Journal of Cases on Information Technology
Volume 24 • Issue 4

Software Effort Estimation Development From Neural Networks to Deep Learning Approaches

Poonam Rijwani, J.K. Lakshmipat University, India*
Sonal Jain, J.K. Lakshmipat University, India
https://orcid.org/0000-0002-5129-7161

ABSTRACT

Software engineering is a branch of computers that includes the development of structured software applications. Estimation is a significant measure of software engineering projects, and the skill to yield correct effort estimates influences vital economic processes, which include budgeting and bid tenders. But it is challenging to estimate at an initial stage of project development. Numerous conventional and machine learning-based methods are utilized for estimating effort, and still, it is a challenge to achieve consistency in precise predictions. In this research exploration, various ANN-based models are compared with conventional algorithmic methods. The study also presents the comparison of results on various datasets from the artificial neural network models, deep learning models, higher-order neural network models, leading to the conclusion that hybrid methods yield better results. This paper also includes an analysis of primary data collected from software project professionals using the questionnaire method involving questions related to software cost estimation.

KEYWORDS

Artificial Neural Networks, COCOMO, Deep Learning Neural Networks, Effort Estimation, Function Point, Higher-Order NN, Machine Learning, Software Projects

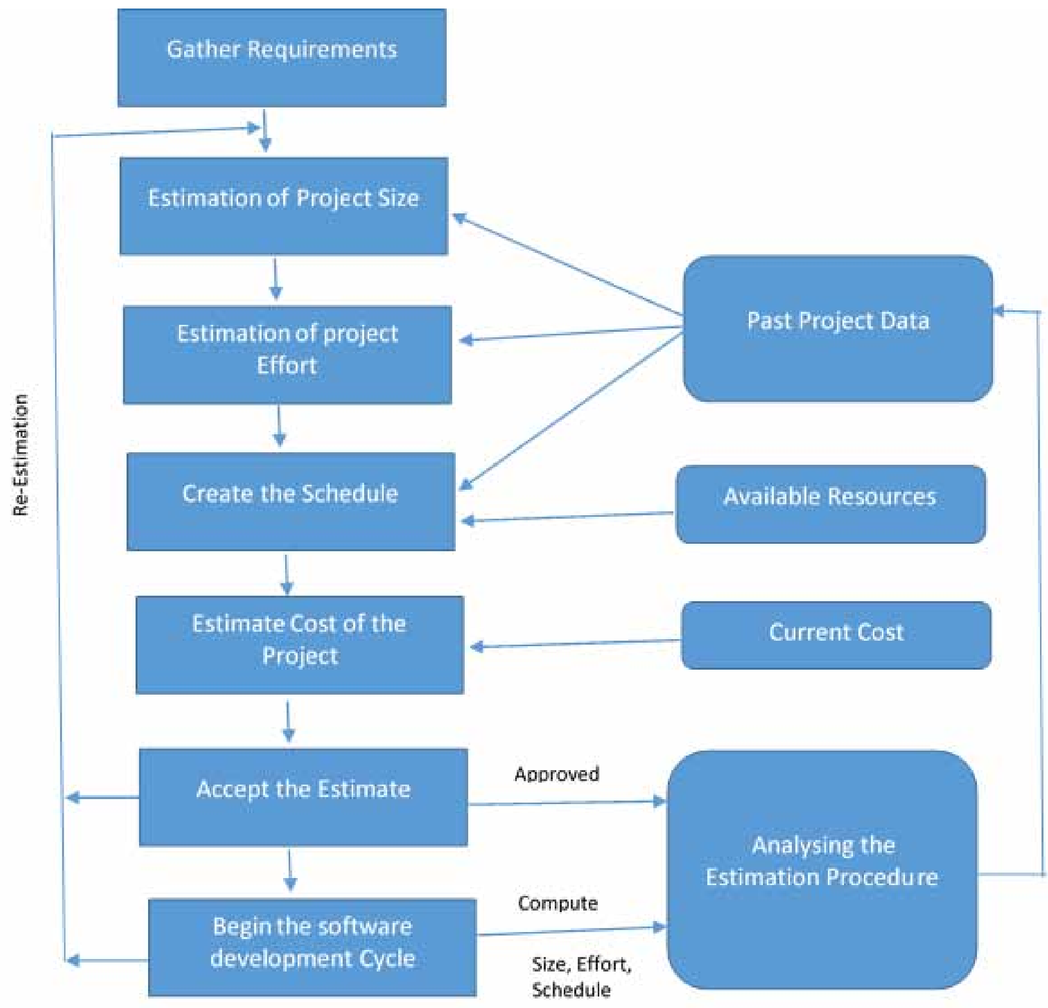
1. INTRODUCTION

The necessity for software development is growing progressively, causing continuous development in software projects and this progress has augmented the competition amongst corporations to yield extraordinary quality products at low-cost in less time. For the successful development of a software project, Prediction of cost and effort is important and has to be done at the primary phase of the software development procedure. Making software estimates encompasses effort, size, time, cost, and staff. We can easily make from Figure 1 that for estimating effort and schedule, comprehensive steps need to be completed with adhered guidelines.

Precise software estimate empowers the project manager towards effectively planning the project and allot resources efficiently. Under-estimating a project contributes to under-employment (the consequence is burnout of staff), under-scale quality control measures (the possibility of reduced

DOI: 10.4018/JCIT.296715
*Corresponding Author

Figure 1. Simple procedure of software development estimation



quality supplies)), and set a small schedule (causing destruction of trustworthiness as targets remain unfulfilled). Overestimating a project is equally bad for the business. If more resources are provided than needed in real for a project without adequate scope control, it will consume them. The project is thus possibly taking longer than expected to execute (which leads to loss of chance) as well as prolong the usage of capital for the next project. The calculation of software costs is the mechanism by which the practical effort to create and maintain software is estimated based on incomplete, unpredictable, and noisy data.

Numerous models and simulations for software effort estimation have been suggested by software researchers. Accordingly, these models can be classified based on their basic formulation patterns: analogy based estimation (Chiu & Huang, 2007; Shepperd & Kadoda, 2000), expert-judgment method (Jørgensen & Sjøberg, 2004), and algorithmic models containing empirical methods like Function points, constructive cost model(COCOMO) (Kaczmarek & Kucharski, 2004). Still, the failure rate of software projects is very high. One of the major causes behind the non-success of software projects is inefficient cost and effort estimation which leads to overall poor project management. The restrictions in algorithmic models have headed to the study of models that are non-algorithmic and are centered on the concept of soft computing. Machine learning was found to be suitable as it provides improved accuracy due to their predictive ability by leading repetitive training sequences and handle problems with efficacy. The often utilized soft computing practices in software development estimation are logistic regression, linear regression, decision trees, Artificial Neural Network, K Nearest Neighbor algorithm, Naive Bayes, classification, Support Vector M, deep learning, etc. The methods provide different precision on varied datasets. Consequently, it is difficult to determine the finest method with the necessary accuracy. Subsequently, a lot of investigation is going on Estimating software effort exhausting machine learning approaches that have shown their importance in estimating the effort.

This paper is an organized literature review going on for software development effort estimation exhausting diverse artificial neural networks, deep learning, and higher-order neural networks. A thorough review is conducted for various techniques and datasets utilized for performing the experiments, various performance assessment measures used for comparing the performance and findings, and conclusions from those studies. The model studies include the FFNN (feed-forward neural network) with backpropagation, RBFN (Radial basis function neural network), FLNN (Functional link neural network), GRNN (General regression neural network), LSTM (Long-short term memory), WNN (Wavelet neural network), artificial recurrent neural network, and ENN (Elman neural network). Additionally, a survey process using questions implemented concerning effort estimation of software to further investigate the study. The continuing article is systematized in 4 sections: The organized literature analysis and application of various neural network techniques, higher-order Neural Network, deep- learning Neural Network are shown in Section 2. Further, investigation and analysis are conducted in Section 3 of the paper, and to finish, the conclusion is presented in Section 4.

2. SYSTEMATIC LITERATURE SURVEY AND REVIEW PROCESS

The very first step in the systematic review process is data extraction. In this, data is extracted from data sources and then analyzed from the information stored in the pre-recorded datasets. Then data synthesis is performed to summarise the data and arranging it specifying relationship among data. This paper emphases mainly artificial neural network techniques utilized for software development estimation. Here, we will primarily find out the most common ANN technique and dataset used in predicting Software development Efforts widely. Also, we will look at various performance evaluation measures used by various researchers. As an amalgamation of techniques is also perceived to be implemented by researchers, the prevalent ones are also surveyed here. After doing an extensive survey on various techniques and datasets in the last decade, a future technique that can be used for predicting effort can be suggested. We have selected the literature from the last decade as extensive work is conducted using machine learning and hybrid techniques.

An Artificial Neural Network (‘Universal Approximator’) is a powerful and most prominent computational tool that draws its inspiration from the architecture and processing abilities of biological neurons, for instance, the human brain. Similar to the biological brain ANN is a network of simple processing components (nodes) that operate on local data and communicate with other elements. Each node in the network accepts the input signal, processes it, and directs output to other nodes. Each node must be linked to at least one node and the degree of importance of each connection (synapse) is evaluated by weight coefficient which is a real number. Neural networks are called universal approximators as they can accomplish the mapping of one vector space into another vector space. Every layer has interrelated neurons through associated weights and the activation functions. Artificial Neural Network learns from its former inputs, tuning themselves, and does the job.

Various types of Artificial Neural networks have been tried, implemented, and tested for refining the precision of the estimated effort for software development. Applications of various prime neural network techniques in software development effort estimation from the last decade are specified in Table 1.

Table 1. Neural Network techniques applied for Software effort estimation

S.No.	Author	Role	Machine learning Technique used	Datasets	Performance valuation Parameter
1.	(Jodpimai et al., 2010)	By integrating mathematical principle in Artificial Neural Network, software effort estimation is enhanced	Feed-Forward Neural Network	Desharnais, COCOMO81, NASA 93, NASA60, CF, Albrecht,	MRE, MMRE, PRED
2.	(Kalichanin-Balich & Lopez-Martin, 2010)	Two groups of software development projects are matched by statistical regression and feed-forward neural network	statistical regression, feed-forward neural network	132 Projects Data	The magnitude of Relative Error to Estimate (MER), Mean MER (MMER)
3.	(Attarzadeh & Siew Hock Ow, 2010)	Suggested COCOMO II grounded on Artificial Neural Network as compared to COCOMO	Back-propagation Neural Network	Artificial dataset, COCOMO	Mean Magnitude of relative error (MMRE), Magnitude of relative error (MRE), PRED
4.	(Reddy & Raju, 2010)	Suggested Artificial neural network based on Radial Basis and General Regression	Radial basis and General Regression NN	COCOMO’81	Mean Absolute relative error (MARE), Variance Absolute Relative Error (VARE), Prediction(PRED), Balance Relative Error (BRE), Mean Magnitude of Relative Error (MMRE)
5.	(López-Martín et al., 2011)	Done comparison of results from General Regression Neural Network and statistical regression	Generalized Regression Neural Networks	156 Project Data	The mean magnitude of error relative (MMER), mean error relative (MER)
6.	(Lopez-Martin et al., 2012)	Estimation of effort using Generalised Regression Neural Network is better or equal achieved by statistical regression	Generalized Regression Neural Networks	ISBSG dataset	The mean magnitude of error relative (MMER), mean error relative (MER)
7.	(Attarzadeh et al., 2012)	Suggested an ANN–COCOMO II and competed with the results of COCOMO II	Back-propagation Neural Network	COCOMO81, NASA 93	The magnitude of relative error (MRE), Mean MRE(MMRE), prediction(n)PRED
8.	(Babu et al., 2014)	Proposed a Two-Fold approach dependent on ANN	Back-propagation Neural Network	NASA dataset	The magnitude of relative error (MRE), Mean MRE(MMRE)
9.	(Sarno et al., 2015)	Utilized Gaussian Membership Function to restructure the important Effort Multipliers of COCOMOII	Fuzzy logic to represent Effort multipliers and feed-forward Neural network for improving accuracy.	NASA dataset	The magnitude of relative error (MRE)

continued on following page

Table 1. Continued

S.No.	Author	Role	Machine learning Technique used	Datasets	Performance valuation Parameter
10.	(Azzeh & Nassif, n.d.)	The effort is calculated exhausting the hybrid model with Support Vector Machine and radial basis neural networks and compared with Use Case Points prediction Model	Radial basis neural networks	Industrial Projects: 45 Educational projects:65	Absolute Error(AE), Mean Absolute Error (MAE), Mean Relative Error(MRE), Mean Balanced Relative error (MBRE), Mean Inverted Balanced Relative Error (MIBR)
11.	(Saraç & Duru, 2013)	Combining COCOMO used Artificial Neural Networks with K-Means and compared with COCOMO and ANN	BPNN with K-Means	COCOMO	The magnitude of relative error (MRE), Mean MRE(MMRE)
12.	(Parasana Sankara Rao & Kumar, 2015)	Suggested GRNN and contrasted with RBF Kernel, SMO Poly- kernel, M5, Linear Regression,	Generalized Regression Neural Network	COCOMO dataset	Mean Magnitude Relative Error (MMRE) and Median Magnitude Relative Error (MdMRE)
13.	(de A. Araújo et al., 2017)	Proposed multilayer dilation-erosion-linear perceptron (MDELP) and compared with various techniques exhausting many datasets	Hybrid multilayer perceptrons	Desharnais, Albrecht, COCOMO, Kemerer, Kotengray, NASA	Mean Magnitude Relative Error (MMRE), Pred(25)
14.	(Kumar & Singh, 2020)	Linear Regression (LR), Multi-layer perceptron (MLP), and Random Forest (RF) algorithms with the 12 elements presented that LR computes greater estimation outcomes than other ML techniques.	LR, MLP, and RF	NASA, Desharnais	Correlation Coefficients, Mean Absolute Error(MAE), **Root Mean Squared Error on Prediction** (RMSE), Relative Root Absolute Error (RRAE), and Relative Squared Error (RSE)
15.	(López-Martín et al., 2011)	Proposes a hybrid WNN(Wavelet neural network) with a metaheuristic algorithm for estimating software development effort	WNN, Firefly algorithm, Bat algorithm	COCOMO, NASA93, Maxwell, China	Mean Magnitude of Relative Error (MMRE), Pred(25), Median Magnitude Relative Error (MdMRE),
16.	(Edinson & Muthuraj, 2018)	The effort is considered using ANFIS, FCM, SC, and contrasted with Elman neural network.	ELMAN, Adaptive Neuro-Fuzzy Inference System(ANFIS) based Fuzzy C Means clustering, Subtractive Clustering	COCOMO, Desharnais, Maxwell, and IKH(IBM+Kemerer+ Hallmark)	The magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE), PRED(25), Root Mean Squared Error(RMSE)

continued on following page

Table 1. Continued

S.No.	Author	Role	Machine learning Technique used	Datasets	Performance valuation Parameter
17.	(Qin et al., 2019)	Proposed Deep Neural network (DNN) with the usage of function point estimation	BiLSTM-CRF structure, function-point analysis	52 project data	
18.	(Kaushik et al., 2020)	RBFN, FLANN with Whale Optimization Algorithm (WOA)	ANN simulations achieve well after joined using the metaheuristic technique	Zia dataset, Company Dataset 1 (CD1), and Company Dataset2 (CD2)	MMRE, MdMRE, PRED(0.25)
19.	(López-Martín, 2015)	Is Multilayer perceptron *(MLP), with a radial basis function neural network (RBFNN) statistically improved than that achieved by multiple linear regression (MLR)*	MLP- New projects 15 neurons in one hidden layer, Enhanced projects 35 neurons in 2 hidden layer (best accuracy) RBFN- Spread 1.1, 5 for new and enhanced projects GRNN- Spread 0.15, 4 for new and enhanced projects	Projects from ISBSG release 11	RMSE, MMRE, MMER, MBRE, MIBRE RBFN gives added precise results after Functional size is used as an independent variable.
20.	(Resmi & Vijayalakshmi, 2019)	Suggested Output layer self-connection recurrent neural networks (OLSRNN) with kernel fuzzy c-means clustering (KFCM) to improve software cost estimation	OLSRNN(Output layer self-connection recurrent neural networks with KFCM (kernel fuzzy c-means clustering)	5 openly available datasets	MMRE, MdMRE, PRED(25)
21.	(Benala et al., 2012)	Suggested UKW/ DBSCAN and FLANN and compared with SVR, RBF, and CART	FLANN, UKW/ DBSCAN	COCOMO’81, NASA’93, Desharnais	MMRE, MdMRE, PRED
22.	(Kaushik et al., 2016)	Author Proposed FLANN using intuitionistic fuzzy c-means clustering	IFCM–FLANN	COCOMO’81, NASA’93, Maxwell, China Dataset	MMRE, MdMRE, PRED
23.	(Venkataiah et al., 2019)	Proposed spiking neural network to improve cost estimation	Spiking neural network	IBM, ISBSG, CHINA	MMRE, RMSE
24.	(Wani & Quadri, 2016)	Suggested FLANN trained with Artificial Bee Colony for effort estimation	ABC-based FLANN model	NASA’93, COCOMO’81 and COCOMO_SDR	MRE, MMRE, and MdMRE
25.	(Goyal & Bhatia, 2020)	Proposes a non-linear prototype put up on MLP design for effort estimation	PRED_MLA, PRED_MLP_FS	Desharnais Project	MRE, MMRE

continued on following page

Table 1. Continued

S.No.	Author	Role	Machine learning Technique used	Datasets	Performance valuation Parameter
26.	(P Sankara Rao et al., 2017)	Multilayer Perceptron Neural Network (MLPNN) is used with a hybrid process which is an amalgamation of Artificial Bee Colony (ABC) and Local search processes are proposed to improve software effort estimation.	MLPNN	Constructive Cost Model (COCOMO) dataset	Mean Magnitude Relative Error (MMRE) and Median Magnitude Relative Error (MdMRE)

3. RESEARCH ANALYSIS AND EXPLORATION

In various studies that have been conducted, it can be seen that nearly 20 datasets have been utilized for investigating the estimation of effort. From the papers that we have studied from the last decade, the COCOMO dataset is the most experimented in 42 investigations. The Second most utilized one is the NASA dataset used in 24 different research studies. Desharnais, ISBSG dataset, and Maxwell's dataset are also utilized in 12 studies, trailed by Random and combination dataset used in 5 studies. Subsequent, are the IBMDPS, Albrecht, and China dataset used in 5 studies, and the dataset from Kemerer is utilized in 3 research studies. Finally, CF dataset, Kotengray dataset, artificial dataset, Canadian function dataset, Tukutuku dataset, SAMOA, IKH dataset, UMIS, QUES, and Zia, individually utilized rarely i.e. in one study. The effort output of the datasets like Maxwell, Desharnais, and China are calculated in the unit of hours and that of the Albrecht dataset is in the unit of months. Details about this Datasets repository, its attributes are mentioned in Table 2. Desharnais dataset encompasses 81 records and when inconsequential attributes were removed, it comprises 12 attributes. The dataset from Maxwell contains 62 project data and 27 features. China dataset contributes 16 attributes in 499 records. Albrecht dataset entails 24 records and 8 attributes.

Table 2. Description of Datasets utilized during 2010-2020

Dataset	Repository from where it is taken	Number of Records	Number of Attributes	Output (Effort)
Desharnais	GITHUB	81	12	Person-hours
China	Promise	499	16	Person-hours
Albrecht	Promise	24	8	Person-months
NASA	Promise	63	15	Person-months
Kemerer	zenodo	15	6	Man-months
ISGSG	ISBSG	2321	16	Man-hours
Tukutuku		53	9	Person-months
Maxwell	Promise	62	27	Person-hours
COCOMO	Promise	97	17	Person-Months

It is observed that the functioning of different intelligent methods to estimate the effort on diverse datasets differs when it is tested. Figure 2, 3, 4, 5, 6 showcases the performance of various neural network based methods concerning datasets utilized for effort estimation. As MMRE is the frequently used evaluation metric in most papers, the performance evolution of the approaches is presented in terms of MMRE. Using the COCOMO dataset, it is noted that Feed-forward neural network including a Multi-layer perceptron and back-propagation is the most used technique followed by the Radial Basis function neural network. Moreover, the linear regression technique is also used in many studies followed by General Regression Neural Network and the Fast Approximate nearest neighbour search algorithm is used in a few experiments. COCOMO model and Elman neural network are also tried but by very few researchers. Case Based Reasoning, Use-cases, WNN, ABC, and Spiking neural

Figure 2. Evaluation of the Suggested Methods on the COCOMO Dataset

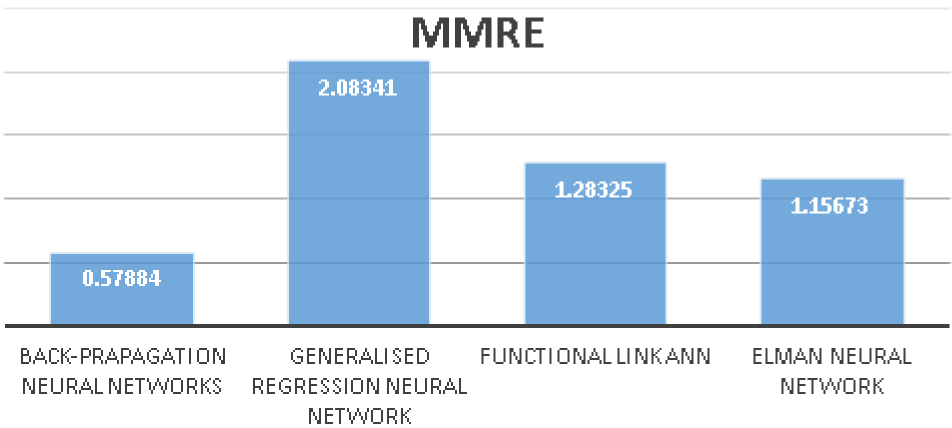


Figure 3. Evaluation of the suggested methods on the Albrecht Dataset

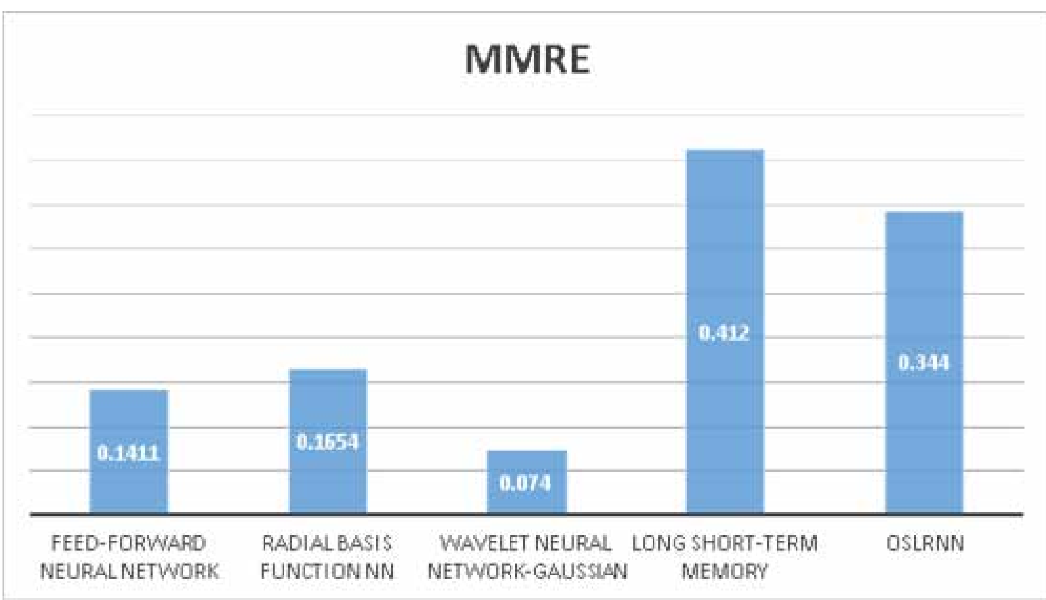


Figure 4. Evaluation of the Suggested Methods on the NASA Dataset

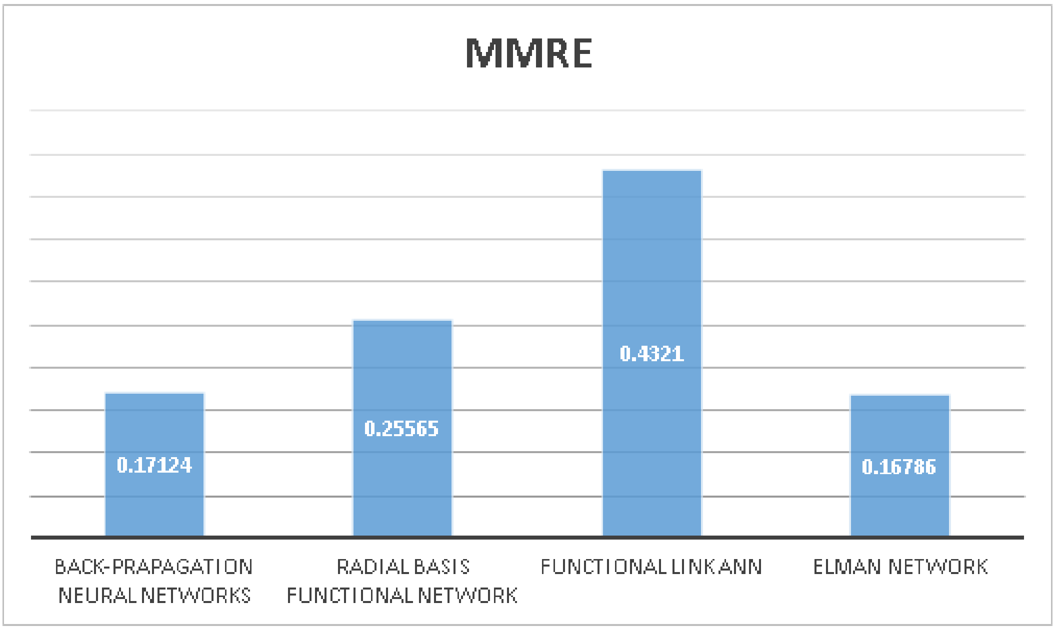
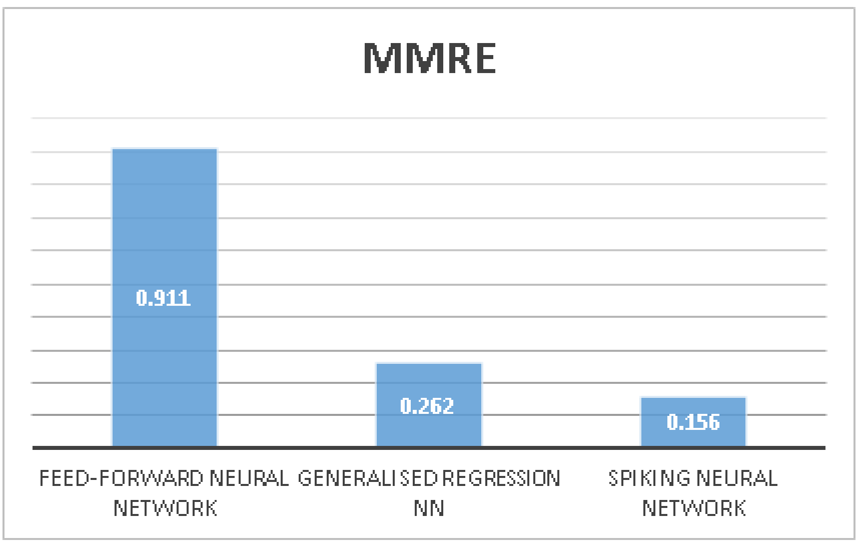


Figure 5. Evaluation of the Suggested Methods on the ISGSB Dataset



network practices are separately used in two studies. The methods for instance Rule induction, Quasi-optimal neural network, ANFIS, is used in a study. The results obtained post-application of intelligent techniques on different datasets are shown in figures 2,3,4,5,6.

From the analysis and extensive survey, it is evident that there is no one impeccable method that works on all the datasets, thus raising a need for an amalgamated approach. Further, a questionnaire was distributed with the intent of getting information from the experts so that efforts towards

Figure 6. Evaluation of the Suggested Methods on the Desharnais Dataset

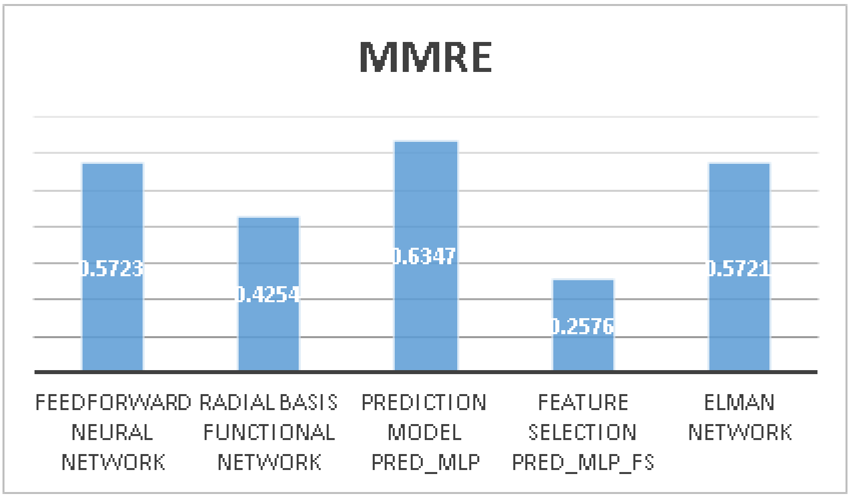
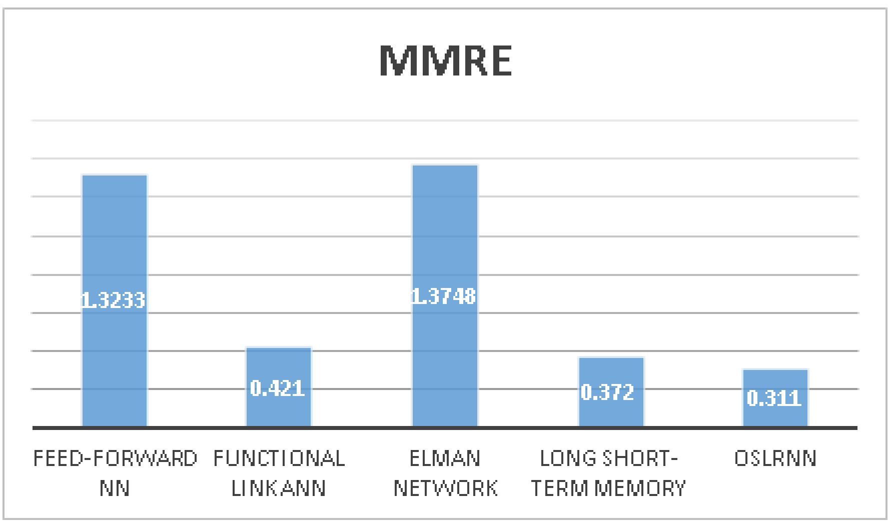


Figure 7. Evaluation of the Suggested Methods on the Maxwell Dataset

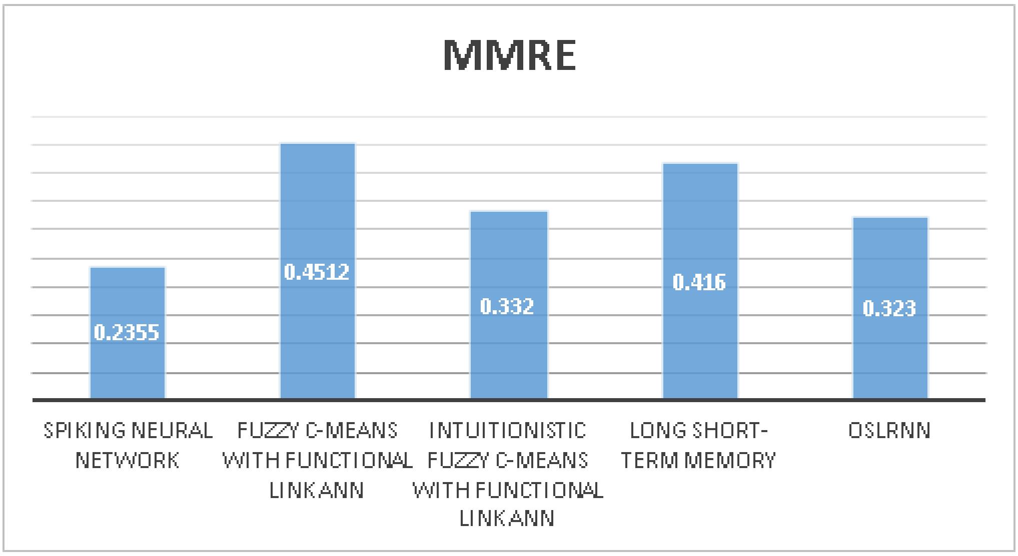


accomplishing a better approach can be prepared. Primary data will help the researchers to validate the works done on various models. Even if amalgamated approaches are considered to be worked on, with the availability of primary data, a sense of moving forward in increasing accuracy takes place.

To collect the Primary data, we investigated the method software industries are using to estimate the software cost and effort. In an attempt to collect data from companies, a questionnaire keeping in mind the following research questions (RQ) has been designed.

RQ1: How significant is the accuracy of estimation is for the software company?
RQ2: Up to what degree are cost estimate techniques applied in the software organizations?
RQ3: What are the purposes of estimating effort in the software organization?
RQ4: Are different software size estimation models utilized in the software company?

Figure 8. Evaluation of the Suggested Methods on the China Dataset



RQ5: What are the reasons for imprecise estimations?
RQ6: Are there any obstructions and complications in the use of software effort estimation approaches?

To get the information about the kind of cost and effort estimation the software organization is using, a survey was conducted. The response was collected telephonically and via e-mail. Software professionals have participated in the process of research.

3.1 Questionnaire Design

Based on the research aim, the questionnaire was distributed into three parts. In the first part of the questionnaire, Common information about the software organizations and the respondents were attained. In the second section of the form, questions were designed so that to acquire the material about the kind of software cost estimation practices used in the software firms. Finally, in the third part of the questionnaire, questions were kept about the difficulties and barriers of the various software cost estimation practices used by the organization. Questions designed in the form are as shown in Table 3.

3.2 Summarized Survey Response

RQ1: How significant is the accuracy of estimation is for the software company?

Almost all software companies consider the accuracy of software estimation is extremely important (70%) or very important (30%).

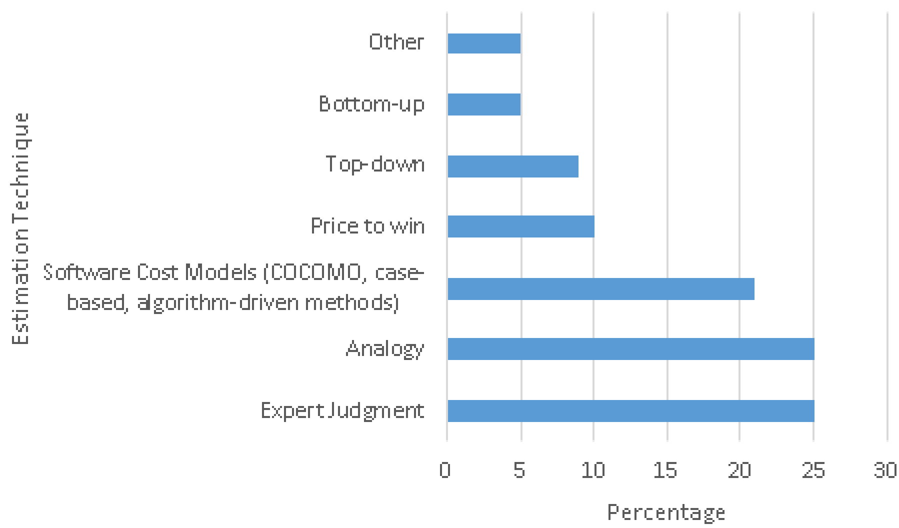
RQ2: Up to what degree are cost estimation techniques are applied in the software organization?

Mainly, the widely used estimation technique is the expert judgment and analogy method while the software cost model is ranked second. Models are mainly applied by large companies as shown in Figure 9.

Table 3. Questions in the form

Section 1: General Information 1. Organization Name: Organizational Size (Total Number of employees): (Number of Software developers): 2. Respondent Designation Working Experience (in years) of project management:
Section 2: Software Effort Estimation Practices 3. Do organizations follow a particular software effort estimation procedure? 4. How significant is the software estimation precision observed? 5. What are the effort estimation procedures utilized in the organization? 6. What drives the cost estimation? 7. On what criteria a cost estimation process is designated? 8. Does your organization utilize any software size estimation integration for effort estimation?
Section 3: Difficulties and obstacles 9. What do you think are the main reasons for imprecise estimation? 10. State if any obstructions and complications occurred in the use of software effort estimation approaches utilized in the organization?

Figure 9. The Cost Estimation method used in the Software Organization

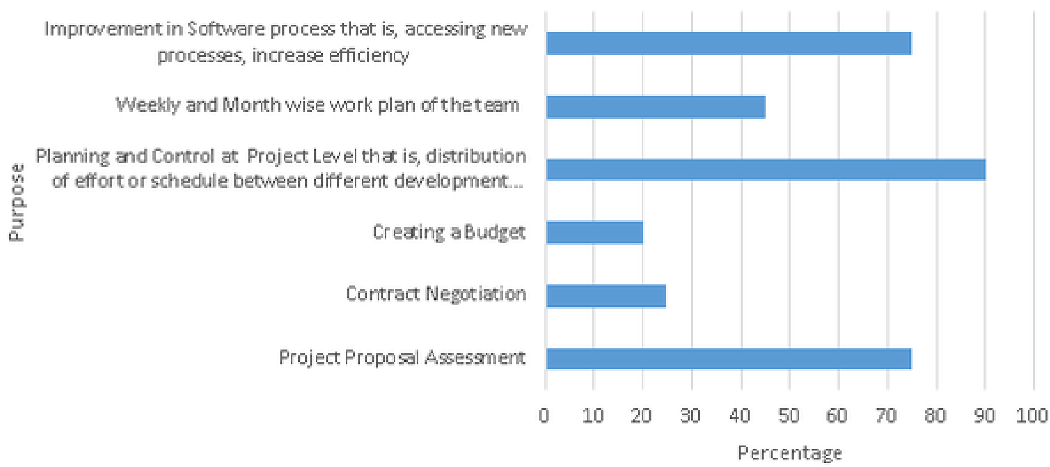


RQ3: What are the purposes of estimating effort in the software organization?

Figure 10 shows that the maximum number of software industries informed that the purpose of cost estimation is used for project planning and control (85%). Then, the next inputs are for the improvement of the software process, for instance, evaluate new processes, and increase efficiency (70%).

RQ4: Are different software size estimation models utilized in the software company?

Figure 10. Purpose of Cost Estimation



Only very few software companies exploit software size models for estimating Size, Investigation exposes that the best software size estimation method used is Function Points (34%) and the second most is use case Points (25%) while Feature Points and Object-Oriented ranked third (16%) and other various methods are also used rarely.

RQ5: What are the reasons for imprecise estimations?

"Recurrent requirements change request by users" and "User's lack of understanding of requirements" are the greatest reasons for inaccuracy and imprecision stated by the software development companies as shown in Figure 11.

RQ6: Are there any obstructions and complications in the use of software effort estimation approaches?

Figure 11. The reasons for inaccurate Estimation

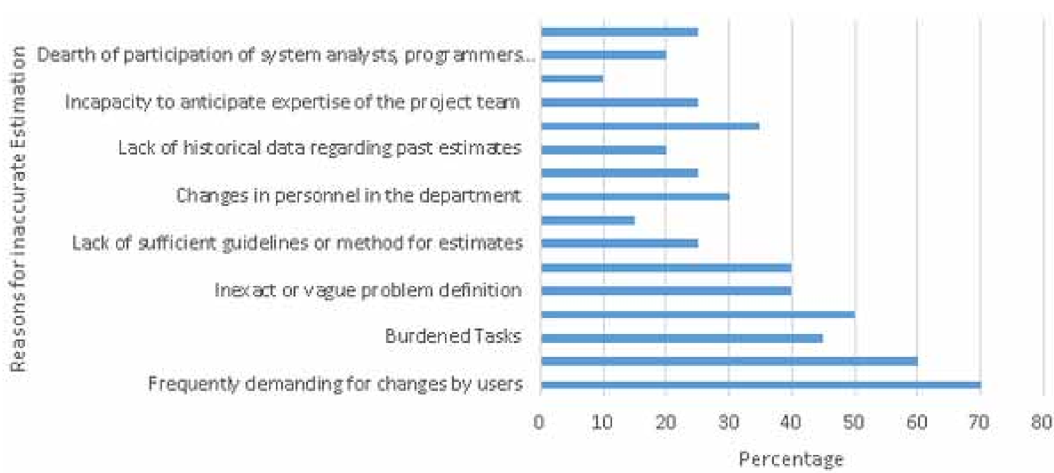
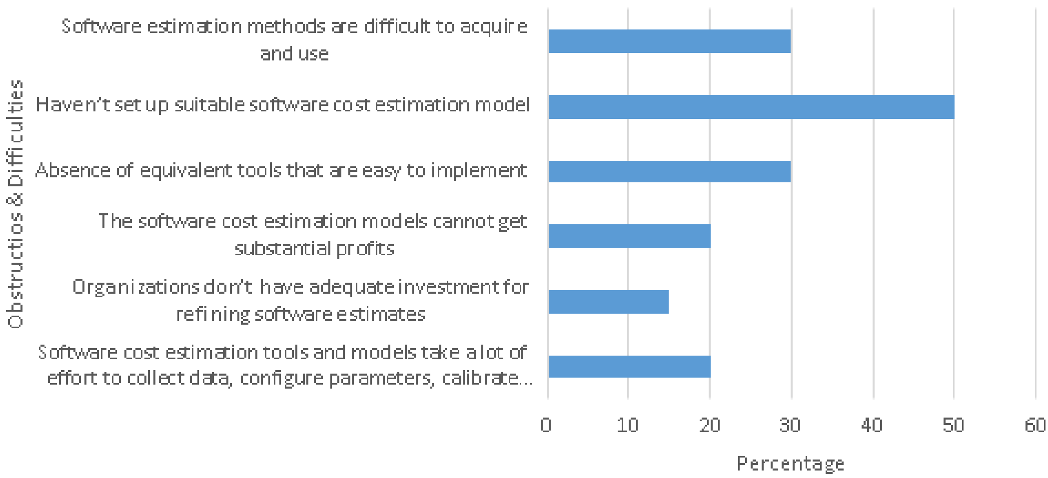


Figure 12. The Barriers and Difficult in the Application of Software Cost Estimation Models



"Not found suitable software cost estimation tool or model" is represented by the software industries to be the most revealed obstructions and problems in the use of software cost estimation methods (46%). The next most cited is "Lack consistent tools which are user-friendly" and "Software cost estimation models are complicated and difficult to use" (equally with 29%) as presented in Figure 12.

4. CONCLUSION

We presented a contemporary methodical literature survey for numerous Artificial Neural Networks utilized for estimation of software effort. The emphasis of the paper to include papers of the last decade in the arena of software engineering where software effort is estimated by making use of ANN. In this Systematic Literature Review, analysis of various datasets used in the research studies and the intelligent methods considered are surveyed. By considering the survey consisting of questions to get different aspects of the study are also examined. From the review, it is marked that a lot of investigators have focussed on calculating the development effort. From the survey, it is seen that numerous methods trained using neural networks, deep learning neural networks, and higher-order neural networks are extensively applied for software effort estimation and thereby support the planning of software development. After considering the papers, it is determined that the COCOMO dataset is the most commonly used for experimentations. The other frequently used dataset is the NASA63 and NASA93 data. Among various intelligent methods, it is perceived that Feed-Forward Neural Network is broadly implemented with MLP (multi-layer perception) and backpropagation trailed by Radial Basis Function Network. Several other hybrid methods are amalgamated with Artificial Neural Networks for better calculation of various software. The performance evaluation criteria used frequently is MMRE followed by PRED(n) and MRE, MdMRE. The prospect development in these areas should emphasize precise and exact evaluations which reduce budget overrun, barrier in delivery. Furthermore, for big-scale developments, researchers emphasize deep learning Neural Network for improved performance of quality factors.

REFERENCES

Attarzadeh & Ow. (2010). Proposing a new software cost estimation model based on artificial neural networks. *2010 2nd International Conference on Computer Engineering and Technology*, V3-487-V3-491. 10.1109/ICCET.2010.5485840

Attarzadeh, I., Mehranzadeh, A., & Barati, A. (2012). Proposing an Enhanced Artificial Neural Network Prediction Model to Improve the Accuracy in Software Effort Estimation. *2012 Fourth International Conference on Computational Intelligence, Communication Systems and Networks*, 167–172. doi:10.1109/CICSyN.2012.39

Azzeh, M., & Nassif, A. B. (n.d.). *A Hybrid Model for Estimating Software Project Effort from Use Case Points*. Academic Press.

Babu, B. V., Nagar, A., Deep, K., Pant, M., Bansal, J. C., Ray, K., & Gupta, U. (2014). *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012*. Springer.

Benala, T. R., Mall, R., Dehuri, S., & Babu, K. C. (2012). Software effort prediction using unsupervised learning (clustering) and functional link artificial neural networks. *2012 World Congress on Information and Communication Technologies*, 115–120. doi:10.1109/WICT.2012.6409060

Chiu, N.-H., & Huang, S.-J. (2007). The adjusted analogy-based software effort estimation based on similarity distances. *Journal of Systems and Software*, *80*(4), 628–640. doi:10.1016/j.jss.2006.06.006

de A. Araújo, R., Oliveira, A. L. I., & Meira, S. (2017). A class of hybrid multilayer perceptrons for software development effort estimation problems. *Expert Systems with Applications*, *90*, 1–12. doi:10.1016/j.eswa.2017.07.050

Edinson, P., & Muthuraj, L. (2018). *Performance Analysis of FCM Based ANFIS and ELMAN Neural Network in Software Effort Estimation*. Academic Press.

Goyal, S., & Bhatia, P. K. (2020). Feature Selection Technique for Effective Software Effort Estimation Using Multi-Layer Perceptrons. In P. K. Singh, B. K. Panigrahi, N. K. Suryadevara, S. K. Sharma, & A. P. Singh (Eds.), *Proceedings of ICETIT 2019* (pp. 183–194). Springer International Publishing. doi:10.1007/978-3-030-30577-2_15

Jodpimai, P., Sophatsathit, P., & Lursinsap, C. (2010). Estimating Software Effort with Minimum Features Using Neural Functional Approximation. *2010 International Conference on Computational Science and Its Applications*, 266–273. doi:10.1109/ICCSA.2010.63

Jørgensen, M., & Sjøberg, D. I. K. (2004). The impact of customer expectation on software development effort estimates. *International Journal of Project Management*, *22*(4), 317–325. doi:10.1016/S0263-7863(03)00085-1

Kaczmarek, J., & Kucharski, M. (2004). Size and effort estimation for applications written in Java. *Information and Software Technology*, *46*(9), 589–601. doi:10.1016/j.infsof.2003.11.001

Kalichanin-Balich, I., & Lopez-Martin, C. (2010). Applying a Feedforward Neural Network for Predicting Software Development Effort of Short-Scale Projects. *2010 Eighth ACIS International Conference on Software Engineering Research, Management and Applications*, 269–275. doi:10.1109/SERA.2010.41

Kaushik, A., Soni, A. K., & Soni, R. (2016). An improved functional link artificial neural networks with intuitionistic fuzzy clustering for software cost estimation. *International Journal of System Assurance Engineering and Management*, *7*(1), 50–61. doi:10.1007/s13198-014-0298-2

Kaushik, A., Tayal, D. K., & Yadav, K. (2020, April 1). The Role of Neural Networks and Metaheuristics in Agile Software Development Effort Estimation. *International Journal of Information Technology Project Management*. www.igi-global.com/article/the-role-of-neural-networks-and-metaheuristics-in-agile-software-development-effort-estimation/255102

Kumar, M., & Singh, P. (2020). *Comparative Analysis on Prediction of Software Effort Estimation Using Machine Learning Techniques*. Academic Press.

López-Martín, C. (2015). Predictive accuracy comparison between neural networks and statistical regression for development effort of software projects. *Applied Soft Computing*, *27*, 434–449. doi:10.1016/j.asoc.2014.10.033

López-Martín, C., Chavoya, A., & Meda-Campaña, M. E. (2011). Software Development Effort Estimation in Academic Environments Applying a General Regression Neural Network Involving Size and People Factors. In J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, C. Ben-Youssef Brants, & E. R. Hancock (Eds.), *Pattern Recognition* (pp. 269–277). Springer. doi:10.1007/978-3-642-21587-2_29

Qin, M., Shen, L., Zhang, D., & Zhao, L. (2019). Deep Learning Model for Function Point Based Software Cost Estimation -An Industry Case Study. *2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*, 768–772. doi:10.1109/ICICAS48597.2019.00165

Rao, P. S., Reddi, K. K., & Rani, R. U. (2017). Optimization of neural network for software effort estimation. *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, 1–7. doi:10.1109/ICAMMAET.2017.8186696

Rao, P. S., & Kumar, R. K. (2015). Software Effort Estimation through a Generalized Regression Neural Network. In S. C. Satapathy, A. Govardhan, K. S. Raju, & J. K. Mandal (Eds.), *Emerging ICT for Bridging the Future—Proceedings of the 49th Annual Convention of the Computer Society of India (CSI) Volume 1* (pp. 19–30). Springer International Publishing. 10.1007/978-3-319-13728-5_3

Reddy, C. S., & Raju, K. (2010). *An Optimal Neural Network Model for Software Effort Estimation*. Academic Press.

Resmi, V., & Vijayalakshmi, S. (2019). Kernel Fuzzy Clustering with Output Layer Self-Connection Recurrent Neural Networks for Software Cost Estimation. *Journal of Circuits, Systems, and Computers*, *29*(06), 2050091. doi:10.1142/S0218126620500917

Saraç, Ö. F., & Duru, N. (2013). A novel method for software effort estimation: Estimating with boundaries. *2013 IEEE INISTA*, 1–5. 10.1109/INISTA.2013.6577643

Sarno, R., Sidabutar, J., & Sarwosri. (2015). Improving the accuracy of COCOMO's effort estimation based on neural networks and fuzzy logic model. *2015 International Conference on Information & Communication Technology and Systems (ICTS)*, 197–202. doi:10.1109/ICTS.2015.7379898

Shepperd, M., & Kadoda, G. (2000). Using simulation to evaluate prediction techniques [for software]. *Proceedings Seventh International Software Metrics Symposium*, 349–359. doi:10.1109/METRIC.2001.915542

Venkataiah, V., Mohanty, R., & Nagaratna, M. (2019). Prediction of Software Cost Estimation Using Spiking Neural Networks. In S. C. Satapathy, V. Bhateja, & S. Das (Eds.), *Smart Intelligent Computing and Applications* (pp. 101–112). Springer. doi:10.1007/978-981-13-1927-3_11

Wani, Z. H., & Quadri, S. M. K. (2016). Artificial Bee Colony-Trained Functional Link Artificial Neural Network Model for Software Cost Estimation. In M. Pant, K. Deep, J. C. Bansal, A. Nagar, & K. N. Das (Eds.), *Proceedings of Fifth International Conference on Soft Computing for Problem Solving* (pp. 729–741). Springer. doi:10.1007/978-981-10-0451-3_65

Poonam Rijwani is B.E., M.Tech, and pursuing her Ph.D. from the Department of Computer Science Engineering, JK Lakshmipat University, Jaipur. She has several publications to her credit and has presented research papers at National and International conferences. Her area of interest includes AI for education, software development and communication technology.