Fault Prediction and Preventive Maintenance for Wind Turbines Using Machine Learning

*EECE 690 Introduction to Machine Learning Project – Maroun Semaan Faculty of Engineering and Architecture - American University of Beirut

Amina Iskandarani
Dept. of Electrical and Computer Engineering *American University of Beirut* Beirut, Lebanon ami43@mail.aub.edu

Ahmad Daher
Dept. of Electrical and Computer Engineering *American University of Beirut* Beirut, Lebanon ahd43@mail.aub.edu

Ahmad Younes
Dept. of Electrical and Computer Engineering *American University of Beirut* Beirut, Lebanon any04@mail.aub.edu

***Abstract*—This paper explores the modeling of a machine learning-based framework for predictive maintenance in wind turbines with the aid of SCADA sensor data. The system leverages synthetic and historical fault data, dimensionality reduction, and anomaly detection through Random Forest and Multi-Layer Perception. The solution aims to flag abnormal behavior early, reduce downtime, and enhance turbine reliability.**

***Index Terms*—Wind Turbines, Predictive Maintenance, Fault Detection, SCADA Data, Anomaly Detection, PCA, Machine Learning, Classification, Accuracy, Loss, Random Forest, Multi-Layer Perceptron**

I. Introduction

Due to the drastic effects of greenhouse gas (GHG) emissions, several agreements, such as the Paris Climate Agreement held in December 2015, have highlighted the importance of decarbonization in the power industry. Consequently, the world has referred to non-fossil fuel electricity generation sources that promote a greener environment. Moreover, renewable energy sources (RES) provide energy security and political stability, which both stand equally essential to guarantee a country's sustainability. The dependence on RES makes their reliability and continuity crucial. For that, early detection and maintenance of faults are essential to avoid downtime and full reliance on generator grids. To tackle such an issue, this project investigates two supervised learning algorithms. Due to the availability of SCADA sensor recordings, models are to be trained to ensure correct and robust predictions to detect faults early based on wind turbines' behavior.

II. Fault Detection and Maintenance in Wind Turbines

Novel wind turbines are complicated and huge electromechanical systems faced with highly variable environmental and operational conditions. Due to unpredictable conditions, deterioration and faults are possible. Fault detection in early stages is critical to reduce downtime and maintenance costs.

Classical fault detection techniques depend on scheduled maintenance and rule-based thresholds applied to SCADA data. For instance, abnormal vibration and temperature increase, or irregular voltage drops, are usually used as early warning signs. But these methods are reactive and may not detect subtle or emerging faults. Moreover, they often need expert and engineers' analysis. Human interpretation may not generalize properly across turbine types or operating conditions [1], [2]. As summarized in Table I, ML fault detection models outperform traditional approaches in terms of accuracy, scalability, and early detection capabilities [**?**].

To overcome these constraints, the industry has focused on predictive maintenance frameworks powered by machine learning and statistical modeling. These models utilize historical SCADA sensor data and fault logs to learn sophisticated patterns indicative of incipient failures. Notably, random forests, support vector machines, and neural networks such as multi-layer perceptrons (MLPs) have proved robust performance in fault classification and anomaly detection assignments [3], [4].

Further approaches have employed signal processing techniques on high-frequency data, like vibration or current signals, and model-based diagnostics using turbine dynamics [5]. However, these studies often need high-bandwidth sensors and turbine-specific calibration, which may limit their scalability. SCADA-based machine learning models stand to be very cost-effective. Also, they provide clear and continuous readings injected into models to learn complex patterns.

III. Problem Formulation

A. High Cost of Wind Turbines Downtime

Although wind turbines form a crucial and effective part of today's renewable energy systems (RES), several electrical and mechanical complications challenge their reliability and operational effectiveness. These setbacks lead to energy

This project is conducted as part of EECE 690 – Introduction to Machine Learning at the American University of Beirut.

TABLE I
COMPARISON OF TRADITIONAL AND ML-BASED FAULT DETECTION METHODS

Metric	Traditional Method	ML-Based Method
Detection Delay	High	Low
False Alarms	High	Low
Implementation Cost	Low	Moderate
Scalability	Low	High
Accuracy	Moderate	High

production losses, international standard violations, and high maintenance costs. In numbers, the logistics and man labor costs of maintaining a single unanticipated turbine failure might vary from $20,000 to $50,000. Every failure, when considering power outages, can result in energy losses of over $100,000, particularly during the busiest wind seasons. Unexpected problem downtime can cost large farms with more than 50 turbines between $1 and $3 million a year. Traditional scheduled and rule-based maintenance techniques are reactive and often miss early-stage or build-up abnormalities that happen before severe failures. This paper formulates the problem as a time-series early fault detection mechanism with the aid of high-resolution Supervisory Control and Data Acquisition (SCADA) data. The data is available and regularly records for the internal and external operational conditions of wind turbines. Through the detection of odd and abnormal patterns in the diversified sensor readings, a model could be trained and generated to be a detector of early failures.

B. Aim of the Project

The aim of this project is to model a smart machine learning system for early fault detection in four wind turbines with the assistance of two-years worth of SCADA sensor data. Utilizing both artificially developed and real-world fault scenarios, the system is intended to look for unusual trends that classify as potential mechanical or electrical malfunctions, reduce maintenance expenses and losses, enable wind farm laborers to take preemptive measures, and enhance the general reliability and efficiency of such clean energy production.

C. Project Definition and Requirements

This project includes the generation of a data-driven fault prediction model for wind turbines, with the goal of going from reactive to predictive maintenance mechanisms. The system is designed to interpret SCADA sensor data in real-time, extract abnormal operating conditions, and flag early-stage anomalies that may be coupled with future mechanical or electrical malfunctions.

The modeling process involves proper Exploratory Data Analysis (EDA), sequence learning, and synthetic data augmentation to ensure model effectiveness across several turbine conditions. It is tested and validated using synthetic faults.

Data is labeled per fault type and between normal and abnormal operations. This is essential for a supervised learning classification problem.

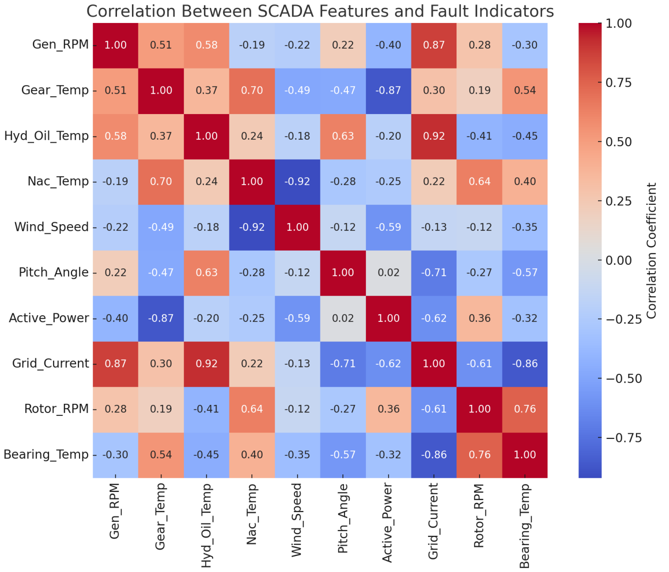

Fig. 1. Correlation Between SCADA Features and Fault Indicators

D. SCADA Dataset Overview

The dataset adopted in this project consists of SCADA sensor data across four anonymous wind turbines operating over two years (2016 & 2017). A heatmap that shows correlation between major features is presented in Figure 1. The collection involves event logs with fault annotations as well as healthy operational data that was gathered at a 10-minute interval for several turbine components. While the problem data was taken from a historical logbook that contained labeled failure events grouped by remarks such as gearbox overheating, transformer malfunction, and generator instability, the healthy sensor dataset was taken from the readings.

Continuous readings were of both electrical and mechanical conditions. The features are as follows:

TABLE II
CATEGORIZED SCADA FEATURES

Category	Key Features (Abbreviated)
Electrical	Gen_RPM_Avg, Gen_Phase1_Temp_Avg, Gen_SlipRing_Temp_Avg, Prod_LatestAvg_TotActPwr, Grd_Prod_Pwr_Avg, Grd_Prod_CurPhse1_Avg, Grd_Prod_ReactPwr_Avg, Grd_Prod_CosPhi_Avg, HVTrafo_Phase1_Temp_Avg, Generator_Phase_Max_Temp, ...
Mechanical	Gear_Oil_Temp_Avg, Gear_Bear_Temp_Avg, Hyd_Oil_Temp_Avg, Gen_Bear_Temp_Avg, Gen_Bear2_Temp_Avg, Rtr_RPM_Avg, Blds_PitchAngle_Avg, Cont_VCP_Temp_Avg, Spin_Temp_Avg, Grd_InverterPhase1_Temp_Avg, ...
Environmental	Amb_WindSpeed_Avg, Amb_WindSpeed_Std, Amb_WindDir_Relative_Avg, Amb_Temp_Avg, Nac_Temp_Avg, Nac_Direction_Avg, Amb_WindSpeed_Est_Avg, ...

The features shown in Table II are a part of turbine SCADA readings and represent operational parameters and component readings across electrical, mechanical, and environmental aspects as shown in Figure 13. Electrical features include generator behavior, power output, voltage, current, and transformer conditions. Mechanical features monitor components such as the gearbox, bearings, hydraulic systems, and blade pitch. Environmental features provide essential information on turbine performance under varying wind and temperature conditions.

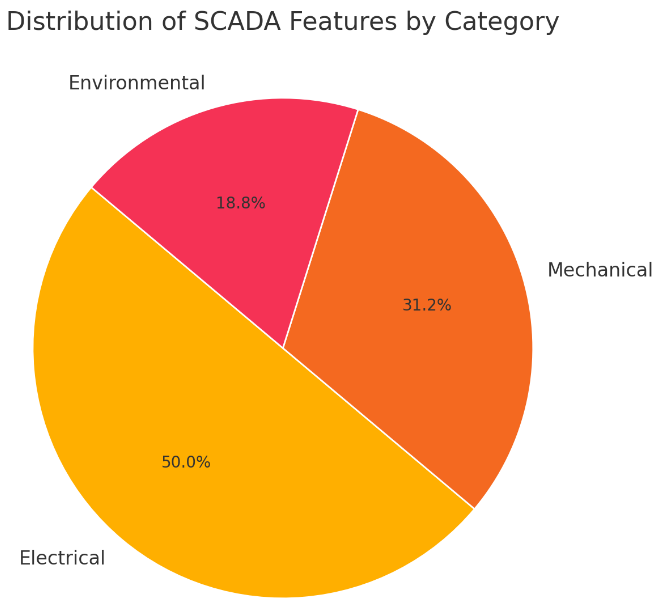

Fig. 2. Distribution of SCADA Features by Category

Alongside the SCADA sensor data, historical fault logs of each designated wind turbine are provided. The failure log's ”Remarks” column was utilized to employ labels to particular defect categories to facilitate supervised categorization. Following a mean-based technique, the sensor recordings and fault logs were normalized. A synthetic fault injection approach was used to alleviate the class imbalance present in actual turbine failures. To do this, patterns taken from actual defects, such as Gaussian noise and overlapping fault signatures, were used to disturb good sensor data. The condensed SCADA dataset guarantees a successful and high efficiency modeling ahead.

IV. Theoretical Overview

A. Random Forest Classifier

An ensemble learning approach named Random Forest (RF) introduces several decision trees during training and produces a class that is the average of the classes produced by each tree alone. A bootstrap sample of the dataset is used to train each tree, and a random subset of characteristics is taken into consideration for splitting at each node to encourage model variety and reduce the chances of overfitting.

Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^d$ represents the feature vector and y_i the class label, each tree $T^{(k)}$ provides a prediction $\hat{y}_i^{(k)}$. The final prediction $\hat{y}_i$ is determined by majority:

$$\hat{y}_i = \text{mode}\left(\hat{y}_i^{(1)}, \hat{y}_i^{(2)}, \ldots, \hat{y}_i^{(K)}\right)$$

To further understand this approach, several advantages are listed as follows:

- RF models are robust to outlier data points and noise.
- RF models are efficient in handling numerical and categorical features.
- RF models can interpret feature importance and weight.
- RF models, when employed with an optimized number of trees, can reduce the chances of overfitting.

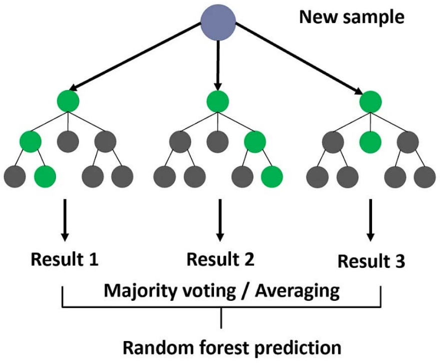

Fig. 3. Schematic of a Random Forest classifier where individual decision trees output predictions for a new sample, and the final classification is determined through majority voting [8].

The efficiency of RF in wind turbine defect detection has been shown in recent papers. For instance, Knes and Dao [6] used RF to SCADA data for early failure prediction, and they were able to detect gearbox anomalies with high accuracy. Similarly, Weber and Preisach [7] demonstrated the resilience of RF in cross-domain defect diagnostic scenarios by employing it inside a supervised transfer learning framework.

B. Multi-Layer Perceptron (MLP)

A Multi-Layer Perceptron (MLP) is a branch of feedforward artificial neural network (ANN) that consists of three layers: an input layer, one or more hidden layers, and an output layer. The input layer includes the dataset features. The hidden layers are the computations done between features. And finally, the output layer is the desired output or prediction. Within this model, every neuron calculates a weighted sum of the injected inputs. Furthermore, it uses a nonlinear activation function.

The output of the l-th layer is given by:

$$\mathbf{h}^{(l)} = \sigma\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right)$$

where σ denotes the activation function (e.g., ReLU), $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the weight matrix and bias vector of layer l, and $\mathbf{h}^{(0)} = \mathbf{x}$ is the input vector.

Training involves minimizing a loss function $\mathcal{L}(y, \hat{y})$ over the dataset using optimization algorithms like stochastic gradient descent:

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(y, \hat{y})$$

where θ represents all trainable parameters and η is the learning rate.

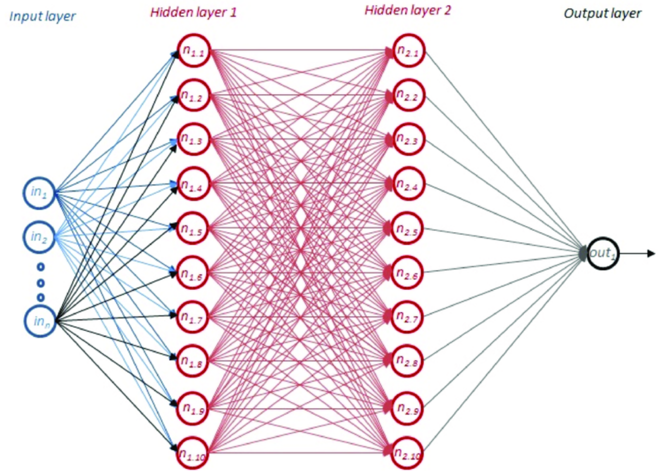

Fig. 4. Architecture of a multi-layer perceptron (MLP) with two hidden layers, showing full connectivity between layers. Each neuron applies a weighted sum followed by a non-linear activation function [9].

MLP stands as a solid model when dealing with complex and nonlinear correlations and functions. MLPs were shown to be quite robust when compared to other machine learning models for fault detection by Knes and Dao [6]. Furthermore, in their supervised transfer learning framework for diagnosing bearing and sensor problems, Weber and Preisach [7] found that MLPs had the best classification model.

C. Long Short-Term Memory (LSTM)

Recurrent Neural Networks (RNNs) are an area of AI. They learn from and recognize patterns in datasets. It introduces loops that permit data to persist. This means that the output depends on the input and hidden states, and layers. However, RNNs tend to have vanishing gradients and to forget data over time. On the other hand, LSTMs, a part of RNNS, are designed to model long-term temporal reliability in time series data. They utilize memory cells to preserve and save the information over time. LSTMs host gating algorithms that help save the information over a period of time [11].

Within an LSTM cell, there exist three gates: the forget gate (f_t), the input gate (i_t), and the output gate (o_t). The function behind these gates is to manage the flow of data and permit the model to recall or discard temporary features selectively [12]. A standard LSTM update mechanism is given by:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{3}$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{4}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \cdot \tanh(C_t) \tag{6}$$

Here, x_t is the input at time t, h_{t-1} is the previous hidden state, C_t is the cell state, and σ denotes the sigmoid activation function.

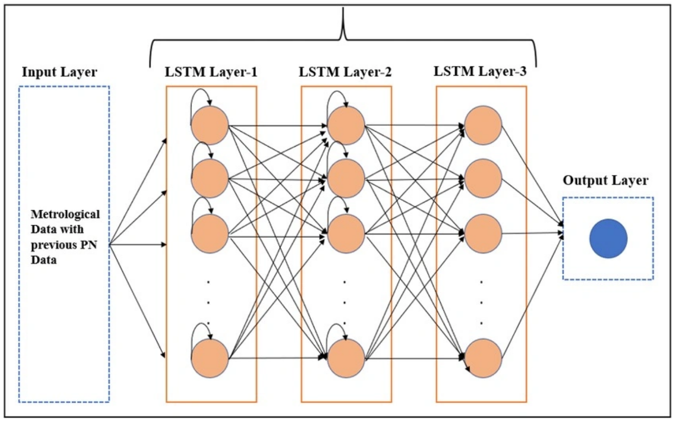

Fig. 5. LSTM Structure

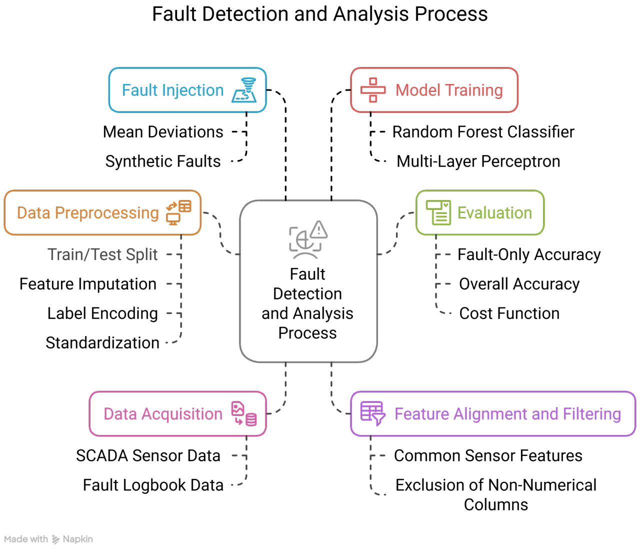

Fig. 6. Modeling and Pipeline

V. Modeling

This section involves several model algorithms and pipelines for wind turbine fault and anomaly detection using SCADA sensor data and a historical failure logbook. The two approaches are different in terms of their sophisticated modeling, fault simulation, and data augmentation. However, all aim for robust identification and classification of abnormality in operational conditions.

A. Supervised Learning

Supervised machine learning is an important approach that involves labeled data connecting input features and target outputs. Highlighting supervised learning in this project is adopted due to the presence of several fault types or classes occurring in wind turbines. Each fault triggers a set of features corresponding to its type. Training on such data assists the model in learning specific sensor data patterns related to a specific fault type, such as generator overheating, gearbox anomalies, or hydraulic oil temperature spikes. The categorical faults are transformed into encoded labels that are injected into several classifier methods, like Random Forest. Methods are evaluated in terms of accuracy based on the data type.

This section introduces a supervised learning pipeline adopted in this project for detecting and classifying faults in wind turbines. This pipeline includes data cleaning and processing, dimensionality reduction, anomaly detection, and fault classification to support the model.

1) **Data Labeling and Processing**
After implementing SCADA sensor data, several data cleaning techniques were applied to ensure a suitable format for processing. Furthermore, the dataset is then labeled as 0 and 1 for normal and abnormal operations, respectively. Also, to maintain precise alignment along the data, timestamps are standardized.
2) **Feature Engineering Using PCA**
It is important to note that faults occur due to several factors and features that can be highly correlated. While this helps the model understand the root of the fault, it can be computationally heavy and lead to overfitting. Figure 2 shows how features are correlated. PCA is applied to positively and negatively correlated features. Essentially, normalization using StandardScaler ensures that features have zero mean and a unit standard deviation. This assists with revealing underlying patterns and not just raw weights. PCA reduces the dimensionality while preserving 95% of the data's variance. A sharp rise followed by flattening indicates that few components efficiently capture most of the dataset's variance. This makes the dimensionality reduction efficient for the model. The reduced components are now fed to the model as inputs.
Ensuring a high-efficiency model, the dataset is split into an 80% training and 20% testing sets. The model is trained on a Random Forest Classifier to differentiate between normal and abnormal conditions. Random Forest similarity-based weights to features based on their contribution to splits and predictions. The model's performance is put to the test and evaluated through the model's accuracy and precision, recall, and F1-score.
3) **Sensor Level Detection**
A fault dictionary is fed to the model to differentiate between fault types and their respective affected features. The classification process introduces an exact or best-fit matching. An exact match matches the abnormal behaving sensors with their respective fault type. The best fit match adopts a similarity based method if no match is to be found. Figure 7 shows mismatch proportions and percentages between types. Training samples with a normal label are chosen to compute the mean and standard deviation of the sensors. To determine how much each sensor deviates from its typical performance, z-scores are calculated for each test sample that is projected to have an issue. Sensors are labeled as abnormal if their z-score is more than a predetermined threshold (such as 3.0).

B. Random Forest Model

Due to the advantages mentioned earlier in this paper, an RF classifier is employed to train the model. The model was implemented using ”scikit-learn”'s ”RandomForestClassifier”, with grid search tuning and custom evaluation metrics. It is important to note that two pipelines were considered. Figure 8 displays feature importance in RF modeling.

Pipeline A – Real Fault Pattern Injection:

- **Data Preparation:** Both normal and abnormal operation readings were uploaded in a form of a CSV file. The common columns were extracted, and faulty samples were labeled using the ”Remarks” field.
- **Studying Patterns:** Faults differ in categories and types. Hence, the mean deviation is computed from normal operation and selected only sensors with $|\Delta| \geq 10\%$ deviation.
- **Fault Injection:** Patterns were applied in a random order to rows from the normal dataset (20–25% of rows). Gaussian noise is applied to simulate real-world variability.
- **Train-Test Split:** Performed on the normal dataset before injection to prevent data leakage.
- **Preprocessing:** Label encoding of the ”Remarks” section, and normalization of numerical features using ”StandardScaler”.
- **Training and Selection:** RF models were trained using grid search on ”n_estimators” and ”max_depth”. The cost metric was considered to get an optimized trade-off between the accuracy of predicting faults and model accuracy:

$$\text{Cost} = \alpha(1 - \text{Fault Accuracy}) + \epsilon(\text{Model Size})$$

TABLE III
RANDOM FOREST PERFORMANCE ACROSS PIPELINES

Metric	Pipeline A	Pipeline B
Best Fault Accuracy (%)	91.4	88.2
Model Size (MB)	2.64	2.88
Training Time (s)	1.95	2.42
Inference Time (s)	0.007	0.009

Pipeline B – Complex Fault Simulation: This extended pipeline introduces:

- **Overlapping Faults:** Due to the presence of several features that share strong relationships, which can contribute to a single or several fault types, overlapping was done. This simulates combinations of fault types using averaged deviation patterns.
- **Progressive Drift:** Introduces a scaling factor to fault severity. This mimics the reality of wind turbines and their nature, which degrade over time.

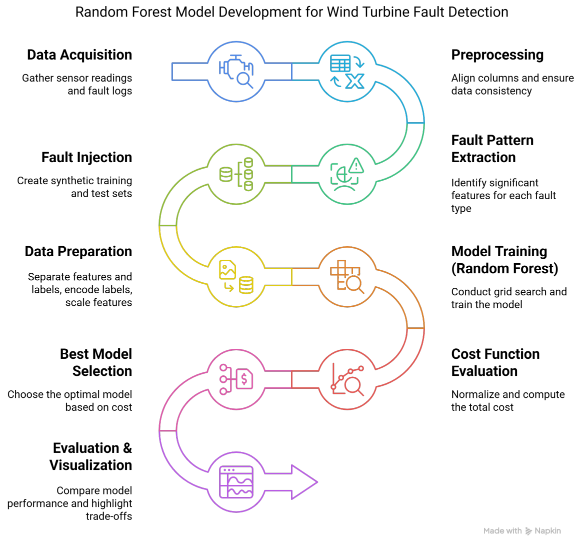
Random Forest Model Development for Wind Turbine Fault Detection
Data Acquisition
Gather sensor readings and fault logs
Preprocessing
Align columns and ensure data consistency
Fault Injection
Create synthetic training and test sets
Fault Pattern Extraction
Identify significant features for each fault type
Data Preparation
Separate features and labels, encode labels, scale features
Model Training (Random Forest)
Conduct grid search and train the model
Best Model Selection
Choose the optimal model based on cost
Cost Function Evaluation
Normalize and compute the total cost
Evaluation & Visualization
Compare model performance and highlight trade-offs
Made with Napkin

Fig. 7. Random Forest Pipeline

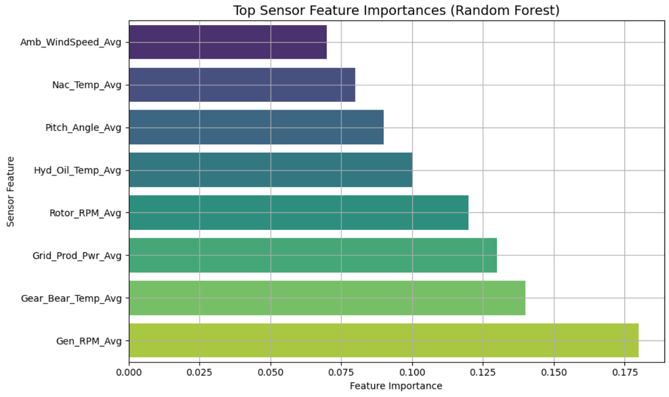

Fig. 8. Feature Importance: Random Forest

- **Data Processing:** Similar preprocessing is applied (encoding, scaling), followed by the same grid search over RF hyperparameters.

C. MLP Model

This section explains the training and evaluation methodology for the MLP model, implemented using ”scikit-learn”'s ”MLPClassifier”. The model was tested under both fault injection pipelines.

Pipeline A – Real Fault Injection:

- **Data Preparation:** Sensor data from normal and abnormal operations were aligned and cleaned. Columns were transformed to ”float64” types.
- **Pattern Extraction and Injection:** Each fault type was associated with specific sensor variations. Faults were injected into randomly selected rows of the healthy dataset with added Gaussian noise to simulate stochastic variability.
- **Preprocessing:** Features were scaled and standardized using ”StandardScaler”, and fault types ”Remarks” were encoded using ”LabelEncoder”.
- **Training:** Grid search was conducted on:
  - `hidden_layer_sizes` $\in$ {(64,), (64,32), (128,64,32)}
  - `learning_rate_init` $\in \{0.001, 0.01\}$
- **Evaluation:** The cost metric was considered to get an optimized trade-off between training time, accuracy of predicting faults, and model accuracy:

$$\text{Cost} = \alpha(1 - \text{Fault Accuracy}) + \epsilon(\text{Model Size})$$

Pipeline B – Advanced Fault Simulation: This more realistic pipeline incorporates:

- **Overlapping Faults:** Due to the presence of several features that share strong relationships, which can contribute to a single or several fault types, overlapping was done. This simulates combinations of fault types using averaged deviation patterns.
- **Progressive Fault Drift:** Applied a multiplicative factor to simulate time-based complexity of faults, which is represented through noise.
- **Training and Selection:** A similar grid search was applied, using fault-only accuracy and size as the main trade-off criteria.

TABLE IV
MLP Performance Across Pipelines

Metric	Pipeline A	Pipeline B
Best Fault Accuracy (%)	92.7	89.6
Model Size (MB)	3.80	4.56
Training Time (s)	4.73	5.82
Inference Time (s)	0.021	0.027

D. Long Short-Term Memory

LSTM stood as the best model of all models in this project. The reason behind utilizing this model is that it works well with time-series data. Data from Turbine-01 was used for training, while Turbine-06 was put to the test. This prevents data leakage and overfitting. Missing values were handled using ”SimpleImputer” with the mean. Data was also standardized. To create input sequences, sliding window (size=10) was utilized on the time-series data. LSTM inputs were in the shape of a 3D tensor. The optimizer used was ”Adam” and a Softmax activation function. Additionally, the model was trained with 50 epochs and a batch size of 32. However, early stopping was necessarily done with patience=3 to prevent overfitting. Model architecture is displayed through the figures below.

VI. Model Comparison

Figures 9,10, and 11 show all three models' performance and some important parameters. LSTM proved to be superior in performance in terms of higher accuracy than other models. Also, due to the LSTM's ability to learn from patterns and behaviors that are temporary, it showed robust performance

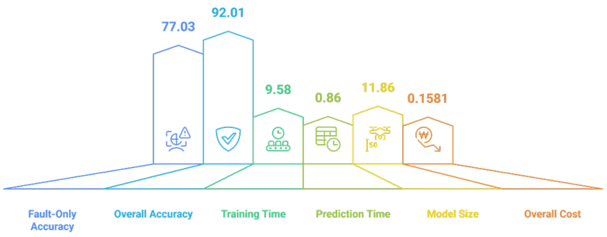

Fig. 9. Random Forest Performance and Parameters

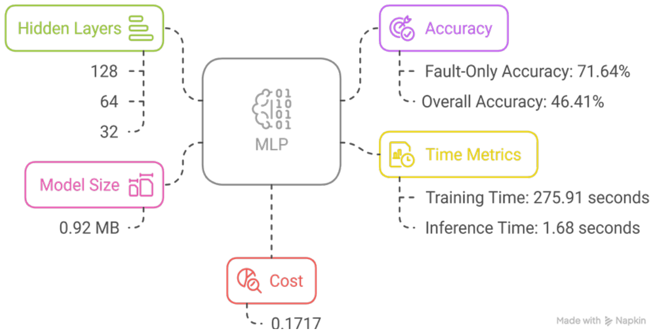

Fig. 10. MLP Performance and Parameters

with wind turbines, knowing that they may have unpredictable and temporary faults. But it takes a long time to train and is complex to deploy on edge devices. Although RF trained fast, it lacked temporal modeling and was not robust to overlapping faults. As for MLP, it was also lightweight but showed very low accuracy.

VII. Fault Prediction Dashboard- An Interactive Web

For visualization and implementation, an interactive web has been developed using ”Streamlit” to predict and classify faults, estimate energy losses, and costs. The models were employed on the web with the respective dataset. Visualizations were included to aid with understanding and clarity. Also, a realistic cost map is included to asses real damage. This also helps with proper decision-making to minimize risks and losses as much as possible. These models perform efficiently when coupled with Building Management Systems (BMS).

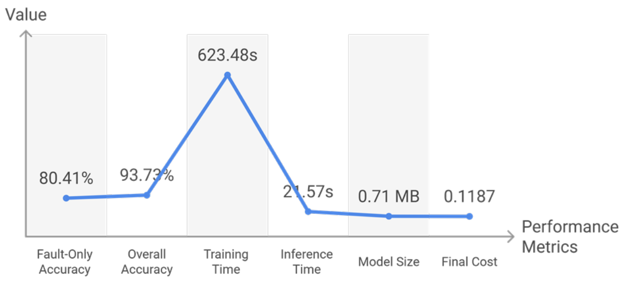

Fig. 11. LSTM Performance and Parameters

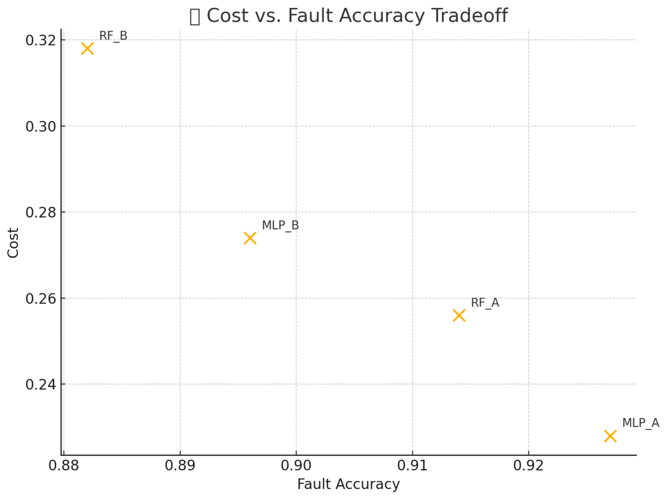

Fig. 12. Cost Vs. Fault Accuracy Tradeoff

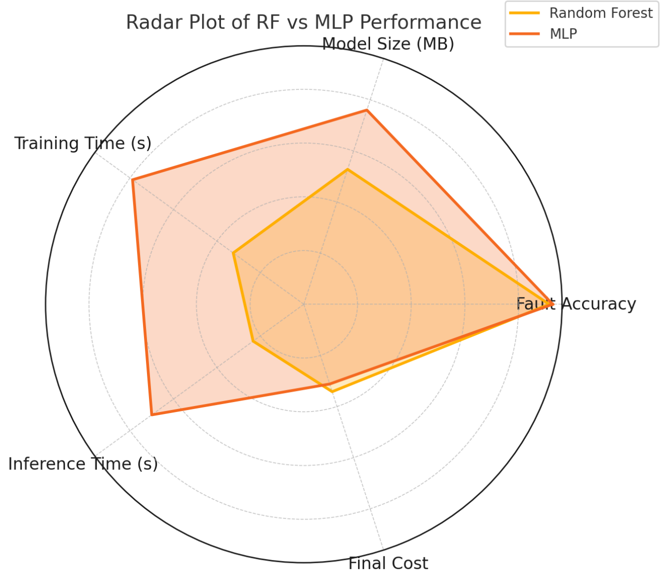

Fig. 13. Radar Plot of RF Vs MLP Performance

VIII. Future Work

While the LSTM model stands as a solid model, the team is willing to explore enhancements and improvements that will elevate such a project. Essentially, sequence-to-sequence models may be considered. Moreover, fault-explainability tools might be included to fully elaborate the reason and features behind the fault. A more advanced pipeline includes a model that predicts faults on raw data without historical logged faults. This adapts to the evolvement and enhancements done in wind turbines.

Abbreviations and Acronyms

- **ANN** – Artificial Neural Network
- **API** – Application Programming Interface
- **CSV** – Comma-Separated Values
- **EDA** – Exploratory Data Analysis
- **FPR** – False Positive Rate
- **ML** – Machine Learning
- **MLP** – Multi-Layer Perceptron
- **PCA** – Principal Component Analysis
- **RES** – Renewable Energy Systems
- **RF** – Random Forest
- **ROC** – Receiver Operating Characteristic
- **SCADA** – Supervisory Control and Data Acquisition
- **SVM** – Support Vector Machine
- **TPR** – True Positive Rate

References

[1] Y. Qin, H. Tian, and M. Zuo, “A review on wind turbine condition monitoring and fault diagnosis—Part I: Components and subsystems,” *IEEE Trans. Ind. Electron.*, vol. 66, no. 7, pp. 5152–5162, Jul. 2019.

[2] P. F. Odgaard, K. Johnson, and M. Blanke, “Wind turbine fault detection and fault-tolerant control—A review,” *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1378–1392, Jul. 2013.

[3] K. Zhou, X. Yang, C. Qiu, and C. Ma, “A deep learning method for wind turbine gearbox fault diagnosis based on SCADA data,” *Energies*, vol. 13, no. 4, p. 1052, Feb. 2020.

[4] L. Zhang, B. Chen, and J. Li, “Data-driven fault diagnosis for wind turbines using random forest and SCADA data,” *Renew. Energy*, vol. 163, pp. 1989–2002, Feb. 2021.

[5] Z. Hameed, Y. H. Hong, and Y. M. Cho, “Condition monitoring and fault detection of wind turbines and related algorithms: A review,” *Renew. Sustain. Energy Rev.*, vol. 13, no. 1, pp. 1–39, Jan. 2009.

[6] P. Knes and P. B. Dao, “Machine Learning and Cointegration for Wind Turbine Monitoring and Fault Detection: From a Comparative Study to a Combined Approach,” *Energies*, vol. 17, no. 20, p. 5055, Oct. 2024.

[7] K. Weber and C. Preisach, “Supervised Transfer Learning Framework for Fault Diagnosis in Wind Turbines,” *arXiv preprint arXiv:2411.02127*, Nov. 2024.

[8] M. Y. Khan, A. Qayoom, M. S. Nizami, M. S. Siddiqui, S. Wasi, S. M. K. Raazi, and S. Sarfraz, “Automated prediction of good dictionary examples (GDEX): A comprehensive experiment with distant supervision, machine learning, and word embedding-based deep learning techniques,” *Complexity*, vol. 2021, Article ID 2553199, pp. 1–18, Sep. 2021, doi: 10.1155/2021/2553199.

[9] J. Foroozesh, A. Khosravani, A. Mohsenzadeh, and A. H. Mesbahi, “Application of artificial intelligence (AI) modeling in kinetics of methane hydrate growth,” *American Journal of Analytical Chemistry*, vol. 4, no. 11, pp. 616–622, Nov. 2013, doi: 10.4236/ajac.2013.411073.

[10] N. Fatima, et al., “Comparative Study of Deep Learning Models Versus Machine Learning Models for Wind Turbine Intelligent Health Diagnosis Systems,” *Sensors*, vol. 24, no. 1, pp. 123, 2024.

[11] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A critical review of recurrent neural networks for sequence learning,” *arXiv preprint arXiv:1506.00019*, 2015.

[13] Y. Li, T. Zhang, and J. Gao, “SCADA-based Wind Turbine Fault Diagnosis using LSTM with Attention Mechanisms,” *Renewable Energy*, vol. 212, pp. 1055–1067, 2023.