Data Injection Attacks on Predictive Maintenance Systems

Jayanth Guduru
jg7162
jg7162@nyu.edu

Prathyusha kadali
pk2669
pk2669@nyu.com

Nelanuthula sai goutham
sn3533
sn3533@nyu.com

Abstract

Predictive Maintenance (PdM) systems, leveraging advanced Machine Learning technologies, play a crucial role in preempting equipment failures, thereby minimizing downtime and extending equipment lifespan. Our project delves into the vulnerability of PdM systems when subjected to False Data Injection (FDI) attacks, a critical concern in the era of Industry 4.0. We concentrate our efforts on assessing the robustness of Deep Learning models, specifically Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN), in their ability to predict the Remaining Useful Life (RUL) of machinery when faced with such challenging conditions. The project encompasses the modeling and application of two distinct types of data injection attacks : Biased Data Injection Attack and Random Data Injection Attack, targeting turbofan engine sensor data. Our evaluation critically examines the impact of these attacks on the predictive accuracy of CNN, LSTM, and GRU models within PdM systems. The findings reveal that even a few compromised IoT sensors through FDI attacks can significantly impair the RUL prediction across all models, highlighting a critical area of concern in the reliability and security of PdM systems.

https://github.com/jayanthguduru/DataInjectionAttacks

1 Introduction

The integration of Machine Learning (ML) techniques with Internet-of-Things (IoT) sensors marks a transformative era, prominently illustrated in the development of Predictive Maintenance (PdM) systems. This innovative approach employs advanced analytics on sensor data to identify patterns, thereby preempting potential asset issues before their occurrence. Cutting-edge Predictive Maintenance (PdM) systems have demonstrated significant advantages, notably in reducing downtime by 35%-45%, lowering maintenance expenses by 20%-25%, and enhancing production efficiency by 20%-25%. As Predictive Maintenance solutions, powered by Machine Learning (ML) and the Internet of Things (IoT), transform industries such as automotive, aerospace, oil and gas, transportation, manufacturing, and national defense, their effectiveness becomes essential for the uninterrupted functioning of critical assets. This research paper extends the current body of knowledge to investigate the consequences of False Data Injection Attacks (FDIAs) on predictive analytics powered by deep learning. It specifically focuses on the application of these attacks within the realm of Predictive Maintenance (PdM) systems for monitoring the health of aircraft engines. The susceptibility of IoT sensors and deep learning algorithms to cyber threats presents a significant risk to the reliability of PdM systems. With a reported surge of over 200% in cyber threats targeting businesses and industrial settings, the

urgency to address sophisticated and stealthy attacks like FDIAs is paramount. FDIAs, characterized by the manipulation of sensor measurements, can severely compromise the integrity of PdM systems. Such attacks might lead to delayed maintenance interventions and potentially catastrophic failures, especially in safety-critical applications. Drawing inspiration from events like the 2003 Northeast blackout and the Ukrainian power grid attack, our study ventures into the largely uncharted territory of the effects of False Data Injection Attacks (FDIAs) on Predictive Maintenance (PdM) systems, with a particular emphasis on monitoring the health of aircraft engines. Our contribution in this research lies in modeling both continuous and intermittent FDIAs on IoT sensors and evaluating their effects on PdM models. Utilizing the C-MAPSS dataset, we extensively train and evaluate three deep learning algorithms: LSTM, GRU, and CNN. The results not only validate the GRU model's accuracy in predicting RUL but also expose the vulnerabilities of PdM systems to FDIAs. This research, a pioneering effort in exploring IoT sensor attacks within a deep learning-enabled PdM framework, provides crucial insights for safeguarding critical assets against evolving cyber threats.

2 Related Work

The Advancement of Predictive Maintenance In the realm of industrial maintenance, a significant evolution has been the shift from traditional reactive maintenance to Predictive Maintenance (PdM). This paradigm shift, leveraging the capabilities of advanced Machine Learning (ML) and Internet of Things (IoT) technologies, has transformed how equipment maintenance is approached. Predictive Maintenance, by analyzing data to predict failures before they occur, offers substantial benefits including reduced downtime and maintenance costs, and increased equipment lifespan. The integration of ML and IoT in PdM systems has proven particularly effective in industries such as automotive, aerospace, and manufacturing, where equipment reliability is crucial.

Machine Learning Models in PdM Systems The application of various ML models, including Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN), has been a focal point in recent PdM advancements. These models have been employed for their ability to process complex sensor data and predict equipment failures with high accuracy. Research in this area has predominantly focused on the effectiveness of these models in different industrial scenarios, highlighting their strengths in handling time-series data and multi-dimensional sensor readings.

Cybersecurity Challenges in PdMs With the increasing reliance on IoT and connected systems in PdM, cybersecurity has emerged as a critical concern. The vulnerability of these systems to cyber-attacks, such as False Data Injection Attacks (FDIAs), poses a significant risk. These attacks, which involve manipulating sensor data, can lead to incorrect maintenance decisions and potential equipment failures. The literature reveals a growing interest in understanding these threats and developing strategies to mitigate them. However, there remains a gap in research specifically targeting the resilience of ML models in PdM systems against such sophisticated cyber threats.

Our project situates itself within this context, addressing the underexplored area of the impact of FDIAs on ML models in PdM systems. By focusing on deep learning models and their vulnerability to FDIAs, especially in critical applications like aircraft engine health monitoring, our research contributes valuable insights to the field. We explore not only the accuracy of these models in predicting Remaining Useful Life (RUL) but also their robustness against cyber threats, an aspect often overlooked in existing studies. In summary, while the integration of ML in PdM has been extensively researched, the aspect of cybersecurity, particularly in the face of sophisticated FDIAs, requires further exploration. Our study aims to fill this gap, providing crucial insights into securing PdM systems in an increasingly interconnected industrial landscape.

3 Data Description

The NASA C-MAPSS (Commercial Modular Aero-Propulsion System Simulation) dataset is a key resource in the field of prognostics and health management. It's primarily used for modeling the degradation of assets, specifically turbofan jet engines. Originating from simulations performed using the C-MAPSS framework, this dataset is provided by NASA's Prognostics Center of Excellence at

Ames. It's instrumental in developing predictive models to anticipate engine failures and manage maintenance schedules effectively.

3.1 Objective of the Dataset

The dataset's main purpose is to facilitate the prediction of the Remaining Useful Life (RUL) of jet engines. RUL represents the estimated operational cycles remaining before an engine is likely to fail, starting from the last recorded data point in the test set. The dataset comprises several multivariate time series, each corresponding to a different engine. These engines form part of a larger fleet and exhibit individual variations in wear and manufacturing, considered normal and not indicative of faults.

3.2 Structure and content

Normal operation at the start of each time series, with a fault developing over time. The training set data extends up to the point of engine failure. The test set concludes before engine failure, with the challenge being to predict the number of remaining operational cycles.

The dataset is divided into four subsets (FD001 to FD004), each with unique operational conditions and fault modes:

- FD001: 100 training and 100 test trajectories, single operational condition, one fault mode.
- FD002: 260 training and 259 test trajectories, multiple operational conditions, one fault mode.
- FD003: 100 training and 100 test trajectories, single operational condition, two fault modes.
- FD004: 248 training and 249 test trajectories, multiple operational conditions, two fault modes.

Each data entry includes operational settings, cycle information, and measurements from 26 different sensors. The columns in the dataset represent Engine unit number, Operational cycle, Three operational settings and Twenty-three sensor measurements. Characterized by a whitespace-separated format, the dataset is rich in operational and sensor-based metrics. The absence of header information necessitated the explicit definition of column names, ensuring clarity for further analysis. These columns, including "id", "cycle", "op1", "op2", and "sensor1" through "sensor21", provide a multidimensional view of engine performance, mirroring real-world scenarios. This data structure is particularly suited for training machine learning models to detect sophisticated cybersecurity threat

3.3 Exploratory Data Analysis

The exploratory data analysis of the NASA C-MAPSS dataset involves several steps. The initial step in our data preprocessing involved importing necessary libraries for data manipulation and visualization. The Pandas library was used to load the dataset from Google Drive, while Matplotlib and Sklearn's IterativeImputer hinted at potential data cleaning steps. A preliminary check revealed missing values across columns, prompting further data cleaning and imputation as necessary steps in our preprocessing pipeline.

3.3.1 Data Loading

The dataset is imported using Pandas, ensuring a structured format for analysis. No missing values are detected, confirming data completeness.

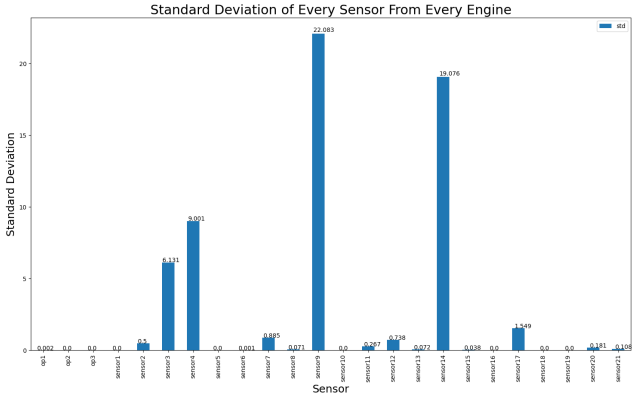
3.3.2 Feature Engineering

A new feature, Remaining Useful Life (RUL), is calculated for each engine, pivotal for predictive maintenance applications.

3.3.3 Standard Deviation Analysis

Standard deviations of sensor readings are plotted, highlighting variability and potential anomalies in the data.

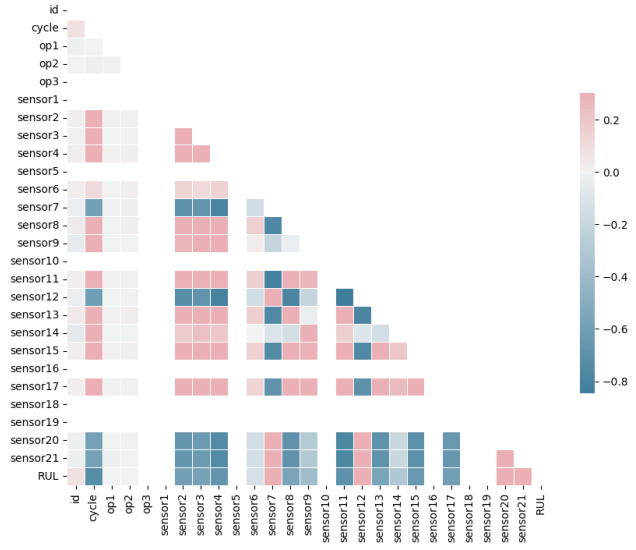
Figure 1: Standard Deviation Analysis



3.3.4 Correlation Study

A heatmap is generated to visualize the correlations between various features, aiding in understanding their interrelationships.

Figure 2: Correlation Matrix



3.3.5 Regression Analysis

An Ordinary Least Squares (OLS) regression model is applied to explore the relationship between features and RUL, with detailed results including feature coefficients and significance levels.

Figure 3: Correlation Matrix

OLS Regression Results

Dep. Variable:		RUL	R-squared:			−1.870	
Model:		OLS	Adj. R-squared:			−1.872	
Method:		Least Squares	F-statistic:			−839.5	
Date:		Fri, 15 Dec 2023	Prob (F-statistic):			1.00	
Time:		23:49:45	Log-Likelihood:			−1.2747e+05	
No. Observations:		20631	AIC:			2.550e+05	
Df Residuals:		20614	BIC:			2.551e+05	
Df Model:		16					
Covariance Type:		nonrobust					
	coef	std err	t	P>|t|	[0.025	0.975]
op1	0.1119	0.813	0.138	0.891	−1.483	1.706
op2	0.4528	0.813	0.557	0.578	−1.141	2.047
op3	2.089e−15	8.76e−16	2.384	0.017	3.71e−16	3.81e−15
sensor1	5.891e−16	4.44e−16	1.328	0.184	−2.8e−16	1.46e−15
sensor2	−3.4090	1.320	−2.582	0.010	−5.997	−0.821
sensor3	−2.7037	1.230	−2.198	0.028	−5.114	−0.293
sensor4	−6.8838	1.739	−3.958	0.000	−10.293	−3.475
sensor5	−9.377e−16	3.07e−15	−0.305	0.760	−6.96e−15	5.09e−15
sensor6	−0.7092	0.825	−0.859	0.390	−2.327	0.908
sensor7	6.0910	1.691	3.601	0.000	2.776	9.406
sensor8	−0.9395	1.764	−0.533	0.594	−4.398	2.519
sensor9	−7.7186	3.433	−2.248	0.025	−14.448	−0.989
sensor10	−4.124e−16	5.03e−16	−0.819	0.413	−1.4e−15	5.74e−16
sensor11	−9.9388	1.991	−4.992	0.000	−13.841	−6.037
sensor12	7.8355	1.865	4.201	0.000	4.180	11.491
sensor13	−0.8532	1.762	−0.484	0.628	−4.308	2.601
sensor14	−5.2231	3.376	−1.547	0.122	−11.841	1.394
sensor15	−4.4783	1.476	−3.034	0.002	−7.372	−1.585
sensor16	−2.161e−28	1.88e−28	−1.150	0.250	−5.84e−28	1.52e−28
sensor17	−2.8623	1.300	−2.201	0.028	−5.411	−0.314
sensor18	0	0	nan	nan	0	0
sensor19	0	0	nan	nan	0	0
sensor20	3.5898	1.422	2.524	0.012	0.803	6.377
sensor21	4.4471	1.433	3.103	0.002	1.638	7.256
Omnibus:		3258.323	Durbin-Watson:			0.013	
Prob(Omnibus):		0.000	Jarque-Bera (JB):			5724.588	
Skew:		1.027	Prob(JB):			0.00	
Kurtosis:		4.562	Cond. No.			2.31e+39	

3.3.6 Data Normalization

The features are standardized using scaling techniques to ensure uniformity in the data range. This comprehensive analysis lays the groundwork for predictive modeling, especially in estimating the RUL of engines in the dataset.

4 Methodology

In our methodology for analyzing the NASA C-MAPSS dataset, we leveraged advanced neural network models—specifically, Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN)—owing to their compatibility with time-series data. LSTM and GRU are particularly adept at capturing temporal dependencies and patterns in complex, sequential datasets like ours, which features intricate variability in engine sensor readings and operational settings. Meanwhile, CNNs enhance the model's performance by effectively extracting key features from the time-series data, providing a comprehensive understanding of the dataset's characteristics. This combination of models ensures a robust approach to predicting the Remaining Useful Life (RUL) of engines, a task crucial for effective predictive maintenance in aerospace applications.

LSTM Model Summary

Structure and Configuration

- **Model Type:** Sequential
- **Layers:**
  1. LSTM layer with 256 units, `return_sequences=True`, and `activation='tanh'`. Input shape is defined as `(window_length, 14)`.
  2. LSTM layer with 128 units, `return_sequences=True`, and `activation='tanh'`.
  3. LSTM layer with 64 units and `activation='tanh'`.
  4. Dense layer with 128 units and `activation='relu'`.

5. Dense layer with 64 units and `activation='relu'`.
6. Dense layer with 32 units and `activation='relu'`.
7. Dense output layer with 1 unit.

- **Total Parameters:** 542,721 (Trainable: 542,721, Non-trainable: 0)
- **Optimizer:** Adam with a learning rate of 0.001
- **Loss Function:** Mean Squared Error (MSE)

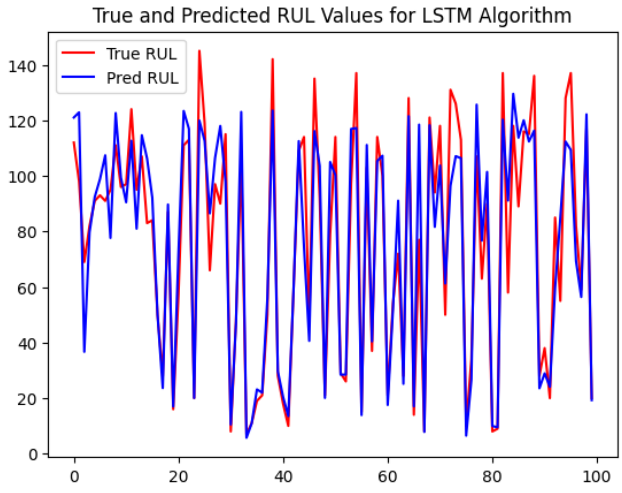
Training Details

- **Epochs:** 100
- **Training and Validation Loss:** Starts with 2943.6201 and 939.6226 respectively for the first epoch, showing a consistent decrease over the epochs. By the 100th epoch, the training loss is 106.8463 and the validation loss is 111.6098.
- **Learning Rate Scheduler:** Implemented, with the learning rate adjusted in each epoch.

Performance Metrics

- **Root Mean Square Error (RMSE):**
  - For the LSTM model: 14.3893
  - Considering only the last examples: 14.5950
- **S-score:** 345.7814

Figure 4: True vs Predicted RUL using LSTM



GRU Model Summary

Structure and Configuration

- **Model Type:** Sequential
- **Layers:**
  1. GRU layer with 256 units, `return_sequences=True`, and `activation='tanh'`. Input shape is defined as `(None, 14)`.
  2. GRU layer with 128 units, `return_sequences=True`, and `activation='tanh'`.
  3. GRU layer with 64 units and `activation='tanh'`.
  4. Dense layer with 128 units and `activation='relu'`.
  5. Dense layer with 64 units and `activation='relu'`.

6. Dense layer with 32 units and `activation='relu'`.
7. Dense output layer with 1 unit.

- **Optimizer:** Adam with a learning rate of 0.001
- **Loss Function:** Mean Squared Error (MSE)

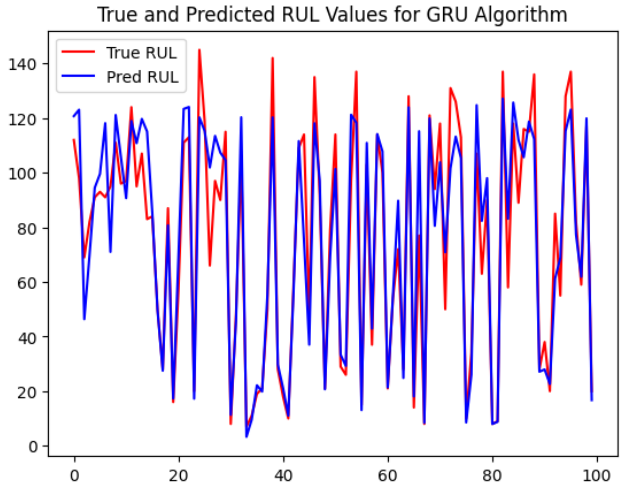
Training Details

- **Epochs:** 100
- **Learning Rate Scheduler:** Implemented, with the learning rate adjusted based on the epoch.
- **Training and Validation Loss:** Starts at 3388.3018 and 958.8034 respectively for the first epoch, with a gradual decrease across epochs. By the 100th epoch, the training loss is 100.4505 and the validation loss is 102.0741.

Performance Metrics

- **Root Mean Square Error (RMSE):**
  - For the GRU model: 13.5357
  - Considering only the last examples: 13.9304
- **S-score:** 318.7967

Figure 5: True vs Predicted RUL using GRU



CNN Model Summary

Structure and Configuration

- **Model Type:** Sequential
- **Layers:**
  1. Input layer with shape (sequence_length, nb_features).
  2. Batch Normalization layer.
  3. Four Conv1D layers each with 64 filters, kernel size 3, 'valid' padding, 'relu' activation, and L2 kernel regularization. Each Conv1D layer is followed by a Batch Normalization layer and a Dropout layer with 0.2 dropout rate.
  4. Flatten layer to convert the 3D output to 1D.
  5. Dense layer with 40 units followed by 'relu' activation.
  6. Dropout layer with 0.2 dropout rate.

7. Dense layer with 30 units followed by 'relu' activation.
8. Dense output layer with 1 unit and 'relu' activation.

- **Optimizer:** Adam with a learning rate of 0.0001
- **Loss Function:** Mean Squared Error (MSE)

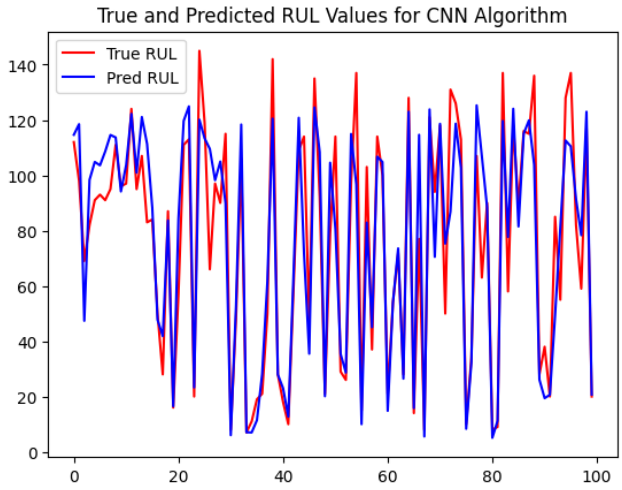
Training Details

- **Epochs:** 100
- **Training Metrics:** Loss, Mean Squared Error (MSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE).
- **Training and Validation Loss:** Starting with a high loss at the first epoch (8141.7515 for training and 7814.3311 for validation) and showing significant reduction over the epochs.

Performance Metrics

- **Root Mean Square Error (RMSE):**
  - For the CNN model: 17.1507
  - Considering only the last examples: 16.8152
- **S-score:** 520.8966

Figure 6: True vs Predicted RUL using GRU



5 Results

In our study, we executed two types of noise injection attacks on LSTM, GRU, and CNN models to assess their robustness in adversarial conditions: a biased noise attack and a random noise attack. Both attacks were aimed at evaluating the impact of noise on the models' prediction accuracy, particularly their RMSE.

Biased Noise Injection: The biased noise attack involved introducing noise at predetermined levels $[-0.06, -0.04, -0.02, -0.002, 0]$ into specific sensors ('sensor2', 'sensor8', 'sensor14') of the engine data. This approach simulated a systematic and consistent pattern of noise injection across the data.

Random Noise Injection: Conversely, the random noise attack involved randomly selecting noise bounds within the range of -0.06 to 0 for each test. The noise was then introduced into the same set of sensors. This method represented a less predictable and more varied attack pattern.

Procedure: For both attacks, we followed these steps:

1. Loaded the test data and introduced noise into the readings of the selected sensors.
2. Made predictions using the LSTM, GRU, and CNN models on the noise-injected data.
3. Calculated the RMSE between the true Remaining Useful Life (RUL) and the model predictions under each noise level.

Results: The RMSE values varied across both types of attacks, noise levels, and models. Notably, the GRU model often showed better resilience compared to LSTM and CNN models, particularly in the biased noise scenario. Below are the RMSE results for each noise level and model:

- Biased Noise Injection:
  - -0.06 Noise Level: LSTM RMSE = 40.69, GRU RMSE = 26.24, CNN RMSE = 64.90.
  - -0.04 Noise Level: LSTM RMSE = 38.57, GRU RMSE = 26.62, CNN RMSE = 40.01.
  - -0.02 Noise Level: LSTM RMSE = 38.55, GRU RMSE = 31.48, CNN RMSE = 40.03.
  - -0.002 Noise Level: LSTM RMSE = 32.76, GRU RMSE = 28.09, CNN RMSE = 50.73.
  - 0 Noise Level: LSTM RMSE = 14.39, GRU RMSE = 13.54, CNN RMSE = 17.15.
- Random Noise Injection:
  - Random Bound 1 (-0.037): LSTM RMSE = 14.79, GRU RMSE = 14.14, CNN RMSE = 17.40.
  - Random Bound 2 (-0.051): LSTM RMSE = 16.88, GRU RMSE = 13.35, CNN RMSE = 18.38.
  - Random Bound 3 (-0.024): LSTM RMSE = 15.55, GRU RMSE = 13.64, CNN RMSE = 17.14.
  - Random Bound 4 (-0.010): LSTM RMSE = 16.23, GRU RMSE = 13.57, CNN RMSE = 17.94.

Figure 7: Biased FDIA Attack impact on RMSE of DL Algorithms

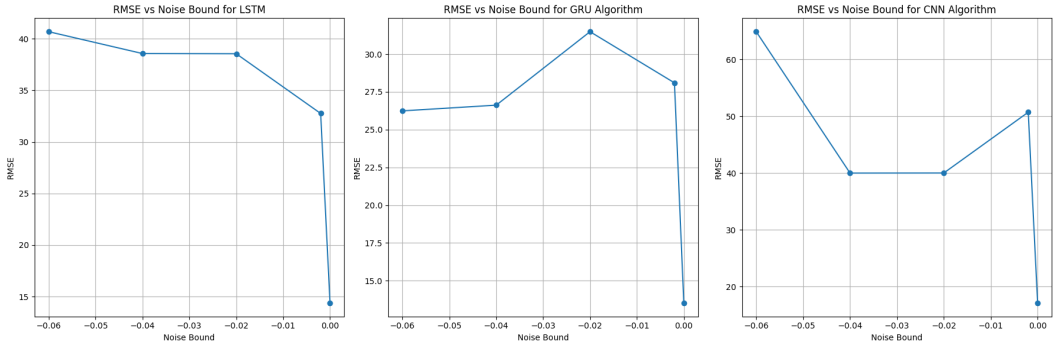
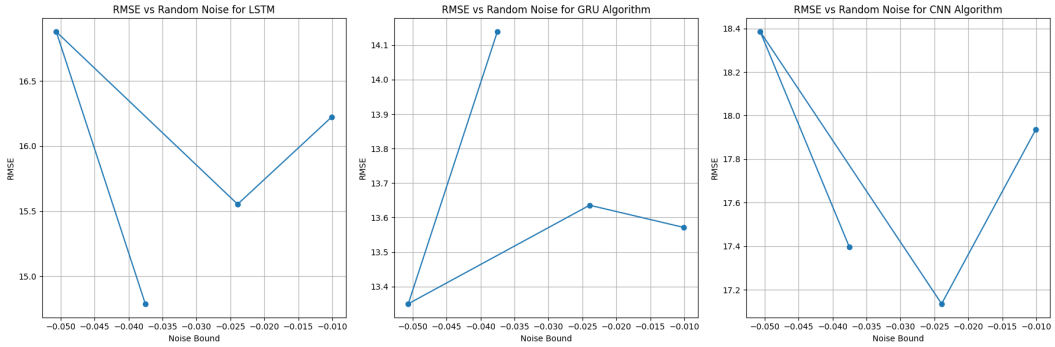


Figure 8: Random FDIA Attack impact on RMSE of DL Algorithms



Observations: The experiments highlighted the impact of noise on predictive maintenance models. The GRU model generally exhibited better performance under noise conditions, suggesting its robustness against both biased and random noise attacks. The CNN model, on the other hand, showed greater sensitivity, particularly to higher noise levels in the biased attack scenario.

Conclusion: These findings emphasize the necessity for considering different types of adversarial attacks in evaluating the robustness of predictive maintenance models and underscore the importance of noise resilience in model development. As a natural extension of our research, we propose delving into the identification and prediction of attacks on individual sensors within Predictive Maintenance (PdM) systems. This avenue holds promise for enhancing threat detection precision and fortifying the security of PdM systems, thereby advancing the understanding of cybersecurity threats in predictive maintenance.

References

[1] Gautam Raj Mode, Prasad Calyam, and Khaza Anuarul Hoque. *False Data Injection Attacks in Internet of Things and Deep Learning enabled Predictive Analytics*. Extended version, accepted for publication in the 32nd IEEE/IFIP Network Operations and Management Symposium (NOMS 2020).

[2] Ayesha Siddique, et al. *RobustPdM: Designing Robust Predictive Maintenance against Adversarial Attacks*. Published on January 25, 2023.

[3] Hajar Moudoud, et al. *Prediction and Detection of FDIA and DDoS Attacks in 5G Enabled IoT*. Published on September 21, 2020.

[4] Manoj Basnet, et al. *Ransomware Detection Using Deep Learning in the SCADA System of Electric Vehicle Charging Station*. Published on April 15, 2021.

[5] Fayha ALmutairy, et al. *Identification and Correction of False Data Injection Attacks against AC State Estimation using Deep Learning*. Published on August 4, 2020.