Int. j. inf. tecnol. (2019) 11:569–573
https://doi.org/10.1007/s41870-018-0131-2

ORIGINAL RESEARCH

Effort estimation in agile software development using experimental validation of neural network models

Saurabh Bilgaiyan[1] · **Samaresh Mishra**[1] · **Madhabananda Das**[1]

Received: 27 October 2017 / Accepted: 28 March 2018 / Published online: 6 April 2018

Abstract Frequent requirement changes are a major point of concern in today's scenario. As a solution to such issues, agile software development (ASD) has efficiently replaced the traditional methods of software development in industries. Because of dynamics of different aspects of ASD, it is very difficult to keep track, maintain and estimate the overall product. So, in order to solve the effort estimation problem (EEP) in ASD, different types of artificial neural networks (ANNs) have been applied. This work focuses on two types of ANN-feedforward back-propagation neural network and Elman neural network. These two networks have been applied to a dataset which contains project information of 21 projects based on ASD from 6 different software houses to analyze and solve the EEP. Also, the proposed work uses three different performance metrics i.e. mean magnitude of relative error (MMRE), mean square error (MSE) and prediction (PRED(x)) to examine the performance of the model. The results of the proposed models are compared to the existing models in the literature.

Keywords Artificial neural network (ANN) · Agile software development · Effort estimation · Feedforward back-propagation neural network · Elman neural network

✉ Saurabh Bilgaiyan
saurabh.bilgaiyanfcs@kiit.ac.in

Samaresh Mishra
smishrafcs@kiit.ac.in

Madhabananda Das
mndas_prof@kiit.ac.in

[1] School of Computer Engineering, KIIT, Deemed to be University, Bhubaneswar, Odisha 751024, India

1 Introduction

The process of determining the best predicted effort required to develop a software project is termed as software effort estimation. This estimation can be divided into three levels—first stage consists of size estimation, second stage involves effort estimation and the third stage is cost estimation, the effort estimation is calculated in terms of PM (Person-Months) [1–3].

ASD process is an iterative and incremental approach which improves the overall product development routine by using customer feedback and continuous evaluation of documentation of projects. Basically, agile methodology plays a bigger role in implementing communication goals between customers in a planned manner. The prime issue of determining effort estimation in agile methods is to focus on the effort and degree of difficulties of teamwork rather than an individual [4]. ASD uses different techniques to deliver products with maximum satisfaction of stakeholders and customers. Some of the techniques include XP (Extreme Programming), Scrum, ASD (Adaptive Software Development). Some of the existing work shows that while using ASD, team uses story point approach to calculate the effort with the help of user story and project velocity as inputs. The main task is to estimate the required effort keeping in mind the technical complexity of the project. Agile software has many advantages as it is iterative, modular, incremental, customer oriented, and time-bound [5].

Artificial neural networks is an adaptive processing massive network that is based on human brain or biological neuron system. It consists of many neurons interconnected to each other to solve highly computational problems. With the advancement of modern technology, neural networks have become expert advisors in self-organized and

adaptive learning areas [6, 7]. ANNs are used to store information for recognizing patterns and solving problems for a new field of computation in the current ANNs. This learning process helps to create biological systems that can be used to configure artificial intelligence (AI) problems by creating new AI models. They are based on totally different approach for training, testing and learning data [8]. This paper describes two types of ANNs: feedforward back-propagation neural network and Elman neural network. Feedforward back-propagation ANN is the simplest type of ANN where input nodes send data to output nodes through hidden layers. It can be of two types—single layer and multi-layer perceptron. Elman networks are a special type of recurrent neural networks which allow connection between different neurons and exhibit sequence of inputs. In Elman neural network, the data is sent through hidden layers to a special layer which is treated as a set of inputs to generate the output in a directed loop. Thus, these types of network process inputs and produce outputs which are compared with actual outputs for finding the best estimation accuracy and the error [9, 10].

Further, the arrangement of paper is given as: Sect. 2 presents some of the existing work in the related area. Section 3 represents the proposed work. Section 4 describes the performance metrics used in the work followed by Sect. 5 which describe the results and analysis and finally Sect. 6 discusses the conclusion and future scope of the proposed work.

2 Background

Considerable effort is missing in the domain of effort estimation for ASD and most of the researchers have used traditional EE techniques for determining the software effort which sometimes gives inaccurate results. Singh and Sahoo [11] has worked on different neural networks and analyzed their performance which have been found to be better in case of agile dataset. Using agile methodology, Zia et al. [12] have presented a mathematical model for EE in ASD. The proposed model has been applied over self developed dataset built over 21 software projects from 6 software houses. The test results show a better precision in terms of EE. Panda et al. [13] used different kernel functions along with regression analysis to simulate an estimated effort model for the ASD. After determining the software project dataset, different ANNs applied over it to predict the overall effort. Here, simulation has given optimized results based on given dataset with less memory and less completion time. Cascade correlation networks have been found to be better in all the cases. Abrahamsson and Koskela [14] have presented a systematic approach to collect different metrics to measure cost, effort, quality and productivity of software built using XP technique of ASD. Popli et al. have proposed a mathematical model for cost and effort estimation in ASD. Tanveer et al. [15] have presented a case study on EE in ASD and depending upon the observation and findings, a framework is proposed which can be further used for optimizing the existing models of EE in ASD. Bilgaiyan et al. have presented a survey on software cost estimation in ASD using soft computing techniques. This study compares all the existing soft computing techniques for cost estimation [5].

3 Proposed work

Based on the analysis of different ANNs, the proposed approach is simulated over a dataset collected from 21 software projects using ASD methodologies from 6 software houses [12]. The dataset used for the approach is particularly based on agile methodologies and it is three-dimensional. The three dimensions consist of number of user stories, initial velocity of the project and the effort needed to complete the project in the given sprint size [13]. For increasing the prediction accuracy of the overall EE, feedforward back-propagation and Elman neural networks have been used (referring Fig. 1).

Following are the steps applied to calculate effort by using various ANNs:

- Collecting dataset: Project velocity, count of story points and actual effort are calculated from 21 projects as proposed by [12].
- Normalization of dataset: For better distribution of solution space, dataset is normalized between [0, 1].
- Classification of dataset: The selected dataset is then divided into two major parts—training set and validation set.
- Designing neural network: In this step, different ANNs are selected for analysis with the count of input layer, hidden layer, output layer and count of neurons.
- Train the dataset: The given dataset is trained according to the neural networks to get the effort estimation values.
- Validating the dataset: After, training is done, we have validated the dataset with different ANNs and tested it simultaneously.
- Performance evaluation: The evaluation criteria are based on MSE, MMRE and PRED(x) values which are calculated manually using the estimated values and the best model is considered by giving preference to PRED values.

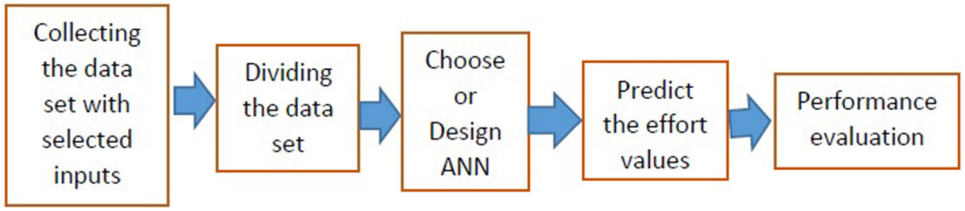

Fig. 1 Steps for proposed method

3.1 Implementation

We have simulated the neural network models using MATLAB. In our research, we have used MATLAB to find the estimated time and cost values for the chosen neural networks, thus, determining the estimation accuracy of ASD. For implementing the proposed approach, all models are tested according to the requirements and results are compared with the existing models.

3.1.1 Feedforward back-propagation network

In this network, first input gives signal to the hidden layer which in turn gives the output. Each preceding layer receives signal from its another subsequent layer. We used TANSIG function in MATLAB and two hidden layers along with input and output layer (referring Fig. 2) [10].

3.1.2 Elman network

Elman is a type of recurrent neural network (RNN) which gives an extra set of "context units" which is linked to output and input layers. The hidden layer is connected with input and output to give the desired results (referring Fig. 3) [9].

We used MATLAB with the help of NN tool and the analysis has been done for effort estimation. For achieving better accuracy, the performance graph should coincide with training, validation and testing dataset.

4 Performance criteria

There are number of performance measures criteria existing in the literature. The performance of different models can be evaluated by using three main criteria such as:

1. Mean magnitude of relative error (MMRE): It is used to calculate the percentage of difference between the actual and estimated target and finally assess the relative error averaged over a large number of values for a training set [16]:

$$MMRE_i = \frac{1}{N}\sum_{i=1}^{N}\frac{|Actual_i - Predicted_i|}{Actual_i}. \quad (1)$$

Here, $Actual_i$ is the actual effort required of ith test data, $Predicted_i$ is the estimated effort from the prediction of ith test data. N represents the total count of variables in the dataset.

2. Mean squared error (MSE): It is used to calculate the square of mean relative error by finding the difference between actual and estimated target and dividing by the total number of datasets [16]:

$$MSE_i = \frac{\frac{1}{N}\sum_{i=1}^{N}(Actual_i - Predicted_i)}{N}. \quad (2)$$

3. Prediction (PRED(x)): The third performance measure metric is percentage of prediction also represented as PRED(x), given by [16]:

$$Pred(x) = \frac{100}{N}\sum_{i=1}^{N} Q_i \quad (3)$$

$$Q_i = \begin{cases} 1, & if\ (MMRE_i) < \frac{x}{100} \\ 0, & Otherwise \end{cases} \quad (4)$$

where x = 25.

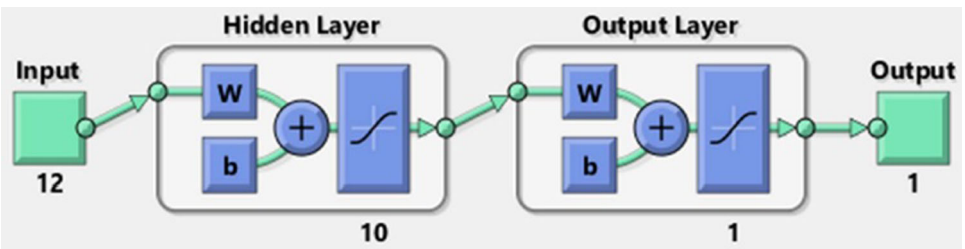

Fig. 2 Feedforward back-propagation ANN using MATLAB

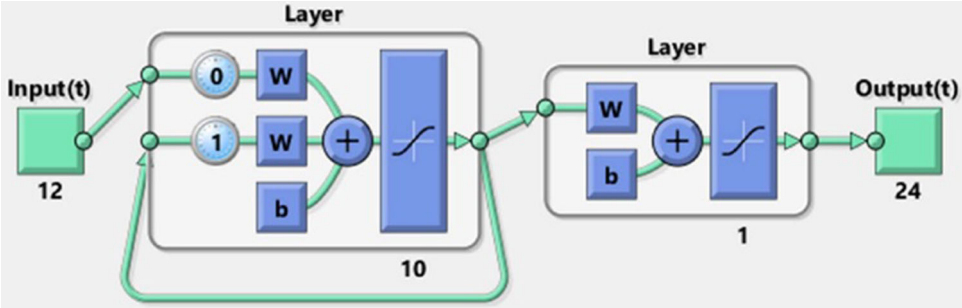

Fig. 3 Elman network using MATLAB

5 Result and analysis

The experimental analysis has been carried out on a agile dataset as discussed above [12]. In this analysis, we have predicted the estimated time values for both the networks and finally calculated the prediction results. The process takes input values as user story, team velocity and actual effort required. Effort for User story is calculated by complexity and size of the project. Story size gives an estimation of the work completed for a project based on effort. Team velocity is simply defined in terms of number of efforts completed within sprint time. Actual effort represents the effort required to complete the story.

On comparing the experimental results with the previously proposed results, it has been found that feedforward back-propagation network has given better prediction accuracy in comparison to Elman recurrent network. The prediction values have been calculated in terms of MMRE, MSE and PRED.

Table 1 shows the results of simulation in terms of MSE, MMRE and PRED(25) and from the table it can be observed that the feedforward back-propagation NN shows a improvement of 7.14, 8.851 and 0.3642% in terms of MSE, MMRE and PRED(x) respectively over Elman NN.

6 Conclusion and future work

The results from various ANN models have been compared and analyzed at the end of this study and their performance has been measured. Eventually, it has been found that the effort estimation of feedforward back-propagation network has been far better from Elman network and cascade correlation network (which is the best model in literature). Feedforward back-propagation outperforms the other two networks with the values of 0.052, 0.1349 and 95.2301 for MSE, MMRE and PRED(25) respectively. It has high computation speed, fixed computation time and fault tolerance with respect to Elman network. One limitation as recognized by the authors is that the proposed method may not perform equally good for other data sets collected from heterogeneous software development methods. For future work, we will simulate other ANNs over some more robust datasets for effort estimation in ASD.

Table 1 Comparative analysis of predicted efforts for different types of ANNs

Name of performance metric	MSE	MMRE	PRED(25)
Cascade correlation NN [13]	0.059	0.1486	94.7649
Elman NN	0.056	0.1480	94.8659
Feedforward back-propagation NN	0.052	0.1349	95.2301

References

1. Zare F, Zare HK, Fallahnezhad MS (2016) Software effort estimation based on the optimal Bayesian belief network. Appl Soft Comput 49(1):968–980
2. Jorgensen M, Shepperd M (2007) A systematic review of software development cost estimation studies. IEEE Trans Softw Eng 33(1):33–53
3. Sagar K, Saha A (2017) A systematic review of software usability studies. Int J Inf Technol. https://doi.org/10.1007/s41870-017-0048-1
4. Strode DE (2016) A dependency taxonomy for agile software development projects. Inf Syst Front 18(1):23–46
5. Bilgaiyan S, Mishra S, Das M (2016) A review of software cost estimation in agile software development using soft computing techniques. In: 2nd International conference on computational intelligence and networks, KIIT, Bhubaneswar. IEEE, pp 112–117
6. Dave VS, Dutta K (2014) Neural network based models for software effort estimation: a review. Artif Intell Rev 42(2):295–307
7. Kumar A, Sharma R (2018) Neural/fuzzy self learning Lyapunov control for non linear systems. Int J Inf Technol. https://doi.org/10.1007/s41870-017-0074-z
8. Cabessa J, Villa AEP (2015) Recurrent neural networks and super-Turing interactive computation. Artif Neural Netw 4:1–29
9. Ciarlini P, Maniscalco U (2008) Wavelets and Elman neural networks for monitoring environmental variables. J Comput Appl Math 221(2):302–309
10. Konaté AA, Pan H et al (2015) Generalized regression and feed-forward back propagation neural networks in modeling porosity from geophysical well logs. J Pet Explor Prod Technol 5(2):157–166

11. Singh J, Sahoo B (2012) Application of artificial neural network for procedure and object oriented software effort estimation. Int J Softw Eng Technol 1–9
12. Ziauddin, Zia S, Tipu SK (2012) An effort estimation model for agile software development. Adv Comput Sci Appl 2(1):314–324
13. Panda A et al (2015) Empirical validation of neural network models for agile software effort estimation based on story points. Procedia Comput Sci 57:772–781
14. Abrahamsson P, Koskela J (2004) Extreme programming: a survey of empirical data from a controlled case study. In: International symposium on empirical software engineering. IEEE, pp 73–82
15. Tanveer B et al (2017) Effort estimation in agile software development: case study and improvement framework. J Softw Evol Process 29(11):1–14
16. de Arajo RA et al (2012) An evolutionary morphological approach for software development cost estimation. Neural Netw 32:285–291