International Research Journal of Advances in Computing Sciences, Vol 1 (Issue 1), pages 1-8, 2023

Comparative Analysis of Neural Network Architectures for Software Cost Estimation

Mehmood Ahmed*, Adeel Ahmed

Department of Information Technology, The University of Haripur, KPK, Pakistan

ABSTRACT

Software cost estimation is a critical aspect of project management, with significant implications for decision-making and resource allocation. In recent years, the use of neural networks (NNs) has shown promise in improving the accuracy of cost estimation models. This research, unique in its systematic investigation of various NN architectures, aims to determine their strengths and weaknesses in the context of cost estimation. The study focuses on three primary neural network architectures: feedforward neural networks (FNNs), recurrent neural networks (RNNs), and convolutional neural networks (CNNs). It scrutinizes each architecture's ability to capture and model the complex relationships within software development datasets, with a particular emphasis on their performance in terms of prediction accuracy, scalability, and adaptability to diverse project scenarios. By providing a comprehensive understanding of the nuanced dynamics between neural network structures and their effectiveness in software cost estimation, this review aims to offer practical insights that can guide practitioners and researchers in selecting the most suitable neural network architecture for specific cost estimation challenges. Additionally, it identifies potential areas for future research, thereby offering a roadmap for advancing the application of neural networks in software cost estimation.

Keywords: Neural Network Architectures, Software Cost Estimation, Comparative Analysis, Feedforward Neural Networks, Prediction Accuracy

1 INTRODUCTION

Software cost estimation is pivotal in managing software development projects, influencing crucial aspects such as budgeting, resource allocation, and project planning. While traditional models like COCOMO and function points have been widely used, their effectiveness in addressing modern software projects' dynamic and complex nature is often limited [1, 2]. As the field of software development continues to evolve, there is an increasing demand for more adaptive and accurate estimation methods. This research, by providing a comprehensive comparative analysis of various neural network architectures for software cost estimation, offers practical insights that can significantly enhance the accuracy and adaptability of cost estimation methods in real-world project scenarios.

The advent of artificial intelligence and machine learning offers promising avenues for enhancing software cost estimation. Among these technologies, neural networks (NNs) stand out due to their ability to model complex, non-linear relationships inherent in software projects [3]. Neural networks, inspired by biological neural networks, comprise layers of interconnected nodes capable of learning patterns from [4].

This paper compares various neural network architectures, including feedforward neural networks, recurrent neural networks (RNNs), and convolutional neural networks (CNNs), to evaluate their applicability and performance in software cost estimation. Each architecture offers unique characteristics: feedforward neural networks provide a straightforward approach to pattern recognition; RNNs are suited for sequential data processing, making them ideal for projects with temporal dynamics; and CNNs excel in identifying spatial hierarchies in data, which could be leveraged for analyzing complex project features [5].

Through this analysis, the paper seeks to identify the strengths and limitations of each NN architecture in capturing the nuances of software cost estimation, thereby guiding practitioners and researchers towards the most effective models for their specific needs. Furthermore, this study contributes to the ongoing discourse on integrating machine learning techniques in software engineering, highlighting potential areas for future research and development.

In summarizing the evolution of software cost estimation and the rise of neural networks within this domain, this introduction sets the stage for a detailed exploration of how these advanced computational models can revolutionize estimation practices, aligning with the broader trend of employing machine learning to tackle complex problems in software engineering [6].

* Corresponding author email: mehmood@uoh.edu.pk

2 BACKGROUND

2.1 Software Cost Estimation Models

Cost estimation in software engineering is a pivotal process integral to project management and planning. Numerous models have evolved to accurately forecast the resources, time, and budget required to complete software projects. Traditional cost estimation models can be broadly categorized into expert judgment, algorithmic models, and Function Point Analysis.

Expert judgment relies on the insights and intuition of seasoned professionals who draw upon their extensive experience with similar projects. This approach values the tacit knowledge and heuristics individuals develop over time, enabling them to make informed guesses about project costs [7]. While expert judgment is highly flexible and can quickly adapt to new information, its accuracy is heavily dependent on the experience level of the individuals involved and can suffer from biases and inconsistencies.

Algorithmic models, such as the Constructive Cost Model (COCOMO), offer a more structured approach. These models use historical project data and mathematical formulas to estimate costs, considering factors like project size, complexity, and team capability [8]. COCOMO, for instance, classifies projects into different categories and applies specific formulas to each, allowing for a more nuanced estimation process. However, the effectiveness of algorithmic models can be limited by the availability and relevance of historical data, as well as their inherent assumptions about project characteristics.

Function Point Analysis evaluates the software's functional components, such as inputs, outputs, user interactions, files, and data complexity, to estimate the effort required [9]. This model shifts the focus from lines of code to the functionality delivered, offering a more direct measure of the software's accomplishments. While Function Point Analysis can be more closely aligned with customer and user needs, it requires a detailed understanding of the software's functionality and can be subject to interpretation variance.

Each of these traditional models has its strengths and offers valuable insights into the cost estimation process. However, they often struggle with modern software projects' dynamic and non-linear nature, which can involve rapidly changing requirements, emerging technologies, and varied development methodologies.

2.2 Introduction to Neural Networks

Neural Networks (NNs) are computational models that draw inspiration from the human brain's structure and function. Comprising interconnected units or nodes (neurons), NNs process data and generate outputs based on the inputs received [10]. The feedforward neural network's simplest form consists of an input layer, one or more hidden layers, and an output layer, with data flowing in a single direction [11]. This structure allows complex relationships between inputs and outputs to be modelled without cyclic or looped connections.

Recurrent Neural Networks (RNNs) introduce the concept of memory into neural networks by allowing information to persist through loops, making them particularly suited for sequential data processing, such as time series analysis or natural language processing [12]. Convolutional Neural Networks (CNNs), on the other hand, are designed to learn spatial hierarchies of features from input images automatically and adaptively, making them powerful tools for image recognition and processing tasks [13].

These architectures enable NNs to model complex, non-linear relationships within data, a precious capability in domains where traditional linear models fall short.

2.3 Relevance of Neural Networks in Cost Estimation

Incorporating NNs into software cost estimation marks a significant advancement in the field. The complex, non-linear nature of software projects, characterized by intricate dependencies and varying scales, poses a challenge to traditional estimation models. NNs, with their ability to model such non-linear relationships and adapt to evolving data patterns, emerge as a potent solution [14]. Studies have shown that NNs can outperform traditional models in cost estimation by providing more accurate and reliable predictions [15]. This improvement is attributed to NNs' capacity to process and learn from large datasets, uncovering patterns and relationships that may not be evident through conventional analysis [16].

Integrating NNs into cost estimation practices signifies a shift towards more adaptive, data-driven methodologies. This transition is aligned with the broader adoption of AI and ML across various domains, underscoring the potential of NNs to redefine software cost estimation [17]. With their versatility and the increasing ease of access to advanced computational resources, NNs stand as a promising avenue for future exploration and development in this area [18].

As highlighted in this paper, the comparative analysis of NN architectures feed forward, RNNs and CNNs within software cost estimation underscores the nuanced dynamics between neural network structures and their applicability to the challenges of accurately estimating software project costs. This investigation not only sheds light on the strengths and weaknesses of each architecture but also offers practical insights for selecting the most suitable NN model for specific estimation challenges, paving the way for further advancements in the application of neural networks in software cost estimation.

3 METHODOLOGY OF THE REVIEW

This section delineates the methodology for conducting a thorough literature review about applying neural network architectures in software cost estimation. It encompasses a detailed literature search strategy, clearly defined selection criteria, and a robust approach to data synthesis, ensuring a comprehensive and systematic review of the existing research.

3.1 Literature Search Strategy

The literature search was executed across several esteemed academic databases and digital libraries, including IEEE Xplore, Science Direct, Google Scholar, and the ACM Digital Library, to amass various studies relevant to neural network architecture and their application in software cost estimation [19, 20]. Keywords and phrases such as "Neural Networks in Software Cost Estimation," "Artificial Neural Network (ANN) Models in Software Engineering," "Feedforward Neural Networks (FNNs)," "Recurrent Neural Networks (RNNs)," "Convolutional Neural Networks (CNNs)," and "Comparative Analysis of Neural Network Architectures" were meticulously selected and combined with Boolean operators for refined search results. Manual searches were conducted within select journals and conference proceedings to ensure no pertinent study was overlooked [21].

3.2 Selection Criteria

The literature was filtered based on predefined inclusion and exclusion criteria to ensure the selection of studies of high relevance and quality. The inclusion criteria mandated that studies be published in peer-reviewed journals or conference proceedings, written in English, focused on specific neural network architectures (FNNs, RNNs, and CNNs), and applicable to software cost estimation [19, 22]. Conversely, exclusion criteria eliminated non-peer-reviewed articles, papers not in English, studies not directly addressing neural network applications in cost estimation, and outdated research not reflective of current advancements in neural network technologies [19]. This meticulous selection process also involved assessing the methodological rigor of studies and their direct relevance to the comparative analysis goal of this review [19, 23].

3.3 Approach to Data Synthesis

The information of the selected papers was synthesized through a multistep process. Initially, key data points such as the neural network architecture utilized, application context in software cost estimation, methodology, results, and conclusions were extracted and organized in a tabular format for preliminary comparative analysis [23, 24]. Following this, a thematic analysis was undertaken, categorizing findings from the studies based on similarities and differences in approaches, results, and implications. This facilitated the identification of predominant themes, trends, and research gaps [19, 23, 24]. The synthesis aimed to integrate insights across studies, comparing the efficacy and applicability of different neural network architectures in software cost estimation, providing a nuanced understanding of the field [25].

4 NEURAL NETWORK ARCHITECTURES FOR COST ESTIMATION

The application of various neural network architectures in software cost estimation represents a significant advancement in predicting project costs more accurately and dynamically. This section examines the roles and effectiveness of Feedforward Neural Networks (FNNs), Recurrent Neural Networks (RNNs), and Convolutional Neural Networks (CNNs) in software cost estimation.

4.1 Feedforward Neural Networks (FNNs)

Feedforward Neural Networks (FNNs) are foundational to artificial neural network designs, characterized by a unidirectional data flow from the input to the output layers, without feedback loops [26]. FNNs' structure comprises an input layer representing cost-driving factors, one or more hidden layers for data processing, and an output layer for cost prediction. Their capacity to model complex, non-linear relationships between input variables and outputs renders FNNs particularly suitable for software cost estimation, where project data is often multidimensional and varied [27]. The primary advantage of FNNs in this domain lies in their ability to learn from and adapt to historical project data, facilitating refined and accurate cost predictions based on identified patterns [28].

4.2 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) distinguish themselves by loops within their architecture, allowing for the retention of information across sequences in their internal state or memory [29]. This feature is invaluable for software cost estimation tasks involving sequential or time-series data, such as project timelines or iterative development phases [29, 30]. RNNs provide a dynamic perspective on cost estimation, analyzing the impact of previous project stages on overall costs. Despite their advantages, RNNs encounter issues like the vanishing gradient problem, which impedes learning over long sequences. This challenge has led to the development of advanced variants like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) networks, designed to surmount these limitations [30, 31].

4.3 Convolutional Neural Networks (CNNs)

While Convolutional Neural Networks (CNNs) are primarily recognized for their prowess in image processing and computer vision, their adaptability has seen them employed in diverse domains, including software cost estimation [32]. CNNs feature convolutional layers that learn spatial hierarchies of features from input data, making them adept at processing and analyzing patterns within multidimensional project data [33]. In software cost estimation, CNNs can examine data matrices representing various project attributes, identifying complex interrelationships and patterns [34]. Their ability to discern spatial and hierarchical patterns offers profound insights into cost determinants and their interplay, significantly enhancing prediction accuracy in software projects [35].

5 COMPARATIVE ANALYSIS

5.1 Comparative Criteria

Evaluating neural network architectures for software cost estimation involves several crucial criteria influencing their effectiveness and suitability for specific project requirements. These criteria are central to understanding the strengths and limitations of each architecture within the context of cost estimation.

Accuracy: This measures the precision of the network's cost estimations, typically assessed using metrics like Mean Squared Error (MSE) and Mean Absolute Percentage Error (MAPE). Accuracy is paramount, as even minor errors can significantly impact project budgeting and resource allocation.

Learning Ability: This criterion evaluates the efficiency of an architecture learning from training data. It encompasses the convergence speed to an optimal solution and the model's ability to mitigate overfitting, ensuring it generalizes well to new, unseen data.

Scalability: Scalability evaluates a model's ability to manage datasets of different sizes and project complexities, ensuring it can adapt to increasing data volumes and more intricate requirements while maintaining performance.

Robustness: Robustness measures a model's consistency and resistance to data noise. It ensures reliable estimations across diverse conditions and scenarios by maintaining performance despite anomalies.

Complexity Handling: This criterion evaluates an architect's proficiency in modelling non-linear relationships among cost factors, which is essential for accurate software project cost estimations and capturing the complexity of development tasks.

5.2 Performance Analysis

The analysis of performance across neural network architectures reveals distinct characteristics and capabilities of each, as outlined in the table below.

Table 1: Performance Characteristics of Neural Network Architectures

Architecture	Strengths	Challenges
FNNs	Efficient in straightforward tasks, fast learning with small datasets	Performance plateaus with complex/large datasets

RNNs (LSTM, GRU)	Excellent at capturing temporal dependencies, suitable for sequential data	Higher computational and training resource requirements
CNNs	Potent in identifying patterns in multidimensional data, scalable	It may add unnecessary complexity to simple tasks

5.3 Cross-Architecture Insights

The analysis highlights adaptability, data requirements, and the trade-off between complexity and performance, guiding the choice of neural network architecture for tasks.

Table 2: Cross-Architecture Insights

Criterion	FNNs	RNNs	CNNs
Adaptability to Complexity	Low	High	Medium
Data Requirement for Training	Low	High	High
Complexity vs. Performance	Favourable for simple tasks	Balanced for complex tasks	Complex, may overfit on simple tasks
Generalization Capability	Moderate	High	High

Adaptability to Project Complexity: RNNs, particularly LSTM and GRU, are superior in managing complex, time-sensitive project data. In contrast, FNNs are better suited for less dynamic projects with well-defined cost drivers.
Data Requirement and Training: FNNs require less data for effective training, making them a practical choice for projects with limited historical data. RNNs and CNNs, however, need extensive datasets to learn and capture underlying patterns accurately.
Model Complexity vs. Performance Trade-off: While CNNs and RNNs can model more intricate relationships, they may not always surpass FNNs in more straightforward estimation scenarios due to the potential for overfitting.
Generalization Capability: CNNs and RNNs generally exhibit superior generalization across diverse and extensive datasets, an essential feature for scalable and adaptable cost estimation models in software development.
The selection of a neural network architecture for software cost estimation should thus be carefully considered, considering the project data's nature, estimation complexity, and available resources for model training and deployment [36].

6 DISCUSSION

6.1 Synthesis of Findings

The comparative analysis of neural network architectures for software cost estimation uncovers distinct advantages and appropriateness depending on various project scenarios. Feedforward Neural Networks (FNNs) have shown efficiency and quick learning with smaller, more straightforward datasets, positioning them as a dependable option for straightforward cost estimation tasks. However, their performance doesn't necessarily improve at the same rate as the complexity of the dataset [37].
Recurrent Neural Networks (RNNs), particularly the Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) variants, are adept at managing sequential data, providing a dynamic method for cost estimation in projects with time-sensitive elements [38]. Despite their superior capabilities, implementing RNNs requires higher computational requirements and a complex training phase.
Convolutional Neural Networks (CNNs) are recognized for their effectiveness in analyzing multidimensional data and identifying complex patterns potentially overlooked by other models. Nevertheless, given their original development for spatial data

interpretation, their deployment in cost estimation might bring unnecessary complexity for projects with more straightforward demands [39].

6.2 Implications for Practice

This study's outcomes hold substantial implications for existing software cost estimation methodologies. The selection of a neural network architecture must be customized to the project's specific data characteristics and requirements. RNNs present the most refined and precise estimates for projects enriched with temporal or sequential information. Conversely, FNNs could offer adequate accuracy for projects with static and straightforward parameters, requiring less computational effort.

CNNs' application in cost estimation might be especially beneficial for extensive projects characterized by complexly interacting multidimensional data. However, employing such sophisticated models requires judicious consideration of accuracy versus model complexity and the sufficiency of training data to leverage their capabilities thoroughly.

6.3 Limitations and Challenges

Despite the promising advantages of neural networks in software cost estimation, several limitations and challenges persist in the research and application phases. A primary concern is the adequacy and quality of data necessary for training these networks to attain high precision and generalization. Accessing extensive datasets for training can be particularly challenging in proprietary or sensitive project settings [40].

Another limitation of neural network models is their interpretability. Contrary to traditional estimation models, neural networks, especially those based on deep learning, tend to function as "black boxes," complicating the understanding of how specific inputs affect outputs. This opacity can be problematic when illustrating the basis for cost estimations, which is as crucial as the estimations themselves.

Moreover, the computational resources required to train and implement advanced neural network models can be significant, restricting their application in smaller enterprises or projects with constrained IT infrastructure [41].

These insights emphasize the importance of a balanced approach when choosing and applying neural network models for cost estimation, considering the accuracy benefits and the practical limitations and necessities of the specific project scenario.

Table 3: Comparative Overview of Neural Network Architectures

Architecture	Strengths	Weaknesses	Best Use Case
FNNs	Quick learning, efficient with small datasets	Limited scalability with complex data	Straightforward tasks with well-defined parameters
RNNs (LSTM/GRU)	Excellent at sequential data, dynamic estimations	High computational demand, complex training	Projects with temporal data or sequential processes
CNNs	Effective in multidimensional pattern recognition	Potentially excessive complexity for simple tasks	Large-scale projects with complex data interactions

7 CONCLUSION

This review has meticulously analyzed and compared various neural network architectures, Feedforward Neural Networks (FNNs), Recurrent Neural Networks (RNNs), and Convolutional Neural Networks (CNNs) in the nuanced field of software cost estimation. The analysis revealed that FNNs are particularly suited for straightforward and less complex projects, thanks to their simplicity and effective pattern recognition capabilities. However, their performance tends to plateau with the increase in dataset complexity. On the other hand, RNNs, with a focus on LSTM and GRU variants, excel in processing sequential or time-series data, proving invaluable for projects influenced by temporal dynamics. CNNs, known for their adeptness at handling multidimensional data, excel in identifying complex patterns across various data layers. However, their application is most beneficial in projects where the complexity and nature of data justify their use. The core insight derived from this comparative study emphasizes that the appropriateness of a neural network architecture for cost estimation tasks heavily depends on the project's specific requirements, including the nature of the data, its complexity, and the computational resources at hand.

This review significantly contributes to the evolving domain of software cost estimation by offering a comprehensive comparison of neural network architectures and shedding light on their applicability and potential to improve estimation practices. It not only provides practitioners and researchers with a clearer understanding of the strengths and limitations associated with each neural network type but also highlights the importance of selecting the most suitable model based on the specific needs of their projects. The findings underscore a trend towards adopting more sophisticated, data-driven approaches in software cost estimation, advocating for continuous exploration and adaptation of these technologies, and addressing the challenges related to data requirements, model interpretability, and computational demands. As the integration of neural networks in software cost estimation becomes increasingly refined, it holds the promise of leading to more accurate and efficient project planning and management within the software industry, marking a significant stride towards leveraging AI to enhance traditional estimation methodologies.

REFERENCES

[1] Boehm, B. W.: Software Engineering Economics. IEEE Transactions on Software Engineering 10(1), 4–21 (1981).

[2] Albrecht, A. J.: Measuring Application Development Productivity. In Proc. Joint SHARE, GUIDE, and IBM Application Development Symposium, 83–92 (1979).

[3] Hinton, G. E., Osindero, S., Teh, Y. W.: A Fast-Learning Algorithm for Deep Belief Nets. Neural Computation 18(7), 1527–1554 (2006).

[4] LeCun, Y., Bengio, Y., Hinton, G.: Deep Learning. Nature 521(7553), 436–444 (2015).

[5] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016).

[6] Zhang, H., Babar, M. A., Tell, P.: Identifying Relevant Studies in Software Engineering. Information and Software Technology 53(6), 625–637 (2011).

[7] Mahmood, Y., Kama, N., Azmi, A.: A Systematic Review of Studies on Use Case Points and Expert-based Estimation of Software Development Effort. Journal of Software: Evolution and Process 32(7), e2245 (2020).

[8] Khan, B., Khan, W., Arshad, M., Jan, N.: Software Cost Estimation: Algorithmic and Non-algorithmic Approaches. Int. J. Data Sci. Adv. Anal. 2(2), 1–5 (2020).

[9] Villalobos-Arias, L., Quesada-López, C.: Comparative Study of Random Search Hyper-Parameter Tuning for Software Effort Estimation. In Proceedings of the 17th International Conference on Predictive Models and Data Analytics in Software Engineering, 21–29 (August 2021).

[10] Wu, L., Cui, P., Pei, J., Zhao, L., Guo, X.: Graph Neural Networks: Foundation, Frontiers, and Applications. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 4840–4841 (August 2022).

[11] Kumaraswamy, B.: Neural Networks for Data Classification. In Artificial Intelligence in Data Mining, 109–131. Academic Press (2021).

[12] Manchev, N., Spratling, M.: Target Propagation in Recurrent Neural Networks. The Journal of Machine Learning Research 21(1), 250–282 (2020).

[13] Xia, W., Yin, H., Dai, X., Jha, N. K.: Fully Dynamic Inference with Deep Neural Networks. IEEE Transactions on Emerging Topics in Computing 10(2), 962–972 (2021).

[14] dos Santos, R. A., Vieira, D., Bravo, A., Suzuki, L., Qudah, F.: A Systematic Mapping Study on the Employment of Neural Networks on Software Engineering Projects: Where to Go Next? Journal of Software: Evolution and Process 34(3), e2402 (2022).

[15] Tayefeh Hashemi, S., Ebadati, O. M., Kaur, H.: Cost Estimation and Prediction in Construction Projects: A Systematic Review on Machine Learning Techniques. SN Applied Sciences 2, 1–27 (2020).

[16] Otchere, D. A., Ganat, T. O. A., Gholami, R., Ridha, S.: Application of Supervised Machine Learning Paradigms in the Prediction of Petroleum Reservoir Properties: Comparative Analysis of ANN and SVM Models. Journal of Petroleum Science and Engineering 200, 108182 (2021).

[17] He, X., Liu, R., Anumba, C. J.: Data-driven Insights on the Knowledge Gaps of Conceptual Cost Estimation Modeling. Journal of Construction Engineering and Management 147(2), 04020165 (2021).

[18] Ahmed, M., Ibrahim, N. B., Nisar, W., Ahmed, A., Junaid, M. et al.: A Hybrid Model for Improving Software Cost Estimation in Global Software Development. Computers, Materials & Continua 78(1), 1399–1422 (2024).

[19] Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., Linkman, S.: Systematic Literature Reviews in Software Engineering - A Systematic Literature Review. Information and Software Technology 51(1), 7–15 (January 2009), doi: 10.1016/j.infsof.2008.09.009.

[20] Ahmed, M., Ibrahim, N., Nisar, W., Ahmed, A.: Systematic Literature Review on Global Software Development Based Software Cost Estimation Models and Cost Drivers. Indonesian Journal of Electrical Engineering and Computer Science 32(3), 1485–1494 (2023).

[21] Dickersin, K., Scherer, R., Lefebvre, C.: Systematic Reviews: Identifying Relevant Studies for Systematic Reviews. BMJ 309(6964), 1286–1291 (1994).

[22] Kumpulainen, S.: Artificial General Intelligence: A Systematic Mapping Study (2021).

[23] Stapić, Z., Antonio, G., López Cabot, E. G., Ortega, L. M., Strahonja, V.: Performing Systematic Literature Review in Software Engineering. In Central European Conference on Information and Intelligent Systems, 441–493 (2012), doi: 10.1145/1134285.1134500.

[24] Putro, B. L., Rosmansyah, Y., Suhardi: An Intelligent Agent Model for Learning Group Development in the Digital Learning Environment: A Systematic Literature Review. Bulletin of Electrical Engineering and Informatics 9(3), 1159–1166 (June 2020), doi: 10.11591/eei.v9i3.2009.

[25] Konda, S. R., Shah, V.: Machine Learning-Enhanced Software Development: State of the Art and Future Directions. INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 6(4), 136–149 (2022).

[26] Hagen, M. T., Demuth, H. B., Beale, M.: Neural Network Design. PWS Publishing Co. (1996).

[27] Rao, B. T., Dehuri, S., Mall, R.: Functional Link Artificial Neural Networks for Software Cost Estimation. International Journal of Applied Evolutionary Computation (IJAEC) 3(2), 62–82 (2012).

[28] Rijwani, P., Jain, S.: Enhanced Software Effort Estimation Using Multi-Layered Feed Forward Artificial Neural Network Technique. Procedia Computer Science

89, 307–312 (2016)

[29] Salehinejad, H., Sankar, S., Barfett, J., Colak, E., Valaee, S.: Recent Advances in Recurrent Neural Networks. arXiv preprint arXiv:1801.01078 (2017).

[30] Wang, D., Fan, J., Fu, H., Zhang, B.: Research on Optimization of Big Data Construction Engineering Quality Management Based on RNN-LSTM. Complexity, 2018 (2018).

[31] Alkinani, M. H., Khan, W. Z., Arshad, Q.: Detecting Human Driver Inattentive and Aggressive Driving Behavior Using Deep Learning: Recent Advances, Requirements and Open Challenges. IEEE Access 8, 105008–105030 (2020).

[32] Turarbek, A., Bektemesov, M., Ongarbayeva, A., Orazbayeva, A., Koishybekova, A., Adetbekov, Y.: Deep Convolutional Neural Network for Accurate Prediction of Seismic Events. International Journal of Advanced Computer Science and Applications 14(10) (2023).

[33] Liu, J., Zhang, K., Wu, S., Shi, H., Zhao, Y., Sun, Y., ..., Fu, E.: An Investigation of a Multidimensional CNN Combined with an Attention Mechanism Model to Resolve Small-Sample Problems in Hyperspectral Image Classification. Remote Sensing 14(3), 785 (2022).

[34] Chirra, S. M. R., Reza, H.: A Survey on Software Cost Estimation Techniques. Journal of Software Engineering and Applications 12(6), 226 (2019).

[35] Ravì, D., Wong, C., Deligianni, F., Berthelot, M., Andreu-Perez, J., Lo, B., Yang, G. Z.: Deep Learning for Health Informatics. IEEE Journal of Biomedical and Health Informatics 21(1), 4–21 (2016).

[36] Hegazy, T., Ayed, A.: Neural Network Model for Parametric Cost Estimation of Highway Projects. Journal of Construction Engineering and Management 124(3), 210–218 (1998).

[37] Reed, R., MarksII, R. J.: Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks. MIT Press (1999

[38] Haartveit, A., Husum, H.: Learning Event-Driven Time Series with Phased Recurrent Neural Networks. Master's Thesis, NTNU (2018).

[39] Ortac, G., Ozcan, G.: Comparative Study of Hyperspectral Image Classification by Multidimensional Convolutional Neural Network Approaches to Improve Accuracy. Expert Systems with Applications 182, 115280 (2021).

[40] Basheer, I. A., Hajmeer, M.: Artificial Neural Networks: Fundamentals, Computing, Design, and Application. Journal of Microbiological Methods 43(1), 3–31 (2000).

[41] Lippmann, R. P.: Pattern Classification Using Neural Networks. IEEE Communications Magazine 27(11), 47–50 (1989).