effort multipliers with missing values, a decision of including or excluding these projects is made. Conversion of discrete values to numeric values is also done to facilitate the combination of the project datasets for the purpose of validating the proposed method (see section IV-A).

B. *K-means Clustering*

K-means clustering technique is selected to cluster the projects because it is relatively scalable and efficient in processing large datasets. The projects are clustered as follows.

Input: A set of K-software development projects with r-features, C: number of clusters to be created where C<=K. C=K means each cluster will have only one project this is unrealistic.

Processing: K-means will organize the projects into C-clusters so that the similarity is high within the clusters and low between clusters. Assuming C=3 (i.e., 3 clusters will be formed), 3 projects each of which initially represents a cluster mean or center, will be randomly selected. Each of the remaining projects (K-3) is assigned to a cluster based on the distance between the project and the cluster mean. The project is basically assigned to the nearest (i.e., the lowest distance) cluster. Then, the mean value (i.e., center) for each cluster is recalculated according to the current projects in the cluster. The projects will be reassigned to the clusters according to the

updated cluster centers and based on which cluster center is the closest. The process will be repeated till no reassignment of projects in any cluster occurs.
Output: C clusters are created.

C. ANN Ensembles-based Effort Estimation

The ANN model used in the proposed method is shown in Figure 2.

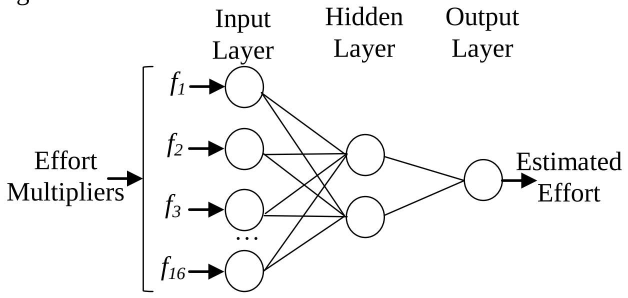


Figure 2. ANN Model

The input layer has 16 nodes each of which corresponds to an effort multiplier. The hidden layer has 2 nodes and the output layer has one node.

The ANN-based effort estimation algorithm shown in Figure 3 was developed around neural network ensembles concept. This is because the performance of neural networks and the accuracy of effort estimation can be impacted by how the data is split into training and validation datasets. Different splits of training and validation datasets produce different effort estimations. Therefore, 20 ANN models (i.e. ANN ensembles) with different training (80%) and validation (20%) splits were used to estimate the project effort. During the neural network training, weights are continuously adjusted till reaching their optimal values where the mean square error (MSE) is less than or equal to a threshold value (T_v). Datasets may contain numerous local minima. A widely used technique to overcome the problem of local minima is training ANN more than once starting with different random weights. The best ANN which has with the lowest MRE is selected because it represents most likely the global optimal values of ANN weights. In the proposed method, Each ANN ensemble is trained several times (e.g., up to 25 runs) using different random weights for the same dataset. Training is repeated until achieving an acceptable accuracy (e.g., MRE is less than or equal 10%) or reaching the maximum number of runs (e.g., 25 runs). If 25 runs are completed and MRE is still more than 10%, the project effort estimated by the ANN having the lowest MRE will be recorded. The 20 estimated efforts were combined using a weighted average technique as follows:

1. A weight which ranges from 0 to 1 is assigned to each ANN ensemble depending on its magnitude relative error (MRE) which measures the difference between actual and estimated effort for a given project relative to the actual effort. The lower MRE is the higher weight is assigned to the ANN ensemble. The MRE and weight of a ANN ensemble is calculated as follows:

$$MRE_{i,j} = \frac{|A_j - E_{i,j}|}{A_j} \tag{1}$$

$$w_i = 1 - MRE_{i,j} \tag{2}$$

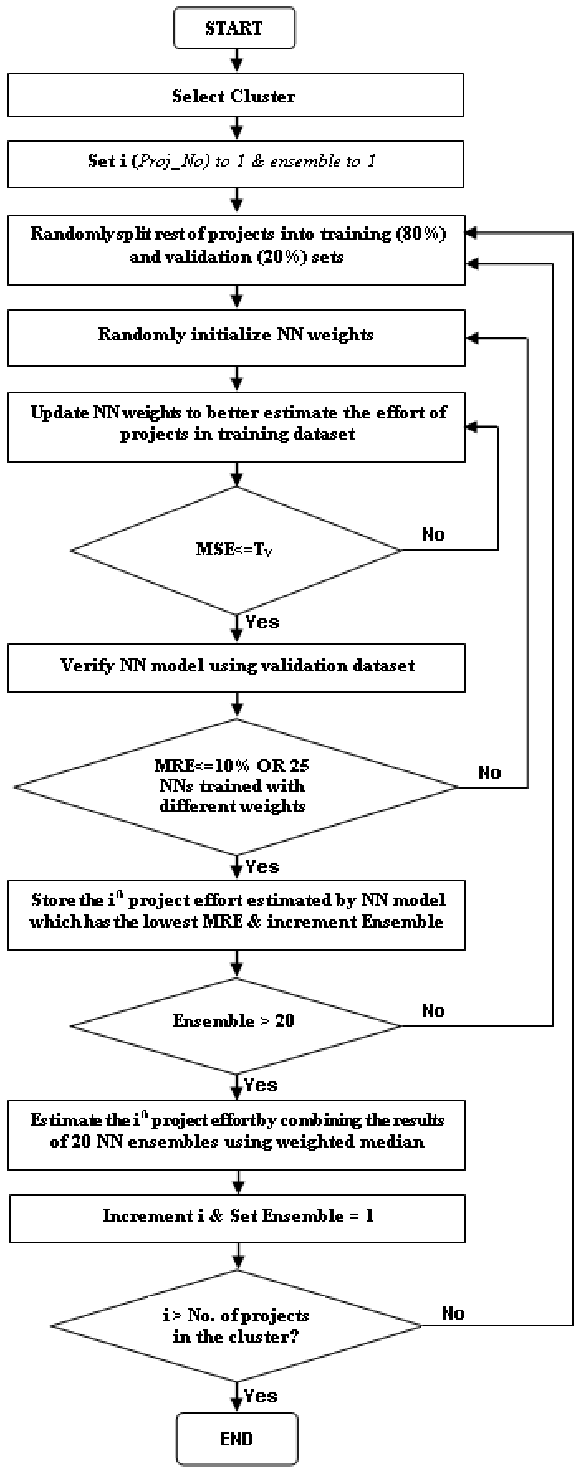

START
Select Cluster
Set i (Proj_No) to 1 & ensemble to 1
Randomly split rest of projects into training (80%) and validation (20%) sets
Randomly initialize NN weights
Update NN weights to better estimate the effort of projects in training dataset
MSE<=Tv
No
Yes
Verify NN model using validation dataset
MRE<=10% OR 25 NNs trained with different weights
No
Yes
Store the ith project effort estimated by NN model which has the lowest MRE & increment Ensemble
Ensemble > 20
No
Yes
Estimate the ith project effort by combining the results of 20 NN ensembles using weighted median
Increment i & Set Ensemble = 1
i > No. of projects in the cluster?
No
Yes
END

Figure 3. ANN Ensembles Estimation Process

Where:

$MRE_{i,j}$: Magnitude relative error of ANN ensemble i for project j.

A_j: Actual effort for project j.

$E_{i,j}$: Effort for project j estimated by ensemble i.

w_i: weight assigned to ANN ensemble i.

For example, a weight of 1 is assigned to the ANN ensemble whose MRE is 0 (i.e., the estimated project effort equals to the actual project effort).

2. The weighted median effort of a project j ($\overline{E_j}$) is calculated as follows:

$$\overline{E_j} = \frac{1}{N}\sum_{i=1}^{N} w_i E_{i,j} \qquad (3)$$

Where:
N: Number of ANN ensembles.
w_i: weight assigned to ANN ensemble i.
$E_{i,j}$: Estimated effort for project j by ANN ensemble i.

III. Perfomance Evaluation Measures

Two measures are used to evaluate the performance of the proposed estimation technique.

1. Mean Magnitude Relative Error (MMRE): It is the most widely used criterion for evaluating the performance of software estimation models. MMRE is calculated as follows:

$$MMRE_k = \frac{1}{k}\sum_{j=1}^{k} MRE_j \qquad (4)$$

Where:
K: number of projects for which the effort is estimated
MRE_j: Magnitude relative error for project j

Suppose that there are 2 estimation models with $MMRE_1$ and $MMRE_2$ respectively. If $MMRE_2$ is less than $MMRE_1$ then the performance of the second estimation model is higher than the performance of the first estimation model. Therefore, the lower the MMRE is the higher the performance of the estimation model is.

2. Percentage of Prediction (PRED): PRED(x) refers to the percentage of the projects for which MRE is less than or equal to x. PRED(x) is calculated as follows:

$$PRED(x) = \frac{S}{K} \qquad (5)$$

Where:
S: Number of projects with MRE less than or equal x
K: Total number of projects

The ideal value of x which is used in software estimation technique is 0.25. Therefore, PRED(0.25) is used to compare between the proposed estimation method and other estimation methods (i.e. single ANN and expert-based estimation).

IV. Evaluation and Results Discussion

In this section, the performance of the proposed estimation technique is evaluated and compared with (1) ANN-based technique and (2) Expert-based technique. The section is divided into three subsections as follows:

1- *Datasets* where the characteristics of the three datasets used in the method evaluation are summarized. Furthermore, the process of combining the three datasets into a large dataset is discussed.
2- *Clustering* where the combined datasets are clustered into 3 clusters using K-means clustering technique. Results of clustering are presented.
3- *Effort Estimation* where the effort is estimated using ANN ensembles, ANN, and expert methods for the three clusters and the combined datasets. MMRE and PRED(0.25) measures are used to compare between the three estimation methods.

A. *Datasets [21]*

Three public datasets from PROMISE Software Engineering Repository [21] are used to evaluate the performance of the proposed method. Characteristics of the datasets are summarized in Table 1.

TABLE I. Characteristics of the 3 Datasets

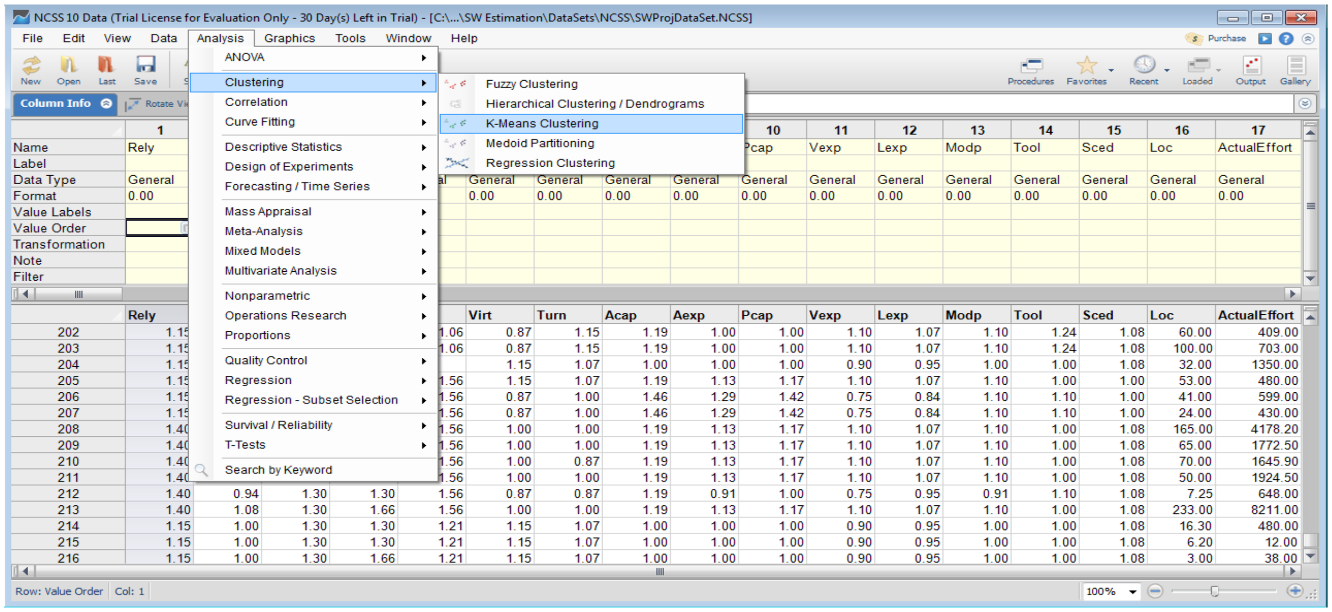
Dataset		**No. of Projects**		**No. of Effort Multipliers**		
COCOMO81		63		16		
COCOMO-NASA		60		16		
COCOMO-NASA2		93		16 + 7 General Attributes		
Effort Multiplier	**Description**	**MIN**	**MAX**	**MEAN**	**STD DEV**	**Variance**
Rely	Required software reliability	0.75	1.4	1.12	0.16	0.02
Data	data base size	0.94	1.16	1.01	0.08	0.01
CPLX	process complexity	0.7	1.65	1.16	0.16	0.02
Time	time constraint for CPU	1	1.66	1.14	0.17	0.03
STOR	main memory constraint	1	1.56	1.13	0.16	0.03
VIRT	machine volatility	0.87	1.3	0.95	0.11	0.01
Turn	turnaround time	0.87	1.15	0.96	0.09	0.01
ACAP	analysts capability	0.71	1.46	1.08	0.18	0.03
AEXP	Application Ex perience	0.82	1.29	1.06	0.12	0.02
PCAP	Programmers Capability	0.7	1.42	1.06	0.16	0.03
VEXP	virtual machine experience	0.75	1.21	1	0.08	0.01
LEXP	language experience	0.7	1.14	1.03	0.07	0.004
MODP	modern programing practices	0.82	1.24	1.03	0.11	0.01
TOOL	use of software tools	0.83	1.24	1.02	0.1	0.01
SCHED	schedule constraint	1	1.23	1.03	0.05	0.002
LOC	Line of code	0.9	1150	83.72	135.69	18413.03

The three datasets have 16 common effort multipliers. In COCOMO81 and COCOMO-NASA datasets, the 16 effort multipliers are numeric. However, in COCOMO-NASA2, the effort multipliers have discrete values ranging from very low to extra high. In pre-processing step, the numeric-to-discrete conversion table provided by [21] is used to convert discrete values to numeric. The numeric-to-discrete conversion is required to combine the three datasets. A total of 216 projects form the combined dataset. There are two types of correlation between the effort multipliers and the effort.

- *Positive correlation*: when the effort multiplier increases, the effort also increases and vice versa. For example by increasing the software reliability (Rely) or the complexity of the process (CPLX), the effort will increase.
- *Negative correlation*: when the effort multiplier increases, the effort decreases and vice versa. Increasing analyst's capability (ACAP), programmer's capability (PCAP), language experience (LEXP), or application experience (AEXP) decrease the effort.

B. *Clustering*

NCSS data analysis tool [22] is used to cluster the combined datasets into 3 clusters using K-means technique. Figure 4 shows a screenshot for NCSS tool. The summary of clusters created using K-means is shown in Table 2 and Table 3 shows the means of effort multipliers in the three clusters.


NCSS 10 Data (Trial License for Evaluation Only - 30 Day(s) Left in Trial) - [C:\...\SW Estimation\DataSets\NCSS\SWProjDataSet.NCSS]

	1	10	11	12	13	14	15	16	17
Name	Rely	Pcap	Vexp	Lexp	Modp	Tool	Sced	Loc	ActualEffort
Label									
Data Type	General	General	General	General	General	General	General	General	General
Format	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Value Labels									
Value Order									
Transformation									
Note									
Filter									

	Rely					Virt	Turn	Acap	Aexp	Pcap	Vexp	Lexp	Modp	Tool	Sced	Loc	ActualEffort
202	1.15				1.06	0.87	1.15	1.19	1.00	1.00	1.10	1.07	1.10	1.24	1.08	60.00	409.00
203	1.15				1.06	0.87	1.15	1.19	1.00	1.00	1.10	1.07	1.10	1.24	1.08	100.00	703.00
204	1.15					1.15	1.07	1.00	1.00	1.00	0.90	0.95	1.00	1.00	1.08	32.00	1350.00
205	1.15				1.56	1.15	1.07	1.19	1.13	1.17	1.10	1.07	1.10	1.00	1.00	53.00	480.00
206	1.15				1.56	0.87	1.00	1.46	1.29	1.42	0.75	0.84	1.10	1.10	1.00	41.00	599.00
207	1.15				1.56	0.87	1.00	1.46	1.29	1.42	0.75	0.84	1.10	1.10	1.00	24.00	430.00
208	1.40				1.56	1.00	1.00	1.19	1.13	1.17	1.10	1.07	1.10	1.00	1.08	165.00	4178.20
209	1.40				1.56	1.00	1.00	1.19	1.13	1.17	1.10	1.07	1.10	1.00	1.08	65.00	1772.50
210	1.40				1.56	1.00	0.87	1.19	1.13	1.17	1.10	1.07	1.10	1.00	1.08	70.00	1645.90
211	1.40				1.56	1.00	1.00	1.19	1.13	1.17	1.10	1.07	1.10	1.00	1.08	50.00	1924.50
212	1.40	0.94	1.30	1.30	1.56	0.87	0.87	1.19	0.91	1.00	0.75	0.95	0.91	1.10	1.08	7.25	648.00
213	1.40	1.08	1.30	1.66	1.56	1.00	1.00	1.19	1.13	1.17	1.10	1.07	1.10	1.00	1.08	233.00	8211.00
214	1.15	1.00	1.30	1.30	1.21	1.15	1.07	1.00	1.00	1.00	0.90	0.95	1.00	1.00	1.08	16.30	480.00
215	1.15	1.00	1.30	1.30	1.21	1.15	1.07	1.00	1.00	1.00	0.90	0.95	1.00	1.00	1.08	6.20	12.00
216	1.15	1.00	1.30	1.66	1.21	1.15	1.07	1.00	1.00	1.00	0.90	0.95	1.00	1.00	1.08	3.00	38.00

Figure 4. NCSS Data Analysis Tool

TABLE II. CLUSTERS SUMMARY

Cluster Number	Number of Projects	Percentage (%)
1	100	46%
2	80	37%
3	36	17%
Combined Dataset	216	100%

TABLE III. MEANS OF EFFORT MULTIPLIERS IN THE 3 CLUSTERS

Effort Multiplier	Cluster 1	Cluster 2	Cluster 3
Rely	1.14	1.05	1.19
Data	0.98	1.01	1.10
CPLX	1.19	1.11	1.23
Time	1.11	1.07	1.36
STOR	1.11	1.07	1.28
VIRT	0.97	0.94	0.93
Turn	0.94	0.95	1.05
ACAP	0.94	1.15	1.28
AEXP	0.97	1.14	1.12
PCAP	0.95	1.18	1.09
VEXP	1.01	1.00	0.96
LEXP	1.03	1.02	1.04
MODP	1.02	1.00	1.13
TOOL	1.03	0.96	1.15
SCHED	1.04	1.02	1.04
LOC	39.49	135.88	90.67

Table 4 presents the distance between the first 20 projects and the center of the three clusters. Distance 1 represents the distance between a project and the center of cluster 1. For example, (1) Project 1 is assigned to cluster 2 because the distance between project 1 and center of cluster 2, is the smallest distance, (2) Project 3 is assigned to cluster 1 because the distance between project 3 and center of cluster 1 is the smallest distance, and (3) Project 18 is assigned to cluster 3 because the distance between project 18 and center of cluster 3 is the smallest distance. Smallest distances for all projects are highlighted in grey.

TABLE IV. DISTANCE TO CENTER OF THE 3 CLUSTERS

Project No.	Cluster	Distance 1	Distance 2	Distance 3
1	2	5.92	5.14	5.64
2	2	4.91	4.56	5.55
3	1	4.63	4.89	6.63
4	2	6.28	5.26	6.39
5	1	3.90	4.31	5.53
6	2	6.10	5.05	5.65
7	2	4.12	3.73	6.41
8	1	6.47	7.81	6.94

9	1	3.12	4.56	5.25
10	1	4.30	5.88	5.53
11	1	4.30	5.88	5.53
12	1	2.56	4.51	5.60
13	1	3.71	5.29	6.63
14	1	6.80	8.41	7.10
15	1	6.10	7.11	7.60
16	1	3.85	5.26	5.04
17	1	4.02	5.75	5.50
18	3	4.42	4.92	3.70
19	2	8.74	8.36	9.55
20	1	4.90	6.04	5.77

Figure 5 shows a bivariate plot example which graphs the relationship between reliability (RELY) and process complexity (CPLX) effort multipliers. The bivariate plot helps to identify the degree and pattern relation between the two effort multipliers within different clusters. In cluster 1, increasing the process complexity reduces the software reliability and vice versa.

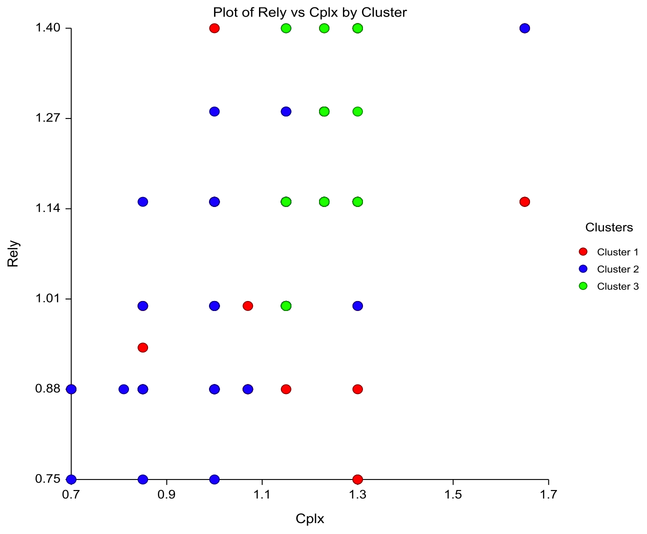


Figure 5. RELY vs. CPLX Bivariate Plot by Clusters

C. Comparison between ANN ensembles, ANN, and Expert based method

Before presenting and discussing the comparison results between the three estimation methods, we have noticed that the most influential effort multipliers vary from cluster to another (see Figures 6 and 7). For example In the combined datasets, the most influential effort multipliers are SCHED, ACAP, MODP, TOOL, LOC, VIRT, and TIME. However, the most influential effort multipliers in cluster 1 are LOC, VEXP, RELY, and AEXP. These multipliers have impact on project effort estimation more than the rest of 16 multipliers. Figure 8 shows the estimated effort versus the actual effort for 100 projects which belong to cluster 1.

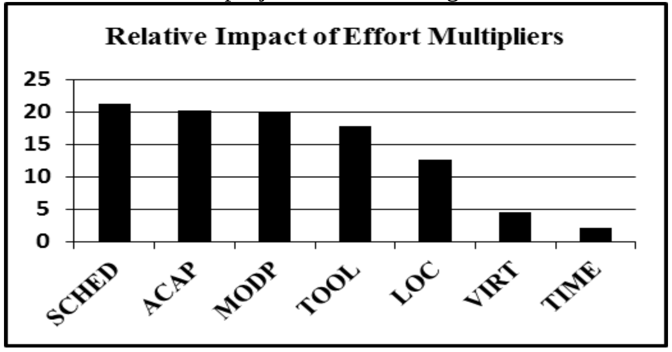


Figure 6. Relative Impact of Effort Multipliers in the Combined Datasets

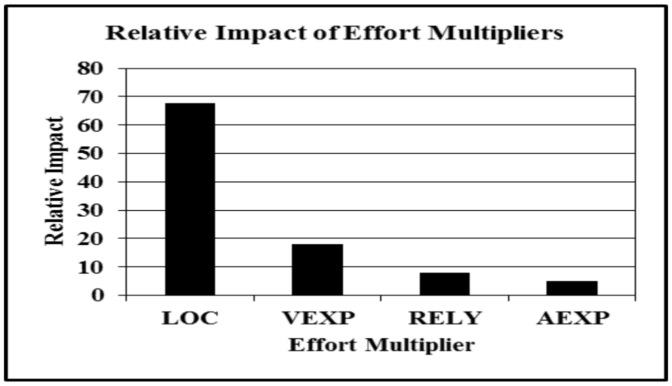


Figure 7. Relative Impact of Effort Multipliers in Cluster 1

The comparison between ANN ensembles, ANN, and expert estimation techniques using MMRE and PRED(0.25) measures is presented in Table 5.

The simulation results show that the proposed method performs better than using a single ANN and expert-based model.

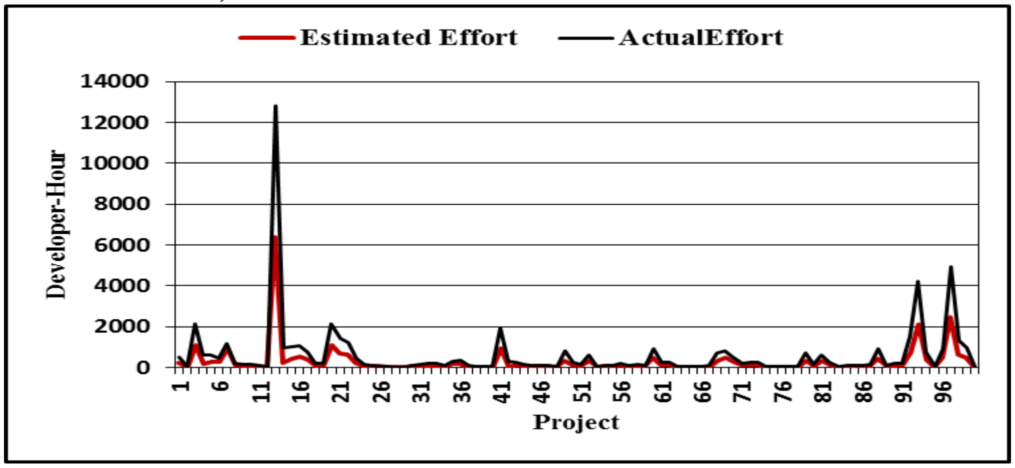


Figure 8. Estimated Effort versus Actual Effort in Cluster 1

TABLE V. MMRE AND PRED(0.25) SUMMARY

Dataset/ Estimation		ANN Ensembles	ANN	Expert
Combined Dataset	**MMRE**	**0.451**	**0.71**	**1.86**
	PRED(0.25)	**36.31%**	**27.2%**	**19.51%**
Cluster 1	**MMRE**	**0.115**	**0.322**	**1.05**
	PRED(0.25)	**82%**	**70.65%**	**21.32%**
Cluster 2	**MMRE**	**0.164**	**0.34**	**0.73**
	PRED(0.25)	**71.45%**	**69.1%**	**29.3%**
Cluster 3	**MMRE**	**0.35**	**0.403**	**0.30**
	PRED(0.25)	**55.25%**	**51.5%**	**67.22%**

The best results of the MMRE and PRED (0.25) measures are achieved by applying the ANN ensembles to projects in cluster 1. The values of MMRE and PRED(0.25) are 0.115 and 82% (82 projects out of 100 have MRE which is less than or equal 0.25) respectively. Overall, ANN ensembles perform better when it is applied to cluster 1. The worst results of the ANN ensembles and ANN was achieved when it is used to estimate the effort of projects belonging to the combined datasets. The estimation accuracy using ANN ensembles and ANN methods is decreased when it is trained and validated using heterogeneous projects or small number of projects. Expert-based model performs well when it is applied to a cluster with a small number of projects. Overall, clustering significantly improves the estimation accuracy and the performance of the three estimation methods. In addition, clustering the projects then using ANN ensembles to estimate the effort also enhances the estimation accuracy.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed an effort estimation method. The proposed method has been developed using clustering and artifical neural network (ANN) ensembles. It consists of three phases: pre-processing, k-means clustering, and ANN ensembles based effort estimation. The paper briefly discussed the three phases and the used estimation algorithm . NCSS data analysis tool was used to cluster the combined datasets into 3 clusters using k-means clustering method. The proposed ANN ensembles estimation algorithm, a single ANN and expert-based methods were applied to the combined datasets and the created three clusters. Two measures (i.e. MMRE and PRED(0.25) were used to evaluate performance of the proposed estimation method and compare it to the performance of ANN and expert based estimation methods. Overall, the simulation results show that:

1- The proposed estimation method outperforms the ANN and expert based estimation methods.

2- The best results for the proposed method was achieved when it is applied to cluster 1 because cluster 1 has homogenous and/or similar projects and the number of projects is fair enough to train and validate the ANN enesmbles.

3- The performance of the proposed method was low when it is applied to the combined datasets and cluster 3 because the combined datasets has a large number of hetrogenous projects and cluster 3 does not have enough projects to train and validate the ANN ensembles.

4- Clustering projects then using ANN ensembles, ANN, or expert-based estimation significantly improves the estimation accuracy.

Our future research will focus on:

5- Investigating the impact of using a subset of relevant effort multipliers on the accuracy and performance of the proposed estimation method.

6- Studying the impact of number of nodes in the hidden layer as well as number of hidden layers on the perfoamnce and accuracy of the proposed estimation method.

7- More experimentation using different datasets and comparing the results of the proposed estimation method with other estimation techniques proposed in the literature.

ACKNOWLEDGMENT

The authors would like to thank Electrical and Computer Engineering department at the University of Calgary, Alberta, Canada, for their logistics and financial support. The authors would like to thank Faculty of Computers and Information, Menoufia University, Egypt for their support.

REFERENCES

[1] Magne Jorgensen , "What We Do and Don't Know about Software Development Effort Estimation," IEEE Software, Vol. 31 (2), March-April 2014, pp. 37-40.

[2] Barry W. Boehm and Richard E. Fairley, "Software Estimation Perspectives," IEEE software, November-December 2000, pp. 22-26

[3] J. S. Chou and C. C. Wu, "Estimating software project effort for manufacturing firms," Computers in Industry, Vol. 64 (6), August 2013, pp. 732–740.

[4] Lionel C. Briand and Isabella Wieczorek, "Resource Estimation in Software Engineering," Encyclopedia of Software Engineering, 2002

[5] Dirk Basten and Ali Sunyaev, "Guidelines for Software Development Effort Estimation," IEEE Computer Society, Vol. 44 (10), October 2011, pp.88-90.

[6] Cuauhtémoc López-Martín, "Predictive Accuracy Comparison between Neural Networks and Statistical Regression for Development Effort of Software Projects," Journal of Applied Soft Computing, Vol. 27, Feburary 2015, pp. 434-449.

[7] Cuauhtémoc López-Martín and Alain Abran, "Neural Networks for Predicting the Duration of New Software Projects," Journal of Systems and Software Volume 101, March 2015, pp. 127-135.

[8] T. Halkjelsvik and M. Jørgensen, "From Origami to Software Development: A Review of Studies on Judgment-Based Predictions of Performance Time," Psychol Bull, Vol. 138 (2), 2012, pp. 238–271.

[9] C. López-Martín, C. and A. Abran, "Applying Expert Judgment to Improve an Individual's Ability to Predict Software Development Effort," International Journal of Software Eng. And Knowledge Eng. (IJSEKE), Vol. 22 (4), 2012, pp. 467–483.

[10] J. Wen, S. Li, Z. Lin, , Y. Hu, and C. Huang, "Systematic Literature Review of Machine Learning Based Software Development Effort

Estimation Models," Information Software. Technologies, Vol. 54 (1), 2012, pp. 41–59.

[11] M. Jørgensen, "A Review of Studies on Expert Estimation of Software Development Effort," Journal of Systems and Software, Vol. 70, 2004, pp. 37-60.

[12] J. Kaur, et al., "Neural Network-A Novel Technique for Software Effort Estimation," International Journal of Computer Theory and Engineering, Vol. 2, 2010, pp. 17-19.

[13] C. S. Reddy and K. Raju, "A concise Neural Network Model for Estimating Software Effort," International Journal of Recent Trends in Engineering, Vol. 1, 2009, pp. 188-193.

[14] I. Attarzadeh and S. H. Ow, "Software Development Cost and Time Forecasting Using a High Performance Artificial Neural Network Model," Intelligent Computing and Information Science, Vol. 134, 2011, pp. 18-26.

[15] A. Idri, et al., "Design of Radial Basis Function Neural Networks for Software Effort Estimation," International Journal of Computer Science, Vol. 7, 2010, pp. 11-16.

[16] K. V. Kumar, et al., "Software Development Cost Estimation using Wavelet Neural Networks," Journal of Systems and Software, Vol. 81, 2008, pp. 1853-1867.

[17] I. K. Balich and C. L. Martin, "Applying a Feedforward Neural Network for Predicting Software Development Effort of Short-Scale Projects," Software Engineering Research, Management and Applications (SERA), 2010, pp. 269-275.

[18] V. B. Khatibi and D. N. A. Jawawi, "Software Cost Estimation Methods: A Review," Journal of Emerging Trends in Computing and Information Sciences, Vol. 2, 2011, pp. 21-29.

[19] V. B. Khatibi, et al., "Neural Networks for Accurate Estimation of Software Metrics," International Journal of Advancement in Computing Technology, Vol. 3, 2011, pp. 54-66.

[20] Dinesh R. Pai, Kevin S. McFall, and Girish H, Subramanian, "Software Effort Estimation using a Neural Network Ensemble," The Journal of Computer Information Systems, Vol. 53 (4), July 2013,pp. 49-58.

[21] Datasets: http://promise.site.uottawa.ca/SERepository/datasets-page.html.

[22] NCSS Data Analysis Tool: http://www.ncss.com/.