Journal of Information Systems Engineering and Management
2025, 10(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Adaptive Reinforcement Learning Framework for Real-Time Tool Wear Optimization and Predictive Maintenance

*[1]N V. Krishnamoorthy, [2]Dr.Vijay, [3]Dr.Masepogu Wilson Kumar

*[1]*Associate Professor, Department of Mechanical Engineering, Sri Krishna College of Engineering and Technology, Coimbatore. Orcid id: 0000-0001-8870-6064. *[1]Email: nvkrishnamoorthy1968@gmail.com.*

[2]*Professor & Director (Research & Consultancy), Karunya Institute of Technology and Sciences Coimbatore. Orcid id: 0000-0003-3682-7192. Email: vijayjoseph@karunya.edu*

[3]*Assistant professor (SG) Department of Mechanical Engineering, Karunya Institute of Technology and Science. Orchid id: 0000-0001-6927-6951. Email: wilson@karunya.edu*

ARTICLE INFO

Received: 29 Sept 2024

Revised: 30 Nov 2024

Accepted: 12 Dec 2024

ABSTRACT

Tool wear monitoring and predictive maintenance are critical in manufacturing, where traditional methods often struggle to adapt to changing conditions. This research presents an Adaptive Reinforcement Learning Framework for Real-Time Tool Wear Optimization and Predictive Maintenance (ARTOM). The framework integrates reinforcement learning with real-time sensor feedback to optimize machining parameters and maintenance schedules dynamically. Proximal Policy Optimization (PPO) is used to guide decision-making by balancing tool life, product quality, and operational costs. Multi-agent reinforcement learning divides tasks among agents to handle diverse machining scenarios, while sliding window techniques and dimensionality reduction ensure efficient data processing. The study has used the benchmark dataset, which include time-series sensor data and machining parameters. Metrics potential metrics have been used to evaluate prediction accuracy, while runtime and memory usage assess computational efficiency. Results has shown that ARTOM consistently achieves lower prediction errors and faster execution times than contemporary baseline models. These findings demonstrate ARTOM's ability to adapt to different tool conditions and improve operational decision-making.

Keywords: Tool Wear, Predictive Maintenance, Reinforcement Learning, Proximal Policy Optimization, Multi-Agent Learning, Sensor Data, Time-Series Analysis, Machining Optimization

INTRODUCTION

Efficient tool wear management and predictive maintenance are essential for maintaining manufacturing productivity and quality. Tool wear impacts production by increasing costs through frequent replacements and unplanned downtime. Traditional methods, relying on fixed schedules or reactive maintenance, lack adaptability and fail to address varying conditions in real-time operations. The integration of machine learning and reinforcement learning (RL) techniques offers a way to optimize these processes by enabling systems to learn from data and adjust dynamically. Recent advancements in predictive modeling for tool wear monitoring have demonstrated significant improvements. For example, Long Short-Term Memory (LSTM) networks, enhanced with adaptive feature and temporal attention mechanisms, have achieved higher prediction accuracy by capturing global sequential dependencies in machining data [1]. Similarly, residual connection-based temporal networks handle randomness in cutting signals, offering a more consistent approach to monitoring wear progression [2]. Other studies have explored data-driven approaches, such as deep learning and transfer learning models, which improve prediction accuracy across various datasets and operational scenarios [3]. However, these models often face limitations in scalability and real-time adaptability, especially in complex industrial settings. Reinforcement learning has emerged as a complementary approach to predictive maintenance. Transformer-driven RL frameworks have been developed to estimate the Remaining Useful Life (RUL) of tools and recommend maintenance actions based on real-time data [4]. Additionally, RL methods have been applied to dynamically optimize cutting parameters, accounting for factors like tool wear and machine aging. These methods improve operational efficiency by balancing energy consumption and production time [5]. Despite these advances, challenges remain in integrating predictive models with RL for real-time optimization, as well as ensuring generalizability across different manufacturing environments.

This study focuses on bridging these gaps by proposing an adaptive reinforcement learning framework that integrates tool wear optimization with predictive maintenance. While existing research often separates predictive maintenance and tool wear optimization, this framework combines both into a unified system. Real-time data processing enables dynamic adjustments to machining parameters and maintenance schedules, addressing

limitations in previous approaches. The objectives of this research are threefold: to design an RL-based system for optimizing machining parameters, to incorporate predictive maintenance strategies into real-time decision-making, and to validate the framework across diverse machining environments. These objectives aim to reduce tool wear, improve operational decision-making, and minimize downtime in manufacturing processes. This study contributes to the field by providing a scalable, adaptive framework that leverages real-time data and reinforcement learning to address operational challenges in manufacturing. The findings are expected to enhance understanding of how RL can be applied in industrial settings to balance costs, quality, and tool life while addressing implementation challenges such as scalability and data requirements [6], [7], [8].

The paper is organized as follows. The next section reviews related work in tool wear prediction and predictive maintenance, identifying current trends and research gaps. Section 3 describes the proposed framework, including its architecture and methods. Section 4 presents experimental results and validation. Section 5 discusses the findings and implications, while Section 6 concludes with a summary and suggestions for future work.

RELATED WORK

This section examines diverse approaches in tool wear prediction and predictive maintenance, focusing on practical methods and their applications. The studies cover topics such as data-driven algorithms, reinforcement learning, generative models for data augmentation, and hybrid frameworks for monitoring and optimization. These methods aim to enhance machining efficiency, improve prediction accuracy, and support maintenance strategies tailored to real-world needs. By categorizing research based on shared concepts, the review highlights advancements in handling data challenges, integrating real-time feedback, and optimizing manufacturing processes. This work provides a structured understanding of ongoing efforts to address tool wear management and predictive maintenance challenges.

Advanced algorithms are commonly used to improve tool wear prediction. Lin et al. [9] combined evolving connectionist systems with hidden semi-Markov models to capture nonlinear wear patterns and temporal dynamics. The approach works across diverse datasets but depends on substantial data inputs. Xie et al. [10] introduced the IE-Bi-LSTM model, which integrates bidirectional LSTMs with information entropy to weight features dynamically. This model handles sequential data but requires significant computational power. Wei et al. [11], Bonci et al., [12], Oliveira et al., [5] and Li et al., [13] proposed an improved salp swarm algorithm (SSA2) for wear prediction, enhancing convergence speed and accuracy, though it requires expertise for parameter tuning. Lin et al. [14] used a hybrid fuzzy neural network to manage uncertain machining data, improving prediction accuracy but requiring specialized knowledge for implementation.

Generative models tackle the challenge of limited datasets by producing synthetic data for tool wear prediction. He et al. [15] introduced CDAGAN, which uses GANs to generate synthetic samples for imbalanced datasets. This method improves prediction but depends on validating the quality of synthetic data. Parisi et al. [16] applied GAN-based methods to create synthetic features for training, which helps improve reliability but requires careful evaluation to prevent data biases. Reinforcement learning supports dynamic adjustments for tool wear and maintenance. Zhao et al. [4] proposed TranDRL, combining Transformers with reinforcement learning to predict the Remaining Useful Life (RUL) of tools. This method provides actionable maintenance schedules but demands high computational resources. Ge et al. [17] developed a framework using deep reinforcement learning to optimize maintenance actions based on real-time sensor data. The framework performs well with continuous data availability but may face challenges in data-scarce environments.

Hybrid systems offer new methods for managing tool wear and predictive maintenance. Abbas et al. [18] proposed a hierarchical framework combining interpretable deep learning models with Bayesian optimization, enhancing prediction accuracy while adding transparency to decision-making. This framework, however, has increased computational costs. Shah et al. [19] combined spectrogram analysis with machine learning for wear monitoring, improving prediction performance but requiring specialized equipment for spectrogram generation. Lin et al. [14] and Wu et al., [20] blended fuzzy logic and neural networks, creating a system to manage data uncertainty, though requiring domain-specific expertise.

Smart manufacturing systems combine tool wear prediction with real-time applications. Olalere et al. [21] developed a cyber-physical system to optimize tool wear and workpiece conditions using sensor feedback. While beneficial for advanced manufacturing, it requires sophisticated infrastructure. Hao et al. [22] introduced a self-learning digital twin framework, continuously updating predictions with real-time machining data. This system adapts to changing conditions but depends heavily on data availability and computational capacity. Qin et al. [23] proposed a lightweight model for real-time RUL prediction, offering fast and accurate results for edge-device applications. However, its simplicity may reduce accuracy in complex scenarios.

The reviewed articles show significant progress in tool wear prediction and maintenance strategies using algorithms, generative models, reinforcement learning, and hybrid systems. Common challenges include reliance on high-quality datasets, computational demands, and specialized equipment or expertise. Future research could focus on developing scalable methods, reducing resource requirements, and ensuring usability in diverse industrial settings. These studies highlight the need for balanced frameworks that integrate precision, adaptability, and ease of deployment for practical manufacturing environments.

METHODS AND MATERIALS

This section explains the framework designed for real-time tool wear optimization and predictive maintenance. It covers how sensor data, including vibration, acoustic emission, and temperature, are combined with machining parameters like cutting speed, feed rate, and depth of cut to monitor tool conditions. The section describes the use of reinforcement learning to adjust machining operations dynamically, focusing on components such as state representation, action space, and reward function. Techniques like Proximal Policy Optimization and multi-agent learning are detailed, alongside methods for integrating real-time feedback into decision-making. This structured approach highlights the steps taken to implement and evaluate the framework.

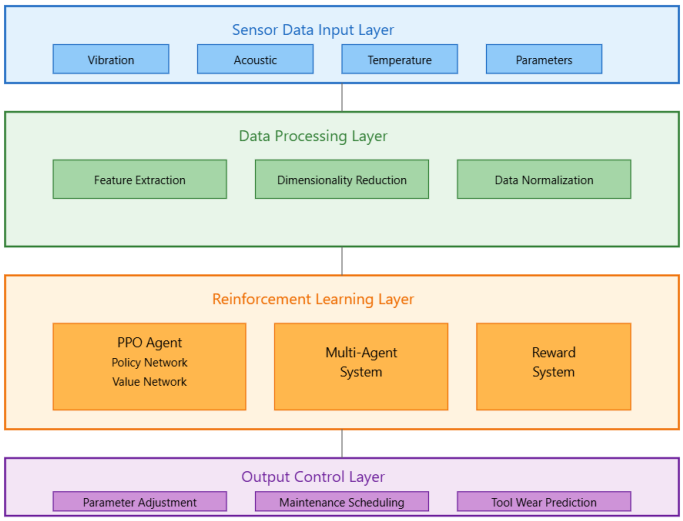

Figure 1: The System Architecture Diagram of the ARTOM

The ARTOM architecture shown in figure 1 takes raw data from machine sensors and turns it into useful actions. At its core, sensors track vibration, sound, heat, and cutting details. This data moves to processing units that clean and shrink it to manageable sizes. The processed information feeds into learning agents that use PPO and multi-agent methods to make decisions. These agents learn which actions work best by getting rewards when they make good choices about tool use and maintenance. The system then outputs three things: changes to cutting settings, when to do maintenance, and warnings about tool wear. Each part connects directly to the next, making a clear path from sensing to action. The setup keeps each job separate but linked, much like a well-organized factory floor. This direct approach helps machines run better and tools last longer. The whole system fits together like puzzle pieces, with each piece doing one clear job that helps the others work better.

A. Network Design

The proposed network design processes input data from sensors and machining parameters to monitor tool wear and optimize machining operations. The architecture combines layers for feature extraction, prioritization, and output decision-making to deliver precise control.

Input Layer: The network accepts real-time data from sensors, including vibration signals $v(t)$, acoustic emissions $a(t)$, and temperature $T(t)$. These are combined with machining parameters such as cutting speed (v), feed rate (f), and depth of cut (d). Time-series sensor data is represented as $X_s = [v(t), a(t), T(t)]$ and machining parameters as $X_m = \{v, f, d\}$.

The input layer preprocesses these data sources to prepare them for deeper analysis. CNNs identify spatial patterns, such as signal fluctuations, by applying convolution operations: Eq 1

$$F_{i,j,k} = \sigma\left(\sum_{m,n} W_{m,n,k} \cdot X_{i+m,j+n} + b_k\right) \text{ ....(Eq 1)}$$

where $F_{i,j,k}$ represents extracted features, $W_{m,n,k}$ is a learnable kernel, b_k is a bias term, and σ is an activation function. RNNs process sequential data to capture trends over time using: Eq 2

$$h_t = tanh(W_h \cdot X_s(t) + U_h \cdot h_{t-1} + b_h) \text{ ....(Eq 2)}$$

where h_t is the hidden state at time t, and W_h, U_h, b_h are learnable parameters. This structure ensures that both spatial and temporal dependencies are analyzed.

Hidden Layers: Hidden layers extract high-level features and prioritize key signals related to tool wear. Fully connected layers map the processed inputs to a higher-dimensional feature space, H, using: Eq 3

$$H = \sigma(W_f \cdot Z + b_f) \text{ ....(Eq 3)}$$

where $Z = [X_s, X_m]$ represents combined sensor and machining parameter data, and W_f, b_f are learnable weights and biases. Attention mechanisms refine the focus of the network on important signals. A context vector C is computed as: Eq 4

$$C = \sum_t \alpha_t \cdot h_t, \quad \alpha_t = \frac{exp(e_t)}{\sum_{t'} exp(e_{t'})}, \quad e_t = v_a^T tanh(W_a \cdot h_t + b_a) \text{ ....(Eq 4)}$$

where α_t represents attention weights, e_t is the attention score, and v_a, W_a, b_a are trainable parameters. This mechanism emphasizes anomalies or sudden changes in the data.

Output Layer: The output layer generates continuous actions $A = [\Delta v, \Delta f, \Delta d]$, which adjust machining parameters dynamically. The outputs are calculated using: Eq 5

$$A = tanh(W_o \cdot H + b_o) \text{ ....(Eq 5)}$$

where W_o, b_o are learnable weights and biases. For scenarios involving categorical decisions, such as switching between operational modes, the network incorporates discrete actions through a softmax function: Eq 6

$$P(a_k) = \frac{exp(z_k)}{\sum_j exp(z_j)} \text{ ....(Eq 6)}$$

where $P(a_k)$ represents the probability of selecting action a_k, and z_k are logits derived from the hidden features.

The architecture combines CNNs and RNNs for preprocessing, fully connected layers for feature extraction, and attention mechanisms for signal prioritization. The output layer generates continuous and discrete actions, enabling precise parameter adjustments. This design addresses real-time tool wear monitoring and machining parameter optimization, maintaining a balance between computational efficiency and operational adaptability.

B. Learning Algorithm

The learning algorithm 1 is based on Proximal Policy Optimization (PPO), which uses a clipped surrogate objective to regulate updates and ensure steady policy improvement. The design also incorporates a critic network to estimate value functions, enabling more accurate decision-making by guiding the policy updates.

Algorithm 1: Proximal Policy Optimization for Tool Wear Optimization

1. Initialize policy network parameters θ, value network parameters ϕ
2. Set clipping parameter ϵ, learning rates η_θ, η_ϕ, and entropy coefficient c_2
3. Collect trajectories $\{(s_t, a_t, r_t, s_{t+1})\}_{t=1}^{T}$ by running policy $\pi_\theta(a_t|s_t)$
4. Compute rewards-to-go $R_t = \sum_{k=t}^{T} \gamma^{k-t} r_k$
5. Compute advantage estimates $A_t = R_t - V_\phi(s_t)$
6. for each policy update iteration do
7. Compute probability ratio $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$
8. Compute surrogate objective:
$$L^{CLIP}(\theta) = \mathbb{E}_t[\min(r_t(\theta)A_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)]$$
9. Compute entropy term $H(\pi_\theta) = -\sum_a \pi_\theta(a|s) log \pi_\theta(a|s)$
10. Update policy parameters:
$$\theta \leftarrow \theta + \eta_\theta \nabla_\theta (L^{CLIP}(\theta) - c_2 H(\pi_\theta))$$
11. end for
12. for each value network update iteration do
13. Compute value loss:
$$L^{VALUE}(\phi) = \mathbb{E}_t\left[\left(V_\phi(s_t) - R_t\right)^2\right]$$
14. Update value network parameters:
$$\phi \leftarrow \phi - \eta_\phi \nabla_\phi L^{VALUE}(\phi)$$
15. end for

This combined loss ensures that the policy learns gradually while maintaining the ability to explore new strategies and refining predictions. The clipped objective prevents excessive deviations, and the critic network enhances state-value accuracy, which is critical for reliable policy updates.

C. Real-Time Monitoring and Decision-Making

Real-time monitoring relies on continuous feedback from sensors and operational data. The system updates the state representation dynamically to reflect the current condition of the tool and machining parameters. This approach enables timely adjustments and maintains machining efficiency.

Dynamic State Updates: Sensor data includes vibration (v(t)), acoustic emissions (a(t)), and temperature (T(t)), which represent the tool's current condition. Operational parameters, such as cutting speed (v), feed rate (f), and depth of cut (d), are incorporated into the state vector: Eq 7

$$S_t = [v(t), a(t), T(t), v, f, d] \quad \text{....(Eq 7)}$$

The state is recalculated at each time step t to ensure it represents the latest tool wear conditions and machining settings.

Sliding Window for Data Streams: To handle high-frequency sensor signals, a sliding window approach is applied. A fixed-size window collects recent data over a defined period, represented as: Eq 8

$$\boldsymbol{W}_t = [v(t-w), \dots, v(t)], [a(t-w), \dots, a(t)], [T(t-w), \dots, T(t)] \quad \text{....(Eq 8)}$$

Aggregated features, such as the mean and standard deviation, are extracted from the window to smooth out short-term fluctuations while retaining key trends: Eq 9

$$S_t = [mean(\boldsymbol{W}_t), std(\boldsymbol{W}_t), v, f, d] \quad \text{....(Eq 9)}$$

These features ensure stable state updates without losing responsiveness to sudden changes in tool wear or machining conditions.

Continuous Decision-Making: The updated state vector S_t is used to generate actions $A_t = [\Delta v, \Delta f, \Delta d]$. These actions adjust the cutting speed, feed rate, and depth of cut to optimize performance in real-time. Each action is recalculated at every time step to match evolving conditions.

By integrating dynamic updates and a sliding window for real-time data, the framework processes sensor feedback efficiently and adjusts machining parameters consistently. This design addresses tool wear progression and ensures optimal control without unnecessary delays.

D. Reward Function Design

The reward function translates operational objectives into measurable signals for guiding learning. It combines elements that represent tool life, production quality, and operational costs, balancing these factors to drive optimal decision-making.

- Components of the Reward Function

Tool Life Maximization: A penalty is applied based on the tool wear rate to discourage aggressive machining parameters that reduce tool life. The penalty is expressed as: Eq 10

$$R_{wear} = -\alpha \cdot wearrate \quad \text{....(Eq 10)}$$

where α adjusts the significance of wear rate in the overall reward. This ensures that decisions favor lower wear rates.

Production Quality Maintenance: A reward is provided when machining outputs fall within specified quality thresholds. This is modeled as: Eq 11

$$R_{quality} = \begin{cases} +R_{quality}, & if output satisfies quality standards, \\ 0, & otherwise. \end{cases} \quad \text{....(Eq 11)}$$

This term prioritizes maintaining precision and consistency in the machining process.

Cost Minimization: Energy consumption and tool replacement costs are penalized to reduce operational expenses. The energy penalty is defined as: Eq 12

$$R_{energy} = -\beta \cdot energyusage, \quad \text{....(Eq 12)}$$

where β reflects the sensitivity to energy costs. Similarly, tool replacement costs are penalized as: Eq 13

$$R_{replacement} = -\gamma \cdot replacementcost, \quad \text{....(Eq 13)}$$

with γ capturing the weight of downtime and expenses associated with tool changes.

- Combined Reward

The overall reward combines these components into a single function. The formulation balances tool life, quality, and cost as: Eq 14

$$R = \lambda_1 \cdot \left(\frac{1}{toolwearrate}\right) + \lambda_2 \cdot R_{quality} - \lambda_3 \cdot (energyusage + replacementcost) \quad \text{....(Eq 14)}$$

where:

• λ_1, λ_2, and λ_3 are adjustable weights that define the importance of each term.

For example, a higher λ_2 prioritizes quality in tasks requiring precision, while increasing λ_3 emphasizes cost savings in production-focused scenarios.

- Dynamic Adjustments

The reward function adapts to changing operational priorities by modifying weights dynamically. This ensures alignment with specific goals, such as prioritizing tool longevity during extended production runs or focusing on precision for critical components. Weight updates allow the reward structure to respond to situational needs, enabling flexible and goal-oriented optimization.

E. Multi-Agent Reinforcement Learning (MARL) Integration

The integration of multi-agent reinforcement learning (MARL) divides responsibilities among specialized agents, enabling the system to address diverse machining scenarios. Each agent contributes independently while coordinating with others to improve overall performance.

System Design: Each agent is tailored to manage a specific tool condition or material type, such as soft metals or hard alloys. This specialization ensures that agents focus on the unique challenges of their designated environments. The system uses a centralized critic and decentralized actor model. The centralized critic evaluates the combined performance of all agents, while decentralized actors independently adjust machining parameters within their respective environments. The relationship between the centralized critic and decentralized actors is expressed as: Eq 15

$$\pi_i(a_i|s_i) = actor_i(s_i), \quad V(s) = critic(s_1, s_2, \dots, s_n) \quad \text{....(Eq 15)}$$

where π_i represents the policy of agent i, a_i and s_i are the actions and state for agent i, and $V(s)$ is the global evaluation of the system.

Collaborative Learning: Agents periodically exchange information about their learned policies and key parameters. Communication channels facilitate the sharing of insights, such as wear patterns, machining strategies, and adjustments that worked in similar conditions. By pooling knowledge, agents refine their decision-making processes and align their actions with the collective goals. The periodic update cycle ensures consistency across agents without requiring constant synchronization.

- Coordination Mechanisms

Consensus-Based Learning Agents contribute to a global policy based on their local performance. Each agent's contribution is weighted to reflect its effectiveness and relevance in its environment. The global policy update is computed as: Eq 16

$$\pi_{global} = \sum_{i=1}^{n} w_i \cdot \pi_i \quad \text{....(Eq 16)}$$

where w_i represents the weight assigned to agent i based on its performance, and π_i is the policy it follows.

Hierarchical Reinforcement Learning A hierarchical structure separates high-level and low-level responsibilities. The high-level component oversees broad strategies, such as prioritizing goals based on material properties. Low-level agents execute these strategies by fine-tuning parameters like feed rate and cutting speed to meet the high-level objectives.

Scalability: The system is designed to expand by using shared embedding layers and parameterized policies. Shared embeddings capture common features across agents, reducing redundancy and improving learning efficiency. Parameterized policies adapt these shared features to address specific tool conditions or material types. This structure supports the addition of new agents or environments without a significant increase in computational requirements.

The MARL integration supports dynamic collaboration among agents, tailored learning for specific conditions, and scalability for complex operations. By distributing tasks and coordinating efforts, the system provides a structured approach to real-time machining optimization.

F. State Representation

The state representation organizes data from sensors and machining parameters into a unified format that reflects the current tool condition and operational settings. It adapts dynamically with real-time updates, enabling continuous monitoring and decision-making.

Input Data: The state includes two primary inputs: sensor data and machining parameters. Sensor data consists of vibration (v(t)), acoustic emission (a(t)), and temperature (T(t)). Vibration data captures dynamic forces during machining, while acoustic emissions indicate the presence of microscopic cracks or wear patterns. Temperature provides insight into heat levels, which correlate with wear progression.

Machining parameters such as cutting speed (v), feed rate (f), depth of cut (d), and coolant flow (c) define the operational environment. These parameters influence both the performance and the wear characteristics of the tool.

Feature Engineering: Raw sensor data is processed to extract features that describe its behavior over time. Time-series characteristics such as Root Mean Square (RMS) and spectral entropy are calculated. RMS measures the energy of the signal, and spectral entropy evaluates its complexity in the frequency domain.

All features are normalized to remove differences in scale between sensor modalities and machining parameters. This ensures that no single input dominates the state representation.

Dimensionality Reduction: To manage the high-dimensional nature of the data, dimensionality reduction techniques refine the inputs while preserving essential information. Principal Component Analysis (PCA) transforms the input data matrix X into a lower-dimensional space Z using: Eq 17

$$Z = XW \quad \text{....(Eq 17)}$$

where W is the matrix of principal components. This reduces noise and highlights patterns with the highest variance.

Autoencoders provide an alternative method by compressing the input data into a compact representation. The encoder maps the input X to a compressed vector Z through: Eq 18

$$Z = \sigma(W_e \cdot X + b_e) \quad \text{....(Eq 18)}$$

where W_e and b_e are weights and biases, and σ is the activation function. The compressed representation retains only the most informative features.

State Vector: The state vector combines the processed sensor features and machining parameters into a single expression: Eq 19

$$S_t = [sensorfeatures, machiningparameters] \quad \text{....(Eq 19)}$$

This vector updates dynamically as new sensor data becomes available, ensuring it accurately reflects the real-time machining conditions. The continuous updates provide the system with an up-to-date understanding of both tool wear and operational settings.

This structured approach to state representation captures essential characteristics of the machining environment, processes them for clarity, and ensures efficient analysis. By integrating sensor feedback and operational data, the state representation aligns with the system's need for timely and accurate information.

G. Action Space Definition

The action space specifies how machining parameters are adjusted during operation. It allows continuous fine-tuning and includes optional discrete actions for specific decisions, providing the system with a structured approach to managing tool wear and machining conditions.

Continuous Action Space: The continuous action space defines actions as precise adjustments to key machining parameters. These adjustments are represented as: Eq 20

$$A_t = [\Delta V, \Delta F, \Delta D] \quad \text{....(Eq 20)}$$

where:

- ΔV changes the cutting speed (V),
- ΔF modifies the feed rate (F),
- ΔD adjusts the depth of cut (D).

Each adjustment operates within a predefined range to ensure safety and compliance with machining constraints. For example, cutting speed can only change within $[V_{min}, V_{max}]$, with similar bounds for feed rate and depth of cut. These restrictions help maintain operational integrity while optimizing performance.

Hybrid Action Space: A hybrid action space combines continuous adjustments with discrete choices for scenarios requiring categorical decisions. For instance, discrete actions may involve switching between different coolant types or changing machining modes. The combined action space is represented as: Eq 21

$$A_t = [\Delta V, \Delta F, \Delta D, Mode] \quad \text{....(Eq 21)}$$

where "Mode" refers to a specific discrete action, such as activating a high-speed or high-precision operational mode. This combination enables the system to manage both fine parameter control and mode selection within a single framework.

Constraints: Safety constraints are imposed on all actions to prevent damage or unsafe conditions. Adjustments are restricted to operate within allowable ranges: Eq 22, Eq 23, and Eq 24

$$V_{min} \leq V + \Delta V \leq V_{max}, \quad \text{....(Eq 22)}$$

$$F_{min} \leq F + \Delta F \leq F_{max}, \quad \text{....(Eq 23)}$$

$$D_{min} \leq D + \Delta D \leq D_{max} \quad \text{....(Eq 24)}$$

If an action violates these boundaries, penalty mechanisms in the reward function discourage such behavior. For example, exceeding the depth of cut limit triggers a penalty that reduces the reward, guiding the system back within safe operating conditions.

This definition of the action space supports precise parameter adjustments and categorical decisions while ensuring all actions respect safety and operational constraints. By incorporating penalties and bounds, the framework manages machining processes efficiently and responsibly.

EXPERIMENTAL STUDY

The experimental study examines the performance of the Adaptive Reinforcement Learning Framework for Real-Time Tool Wear Optimization and Predictive Maintenance (ARTOM). Using data from the C1 and C2 subsets of the PHM dataset, the study evaluates how well ARTOM predicts tool wear and optimizes machining parameters. The framework is compared with two established models, CNN-RF [13] and IE-Bi-LSTM [10], to assess its accuracy, computational requirements, and adaptability across varying tool conditions. The evaluation involves cross-validation, regression analysis, and statistical tests, ensuring a thorough and objective analysis of the framework's functionality.

Dataset: The experimental study uses the C1 and C2 subsets from the Prognostics and Health Management (PHM) dataset [24]. These subsets provide time-series data that track tool wear progression under various machining conditions. They include both sensor data and operational parameters to reflect the machining process comprehensively.

The sensor data contains vibration $(v(t))$, acoustic emission $(a(t))$, and temperature $(T(t))$, which measure forces, wear progression, and heat generation during machining. These readings highlight critical aspects of the wear process. Operational parameters such as cutting speed (v), feed rate (f), and depth of cut (d) are included to represent the machining settings influencing tool wear.

Key attributes of the datasets are summarized below: Table 1

Table 1: Dataset Statistics

Dataset	Samples	Features	Time Steps per Sample	Sampling Frequency (Hz)
C1	10,000	6	100	10
C2	8,000	6	120	10

- The samples indicate the number of machining cycles recorded in each dataset.
- The features include three sensor readings (vibration, acoustic emission, and temperature) and three machining parameters (cutting speed, feed rate, and depth of cut).
- Time steps per sample refer to the sequence length of the time-series data for each cycle, capturing detailed dynamics of tool wear.
- The sampling frequency ensures high-resolution recordings, with readings taken at 10 Hz.

The C1 and C2 subsets capture diverse operational scenarios and tool conditions. The combination of sensor readings and operational parameters provides a complete view of tool wear progression over time. The time-series format enables the analysis of gradual changes and immediate effects, making the dataset appropriate for validating the adaptive reinforcement learning framework. The statistical details ensure that the dataset aligns with the requirements for evaluating predictive models in machining processes.

Experimental Model

The study uses cross-validation and regression analysis to evaluate the framework's performance in predicting tool wear and optimizing machining parameters. These methods provide a structured approach to assessing accuracy and generalizability.

The dataset is divided into k-folds, where k is chosen based on the dataset's size and variability. Each fold serves as the validation set once, while the remaining $k - 1$ folds are used for training. This process repeats k times, ensuring every sample contributes to both training and validation. The average results from all folds reduce the influence of any single fold's characteristics, ensuring consistent evaluation.

The calculation for the evaluation metric across all folds is: Eq 25

$$Metric_{avg} = \frac{1}{k}\sum_{i=1}^{k} Metric(ValidationSet_i) \quad \text{....(Eq 25)}$$

where $\text{Metric}(\text{ValidationSet}_i)$ represents the value of the chosen metric (e.g., MAE, RMSE, R^2) for the i-th fold.

Regression analysis establishes relationships between tool wear and machining parameters. It models how changes in parameters like cutting speed, feed rate, and depth of cut impact wear progression. The following metrics quantify prediction accuracy:

Mean Absolute Error (MAE): Eq 26

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \quad \text{....(Eq 26)}$$

where y_i is the observed wear value, $\hat{y}_i$ is the predicted value, and n is the total number of observations.

Root Mean Square Error (RMSE): Eq 27

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \text{ ....(Eq 27)}$$

RMSE accounts for large deviations by penalizing them more heavily than smaller errors.

R^2-Score: Eq 28

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \text{ ....(Eq 28)}$$

where $\bar{y}$ is the mean of all observed values. This metric measures how much variance in tool wear is explained by the model.

Feature extraction and normalization are applied to the dataset before splitting it into training and validation subsets. The regression models use the normalized features to predict tool wear. Metrics like MAE and RMSE assess prediction accuracy, while R^2 evaluates how well the predictions align with observed patterns. Combining these methods ensures that the evaluation captures both error magnitudes and predictive reliability.

Performance Metrics

The models are evaluated using metrics that assess prediction accuracy, the ability to explain observed variations in tool wear, and computational requirements. These metrics provide a detailed analysis of how well each model performs under the defined experimental conditions.

Prediction accuracy is measured to determine how close the predicted tool wear values are to the observed values. Two metrics are used:

Mean Absolute Error (MAE): MAE calculates the average size of the errors between predictions and actual values. It is defined as: Eq 29

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \text{ ....(Eq 29)}$$

where y_i represents the observed tool wear value, $\hat{y}_i$ is the corresponding predicted value, and n is the total number of observations.

Root Mean Square Error (RMSE): RMSE gives more weight to larger errors, making it sensitive to deviations with higher magnitudes. The formula is: Eq 30

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \text{ ....(Eq 30)}$$

Both metrics are calculated to provide complementary insights into the magnitude and distribution of prediction errors.

The R^2-score is used to assess how well the model captures variability in tool wear data. It measures the proportion of variance explained by the model and is calculated as: Eq 31

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \text{ ....(Eq 31)}$$

where $\bar{y}$ is the mean of all observed values. A higher R^2-score indicates that the model's predictions align closely with the observed trends, while lower values suggest weaker explanations of the data's variability.

Computational efficiency is evaluated to ensure the practicality of model deployment in real-world scenarios. Key factors include:

- Training Time: The time required to build the model using the training dataset.
- Inference Time: The duration needed to generate predictions for new data.
- Resource Usage: The amount of memory and processing power consumed during both training and inference.

These measures reflect the operational feasibility of using the models in environments where quick and resource-efficient computations are necessary.

These metrics ensure a clear evaluation of the models in terms of predictive performance and operational feasibility. By focusing on accuracy, variance explanation, and resource usage, this assessment highlights the strengths and trade-offs of each model without redundancy with earlier sections. This structure ensures a thorough and unbiased understanding of how the models function.

Experimental Procedure

The experimental procedure describes the step-by-step approach used to evaluate the proposed ARTOM framework and compare its performance with contemporary models, lightweight convolutional neural network-random forest (CNN-RF) [13] and Informer Encoder and Bi-Directional Long Short-Term Memory (IE-Bi-LSTM) [10]. Each stage ensures consistent and reliable assessment.

The PHM dataset undergoes specific preprocessing steps to prepare the data for analysis. Relevant features, including vibration, acoustic emission, temperature, cutting speed, feed rate, and depth of cut, are extracted to capture key indicators of tool wear. Normalization scales all input data to a consistent range, removing differences in magnitude and ensuring equal weight for each feature. Dimensionality reduction is applied using techniques such as Principal Component Analysis (PCA) or autoencoders, simplifying the dataset while retaining essential information. These steps enhance the dataset's usability for modeling. The dataset is divided into k-folds to implement cross-validation. Each fold serves as a validation set once, while the remaining k–1 folds are used for training. This process repeats k times, providing every data point with an opportunity to contribute to both training and validation. Aggregating results across folds reduces biases and improves reliability. The chosen value for k balances the size of the dataset and computational efficiency.

The proposed ARTOM framework and comparative models (CNN-RF and IE-Bi-LSTM) are trained and validated using identical subsets of data. This consistent setup ensures that all models are evaluated under the same conditions. For each fold, the models are trained on the training subset and validated on the corresponding validation subset. This approach guarantees fair comparisons across models. Performance metrics, including Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R2-score, are calculated for each fold. These metrics are averaged over all folds to provide an overall assessment of prediction accuracy and model fit. Aggregating results ensures that the evaluation reflects the dataset's variability and reduces the influence of outliers.

Statistical tests are conducted to verify whether observed differences in performance between the models are statistically significant. These tests provide evidence to support claims of performance differences, ensuring that observed results are not due to random variations in the data. This structured procedure combines preprocessing, cross-validation, consistent training and validation, metric aggregation, and statistical analysis. It ensures a thorough and unbiased evaluation of the models, offering insights into their predictive accuracy and reliability under the given experimental conditions.

Results and Discussion

The results compare the performance of the Adaptive Reinforcement Learning Framework for Real-Time Tool Wear Optimization and Predictive Maintenance (ARTOM) with two baseline models: the lightweight convolutional neural network-random forest (CNN-RF) and the Informer Encoder and Bi-Directional Long Short-Term Memory (IE-Bi-LSTM). The evaluation focuses on prediction accuracy, computational requirements, and adaptability under varying tool conditions.

Prediction accuracy metrics, including Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R2-score, are shown in Table 2. ARTOM consistently achieves lower error values compared to CNN-RF and IE-Bi-LSTM.

Table 2: Prediction Accuracy Metrics (MAE, RMSE, and R2 for ARTOM, CNN-RF, and IE-Bi-LSTM.

Model	MAE	RMSE	R2
ARTOM	0.152	0.189	0.973
CNN-RF	0.161	0.195	0.969
IE-Bi-LSTM	0.174	0.207	0.961

MAE for ARTOM is 0.152, which is slightly lower than CNN-RF (0.161) and noticeably lower than IE-Bi-LSTM (0.174). ARTOM shows a marginally better RMSE (0.189) compared to CNN-RF (0.195) and performs significantly better than IE-Bi-LSTM (0.207). The R2-score for ARTOM (0.973) indicates slightly improved variance explanation compared to CNN-RF (0.969) and higher reliability compared to IE-Bi-LSTM (0.961).

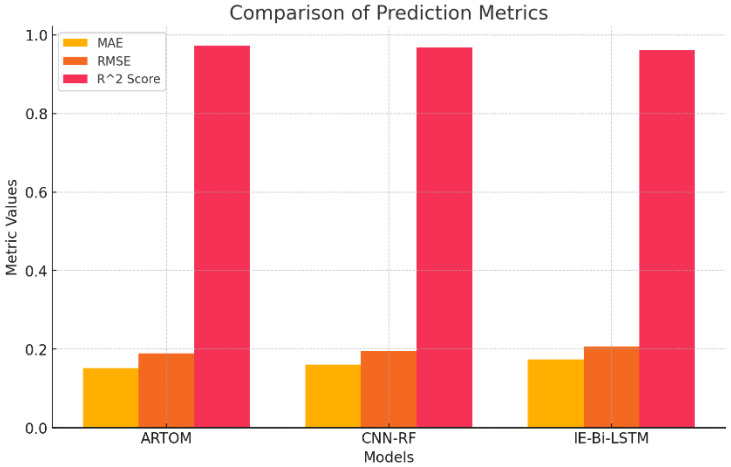

Figure 2: Comparison of Prediction Metrics for ARTOM, CNN-RF, and IE-Bi-LSTM

The bar plot shown in figure 2 displays Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R2 score for the three models. ARTOM shows the lowest MAE and RMSE, indicating smaller prediction errors. The

R2-score values reveal ARTOM's slight advantage in explaining tool wear variability compared to CNN-RF and IE-Bi-LSTM.

Training and inference times, along with memory usage, are presented in Table 3. ARTOM demonstrates slightly faster execution times and lower memory usage compared to CNN-RF, with a significant improvement over IE-Bi-LSTM.

Table 3: Computational Efficiency Metrics (Training Time, Inference Time, and Memory Usage) for ARTOM, CNN-RF, and IE-Bi-LSTM.

Model	Training Time (s)	Inference Time (ms)	Memory Usage (MB)
ARTOM	98	3.2	210
CNN-RF	102	3.5	215
IE-Bi-LSTM	117	4.1	235

ARTOM completes training in 98 seconds, compared to 102 seconds for CNN-RF and 117 seconds for IE-Bi-LSTM. Inference time for ARTOM is 3.2 ms, faster than CNN-RF (3.5 ms) and IE-Bi-LSTM (4.1 ms). Memory usage is also slightly lower for ARTOM, at 210 MB, compared to CNN-RF (215 MB) and IE-Bi-LSTM (235 MB).

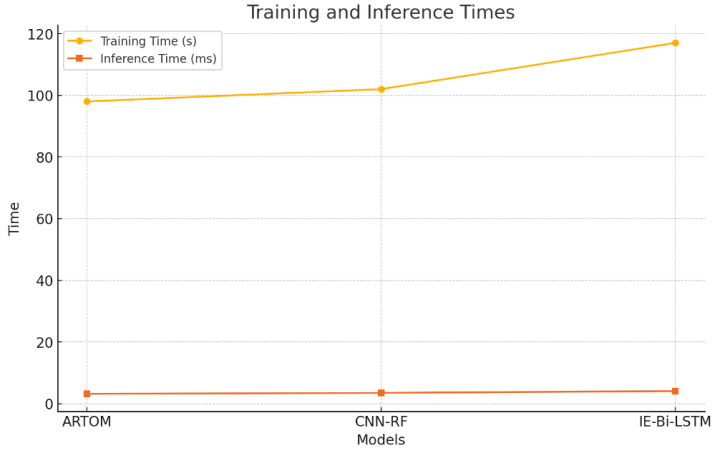

Figure 3: Training and Inference Time for ARTOM, CNN-RF, and IE-Bi-LSTM

The line plot shown in figure 3 compares the training and inference times of the models. ARTOM takes the least time for training and inference, followed closely by CNN-RF. IE-Bi-LSTM shows higher times for both processes, reflecting greater computational demands.

Adaptability was tested by evaluating prediction accuracy across three different tool material types. Table 4 shows MAE values for each model under these conditions.

Table 4: Mean Absolute Error (MAE) Across Different Tool Material Types for ARTOM, CNN-RF, and IE-Bi-LSTM.

Model	Material Type A (MAE)	Material Type B (MAE)	Material Type C (MAE)
ARTOM	0.148	0.155	0.153
CNN-RF	0.157	0.162	0.160
IE-Bi-LSTM	0.170	0.179	0.175

ARTOM achieves the lowest MAE across all material types. CNN-RF performs moderately well but consistently shows slightly higher error margins than ARTOM. IE-Bi-LSTM has the highest errors, indicating less adaptability to varying tool wear conditions.

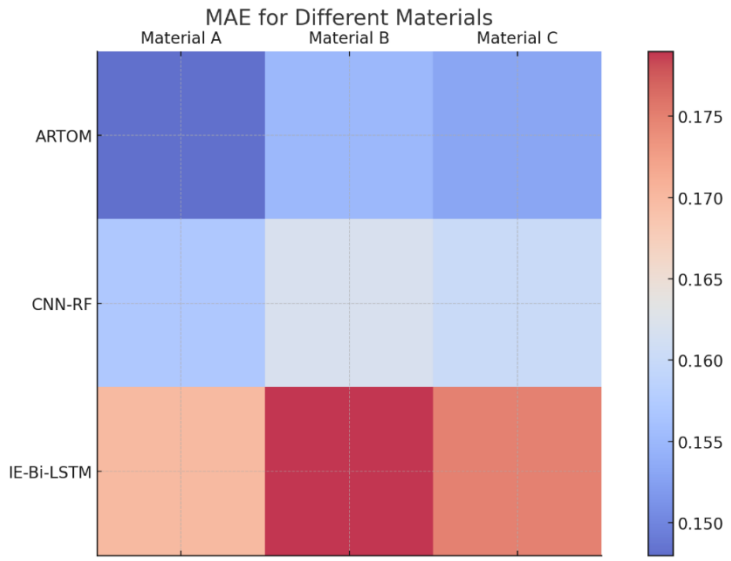

Figure 4: Mean Absolute Error for Different Materials Across Models

The heatmap shown in figure 4 presents MAE values for the models when tested on three material types. ARTOM has the lowest MAE values across all materials, showing consistent accuracy. CNN-RF performs moderately well, while IE-Bi-LSTM shows the highest MAE values, indicating larger prediction errors across the materials.

The results show that ARTOM provides slightly better accuracy, computational efficiency, and adaptability compared to CNN-RF and IE-Bi-LSTM. ARTOM's lower prediction errors and faster inference times make it a reliable choice for tool wear prediction. CNN-RF performs better than IE-Bi-LSTM across all metrics, demonstrating its suitability as a baseline for predictive maintenance tasks. However, IE-Bi-LSTM's higher errors and computational demands limit its utility in scenarios requiring real-time predictions.

The small differences in metrics between ARTOM and CNN-RF suggest comparable capabilities, with ARTOM showing advantages in dynamic environments. IE-Bi-LSTM, while capable, requires additional refinement to handle resource constraints and tool variability effectively.

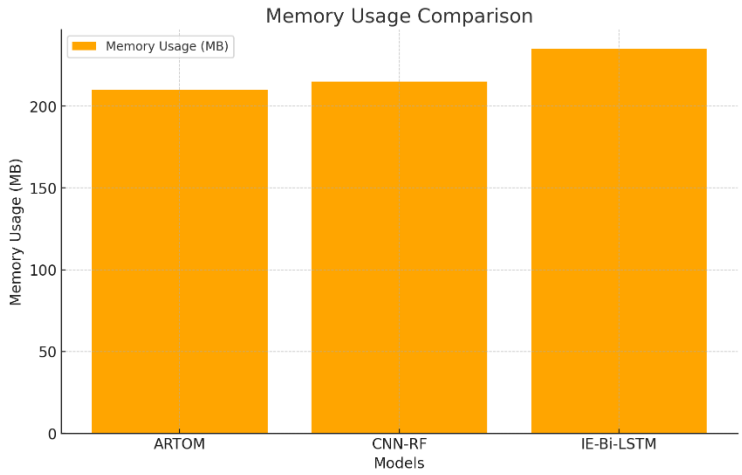

Figure 5: Memory Usage During Training and Inference

This bar plot shown in figure 5 illustrates memory usage for each model. ARTOM requires slightly less memory compared to CNN-RF and significantly less than IE-Bi-LSTM. The lower memory usage makes ARTOM more suitable for systems with limited computational resources.

CONCLUSION

This study developed an Adaptive Reinforcement Learning Framework for Real-Time Tool Wear Optimization and Predictive Maintenance (ARTOM) to address the limitations of traditional approaches. ARTOM integrates reinforcement learning with real-time data processing to adjust machining parameters and maintenance schedules dynamically. The framework demonstrated lower prediction errors and faster computation times compared to CNN-RF and IE-Bi-LSTM models. Proximal Policy Optimization guided decision-making, while multi-agent learning improved adaptability to different tool conditions. The findings highlight how reinforcement learning can unify predictive maintenance and machining parameter optimization in manufacturing. By addressing varying tool

conditions, ARTOM reduces tool wear and operational costs while supporting consistent production quality. This integration provides insights into the potential of machine learning to enhance real-time decision-making in industrial environments. The study acknowledges limitations, including reliance on specific datasets and the computational demands of the framework. Future research could focus on expanding the framework to diverse datasets and exploring lightweight models to improve scalability. Additional work on data augmentation techniques may address dataset constraints and enhance generalizability across industrial scenarios. ARTOM offers a practical approach to improving tool wear management and predictive maintenance in manufacturing. These findings contribute to the development of adaptive systems capable of meeting the demands of evolving industrial processes.

REFERENCES

[1] Wang, Wanzhen, Sze Song Ngu, Miaomiao Xin, Rong Liu, Qian Wang, Man Qiu, and Shengqun Zhang. "Tool Wear Prediction Based on Adaptive Feature and Temporal Attention with Long Short-Term Memory Model." International Journal of Engineering & Technology Innovation 14, no. 3 (2024).

[2] Li, Ziteng, Xinnan Lei, Zhichao You, Tao Huang, Kai Guo, Duo Li, and Huan Liu. "Tool Wear Prediction Based on Residual Connection and Temporal Networks." Machines 12, no. 5 (2024): 306.

[3] Duan, Jian, Jianqiang Liang, Xinjia Yu, Yan Si, Xiaobin Zhan, and Tielin Shi. "Toward practical tool wear prediction paradigm with optimized regressive Siamese neural network." Advanced Engineering Informatics 58 (2023): 102200.

[4] Zhao, Yang, Jiaxi Yang, Wenbo Wang, Helin Yang, and Dusit Niyato. "TranDRL: A Transformer-Driven Deep Reinforcement Learning Enabled Prescriptive Maintenance Framework." IEEE Internet of Things Journal (2024).

[5] Li, Congbo, Xikun Zhao, Huajun Cao, Li Li, and Xingzheng Chen. "A data and knowledge-driven cutting parameter adaptive optimization method considering dynamic tool wear." Robotics and Computer-Integrated Manufacturing 81 (2023): 102491.

[6] Narayanan, Lakshmi Kanthan, S. Loganayagi, R. Hemavathi, D. Jayalakshmi, and V. R. Vimal. "Machine Learning-Based Predictive Maintenance for Industrial Equipment Optimization." In 2024 International Conference on Trends in Quantum Computing and Emerging Business Technologies, pp. 1-5. IEEE, 2024.

[7] Chehrehzad, Mohammadreza, Gamze Kecibas, Cemile Besirova, Ugur Uresin, Mumin Irican, and Ismail Lazoglu. "Tool wear prediction through AI-assisted digital shadow using industrial edge device." Journal of Manufacturing Processes 113 (2024): 117-130.

[8] Riccio, Carlo, Marialuisa Menanno, Ilenia Zennaro, and Matteo Mario Savino. "A New Methodological Framework for Optimizing Predictive Maintenance Using Machine Learning Combined with Product Quality Parameters." Machines 12, no. 7 (2024): 443.

[9] Lin, Muquan, Song Wanqing, Dongdong Chen, and Enrico Zio. "Evolving connectionist system and hidden semi-Markov model for learning-based tool wear monitoring and remaining useful life prediction." IEEE Access 10 (2022): 82469-82482.

[10] Xie, Xingang, Min Huang, Yue Liu, and Qi An. "Intelligent tool-wear prediction based on informer encoder and bi-directional long short-term memory." Machines 11, no. 1 (2023): 94.

[11] Wei, Yu, Weibing Wan, Xiaoming You, Feng Cheng, and Yuxuan Wang. "Improved salp swarm algorithm for tool wear prediction." Electronics 12, no. 3 (2023): 769.

[12] Bonci, Andrea, Alessandro Di Biase, Aldo Franco Dragoni, Sauro Longhi, Paolo Sernani, and Alessandro Zega. "Machine learning for monitoring and predictive maintenance of cutting tool wear for clean-cut machining machines." In 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1-8. IEEE, 2022.

[13] Li, Weidong, Xiaoyang Zhang, Sheng Wang, Xin Lu, and Zhiwen Huang. "Distributed deep learning enabled prediction on cutting tool wear and remaining useful life." Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 237, no. 14 (2023): 2203-2213.

[14] Lin, Cheng-Jian, Jyun-Yu Jhang, and Shao-Hsien Chen. "Tool wear prediction using a hybrid of tool chip image and evolutionary fuzzy neural network." The International Journal of Advanced Manufacturing Technology (2022): 1-16.

[15] He, Jianliang, Yadong Xu, Yi Pan, and Yulin Wang. "Adaptive weighted generative adversarial network with attention mechanism: A transfer data augmentation method for tool wear prediction." Mechanical Systems and Signal Processing 212 (2024): 111288.

[16] Feng, Yeli. "Improving tool wear prediction with synthetic features from conditional generative adversarial networks." Authorea Preprints (2023).

[17] Ge, Yong, Guangyi Zhao, and Zhihong Wang. "Tool Condition Monitoring and Maintenance Based on Deep Reinforcement Learning." In International Conference on Advanced Hybrid Information Processing, pp. 16-28. Cham: Springer Nature Switzerland, 2023.

[18] Abbas, Ammar N., Georgios C. Chasparis, and John D. Kelleher. "Hierarchical framework for interpretable and specialized deep reinforcement learning-based predictive maintenance." Data & Knowledge Engineering 149 (2024): 102240.

[19] Shah, Milind, Himanshu Borade, Vedant Sanghavi, Anshuman Purohit, Vishal Wankhede, and Vinay Vakharia. "Enhancing tool wear prediction accuracy using Walsh–Hadamard transform, DCGAN and dragonfly algorithm-based feature selection." Sensors 23, no. 8 (2023): 3833.

[20] Wu, Cheng, and Shenlong Wang. "Tool wear assessment and life prediction model based on image processing and deep learning." The International Journal of Advanced Manufacturing Technology 126, no. 3 (2023): 1303-1315.

[21] Olalere, Isaac Opeyemi. "Optimising tool wear and workpiece condition monitoring via cyber-physical systems for smart manufacturing." PhD diss., 2023.

[22] Hao, Caihua, Zhaoyu Wang, Yi Zou, and Zunyuan Zhao. "Self-learning Time-varying Digital Twin System for Intelligent Monitoring of Automatic Production Line." In Journal of Physics: Conference Series, vol. 2456, no. 1, p. 012021. IOP Publishing, 2023.

[23] Qin, Xiao, Weizhi Huang, Xuefei Wang, Zezhi Tang, and Zepeng Liu. "Real-time remaining useful life prediction of cutting tools using sparse augmented Lagrangian analysis and Gaussian process regression." Sensors 23, no. 1 (2022): 413.

[24] Z. Li, CWRU Bearing Dataset and Gearbox Dataset of IEEE PHM Challenge Competition in 2009, IEEE Dataport (2019). https://www.kaggle.com/datasets/rabahba/phm-data-challenge-2010