Hindawi
IET Software
Volume 2023, Article ID 4324783, 19 pages
https://doi.org/10.1049/2023/4324783

Research Article

A Data-Driven Artificial Neural Network Approach to Software Project Risk Assessment

Mohammed Naif Alatawi,[1] **Saleh Alyahyan**,[2] **Shariq Hussain**,[3] **Abdullah Alshammari**,[4] **Abdullah A. Aldaeej**,[5] **Ibrahim Khalil Alali**,[6] **and Hathal Salamah Alwageed**[7]

[1]*Information Technology Department, Faculty of Computers and Information Technology, University of Tabuk, Tabuk, Saudi Arabia*
[2]*Applied College in Dwadmi, Shaqra University, Shaqra, Saudi Arabia*
[3]*Department of Software Engineering, Foundation University Islamabad, Islamabad, Pakistan*
[4]*College of Computer Science and Engineering, University of Hafr Al-Batin, Hafar Al-Batin 31991, Saudi Arabia*
[5]*Department of Management Information Systems, College of Business Administration, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia*
[6]*Department of Instructional Technology, Jouf University, Sakaka, Saudi Arabia*
[7]*College of Computer and Information Science, Jouf University, Sakaka, Saudi Arabia*

Correspondence should be addressed to Shariq Hussain; shariq@fui.edu.pk

Received 14 May 2023; Revised 11 October 2023; Accepted 25 November 2023; Published 19 December 2023

Academic Editor: Nadeem Sarwar

In the realm of software project management, predicting and mitigating risks are pivotal for successful project execution. Traditional risk assessment methods have limitations in handling complex and dynamic software projects. This study presents a novel approach that leverages artificial neural networks (ANNs) to enhance risk prediction accuracy. We utilize historical project data, encompassing project complexity, financial factors, performance metrics, schedule adherence, and user-related variables, to train the ANN model. Our approach involves optimizing the ANN architecture, with various configurations tested to identify the most effective setup. We compare the performance of mean squared error (MSE) and mean absolute error (MAE) as error functions and find that MAE yields superior results. Furthermore, we demonstrate the effectiveness of our model through comprehensive risk assessment. We predict both the overall project risk and individual risk factors, providing project managers with a valuable tool for risk mitigation. Validation results confirm the robustness of our approach when applied to previously unseen data. The achieved accuracy of 97.12% (or 99.12% with uncertainty consideration) underscores the potential of ANNs in risk management. This research contributes to the software project management field by offering an innovative and highly accurate risk assessment model. It empowers project managers to make informed decisions and proactively address potential risks, ultimately enhancing project success.

1. Introduction

Initially and traditionally, time cost and quality were termed as the critical factors for the project's success. Any project completed within these constraints was termed as a successful project and successful management but as the passage of time, the constraints have had a paradigm shift from these to customers' demands, stakeholder's affirmation, and future possibilities [1]. So, currently the focus has shifted to management skills as well.

With the evolvement of knowledge and passage of time, project risk management (RM)has appeared as a critical tool for project-based organizations' success rates. However, despite this awareness, projects when classified as successful and unsuccessful, the ratio is nearly the same for both [2]. The CHAOS [3] stated that 36% of the projects were conveyed on time with

required peculiarities and capacities, 45% were confronted, which were late or over-budgeted, and the rest 19% failed, which were canceled before completion [4]. The projects not classified as successful usually fail in different aspects of management i.e., escalating from constraints, not coping up to the scope, or hazardous occurrence of the unexpected events. All these failure aspects of project management revolve around one planet, RM.

Managing risk is an extremely important concern for all sorts of organizations, including high fraction of software houses. Risk, as obvious it is needs to be mitigated in all domains of organizations whether it is financial, technical, environmental, or political. In the concern of software industry, risk can be interpreted as potential to success or failure of the project [5]. Provided these concepts, the project objectives can be achieved through identifying and evaluating and finally responding to the triggers that generate the risks. These estimations are a critical point to enhance the probability of the success of the project. In the case of IT projects, the failure factors include incomplete information in the initiation phase of the project, time and cost escalations [2].

Risk: project risk is defined in PMBOK as: "an uncertain event or condition that, if it occurs, has a positive or negative effect on one or more project objectives such as scope, schedule, cost, or quality" (PMI, 2013).

RM is trending as the most important and essential procedure among project management activities yet basic terminologies of the implementation are often misinterpreted or mixed even being completely different concepts. Knight (1921) differentiated the two most unclear and similar concepts: risk and uncertainty [3, 4].

(1) Risk: risks are defined as external or internal quantifiable events that might have negative or positive impact on the life cycle of the project if occurred [5].
(2) Uncertainty: uncertainty is referred to unknown and unquantifiable events that may and may not be foreseen [5].

Majority of the risks are identifiable and quantifiable. Through proper management their impact can be minimized and coped up but uncertainties unlike risks are unquantifiable.

This paper will solely focus on the evaluation of hard risks.

The identification, building strategies to control and monitor later is known as RM. Literature has defined this as "a structured approach to administrate (analyse, evaluate, and control) risks" (Chapman et al., 1996).

RM is the most accustomed and critical procedure used in the project management processes. It is a continuous process having multiple phases typically meaning to apply methodologies, techniques, and strategies to mitigate, avoid or prepare for the risk expected to occur. Several studies show several steps involved in the RM process with respect to their context, however, basic steps of RM are identify, analyze, plan, handle, monitoring, and control. Traditionally, RM wherever performed is done qualitatively majorly. The project managers or focal persons use various typical techniques like Delphi Method, Swift analysis, Decision tree, or matrices for the RM certainly not very successful for assessing risks properly.

The initial phase, identification; involves identification of internal or external risk through planned strategy. It varies with respect to the nature of project and the more stakeholders involved; the more precise risk identification becomes. After identification, second phase i.e., analyze refers to using prior information, to extract when, where, and why risk exists in a project and to estimate its impact through thorough communication with the stakeholders or evaluating the probability that an unwanted event can occur in future. Third stage, planning includes defining and selecting strategies to decide whether to avoid or mitigate the risk or adapt any contingency plan to minimize the impact. Any strategy adopted is directly related to the nature and expected impact of the risk and varies with the nature of the projects. The planned strategy is then executed to handle the risk if it occurs and is termed as the fourth phase of the RM process. After execution of the strategy comes the fifth phase; monitoring. This is an on-going process, monitors the status of the project development mapping against the risk expected. The performance indicators are used to check the procedure in case of deviations from the risk mitigation plan, the contingency plan is executed immediately to fix the problem. The deviations might be identified through the performance indicators and this final stage is termed as control (Victor et al., 2007).

Unlike qualitative risk analysis, quantitative analysis operates on the numeric data and depicts the numeric estimates to somewhat predict the probability of project success/failure and yet develop contingency plans. The quantitative risk analysis is required to assess the overall project risk and the whole of resources while on the contrary; qualitative risk focuses over the individual risks only. The multiple tools and techniques for quantitative risk analysis are three point estimates, decision tree analysis, expected monetary value, montecarlo analysis, sensitivity analysis, and fault tree analysis. As for the management of this critical phase of project development, many researchers have worked and contributed to the RM strategies and methodologies to the date. The method has centered on identifying risks, ranking them, estimating their effects, and figuring out how much work would be needed to implement the best management plan [6]. Techniques like Monte Carlo simulation, fuzzy logic, hierarchical analytical analysis (AHP), failure mode and effect analysis (FMEA) have been trending for the risk assessment. (Thompson et al., 1992; Abdelgawad and Fayek [6]; Sadeghi et al., 2010).

An information processing technology of the artificial intelligence; artificial neural networks (ANNs) introduced have emerged as the data-driven decision-making technique for multiple purposes. ANNs were described as technology that imitates the human brain, using the terminologies of nervous system i.e., neurons. He explained them as networks that can be trained to behave as humans, learn from the previous events and extract essential information from input

provided to them. A neural network typically contains three layers, an input, a hidden layer, and an output layer. The number of hidden layers may vary depending upon the nature of prediction [7]. The input layer receives the initial data forwarded to the hidden layer where the data are analyzed and thus signal, i forwarded to the output layer which ultimately become next input for the next iteration. The nodes/neurons in the input layer receives input say $x1, x2, \ldots, xn$. A weight value to each node is assigned $(w1, w2, \ldots, wn)$ which in the hidden layer are processed as the summation $\sum xiwi$. The output suppose y is thus calculated as follows:

$$y = f(\sum xiwi). \tag{1}$$

The nodes of the output layer are perceived as the results.

Neural networks have been used by Victor et al. (2007) to identify the risk resources. For high-tech investment projects, Jiang (2009) used ANNs for the assessment of risks and RM models, Ahmad et al. [8] used ANNs in National Iranian Oil Products Distribution Company (NIOPDC) for the prioritization of risk. ANNs according to the research have been proved to have multiple applications in the risk assessment phase of software industry as well. Different faces of ANNs are being used with multiple other AI techniques for obtaining the most appropriate results [9, 10].

Statistics according to the CHAOS Report 2015 depict that only 15% of IT projects in Pakistan are successful rest of all resulting in exceeding budget and schedule, major financial losses, market value, time, and resources for the organizations. In literature, researchers have identified the reasons of failure of the IT projects in Pakistan, enlisted five reasons among which three were the consequences of inadequate risk assessment. Multiple RM techniques are in trend yet losses can be saved twice quickly if early and timely risk assessment is done. To overcome this problem, this paper will present a data-driven prediction technique for timely risk assessment, calculating the intensity and consequently allowing the project manager to estimate the effort required for the measures for risks confrontation.

For the purpose of risk assessment, many researchers have implemented artificial intelligence as a prediction technique of the total project risk, yet individual risks have not been forecasted throughout the literature. This research will forecast the intensity of total project risk and the intensity of individual risk factors; financial risks, schedule risks, user risks, performance risks, and complexity risks to locate the most frequently occurring types of risks in the software industry.

The algorithms developed for the prediction among the literature are hybrid algorithms; combination of two or more techniques to attain the accuracy in the prediction leading the algorithm complex, time taking for training and heavier to run on simpler systems. The research will use single algorithm for the forecasting purpose making it less costly, less time taking and easy to build and run. Additionally, the results will be displayed on graphs for the ease of understanding of the user

In this study, we provide empirical evidence through data-driven approach. Real data of the software houses from Pakistan will be used to predict the risks individually in the range of 0–1, making 0.5 as the cutoff point. The algorithm; ANNs, will enable the project managers to assess the project risk in the initiation phase of SDLC leading to timely management of risks.

Multiple hybrid tools of AI are now appearing in trend for the project RM praxis all over the world. Data-driven predictions for various processes in RM have come up as a cutting edge for success of the projects furthering the success of the organization in the past decade. Although use of artificial intelligence for RM purposes has been recognized in literature, yet no literature exists on the provision of empirical evidence using ANN as a prediction analysis in the IT business of Pakistan on real data leaving a major research gap. In this paper, we target the Software Houses of Pakistan to execute projects provided this RM methodology to assess, predict the impact, and prioritize the expected risk using their historical data. The objectives of this study are:

(1) The primary objective of this research is to develop an ANN model for accurately predicting and assessing project risks in the software industry.
(2) This study aims to identify the optimal ANN architecture, including the number of input and hidden layer neurons, to achieve the highest prediction accuracy.
(3) Another goal is to compare the performance of mean squared error (MSE) and mean absolute error (MAE) as error functions within the ANN model to determine which yields superior risk predictions.
(4) This research intends to predict the total project risk by leveraging historical data, including factors related to project complexity, financial aspects, performance metrics, schedule adherence, and user-related variables.
(5) In addition to overall project risk, this study seeks to predict individual risk factors, such as complexity risk, financial risk, performance risk, schedule risk, and user risk, to provide a comprehensive risk assessment.
(6) The research aims to validate the robustness of the developed ANN model by applying it to previously unseen data, confirming its accuracy and reliability.

The paper contributed to introduces a novel approach that leverages ANN for software project risk assessment. This contribution is significant as it explores the application of advanced machine learning techniques to address complex challenges within the software development life cycle. The study showcases the potential of data-driven ANN-based solutions in mitigating the project uncertainties. By using ANN, the paper offers a more data-driven and sophisticated method for risk assessment, which can provide more accurate and reliable risk predictions. The paper provides a comprehensive account of the data collection and preprocessing procedures used. This transparency reinforces the research's rigor by ensuring the quality and integrity of the collected data. The study identifies and analyzes specific risk factors

relevant to the software project management. This contributes to a better understanding of the risk landscape in the software projects.

The paper has been divided into eight sections. Section 1 contains introduction, Section 2 contains literature review, Section 3 contains industry trends in project risk management in Pakistan, Section 4 contains identification of risks, Section 5 contains proposed methodology, Section 6 contains analysis procedure for prediction of total project risk, Section 7 contains results and discussion, and Section 8 concluded the research.

2. Literature Review

2.1. Risk Management. Risk assessment practice requires two main phases, with three dependent phases each. Risk evaluation, the first basic phase, includes risk recognition, risk analysis, and risk prioritization: risk identification provides lists of project-specific risk objects that are likely to affect the performance of a project (Ahmed et al. 2020). Checklists, study of decision generators, comparison with practice (assumption analysis), and decomposition are common risk-identification methods. For each defined risk item, risk analysis assesses the loss likelihood and loss severity and assesses compound risks of risk-item encounters. Performance models, cost models, network analysis, mathematical decision analysis, and quality-factor analysis (such as reliability, availability, and security) are traditional techniques. A ranked ordering of the detected and evaluated risk items is generated by risk prioritization. Risk-exposure analysis, risk-reduction leveraging analysis (particularly including cost–benefit analysis), and Delphi or group-consensus techniques are common techniques. The second primary phase, RM, includes RM preparation, risk assessment, and risk monitoring: ask-management planning helps train you to handle each risk item (e.g., by knowledge purchase, risk prevention, risk transfer, or risk reduction), through communicating with each other and with the larger project strategy the actual risk item plans. Checklists of risk-resolution methods, cost-benefit analyses, and common sketches and aspects of the RM strategy are traditional techniques. Risk resolution provides a condition in which risk things are withdrawn or otherwise addressed (for example, risk avoidance via relaxation of requirements). Prototypes, simulations, benchmarks, task evaluations, key-personnel arrangements, methods to design-to cost, and gradual growth are common strategies. Risk monitoring entails monitoring the progress of the project toward the resolution of the risk items and, if applicable, taking corrective steps. Typical approaches include achievement monitoring and a top-10 list of risk issues that are outlined per weekly, monthly, or milestone project analysis on January 30, 1991 and adequately followed up with a reassessment of the risk item or corrective action. In comparison, risk assessment provides an improved method for the life cycle to be handled and coordinated. Risk-driven methods such as the software process spiral model: eliminate many of the problems found with the previous process models such as the waterfall model and the evolutionary model of growth. These risk-driven methods also explain how and when to implement emerging software technology, such as rapid prototyping, fourth-generation languages, and commercial software products into the life cycle.

2.2. Implementing the Risk Management. Trying to implement RM means inserting the ideas and methods of the RM into your current life-cycle management practices. The full application of RM entails the use of risk-driven software process models and the spiral model, where the total sequence of life cycle operations, the use of prototypes and other methods for the risk resolution are decided by the risk factors assessment techniques. The best implementation strategy, however, is an incremental one, which allows the culture of an organization to gradually adjust to risk-oriented management practices and risk-driven process models. A good way to start is to establish a process of top-IO risk-item tracking. It is quick and affordable to introduce, offers early enhancements and beeps to create a familiarity with the other concepts and procedures of RM. Once some RM expertise has been gained by the company on this initial project, successive steps will intensify the complexity of the RM strategies and extend their implementation to broader project groups. It is viewed in the form of a contractual software purchase, but it can also be tailored to the needs of a company for intimate growth. As an elaboration of the "why, what, when, who, where, how, how much" structure, the project manager should coordinate the life-cycle RM strategy. This strategy is largely the responsibility of the client; it is very beneficial to include the group of developers in its planning as well.

2.3. Importance of Risk Management in Software Projects. Software projects usually produce variable results being the high-risk activities and complex in nature, particularly susceptible to failure (Bannerman 2008). Hence, even when the measures for the RM are highly understood, one is not completely confident on the success of the project (Rodriguez-Repiso et al. 2007). Since the past decade, remarkable improvements have been achieved in the context however many software projects face delays in delivery, require more resources than planned and do not achieve the quality desired by the client (Barros et al. 2004). Among the major reasons of the failure of software Projects Charette (2005) mentioned poor RM as the top most area for the improvement. According to Dey et al. (2007), much more is required than just identifying the risks in the early stage for the success of software projects. Proper technique to quantify and manage the risks is the vital phase of RM (Jiang et al. 2001). According to Cooper (2003), a very basic and significant tool for RM is the previous knowledge, experience, and information. The way toward distinguishing and assessing risks of projects can be refined by a variety of procedures and approaches. Among the techniques are cited: regression analysis, expert systems, and stochastic models (Houston et al. 2001); influence diagram; Monte Carlo Simulation; program evaluation and review technique (PERT); sensitivity analysis; analytic hierarchy process (AHP); fuzzy set approach (FSA); neural networks; decision tree; and fault tree analysis; risk checklist; risk map; diagram of cause and effect; Delphi technique; combination of decision tree; and AHP (Dey and Ogunlana 2004).

3. Industry Trends in Project Risk Management in Pakistan

As a result of literature review on quantitative techniques for RM, a comprehensive online study was done, which pointed toward understanding the level of acknowledgment, penetration, and the use of RM process, strategies, and tools in the software development area explicitly in Pakistan.

In Pakistan; two basic approaches are in practice: Jehan et al. [11] conducted a research that 36% of their respondents (managers from software houses in Pakistan) could not assess or control risks in organizations. Out of the organization that assess and control the risks, 74% organizations handle risks when they occur, 14% address risks before they occur while 12% use both methodologies [12].

Of different other discoveries, some fascinating disclosures were uncovered as far as quantitative risk examination methods: study results, as introduced infer that 64% respondents use "Expert Judgment" and 44% use "Interviewing" as quantitative assessment methods. Respondents proposed "Brainstorming" as a procedure for quantitative examination. Likewise, 2% respondents try not to utilize any quantitative risk assessment strategies. All the examined strategies are valuable for decision-making, and all decision-makings require considering the risks of the other options. A few techniques yield precise results in quantification depending on the type of problem. Quantitative techniques, despite the fact of requiring more resources and being complex in nature than qualitative techniques are yet well analyzed, well accurate and detailed. The methods the implementation of certain techniques introduced for the arrangement of other genuine issues while the subsequent course implies refining the other quantitative strategies by presenting or potentially growing new or further developed advance technologies, keeping the understandability and convenience as a primary concern [13].

4. Identification of Risks

4.1. Financial Risk. The financial risk occurs where the cost of the project increases the expected cost. The risk of failure is due to over-running expenses. In a software project, cost risk results in a chain of other risks. The following are the illustration of factors that contribute to cost risks in a soft project

(1) Incorrect estimate of expenditures
(2) Overruns in costs due to underutilization of capital
(3) Sudden project scope extension
(4) Bad management of finances
(5) Unfeasible timeline
(6) Failure of hardware
(7) Lack of testing and recording
(8) Transition of technology
(9) Change in management
(10) Malfunctioning of instruments
(11) Schedule risk

Schedule management requires the required procedures to ensure the job is done on a well-timed basis. Risk and uncertainty influence the timeline of the project, thereby causing risk to the schedule. Risks in the timetable are the cause for delays in projects that could again lead to other risks. There is a risk that the planned preparations will not be fulfilled. Owing to the lack of tracking metrics and inadequate use of past estimates or documentation, a schedule management vulnerability refers to an excessively optimistic task scheduling. Weak coordination and management also contribute to the unregulated timetables and expenditures and a loss of flow in the work of project growth. The following example of risks in software project risk scheduling

(1) Specification extension and alteration
(2) Insufficient knowledge with tools and methodologies
(3) Technology change
(4) Improper training for personnel
(5) Individual weaknesses
(6) Slow Control Levels
(7) Alterations to the Environment
(8) Insufficient funding
(9) Lack of suitable information, aim specifications and resources

4.2. Performance Risk. Performance risk is the risk that a project encounters when it fails to satisfy or does not fulfill the project condition that justifies the end result to be accomplished by it. If there is a significant technical challenge raising the duration or costs of the project, success risk contributes to schedule risk and cost risk. The project's failure to offer the necessary production contributes to a risk of success. A risk of performance management is the result of a low-quality project implementation that happens again due to different reasons such as mistake or glitches, time loss and over budgeting problems. The following illustration of uncertainties in software project risk for results

(1) Lack of project quality specifications
(2) Lack of design documentation
(3) Unrealistic Timeline
(4) Loss of competence
(5) Insufficient understanding of tools
(6) Modifications in technologies
(7) Environmental alterations
(8) Extension to modifications to specifications
(9) Incomplete data

4.3. User Risk. The most cited risk is the lack of the user's involvement during project development phase. If the user has no interest in the creation of the project or no attempts are seen from the user's side, the project will fail. This user vulnerability emerges out of the mind-set and actions of the users during the project's device growth process. The

following example of hazards that cause user risk in software project

(1) Lack of coordination between users
(2) User's unwillingness to change
(3) Lack of users' dedication to the project
(4) Lack of consumer interaction
(5) Conflict among users
(6) The pessimistic mind-set of the user
(7) Negative attitude of users about the project

4.4. Complexity Risk. In spite of the difficulty of the project being carried out, the inherent uncertainty of a software project reflects another level of the risk of a software project. Each project passes through complications such as the use of higher technology versions, less user-friendly techniques, modern job schemes that need preparation, and modified working methods. The following example of the hazards that cause complexity risk in software project.

(1) The initiative requires the use of modern models of technologies not previously used.
(2) Wide number of interactions with other networks
(3) High technological difficulty standard
(4) Immature applications
(5) Automation of increasingly complex functions
(6) One of the organization's main projects

5. Proposed Methodology

The research approach is quantitative in nature and it ensures the authenticity and reliability of the sample information selected for this research. Five types of frequently occurring risks in the software projects have been shortlisted through the comprehensive literature review:

(1) Financial risks
(2) Schedule risks
(3) User risks
(4) Performance risks
(5) Complexity risks.

The data of these risks are collected from the risk registers obtained from the software houses, fulfilling the specified criteria. We collected the data for our study as members of a team assessing and predicting the risks in the upcoming software projects. Figure 1 shows the proposed methodology.

5.1. Sample and Data Collection. The unit of analysis for this study is the medium to large sized projects. We required the data from the risk registers obtained from the software houses. Value of each risk in the risk register was obtained categorically and five categories of risks were chosen with the help of the literature for the analysis. In Pakistan, a smaller number of software houses practice this activity so all the data we obtained were of around 120 projects.

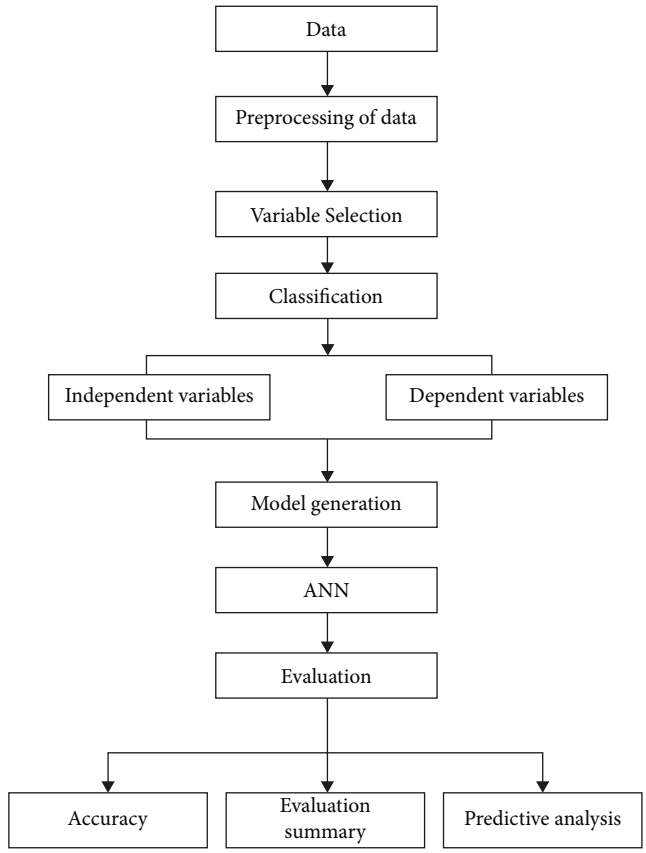

Figure 1: Proposed methodology.

Table 1: Dataset.

Project size	Effort	Months	Ideal staffing
Small	8–360	0–3	1 Person
Medium	361–3,600	3–6	3–7 Persons
Large	3,601–24,000	6–12	7–24 Persons

Purposive sampling has been used to identify those software houses or IT organizations (registered with Pakistan Software Export Board (PSEB)) that have been maintaining the risk registers for at least 5 years. Upon identification, a total of 18 mature organizations came up as the ones maintaining the registers, out of which, 10 organizations responded with their data for the analysis. As decided, data of medium to large sized projects were collected for analysis (size is defined as per kilo lines of code, monthly duration, and staffing). Data of around 18 medium to large sized projects was collected, analyzed and preprocessed for the prediction analysis. Table 1 shows the dataset description.

(1) Multivariate: the dataset includes multiple attributes, allowing for the examination of various risk factors simultaneously.

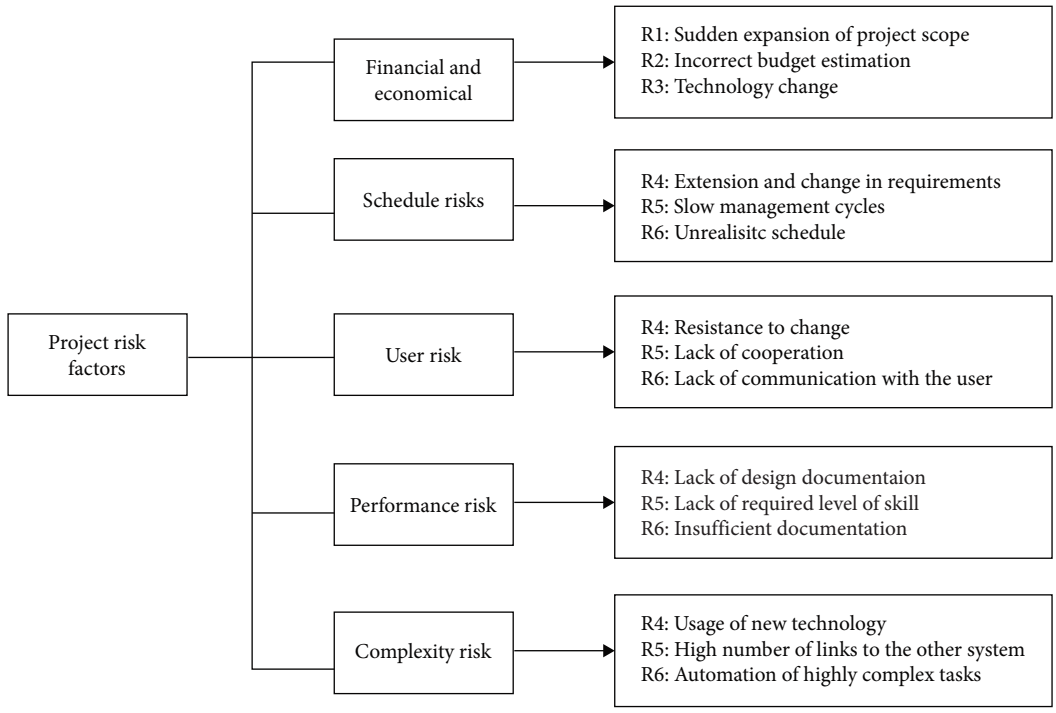

Figure 2: Classification of risk factors, reference from Malaya et al. 2020.

(2) Temporal: it captures historical project data, enabling the assessment of how risk factors have evolved over time.
(3) Heterogeneous: the dataset encompasses a wide range of risk factors, including both quantitative and qualitative attributes, such as project complexity scores, financial data, performance metrics, and categorical variables related to user factors.
(4) Real-world data: the dataset reflects real-world scenarios and challenges faced by software projects in the (mention the industry or domain) sector, providing practical insights into risk assessment.
(5) Imbalanced: the dataset may exhibit class imbalance, particularly in the case of risk categories, where some projects may have higher risk levels while others have lower risk levels. This imbalance should be addressed during preprocessing and modeling stages.
(6) Noisy data: as with most real-world datasets, there may be noise, missing values, or outliers in the dataset that require appropriate data preprocessing techniques.

The Preprocessing steps are as follow:
Data preprocessing steps:

Data Cleaning. The initial dataset underwent a rigorous data cleaning process to address issues such as missing values and outliers. Missing data points were identified, and appropriate strategies were employed to handle them. Common techniques, such as mean imputation or deletion of rows with missing values, were used depending on the nature and extent of missingness.

Outlier Detection and Handling. Outliers can significantly impact the performance of neural network models. Robust outlier detection methods, such as the Z-score or IQR (interquartile range), were applied to identify and address outliers. Depending on the nature of the outliers, they were either corrected, removed, or transformed to reduce their influence on model training.

Feature Selection. Feature selection techniques were employed to determine the most relevant attributes for the risk assessment task. This step involved analyzing the correlation between attributes, identifying redundant features, and selecting a subset of attributes that contributed significantly to the prediction of risk levels. Feature importance scores from the neural network models themselves were also considered in this process.

Normalization or Scaling. Neural networks benefit from having input data within a standardized range. Therefore, numerical attributes were normalized or scaled to have a consistent mean and standard deviation. Common techniques include Min–Max scaling or Z-score normalization.

Encoding Categorical Variables. As the dataset contained categorical variables (e.g., project types, user roles), these were encoded into numerical format using techniques like one-hot encoding or label encoding to ensure compatibility with the neural network models.

Training and Testing Split. To evaluate the ANN models effectively, the dataset was split into a training set (comprising 80% of the data) and a testing set (comprising the remaining 20%). This split ratio was chosen to ensure a robust evaluation of model performance.

Figure 2 shows the classification of risk factors, from (Malaya et al. 2020). These risk factors will be located in the risk registers.

Table 2: Profile of the sample collected.

S. no.	Name	URL	Duration	Budget	Staffing
1	The Society of Cable Telecom Engineers (SCTE)	https://www.scte.org/	2 Years	$100k	5 > 10
2	Litecure	https://www.litecure.com/	6 Months	$40k	5 > 15
3	Furbaby Tracker	Android app	4 Months	$20k	5 > 15
4	Primier Orthopedics	https://premierortho.com/	3 Months	$15k	10 > 15
5	Global Tax Managament	https://gtmtax.com/	4 Months	$25k	5 > 10
6	doctorcare247	https://doctorcare247.com/	1 Year	$80k	5 > 15
7	Whisper- Android app	app	3 Months	$12k	10 > 15
8	IsupportCause	https://www.isupportcause.com/	4 Months	$10k	5 > 10
10	PIB Group	https://www.pibgroup.co.uk/	4 Months	$10k	10 > 15
11	E-pal	https://www.epal.gg/	8 Months	$60k	10 > 15
12	Tintabudi	tintabudi.com	5 Months	$30k	5 > 15
13	buyon.pk	buyon.pk	4 Months	$15k	5 > 10
14	Datavis	Datavis.sg	6 Months	$8k	10 > 15
15	peymynt financial	peymynt.com	2 Years	$2.3M	5 > 10
16	AnygivenSunday	anygivensuday.co	1 Year	$10k	10 > 15
17	Ikigawi	Ikigawi.com	7 Months	$7k	10 > 15
18	Boop	https://www.awwwards.com/sites/boop	4 Months	$15k	5 > 10

5.2. Data Collection Tool

5.2.1. The Risk Register. For this study, we have collected and processed the information obtained from the risk registers. Risk register is a document maintained by the project-based organizations containing all the risks involved in a project categorically. During development of a project, each risk with its unique ID, title, category, assigned owner, its likelihood and impact, causes and treatment of the risk is logged in. Additional information might be fuzed into a risk register, mentioning documentation of existing controls for request to help with monitoring their operational advantage and effectiveness with application and viability, the risk status (e.g., open, close, increasing or decreasing, and so forth) to help with following the general risk profile, the sort of risk and related mishaps (for example security, money related, notoriety, legitimate, and so on), and the objective risk level.

The document contains this information of all the projects that have been executed by the organization and help in future decision-making as well (Whipple and Pitblado, 2010). Following are the major advantages of maintaining a risk register:

(1) Help to identify potential behavioral patterns or environmental concerns.
(2) Identify and assess the magnitude of risks that may be subject to legislation or business changes.
(3) Exhibit to stakeholders (controllers, speculators, organization partners, and others) that dangers are being overseen.
(4) Configuration controls or relief measures to decrease or eliminate the risk(s) before they happen,
(5) Report better safe work methods.

Nationally and internationally, stable organizations including software houses maintain the register for better performance. However, in Pakistan, not all software houses practice this. In this study, we target the IT industry of Pakistan, identify the software houses that are operating for more than 5 years and maintain the risk registers.

5.2.2. Profile of the Sample Collected. Data of the following projects were collected from the risk registers. The risk register is a confidential property of the organization; most of the IT firms we approached hesitated to provide their register to us, yet though links effort, a few software houses provided us with data of their projects. As the study is of prediction instead of generalization, 18 projects seemed enough to fulfill the purpose. Multiple researches have been conducted over hypothetical data, small dataset and auto generated data for the prediction purpose but one of the aims for this study was to use real data for prediction. For reference, Hong Haa et al. (2018) conducted a data-driven prediction through multilayered feed forward network using the data of 15 construction projects, achieving 98% accuracy. Table 2 is the profile of the data including their ideal staffing, budget and duration. Table 2 shows the Profile of the Sample collected.

5.2.3. Descriptive Analysis. In Figure 3, descriptive analysis in terms of duration is depicted. Ten projects were completed between 3–6 months, four projects were between 6 months to 1 year, and four projects exceeded 12 months. Similarly, in Figure 4 analysis in terms of staffing is depicted.

In six projects, 5–10 staffing was involved, in eight projects, 10–15 developers were included, and four projects included more than 15 people to work in.

6. Analysis Procedure for Prediction of Total Project Risk

The procedure of assessing the risk is divided into five major phases as explained under:

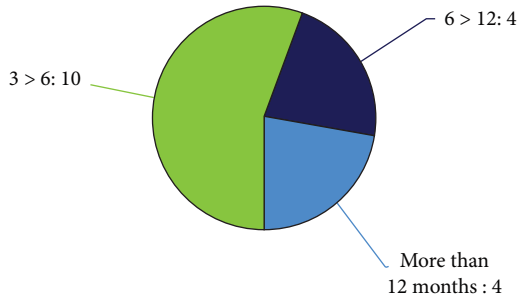

Figure 3: Descriptive analysis w.r.t duration.

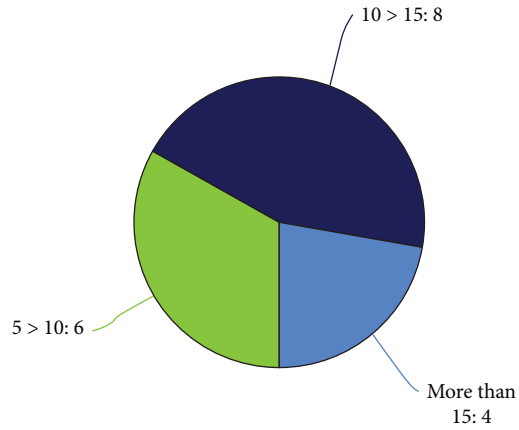

Figure 4: Descriptive analysis w.r.t staffing.

6.1. Risk Factors Assessment Phase. The first phase included the identification of hard risks under risk resources studied from the literature [14]. Among these, the most common risk factors which have been used for the forecasting classified are financial and economic risks, schedule risks, user risks, performance risks, and complexity risks [15–19]. The risk values of these factors were obtained from the risk registers [20].

6.2. Calculation of Project Risk Values (RV). In the project risk registers, risk value (RV) for each risk factor is assessed through which the output is calculated using risk calculation functions such as COCOMO, COCOMO II etc., termed as project risk (PR) [21–24]. Among all the risk registers, a few of them did not have the total project risk calculated so first during preprocessing, the missing values were calculated, project risk and risk values were calculated during the qualitative risk assessment process in terms of their relative probability in 0–1 range and impact (potential loss a risk can cause) on the scale of 0–1.0 in case of no impact, 0.5 for medium and 1 for high impact through the equation mentioned below [25–29].

$$\mathrm{RV} = IxP, \tag{2}$$

where I = impact of occurred risk; P = probability of occurrence

ANN design, training, and testing phase

After the preprocessing of data, ANN was designed. The steps of ANNs as described by Kaastra (1996) are as follows:

(1) Variable selection
(2) Data collection
(3) Data preprocessing
(4) Training, testing, and validation sets (percentages)
(5) Neural network paradigms
(6) Number of hidden layers which will be 1 in this study
(7) Number of hidden neurons, taking 15 neurons
(8) Number of output neurons,
(9) Activation function, sigmoid.
(10) Evaluation criteria
(11) Error function
(12) Neural network training

Number of training iterations are 300 where the difference in MAE error function has become negligible based on the intervals.

6.3. Implementation. The functional procedure of ANNs is totally based on the trial and error. Paradigms adapted for the prediction are observed from the previous successful models and implemented to the current problem. Modifications are incorporated if required, e.g., in the error function, in the activation function or the number of hidden layers.

Mathematical framework

where

$$\text{Input nodes} = [\mathrm{RV}]\_1, [\mathrm{RV}]\_2, [\mathrm{RV}]\_3, [\mathrm{RV}]\_4°, [\mathrm{RV}]\_5. \tag{3}$$

This is the risk factors value from risk registers

$$\text{Weights assigned} = w1, w2, w3 \ldots wn, \tag{4}$$

$$\text{Net input function} = \sum_{i=1}^{n} x_i w_{ij}, \tag{5}$$

$$\text{Sigmoid Function } f(x) = \varnothing(x) = \frac{1}{1 + e^{-x}}, \tag{6}$$

$$\text{Error function (mean square error)} = \frac{1}{2}\sum_{i=1}^{p} (y_i - d_i)^2, \tag{7}$$

$$\text{Error function (mean absolute error)} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}, \tag{8}$$

where y_i is the predicted output produced by the network and d_i is the desired output, p is the number of datasets

Paradigm of ANN

For this research, the paradigm adapted is as under

1 hidden layer has been used.

5 values input

One output layer

Sigmoid activation function.

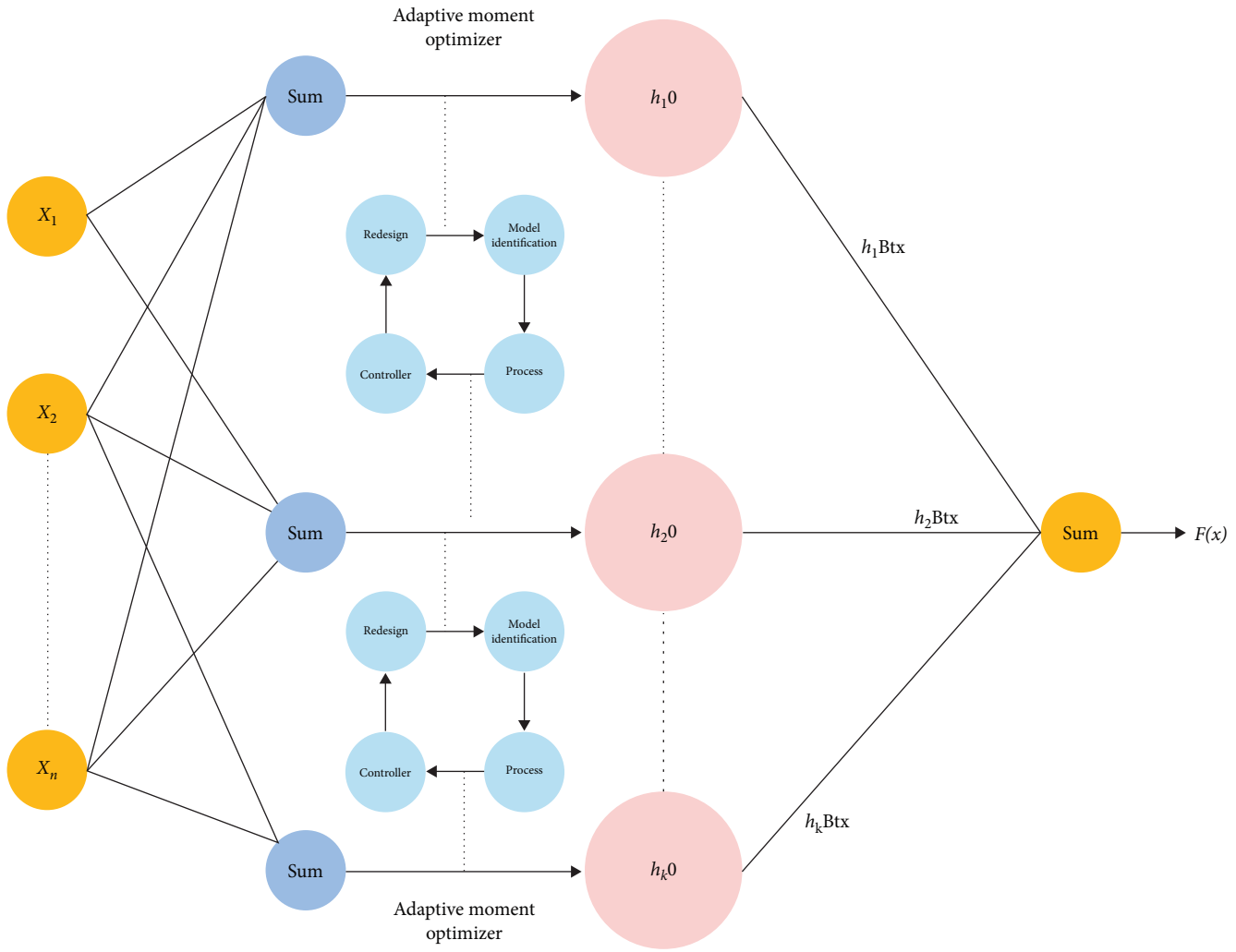

Figure 5: Proposed ANN model.

Figure 5 shows the proposed ANN model. The decision to use a neural network with one hidden layer and eight neurons is a balance between model complexity and simplicity. Complex neural network architectures with numerous hidden layers and neurons may be prone to overfitting, especially when dealing with a limited dataset. On the other hand, an overly simplistic architecture may not capture the underlying patterns in the data. The chosen architecture strikes a balance to prevent overfitting while having sufficient capacity to learn from the data shown in algorithm below.

6.4. Analysis Procedure for Individual Risk Prediction. The basic ANN was adapted for total risk prediction as the patterns were derived from the five factors of the risk calculation. In the dilemma of individual prediction, the input value is already a unit value which could not be simplified. For this purpose, a slightly changed methodology was required using time series as a basis for prediction. The methodology is empowered by regression neural network.

The values in the data are the continuous values hence the date and time series intervals were added and aligned with the individual values of each project.

Algorithm is as follows:

(1) Visualize relation between the dataset
(2) Apply regression neural network
(3) Predict each factor based on month, year assuming resources are dependent on month
(4) Send the predicted values to total ANN's prediction of risk
(5) Robustness Check (Diebold–Mariano (DM) test).

To obtain the best forecasting model and to check its robustness and authenticity, the researchers regularly save some portion of their data for the out-of-sample prediction experiments. Many researches regarding the robustness test recommend the training sample comparisons. The most commonly used test is a DM test or a variant thereof, following the publication of the seminal work by Diebold and Mariano (1995, DM), it has become customary and often required to add an evaluation of significance to forecast comparisons.

In the majority of analyses, an alpha of 0.05 is used as the cutoff for significance. If the p-value is less than 0.05, we

1. Load the Dataset
2. Pre-process to omit and sort NULL values
3. Split the dataset for training and testing (80% for training and 20% for testing)
4. Visualize the data for each factor in relation to other factors
5. Train the dataset
6. Use the designed ANN model with five input values, 1 hidden dense layer and 1 output value
7. Use MSE/MAE with ANN Model
8. Run cycles and Store the history
9. Plot the error as loss function
10. Check Robustness of the model through Diebold–Mariano test
11. After minimal loss, forecast the total risk for next project.

Algorithm 1: Proposed ANN model.

Table 3: The splitting of data for the training of ANN was taken an 80–20 distribution.

Project no.	Complexity	Financial	Performance	Schedule	User	Total PR
1	0	0.01	0.08	0.04	0.04	0.034
2	0.01	0.01	0.01	0.02	0.08	0.01
3	0.08	0.02	0.01	0.04	0.01	0.032
4	0.06	0.01	0.01	0.02	0.02	0.024
5	0.42	0.15	0.04	0.07	0.21	0.164
6	0.04	0.05	0.2	0.03	0.04	0.066
7	0.02	0.04	0.3	0.12	0.1	0.116
8	0.02	0.13	0.2	0.18	0.01	0.08
9	0.24	0.10	0.16	0.2	0.05	0.09
10	0.03	0.28	0.12	0.4	0.25	0.216
11	0.12	0.08	0.03	0.01	0.04	0.056
12	0.03	0.06	0.09	0.04	0.06	0.056
13	0.12	0.08	0.03	0.01	0.04	0.056

Table 4: Training dataset.

Project no.	Complexity	Financial	Performance	Schedule	User	Total PR
14	0.03	0.06	0.09	0.04	0.06	0.051
15	0.03	0.06	0.09	0.04	0.03	0.056
16	0.12	0.08	0.03	0.01	0.04	0.038
17	0.01	0.06	0.04	0.06	0.02	0.108
18	0.06	0.15	0.02	0.21	0.02	0.108

conclude that a significant difference does exist. The p-value for the DM test is checked against the first testing values and the second testing values (validation set). A total of data of 10 projects have been taken for the validation purpose. The test is applied in the algorithm and the p-value and the validation graph are plotted in the results section.

7. Results and Discussion

7.1. Dataset and Split Ratio. Separating data into training and testing sets is the first step after data preprocessing for the ANN. Typically, while splitting, most of the data is used for training, and a smaller portion of the data is used for testing.

After a model has trained itself by using the training set, it can be tested through the other portion by making predictions against the test set. Because the data in the testing set already contain known values for the attribute that you want to predict, it is easy to determine whether the model's guesses are correct. The splitting of data for the training of ANN was taken an 80–20 distribution. About 80% of the total data has been used for training of the algorithm and 20% used for the testing as shown in table. Table 3 shows the splitting of data for the training of ANN was taken an 80–20 distribution. Table 4 shows the training dataset.

After the splitting of data, it was induced into the algorithm with the ReLU activation function, five inputs, one

Table 5: Optimal algorithm paradigm selection.

Network (input, HLN, output)	MSE (min)	MAE (min)
5-1-1	0.0087	0.0101
5-2-1	0.0428	0.2183
5-3-1	0.0020	0.0089
5-4-1	0.1093	0.1171
5-5-1	0.0143	0.1025
5-6-1	0.0023	0.0091
5-7-1	0.0118	0.0906
5-8-1	0.0047	0.0088
5-9-1	0.0080	0.0757
5-10-1	0.0049	0.0089

Table 6: Comparison of the error functions.

Iterations	MAE	MSE
0–20	0.1523	0.0542
21–40	0.1414	0.0328
41–60	0.1131	0.0184
61–80	0.0943	0.0124
81–100	0.0869	0.0107
101–120	0.0811	0.0093
121–140	0.0700	0.0069
141–160	0.0567	0.0046
161–180	0.0428	0.0027
181–200	0.0297	0.0014
201–220	0.0190	7.0020
221–240	0.0121	3.7350
241–260	0.0102	2.6064
261–280	0.0089	2.0467
281–290	0.0088	1.9205
291–300	0.0088	1.8001

hidden layer with eight neurons in the hidden layer and one output layer. The two error functions have been embedded as to check the optimal performance for identifying the optimal set of hidden neurons.

In Table 5, the least scores are recorded for both the error functions against each paradigm. The first column shows the three-digit value, 1st is the number of input layer neurons, 2nd digit depicts the number of neurons in the hidden layer, and 3rd digit is the output neurons. The most optimal results with respect to the least error function MAE came up in the paradigm with eight hidden layer neurons. Table 5 shows the optimal algorithm paradigm selection.

The algorithm was run for 300 iterations. Table 6 shows the difference of the two error functions used in the algorithm with respect to the iterations. The MAE achieves the state of negligible difference in the last iterations while the MSE is nowhere near that stage, hence for the predictive analysis, MAE has been used. Table 6 shows the comparison of the error functions.

7.2. Total Risk Prediction. The first portion of the study was aimed at predicting the total risk of the project. The risk registers obtained had the total risk values calculated through COCOMO-II formula. Registers also had the separate risk factor values for 8–10 different types of risks through which total risk had been calculated.

Table 7: Comparison of the actual vs. predicted values.

Project no.	Actual PR	PR while training	MAE	Dataset
1	0.034	0.1270	0.1523	80–20
2	0.01	0.1310	0.1414	80–20
3	0.032	0.1070	0.1131	80–20
4	0.024	0.1700	0.0943	80–20
5	0.164	0.1071	0.0869	80–20
6	0.066	0.0204	0.0811	80–20
7	0.116	0.0461	0.0700	80–20
8	0.08	0.0233	0.0567	80–20
9	0.09	0.0472	0.0428	80–20
10	0.216	0.1863	0.0297	80–20
11	0.056	0.0370	0.0190	80–20
12	0.056	0.0439	0.0121	80–20
13	0.056	0.0458	0.0102	80–20
14	0.056	0.0422	0.0101	80–20
15	0.051	0.0512	0.0101	80–20
16	0.056	0.0572	0.0098	80–20
17	0.038	0.0372	0.0088	80–20
18	0.108	0.1071	0.0088	80–20

After the optimal set is found, the algorithm is run over the dataset and their mean absolute error is recorded along with the project risk while training. As it can be seen that the difference in error function has become negligible and the total project risk values eventually became similar, the algorithm stopped right in the 300th iteration. Table 7 shows the actual project risk against the predicted project risks while training, and the decrease of the error function in the 4th column on an 80–20 distribution. Table 7 shows the comparison of the actual vs. predicted values.

7.3. The Model Loss Graph. The model loss graph shows the similarity and difference in the training and testing data values of risk. On x-axis, iterations are plotted while on y-axis the error function is plotted. As it is clear from the graph the error function has decreased as the iterations increase and from 250th iteration, the consistency in the error is achieved, with value 0.0088 well till the 300th iteration and therefore, desired accuracy is achieved i.e., 97.12%.

Model loss graphs are essential and main portion of the ANN results as through the iterations the efficiency in training is observed consequently illustrating the robustness of the neural network. Epoch is the number of iterations plotted on x-axis, while loss is the error function on y-axis. Figure 6 shows the model loss graph.

7.4. Graph of Actual vs. Predicted Risk Values. Figure 7 is an illustration of the total risk values against the number of projects. The darker shade depicts the actual risk values obtained from the risk registers while the lighter shade along each project shows the predicted value against each actual

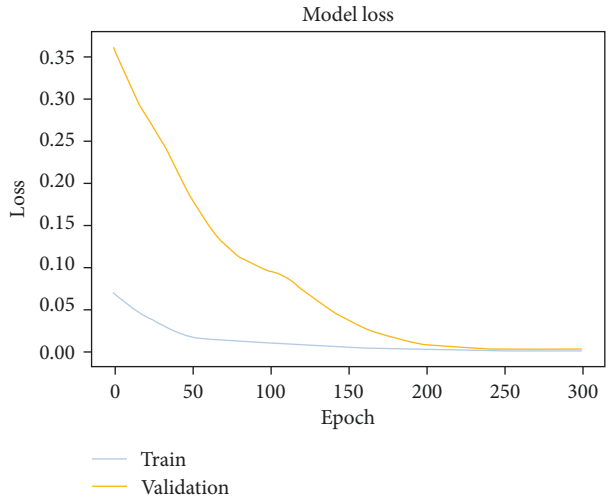

Figure 6: Model loss graph.

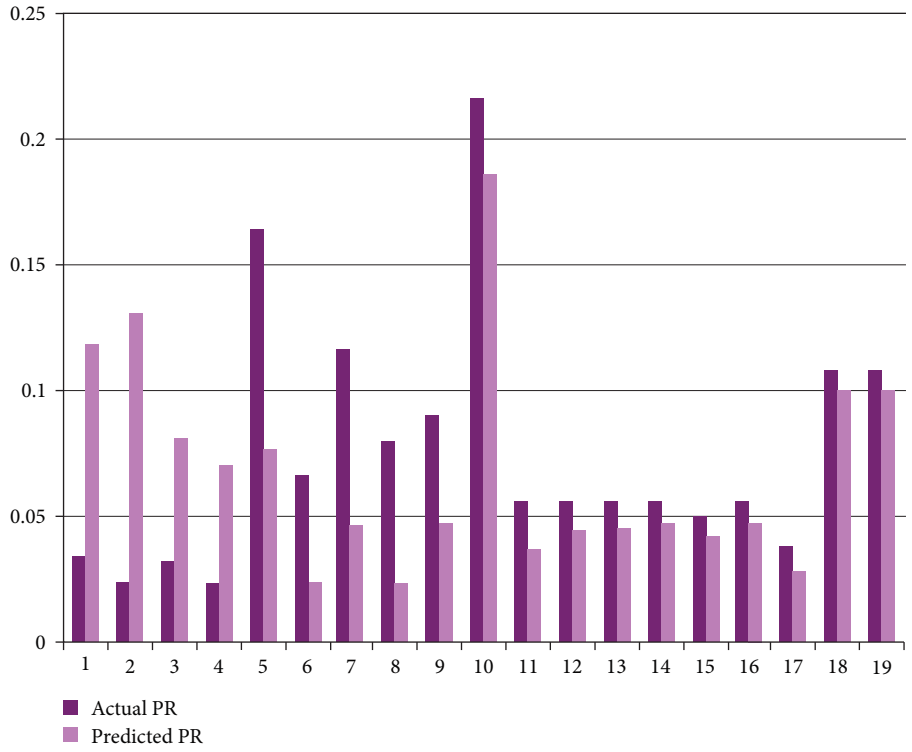

Figure 7: Actual vs. predicted values.

value. As it is visible from the graph that in the start of the projects, the difference in original and predicted value is huge, while as along with the training, the values consequently have become closer while in project no. 17 and 18 the value is closest to each other. Hence, the very next value the algorithm has yielded is the risk value predicted for the next target project i.e., Project No. 19.

Figure 7 shows the actual vs. predicted values. Table 8 shows the predicted risk value of the 19th project along with the margin of error MAE. The first column contains the number of projects, which were 18 in the total dataset. The 2nd column displaying the total project risk obtained from the risk registers and the 3rd column is the error function from the start of the running of algorithm until the end. The row in the last is Project No. 19, which is the target project whose risk value was supposed to be predicted. Table 8 shows the predicted value of total project risk.

With an error of 0.0088, the predicted value is 0.188 or 18% as is in the last row. Through the value, the project manager can interpret in the initiation phase that the next project will have 18% total risk factor. This prediction will assist the project manager to cross check if the company has

Table 8: Predicted value of total project risk.

Project no.	Actual PR	MAE
1	0.034	0.1523
2	0.01	0.1414
3	0.032	0.1131
4	0.024	0.0943
5	0.164	0.0869
6	0.066	0.0811
7	0.116	0.0700
8	0.08	0.0567
9	0.09	0.0428
10	0.216	0.0297
11	0.056	0.0190
12	0.056	0.0121
13	0.056	0.0102
14	0.056	0.0101
15	0.051	0.0101
16	0.056	0.0098
17	0.038	0.0088
18	0.108	0.0088
19	0.188	0.0088

a potential to bear the loss project might cause if there are 18% chances of failure of the project.

In our research on software project risk assessment using a neural network approach, we employed a robust validation methodology to ensure the credibility of our results. Specifically, we utilized a k-fold cross-validation technique. K-fold cross-validation is a widely accepted method for assessing the performance and generalization of machine learning models, including neural networks. In our study, we chose to use k-fold cross-validation with $k = 5$ for the following reasons: Mitigating overfitting: K-fold cross-validation helps mitigate overfitting, a common concern in machine learning. By dividing our dataset into five equally sized subsets (folds), we could iteratively train and evaluate our neural network model five times, with each fold serving as the validation set once and the remaining four as the training set. This process ensures that our model's performance is assessed on various subsets of the data.

7.5. Individual Risk Factors Prediction. Regression neural networks have been used for the individual risk prediction purpose as basic designing of neural networks do not predict on continuous values. ANNs can be used as regression models by reducing their number of input and output neurons. A neural network can "pretend" to be any type of regression model. For example, this very simple neural network, with only one input neuron, one hidden neuron, and one output neuron, is equivalent to a logistic regression.

Figure 8 shows the individual risk factors prediction. The regression graphs for each individual risk factor prediction are as Figure 8. A regression graph holds two values, line as prediction and dots as the original values. The algorithm draws a line in such a manner that the distance from each point to the line is minimal. The distance shows the error between the original dataset and the prediction so as to follow the concept of prediction algorithms, aim is to minimize the error. The next point on the line is termed as the predicted value for the next project of the specific risk. Same algorithm has been just modified a little for the following graphs.

7.6. Predicted Values of Individual Factors for Target Project. Table 9 shows the predicted values of individual risk factors. With the same error function i.e., 0.0088, the predicted values for the next project; Project 19 are recorded in the table. Interpreting the table as the complexity risk in the next project will be 0.08, the financial risk in the next project will be 0.17, the performance risk will be 0.17, the schedule risk will be 0.32, and the user risk will be 0.5.

7.7. Robustness Check and Validation. The validation is the third step of the ANN. According to Russell and Norvig, to get an early estimate of the skill of the model, to avoid it from over fitting and to check the robustness of the model, dataset that is completely novice to the algorithm is induced as the testing set so that to check if the algorithm still yields output with same accuracy or not. Validation set can be the subset of the training dataset (Russel and Norvig, 2010). Table 10 shows the forecast 1, forecast 2, and the error function. Table 10 shows the validation of the forecast.

Following graph shows the predicted values against the test values (prediction 1 and prediction 2). For the validation purpose, data of 10 more projects were collected and through similar procedure, preprocessed, and predicted. Their difference is recorded in Figure 9 with the p-value.

Figure 9 shows the validation graph. On the x-axis is plotted the error while on the y-axis, the next 17 iterations are plotted. The p-value according to the DM test is; p-value $= 0.020874830833241643$, as is less than 0.05 we conclude that no significant difference exist in the two forecasts.

7.8. Analysis on the Basis of Predictions. Once the predicted values have achieved, it is important to interpret the values and analyze in order to make proper measure for the risk. The predictions are rated on the scale of 0–1.0 being the lowest risk or no risk, 0.5 being the cutoff point, and 1 being the highest risk value in the target project. Figure 10 shows the framework and output of the total project risk plotted on the scale.

Figure 10 shows the total project risk = 0.188. Similarly, the values of all the five risk factors have been plotted on the same model as shown in the Figure 11.

In the context of software project risk assessment, it is essential to evaluate the performance of the proposed neural network approach against a baseline model to gauge its effectiveness. We conducted a comparative analysis between the neural network model and a simple linear regression (LR) model, a widely used baseline in predictive modeling. Here are the key findings from the comparison:

(1) Predictive accuracy:
   (i) Neural network: the neural network model demonstrated superior predictive accuracy compared

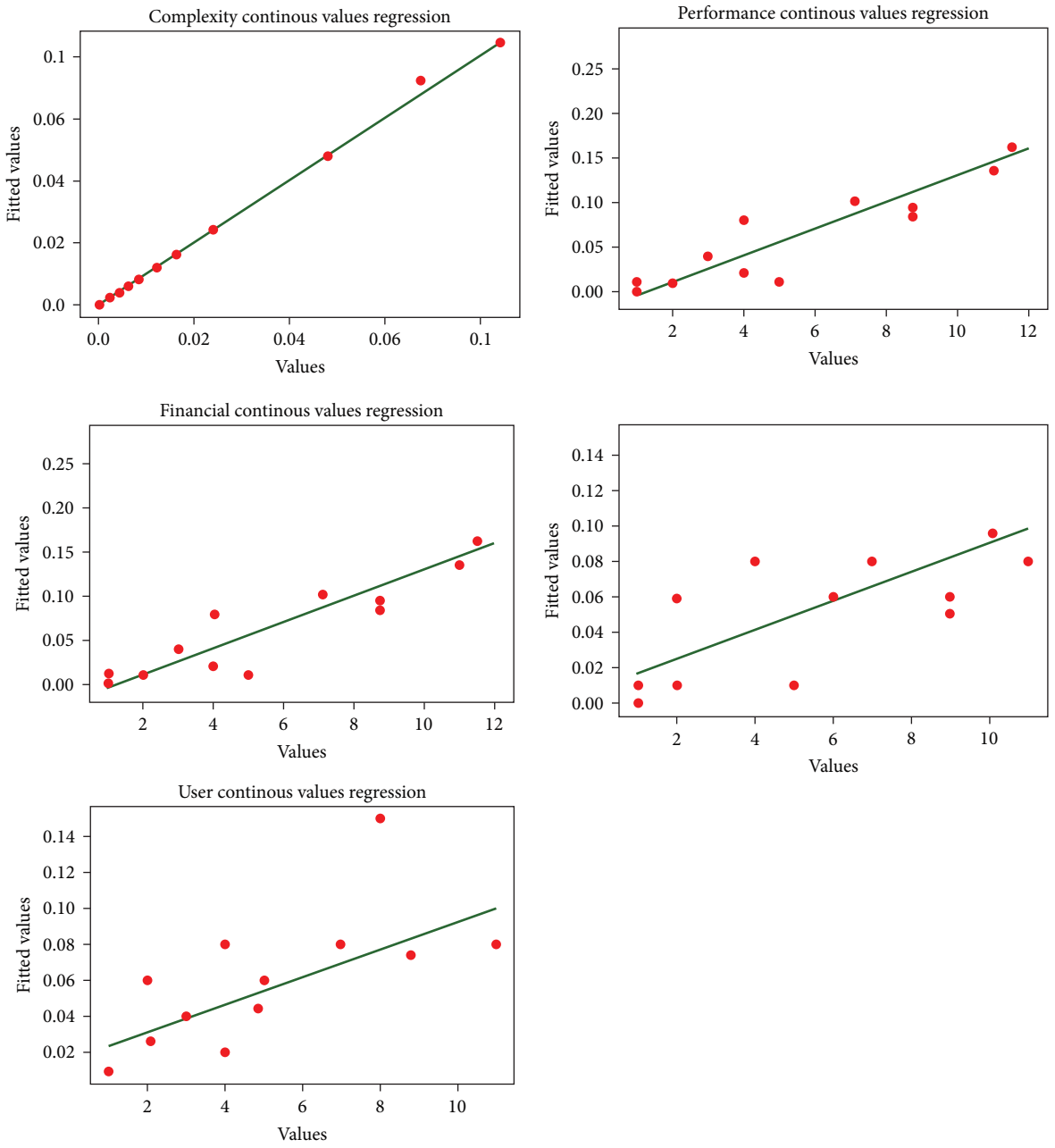

Figure 8: Individual risk factors prediction.

Table 9: Predicted values of individual risk factors.

Project no.	Complexity	Financial	Performance	Schedule	User
1	0	0.01	0.08	0.04	0.04
2	0.01	0.01	0.01	0.02	0
…	…	…	…	…	…
…	…	…	…	…	…
…	…	…	…	…	…
19	0.08	0.17	0.17	0.32	0.5

Table 10: Validation of the forecast.

Forecast 1	Forecast 2	MAE
0.0233	0.0145	0.0088
0.0472	0.0384	0.0088
0.1863	0.1775	0.0088
0.037	0.0282	0.0088
0.0439	0.0351	0.0088
0.0458	0.037	0.0088
0.0422	0.0334	0.0088
0.0512	0.0424	0.0088
0.0572	0.0484	0.0088
0.0372	0.0284	0.0088

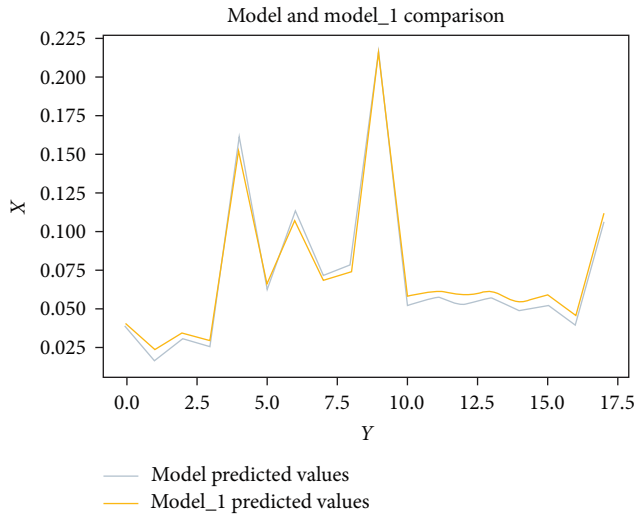

Figure 9: Validation graph.

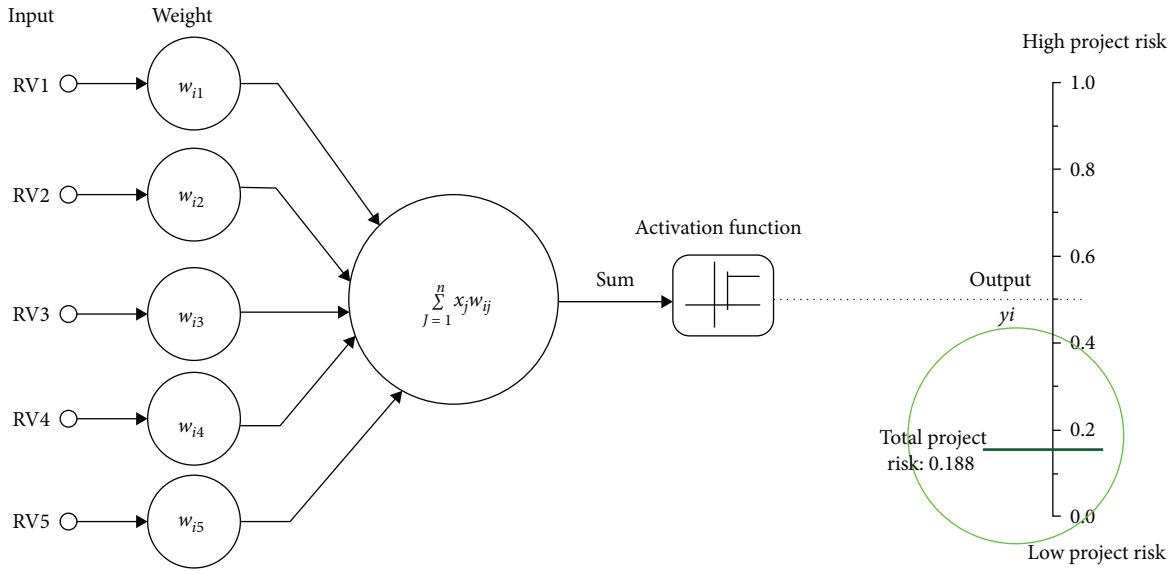

Figure 10: Total project risk = 0.188.

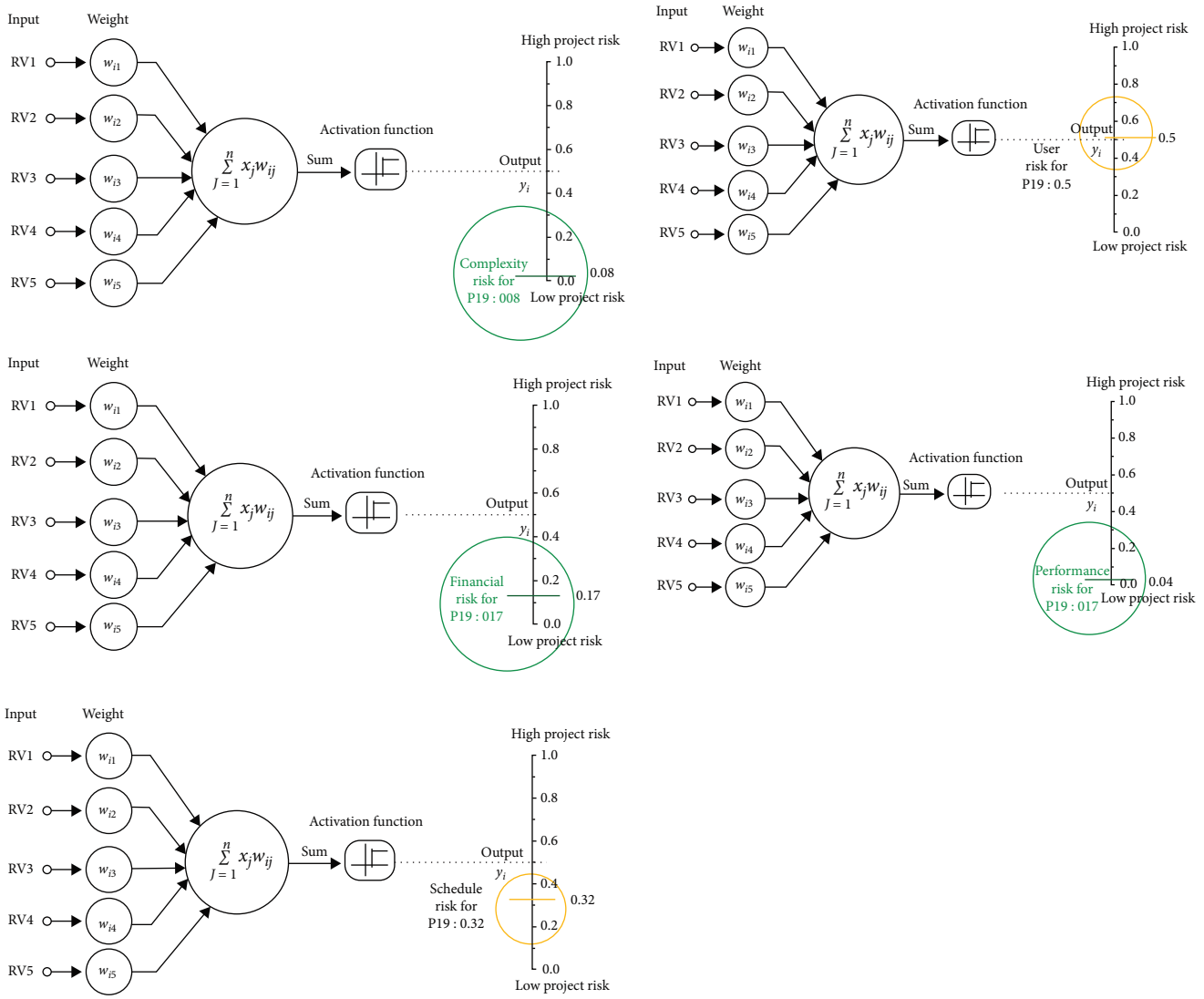

FIGURE 11: Five risk factors.

to the LR model. This was evident in terms of lower MSE, higher R-squared values, and other relevant evaluation metrics. The neural network's ability to capture nonlinear relationships and complex patterns in the dataset provided a clear advantage over LR, which assumes linear relationships between variables.

(ii) LR: the LR model, while straightforward and interpretable, struggled to capture the nuances and nonlinearities present in the software project risk dataset. As a result, its predictive accuracy was comparatively lower.

(2) Generalization:

(i) Neural network: the neural network model exhibited better generalization performance. It was less prone to overfitting, as evidenced by consistent performance on both the training and validation datasets. This suggests that the neural network model is more likely to perform well on unseen data.

(ii) LR: the LR model tended to overfit the training data to some extent, leading to a larger performance gap between training and validation datasets. This indicates a limited ability to generalize to new, unseen projects.

8. Conclusion

In this research we examined the efficiency of ANNs for the risk assessment of software projects. Use of artificial intelligence has been tested for the prediction of project risks in the software industry of Pakistan. Having a clear idea of the fact that Pakistan's economic conditions are not very nurturing for any business leads to the fact that the projects are more prone to the risks related to human capital, financial aspects, and user personal. The study focused on predicting the total project risk from the historical data of the next project and predicting individual values of the risk factors in the upcoming project. Consistent with the theoretical argument, ANNs

have proven to yield highest accurate results depending on the consistency of data. The model loss graph comprehensively represents the original and the predicted values and their difference which has decreased along with the training of model. According to the results, the minimum difference left in the original and predicted value is 0.0088 in both cases; hence 99.12% accuracy has been achieved through this model for total risk prediction. Adding Lambda as the uncertainty value (most commonly taken 2%), the accuracy achieved is 97.12%. The developed will allow the project managers to assess if the specific risk can occur, what could be the intensity of that risk and consequently evaluate what measures are required to cope up with the risk. In conclusion, our research has demonstrated the effectiveness of a neural network-based approach for the software project risk assessment. By training a neural network model on a comprehensive dataset, we have shown that it can provide accurate predictions of project risk levels, thereby aiding project managers and stakeholders in making informed decisions. Future research can delve deeper into identifying and categorizing risk factors specific to various types of software projects, such as agile, waterfall, or hybrid methodologies. This could lead to more tailored risk assessment models.

Data Availability

The author used data to support the findings of this study that are included within this article.

Conflicts of Interest

The authors claim that this paper does not include any conflicts of interest.

Authors' Contributions

M.N Alatawi and S.Hussain contributed in the conceptualization, methodology, validation, investigation, resources, data curation, formal analysis, and writing—review and editing. AA. Aldaeej and H.S Alwageed contributed in the validation and writing—review and editing. A. Alshammari and I.K Alali contributed in the software and writing—review and editing. S.Alyahyan contributed in the methodology and writing—review and editing. S.Hussain contributed in the software. I.K Alali and AA. Aldaeej contributed in the investigation. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

This study is supported by the Department of Information Technology, University of Tabuk, Tabuk 71491, Saudi Arabia.

References

[1] M. M. Raziq, F. M. Borini, O. F. Malik, M. Ahmad, and M. Shabaz, "Leadership styles, goal clarity, and project success," *Leadership & Organization Development Journal*, vol. 39, no. 2, pp. 309–323, 2018.

[2] T. O. Lehtinen, M. V. Mäntylä, J. Vanhanen, J. Itkonen, and C. Lassenius, "Perceived causes of software project failures—an analysis of their relationships," *Information and Software*, vol. 56, no. 6, pp. 623–643, 2014.

[3] N. Aslam, W. Y. Ramay, K. Xia, and N. Sarwar, "Convolutional neural network based classification of app reviews," *IEEE Access*, vol. 8, pp. 185619–185628, 2020.

[4] I. Alam, N. Sarwar, I. Noreen, and M. U. Ashraf, "Statistical analysis of software development models by six-pointed star framework," *PLoS One*, vol. 17, no. 4, Article ID e0264420, 2022.

[5] I. J. Schwarz and I. P. M. Sánchez, "Implementation of artificial intelligence into risk management decision-making processes in construction projects," pp. 358–378, 2015, MS Thesis Iceland School of Energy at Reykjavík University.

[6] M. Abdelgawad and A. R. Fayek, "Risk management in the construction industry using combined fuzzy FMEA and fuzzy AHP," *Journal of Construction Engineering and*, vol. 136, no. 9, pp. 1028–1036, 2019.

[7] M. Bilal, A. Gani, M. Liaqat, N. Bashir, and N. Malik, "Risk assessment across life cycle phases for small and medium software projects," *Journal of Engineering Science and Technology*, vol. 15, no. 1, pp. 572–588, 2020.

[8] A. Ahmad, M. Bilal, K. Latif, and Zainab, "A study of project management processes for sustainable and successful projects in software industry: expectations vs perceptions of managers," *Journal of Accounting and Finance in Emerging Economies*, vol. 7, no. 1, pp. 103–115, 2021.

[9] M. Ibrahim, I. S. Bajwa, N. Sarwar, H. Abdul Waheed, M. Zulkifl Hasan, and M. Zunnurain Hussain, "Improved hybrid deep collaborative filtering approach for true recommendations," *Computers, Materials & Continua*, vol. 74, no. 3, pp. 5301–5317, 2023.

[10] M. Ibrahim, I. S. Bajwa, N. Sarwar, F. Hajjej, and H. A. Sakr, "An intelligent hybrid neural collaborative filtering approach for true recommendations," *IEEE Access*, vol. 11, pp. 64831–64849, 2023.

[11] B. Jehan, K. Ghani, and M. Shafi, "Assessment of project management practices in Pakistan software industry," in *IEEE SCONEST*, 2014.

[12] I. Basharat, T. Nafees, and M. Abbas, "Risks factors identification and assessment in virtual projects of software industry: a survey study," in *2013 Science and Information Conference IEEE*, pp. 176–181, 2013.

[13] M. J. Thaheem, A. De Marco, and K. Barlish, "A review of quantitative analysis techniques for construction project risk management," in *Proceedings of the Creative Construct Conference*, pp. 656–667, 2012.

[14] M. Nayak and T. Abdullah, "Short-term predication of risk management integrating artificial neural network (ANN)," *International Journal of Advanced Science and Technology*, vol. 29, no. 3, pp. 4876–4883, 2020.

[15] G. Auth, O. Jokischpavel, and C. Dürk, "Revisiting automated project management in the digital age—a survey of AI approaches," *Online Journal of Applied Knowledge Management*, vol. 7, no. 1, pp. 27–39, 2019.

[16] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," arXiv: 1505.05424, 2015.

[17] D. Dellermann, P. Ebel, M. Söllner, and J. M. Leimeister, "Hybrid Intelligence," *Business & Information Systems*, vol. 61, no. 5, pp. 637–643, 2019.

[18] A. Foote and L. A. Halawi, "Knowledge management models within information technology projects," *Journal of Computer Information*, vol. 58, no. 1, pp. 89–97, 2018.
[19] E. Fernandes, M. Holanda, M. Victorino, V. Borges, R. Carvalho, and G. V. Erven, "Educational data mining: predictive analysis of academic performance of public school students in the capital of Brazil," *Journal of Business Research*, vol. 94, pp. 335–343, 2019.
[20] S. Goyal, "Comparison of machine learning techniques for software quality prediction," *International Journal of Knowledge and Systems Science*, vol. 11, no. 2, pp. 20–40, 2020.
[21] S. H. Han and J. Diekmann, "Judgment-based cross-impact method for predicting cost variance for highly uncertain projects," *Journal of Construction*, vol. 5, no. 2, pp. 171–192, 2004.
[22] L. Hung, "A risk assessment framework for construction project using artificial neural network," *Journal of Science and Technology in Civil Engineering*, vol. 12, no. 5, pp. 51–62, 2018.
[23] J. R. Meredith, S. M. Shafer, and S. J. Mantel, *Project Management: A Strategic Managerial Approach*, John Wiley & Sons, 2017.
[24] N. Muthukrishnan, F. Maleki, K. Ovens, C. Reinhold, B. Forghani, and R. Forghani, "Brief history of artificial intelligence," *Neuroimaging Clinics of North America*, vol. 30, no. 4, pp. 393–399, 2020.
[25] J. Menezes, C. Gusmão, and H. Moura, "Risk factors in software development projects: a systematic literature review," *Software Quality*, vol. 27, no. 3, pp. 1149–1174, 2019.
[26] A. Mossalam and M. Arafa, "Using artificial neural networks (ANN) in projects monitoring dashboards' formulation," *HBRC Journal*, vol. 14, no. 3, pp. 385–392, 2018.
[27] B. Ramzan, I. S. Bajwa, N. Jamil et al., "An intelligent data analysis for recommendation systems using machine learning," *Scientific Programming*, vol. 2019, Article ID 5941096, 20 pages, 2019.
[28] D. Reed, "Data smart: using data science to transform information into insight," *Journal of Direct, Data and Digital Marketing Practice*, vol. 15, no. 4, pp. 354-355, 2014.
[29] B. Roy, R. Dasgupta, and N. Chaki, "A study on software risk management strategies and mapping with SDLC," in *Advanced Computing and Systems for Security*, pp. 121–138, Springer, New Delhi, 2016.