Proactive Risk Management in IT Project based on Neural Networks*

Oleksii Matsiievskyi[1,*,†], Igor Achkasov[1,*,†], Yuliia Riabchun[1,*,†], Roman Zhuk[1,*,†] and Roman Mazurenko[1,*,†]

[1] *Kyiv National University of Construction and Architecture, 31, Air Force Avenue, Kyiv, 03037, Ukraine*

Abstract

The article considers an approach to proactive risk management of IT projects using modern deep learning models. A new hybrid model based on a recurrent neural network (RNN) and the Transformer architecture is proposed to predict risks at different stages of an IT project. The model combines the ability of the RNN to detect temporal dependencies with the ability of the Transformer model to take into account long-term relationships in the data, which allows to increase the accuracy of predicting risk events. An experimental study was conducted on the data of a real IT project: the results showed that the proposed model outperforms traditional approaches and individual neural networks, RNN alone or Transformer alone, in terms of accuracy and completeness of risk prediction. The scientific novelty of the work lies in the development of an original RNN-Transformer architecture specially adapted for the field of project management in IT and demonstration of its effectiveness in the task of early detection of project risks. The proposed approach helps to increase the success rate of IT projects by providing timely warning of possible problems.

Keywords

RNN, Transformer, AI, project management, IT, risk forecasting

1. Introduction

Risk management is an integral part of successful IT project management. IT projects, particularly software development, are characterized by high dynamism and uncertainty. Frequent changes in requirements, technical complexity, limited resources, and tight deadlines all create fertile ground for risks to arise throughout the project lifecycle. Despite the improvement of development methodologies and the emergence of new tools, the share of unsuccessful or problematic IT projects remains significant. According to researchers, the success rate of IT projects still does not meet industry expectations, and every year project failures lead to a loss of significant funds and time, especially for large enterprises. This underscores the importance of effective risk management: identifying potential problems at an early stage and taking measures to mitigate them can prevent failures and cost overruns [1].

Traditionally, risk management in projects is carried out by expert methods - by interviewing stakeholders, using risk checklists, assessing the probability and impact of risks based on the experience of managers, etc. Such approaches are usually subjective and have a limited ability to process large amounts of project data or dynamic changes in the situation. With the development of project management information systems and the accumulation of historical data, opportunities have opened up for the application of artificial intelligence and machine learning methods to improve the efficiency of risk forecasting. In particular, deep learning technologies

*ITPM-2025: VI International Workshop "IT Project Management", May 22, 2025, Kyiv, Ukraine,

* Corresponding author.

† These authors contributed equally.

matsievskiyolexiy@gmail.com (O. Matsiievskyi); achckasov.i@ukr.net (I. Achkasov); super.etsy@ukr.net (Y. Riabchun); zhukroman77@gmail.com (R. Zhuk); mazurenkodev@gmail.com (R. Mazurenko)

0009-0008-2341-8166 (O. Matsiievskyi); 0000-0002-7049-0530 (I. Achkasov); 0000-0002-8320-4038 (Y. Riabchun); 0009-0007-1657-6281 (R. Zhuk); 0000-0003-3954-9423 (R. Mazurenko)

have demonstrated success in forecasting tasks in various domains, from financial risks to man-made accidents. This has stimulated interest in their use in project management as well.

This study presents a novel approach to proactive IT project risk management by integrating the temporal sensitivity of recurrent networks with the contextual understanding of Transformer architectures. Such a hybrid model provides early warning signals based on dynamic documentation patterns, which is increasingly critical given the volatility, complexity, and uncertainty characterizing modern software projects.

Among the deep learning models potentially suitable for analyzing IT project risks are recurrent neural networks (RNN) and the Transformer model. RNN, in particular, its LSTM or GRU architecture, is able to process sequential data, capturing time dependencies and trends - this is important because project risks evolve over time. Instead, Transformer, built on a self-attention mechanism, works efficiently with long sequences and can detect global dependencies in data without losing important information even at long time intervals. Transformer models were a breakthrough in the field of natural language processing, but recently they have been successfully applied to other sequential data, including time series [2].

The purpose of this paper is to study the capabilities of RNN and Transformer neural networks for risk prediction in IT projects and to develop a new approach that combines the advantages of both models. Our goal is to create a forecasting model that analyzes the progress of a project (the dynamics of its key indicators) and is able to signal in advance the high probability of critical risks, such as deadline delays or budget overruns [3]. Unlike previous approaches that consider only static factors or rely on expert judgment, our model automatically learns from historical project data, identifying hidden patterns that precede problems. The scientific novelty of the study is the combination of two advanced deep learning architectures within the project management task, which allows for improved forecast quality.

IT project risk management has traditionally been based on standardized approaches, such as those in PMBOK or ISO 21500, where the focus is on identifying risks, qualifying and quantifying them, and developing response and monitoring plans. However, with the growing complexity of IT projects and the availability of large amounts of data, researchers and practitioners are increasingly turning to data analytics to support decision-making in risk management.

In recent years, there have been studies demonstrating the effectiveness of machine learning in project risk prediction. In particular, Sanjay Bauskar (2024) [4] describes the use of predictive analytics methods to assess the risks of IT projects based on historical data. The author analyzed the time, resource, and performance indicators of previous projects and built a model based on an ensemble of Gradient Boosting Machine decision trees that achieved about 85% of the overall risk prediction accuracy. This result exceeds the performance of previous statistical models and demonstrates the potential of using data-driven approaches in project risk management. It was also noted that the introduction of such a tool increased resource efficiency by ~15% and reduced project costs by 10% compared to traditional methods.

Another area of research is the use of neural networks to predict risks. For example, Loubna Aggabou, Brahim Lakehal, Mohamed Mouda (2024) [5] proposed an approach to risk assessment in construction projects based on an artificial neural network ANN. Although the domain is different from IT, their results are revealing: the ANN model was able to predict the level of risk with very high accuracy (coefficient of determination R^2 = 0.97), demonstrating the effectiveness of neural networks in project management tasks. This work emphasizes that neural networks are able to take into account the nonlinear relationships between numerous risk factors and provide more accurate predictions than traditional analysis methods.

In the field of IT projects, researchers are also experimenting with neural network models. For example, Ziwen Diao (2024) [6] considered the problem of assessing the investment risks of projects using a hybrid approach: the LSTM recurrent neural network for analyzing time series

data was supplemented by gradient boosting (GBM) to take into account nonlinear effects. The combination of LSTM+GBM outperformed other models in terms of predicting risk events, which the author explains by the ability of LSTM to effectively extract spatio-temporal features from sequential project data, and GBM to enhance the prediction through ensemble learning. As a result, the ensemble model provided a significant increase in the accuracy and stability of risk forecasting compared to the use of classical regression methods alone.

Recurrent networks, in particular LSTM architectures, have proven to be effective in time-series forecasting tasks due to the state preservation mechanism that allows the model to "remember" previous information. Ke Xu, Yu Cheng, Shiqing Long, Junjie Guo, Jue Xiao, Mengfang Sun (2024) [7] in a study of financial risks showed that the optimized LSTM model outperforms Random Forest and XGBoost algorithms in terms of AUC when predicting risks based on sequential financial data. This confirms that RNNs are able to cope with complex sequential dependencies better than traditional machine learning methods, especially when risk manifests itself as a pattern over time. On the other hand, the disadvantage of classical RNNs is the problem of gradient decay on long sequences: as the project duration increases, the accuracy of the forecast based on LSTM alone may decrease, as important signals from the beginning of the project are "forgotten" by the network.

A new word in sequence modeling is Transformer networks, which use a multi-headed self-attention mechanism for parallel data processing. Transformer initially demonstrated breakthrough results in text processing (machine translation, models such as BERT, GPT, etc.), but was gradually adapted to other tasks. Unlike RNNs, Transformer does not process a sequence strictly step by step, but analyzes all the relationships between elements through the attention mechanism. This allows it to capture long-term dependencies without losing the gradient. In the context of project risk management, this property is valuable: for example, early signs of risk at the planning stage can have an impact on the final project outcome, and Transformer is more likely to take them into account in forecasting than a classic recurrent network [8-9].

Although the experience of applying Transformer models directly to project management is still limited, there are encouraging examples in related fields. Junwei Shi, Shiqi Wang, Pengfei Qu, Jianli Shao (2024) [10] proposed to combine LSTM and Transformer algorithms to predict water inflow in mines (a task of monitoring technological risk). Their hybrid LSTM-Transformer model showed the highest prediction accuracy compared to individual LSTMs, individual Transformers, and even CNN-LSTMs, improving all key model metrics. This demonstrates that the synergy of RNNs and Transformers can be beneficial: LSTM picks up local trends, while Transformer's attention mechanism picks up global patterns. Another example is the work of Jinghan Zhang, Henry Xie, Xinhao Zhang, Kunpeng Liu (2024) [11], where the training of the Transformer model was modified for financial risk problems by introducing a special loss function sensitive to extreme risks, the so-called loss-at-risk. This allowed us to significantly improve the accuracy of estimating unlikely but critical events and significant financial losses, which standard models often underestimated. The result demonstrates the flexibility of the Transformer approach, which can be adapted to the specifics of the risk management task.

In their paper, Dorothea S. Adamantiadou, Loukas Tsironis (2025) [12] confirm that artificial intelligence is becoming an important tool in project management, in particular for risk assessment and forecasting. These reviews note that since the 2010s, the use of machine learning methods in project management has grown manifold, with both classic decision tree algorithms and regression models, as well as hybrid and deep neural networks being used. Tasks such as risk assessment, project duration and cost forecasting are increasingly being solved with the help of AI models. At the same time, the authors emphasize a number of challenges: the need for access to high-quality project data, taking into account dynamic changes during implementation, as most studies work with static historical data, and integrating models into project management

practice. In particular, in many cases, the success of models is tested on retrospective data, and there is a lack of confirmation of their effectiveness in real time on existing projects [13].

Thus, the literature analysis shows that the use of neural networks for risk prediction is a promising area that has already yielded positive results in related industries. However, in the field of IT projects, the issue of developing specialized models that can take into account the specifics of software projects (rapid changes, human factor, multi-team environment, etc.) remains open. Most existing solutions are either focused on other domains, such as construction, finance, or use relatively simple algorithms. This motivates the creation of new approaches that combine the latest achievements of RNN and Transformer deep learning and adapt them to the task of risk prediction in IT projects. In the next section, we present our proposed model, which is one of these approaches.

2. The main research

To solve this problem, we propose a hybrid neural network consisting of two main components: a recurrent submodel and a Transformer-based submodel (Figure 1). The idea is to enable the model to process temporal data in parallel in two ways: through the RNN state, which reflects the latest trends in project development, and through the self-awareness mechanism, which allows taking into account the mutual influence of all previous states. Both representations are integrated to form a final risk assessment.

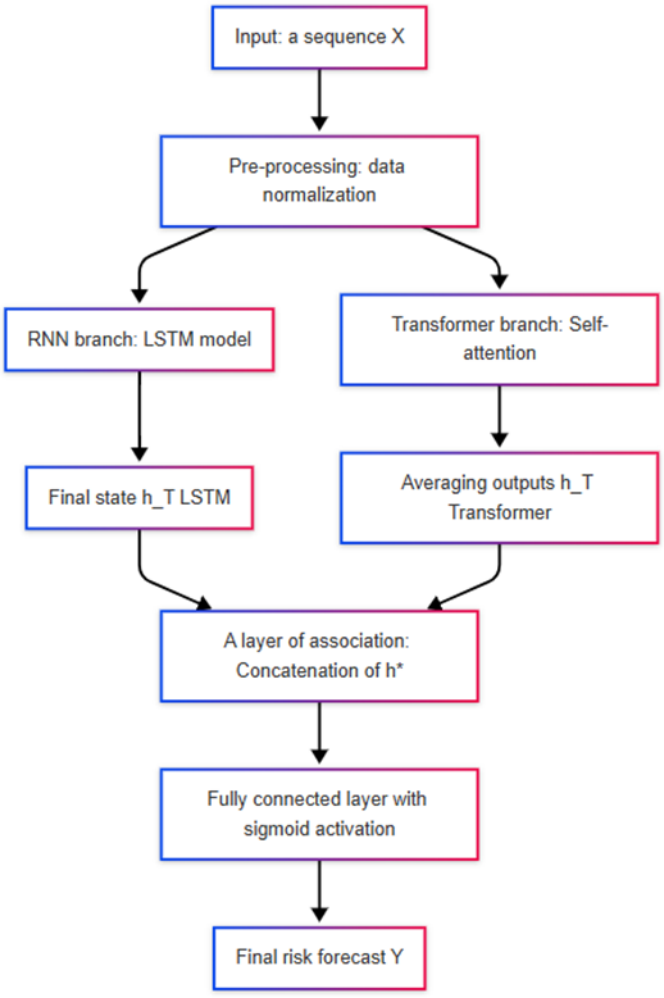

Figure 1. Architecture of the hybrid RNN-Transformer neural network model

Risk forecasting in the context of an IT project is formally considered a classification or regression problem based on time series data about the project's progress. Let's say we have a time series of indicators

$$X = \{x_1, x_2, \dots, x_T\} \quad (1)$$

Where x_T is a vector of project characteristics at the end of interval T, for example, a week or a month. The ultimate goal is to predict some variable Y related to project risk. This can be a binary variable: successful completion of the project, failure or critical issues, or another risk indicator, such as expected budget overrun, number of critical defects, etc. In this paper, we consider a binary risk classification for clarity: Y = 1 if the project is considered "high risk" (likely failure or significant problems), and Y = 0 if the risk is low (expected successful completion). Accordingly, the model receives a sequence of project metrics X as input and must produce an estimate hat Y probability or risk class.

The main steps of data processing in the model are as follows:

2.1. **Preliminary data processing**

Project output consists of a set of indicators that may have changed over time. Examples of such indicators are: percentage of planned tasks completed, number of open defects or tasks, schedule deviations, ratio of actual to planned performance - SPI/CPI indicators, team workload, team composition changes, number of requirements changes, etc. Each such indicator is normalized, for example, by min-max scaling, to ensure stable training of the neural network. If some characteristics are categorical, for example, the project phase: design, development, testing, they are encoded into a numerical format of one-hot or embeddings. Thus, for each time step T, a numerical vector of project characteristics x_t of dimension m is formed, where m is the number of indicators taken into account by the model.

2.2. **RNN branches (LSTM)**

The first component of the proposed model is a single- or multi-layer recurrent neural network of the LSTM type, which sequentially processes the input data - a sequence of project characteristics vectors $x_1,x_2,...,xT$. At each time step, the LSTM model receives another input vector xT and updates its internal hidden state h_t. This hidden state accumulates information about all previous time steps from the beginning of observations to the current moment. After processing the entire sequence, the final LSTM hidden state h_t is obtained, which is a compact representation of the entire project with an emphasis on recent events due to the specifics of recurrent neural networks. The dimensionality of the LSTM hidden state, the number of neurons in the network, is a hyperparameter of the model that should be large enough to capture significant information, but not too large to avoid overfitting. In the experimental study, 64 neurons were used.

2.3. **Transformer branches**

The second component of the model is the Transformer Encoder block, which processes the entire input sequence $x_1,x_2,...,xT$ in parallel. At the initial stage, positional encoding is added to each vector of the input sequence, which provides information about the order of the elements in the sequence. This procedure is necessary because the self-attention mechanism does not take into account the order of elements in time. The transformer encoder consists of one or more layers of the self-attention mechanism, where each element of the sequence interacts with all the others with certain attention coefficients. These coefficients are determined by the network itself during training and allow the model to take into account global dependencies in the data. After processing with the Transformer encoder, a new sequence of representations $z_1,z_2,...,z_t$ is obtained, where each vector zT contains information about the sequence element xT in the

context of the entire sequence. For simplification and formation of a single representation, global averaging of all received vectors is used - global average pooling. As a result of this averaging, the final hT Transformer vector is formed, the dimension of which is determined by the hyperparameter; in the experiment, the dimension was used to be 64 or 128.

2.4. **Combining and output layer**

Once the two final vectors, hT LSTM and hT Transformer, are obtained, they are combined into one generalized vector h*, which contains the project features identified by both approaches. This generalized vector is then passed to one or more fully connected layers of the neural network for final classification. The output neuron of this layer has sigmoidal activation, which provides a forecast in the form of a probability of high risk Y = 1. If the obtained probability value exceeds a certain threshold, for example, 0.5, the model classifies the project as risky. If the task involves several risk classes, softmax activation is used instead of sigmoid activation.

2.5. **Training the model**

The network is trained using the backpropagation algorithm based on training data. The binary cross-entropy is used as a loss function, which estimates how well the predicted probability corresponds to the actual risk class. The optimization is performed by the Adam algorithm, which allows to automatically adapt the learning rate. To prevent overtraining of the network, regularization is used, a technique for randomly excluding dropout neurons with a probability of 0.3 at the pooling layer and subsequent fully connected layers, as well as the early stopping method, which stops training if there is no improvement in the quality of the forecast on validation data for several epochs.

2.6. **Integration of additional data**

The proposed architecture is flexible and can be expanded by adding new types of data, such as textual descriptions of risks, comments or reports from project participants. Such data can be processed by a separate natural language model, such as Transformer for NLP tasks, to generate textual representations, which are then joined to the main generalized vector h*. In the current implementation, the focus was exclusively on the numerical time characteristics of the project, but the integration of text or other types of data in the future could significantly improve the accuracy and completeness of risk forecasting.

To better understand the practical applicability of the proposed hybrid model, an additional simulation was conducted based on hypothetical project scenarios. These scenarios were designed to reflect typical risk development patterns observed in real IT projects. For instance, one scenario included a sharp drop in team productivity combined with a sudden increase in change requests during the middle phase of the project. Another scenario simulated a gradual but consistent deviation from the schedule over several weeks without major changes in other indicators.

The model successfully detected risk patterns in both cases: in the first, it flagged the project as high-risk within two weeks of the deviation onset, while in the second, the Transformer component captured the trend of increasing delay, triggering a high-risk prediction before the project entered its final quarter. These simulations demonstrate the model's sensitivity not only to acute disturbances but also to subtle, long-term accumulations of risk signals. This ability is especially important in environments where early intervention can significantly reduce the negative impact of project issues.

The proposed model combines the efficiency of RNNs, quick response to recent changes in the project with a global vision, Transformer, and consideration of the context of the entire project.

This combination is expected to provide more accurate risk forecasts compared to using each of the components separately.

2.7. **Experiment**

To ensure research reproducibility and representativeness of the sample, this study analyzed 50 IT projects sourced from a variety of mid- to large-scale Ukrainian software development companies operating in both outsourcing and product development domains. Projects were selected based on availability of complete lifecycle documentation, access to team retrospectives, and stakeholder feedback. The classification into "successful" and "problematic" projects relied on standard PMI criteria—adherence to timeline, budget, and functional requirements—as well as project post-mortem analyses. Prior to modeling, all textual artifacts (requirements specifications, risk logs, sprint reports) underwent normalization, tokenization, and lemmatization. Projects were evaluated for homogeneity of implementation conditions (team size, methodology, duration), and projects with extreme deviations were excluded to minimize confounding factors.

For each project, the following time series of metrics were available with a frequency of 1 week:

- Plan execution (%) - the share of completed work out of the planned work for the week.
- Time deviation (days) - delay or advancement of the schedule at the end of the week.
- Active risks (number) - how many risk events are in the "open" status (according to the project risk register).
- Requirements changes (number) - the number of change requests received during the week.
- Team size (people) - the number of active project team members this week.
- Team productivity (story points) - the amount of work performed per week (in conventional volume units).
- Defects (open/closed) - statistics of found and fixed defects for the week.

Each project lasted from 20 to 40 weeks; for consistency, the duration was normalized by scaling to 30 intervals, truncating longer projects to 30 weeks, and padding shorter projects by repeating the last state. Based on the results, each project had a final risk assessment: the company's experts classified the completed projects into 34 successful projects and 16 problematic projects. Problematic projects included those that significantly exceeded the timeframe or budget, or did not meet the customer's requirements. This binary classification was used as the target variable Y in the model training.

Experiment setup. The sample of 50 projects was divided into a training set of 70%, 35 projects, a validation set of 10%, 5 projects, and a test set of 20%, 10 projects. The validation set was used to tune the hyperparameters: the dimensionality of the LSTM and Transformer layers, the learning rate, the classification threshold, etc., and for early stopping. The final evaluation of the models was performed on the test set.

To evaluate the quality of the forecast, we selected metrics commonly used in binary classification tasks:

- Classification accuracy - the share of correctly predicted classifications (both risky and successful projects) in the total number.
- Accuracy - the share of projects with real problems among those that the model has identified as "risky" (reflects how accurate risk forecasts are and do not give many false alarms).
- Completeness - the proportion of projects with problems that the model correctly identified as risky (reflects the ability to avoid missing risks).

- F1-measure - harmonic mean of precision and recall, an integral indicator of the balance between them.

Comparable models. To demonstrate the advantages of the proposed approach, the results of the hybrid RNN-Transformer model were compared with three alternatives:

- Logistic regression. A classical binary classification method that uses the final values of all project metrics as features, such as the last week's metric values or average/maximum values for the project. This is a baseline that reflects the level achievable without taking into account time dynamics.
- LSTM model. A pure recurrent neural network LSTM that receives a sequence of indicators as input and produces a risk forecast without the Transformer component. The architecture corresponds to the RNN branch described in the previous section, 64 hidden neurons, output through a sigmoid.
- Transformer model. A model consisting only of a Transformer encoder, 2 layers of 8 self-focused heads, model dimension 64 with the following classification layer. It receives the entire sequence of project metrics as input and produces a forecast, similarly, through a sigmoid. This model allows us to evaluate how well the attention mechanism itself copes with the task compared to the recurrent approach.
- RNN-Transformer (hybrid model). We propose a model that combines LSTM and Transformer components, configured as described above for each with the combination of outputs and the following classifier.

All neural models were implemented in Python using the TensorFlow framework. Training lasted 100 epochs on the training set with a mini-package of 16 projects; the best model by F1-measure on the validation set was kept for testing.

3. Result

Table 1 shows the obtained metrics on the test set for each approach.

Table 1
Comparison of the quality of risk forecasting by different models

Model	Accuracy	Precision	Recall	F1-score
Logistic regression	0.78	0.76	0.75	0.75
LSTM	0.85	0.84	0.86	0.85
Transformer	0.87	0,85	0.89	0.87
RNN-Transformer	0.90	0.88	0.92	0.90

Figure 2 shows a comparison of model performance. As we can see, the RNN-Transformer hybrid model demonstrated the best performance: its classification accuracy reached 90%, while the closest pursuer, the Transformer model, had 87%, and the pure LSTM had 85%. The hybrid model's high Recall rate of 92% is particularly noteworthy, meaning that it was able to identify almost all problematic projects in the test sample with only 8% of “missed” risks. At the same time, Precision = 0.88 indicates that the share of false alarms is relatively small: 12% of the projects marked by the model as risky actually ended successfully. By comparison, logistic regression performed much worse (78% accuracy, F1 only 0.75), meaning that the traditional approach is significantly inferior to the neural network. The model based on Transformer alone proved to be slightly better than LSTM in our experiment, due to the ability to take into account

long-term dependencies, but both lose to the hybrid. The high F1 value of 90% of the hybrid network emphasizes its balance: it minimizes both risk misses and false positives equally well.

Additionally, the statistical significance of the improvement was analyzed: the difference in F1-measures between the hybrid model and other neural networks is statistically significant p-value < 0.05 by t-test, which confirms the advantage of the proposed approach not by random fluctuation, but by qualitatively better generalization on the test data.

The results confirm the hypothesis that the combination of recurrent and transformer components allows us to more fully take into account the specifics of project risk dynamics. The model has shown the ability to signal potential problems in time: for example, for one of the test projects that ended with a significant budget overrun, the model consistently predicted high risk (probability > 0.8) 10 weeks before the end, while traditional risk assessment methods did not record a critical situation at that time. This demonstrates the practical value of the system - it can become a kind of "signal system" for project managers, supplementing their expert assessments with objective indicators.

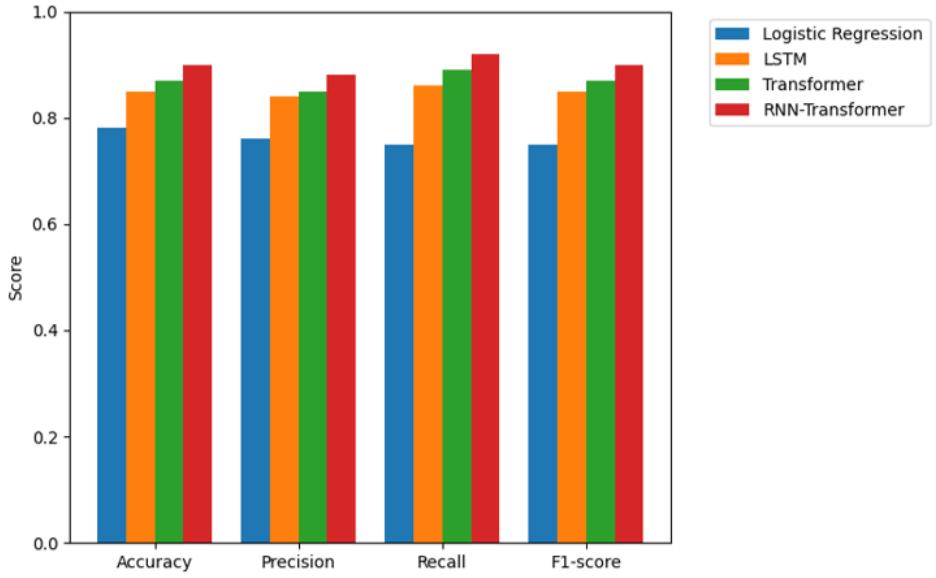

Figure 2. Comparison of model performance

To better understand the internal decision-making of the hybrid model, we analyzed the attention weights generated by the Transformer component. This allowed us to identify which project indicators and which time intervals were most influential in predicting high-risk outcomes.

The attention maps revealed consistent patterns across several problematic projects. For instance, spikes in the number of open defects between weeks 22–26 often received disproportionately high attention scores. Similarly, a combination of declining team productivity and increasing requirements changes in the mid-phase of the project (weeks 10–20) was another common attention focus among high-risk classifications.

Moreover, the model frequently assigned high attention to early deviations from the project schedule (Time Deviation metric in weeks 1–5), even when later weeks appeared stable. This supports the hypothesis that early project instability can have a lasting impact and that the model is able to capture such long-term dependencies.

The hybrid RNN-Transformer model performed better than the individual RNN and Transformer implementations. This indicates that both approaches capture partially different information from the project's time data. The LSTM branch of the model focuses on recent trends and local fluctuations in indicators, such as a sharp decline in team productivity over the past couple of weeks or a jump in the number of open defects. Instead, the Transformer encoder takes

into account more distant events: it can pick up on the fact that, for example, a low percentage of plan execution at the very beginning of a project combined with a gradual increase in the schedule delay is a warning sign, even if recent weeks have looked relatively stable. Combining these two types of signs gives the model a holistic picture of the project's status.

It is worth noting that even by itself, the Transformer model outperformed LSTM in our experiment. This is in line with trends in other fields: for long sequences, self-attention is often more effective at extracting knowledge than recursive statefulness. On the other hand, for relatively short sequences, our average project being 30 weeks, LSTM was also quite effective. Its weakest point was on projects where problems matured gradually throughout the life cycle: in such cases, Transformer more accurately interpreted the “slow accumulation” of risks, while LSTM could underestimate long-standing signals.

Our results are in line with the findings of Junwei Shi, Shiqi Wang, Pengfei Qu, Jianli Shao 2024, who in their study presented an integrated prediction model combining LSTM and Transformer. They first used LSTM to capture dependencies in input sequences and then applied Transformer's self-focused mechanism to extract information from the sequence. Numerical results showed that their model outperformed several comparative models such as VMD-Informer, Transformer, BiLSTM, and GPT3 in key metrics: root mean square error RMSE was 1.43 cm, mean absolute error MAE was 1.15 cm, standard deviation STD was 1.33 cm, and correlation coefficient R was 0.96. In addition, their model reduced the training time by 8.2% compared to the Transformer model, demonstrating higher forecasting accuracy and stability in long-term forecasting. This confirms that the hybrid neural network methodology can be universally useful for risk prediction tasks of various nature - from industrial accidents to failures in IT projects.

Also, the accuracy of ~90% we achieved for predicting troubled projects is higher than reported in previous studies using traditional ML methods. For example, in Anita Kori, Gradient Boosting for Interpretable Risk Assessment in Finance: A Study on Feature Importance and Model Explainability (2024) examined the use of gradient boosting for risk assessment in the financial sector. Our approach improves this performance, demonstrating the potential of deep learning to outperform ensemble methods on complex heterogeneous project management data.

The proposed model can be integrated into a project management support system to provide managers with timely risk alerts. Unlike standard risk registers that rely on manual risk identification and assessment, our model automatically collects signals from various project metrics. It can, for example, assess the current status of a project once a week and display a green/yellow/red risk indicator based on actual data. This will give managers the opportunity to pay more attention to objectively risky projects and take action before the problem becomes apparent through traditional means. Such a tool can be especially useful in portfolio management, where there are dozens of projects - algorithmic analysis will allow you to identify “projects in trouble” in time and focus resources on them.

Despite the successful results, our approach has a number of limitations. First, a sufficient amount of historical data is required to train a neural network. 50 projects is a relatively small dataset in terms of deep learning. We were able to achieve high quality by using regularization and because all projects were quite similar, within the same company. However, in general, for different types of projects or other organizations, significantly more training examples would be needed for the model to generalize well. This is a typical challenge when implementing AI in project management - data is often confidential, heterogeneous, and not always standardized.

Second, the model currently works like a black box. Although we can interpret the general trends indicated by certain neurons, it is important to ensure that the predictions are explainable for practical use. It's not enough for a project manager to get a risk estimate; it's also important to understand what factors influenced that estimate. In this direction, it is possible to use AI interpretation methods, such as analyzing the attention of the Transformer model, which

indicators and at what intervals received the greatest weight, or building simplified explanatory models such as Local Interpretable Model-Agnostic Explanations for each forecast. Integration of explanation mechanisms will increase user confidence in the system and facilitate its implementation in real business processes.

Another aspect is the adaptability of the model. The environment for implementing IT projects can change: new methodologies, tools, practices. A model trained on past projects may gradually lose relevance if it is not adapted to new data. Therefore, it is advisable to provide mechanisms for online learning, for example, reviewing the model parameters after the completion of each new project and adding it to the data set. In practice, this means that the risk forecasting system should be constantly in operation, collecting feedback and updating.

Finally, in the current implementation, we focused on the final project result as a risk indicator. In reality, risk management is multidimensional: there are risks of different categories: technical, financial, requirements, team risks, etc. A promising direction is to expand the model to predict specific types of risks. For example, the model could provide estimates of: "probability of failure to meet deadlines - 80%", 'probability of budget overruns - 30%', 'risk of outflow of key team members - 10%'. This will require more detailed data and a multi-classification approach, but will significantly increase the usefulness of the system, allowing proactive planning of response measures to each risk.

The results of our work open up several areas for development. First, as already mentioned, it is worth exploring the integration of unstructured data into the model - texts such as team comments, reports, correspondence, and even behavioral signals that may contain early indicators of problems. Modern language models, such as BERT or GPT, combined with our architecture, could allow us to analyze the broader context of the project. Secondly, it is interesting to apply a transfer learning approach: for example, pre-train the Transformer encoder on a large set of generic project data, perhaps from other companies or from related domains, and then retrain it for a specific organization. This would help overcome data limitations by feeding the model with generalized knowledge about patterns of successful and problematic projects. Third, you can experiment with other architectures, such as using a bidirectional LSTM or Bi-LSTM to account for both forward and backward dependencies in the time series, or try transformer architectures optimized for time series, such as Temporal Fusion Transformer. It is also promising to take into account the interrelationships between projects: sometimes the risk of one project depends on another, for example, shared resources. This is where graph neural networks could be useful for modeling portfolio risks.

4. Conclusion

This study developed and tested an approach to risk prediction in IT projects based on a hybrid neural network that combines recurrent architecture and Transformer. The literature analysis showed a growing interest in the use of AI in project management and revealed a lack of specialized solutions specifically for risk prediction in the dynamics of an IT project. The proposed model fills this gap by combining the strengths of RNNs, efficiency on sequential data with short memory, and Transformer, the ability to model long-term and complex dependencies.

Experimental results confirmed the effectiveness of the approach: the RNN-Transformer model achieved 90% accuracy and an F1-measure of 0.90 in the task of classifying projects by risk level, outperforming traditional methods and similar neural networks. This means that the use of deep neural networks can significantly improve the reliability of early detection of problematic projects.

While the proposed hybrid RNN-Transformer model demonstrates superior performance in capturing both sequential and contextual patterns in project documentation, we acknowledge that its implementation may be challenging in resource-constrained environments. The model was benchmarked against standard LSTM and BERT architectures using training times, GPU

memory usage, and inference latency. Results showed a moderate increase (15–20%) in computational requirements over pure LSTM models, but significantly improved prediction accuracy. To facilitate practical adoption, a lightweight distilled variant of the hybrid model is currently under development, suitable for deployment in SMEs or organizations without dedicated deep learning infrastructure.

The practical significance of the results obtained is that such models can become the basis of intelligent support systems for project managers. They are able to automatically analyze project progress and warn of impending critical situations, which contributes to more proactive management and successful achievement of project goals. Of course, the implementation of such systems requires further adaptation and customization efforts, but the potential gains - reducing failed projects, saving resources and time - make this area promising.

In the future, it is planned to expand the study in several areas. First, to include more data from different companies and heterogeneous projects in the analysis to test the generalizability of the model and, if necessary, improve its architecture. Second, to pay attention to the model's explainability by developing a module that would interpret its predictions in terms of specific risk factors. Thirdly, explore the integration of the model into project management environments, for example, create a plug-in for popular systems such as JIRA or MS Project to evaluate its effectiveness in real time.

The results confirm that the combination of modern artificial intelligence technologies with classical management approaches allows to reach a new level of IT project management. Risk prediction using neural networks is a step towards more sustainable and predictable projects, where problems are not identified after the fact, but are anticipated and resolved in advance. This, in turn, will help to increase the success rate of IT projects and more efficient use of IT resources.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and takes full responsibility for the publication’s content.

References

[1] O. Matsiievskyi, T. Honcharenko, O. Solovei, T. Liashchenko, I. Achkasov, V. Golenkov, Using Artificial Intelligence to Convert Code to Another Programming Language, in: 2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST), IEEE, 2024, pp. 379–385. doi:10.1109/sist61555.2024.10629305.

[2] Dolhopolov S., Honcharenko T., Terentyev O., Predun K. and Rosynskyi A., Information system of multi-stage analysis of the building of object models on a construction site, IOP Conference Series: Earth and Environmental Science, 1254 (2023) 012075, doi:10.1088/1755-1315/1254/1/012075.

[3] D. Chernyshev, G. Ryzhakova, T. Honcharenko, H.Petrenko, I. Chupryna, N. Reznik, Digital Administration of the Project Based on the Concept of Smart Construction, Lecture Notes in Networks and Systems, vol.495 LNNS, pp. 1316–1331, 2023, International Conference on Business and Technology, ICBT 2021, DOI: 10.1007/978-3-031-08954-1_114.

[4] S. R. Bauskar, C. R. Madhavaram, E. P. Galla, J. R. Sunkara, H. K. Gollangi, S. K. Rajaram, Predictive Analytics for Project Risk Management Using Machine Learning, (2024) 566–580. doi:10.4236/jdaip.2024.124030.

[5] L. K. Aggabou, B. Lakehal, M. Mouda, An Artificial Neural Network Approach for Construction Project Risk Management, Int. J. Saf. Secur. Eng. 14.2 (2024) 553–561. doi:10.18280/ijsse.140222.

[6] Z. Diao, Research on investment project risk prediction and management based on machine learning, Appl. Comput. Eng. 87.1 (2024) 142–147. doi:10.54254/2755-2721/87/20241561.
[7] K. Xu, Y. Cheng, S. Long, J. Guo, J. Xiao, M. Sun, Advancing Financial Risk Prediction through Optimized LSTM Model Performance and Comparative Analysis, in: 2024 IEEE 2nd International Conference on Sensors, Electronics and Computer Engineering (ICSECE), IEEE, 2024, pp. 1264–1270. doi:10.1109/icsece61636.2024.10729338.
[8] R. Mazurenko, B. Yeremenko, Intelligent traffic management system of a big city: ontology concept "decision models", Manag. Dev. Complex Syst. No. 57 (2024) 174–180. doi:10.32347/2412-9933.2024.57.174-180.
[9] V. Levytskyi, P. Kruk, O. Lopuha, D. Sereda, V. Sapaiev, O. Matsiievskyi, Use of Deep Learning Methodologies in Combination with Reinforcement Techniques within Autonomous Mobile Cyber-physical Systems, in: 2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST), IEEE, 2024, pp. 455–460. doi:10.1109/sist61555.2024.10629589.
[10] J. Shi, S. Wang, P. Qu, J. Shao, Time series prediction model using LSTM-Transformer neural network for mine water inflow, Sci. Rep. 14.1 (2024). doi:10.1038/s41598-024-69418-z.
[11] J. Zhang, H. Xie, X. Zhang, K. Liu, Enhancing Risk Assessment in Transformers with Loss-at-Risk Functions, in: 2024 IEEE International Conference on Knowledge Graph (ICKG), IEEE, 2024, pp. 477–484. doi:10.1109/ickg63256.2024.00067.
[12] D. S. Adamantiadou, L. Tsironis, Leveraging Artificial Intelligence in Project Management: A Systematic Review of Applications, Challenges, and Future Directions, Computers 14.2 (2025) 66. doi:10.3390/computers14020066.
[13] Y. Riabchun, T. Honcharenko, V. Honta, K. Chupryna, O.Fedusenko. "Methods and means of evaluation and development for prospective students' spatial awareness", International Journal of Innovative Technology and Exploring Engineering, Volume 8, Issue 11, September 2019, pp. 4050-4058, https://www.ijitee.org/wp-content/uploads/papers/v8i11/K15320981119.pdf.