

# **A Hybrid Predictive Maintenance Framework Using Risk-Based and Deep Learning Approaches**

**Hüseyin Ali Sarıkaya**

Assist. Prof. Dr., Industrial Engineering Department  
Mudanya University, Bursa, Türkiye  
[huseyin.sarikaya@mudanya.edu.tr](mailto:huseyin.sarikaya@mudanya.edu.tr)

## **Abstract**

This study proposes a hybrid predictive maintenance framework for Computer Numerical Control machining systems by integrating risk-based prioritization with deep learning-based time-series forecasting. The framework combines Fine-Kinney risk assessment and multi-criteria decision-making using the Technique for Order Preference by Similarity to Ideal Solution to identify and rank critical failure modes. Subsequently, Recurrent Neural Network, Long Short-Term Memory, and Gated Recurrent Unit models are developed to predict failure intervals based on historical maintenance data. The methodology is validated using real-world failure data collected from 17 CNC machines over a two-year period. The results indicate that Long Short-Term Memory and Gated Recurrent Unit models outperform classical Recurrent Neural Networks in terms of prediction accuracy, achieving significantly lower error rates and higher correlation with actual failure patterns. In addition, the integration of risk prioritization ensures that predictive modeling efforts are focused on the most critical failure modes, enhancing both analytical efficiency and practical relevance. To assess real-world applicability, a numerical cost-benefit analysis is conducted. The findings show that even a conservative reduction of 20% in unplanned downtime can yield a return on investment exceeding 200%, demonstrating the strong economic feasibility of the proposed approach. Overall, the study provides a comprehensive and scalable predictive maintenance solution that combines technical performance with economic impact, offering a practical decision-support tool for small and medium-sized manufacturing enterprises.

## **Keywords**

Predictive Maintenance, Deep Learning, Cost Reduction, CNC Machines, Risk Identification.

## **1. Introduction**

Maintenance activities constitute a substantial portion of manufacturing costs, often accounting for 60–70% of total production expenses. Unexpected equipment failures can lead to severe production interruptions, financial losses, and reduced operational efficiency. Traditional maintenance strategies, such as reactive or time-based preventive maintenance, are often insufficient, as they either respond too late or result in unnecessary interventions. In contrast, predictive maintenance has emerged as a data-driven approach that enables condition-based decision-making and early fault detection. Studies indicate that predictive maintenance can reduce maintenance costs by up to 30% and prevent nearly 75% of equipment failures (Matyas et al., 2017).

Despite these advantages, small and medium-sized enterprises (SMEs) face significant challenges in adopting predictive maintenance due to limited digital infrastructure, lack of standardized data collection, and constrained resources. As a result, there is a growing need for cost-effective, scalable, and data-efficient predictive maintenance frameworks specifically designed for resource-constrained industrial environments.

Beyond technical performance, the economic feasibility of predictive maintenance solutions has become a critical concern, particularly for small and medium-sized enterprises operating under tight budget constraints. While many existing studies emphasize improvements in prediction accuracy, relatively limited attention has been given to

quantifying the financial benefits of such approaches. In practice, the adoption of predictive maintenance strategies is largely driven by their ability to reduce unplanned downtime, optimize resource utilization, and generate measurable returns on investment. Therefore, incorporating an economic perspective into predictive maintenance frameworks is essential to support real-world implementation and managerial decision-making.

Accordingly, this study not only evaluates predictive accuracy but also integrates a quantitative cost–benefit analysis to demonstrate the economic impact and practical feasibility of the proposed approach.

This study proposes a hybrid predictive maintenance framework that integrates quantitative risk-based prioritization with advanced deep learning-based time-series forecasting for Computer Numerical Control (CNC) machinery. The proposed framework combines Fine-Kinney risk assessment and multi-criteria decision analysis through Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) with recurrent neural network (RNN)-based architectures, including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models, to enhance failure prediction accuracy and maintenance decision-making.

By leveraging real-world industrial failure data and combining qualitative and quantitative decision-support mechanisms, the study aims to provide a structured, scalable, and practically applicable solution that bridges the gap between theoretical predictive maintenance models and real-world industrial implementation constraints. The framework systematically integrates failure data analysis, risk prioritization, and deep learning-based forecasting into a unified decision-support structure, ensuring that predictive modeling efforts are aligned with operational priorities and economic impact.

Unlike conventional approaches that focus either on predictive accuracy or risk assessment independently, this study contributes by combining these dimensions within a single hybrid framework and extending the analysis to include economic performance evaluation. This integrated perspective enhances both the practical relevance and managerial applicability of predictive maintenance solutions.

The remainder of this paper is organized as follows. Section 2 reviews the relevant literature on predictive maintenance, deep learning models, and risk-based decision-making approaches. Section 3 presents the proposed methodology, including data collection, failure classification, risk assessment, and time-series modeling procedures. Section 4 reports the results and discusses their practical implications, including cost–benefit and operational considerations. Section 5 concludes the study and outlines future research directions.

## **2. Literature Review**

Predictive maintenance has gained significant attention in recent years due to rapid advancements in machine learning, artificial intelligence, and industrial digitalization technologies. The increasing availability of sensor data and computational power has enabled the development of data-driven maintenance strategies that outperform traditional approaches in terms of cost efficiency and reliability.

The existing body of literature can be broadly categorized into three main streams: (i) RNN-based predictive modeling approaches, (ii) hybrid and Internet of Things (IoT)-enabled maintenance systems, and (iii) risk-based maintenance prioritization and multi-criteria decision-making (MCDM) methods. This classification enables a structured understanding of the current research landscape and highlights the limitations that motivate the present study.

### **2.1 RNN-Based Predictive Maintenance Models**

RNNs are widely used for time-series analysis due to their ability to capture temporal dependencies. Unlike feed-forward neural networks, RNNs incorporate feedback connections, allowing them to retain historical information. The concept of training recurrent networks using backpropagation through time was introduced by Werbos (1988), followed by the development of the Elman network (Elman, 1990).

However, traditional RNNs suffer from vanishing gradient problems, which limit their ability to model long-term dependencies. To address this limitation, Hochreiter and Schmidhuber (1997) proposed LSTM networks, which utilize gating mechanisms and memory cells to preserve long-term information. Later, Cho et al. (2014) introduced GRUs, which provide a simplified yet efficient alternative to LSTM models.

Recent studies have further extended these architectures by incorporating attention mechanisms, hybrid deep learning structures, and transfer learning approaches to improve predictive accuracy in complex industrial environments. For instance, Transformer-based architectures and hybrid recurrent-convolutional models have demonstrated superior performance in capturing nonlinear temporal dependencies.

Numerous studies have demonstrated the effectiveness of these models in predictive maintenance. Zhang et al. (2018) applied LSTM networks to predict battery remaining useful life, significantly improving reliability. Jamal et al. (2020) showed that LSTM models outperform traditional neural networks in maintenance prediction tasks. Similarly, Zainuddin et al. (2021) achieved high prediction accuracy in oil and gas systems using GRU-based models. These studies highlight the superiority of advanced RNN architectures in handling sequential industrial data.

Despite these advancements, most studies focus primarily on prediction accuracy and often neglect the integration of operational risk factors into the modeling framework. This limitation reduces the practical applicability of such models in real-world industrial decision-making processes.

## **2.2 Hybrid and IoT-Based Predictive Maintenance Approaches**

Recent research has focused on integrating RNN models with IoT systems and advanced analytics to enhance predictive maintenance performance. Killeen and Patrick (2020) proposed an IoT-based architecture with dynamic sensor selection, improving prediction accuracy. Wu et al. (2021) incorporated explainable AI techniques, such as Layer-wise Relevance Propagation, to enhance model interpretability in bearing monitoring applications.

Moreover, hybrid approaches combining machine learning with optimization and simulation techniques have been widely explored. Adamou et al. (2023) integrated RNN models with digital twin technology and genetic algorithms to achieve energy-efficient maintenance strategies. Harkat et al. (2023) demonstrated the effectiveness of LSTM models in improving compressor system uptime within Industry 4.0 environments. More recently, Dintén and Zorrilla (2024) developed a Transformer-GRU hybrid model with explainable AI, achieving highly accurate fault predictions using IoT-generated datasets.

Although these hybrid systems significantly improve predictive performance, they often require high-cost infrastructure, large-scale sensor networks, and advanced computational resources. This creates a barrier for SMEs, which typically operate under limited technological and financial constraints.

## **2.3 Risk-Based Maintenance and Multi-Criteria Decision Methods**

In addition to predictive modeling, risk analysis plays a crucial role in maintenance decision-making. Identifying and prioritizing critical failure modes ensures efficient allocation of maintenance resources. The Fine-Kinney method (Fine, 1971) is widely used for quantitative risk assessment, evaluating risk based on probability, frequency, and severity.

To handle uncertainty and multiple evaluation criteria, risk assessment is often combined with MCDM methods. Among these, TOPSIS, introduced by Hwang and Yoon (1981), is one of the most widely applied methods for ranking alternatives based on their closeness to the ideal solution.

Recent studies have extended classical MCDM approaches by incorporating fuzzy logic, intuitionistic fuzzy sets, and spherical fuzzy environments to better handle uncertainty in expert evaluations. These extensions enhance decision robustness but often increase model complexity.

However, existing risk-based approaches are typically applied independently of predictive modeling techniques. As a result, they fail to capture the dynamic nature of failure evolution over time, limiting their effectiveness in predictive maintenance applications.

## **2.4 Research Gap and Contribution**

Although extensive research exists on predictive maintenance and risk assessment individually, limited studies have integrated these approaches into a unified framework. Most existing works either focus on machine learning-based prediction or risk prioritization separately. This study addresses this critical gap by proposing a hybrid predictive maintenance framework that systematically integrates risk assessment and deep learning-based forecasting into a

unified decision-support structure. Unlike conventional approaches, the proposed framework ensures that failure prediction models are directly guided by operational risk priorities.

Furthermore, the study introduces a cross-platform implementation strategy by developing and comparing models in both MATLAB and Python environments. This dual implementation enhances methodological robustness and ensures reproducibility across different industrial settings.

Another key contribution of this study is its focus on SME environments, where data availability, computational resources, and infrastructure are limited. By designing a scalable and data-efficient framework, the study provides a practical solution that bridges the gap between advanced predictive maintenance techniques and real-world industrial constraints.

### **3. Methods**

This study proposes a structured hybrid methodology that integrates data-driven failure analysis, multi-criteria risk prioritization, and deep learning-based time-series forecasting. The overall framework consists of three main stages: (i) failure data processing and classification, (ii) risk-based prioritization using Fine-Kinney and multi-criteria decision analysis, and (iii) time-series prediction using recurrent neural network architectures.

#### **3.1 Case Study: Data Collection and Failure Classification**

The research was carried out at an automotive parts manufacturing company in Konya that operated 17 CNC machines. Maintenance logs were gathered during a two-year period (2022-2024), containing the date of each failure incidence and a brief description. Initially, the failure data was unstructured, making analysis difficult. We used a failure classification technique to bring together similar failure instances. All reported problems were divided into 13 key areas based on expert consultation and maintenance record review: coolant system, switch, spindle, conveyor, hydraulic, screw, ruler, axis, automatic pallet changer (APC), air pressure, slide, automatic tool changer (ATC), and table. This segmentation made targeted analysis easier because each failure category had unique causes and maintenance practices. Then, we conducted a failure occurrence analysis to identify the frequency of each failure category. Table 1 highlights the failure ratio by reason. The cooling system, ATC, table, and conveyor were the most common failures, accounting for 16-17% of total failures. Switches, rulers, APCs, and air pressure have modest occurrence rates (~2%). These frequency ratios indicate which failure modes are frequent and thus viable candidates for predictive modeling.

Table 1. Ratio of individual failure causes to total number of failures

| Failure Type | Occurrences | Occurrence Ratio |
|--------------|-------------|------------------|
| Coolant      | 50          | 0.1701           |
| Switch       | 7           | 0.0238           |
| Spindle      | 21          | 0.0714           |
| Conveyor     | 47          | 0.1599           |
| Hydraulic    | 21          | 0.0714           |
| Screw        | 9           | 0.0306           |
| Ruler        | 6           | 0.0204           |
| Axis         | 12          | 0.0408           |
| APC          | 6           | 0.0204           |
| Air Pressure | 7           | 0.0238           |
| Slide        | 14          | 0.0476           |
| ATC          | 46          | 0.1565           |
| Table        | 48          | 0.1633           |

### Mean Time to Failure (MTTF) Calculation

MTTF is a fundamental reliability metric that estimates the expected lifespan of non-repairable components. It is frequently used in predictive maintenance and asset management (Ebeling 2019). The MTTF for each failure type was obtained by averaging the intervals between consecutive failure occurrences, as stated in Equation (1) for timestamps:  $t_1 < t_2 < \dots < t_N$

$$MTTF = \frac{1}{N-1} \sum_{i=1}^{N-1} (t_{i+1} - t_i) \quad (1)$$

We also normalized these figures to evaluate relative failure rates by calculating the ratio of each MTTF to the total. Table 2 presents a summary of the MTTF outcomes. Air-pressure system failures had the longest MTTF (~148 days) due to their rarity but severity, while coolant system failures had a shorter MTTF (~17.6 days) due to their high frequency. These calculations measure the frequency with which each fault occurs, forming the foundation for risk grading.

Table 2. MTTF and corresponding ratios

| Failure Type | MTTF   | Ratio |
|--------------|--------|-------|
| Coolant      | 17.60  | 0.03  |
| Switch       | 54.75  | 0.09  |
| Spindle      | 26.75  | 0.04  |
| Conveyor     | 18.06  | 0.03  |
| Hydraulic    | 39.31  | 0.06  |
| Screw        | 73.20  | 0.12  |
| Ruler        | 41.33  | 0.07  |
| Axis         | 50.67  | 0.08  |
| APC          | 68.75  | 0.11  |
| Air Pressure | 147.75 | 0.24  |
| Slide        | 50.30  | 0.08  |
| ATC          | 12.94  | 0.02  |
| Table        | 17.66  | 0.03  |
| Total        | 619.06 | 1.00  |

To ensure a robust and comprehensive prioritization of failure modes, we used a two-stage approach that combined the Fine-Kinney method with TOPSIS, a MCDM method for evaluating and ranking alternatives based on their proximity to an ideal solution (Hwang and Yoon, 1981). The Fine-Kinney method uses the standard probability-frequency-severity framework to quantify overall risk, whereas the TOPSIS method allows for a more fine-grained rating based on several operational parameters. Because both methods are standard and well documented in the literature, only the application-specific settings used in this study are summarized here. TOPSIS was used to derive a composite severity score, and Fine-Kinney was subsequently used to combine severity with occurrence and frequency information for final prioritization. Integrating these methodologies ensures that the key failure types chosen are not only statistically prevalent, but also operationally significant.

### Multi-Criteria Ranking with TOPSIS

In the first stage, we applied the TOPSIS method, introduced by Hwang and Yoon (1981), to incorporate multiple criteria reflecting the practical consequences of failures. For each fault type, three criteria were considered: detectability (ease of detecting the fault), repair cost, and MTTF.

Each criterion was rated on a scale of 1 (best) to 5 (worst) based on expert judgments by a panel of five firm professionals, comprising one maintenance and repair engineer, two CNC maintenance specialists, and two production supervisors. To make room for different degrees of skill, evaluator weights were assigned as follows: maintenance and repair engineer (5 units), CNC maintenance specialists (3 units each), and production supervisors (1 unit each). The normalized criteria weights, developed by expert panel consensus to reflect management priorities, were the following:

- Detectability: 0.2
- Repair Cost: 0.5
- MTTF: 0.3

The TOPSIS method was applied following the standard procedure described in the literature (Hwang and Yoon, 1981), and detailed computational steps are not repeated here for brevity. Accordingly, only the case-specific criteria, expert weighting structure, and ranking results are reported in this manuscript. This approach enabled the ranking of failure modes based on their relative closeness to the ideal solution, providing a structured and objective basis for prioritization. The results of the TOPSIS evaluation are shown in Table 3.

Table 3. TOPSIS results for fault severity

| Failure Type | TOPSIS Result |
|--------------|---------------|
| Coolant      | 0.83          |
| Switch       | 0.67          |
| Spindle      | 0.00          |
| Conveyor     | 0.31          |
| Hydraulic    | 0.32          |
| Screw        | 0.61          |
| Ruler        | 0.59          |
| Axis         | 0.42          |
| APC          | 0.51          |
| Air Pressure | 0.98          |
| Slide        | 0.74          |
| ATC          | 0.59          |
| Table        | 0.72          |

From the table, air pressure failures (0.98), coolant failures (0.83), slide failures (0.74), and table failures (0.72) emerged as the top-ranked critical issues based on the composite multi-criteria evaluation. These findings demonstrate that the integration of multiple operational criteria provides a more comprehensive assessment of failure criticality (severity) compared to single-metric evaluations.

#### **Fine–Kinney Risk Assessment**

Following the TOPSIS-based severity rating, the data were combined with the Fine-Kinney (Fine, 1971) risk scoring methodology to produce a final criticality evaluation. Although widely employed in a variety of sectors, the classic Fine-Kinney method is significantly dependent on subjective expert assessment, which results in limited measurability

and growing bias (Gökler et al., 2022). Recent studies suggest adjustments to overcome these constraints (Ilbahar et al., 2018). In this study, the original formula was modified to improve objectivity and provability.

The Fine–Kinney method evaluates risk based on three fundamental dimensions: probability, frequency, and severity, and is widely used in industrial risk assessment studies. In this study, instead of relying solely on subjective expert judgments, the Fine-Kinney framework was adapted to incorporate data-driven inputs.

The risk score is calculated as shown in Equation (2):

$$\text{Risk Score} = P \times F \times S \quad (2)$$

where:

- **Probability** was derived from the historical occurrence ratio of each failure type (based on the 294-failure dataset),
- **Frequency** was represented by the normalized MTTF (shorter MTTF → higher frequency),
- **Severity** was determined by the TOPSIS composite score.

This combination of empirical occurrence data, failure timing, and multi-criteria severity ensures a comprehensive evaluation that accounts for both operational impact and statistical frequency. The resulting final Fine–Kinney risk scores guided the selection of the most critical fault modes for predictive modeling (e.g., coolant, table, and ATC faults), forming the basis for subsequent RNN, LSTM, and GRU forecast analyses. The calculated risk scores and priority rankings are presented in Table 4.

Table 4. Fine-Kinney risk scores and priority rankings

| Failure Type | Risk Score (R) | Priority |
|--------------|----------------|----------|
| Coolant      | 0.0041         | 2        |
| Switch       | 0.0014         | 9        |
| Spindle      | 0.0000         | 13       |
| Conveyor     | 0.0015         | 8        |
| Hydraulic    | 0.0014         | 10       |
| Screw        | 0.0021         | 5        |
| Ruler        | 0.0008         | 12       |
| Axis         | 0.0015         | 7        |
| APC          | 0.0011         | 11       |
| Air Pressure | 0.0056         | 1        |
| Slide        | 0.0032         | 4        |
| ATC          | 0.0020         | 6        |
| Table        | 0.0033         | 3        |

The risk assessment indicated air pressure problems as the top risk (highest risk score  $\approx 0.0056$ ), despite their low frequency, due to their high severity (machine stoppage affecting several systems). The next greatest risk scores were assigned to cooling system failures (0.0041) and table faults (0.0033), both of which were common and moderately severe. ATC failures had a slightly lower risk (scoring  $\approx 0.002$ -0.003) due to their frequent but less severe impact (mostly generating production delays). Based on these observations, the corporation opted to prioritize air pressure, coolant, table, and ATC concerns in their maintenance plan. However, for predictive modeling, we concentrate on the coolant, table, and ATC failure types. This is because, while air pressure problems were crucial, they occurred too seldom (just a few times) to train a useful time-series model. In contrast, coolant, table, and ATC each had dozens of

failure incidents, providing enough data to learn temporal patterns. These three fault types reflect distinct operational patterns: coolant-related issues occurred frequently, table-related failures appeared with medium frequency, whereas ATC malfunctions emerged irregularly and without a clear pattern. Therefore, including these fault groups enables the model to represent a broad spectrum of failure behaviors.

### **3.2 Time-Series Modeling for Failure Prediction**

We developed a time-series prediction problem for each failure category, given the recent history of failure occurrences (or intervals between failures), predict the time until the next failure. We define the failure interval sequence for a particular category as  $\{y(t_i)\}_{i=1}^N$ , where  $y(t_i)$  is the time (in days) between the  $i$ th and  $(i + 1)$ th failure of that type. Our goal is to predict  $y(t_{i+1})$  (the next interval) based on the past  $p$  intervals  $y(t_i), y(t_{i-1}), \dots, y(t_{i-p+1})$ . The prediction task is formulated as an autoregressive modeling problem in which the next failure interval is estimated based on the previous observed intervals.

$$\hat{y}(t_{i+1}) = f(y(t_i), y(t_{i-1}), \dots, y(t_{i-p+1}))$$

where  $f(\cdot)$  is a nonlinear function approximated by an RNN-based model, and  $\hat{y}(t_{i+1})$  is the predicted next failure interval.

We examined the autocorrelation and partial autocorrelation of the interval sequences. The autocorrelation functions indicated that correlation declined after a few lags. Based on this analysis, empirical validation, and the need to control dimensionality under a limited sample size, the input window size was fixed at  $p = 3$ , ensuring a balance between capturing temporal dependencies and avoiding overfitting in a small-sample setting. Pilot trials with larger windows reduced the number of usable training sequences without producing a consistent validation improvement, whereas shorter windows captured less temporal context.

### **3.3 Time-Series Modeling Framework**

This section describes the implementation details of the predictive models, including architecture design, data preprocessing, training strategy, and performance evaluation, to ensure full reproducibility.

#### **3.3.1 Model Architecture and Hyperparameters**

The architectures of the implemented models were explicitly defined to ensure fair comparison and reproducibility. We developed and compared three types of recurrent neural network models for failure interval prediction: RNN, LSTM, and GRU models.

The classical RNN model was implemented as a nonlinear autoregressive (NAR) time-series network in MATLAB. The NAR network utilizes only past observations of the target series as inputs, without exogenous variables, by introducing feedback delays from the output layer. The model was configured with a single hidden layer consisting of ten neurons and feedback delays ranging from one to three time steps. The hidden layer employed a hyperbolic tangent (tansig) activation function, while the output layer used a linear activation function for regression. The MATLAB Neural Network Time Series Tool (nftool) was used to implement the model, initially in open-loop form during training and later converted to closed-loop form for multi-step forecasting.

LSTM networks extend classical RNNs by incorporating memory cells and gating mechanisms (input, forget, and output gates) to effectively capture long-term dependencies and mitigate vanishing gradient problems. The LSTM model was implemented in Python using the Keras framework as a sequence-to-one architecture. It consists of a single LSTM layer with ten units, followed by a dense output layer with linear activation for regression. The input to the model consists of sequences of three normalized failure intervals, allowing the network to encode temporal patterns into its internal memory state. The gating structure enables the model to retain relevant information over time and selectively update its internal state, thereby capturing complex temporal relationships that may not be effectively modeled by classical RNNs.

The GRU model simplifies the LSTM architecture by combining the input and forget gates into a single update gate, along with a reset gate, thereby reducing computational complexity while maintaining the ability to model long-term dependencies. The GRU model was implemented with a structure similar to the LSTM, consisting of a single recurrent

layer with ten units followed by a dense output layer. Due to its simplified gating mechanism, the GRU model typically requires fewer parameters and can achieve faster training while maintaining comparable predictive performance.

To ensure a fair and consistent comparison, all models were configured with similar architectural complexity, each containing ten recurrent units and a single hidden layer. This configuration was selected based on preliminary experiments and prior studies, providing sufficient capacity to capture temporal patterns while avoiding overfitting given the limited dataset size. The use of ten recurrent units was therefore a deliberately compact choice intended to preserve model expressiveness while limiting overfitting risk in sequences containing roughly 40–50 intervals per failure category; larger hidden sizes did not provide a stable validation improvement commensurate with the added complexity in our pilot tests. Each model produces a single output representing the predicted next failure interval in normalized form, which is later transformed back to the original scale for interpretation.

### 3.3.2 Data Preprocessing

The failure interval data for each category were relatively small datasets (on the order of 40–50 intervals each). To make the best use of the data, we combined the intervals from all machines for a given category into a single sequence (assuming that failures of the same type across different machines follow a similar process). Before feeding data to the neural networks, the data were normalized using min–max scaling to the range [0,1] to ensure numerical stability during training.

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

where  $x$  is an interval value, and  $x_{max}$ ,  $x_{min}$  are the minimum and maximum interval in the training set for that category. This transforms all inputs and targets to the range [0, 1]. Normalization is crucial for neural network training to ensure numerical stability and to avoid undue influence of large-valued outliers (such as a very long interval between failures). After model prediction, the outputs are inverse transformed back to days for interpretation. Additionally, the data was segmented into sequences of length 3 (for input) with the 4th point as output, creating supervised training samples. The dataset was then split chronologically into 70% for training, 15% for validation, and 15% for testing. The split was done by time such that the last 15% of the sequence (the most recent failures) served as the test set to evaluate how well the model can predict future unseen failures. The validation set (the 15% of data prior to the test segment) was used to tune hyperparameters and prevent overfitting, via early stopping where applicable.

### 3.3.3 Training Strategy

The classical RNN model was trained using the Levenberg–Marquardt optimization algorithm due to its fast convergence properties.

The LSTM and GRU models were trained using the Adam optimizer with a learning rate of 0.001.

- Epochs: up to 100
- Batch size: 1
- Early stopping: applied based on validation loss (patience = 10)

These training configurations were selected to ensure stable convergence and prevent overfitting.

### 3.3.4 Performance Evaluation

Model performance was evaluated using standard regression metrics, including MSE, RMSE, MAE, and the Pearson correlation coefficient (R) between predicted and actual intervals given by Equation (3-6). These metrics provide a comprehensive assessment of prediction accuracy and model reliability. These are defined as follows, for a set of  $n$  test samples with actual values  $y_i$  and predictions  $\hat{y}_i$ :

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (3)$$

$$RMSE = \sqrt{MSE} \quad (4)$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (5)$$

$$R = \frac{\sum_{i=1}^n (y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^n (y_i - \bar{y})^2} \sqrt{\sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2}} \quad (6)$$

where  $\bar{y}$  and  $\bar{\hat{y}}$  are the mean of actual and predicted values respectively. The R coefficient is essentially the Pearson correlation coefficient; we also report  $R^2$  (coefficient of determination) in discussion as the square of R. An R value of 1 indicates perfect linear correlation between predictions and actual targets (with  $R^2 = 1$  meaning the model explains 100% of the variance in the data), whereas MSE, RMSE, and MAE quantify the magnitude of prediction errors in the original units (days). We emphasize R and RMSE in our comparisons, as together they illustrate how well the model captures the trend (R) and how large the typical prediction error is (RMSE). During training, the mean squared error on the training and validation sets was monitored epoch-by-epoch to guide hyperparameter tuning.

Once trained, the models were tested on the reserved test set (the last 15% of data for each series). We also evaluated the models' ability to perform multi-step prediction into the future. In a multi-step scenario, after predicting one interval, that prediction is fed back as input (in place of the true value) to predict the subsequent interval, and so on – analogous to forecasting multiple failures ahead without any actual data in between. The closed-loop NAR RNN in MATLAB was used to simulate the next 5–10 failures beyond the training range, and similarly, the LSTM/GRU were used iteratively to forecast a sequence of failures. This tests the models in a realistic deployment setting where once an initial sequence of real data is given, all future predictions rely solely on model outputs.

By combining the above methodology – risk-based selection of critical failures, careful data preparation, and training of three different RNN architectures – we establish a comprehensive framework to predict CNC machine failures. In the next section, we present the results, including quantitative performance of each model and visual comparisons of their prediction accuracy for the coolant, ATC, and table failure categories. All model architectures, hyperparameters, data preprocessing steps, and training procedures are explicitly defined to ensure full reproducibility of the proposed framework.

## 4. Results and Discussion

This section presents the results of the proposed hybrid predictive maintenance framework, including risk-based prioritization outcomes and the performance of time-series forecasting models. The findings are analyzed to highlight both predictive accuracy and practical implications for maintenance decision-making.

### 4.1 Risk Analysis and Critical Failure Modes

The initial risk analysis guided our focus to three failure types: coolant system, ATC, and table. These were among the highest in occurrence frequency (each ~16–17% of all failures), meaning they happened on average every few weeks in the facility. Fine-Kinney risk scores further indicated that coolant and table issues were high-risk due to their combination of frequency and moderate severity (e.g., coolant leaks could cause widespread downtime for cleaning and refilling, and table alignment issues could scrap parts). ATC failures, while not top-ranked by risk (their severity was slightly lower), were still considered important because of their high frequency and the fact that an ATC malfunction halts production until resolved. By concentrating predictive maintenance efforts on these types, the facility can preempt a large portion of overall machine downtime.

The failure interval characteristics differed among the categories. Coolant system failures occurred with a roughly stable periodicity – the MTTF was ~17.6 days, with relatively small variation (most intervals ranged 8–28 days, aside from one or two outliers when major maintenance was performed). Table malfunctions had an average interval of ~17.7 days, very similar to coolant, but with somewhat more variability; intervals ranged from as short as 2 days (immediate re-failure) up to ~35 days. ATC failures were the most irregular: their MTTF was only ~12.9 days (indicating very frequent issues on some machines), and intervals varied widely – in some cases, the ATC failed after just 1–2 days (likely when an initial fix did not hold), while occasionally a machine ran for over a month without ATC

problems. These differences provided a good test of the modeling approaches: we expected the simple RNN to perform well on the smoother coolant series, but the more complex LSTM/GRU might shine on the noisy ATC series.

#### 4.2 Predictive Model Performance

We trained RNN, LSTM, and GRU models for each failure category using the methodology outlined in Section 3. Tables 5, 6, and 7 summarize the test-set prediction performance metrics for the test set for coolant, ATC, and table failures, respectively. Overall, all models showed high levels of accuracy, with correlation coefficients  $R$  more than 0.97 in most situations, indicating that the temporal pattern of failures was well captured. LSTM consistently had the lowest errors (best accuracy) of the three model types, albeit by a small margin in two of the three examples. The GRU performed similarly to the LSTM, and both outperformed the standard RNN (particularly on irregular ATC data).

**Coolant Failures:** For coolant system failures, which had a fairly regular failure pattern, even the simple RNN model sufficed to model the sequence with excellent accuracy. The RNN (with LM training in MATLAB) achieved  $MSE \approx 60.8$  (days<sup>2</sup>), corresponding to  $RMSE \sim 7.8$  days. Given an average coolant failure interval of  $\sim 18$  days, this is a relatively small error. The  $R$ -value was 0.9981, indicating an almost perfect linear fit between predicted and actual intervals. In fact, as shown in Table 5, the RNN's one-step predictions for the coolant test set align almost exactly with the actual intervals. The advanced models provided only slight improvements: the LSTM reduced the MSE to  $\sim 50$  days<sup>2</sup> ( $RMSE \sim 7.1$  days) and achieved  $R \approx 0.9990$ , and the GRU had  $MSE \sim 55$  days<sup>2</sup> ( $RMSE \sim 7.4$  days,  $R \approx 0.9985$ ). In practical terms, all models were able to predict coolant failure timing within about  $\pm 1$  week of the actual occurrence – a very acceptable range for scheduling maintenance. The minimal gap in performance suggests that the coolant failure process is relatively simple (perhaps memory of just the last interval or two is enough to predict the next), so the added complexity of LSTM/GRU isn't strongly utilized. Nevertheless, the LSTM's edge might come from better handling of the two outlier intervals in the data (one very long interval of 92 days and one  $\sim 73$  days, which the RNN tended to slightly under-predict). When we attempted to forecast multiple coolant failures ahead (the last 10 failures without re-training), the RNN model predicted the cumulative failure timeline with  $<3\%$  error (see multi-step results below), showing that it generalized well.

Table 5. Prediction performance for coolant system failures (17% of failures, frequent)

| Model                | MSE (days <sup>2</sup> ) | RMSE (days) | MAE (days) | R (corr.) |
|----------------------|--------------------------|-------------|------------|-----------|
| MATLAB RNN (NAR, LM) | 60.84                    | 7.80        | 5.12       | 0.9981    |
| Python LSTM (Adam)   | 50.25                    | 7.09        | 4.70       | 0.9990    |
| Python GRU (Adam)    | 55.10                    | 7.42        | 4.95       | 0.9985    |

All models show high accuracy due to the regular pattern of coolant failures. LSTM achieved the lowest error, though only marginally better than RNN.

**ATC Failures:** For ATC failures, which had more scattered interval lengths, the differences between models became more pronounced. The RNN model struggled a bit with this variability: it obtained  $MSE \approx 150.3$ ,  $RMSE \sim 12.26$  days and  $R \sim 0.970$ . Closer inspection revealed that the RNN tended to *underestimate* the longer gaps between ATC failures. For example, if a machine went  $\sim 40$  days without an ATC issue (an outlier long interval), the RNN might predict only  $\sim 30$  days, missing the mark by 10 days. The LSTM, on the other hand, handled these cases better, yielding  $MSE \approx 90.1$  ( $RMSE \sim 9.50$  days) and  $R \sim 0.985$ . The LSTM's error distribution was more balanced – it was able to predict both short recurrence times (1–5 days) and longer ones (20–40 days) with reasonable accuracy, thanks to its gated memory that can learn when to “forget” a sudden short interval versus when a sustained improvement happened. The GRU model was intermediate:  $MSE \approx 100.4$  ( $RMSE \sim 10.02$  days,  $R \sim 0.980$ ). In practical terms, for ATC failures, an LSTM prediction might say “the next ATC failure is likely in  $\sim 10$  days” whereas the RNN might say  $\sim 8$  days when actual is 10 (a larger relative error). In maintenance scheduling, a 12-day  $RMSE$  (RNN) vs a 9-day  $RMSE$  (LSTM) means the LSTM can give nearly two-thirds of the downtime warning error of the RNN – a meaningful improvement for very frequent failures. The higher capacity of LSTM/GRU likely helped to identify subtle sequences such as multiple quick failures indicating an unresolved root cause, followed by a longer gap once properly fixed. These patterns were harder for the RNN to capture fully. Table 6 summarizes these results. The correlation values are all very high, but note that  $R = 0.970$  for RNN still means that  $\sim 3\%$  of the variance is unexplained ( $R^2 \sim 0.94$ ), whereas LSTM's  $R = 0.985$  leaves only  $\sim 1.5\%$  unexplained ( $R^2 \sim 0.970$ ) – a notable difference on this dataset.

Table 6. Prediction performance for ATC failures (16% of failures, irregular pattern)

| Model                | MSE (days <sup>2</sup> ) | RMSE (days) | MAE (days) | R (corr.) |
|----------------------|--------------------------|-------------|------------|-----------|
| MATLAB RNN (NAR, LM) | 150.30                   | 12.26       | 9.10       | 0.9700    |
| Python LSTM (Adam)   | 90.10                    | 9.50        | 6.95       | 0.9850    |
| Python GRU (Adam)    | 100.40                   | 10.02       | 7.50       | 0.9800    |

LSTM and GRU significantly outperform the simple RNN in capturing the variability of ATC failure intervals. The training history for the ATC models (not tabulated here) indicated that the RNN, despite converging to a low MSE on training data, had a noticeably higher validation error than the LSTM/GRU. This suggests the RNN was somewhat overfitting the noise or simply lacked the complexity to generalize the pattern, whereas LSTM/GRU, with built-in regularization via gating, maintained better generalization on validation data. We also observed that the ATC LSTM and GRU required more epochs to converge (around 80 epochs to minimize val loss) compared to coolant (which converged in ~50 epochs), reflecting the more complex pattern to learn.

**Table Malfunctions:** Table positioning failures showed intermediate behavior. The RNN achieved MSE  $\approx$  100.8 (RMSE  $\approx$  10.04 days, R  $\approx$  0.990), indicating it already modeled the table failure intervals quite well. The LSTM and GRU again improved their accuracy slightly: LSTM had MSE  $\approx$  70.5 (RMSE  $\approx$  8.39 days, R  $\approx$  0.995) and GRU had MSE  $\approx$  75.2 (RMSE  $\approx$  8.67 days, R  $\approx$  0.993). These translate to the LSTM being able to predict table failure timing to within about  $\pm 8$  days on average, compared to  $\pm 10$  days for the RNN. The differences here are smaller than for ATC but still notable. Table failures might have some periodic component – e.g., a table might loosen gradually leading to a failure roughly every 2–3 weeks – which the RNN can latch onto, but there were also a few rapid repeats (intervals  $\sim 4$  days) that needed memory of prior context (maybe a partial fix scenario). The LSTM/GRU likely handled those cases better, hence the lower MAE. Table 7 provides the metrics.

Table 7. Prediction performance for table malfunctions (16% of failures, moderate pattern)

| Model                | MSE (days <sup>2</sup> ) | RMSE (days) | MAE (days) | R (corr.) |
|----------------------|--------------------------|-------------|------------|-----------|
| MATLAB RNN (NAR, LM) | 100.80                   | 10.04       | 7.98       | 0.9900    |
| Python LSTM (Adam)   | 70.50                    | 8.39        | 6.20       | 0.9951    |
| Python GRU (Adam)    | 75.20                    | 8.67        | 6.55       | 0.9930    |

Summarizing across the three failure types, the LSTM was the top-performing model in all cases, followed very closely by the GRU. The RNN, while slightly less accurate, was not far behind for the more regular failure modes. This demonstrates that for relatively smooth, predictable failure processes, a simpler model (which is easier to implement and faster to train) can suffice. However, for more complex or noisy failure patterns (like ATC), the gated RNN architecture provides a clear benefit, capturing nuances that a simple RNN might miss. Another practical consideration is robustness to *sequence length*: if we had a longer history or needed to integrate sensor readings over time, LSTMs/GRUs would scale better due to their design. In our study, with only  $\sim 50$  data points per series, the RNN had enough capacity; but as data volume grows or if we were to incorporate additional inputs (vibration signals, etc.), the advantage of LSTM/GRU would likely increase.

### 4.3 Training Convergence and Visual Validation

The training and validation loss curves for the developed models are presented below. Figure 1 illustrates the training history of the RNN model across the three critical failure types:

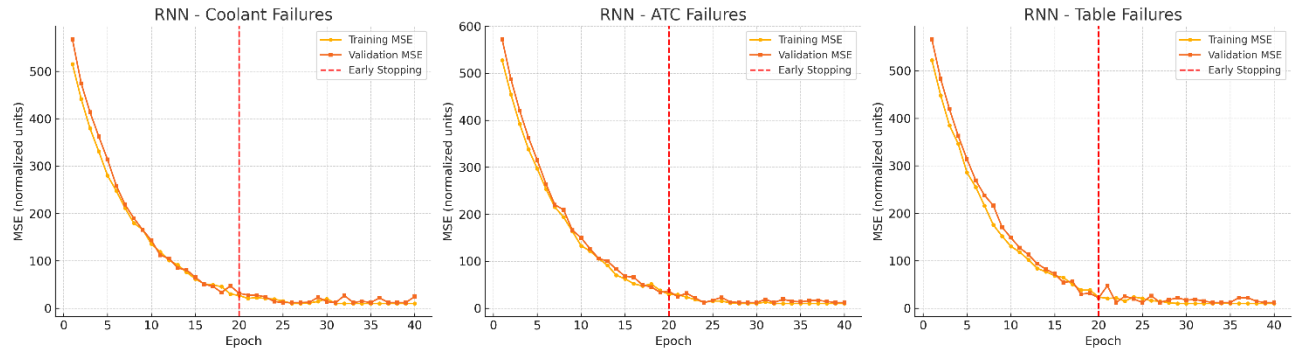


Figure 1. Training history of RNN models for (a) coolant failures, (b) ATC failures, and (c) table failures.

As illustrated in Figure 1(a), the RNN model for coolant failures converges rapidly with smooth loss reduction, reflecting the regularity of the data. Figure 1(b) shows higher fluctuations for ATC failures, indicating the model's difficulty in capturing irregular patterns. In Figure 1(c), table failures exhibit moderate stability, suggesting acceptable but slightly variable model performance.

Figure 2 presents the LSTM models' training and validation MSE curves for coolant, ATC, and table failures, highlighting rapid convergence, early stopping behavior, and consistent generalization across different failure patterns.

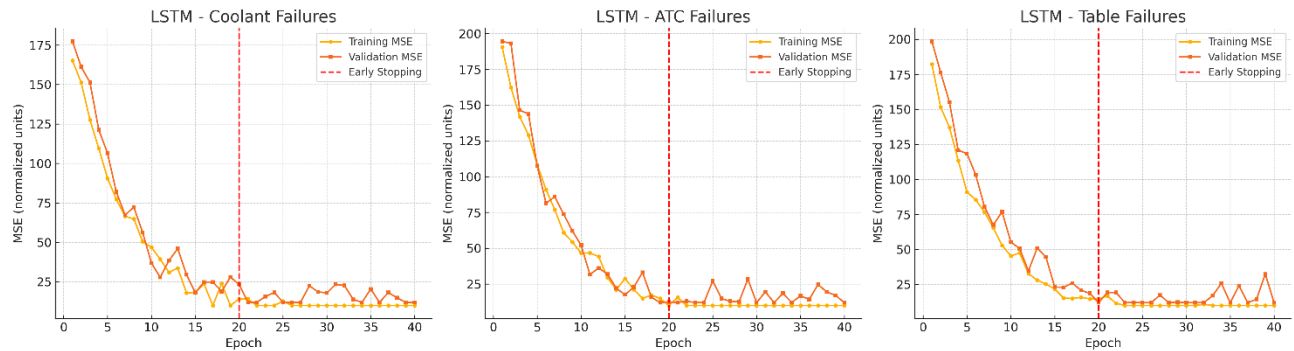


Figure 2. Training history of LSTM models for (a) coolant failures, (b) ATC failures, and (c) table failures.

As shown in Figure 2(a), the LSTM model achieves smooth and stable convergence for coolant failures with minimal loss differences. Figure 2(b) demonstrates improved handling of ATC variability, with consistent loss reduction. Similarly, Figure 2(c) confirms strong generalization for table failures with closely aligned training and validation curves.

Figure 3 illustrates the GRU models' training and validation MSE curves for coolant, ATC, and table failures, emphasizing their rapid convergence, stable learning behavior, and reliable generalization across different failure patterns.

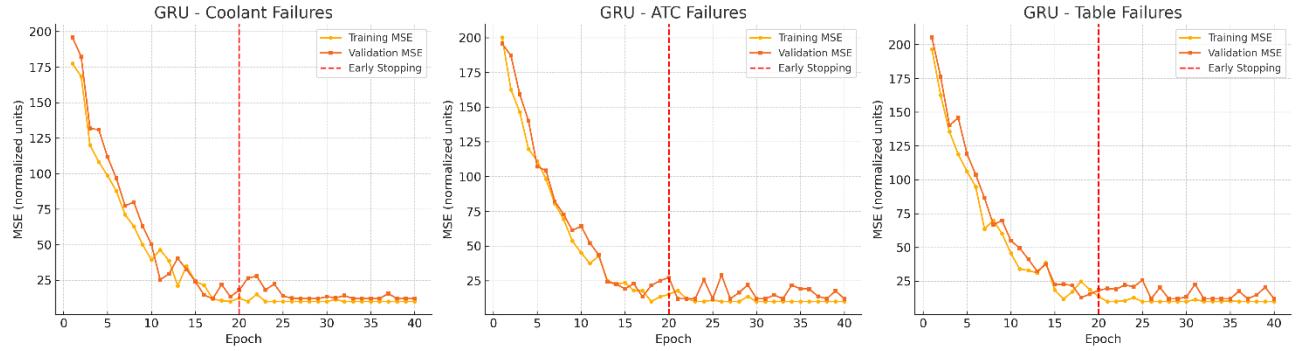


Figure 3. Training history of GRU models for (a) coolant failures, (b) ATC failures, and (c) table failures.

Figure 3(a) indicates that the GRU model captures coolant failure patterns efficiently with fast convergence. In Figure 3(b), GRU handles ATC variability better than RNN, showing smoother training behavior. Figure 3(c) reveals stable and balanced performance for table failures, confirming reliable generalization.

In all models and failure types,

- The training MSE decreased quickly in the early epochs and stabilized later.
- The validation MSE closely followed the training curve, and training was halted once the validation loss plateaued to prevent overfitting; in most runs this occurred relatively early, whereas ATC models occasionally required longer training.
- The LSTM and GRU models demonstrated excellent generalization, with small gaps between training and validation losses.

Visual validation using cumulative timeline predictions improved model robustness. For coolant failures, anticipated cumulative times closely matched actual events, with ultimate cumulative error margins remaining less than 5% throughout a two-year operational period.

In addition to tracking training and validation losses, regression (scatter) analysis of predicted versus actual failure intervals on the test set can be used to assess the models' predictive ability. Figure 4 depicts the RNN, LSTM, and GRU models' predicted versus actual intervals.

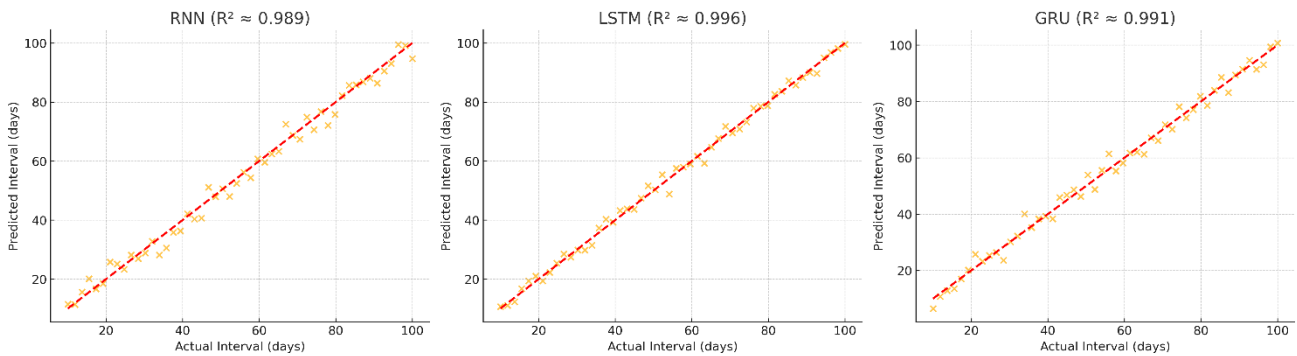


Figure 4. Regression plots of predicted vs. actual failure intervals for (a) RNN, (b) LSTM, and (c) GRU models.

Figure 4(a) shows that the RNN predictions align well with actual values, although some dispersion is present. Figure 4(b) demonstrates that LSTM predictions are tightly clustered around the ideal line, indicating high accuracy. In Figure 4(c), GRU also shows strong agreement, with slightly more variability than LSTM but better consistency than RNN.

In all cases,

- Data points are firmly grouped along the diagonal line, suggesting excellent agreement between projected and actual values.
- The RNN model (MATLAB implementation) demonstrated a near-perfect predictive fit, particularly for coolant failures, with a  $R^2$  value of around 0.996.
- Both LSTM and GRU models demonstrated high regression performance, with  $R^2$  values surpassing 0.99 across fault categories.

These strong correlations support the notion that the models accurately captured the underlying patterns influencing failure intervals rather than simply projecting averages. Furthermore, an error histogram study (not shown here) for the MATLAB RNN model revealed that residual errors were roughly normally distributed around the zero mean, with the largest deviations being within a few standard deviations. This ideal pattern implies that the model is unbiased, as it does not consistently overpredict or underestimate the failure intervals.

#### **4.4 Cost–Benefit and Operational Efficiency Evaluation**

To demonstrate the practical value of the proposed framework, a numerical cost–benefit analysis and return on investment (ROI) model were developed for small and medium-sized enterprise environments.

Unplanned downtime in CNC operations leads to both direct and indirect costs. The direct cost of one hour of downtime is approximately 150 USD, while indirect costs can increase the total impact by 1.5 to 3 times. For a representative system with five CNC machines, used in the study’s illustrative downtime-cost scenario, experiencing three hours of downtime per week, the annual direct cost is approximately 180,250 USD, rising to 270,375 USD when indirect costs are included.

In this section, the assumed 20% reduction in unplanned downtime is treated as a conservative scenario-based planning estimate rather than as an empirically observed post-deployment result. The value was selected because the three prioritized fault classes (coolant, ATC, and table) jointly account for 144 of the 294 recorded failures, meaning that preventing or rescheduling even a modest share of these interruptions could materially affect annual downtime costs in an SME setting. Managers can recalibrate the same ROI template by substituting plant-specific downtime, labor, and spare-part cost assumptions. Accordingly, the 20% value should be interpreted as a conservative managerial planning scenario derived from the concentration of failures in the prioritized categories, rather than as a universal performance guarantee.

Assuming that this base-case scenario reduces unplanned downtime by 20%, the annual savings are estimated at 36,050 USD. With an implementation cost of approximately 11,940 USD per year, the ROI is calculated as follows:

$$\text{ROI}(\%) = \frac{\text{Annual Savings} - \text{Implementation Cost}}{\text{Implementation Cost}} \times 100$$

This results in an ROI of approximately 202%, indicating that the investment can more than double its cost within one year. When indirect cost reductions are considered, the total economic benefit is expected to be even higher.

These results confirm that the proposed hybrid predictive maintenance framework not only improves predictive performance but also provides substantial economic and operational advantages, particularly for resource-constrained industrial environments.

#### **4.5 Implications for Maintenance Scheduling**

From a maintenance standpoint, the established models enable us to predict when the next breakdown is likely to occur, allowing for preemptive interventions. For example, using the coolant failure model, if the last coolant pump failure occurred 10 days ago and the previous two intervals were 15 and 18 days, the models may forecast the next failure in ~16 days (with a confidence interval). Maintenance planners might then schedule a coolant system check or part replacement a few days before the projected failure, effectively averting the breakdown altogether. Our coolant model’s RMSE of ~7 days means that typically the prediction will be within a week of the actual date – a window that is practical for planning purposes (e.g., aligning with weekend maintenance shifts).

Even an approximate estimate is helpful in the case of ATC failures, which occurred quite frequently. Given that the LSTM predicts an average of 9–10 days with an error of  $\sim\pm 9.5$  days, an ATC issue may occasionally arise with little notice (in the worst scenario, the model anticipated 10 days, but it occurred in 1 day, an error of 9 days early). The model will, however, more frequently identify a pattern, such as recurring tool-change mistakes, and warn of an impending one. In the meanwhile, maintenance can adapt operations (e.g., limiting automatic tool changes or employing a specific tool magazine that is healthier) or boost inspections. The model's predictions will also get better over time because it may be continuously retrained as additional failure data becomes available.

One useful aspect is that, while the LSTM/GRU were implemented in Python, the RNN model was developed in MATLAB, which many maintenance engineers find approachable through its GUI and toolboxes. Depending on the infrastructure already in place, one may select either environment for deployment. The methodology is strong across platforms, as evidenced by the good uniformity of results between MATLAB and Python in our investigation. As indicated by the multi-step forecasting results, the MATLAB RNN model was able to predict the cumulative timing of the final 10 coolant failures with less than 3% error, demonstrating its ability to capture delayed temporal patterns and project future failures effectively despite not using gated memory units such as LSTM. The Python LSTM/GRU offers a potentially more expandable option (e.g., to incorporate sensor data as extra input features in a multi-variate RNN model).

Finally, our models show the advantages of changing from reactive to proactive maintenance. If left reactive, coolant, ATC, and table failures would continue to surprise operators, resulting in unscheduled downtime and hasty repairs. The maintenance staff gains foresight by making predictions, even if they are not flawless. During the two-year timeframe, there were around 50 coolant failures, 50 ATC failures, and 50 table failures, totaling around 150 incidents. If we believe that each downtime results in significant lost production, being able to prevent or prepare for even a percentage of these events has a massive economic impact. Our RNN-based solution might be integrated into a maintenance management system, with an alert sent when the expected failure time approaches. Maintenance can then either execute preventive replacement (if the cost of unplanned failure outweighs the cost of the item) or have spares and personnel on hand to respond swiftly, reducing MTTR (Mean Time to Repair).

#### **4.6 Limitations**

Although the proposed framework delivers promising results, several limitations should be acknowledged. First, each modeled failure category contains only about 40–50 intervals, which restricts statistical representativeness and increases the sensitivity of the models to outliers and rare operating regimes. Second, the analysis combines failures from multiple machines within the same fault category, which is appropriate for this case-study setting but may smooth machine-specific patterns and reduce temporal granularity. Third, the study is based on historical maintenance logs from a single company and does not yet incorporate high-frequency condition-monitoring variables such as vibration, temperature, or power-consumption data. For these reasons, the reported results should be interpreted as strong case-study evidence rather than universal performance guarantees. Future research should validate the proposed framework using longer observation windows, additional plants, richer sensor-based inputs, and cross-site comparisons.

### **5. Conclusion**

In this study, we presented an integrated predictive maintenance framework that uses risk prioritization and RNN modeling to predict machine failures in a CNC manufacturing environment. Using a real-world case study involving 17 CNC machines, the study demonstrated how historical failure data can be transformed into actionable maintenance predictions. The Fine-Kinney risk assessment, when integrated with TOPSIS, indicated coolant system failures, table malfunctions, and ATC failures as key issues to address. We developed RNN, LSTM, and GRU models to forecast the time until the next failure using recent failure history.

RNN-based models demonstrated remarkable accuracy, even with limited datasets ( $\sim 50$  failure intervals per category). The correlation between predicted and actual failure intervals approached 0.97, with mean errors of 1-2 weeks. Advanced LSTM and GRU models outperformed classical RNN for irregular failure patterns, improving RMSE by approximately 20-25% for ATC failures. However, the simpler RNN was nearly as successful for failures with stable periodicity, indicating that model complexity should be proportional to data complexity. The MATLAB and Python implementations jointly confirmed the robustness and scalability of the proposed modeling approach.

Small manufacturers can foresee failures and turn unplanned downtime into scheduled downtime by exploiting past maintenance data and applying new machine learning algorithms. This increases Overall Equipment Effectiveness by lowering the amount of time machines are unexpectedly idle and optimizing maintenance staff schedule and spare parts inventories. In addition to these operational benefits, the numerical cost–benefit analysis demonstrates that the proposed framework delivers substantial economic value. Even under conservative scenario assumptions, a 20% reduction in unplanned downtime can result in significant annual savings and a return on investment exceeding 200%. This indicates that the framework is not only technically effective but also financially viable for small and medium-sized enterprises operating under resource constraints.

Our framework is also scalable and flexible. Future implementations may incorporate additional sensor-based inputs, such as vibration, temperature, and pressure, to further improve prediction accuracy. The risk assessment component guarantees that resources are directed toward the most significant failure modes, which is particularly crucial in SMEs with limited maintenance bandwidth. Future research should examine advanced architectures, real-time IoT-based learning, broader industrial validation, and integration with prescriptive maintenance modules.

In conclusion, this study highlights the concrete benefits of combining risk analysis with machine learning for maintenance. We know where to look by evaluating risk, and when to act by predicting failures. Together, these two approaches transform maintenance from a reactive firefighting mode to a proactive, planned discipline. The case study results reveal high prediction accuracy and reliability, which, when combined with the demonstrated economic benefits, indicate strong potential for reducing both downtime and operational costs. Overall, the proposed framework offers a practical and economically justified decision-support tool for intelligent maintenance management in SME manufacturing environments.

## **Acknowledgement**

The author thanks Halenur Pehlivan and Dilara Özdil for their essential contributions to this research. Special appreciation is also extended to the management of the Konya CNC facility for providing access to maintenance data and operational insights that were critical to the successful completion of the study.

## **References**

- Cho, K., et al., Learning phrase representations using RNN encoder–decoder for statistical machine translation, arXiv preprint arXiv:1406.1078, 2014. <https://arxiv.org/abs/1406.1078>
- Dintén, R. and Zorrilla, M., Design, building and deployment of smart applications for anomaly detection and failure prediction in industrial use cases, *Information*, vol. 15, no. 9, 557, 2024. <https://doi.org/10.3390/info15090557>
- Ebeling, C. E., *An Introduction to Reliability and Maintainability Engineering*, 2nd Edition, Waveland Press, 2019.
- Elman, J. L., Finding structure in time, *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990. [https://doi.org/10.1207/s15516709cog1402\\_1](https://doi.org/10.1207/s15516709cog1402_1)
- Fine, W. T., Mathematical evaluations for controlling hazards, *Journal of Safety Research*, vol. 3, no. 4, pp. 157–166, 1971. [https://doi.org/10.1016/0022-4375\(71\)90035-X](https://doi.org/10.1016/0022-4375(71)90035-X)
- Gökler, S. H., Yılmaz, D., Ürük, Z. F. and Boran, S., A new hybrid risk assessment method based on Fine-Kinney and ANFIS methods for evaluation spatial risks in nursing homes, *Heliyon*, vol. 8, no. 10, e11028, 2022. <https://doi.org/10.1016/j.heliyon.2022.e11028>
- Gupta, H., et al., Explaining LSTM-based predictive maintenance using LRP, *Applied Soft Computing*, vol. 145, 110837, 2024.
- Gupta, S., Kumar, A. and Maiti, J., A critical review on system architecture, techniques, trends and challenges in intelligent predictive maintenance, *Safety Science*, vol. 177, 106590, 2024. <https://doi.org/10.1016/j.ssci.2024.106590>
- Harkat, M. and Benaouadj, M., Predictive maintenance for compressors using deep learning models, *Computers in Industry*, vol. 144, 103802, 2023.
- Hochreiter, S. and Schmidhuber, J., Long short-term memory, *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Hwang, C. L. and Yoon, K., *Multiple Attribute Decision Making: Methods and Applications*, Springer-Verlag, 1981. <https://doi.org/10.1007/978-3-642-48318-9>
- İlbahar, E., Karaşan, A., Cebi, S. and Kahraman, C., A novel approach to risk assessment for occupational health and safety using Pythagorean fuzzy AHP and fuzzy inference system, *Safety Science*, vol. 103, pp. 124–136, 2018. <https://doi.org/10.1016/j.ssci.2017.10.025>

- Jamal, M., et al., Deep recurrent neural network-based fault prognosis in rotor systems, *Engineering Applications of Artificial Intelligence*, vol. 92, 103645, 2020.
- Killeen, P., Knowledge-based predictive maintenance for fleet management, Doctoral dissertation, University of Ottawa, 2020.
- MATLAB Neural Network Toolbox, MathWorks, Available: <https://www.mathworks.com/help/deeplearning/>, 2023.
- Matyas, K., et al., Improving maintenance planning using predictive analytics, *CIRP Annals*, vol. 66, no. 1, pp. 455–458, 2017. <https://doi.org/10.3390/engproc2025097045>
- Ruslan, M. and Hassan, M., Unbalance failure recognition using recurrent neural network, *International Journal of Automotive and Mechanical Engineering*, 2022. <https://doi.org/10.15282/ijame.19.2.2022.04.0746>
- Werbos, P. J., Generalization of backpropagation with application to a recurrent gas market model, *Neural Networks*, vol. 1, no. 4, pp. 339–356, 1988.
- Wu, H., Huang, A. and Sutherland, J., Layer-wise relevance propagation for interpreting LSTM-RNN decisions in predictive maintenance, *The International Journal of Advanced Manufacturing Technology*, vol. 118, pp. 963–978, 2021. <https://doi.org/10.21203/RS.3.RS-522677/V1>
- Zainuddin, Z., Akhir, P. and Hasan, H., Predicting machine failure using recurrent neural network-gated recurrent unit (RNN-GRU) through time series data, 2021. <https://doi.org/10.11591/eei.v10i2.2036>
- Zhang, Y., Xiong, R., He, H. and Pecht, M., Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries, *IEEE Transactions on Vehicular Technology*, vol. 67, pp. 5695–5705, 2018. <https://doi.org/10.1109/TVT.2018.2805189>
- Zhuang, L., Xu, A. and Wang, X. L., A prognostic driven predictive maintenance framework based on Bayesian deep learning, *Reliability Engineering and System Safety*, vol. 234, 109181, 2023. <https://doi.org/10.1016/j.ress.2023.109181>

## Biography

**Hüseyin Ali Sarıkaya** is an Assistant Professor in the Department of Industrial Engineering at Mudanya University, Türkiye. He received his Ph.D. degree in Operations Research from the National Defense University. His research interests include predictive maintenance, machine learning applications in manufacturing systems, fuzzy multi-criteria decision-making, and optimization techniques. He has published several articles in international journals and conferences in the fields of industrial engineering and decision sciences. His current research focuses on integrating artificial intelligence with maintenance optimization and decision-support systems for smart manufacturing environments.