Predictive maintenance of Robotic Systems using Machine Learning

Gayatri Moitra 21BRS1293
Alexia Prince Cheenath 21BRS1106

***Abstract*—This project focuses on using machine learning (ML) to predict equipment failures and schedule maintenance for robotic systems, aiming to reduce downtime and enhance reliability. By analyzing historical sensor data, ML models are trained to detect patterns indicative of impending failures. The project involves data preprocessing, feature engineering, model selection, and evaluation to ensure the accuracy and robustness of the predictive maintenance system. Once deployed into the robotic systems' software infrastructure, the trained models continuously monitor real-time sensor data. They trigger alerts and maintenance schedules when potential failure conditions are detected, enabling preemptive actions to prevent disruptions in robotic operations. Through this project, we aim to demonstrate the effectiveness of ML-based predictive maintenance in improving the efficiency and reliability of robotic systems across various industries and applications.**

***Keywords*—Machine Learning, Predictive Maintenance, Robotic Systems, Sensor Data Analysis, Model Selection, Failure Detection**

I. INTRODUCTION

Robotic systems play a pivotal role in modern industries, offering automation solutions that enhance productivity, efficiency, and precision in various applications. However, despite their technological advancements, robotic systems are susceptible to unexpected failures, which can lead to downtime, productivity losses, and safety hazards. Traditional maintenance practices often rely on reactive approaches, where maintenance actions are taken only after a failure occurs, resulting in costly repairs and disruptions to operations. In contrast, proactive maintenance strategies aim to prevent failures before they happen, thereby minimizing downtime and optimizing the reliability of robotic systems.

This project introduces a proactive maintenance approach based on machine learning (ML) techniques, specifically tailored for robotic systems. The core objective is to develop a predictive maintenance system that can pre-emptively identify and address potential failures in robotic equipment, thereby reducing downtime, enhancing reliability, and improving safety in industrial automation environments. By analyzing historical sensor data collected from the robotic systems, the predictive maintenance system can detect patterns indicative of impending failures and schedule maintenance activities accordingly. This proactive approach not only minimizes the risk of unexpected downtime but also enables efficient resource allocation and optimization of maintenance schedules.

A. Motivation

The motivation behind this project stems from the growing demand for predictive maintenance solutions in industrial automation. As industries increasingly rely on robotic systems for various tasks, ensuring their continuous operation and reliability becomes paramount. The ability to predict equipment failures before they occur offers significant advantages in terms of cost savings, operational efficiency, and safety. By proactively addressing maintenance needs, organizations can avoid costly repairs, minimize production losses, and ensure uninterrupted workflow.

B. Objectives

The primary objective of this project is to develop and implement a machine learning-based predictive maintenance system for robotic systems. Specific goals include:

A. Collecting and preprocessing historical sensor data from robotic systems to prepare it for analysis.
B. Engineering relevant features from the sensor data to capture patterns indicative of potential failures.
C. Selecting and training machine learning models capable of accurately predicting equipment failures based on the engineered features.
D. Evaluating the performance of the predictive maintenance system using appropriate metrics and validation techniques.
E. Demonstrating the effectiveness of the predictive maintenance system through real-world case studies in industrial settings.

C. Scope

This project focuses on the development and implementation of a predictive maintenance system for robotic systems in industrial automation environments. The scope encompasses data collection, preprocessing, feature engineering, model selection, training, evaluation, and validation of the predictive maintenance system. The emphasis is on leveraging machine learning techniques to analyze historical sensor data and predict equipment failures proactively. While the project's primary focus is on robotic systems, the proposed methodology and framework can be adapted and extended to other domains and applications requiring proactive maintenance strategies.

II. METHODOLOGY

A. Data Collection and Preprocessing

1. Data Gathering

The data was gathered and collected from a dataset from kaggle.[1]

2. Data Preprocessing

The data was preprocessed in Jupyter Notebook.
a. Loading the data:

```
dataset=pd.read_csv("robotic_systems_data.csv")
dataset.head()
```

b. Describing the data

```
dataset.describe()
```

c. Getting information

```
dataset.info()
```

d. Finding the numbers of rows and columns

```
dataset.shape
```

e. Finding null values

```
dataset.isnull().any()
```

f. Removing redundancy

```
typesTrainData=pd.get_dummies(dataset["Type"],drop_first=False)
typesTrainData.head()
```

```
dataset.drop(["id","Product ID","Type"],axis=1,inplace=True)
dataset.head()
```

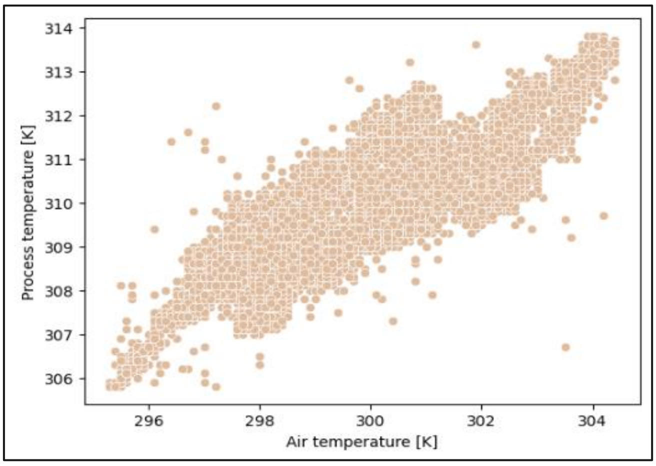
```
y=dataset['Machine failure']
dataset.drop(['Machine failure'],axis=1,inplace=True)
```

g. Label encoding

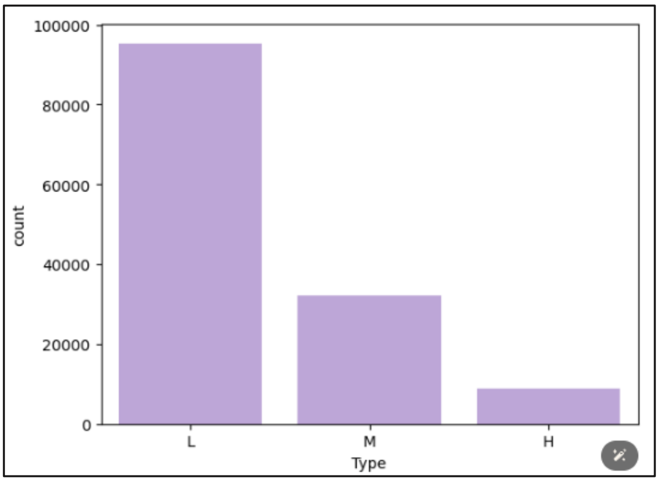
```
map_values={True: 1,False: 0}
typesTrainData['H']=typesTrainData['H'].map(map_values)
typesTrainData['L']=typesTrainData['L'].map(map_values)
typesTrainData['M']=typesTrainData['M'].map(map_values)
typesTrainData.head()
```

f. Plotting

```
sns.scatterplot(data=dataset,x="Air temperature [K]",y="Process temperature [K]",color="#DFBC9E")
plt.show()
```




```
sns.countplot(data=dataset,x="Type",color="#BC9EDF")
plt.show()
```



C. Model Selection and Training

We divide the dataset into training and testing dataset

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(dataset,y,test_size=0.25,random_state=0)
print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

1. Algorithm Selection

Here, we put our dataset through multiple algorithms to check which one gives the best accuracy.

The models that we had used are:

a. XGM Classifier
b. KNN
c. Random Forest Classifier
d. Logistic Regression
e. Decision Tree
f. SVC

2. Training Process

a. XGM Classifier

```
from xgboost import XGBClassifier
xgb_model=XGBClassifier()

xgb_model=XGBClassifier(n_estimators=100,objective='binary:logistic',
                        tree_method='hist',max_depth=3,learning_rate=0.1)
xgb_model.fit(x_train,y_train)
```

b. KNN

```
from sklearn.neighbors import KNeighborsClassifier
knn_model=KNeighborsClassifier()

knn_model=KNeighborsClassifier(n_neighbors=2)
knn_model.fit(x_train,y_train)
```

c. Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
rf_model=RandomForestClassifier()

rf_model=RandomForestClassifier(n_estimators=100,random_state=0)
rf_model.fit(x_train,y_train)
```

d. Logistic Regression

```
from sklearn.linear_model import LogisticRegression
lr_model=LogisticRegression()

lr_model.fit(x_train,y_train)
```

e. Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
dt_model=DecisionTreeClassifier()

dt_model=DecisionTreeClassifier(criterion="entropy",
                    max_depth=3,min_samples_split=4,random_state=0)
dt_model.fit(x_train,y_train)
```

f. SVC

```
from sklearn.svm import SVC
svc_model=SVC()

svc_model=SVC(random_state=0)
svc_model.fit(x_train,y_train)
```

II. RESULTS AND DISCUSSION

A. Performance Metrics

1. Accuracy provided by each model:

Model	Train Accuracy	Test Accuracy
XGM Classifier	0.996071187732723 5	0.9965110824440014
KNN	0.988272202187234 3	0.983728157616981 4
Random Forest	0.999863175692184 4	0.996364489269379 6
Logistic Regression	0.994966820105354 8	0.994869238888237 4
Decision Tree	0.994859315292 07 11	0.994605371173918 1
SVC	0.984343389919957 7	0.9839920253313006

Random Forest Classifier gave the highest accuracy hence the prediction of the model was done with the said model.

```
rf_model.predict([[299.8,8010.3,1427,36.4,0,0,0,0,0,0,1,0]])
```

Output:

```
array([0])
```

B. App Building

1. Implementing pickle

```
import pickle
pickle.dump(rf_model,open('model.pkl','wb'))
```

2. Code

```
import streamlit as st
import pickle
import numpy as np

# Load the pre-trained model
with open("model.pkl", "rb") as f:
    model = pickle.load(f)

st.markdown('<div class="full-page">', unsafe_allow_html=True)

with st.container():
    # Title
    st.title("Predictive Maintenance of Robotic Systems")

    # Collect user inputs
    types=[]
    input_values=[]
    air_temp=st.text_input("Enter Air Temperature(K): ")
    process_temp=st.text_input("Enter Process Temperature(K): ")
    rot_speed=st.text_input("Enter Rotational Speed(rpm): ")
    tor=st.text_input("Enter the Torque(Nm): ")
    tool=st.text_input("Enter Tool Wear(min): ")
    twf=st.text_input("Enter Tool Wear Failure(0/1): ")
    hdf=st.text_input("Enter Heat Dissipation Failure(0/1): ")
    pwf=st.text_input("Enter Power Failure(0/1): ")
    osf=st.text_input("Enter Overstrain Failure(0/1): ")
    rnf=st.text_input("Enter Random Failure(0/1): ")
    typ=st.text_input("Enter Type(H/L/M): ")
    if typ=='H':
        types=[1,0,0]
    elif typ=='L':
        types=[0,1,0]
    elif typ=='M':
        types=[0,0,1]
    input_values=[air_temp,process_temp,rot_speed,tor,tool,twf,hdf,pwf,osf,rnf]+types
    # Button to trigger prediction
    if st.button("Predict"):
        # Convert input values to an array (assuming numeric inputs)
        input_array = np.array([float(val) for val in input_values]).reshape(1, -1)

        # Predict using the model
        prediction = model.predict(input_array)
        if prediction==1:
            st.write("Machine Failure")
        else:
            st.write("Machine is Safe for Use")


st.markdown('</div>', unsafe_allow_html=True)
```

3. Code Breakdown

a. Introduction

The following report details the development of a Streamlit web application designed for predictive maintenance of robotic systems. The application utilizes a pre-trained machine learning model to predict potential failures in robotic equipment based on user-provided input parameters.

b. Model Loading

The application loads a pre-trained machine learning model stored in a serialized format using the pickle library. This model is trained on historical sensor data from robotic systems and is capable of predicting equipment failures.

c. Streamlit App Layout

The application interface is built using Streamlit, a Python library for creating web applications. The layout of the application is designed to occupy the entire page, providing a seamless user experience.

d. User Input Collection

Users are prompted to input various parameters related to the robotic system, including air temperature, process temperature, rotational speed, torque, tool wear, and indicators of potential failures such as tool wear failure, heat dissipation failure, and power failure. Additionally, users can specify the type of failure (High, Low, Medium), which is converted into a one-hot encoded format for model input.

e. Prediction Button

A prediction button is provided to trigger the prediction process based on the user-provided input values. When clicked, the input values are converted into a numpy array and reshaped to match the expected input format of the pre-trained model.

f. Prediction Display

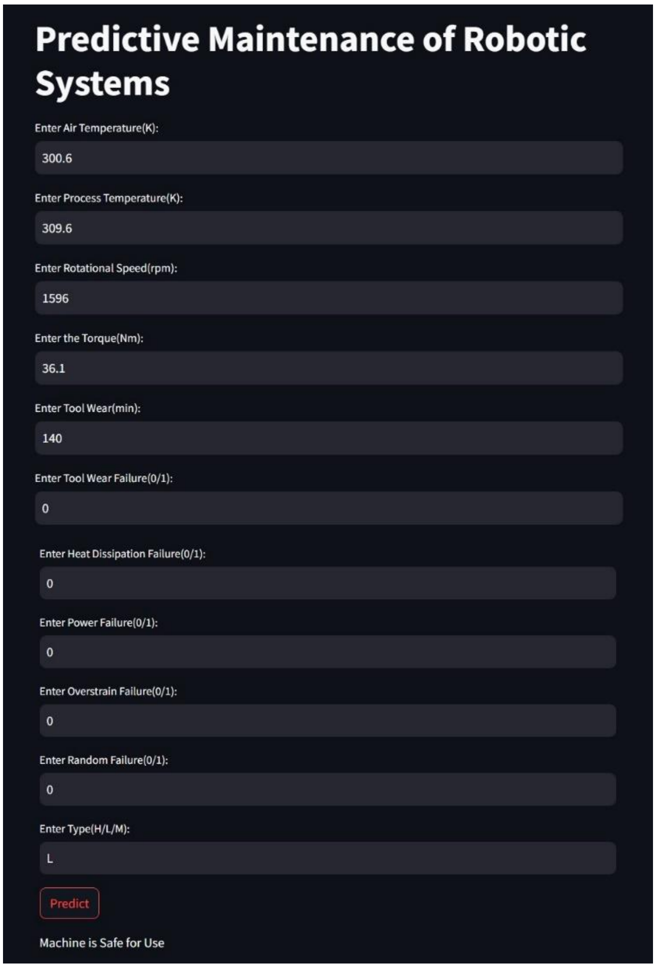
Upon prediction, the application displays a message indicating whether the machine is predicted to fail or is safe for use. This information is derived from the prediction made by the machine learning model.

g. Rendering HTML Elements

The application utilizes the st.markdown function to render HTML elements, allowing for custom styling and layout control. This ensures a visually appealing and user-friendly interface for interacting with the application.

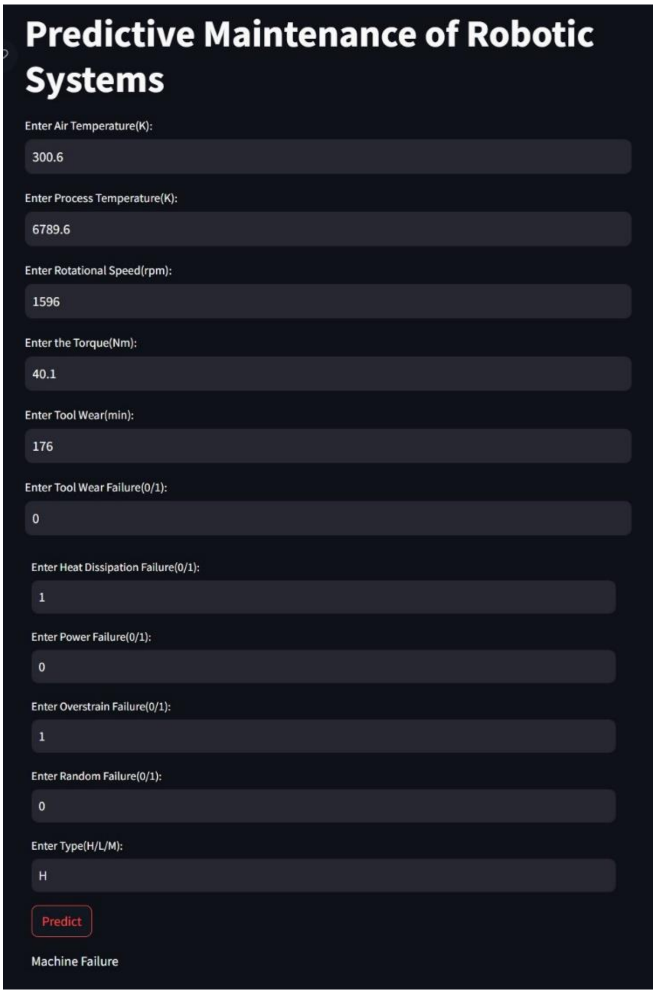
4. *App Interface*

Output 1:



▲ When the Machine is safe to use.

Output 2:



▲ When there is a Machine failure.

5. *Conclusion*

In conclusion, the developed Streamlit web application provides a convenient platform for users to input parameters related to robotic systems and receive predictions on potential equipment failures. The integration of a pre-trained machine learning model enables proactive maintenance strategies, thereby enhancing the reliability and efficiency of robotic systems in industrial settings.

III. FUTURE SCOPES

A. Conclusion

The predictive maintenance system developed for robotic systems using machine learning techniques represents a significant advancement in proactive maintenance strategies. While the current implementation has demonstrated promising results in reducing downtime and enhancing reliability, several avenues for future research and development can further enhance the system's capabilities and applicability in industrial automation environments.

1. Model Refinement and Optimization:

Future research efforts can focus on refining and optimizing the machine learning models used for predictive maintenance. This may involve exploring advanced algorithms, ensemble methods, or deep learning techniques to improve prediction accuracy and robustness. Additionally, hyperparameter tuning and feature selection methods can be employed to enhance model performance further.

2. Incorporation of Additional Data Sources:

To capture a comprehensive understanding of equipment health and performance, future iterations of the predictive maintenance system can incorporate additional data sources beyond sensor data. Integration with maintenance logs, historical records, and external environmental factors can provide valuable insights into equipment behavior and failure patterns, enhancing the system's predictive capabilities.

3. Real-Time Monitoring and Adaptive Maintenance Strategies:

The integration of real-time monitoring capabilities into the predictive maintenance system can enable proactive decision-making and adaptive maintenance strategies. By continuously monitoring sensor data streams and equipment status in real-time, the system can dynamically adjust maintenance schedules, prioritize critical maintenance tasks, and respond to emerging failure conditions promptly.

4. Scalability and Generalization Across Industries:

While the current focus is on robotic systems in industrial automation, future research can explore the scalability and generalization of the predictive maintenance system across diverse industries and applications. Adapting the system to different types of equipment, operational environments, and maintenance requirements can extend its utility and address broader challenges in predictive maintenance.

B. Future Directions

1. Integration with IoT and Edge Computing:

Leveraging the capabilities of the Internet of Things (IoT) and edge computing technologies can facilitate seamless data collection, processing, and analysis for predictive maintenance. Integrating sensor-equipped devices, edge computing platforms, and cloud infrastructure can create a robust ecosystem for real-time monitoring and predictive analytics, enabling proactive maintenance actions at scale.

2. Advanced Analytics for Anomaly Detection and Root Cause Analysis:

Future research can focus on developing advanced analytics techniques for anomaly detection and root cause analysis in equipment failures. By combining machine learning algorithms with domain-specific knowledge and expert systems, the predictive maintenance system can identify subtle deviations from normal behavior, diagnose underlying issues, and recommend targeted interventions to mitigate risks effectively.

3. Collaboration with Industry Partners and Validation in Real-World Settings:

Collaboration with industry partners and validation of the predictive maintenance system in real-world operational settings are essential steps towards its adoption and deployment. Conducting pilot studies, field trials, and case studies in collaboration with industrial stakeholders can provide valuable insights into the system's effectiveness, usability, and economic impact, facilitating its integration into existing maintenance workflows.

4. Continuous Evaluation and Improvement:

Continuous evaluation and improvement of the predictive maintenance system are critical to ensuring its long-term effectiveness and relevance. Monitoring performance metrics, gathering user feedback, and iteratively refining the system based on real-world observations are essential practices for maintaining its accuracy, reliability, and scalability over time.

IV. ACKNOWLEDGMENTS

The authors would like to express their gratitude to Vellore Institute of Technology, Chennai, for providing the necessary resources and support for conducting this research project. Special thanks are extended to Professor Dr. Jannath Nisha O.S. for her invaluable guidance, mentorship, and expertise throughout the duration of this project.

V. REFERENCES

1. Daoudi, N., Smail, Z., Aboussaleh, M. (2023). Machine Learning Based Predictive Maintenance: Review, Challenges and Workflow. In: Masrour, T., El Hassani, I., Barka, N. (eds) Artificial Intelligence and Industrial Applications. A2IA 2023. Lecture Notes in Networks and Systems, vol 771. Springer, Cham.
https://doi.org/10.1007/978-3-031-43524-9_6.

2. Meddaoui, A., Hachmoud, A. & Hain, M. Advanced ML for predictive maintenance: a case study on remaining useful life prediction and reliability enhancement. Int J Adv Manuf Technol 132, 323–335 (2024).
https://doi.org/10.1007/s00170-024-13351-y

3. Chakroun, A., Hani, Y., Elmhamedi, A. et al. A predictive maintenance model for health assessment of an assembly robot based on machine learning in the context of smart plant. J Intell Manuf (2024). https://doi.org/10.1007/s10845-023-02281-3

4. Borgi, Tawfik & Hidri, Adel & Neef, Benjamin & Mohamed Saber, Naceur. (2017). Data analytics for predictive maintenance of industrial robots. 412-417. 10.1109/ASET.2017.7983729.

5. D. V. Lindberg and H. K. H. Lee, “Optimization under constraints by applying an asymmetric entropy measure,” Journal of Computational and Graphical Statistics, vol. 24, no. 2, pp. 379–393,Jun.

2015. [Online]. Available: DOI: 10.1080/10618600.2014.901225.

6. B. Rieder, Engines of Order: A Mechanology of Algorithmic Techniques. Amsterdam, Netherlands: Amsterdam Univ. Press, 2020.

7. I. Boglaev, “A numerical method for solving nonlinear integro-differential equations of Fredholm type,” Journal of Computational Mathematics, vol. 34, no. 3, pp. 262–284, May 2016. [Online]. Available: DOI: 10.4208/jcm.1512-m2015-0241.