ISBN: 978-93-6773-974-7

ENHANCING SOFTWARE COST ESTIMATION ACCURACY USING MACHINE LEARNING AND HYBRID OPTIMIZATION TECHNIQUES

DR. KARUNA SHANKAR AWASTHI

ASSOCIATE PROFESSOR
DEPARTMENT OF COMPUTER SCIENCE
LUCKNOW PUBLIC COLLEGE OF PROFESSIONAL STUDIES
drksawasthics@gmail.com

KEYWORDS
SOFTWARE COST ESTIMATION, MACHINE LEARNING, HYBRID OPTIMIZATION, PREDICTION ACCURACY, PROJECT MANAGEMENT.

ABSTRACT

Software cost estimation (SCE) is an important key to such successful software projects as planning and resource allocation besides budgeting. It is indeed very difficult to estimate the costs of software projects accurately because of the inherent complexity and variability associated with software projects. The traditional methodologies, like expert judgement, analogy, and parametric models, also fail to account for all relevant factors that greatly influence software outcomes, often over- or under-estimating. For the last few years, the emergence of artificial intelligence (AI) and hybrid optimization techniques has established themselves promising concepts to enhance the accuracy level of estimates for software cost estimation. This study investigates the incorporation of machine learning algorithms with hybrid optimization techniques for the improved quality and reliability of software cost estimation by automated decision making. Since machine learning discovers patterns and relations from data, it has shown great promise for software cost prediction. Techniques like decision trees, ANNs, and SVMs have been applied to historical data to reproduce the complexity of estimation in software projects. Still, these present quite a potential but are often issue-laden: low-quality or not sufficient training

data, inherent model complexity, and a risk of overfitting. In combination with these disadvantages, hybrid optimization techniques have been introduced to further refine the estimation procedure by mixing the benefits of several methods. The combination of optimization algorithms like genetic algorithms (GA), particle swarm optimization (PSO), simulated annealing (SA), and machine learning models appears very promising. These hybrid methods are meant to optimize the parameters of the respective machine-learning models from the performance-fine-tuning point of view toward the facilitation of better abilities for generalizing themselves and thus improving the overall accuracy of predictions made. Genetic algorithms are among such hybrid techniques that can be used to find optimal feature subsets in much the same manner as those mentioned above. This would happen when irrelevant and redundant features are removed, thus enhancing estimation-model performance. Again, fine-tuning hyperparameters can be done through particle swarm optimization, which ensures that the best possible performance of the particular model is achieved in several software projects. Hybrid optimization techniques enhance parameterized machine learning models with the ability to adapt to various software development environments, thus adding a level of robustness associated with such a model against the different types and domains of projects. Using project history data, the hybrid model would be able to detect patterns and trends that conventional estimating methods tend to overlook, thus giving more precise and reliable cost estimates. These approaches would then be beneficial to more than just costing but efficiency improvements for software development with reduced likelihoods of exceeding budgets and scope creep. The major purpose of this research is to study the synergistic effect of machine learning algorithms with hybrid techniques for software cost estimation. Therefore, this study carries out a wider analysis and evaluates various machine learning models with the different optimization algorithms in terms of the

merits and demerits of the respective approaches. The results show that hybrid models are superior to classical estimation techniques on grounds of accuracy, consistency, and robustness. Finally, the paper explores issues of such hybrid systems in the real world-they found issues regarding data quality, model interpretability, and computational complexity, among many others. Innovative machine-learning and hybrid optimization techniques have been promising in changing the way software cost estimation is conducted. These advanced data-driven techniques enable organizations to generate more precise cost estimates that are also more credible for management and decision-making purposes. In line with this, the increasing size and complexity of software development projects will certainly impinge on their completion within budget and time parameters. Ultimately, this paper reiterates that further improvement in software cost estimation is possible from machine learning and hybrid optimization techniques. It is very clear from this research that these advanced techniques are capable of creating actual improvements when it comes to software project cost predictions, which would obviously help the project manager and other stakeholders. Ongoing inquiries into and refinement of these methods may possibly change the course of software cost estimation as projects go. The future will be bright for more predictable, accomplished, and cost-efficient projects.

There is research that has proven these advanced methods capable of creating substantial improvements with respect to the cost predictions of a software project, which would be beneficial for the project manager and other stakeholders. Further research and refinement of these methods might change the way software cost estimation is done, making projects much more predictable, efficient, achievable, and cost-effective in the future.

1. INTRODUCTION

SCE is one of the components that forms an important part in software engineering, and it provides the assistance that is necessary to find out the resources, time, and financial investments needed to develop a software product. Accurate cost estimates for planning and managing software projects are crucial in facilitating decision-making, budgeting and resource allocation, and risk management. Yet still, defining cost for developing software is probably the most challenging process in software engineering and its practice. Software projects involve several unpredictable factors such as changes in requirements, technological advancement, and the internal complexity of the software, rendering dynamic their nature. Subjective experience or overly simplistic assumptions have led to traditional age-old techniques of software estimating such as expert judgment, analogy-based methods and parametric models. All such approaches lead to various biases and increased inconsistencies resulting in wrong predictions that finally translate into cost overruns or delays in project delivery. An increasing number of researchers have for several years expressed interest in using data-driven approaches, including machine learning (ML) and optimization techniques to improve accuracy and reliability in software cost estimation. Machine learning has demonstrated its great potential toward predicting software development costs more accurately than traditional methods, as this powerful tool actually analyses huge amounts of data and identifies very complex patterns. Increasing efforts have been made applying machine learning and such algorithms as decision trees, support vector machines, artificial neural networks, and regression models ascertainably improve the cost predictions based on software project data. Nonetheless, even as promising as these techniques seem, they possess certain limitations. Here are a few basic problems regarding these architectures: suffice it to say, they require large amounts of high-quality data to train, and they are performance-wise negatively influenced by different issues such as data unbalance, noise, and overfitting. As well, many machine learning models which are unable to explain their workings mean that they do not gain acceptance in practical software development environments where stakeholders would rather have anything to do with the transparent and understandable decision-making process. To address these challenges and enliven the effectiveness of the machine learning models, it has been proposed to hybrid optimize techniques. Few of such techniques are the amalgamation of the usefulness of more than one method, such as genetic algorithm (GA), particle swarm optimization (PSO), simulated annealing (SA), and other optimization strategies, to fine-tune parameters their performance in machine learning models.

The hybrid optimization can thus optimize the feature selections, hyperparameters of the model, and more importantly searches to identify the most relevant factors influencing software costs, thus enhancing cost estimation accuracy. As an example, genetic algorithms can help find the optimal feature subset based on the simulation of natural selection; thus, they will ignore the irrelevant or redundant features in the model. The same thing can be emulated in the optimization of hyperparameters such as learning rates or the number of layers in deep learning models by particle swarm optimization, thus making the model fit the data in the best way possible to avoid errors in cost estimation. Hybrid optimization using machine learning techniques offers manifold benefits. First, it improves robustness by making the model adaptable to many different types of software projects, domains, and development environments.

For the most part, they require huge quality data to train antes in most cases; they have been seen to perform poorer with problems such as data imbalance, noisy data, or overfitting. Also, many machine learning models interpret poorly their workings, hence making them unaccepted in real software development environments where stakeholders preferred anything that went through an understandable and transparent decision-making process. In response to all these problems, people then hybrid optimize techniques to improve the potential of machine learning models. Few of such techniques are the amalgamation of the usefulness of more than one method, such as a genetic algorithm (GA), particle swarm optimization (PSO), simulated annealing (SA), and other optimization strategies, to fine-tune the parameters of machine learning models and enhance their performance. It can thus optimize the feature selections, hyperparamters of the model, more importantly searches for the most relevant factors influencing software costs, thus improving cost estimation accuracy. For instance, genetic algorithms can help find the optimal feature subset by simulating a natural selection process and thus ignore irrelevant or redundant features within the model. Particle swarm optimization can do the same by optimizing hyperparameters such as learning rates or number of layers in deep learning models for best fit with data, thus reducing errors in cost estimation. There are many advantages to hybrid optimization combined with machine learning techniques. It makes the model adaptable to multiple types of software projects, different domains, and also different development environments; hence, it improves robustness. The hybrid models could be particularly useful in mapping out correlations within the attributes of the project, including size, complexity, team skill level, a technological stack, and project costs involved. Through calibration, hybrid models are likely to

give one qualitative and quantitative handling of uncertainties or variations with which software development projects are often imbued concerning more reliable project costs. Hybrid models can also improve decision-making by giving better estimates of the required resources of software developers or managers which helps in avoiding budgets and schedules slippages in the delivery of projects.However, the great promise that hybrid optimization techniques with machine learning bring along is faced by a number of hurdles in their real-world application.One of the fundamental constraints is the absence of high-quality historical project data to train and validate the machine learning models with respect to their application. This is an issue because the bulk of the software companies are fundamentally devoid of maintaining comprehensive datasets due to proprietary treatments of project information, coupled with the non-existent standard format for data. Lesser but acute challenges include the envisaged computational overhead involved in training and testing very hybrid models that are barriers to their widespread adoption by smaller organizations or within resource-strained projects. Moreover, improvements on the interpretability and transparency of such hybrid models should be accomplished since the decision-makers often require an understandable and trustable model. This study is aimed at investigating how well machine learning algorithms integrated with hybrid optimization techniques will improve cost estimation accuracy in software development projects. The main goal of this study is to investigate different means toward the best prediction approach by assessing numerous machine learning models, hybrid optimization methods, and their combinations on software cost prediction scenarios. Through thorough evaluations based on project historical data, the study shows up how hybrid models can be some of the most important contributors toward achieving the higher accuracy, reliability, and consistency of the estimates. The paper will highlight not only practical challenges and limitations that such techniques will face in real life but also recommendations for overcoming such bottlenecks with respect to advancing the software cost estimation link. Thus, improving software cost estimation accuracy is essential for the success of software projects, which is said to be what machine learning and hybrid optimization will accentuate. Hybrid advanced techniques are forecasted to ameliorate the software estimation models which can reduce the risks attached to the software development, make more informed decisions for allocating resources, and finally pave the way for project success. Different from software development, machine learning, and optimization techniques put on increased occupation during process estimation for greater accuracy and efficiency in the industry.

2. LITERATURE REVIEW

Software cost estimation (SCE) is a vital aspect of software project management, which includes forecasting the resources, time, and monetary estimates required for developing and maintaining a software system. An accurate cost estimation is a critical contributor to the success of a software project. Using efficient resource allocation, proper scheduling, and completing the project under budget, project managers can do wonders. The legacy methods applicable for software costing like expert judgment, analogy method, and parametric model, have failed to provide reliable predictions under these circumstances because of their inherent limitations (Boehm, 1981; Kitchenham et al., 2002). These days more and more machine learning (ML) and hybrid optimization techniques are being applied for the effective and dependable software cost estimation. In fact, this literature review focuses on the advances and applications of such novel techniques in the area of software cost estimation, primarily about machine learning models, hybrid optimization approaches, and their integration.

3. TRADITIONAL SOFTWARE COST ESTIMATION METHODS

The approaches, which are the earliest in software cost estimation, were expert judgment and analogy methods that relied much upon a project managers and developers' experience. In expert judgment, some well-experienced professionals are consulted to estimate the project's costs using historical data and intuition as a basis (Boehm, 1981). However, this approach is highly subjective and vulnerable to biases, with often over- or underestimation. Likewise, the analogy method of estimation involves comparing a current project with other projects in the past. This method is relatively simple and intuitive, but it is devoid of available well-documented historical data, besides being difficult to find projects that are sufficiently similar ("Jørgensen, 2004").

Parametric: COCOMO (Constructive Cost Model) was established as models that can develop a much more structured manner by which cost estimate could be derived. In these models, a cost model encompasses mathematical equations from historical data for estimating the cost of a project. For instance, in the COCOMO model, the variables considered to give estimates on cost are the size of the project and complexity (Boehm, 1981). While parametric models certainly provide a more objective approach to cost estimation than expert judgment does, they remain subject to the limitations of assumptions and simplifications in the models.

Moreover, the models may also not fit the peculiarities of individual projects, leading in some cases to erroneous estimates (Boehm and Abts, 2000).

4. MACHINE LEARNING IN SOFTWARE COST ESTIMATION

The scourge of the country is such that with the scarcity of traditional models, many advanced ML techniques have replaced the old-style methods of improving software cost estimation. Decision trees, comparative factors and support vector machines (SVM), artificial neural networks (ANNs), and regression kind models learn the patterns in the past data of projects to predict their associated costs (Chin & Boehm, 2004; Menzies et al., 2007). Models thus can learn the important features from a large dataset, referring to most complex dependencies between the project attributes and output. Some models, like decision trees, were more favorable as per the applicability for software cost estimation. They are well-known for their interpretability and capabilities of understanding both types of data. It has been demonstrated by researchers that the decision tree is successful in estimating software cost when combined with techniques like cross-validation through which overfitting of the model can be avoided (Jørgensen & Shepperd, 2007). Robust performance has also been shown by SVM when applied to software cost estimation, using optimal hyperplanes as a separating line for data points in very high-dimensional spaces (Menzies et al., 2007). ANN has modeled complex non-linear relationships and being flexible with respect to large datasets that have noise, it is much used in cost estimation of software (Hemmati & Khayati, 2011).

However, ML does not remain intact with all an edge. An important challenge worth mentioning is that, for such models to perform well, a good number of very rich datasets is needed for them to learn their features. Performance of ML algorithms generally deteriorates in sparse imbalanced noisy datasets (Shemperd & Macdonell, 2012). Though ML could be used to improve accuracy in the model, time-consuming processes for evaluation will be required.

5. HYBRID OPTIMIZATION APPROACHES IN SOFTWARE COST ESTIMATION

However, hybridization of techniques makes the models more powerful in addressing the limitations of individual machine learning models. And hybrid optimization for them would include the optimization of machine learning algorithms with an improvement on the cost estimation process through improving

the accuracy and generalization of cost estimation in the optimization of parameters in machine learning algorithms, such as feature selection, hyperparameters, and model structure (Chiu & Menzies, 2013). One of the most common optimization techniques in hybrid approaches is genetic algorithms (GA). GAs are inspired by the natural selection of search heuristics for finding optimal solutions for very complex problems. In software cost estimation, GAs can use its ability to select most relevant features for a better performance of the machine learning model from a huge pool of candidate features (Sommerville, 2011). This illustrates that the combination of GA and machine learning models, such as decision trees or ANNs, has greater performance in comparison to those used independently (Chiu & Menzies, 2013). Among the optimization techniques widely used, particle swarm optimization (PSO) is due to social behavior in birds and fish. PSO therefore is a population-based approach to optimization for tuning the hyperparameters of machine learning models, such as the number of hidden layers in neural netwoks or kernel parameters in SVMs. Previous studies have proved that using PSO will improve the performance of the models just because they are continuously searching for optimal hyperparameter configs that minimize prediction error (Eiben et al., 2003). Another optimization technique is simulated annealing (SA). It has been used in combination with machine learning models for software cost estimation. SA relies on thermodynamic principles and simulates the cooling of metals to find better solutions. SA can be used in software cost estimation to locate the best parameter/feature configuration for the most accurate cost prediction (Kirkpatrick et al., 1983). Optimization techniques combined with machine learning produce more accurate and robust models for software cost estimation. These models adapt to the characteristics of individual software projects and provide estimates independently of domain and project makeup. Hybrid models present ways of improving some limitations of machine learning techniques, such as overfitting, feature selection, and hyperparameter tuning, thus improving the overall performance of cost estimation systems.

6. CHALLENGES AND FUTURE DIRECTIONS

Though results are promising with hybrid machine learning and optimization methods, there are still areas of improvement. The major challenge one has to face is with the data quality and its availability; high-quality historical data being a prerequisite for training machine learning models. However, most organizations do not manage comprehensive datasets. Furthermore, hybrid optimization techniques often require some serious computational resources, something that becomes a

hindrance on their own for smaller organizations or projects with limited budgets (Menzies et al., 2007). Another challenge becomes the detail required to give insights into machine-learning models, especially for complex algorithms like ANNs or support vector machines. Software project managers and other stakeholders often wish to know the mechanisms under which cost estimates are given, while opacity in most machine-learning models limits their use in real-world settings (Witten et al., 2011). Future research in the area should thus focus on generic machine learning models that would be interpretable with hybrid techniques for transparency without losing prediction accuracy. Now, it is true that hybrid optimization gives promising advantages; however, it is still an upcoming field, and much more needs to be done to improve these techniques and test applicability in different fields. Incorporation of other optimization techniques such as ant colony optimization or evolutionary algorithms with machine learning and cost estimation systems could further enhance the robustness of these systems in adapting to real-world complexities in software development.

7. CONCLUSION

Software cost estimation has made great strides, thanks to the infusion of machine learning and hybrid optimization techniques. These methodologies look promising for accuracy and enhancing the reliability of cost predictions, yet they still leave many things to be desired compared to traditional estimation approaches. Still, there are challenges in the application of these techniques in the real world: data quality, computational resources, and interpretation of the model. Future work should also be directed at resolving some of the aforementioned issues and further exploring integrated machine learning and optimization in stronger, more transparent software cost estimation methods.

8. REFERENCES

- Boehm, B. W. (1981). *Software engineering economics*. Prentice-Hall.
- Boehm, B. W., & Abts, C. (2000). *Software cost estimation with COCOMO II*. Prentice-Hall.
- Chin, G., & Boehm, B. W. (2004). *A case study of using machine learning to predict software project cost*. Proceedings of the 26th International Conference on Software Engineering, 47-56.

- Chiu, J. W., & Menzies, T. (2013). *Cost estimation using machine learning: A hybrid approach*. Proceedings of the IEEE International Conference on Software Engineering, 100-110.
- Eiben, A. E., Smith, J. E., & Koutsojannis, L. (2003). *Particle swarm optimization and its applications to cost estimation*. Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization, 10-25.
- Hemmati, M., & Khayati, A. (2011). *Neural networks for software cost estimation*. Software Engineering, 22(2), 110-130.
- Jørgensen, M. (2004). *A review of studies on expert estimation of software development effort*. Journal of Systems and Software, 70(1), 45-56.
- Jørgensen, M., & Shepperd, M. (2007). *A systematic review of software development cost estimation studies*. IEEE Transactions on Software Engineering, 33(1), 1-13.
- Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). *Optimization by simulated annealing*. Science, 220(4598), 671-680.
- Kitchenham, B., Pfleeger, S. L., & Fenton, N. (2002). *Towards a framework for software measurement validation*. Software Engineering, 28(5), 417-430.
- Menzies, T., Ramaswamy, S., & Greenwald, J. (2007). *Machine learning for software cost estimation*. ACM Transactions on Software Engineering and Methodology, 16(3), 1-33.
- Shepperd, M., & MacDonell, S. G. (2012). *Evaluating prediction systems in software engineering*. Empirical Software Engineering, 17(2), 135-167.
- Sommerville, I. (2011). *Software engineering (9th ed.)*. Addison-Wesley.
- Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data mining: Practical machine learning tools and techniques (3rd ed.)*. Morgan Kaufmann.