# An Agile Effort Estimation Based on Story Points Using Machine Learning Techniques

**Ch. Prasada Rao, P. Siva Kumar, S. Rama Sree and J. Devi**

**Abstract** Nowadays, many software companies face the problem of predicting the accurate software effort. Most of the software projects are failed due to over budget and over schedule as well as under-budget and under-schedule. The main reason for the failure of software projects is inaccurate effort estimation. To improve the accuracy of effort estimation, various effort estimation techniques are introduced. Functional points, object points, use case points, story points, etc., are used for effort estimation. Earlier, traditional process models like waterfall model, incremental model, spiral model, etc., are used for developing the software, but none of them have given the successful projects to the customers. Now, 70% of the application software's have been developed by the agile approaches. The success rate of the projects developed by using agile methodologies has been increased. The major objective of this research is to estimate the effort in agile software development using story points. The obtained results have been optimized using various Machine Learning Techniques to achieve an accurate prediction of effort and compared performance measures like MMRE, MMER, and PRED.

**Keywords** Adaptive Neuro-Fuzzy Interface System (ANFIS) Effort estimation · Generalized Regression Neural Networks · Mean Magnitude of Error Relative (MMER) · Mean Magnitude of Relative Error (MMRE) Prediction Accuracy (PRED) · Radial Basis Function Networks (RBFNs)

Ch. Prasada Rao (✉) · P. Siva Kumar
K L University, Vaddeswaram, Guntur, India
e-mail: prasadarao.chatla@aec.edu.in

P. Siva Kumar
e-mail: spathuri@kluniversity.in

S. Rama Sree · J. Devi
Aditya Engineering College, Surampalem, India
e-mail: ramasree_p@rediffmail.com

J. Devi
e-mail: charandevi85@gmail.com



V. Bhateja et al. (eds.), *Proceedings of the Second International Conference on Computational Intelligence and Informatics*, Advances in Intelligent Systems and Computing 712, https://doi.org/10.1007/978-981-10-8228-3_20

## 1 Introduction

Project management in the software development is the prominent responsible activity. The major objectives of software project management are to plan, establish the scope document, determine the business case, estimating the effort accurately, providing the development and testing environments to feature teams, identify the probable risks, etc. The contract between the product owner and organization purely depends on the project scope. The contract includes estimation of the budget, deadlines for the project to be delivered and it is defined as a business case. The estimation of the effort is measured in terms of the size of the project, the time required to complete the project and how much budget is needed to complete the project on time. The accurate prediction of the effort in the software development leads to the success of software project otherwise it leads to failure of the project. The main reason to get a failure of the software projects is poor management, inaccurate estimation, continuous change of requirements, incomplete requirements, lack of communications among developers, unable to identify the risks in the early stage of development, adopting the relevant process model, etc. There is a statement in software development "Good Management can't assure the success of the project but Poor Management leads to failure of the projects." The effective and efficient process model plays a major role in developing the successful software projects. The Standish Group reported in 2015, the success rate of the projects developed by waterfall model is only 11% and when coming to the projects developed by using the agile methodologies: 39% of the projects delivered on time, on a budget, with high quality and satisfy all the customer needs and expectations. Now, 80% of the software industries are transforming from conventional process models (Waterfall, Spiral, Increment, RUP, Prototyping, RAD, etc.) to agile process models (Scrum, XP, FDD, etc.) [1]. The scrum and XP are very popular and widely recognized.

In 2001, some of the experts from top software industries form a team and establish an "Agile Manifesto" [2]. This is called as a bible for software organization to develop the successful software projects. The advantages of agile process models over conventional process models are:

- Deliver some usable software to the Product Owner (Customer) in a reasonable time
- Good chemistry among the team members to improve the productivity
- Involvement of Product Owner in the entire development to give better feedback or he can also change some features
- Agile teams are self-organized as well cross-functional for effective progress of other software development
- Regular Assessment by feature teams and so on

Either in Conventional or Modern Approaches, the accurate effort estimation is essential for developing the successful projects. To predict the effort in software projects, many metrics are introduced like KLOC, Functional Points, Class Points, Use Case Points, Object Points, etc. None of them have given the accurate results. In the agile process model, Story Points are derived from the user stories [3]. User stories can be written in a very general language where everyone can able to understand. User stories are of the form

As a <Designation>
I want to <some task to be done>
So that <goal to be achieved>
**Ex**: As a parent
I want to know the percentage of my son
So that I can take necessary actions to improve their studies.

Let us take an example to estimate the time and cost using User stories:

| | |
|---|---|
| Total User Stories | 20 |
| 10 of them can be measured in 3 Story Points each | 10 * 3 = 30 |
| Remaining can be measured in 5 Story Points each | 10 * 5 = 50 |
| Total Story Points | 30 + 50 = 80 |
| Velocity | 10 |
| Total number of sprints (total Story Points/Velocity) | 80/10 = 8 |
| Each sprint may take | 4 weeks |
| Estimated time for development | 32 weeks |
| Each sprint cost | 50,000 |
| Estimated cost for development | 8 * 50,000 = 400,000 |

There are many techniques available for effort estimation like Delphi Estimation, Planning Pocket, Use Case Points, etc., [4]. But they are mainly designed for conventional process models. We can apply these conventional estimation techniques to agile methodologies, but they will not give accurate results. To improve the accuracy of effort estimation, we have to use story points in agile methodologies. The implementation of effort estimation is carried out using 21 project data set and evaluated using different machine learning techniques. Most of the researchers are considered the performance metrics like MMRE, MMER, and PRED(X). Estimation of the effort has been implemented using some soft computing techniques like ANFIS, RBFN and evaluated and compared using the three performance metrics that are mentioned above.

## 2 Related Work

A. Schmietendorf et al. [5] provided the definition and characteristics of agile methodologies and described different possibilities of effort estimation for different types of agile methods, especially for XP projects and provided an analysis on the importance of prediction of software effort and agile software development. S.M. Satapathy et al. [6] described various Adaptive Regression Techniques such as multi-layer perceptron, projection pursuit regression, multivariate adaptive splines, K nearest neighbor regression, constrained topological mapping and radial basis function networks for predicting the software effort and provided the cost estimation Class Point approach.

S. Keaveney et al. [7] proposed different cost estimation techniques for both conventional estimation techniques as well as for agile software development approaches and examined the causes of inaccurate estimates and steps to improve the process. Shashank Mouli Satapathy et al. [8] described various SVR kernel methods for improving the estimation accuracy that helps in getting optimal estimated values. This estimation is carried out using Story Point approach and performance of different SVR methods were compared in terms of MMRE and PRED.

Hearty et al. [9] provided an investigation about Bayesian Networks, which can combine sparse data, prior assumptions, and expert judgment into a single model. Shows how XP can learn from project data in order to make quantitative effort predictions and risk determinants and also described project velocity. Donald F. Specht et al. [10] described Generalized Regression Neural Network which is a one-pass learning algorithm and also explains neural network implementation with one-dimensional example and adaptive control systems.

## 3 Evaluation Criteria

The performance of various methods discussed in the proposed approach is determined by the following criteria

- The Mean Magnitude of Error Relative can be calculated as

$$\mathrm{MMRE} = \frac{100}{\mathrm{N}} \sum_{1}^{\mathrm{N}} \frac{|\mathrm{P_i} - \mathrm{A_i}|}{\mathrm{A_i}}. \tag{1}$$

- The Mean Magnitude of Error Relative can be calculated as

$$\mathrm{MMER} = \frac{100}{\mathrm{N}} \sum_{1}^{\mathrm{N}} \frac{|P_i - A_i|}{P_i}. \tag{2}$$

- Prediction Accuracy can be calculated as

$$\mathrm{PRED(X)} = \frac{100}{\mathrm{N}} \sum_{1}^{\mathrm{N}} \begin{cases} 1 & \text{if } \frac{|\mathrm{P_i} - \mathrm{A_i}|}{\mathrm{A_i}} < \frac{\mathrm{N}}{100} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

where

- N The total observations or projects in test set
- $P_i$ Predicted effort of ith project in the test set
- $A_i$ Actual effort of ith project in the test set
- X MMRE value of the test set for which we are calculating the prediction.

The model with high PRED value and low MMRE and MMER values will give accurate estimation results.

## 4 Proposed Approach

The proposed approach is based on the data collected from 21 projects' dataset [11]. The dataset is used to evaluate software effort in terms of time and cost. In this paper, Story Points and Project Velocity are taken as inputs.

### *4.1 Story Points*

Story Points are a unit of measure that is used to implement a User Story. When estimating based on Story Points, we will assign a size to each User Story [12]. Generally, Story Points follow Fibonacci like sequence

$$\text{i.e., } 0, \frac{1}{2}, 1, 2, 3, 5, 8, 13, 20, 40, 100, \propto, ?$$

While predicting the size of each user story, most of the industries will prefer the either 3 or 5 as size for the user story. The entire team who are involved in developing the sprints in scrum will be estimating the total story points of all the user stories. While predicting the story points for each user story from all team persons there will be some different opinions and weights for user stories. In this case, the people with different opinions will be explained their opinions and re-estimate it. Still, it has ambiguity, then scrum master will be considered the average weights of the entire team. Later these have been partitioned into sprints. Generally, each sprint may take 2–4 weeks of time with the help of 5–8 people of the team with one scrum master.

## 4.2 Project Velocity

The Project Velocity is the deliveries made per sprint. Sprint is time-boxed iteration. Velocity may differ from initial sprint to next sprints, i.e., velocity may be less for initial sprints and may increase in subsequent sprints [13].

## 4.3 Soft Computing

In recent years soft computing gains popularity over hard computing because of its easiness, availability in Computer Science, Machine Learning, Artificial Intelligence, etc. By applying soft computing we can easily find better solutions to complex problems at a reasonable cost. The techniques provided by soft computing are Fuzzy Systems (FS), Neural Networks (NN), Probabilistic Reasoning (PR), etc. We can get more accurate results by using soft computing than hard computing. Here, in this paper, we have implemented the Machine Learning Techniques such as Adaptive Neuro-Fuzzy Modeling, Generalized Regression Neural Network and Radial Basis Function Networks (RBFNs) using MATLAB software.

### 4.3.1 Adaptive Neuro-Fuzzy Interface System

In this, we need to build Adaptive Neuro-Fuzzy Interface System (ANFIS) to train Sugeno systems. An ANFIS is a fuzzy system and the training is done using neuro-adaptive learning methods as in the case of neural network training. MATLAB provides a Fuzzy Logic Toolbox which provides a number of command-line functions to train Sugeno-type Fuzzy Interface System (FIS) using the data provided for training. First, we need to generate FIS structure using genfis algorithm, because anfis algorithm takes genfis (genfis1/genfis2/genfis3) function as one of the argument. For this we have 3 methods available in MATLAB (Table 1).

In the proposed approach we have used genfis2.

The syntax used in the proposed approach is

**Table 1** Different functions of genfis available in MATLAB

| Function name | Description |
|---|---|
| genfis1 | It will generate FIS structure from data (training set) using grid partition |
| genfis2 | It will generate FIS structure from data (training set) using subtractive clustering |
| genfis3 | It will generate FIS structure from data (training set) using FCM clustering |

$$\text{Fis} = \text{genfis2}(\text{Input}, \text{Output}, \text{Rad}, \text{Bounds}, \text{Options}),$$

where

| | |
|---|---|
| Input | Input from training set |
| Output | Output from training set |
| Rad | In this research method Rad value is [0.5 0.4 0.3] |
| Bounds | Specifies how to map the training set data i.e., The bounds argument can be empty matrix or need not be provided<br>In the proposed approach, it is an empty matrix |
| Options | It is also an optional vector which specifies algorithm parameters to override the default values. The default value for this argument is [1.25 0.5 0.15 0] and this value is only used in the proposed approach. |

The anfis uses genfis (genfis1/genfis2/genfis3) to create initial FIS. Now we have to give this genfis2 (in the proposed approach we have used genfis2 algorithm) output to anfis function which acts as a training routine for Sugeno-type FIS. The syntax of anfis that is used in this research method is

$$\text{fis1} = \text{anfis}(\text{Traindata}, \text{Fis}, \text{Dispopt}),$$

where

| | |
|---|---|
| Traindata | Training set |
| Fis | Output of genfis2 is used in our approach |
| Dispopt | If 1 is specified, training information will be displayed and if 0 is specified information won't be displayed. In the proposed approach 1 is specified. |

In the Adaptive Neuro-Fuzzy Modeling, the last step is performing fuzzy interface calculations. For this we need evalfis algorithm. The syntax used in the proposed approach is:

$$\text{estimated} = \text{evalfis}(\text{Input}, \text{fis1})$$

where

| | |
|---|---|
| Input | Input data from test set |
| fis1 | FIS structure generated by anfis function. |

### 4.3.2 Generalized Regression Neural Network

To implement this algorithm MATLAB provides a function known as newgrnn. GRNN which is a type of Radial basis network. GRNN is used to approximate functions [10]. They are quick to design. The general form of newgrnn function is

$$\text{net } = \text{newgrnn(Input, Output, Spread)},$$

where

| | |
|---|---|
| Input | Input vector (training set inputs) |
| Output | Target vector (training set outputs) |
| Spread | Constant (default 1.0). In the Proposed approach default value is taken. |

### 4.3.3 Radial Basis Function Networks (RBFNs)

RBFNs are similar to neural networks. Input to the RBFNs can be one or more numeric values and output from RBFNs is also one or more numeric values. RBFN is also known as a Radial net which can be used for classifying the data and to make predictions. It contains three layers:

1. Input layer: Takes input vector
2. Hidden layer: Contains neurons
3. Output layer: Contains linear combinations of neurons and It will form network outputs by taking a weighted sum of outputs from the second (hidden) layer.

If there are N neurons then the number of categories will be N + 1. There are some Radial Basis Functions such as Gaussian, Multiquadrics, and Inverse Multiquadrics [14]. There are two types of functions available in MATLAB to implement BBFNs. They are as follows:

1. newrb: It is used to design a radial basis network. This radial basis function is also used for function approximation. The newrb will set a Mean Square Error (MSE) goal and it adds neurons to hidden layer until it reaches the goal. This function stops either when the goal is reached or when maximum number of neurons is reached. The syntax used in the proposed approach:

$$\text{net } = \text{newrb(Input, Output, Goal, Spread)}.$$

   Arguments are discussed later.
2. newrbe: It is used to design exact radial basis network. These are also used for function approximation. This function is used to design radial basis network very quickly. If there are n input vectors then there will be n neurons.
   Syntax:

$$\text{net } = \text{newrbe(Input, Output, Spread)}$$

Let us discuss the arguments of all 3 functions.

| | |
|---|---|
| Input | Input vector (training set inputs) |
| Output | Target vector (training set outputs) |

Spread  Constant (default is 1.0). In the Proposed approach default value is taken
Goal  Mean Square Error goal (default is 0.0). In this research method, default value is taken.

Note: For all three functions are simulated for new input to get the predicted values.

Syntax:

$$\mathrm{sim}(\mathrm{Net}, \mathrm{Newinput})$$

where

Newinput  Input from test set
Net  Net is the output of newgrnn/newrb/newrbe function.

Most of us try to give the outputs directly using matrix, but when we are using different number of columns in input and output arguments it will show error. So that we have to complement all the variables which we are passing as arguments in newgrnn, newrb, newrbe, and also for sim function we need to complement test input set but not the radial basis function (net variable) which is going to be simulated. As we took complement to the inputs, we need to complement the final output, i.e. the output from sim function.

## 5 Experimental Results

Table 2 shows the results of our approach. We have taken three machine learning techniques: Adaptive Neuro-Fuzzy Modeling, Generalized Regression Neural Network, and Radial Basis Function Networks. The obtained results are based on the training set and test set which we have used in the proposed approach. We can observe that all functions differ in terms of MMRE and MMER, but almost all functions gave similar PRED.

**Table 2** Comparision among anfis, newgrnn, newrb, newrbe functions

| Model | Function | MMRE | | MMER | | PRED | |
|---|---|---|---|---|---|---|---|
| | | ***Cost*** | ***Time*** | ***Cost*** | ***Time*** | ***Cost*** | ***Time*** |
| Adaptive Neuro-Fuzzy Modeling | Anfis | 3.9079 | 8.4277 | 3.9587 | 19.5922 | 57.1429 | 76.1905 |
| Generalized Regression Neural Networks | Newgrnn | 4.8335 | 2.7864 | 5.8700 | 3.2042 | 76.1905 | 76.1905 |
| Radial Basis Function Networks | Newrb | 9.9604 | 8.0909 | 16.2868 | 6.6430 | 76.1905 | 76.1905 |
| | Newrbe | 10.6099 | 8.0909 | 11.3974 | 6.6430 | 76.1905 | 76.1905 |

## 6 Threats to Validity

- We have taken only 21 projects' data and examined with few projects. Maybe it gives better results from small set of data set, so we should consider the larger data set and examine
- Automation of User Stories to Story Points is not possible.
- Experienced and skill set of scrum team may also be considered for the success of software projects.

## 7 Conclusion

In this paper Story Point approach is used for predicting the software effort. Story Point approach is one of the most popular effort estimation techniques that can be applied to agile software projects. In this paper, three Machine Learning Techniques are chosen for predicting the software effort. The three techniques are Adaptive Neuro-Fuzzy Modeling, Generalized Regression Neural Networks and Radial Basis Function Network. Adaptive Neuro-Fuzzy Modeling is implemented using anfis function and GRNN is implemented using newgrnn function and RBFNs are further implemented using two algorithms namely newrb, newrbe which are provided by MATLAB software. The results are analyzed using MMRE, MMER, and PRED parameters. This study can also be expanded using Fireworks Algorithm (FA), Random Forest, etc.

## References

1. D. Cohen, M. Lindvall and P. Costa, "An introduction to agile methods," Advances in Computers, vol. 62, pp. 1–66, 20/Jan/2004.
2. M. Flower and J. Highsmith, "The Agile Manifesto", Software Development, vol. 9, no. 8, pp. 28–35, August 2001.
3. "Writing User Stories, Examples and Templates in Agile Methodologies", Jan 8-2016.
4. Tutorialpoint.com, "Estimation Technique".
5. A. Schmietendorf, M. Kunz, and R. Dumke, "Effort Estimation for Agile Software Development Projects", in 5th Software Measurement European Forum, 2008.
6. S.M. Satapathy, M. Kumar, and S.K. Rath, "Fuzzy-class Point approach for software effort estimation using various adaptive regression methods", CSI Transactions on ICT, vol. 1, no. 4, pp. 367–380, 31 December 2013.
7. S. Keaveney and K. Conboy, "Cost estimation in Agile Development Projects", in ECI, 2006, pp. 183–197.
8. Shashank Mouli Satapathy, Aditi Panda, Santanu Kumar Rath, "Story Point Approach based Agile Software Effort Estimation using Various SVR Kernel Methods", International Conference on Software Engineering, 2014 July 1–3.

9. P. Hearty, N. Fenton, D. Marquez, and M. Neil, "Predicting Project Velocity in XP Using a Learning Dynamic Bayesian Network Model," Software Engineering, IEEE Transactions on Software Engineering, vol. 35, no. 1, pp. 124–137, 2009.
10. Donald F. Specht, "A General Regression Neural Network", IEEE Transactions on, vol. 35, no. 1, pp. 124–137, November 2009.
11. Z.K. Zia, S.K. Tipu and S.K. Zia, "An Effort Estimation Model for Agile Software Development", Advances in Computer Science and its Applications, vol. 2, no. 1, pp. 314–324, July 2012.
12. E. Coelho and A. Basu, "Effort Estimation in Agile Software Development using Story Points", Development, vol. 3, no. 7, August 2012.
13. Monty Python and Holy Grail, "Velocity: Measuring and Planning an Agile Project", Dec 10–2007.
14. Shafinaz Elinda Saadon, Zarita Zainuddin, "Radial Basis Function and Multilayer Perceptron Neural Networks in Function Approximation".