Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 2 (2024)

A Novel Effort Estimation Framework for Agile Based Projects

Shivali Chopra, Arun Malik
Lovely Professional University, Phagwara, Punjab, India

Abstract:- Software estimation is the most essential activity in project management. Numerous researchers across the globe have worked on the issue of software effort estimation and have contributed significantly. With advancements in technology and software process models, the old estimation methods may not yield fruitful results for project managers. There is a need to reframe the estimation process in Agile-based project development. We have proposed a novel continuous estimation framework named Agilator to assist project managers involved in software estimation-oriented tasks. This framework provides two novel features. The first one is the auto-adjustment of effort through learning gain accumulated and adjusted from errors deduced during iterations. This feature makes the system end-to-end trainable, laying the foundation for a continuous estimation framework. The second feature is real-time prediction available for Scrum masters. The proposed framework will not replace existing expert-based estimation; instead, it will assist by participating and contributing AI-assisted input to the team. This paper helps to minimize the estimated effort and actual effort for various system stakeholders. We have presented the Agilator framework using ANFIS-EEBAT i.e., Adaptive Neuro-fuzzy Inference System – Energy Efficient BAT algorithm.

Keywords*: Agile, effort, estimation, expert system, framework*

1. Introduction

In a software project management plan, estimation is the second major step, following the software requirement specification document. We propose a novel estimation framework to redefine the entire estimation process in Agile-based projects. The core of the framework is its continuous and end-to-end trainable nature, inspired by human experience in real life. Similar to how humans learn from experience and may complete an assigned task in less time or with less effort compared to the initial time taken for similar tasks, the machine also learns from the initial system-generated estimates. It updates or retrains the system's learning accordingly. A learning gain is calculated from errors, which basically specifies the variation in the actual effort and estimated effort. During retraining, actual estimates are seeded into the system for more realistic and precise predictions.

The remaining paper is structured into several sections. Section 2 presents an overview of related work, followed by the Agilator estimation framework with ANFIS-EEBAT[1] in Section 3. Section 4 sums up and describes the scope of future work.

2. Related Work

Research on scrum estimation is basically defined into two verticals: classification of the text [2] and numerical based estimation [3]. A large amount of work has been done in regression-based estimation [4], but less research has been done in classification-based story point estimation. A few frameworks proposed in field of the agile projects estimation are given in Table 1. In the context provided, Morakot Choetkiertikul et al. (2018) [5] invented a model that predicts story points by integrating two sturdy deep learning frameworks: LSTM and highway nets. They assert that their end-to-end system, Deep-SE, is trainable using raw input data and can forecast the story point estimate without having any dependencies on manual feature engineering. The Long Short-Term Memory (LSTM) network processes the user-supplied sequence of words into vectors, which are subsequently inputted into the Recurrent Highway Network for iterative vector transformation. This process

culminates in a final vector representation, effectively encoding the text. Based on this final vector, linear regression predicts the story point. Their method outperforms well-known techniques like Doc2Vec and Bag-of-Words. Rodrigo G. F. Soares (2018) [6] proposes employing Natural Language Processing autoencoders to estimate effort in publicly available projects. They advocate for the utilization of denoising techniques to enhance the learning process of autoencoders by filtering out invalid inputs. As a result, they employ stacked denoising autoencoders and train them using stochastic gradient descent while performing hyperparameter optimization via randomised search. Support Vector Machines were used to predict the outcomes. Their method outperforms Term Frequency-Inverse Document Frequency while claiming no statistical differences between autoencoders. Przemyslaw Pospieszny et al. (2018) [7] utilize the ISBSG dataset and three ML algorithms and their collaborative accumulation to predict the effort for agile based projects. They discovered that SVM outperformed GLM and ANN with low MMRE and high PRED of 25 and.30, respectively. They also experimented by observing changes in the dependent variable when using a logarithmic transformation, with the result that GLM outperformed ANN but still trailed SVM. They conclude that because of the mixed entries in the ISBSG dataset, the results may vary and that if their technique is applied to a homogeneous set of data, such as that from Sourceforge, the results will differ.

Vlad-Sebastian et al. (2017) [8] propose an automatic effort estimation tool based on software task descriptions of tasks and activities using Natural Language Processing. A skilled software development team can complete complex software projects, albeit at a higher cost. The MMRE is the accuracy estimator used. They assert that regression results outperform classification results. Ziauddin et al. (2012) [9] propose an integrated expert system for estimating software costs. Their system consists of a graphical user interface that answers user questions, a Natural Language Processor that communicates between the graphical user interface and the Inference Engine, an Inference Engine that connects to the knowledge base and computes the results, a knowledge base that stores the information provided, and a database that stores the results. With a mean magnitude of relative error of 5.08, their proposed system outperforms the COCOMO and FPA estimation models. Porru et al. (2016) [10] propose a trustworthy classifier for story point estimation in agile projects based on machine learning. The classifier extracts features using Term Frequency Inverse Document Frequency, then selects features by forming a feature matrix and selecting features based on their statistical distribution, and finally selects based on the classifier. According to their proposed method, the mean magnitude relative error is reported to be as high as 0.61. Cláudio Ratke et al. (2019) [11] propose using natural language processing in conjunction with a naive Bayesian classifier to estimate agile effort. The story description is processed by tokenizing the words using parts of speech tagging and then calculating the keyword probability. They claim an 83% accuracy rate. Shashank Mouli Satapathy et al. (2017) [12] proposed improving effort accuracy by using story points as the foundation of their approach. Their method uses three ML algorithms i.e. gradient boosting, random forest and stochastic gradient boosting, to test and validate data from twenty one projects from Ziauddin et al.'s paper. To normalise the resulting dataset, logarithmic transformations were applied. Based on velocity and story point count, their model predicted effort. They concluded that Stochastic Gradient Boosting outperformed the other algorithms and suggested future research using BN.

Binish Tanveer (2017) [13] clarified that before attempting to predict effort estimation, related strategies must have a probability of success such that the impact analysis changes. They claim that a framework for guidelines is established through expert discussion. Sufyan Basri et al. (2016) [14] proposed the use of non-algorithmic models in agile projects. Similarly to agile, volatile requirements consider a change as a must for cost prediction and must be added to the final cost. They mention a table of change type values as well as the percentage of change effort required for changes based on different phases. Their MRE results were unsatisfactory. Aditi Panda et al. (2015) [15] compared various deep learning-based neural network (NN) techniques for predicting the agile project's effort with the use of story points. Various models like PNN,CCNN, GRNN etc. are used with 21 projects from the Ziauddin [9] et al. paper, which was divided into training and testing data. They discovered that CCNN outperformed the others with a PRED rate of 94% and recommended SGB, RF, and the story point approach for future research work. L.R. Nerkar et al. (2014) [16] compared existing cost estimation techniques using algorithmic models such as COCOMO, Putnam, and the FP model, as well as non-algorithmic

models such as analogy-based, Parkinson's law, price to win, and EJ. They also proposed a web cost model without claiming any impressive evidence. We created a matrix that compares various techniques. DEEP-SE, a novel classification method, outperforms various other contextual literary resources and can be used to eliminate differences between predicted and actual effort.

The consolidated literature review of proposed frameworks is given in Table 1.

TABLE 1. Summary of estimation frameworks in story point estimation

Proposed framework	Techniques used	Findings
Generic scrum estimation framework [17]	Size and duration of user stories	Authors proposed a generic estimation framework which uses the duration and actual size of the user stories as input to calculate how much work a scrum project will require.
Proposed methodology I and II [18]	Fuzzy inference system	Authors created a methodology framework by using fuzzy inference system to estimate effort and cost of agile projects.
Agile Success Estimation Framework [19]	Decision tree machine learning models	To estimate and measure success, a framework has been proposed which includes budget, schedule , scope and stakeholder's satisfaction as four success dimensions
Framework for enhancing story points [20]	Custom algorithm	Authors applied their proposed framework to calculate the "enhanced story point (ESP)" and applied their proposed algorithm on three case studies. As a result, they find ESP is higher in low experience project teams.
Multiclass classification model [21]	Decision Trees, SVM and Naïve Bayes	As per author, naïve Bayes, SVM and decision tree classifiers were applied and support vector machine classifier outperformed all the other classifiers in terms of accuracy.
Story point-based effort estimation model [21]	SVR, Gradient Boosting (GB) Algorithm, Random Forest Regression (RFR)	Gradient Boosting algorithm has highest PRED i.e. 99.8% and lowest MAE i.e., 0.2% when compared to the results of RFR, GB Algorithms and SVR
Effort Estimation in Agile using Story Point approach [22]	SVR Kernel (SVRK) Techniques	Author optimizes the results of the results from various SVRK techniques. SVR Linear Kernel Result, Polynomial Kernel Result, RBF Kernel Result as well as SVR Sigmoid Kernel Result.
GPT2SP approach [23]	Story Point Estimation Approach based on transformers for agile projects	Author has done rigorous evaluation on 23,313 issues and gets a mean Median of 1.16. They also developed proof of concept tool which is basically an explainable AI.
Text level GNN[24]	Graph neural network	Authors created a GNN and claimed an accuracy of 80 percent with their proposed framework
Attention networks [25]	Hierarchical attention networks	Authors made use of HAN for the story point estimation and claimed an accuracy of 87 percent vis-à-vis other state-of-the-art algorithms

3. Agilator estimation framework with ANFIS-EEBAT

Fig. 1 depicts an abstract view of the Agilator framework in which an authorised project stakeholder, such as a project manager, will provide specific inputs via a defined user interface and observe expected outputs. The most important work will be done by a neuro-fuzzy inference engine, which will create complex relationships

and rules for the knowledge base. The hyperparameters of the proposed integrated model can be optimised using a novel optimization technique called ANFIS-EEBAT to scale the search space for the desired solutions. The knowledge base is a collection of models that have already been trained and solutions that have been updated. After each iteration, learning gain, G, saves these models and solutions. The knowledge base is also a collection of pre-trained models and update-train solutions that are recorded after each sprint by learning gain, G. The learning gain G is directly related to the Agile Velocity V, which is the number of units of effort (measured in story points) that are done during a sprint. The algorithmic steps presenting the learning gain between subsequent sprints is given below.

Step1: Calculate the effort (initial) for 1st sprint of a scrum project.

Step2: Calculate team velocity (initial) for the 1st sprint.

Step3: Value of learning gain (G) is initialized to zero for 1st sprint.

Step4: Determine the current sprint velocity $V_{i_{current}}$ and the previous sprint velocity, $V_{i_{previous}}$

Step5: Make an assignment of $V_{i_{previous}}$ as $V_{initial}$

Step6: Calculate the gain i.e., V_G , for 2nd sprint using $V_G = V_{i_{current}} - V_{i_{previous}}$ (1)

Step 7: Calculate the value of VAF i.e., Velocity Adjustment Factor and assign it zero in case of no change.

Step 8: Adjust the gain, V_G by $V_G = (V_G)^{VAF}$ (2)

Step 9: To calculate the value of G, the value of needs to be observed as given below,

V_G = Negative G, for negative change in the velocity

Zero, due no change in the velocity (3)

Positive, for positive change in the velocity

Step 10: Find the value of G from V_G

Step 11: Adjust the value of effort of 2nd sprint by E2 = $(\text{Einitial})^{G}$ (4)

Step 12: Readjust the effort as per the upcoming inputs/features

The pretrained model dynamically adapts after each sprint/iteration, allowing for adjustments in the magnitude of the model coefficients based on the calculated learning gain G.

We have also prepared a methodology using EEBAT in Agilator framework. The steps are given below:

1. Loading Dataset, a collection of Agile based projects of six software houses in Pakistan.

2. Data Set Preprocessing

2.1 Normalization of Dataset

2.2 Perform Principal Component Analysis (PCA) on Dataset to analyze features and labels

2.3 Perform feature selection using output of PCA

2.4 Box-Cox transformation of dataset

3. Data Set partitioning

3.1 Splitting transformed dataset into train and test data.

4. Create a base fuzzy system in training

4.1 Random initialization of Membership Function parameters.

4.2 Pass initial parameters to EEBAT Module.

4.2.1 Optimize the fitness function (Error function, RMSE/MSE)

4.2.2 Perform optimality analysis based on least error

4.2.3 Store the optimal parameters in a vector

4.3 Iteratively check for optimal parameters over epochs

4.4 Add new, optimized parameters to the fuzzy system's foundation.

4.5 Fit model on training data

4.6 Calculate MMRE adecsnd PRED for training

5. Perform AGILATOR testing

5.1 Apply optimized parameters obtained during training to set Membership function parameters

5.2 Calculate MMRE and PRED for testing

6. Tabulate the inferences for comparative analysis of predicted and actual effort.

7. Reiterate from step 4-step 5 and append learning gain, G using (3)

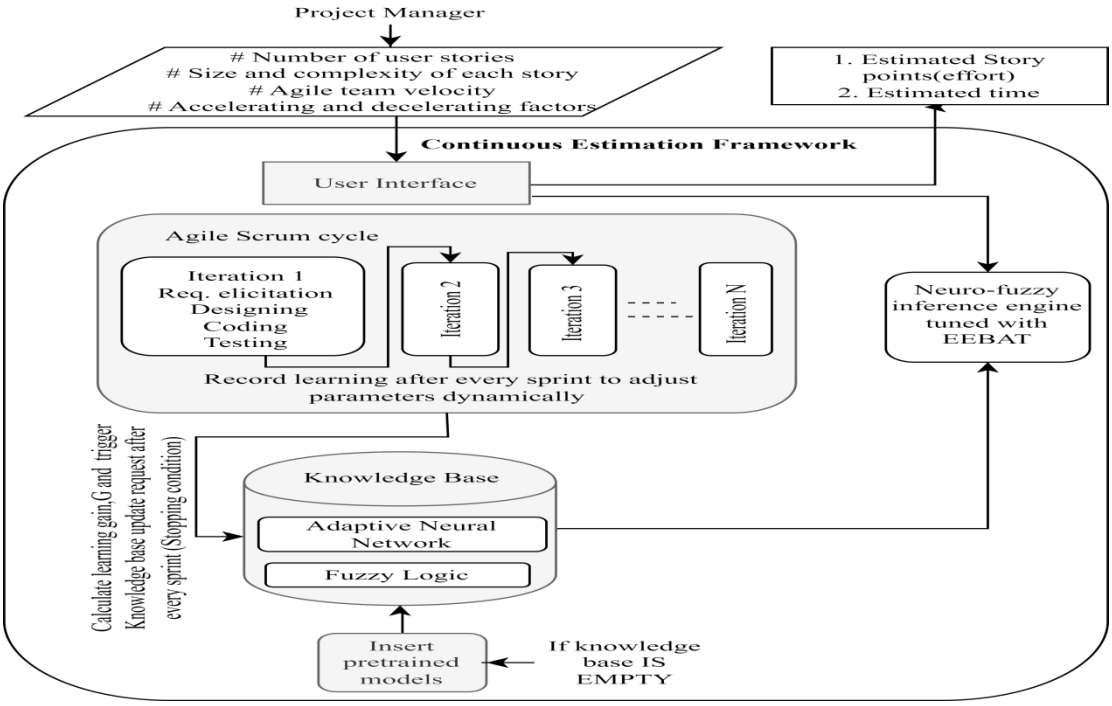

Fig. 1 Agilator framework

4. Conclusion and future scope

The Agilator expert system serves as a comprehensive solution for enhancing project management in Agile-inspired projects. Various IT stakeholders can utilize it as a tool for decision support, actively participating during the procedure of estimation and aiding in closing the gap between actual and estimated effort. We do not propose replacing traditional estimation techniques; instead, our system aims to boost confidence in estimating a Scrum project. The initial effort will be fine-tuned after every sprint using our innovative learning gain concept. As the project progresses, each iteration will document lessons learned, enabling on-the-fly adjustments to parameters. This expert system is poised to establish the groundwork for continuous estimation in the agile environment.

Refrences

[1] M. Arora, S. Verma, Kavita, M. Wozniak, J. Shafi, and M. F. Ijaz, "An efficient ANFIS-EEBAT approach to estimate effort of Scrum projects," Sci. Rep., vol. 12, no. 1, pp. 1–14, 2022, doi: 10.1038/s41598-022-11565-2.

[2] M. Choetkiertikul, H. K. Dam, T. Tran, T. T. M. Pham, A. Ghose, and T. Menzies, "A deep learning model for estimating story points," IEEE Trans. Softw. Eng., vol. 14, no. 8, pp. 1–12, 2018, doi: 10.1109/TSE.2018.2792473.

[3] A. Kaushik, D. K. Tayal, and K. Yadav, "A Comparative Analysis on Effort Estimation for Agile and Non-agile Software Projects Using DBN-ALO," Arab. J. Sci. Eng., vol. 45, pp. 2605–2618, 2020, doi: 10.1007/s13369-019-04250-6.

[4] M. Arora, A. Sharma, S. Katoch, M. Malviya, and S. Chopra, "A State of the Art Regressor Model's comparison for Effort Estimation of Agile software," 2nd Int. Conf. Intell. Eng. Manag., pp. 211–215, 2021, doi: 10.1109/ICIEM51511.2021.9445345.

[5] M. Choetkiertikul, H. K. Dam, T. Tran, T. T. M. Pham, A. Ghose, and T. Menzies, "A deep learning model for estimating story points," IEEE Trans. Softw. Eng., vol. 45, no. 7, pp. 637–656, 2019, doi: 10.1109/TSE.2018.2792473.

[6] R. G. F. Soares, "Effort Estimation via Text Classification and Autoencoders," in International Joint Conference on Neural Networks (IJCNN), 2018, vol. 2018-July, pp. 1–8, doi: 10.1109/IJCNN.2018.8489030.

[7] P. Pospieszny, B. Czarnacka-Chrobot, and A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms," J. Syst. Softw., vol. 137, pp. 184–196, 2018, doi: 10.1016/j.jss.2017.11.066.

[8] V.-S. Ionescu, H. Demian, and I.-G. Czibula, "Natural Language Processing and Machine Learning Methods for Software Development Effort Estimation," Stud. Informatics Control, vol. 26, no. 2, pp. 219–228, 2017, doi: 10.24846/v26i2y201710.

[9] Ziauddin, S. K. Tipu, and S. Zia, "An Effort Estimation Model for Agile Software Development," Adv. Comput. Sci. its Appl., vol. 2, no. 1, pp. 314–324, 2012.

[10] S. Porru, A. Murgia, S. Demeyer, M. Marchesi, and R. Tonelli, "Estimating Story Points from Issue Reports," in Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering, 2016, pp. 1–10, doi: 10.1145/2972958.2972959.

[11] C. Ratke, H. H. Hoffmann, T. Gaspar, and P. E. Floriani, "Effort Estimation using Bayesian Networks for Agile Development," in 2nd International Conference on Computer Applications and Information Security, ICCAIS 2019, 2019, pp. 1–4, doi: 10.1109/CAIS.2019.8769455.

[12] S. M. Satapathy and S. K. Rath, "Empirical assessment of machine learning models for agile software development effort estimation using story points," Innov. Syst. Softw. Eng., vol. 13, no. 2–3, pp. 191–200, 2017, doi: 10.1007/s11334-017-0288-z.

[13] T. Binish, L. Guzman, and U. M. Engel, "Effort estimation in agile software development : Case study and improvement framework," J. Softw. Evol. Process, pp. 1–14, 2017, doi: 10.1002/smr.1862.

[14] S. Basri, N. Kama, F. Haneem, and S. A. Ismail, "Predicting effort for requirement changes during software development," in Proceedings of the Seventh Symposium on Information and Communication Technology - SoICT '16, 2016, pp. 380–387, doi: 10.1145/3011077.3011096.

[15] A. Panda, S. M. Satapathy, and S. K. Rath, "Empirical Validation of Neural Network models for Agile Sooftware Effort Estimation based on Story Points," in 3rd International Conference on Recent Trends in Computing, 2015, pp. 772–781.

[16] L. R. Nerkar, "Software Cost Estimation using Algorithmic Model and Non-Algorithmic Model a Review," Int. J. Comput. Appl., vol. 0975–8887, pp. 4–7, 2014.

[17] M. Arora, S. Verma, and Kavita, "An efficient effort and cost estimation framework for Scrum Based Projects," Int. J. Eng. Technol., vol. 7, no. 4.12 Special Issue 12, 2018.

[18] A. T. Raslan, N. R. Darwish, and H. A. Hefny, "Towards a Fuzzy based framework for effort estimation in agile software development," Int. J. Comput. Sci. Inf. Secur., vol. 13, no. 1, pp. 37–45, 2015.

[19] S. J. . Forney, "An Agile Success Estimation Framework for Software Projects," 2019.

[20] N. Gaffar, H. Moussa, A. Kamel, and G. H. Galal-edeen, "A Proposed Framework for Enhancing Story Points in Agile Software Projects," Indian J. Sci. Technol., vol. 11, no. 31, pp. 1–11, 2018, doi: 10.17485/ijst/2018/v11i31/128780.

[21] T. M. Morakot Choetkiertikul , Hoa Khanh Dam , Truyen Tran , Trang Pham , Aditya Ghose, "A deep learning model for estimating story points," IEEE Trans. Softw. Eng., vol. 45, no. 7, pp. 637–655, 2019, doi: 10.1007/978-3-030-11928-7_88.

[22] S. M. Satapathy, A. Panda, and S. K. Rath, "Story Point Approach based Agile Software Effort Estimation using Various SVR Kernel Methods," in IEEE, 2013, pp. 3–6.

[23] M. Fu and C. Tantithamthavorn, "GPT2SP: A Transformer-Based Agile Story Point Estimation Approach," IEEE Trans. Softw. Eng., no. March, pp. 1–1, 2022, doi: 10.1109/tse.2022.3158252.

[24] H. Phan and A. Jannesari, "Story Point Effort Estimation by Text Level Graph Neural Network," 2022, [Online]. Available: http://arxiv.org/abs/2203.03062.

[25] H. Kassem, K. Mahar, and A. Saad, "Software effort estimation using Hierarchical Attention Neural Network," J. Theor. Appl. Inf. Technol., vol. 100, no. 18, pp. 5308–5322, 2022.