electronics

MDPI

*Article*

# False Data Injection Attack Detection, Isolation, and Identification in Industrial Control Systems Based on Machine Learning: Application in Load Frequency Control

**Sohrab Mokhtari and Kang K. Yen ***

Electrical and Computer Engineering Department, Florida International University, Miami, FL 33174, USA; somokhta@fiu.edu

* Correspondence: yenk@fiu.edu

**Abstract:** The integration of advanced information and communication technology in smart grids has exposed them to increased cyber attacks. Traditional model-based fault detection systems rely on mathematical models to identify malicious activities but struggle with the complexity of modern systems. This paper explores the application of artificial intelligence, specifically machine learning, to develop fault detection mechanisms that do not depend on these models. We focus on operational technology for fault detection, isolation, and identification (FDII) within smart grids, specifically examining a load frequency control (LFC) system. Our proposed approach uses sensor data to accurately identify threats, demonstrating promising results in simulated environments.

**Keywords:** fault detection; fault isolation; fault identification; operational technology; machine learning; industrial control system; intrusion detection system; smart grid; load frequency control

check for updates

**Citation:** Mokhtari , S.; Yen, K.K. False Data Injection Attack Detection, Isolation, and Identification in Industrial Control Systems Based on Machine Learning: Application in Load Frequency Control. *Electronics* **2024**, *13*, 3239. https://doi.org/10.3390/electronics13163239

Academic Editor: Andrea Bonci

Received: 27 June 2024
Revised: 2 August 2024
Accepted: 8 August 2024
Published: 15 August 2024

## 1. Introduction

Industrial control systems (ICSs) play a pivotal role in automating and operating processes across various sectors, including smart grids, transportation, and manufacturing. These systems utilize extensive control mechanisms and components such as plants, actuators, networks, and controllers. Historically, numerous studies have aimed to enhance the reliability and security of control systems by introducing advanced detection mechanisms that utilize measurable parameters from monitoring systems to identify and mitigate abnormal activities. However, recent incidents, including a USD 41M loss at a Norwegian aluminum manufacturer in 2019 and significant disruptions in a US gas pipeline in 2021, underline the ongoing vulnerabilities in such critical infrastructure [1]. ICSs often depend on remote connections that pose security vulnerabilities that are exploitable by cyber attacks. Despite the widespread use of security measures like Kalman filters and network intrusion detection systems (NIDSs), the increasing sophistication and frequency of cyber threats continue to pose significant challenges to system security.

Smart grids, as vital infrastructure, require robust protection systems to maintain a sustainable energy supply. Load Frequency Control (LFC), crucial for synchronizing frequencies across power areas, must operate flawlessly to prevent systemic failures. Traditional methods often use mathematical models to predict and mitigate discrepancies in system performance, but their effectiveness diminishes with the systems' growing complexity and non-linearity [2,3].

Conversely, learning-based methods offer a more adaptable solution by employing ML algorithms to analyze system data directly, bypassing the limitations of model-based approaches [4,5]. Sayghe et al. conducted a comprehensive survey of ML methods for detecting false data injection (FDI) attacks in power systems, highlighting the limitations of traditional residual-based bad data detection (BDD) methods and the effectiveness of data-driven solutions. They reviewed various machine learning algorithms, including supervised, semi-supervised, and unsupervised methods, which have shown superior performance in terms of accuracy and adaptability to dynamic grid environments. The study emphasized the need for advanced detection techniques to overcome the increasing complexity and sophistication of cyber attacks targeting power system state estimation processes [6]. These methods typically focus on IT, such as data associated with the IP packets in wireless connection transmission known as NIDS. Jokar et al. developed an intrusion detection and prevention system for home area networks in smart grids, employing both model-based and learning-based strategies to investigate network traffic and search for a wide range of attack types. Their effectiveness was proven by testing on IEEE 802.15.4 attacks [7,8]. Additionally, Khan et al. introduced a feature-based NIDS for smart grid systems, focusing on the optimal selection of network features to train ML algorithms. They successfully evaluated the performance of their method on two standard datasets called NSLKDD [9] and KDD99 [10]. Moreover, the accelerated adoption of the Industrial Internet-of-Things (IIoT) has introduced critical security vulnerabilities, particularly through FDI attacks, which compromise sensor measurements and evade traditional detection methods. A recent study demonstrated a method for detecting FDI attacks using autoencoders (AEs) and denoising autoencoders (DAEs), significantly outperforming support vector machine (SVM)-based approaches in both detection and data recovery [11]. Despite these advancements, IT-based approaches remain vulnerable to deception by attackers, as evidenced by frequent breaches [12].

Considering these issues, we propose the use of operational technology (OT) and measured data from system sensors collected in the supervisory control and data acquisition (SCADA) system. A measurement intrusion detection system (MIDS) that uses measurement data to train ML algorithms for attack detection was introduced in [13], demonstrating its reliability in enhancing ICS security. To support this approach, Ziadoon et al. reviewed the accuracy of various ML algorithms in ICS anomaly detection as a binary classification problem, using standard datasets like KDD-Cup1999, NSL-KDD, and UNSW-NB15 [14] to demonstrate the effective performance of ML algorithms in both network traffic and measurement data [15]. Moreover, a study conducted by Kumar et al. compared the performance of ML algorithms across four datasets associated with a gas pipeline, a smart grid, a water treatment plant, and the HIL-based Augmented ICS (HAI), revealing variations in ML algorithm performance due to characteristics of datasets and data sources such as power measurement units (PMU), sensors, and actuators [16].

This paper presents a learning-based fault detection, isolation, and identification (FDII) mechanism that employs OT. Unlike IT-based solutions that predominantly manage network traffic between ICS levels, OT-based approaches directly analyze data from system components like sensors in the SCADA system. Our research focuses on evaluating the method's ability to isolate and identify faults in ICSs, particularly in scenarios where the mathematical model of the system is unavailable, rendering traditional methods ineffective. The performance of the proposed method is evaluated within an LFC system, highlighting its potential to significantly enhance security in critical infrastructure.

The main contributions of this paper can be summarized as follows:

1. The LFC model of a two-area interconnected smart grid is simulated in Python, and different types of attacks are injected into the system. Various ML algorithms are tested to classify the normal and abnormal behavior of the system during the attack period.
2. Operational technology is utilized to train multi-classification machine learning algorithms to identify the attack types, leveraging exploited measurement data.
3. The proposed method is employed to accurately locate the origin of detected faults in the LFC system, thereby aiding human operators in managing incidents more effectively and promptly.

The rest of this paper is organized as follows: Section 2 is devoted to describing the LFC model. Section 3 explains the ML method for the detection, isolation, and identification of false data injection attacks. The proposed method is implemented and tested in the simulated LFC system detailed in Section 4. The results are discussed in Section 5. In Section 6, the practical applicability of the proposed FDII mechanism is discussed. Finally, Section 7 concludes the paper.

## 2. LFC Model

The utilized load frequency control (LFC) model is essential for maintaining the stability and efficiency of smart grids, which simulate an n-area interconnected grid using steady-state equations. This model is crucial for the operational reliability of power systems, as it helps synchronize frequency across different areas by adjusting power outputs in response to real-time demand and supply conditions. In the LFC model, a series of mathematical representations is used to describe each area's behavior within the interconnected grid. The state and control vectors, load deviations, and measurement vectors for each area are defined, allowing for precise control and monitoring.

$$\begin{cases} \dot{X}_t^i = A^{ii} X_t^i + \sum\limits_{j=1, j\neq i}^{n} A^{ij} X_t^j + B^i U_t^i + D^i \omega_t^i \\ Y_t^i = C^i X_t^i + \Phi_t^i \end{cases} \tag{1}$$

where:

- $X_t^i$ and $U_t^i$ represent the state and control vectors for the $i$th area, respectively;
- $D^i$ indicates the load deviation for the $i$th area;
- $Y_t^i$ is the measurement vector of the $i$th area, with $\Phi_t^i$ accounting for any bounded noise affecting measurements;
- $C^i$ is an $n$-dimensional identity matrix representing interactions between different grid areas.

This dynamic model provides a robust framework for simulating interconnected power systems and serves as the foundation for analyzing fault detection and isolation in the LFC system.

Matrices $A^{ii}$, $A^{ij}$, $B^i$, and $\omega_t^i$ detail the relationships between and influences of different areas, which are crucial for understanding grid dynamics and optimizing performance in response to operational variances and potential fault conditions. This model is primarily used for simulating the LFC system, which is essential for dataset generation and system design tests that underpin the FDII mechanisms discussed in this paper.

The constant matrices of $A$, $B$, and $\omega_t^i$ are defined as

$$\begin{gathered} A = \begin{bmatrix} A^{11} & A^{12} & \cdots & A^{1n} \\ A^{21} & A^{22} & \cdots & A^{2n} \\ \vdots & \vdots & \vdots & \vdots \\ A^{n1} & A^{n2} & \cdots & A^{nn} \end{bmatrix} \\ B = \mathrm{diag}\left\{ \begin{bmatrix} B^{1T} & B^{2T} & \cdots & B^{nT} \end{bmatrix}^T \right\} \\ \omega_t^i = \mathrm{diag}\left\{ \begin{bmatrix} \omega_t^{1T} & \omega_t^{2T} & \cdots & \omega_t^{nT} \end{bmatrix}^T \right\} \end{gathered} \tag{2}$$

where matrices $A^{ii}$, $A^{ij}$, $B^i$, and $\omega_t^i$, with $i, j = 1, 2, .., n$, are described as

$$A^{ii} = \begin{bmatrix} -\frac{\delta_i}{\mu_i} & \frac{1}{\mu_i} & 0 & -\frac{1}{\mu_i} & 0 \\ 0 & -\frac{1}{\tau_{t_i}} & \frac{1}{\tau_{t_i}} & 0 & 0 \\ -\frac{1}{v_i \tau_{g_i}} & 0 & -\frac{1}{\tau_{g_i}} & 0 & 0 \\ \sum_{j=1, j\neq i}^{n} 2\pi\tau_{ij} & 0 & 0 & 0 & 0 \\ \beta_i & 0 & 0 & 0 & 1 \end{bmatrix};$$

$$A^{ij} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -2\pi\tau_{ij} & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$B^i = \begin{bmatrix} 0 & 0 & \frac{1}{\tau_{g_i}} & 0 & 0 \end{bmatrix}^T;$$

$$\omega_t^i = \begin{bmatrix} -\frac{1}{\mu_i} & 0 & 0 & 0 & 0 \end{bmatrix}^T,$$

where for the $i$th generator, $\delta_i$ is the damping coefficient and $\mu_i$ is the moment of inertia. Furthermore, $\tau_{t_i}$ and $\tau_{g_i}$ are the time constant of the $i$th turbine and the governor, respectively. $\tau_{ij}$ presents the stiffness constant between the $i$th and $j$th areas, and $\beta_i$ is the frequency bias factor of the $i$th turbine. It should be mentioned that the above model is used only for simulating the LFC system for the purpose of dataset generation and design tests.

## 3. False Data Detection, Isolation, and Identification

This section introduces a learning-based fault detection, isolation, and identification (FDII) strategy using ML algorithms to classify system data into normal or abnormal categories, enhancing fault detection. This binary classification leverages both supervised and unsupervised learning, depending on data availability. Supervised learning is ideal for datasets with known outcomes of normal and abnormal behaviors, enabling precise detection of similar future incidents. Unsupervised learning, on the other hand, identifies deviations from normal patterns, detecting a broader range of anomalies.

Fault isolation employs multi-class classification to pinpoint the sources of faults in components like sensors and actuators. Generating a comprehensive dataset, either from historical data or through controlled fault injections, is crucial for this analysis. However, the operational criticality of ICSs often prohibits shutdowns for testing. To address this, a hardware-in-the-loop (HIL) strategy is adopted to prevent system damage during tests [17].

Fault identification further categorizes the nature of detected faults using multi-class classification, facilitating timely and effective responses. Unlike traditional approaches that rely on IT network data, this mechanism uses direct measurement data from OT, providing a robust defense against sophisticated cyber threats.

The subsequent sections detail the methodology of designing and testing this learning-based, OT-focused FDII system, highlighting its advantages over conventional IT-based systems in detecting significant cyber threats.

### *3.1. Learning-Based FDII Mechanism*

Learning-based fault detection is a binary classification strategy that employs ML algorithms to distinguish between normal and abnormal data. Depending on the desires of an ML designer, algorithms can be trained using supervised or unsupervised approaches. Basically, if a set of labeled data, including the normal and abnormal activities (attacks, faults, etc.), is exploited from the system, a supervised learning method can be used to build a fault detection model. The trained ML model based on this method detects any further similar incidents that exist in the training dataset. On the other hand, unsupervised learning algorithms can be trained by employing only normal data. In the unsupervised learning approach, an ML algorithm learns the system's normal behavior, and any outliers relative to normal activity are flagged as anomalies. Supervised learning has a higher rate of accuracy than unsupervised learning. However, unsupervised learning can detect a broader range of incidents [18].

Fault isolation involves locating the source of a detected fault. The learning-based fault isolation problem can be categorized as a multi-class classification ML algorithm.

This algorithm diagnoses the origin of a fault in system components, such as sensors, actuators, etc. The supervised learning strategy is used to address this kind of problem. Therefore, generating a labeled dataset consisting of data associated with the various types of previously occurring faults is vital. This dataset can be generated using the previous incident's data available in the system's log. It is also possible to inject a set of controlled faults into the system components to generate an abnormal dataset. As result, the system probably needs to be shut down. However, shutting down a critical ICS is not possible due to the significance of its operation. For instance, a massive power plant cannot be shut down easily. A solution to address this problem is to use the HIL strategy [17], which simulates the critical parts of a system to prevent severe damage to the system during the controlled fault injection process.

When an incident happens in a system, any kind of relevant information is leveraged to compensate for the incident more accurately in a shorter time. Fault identification is one of the most helpful mechanisms for determining a detected fault's nature and characteristics. This mechanism is based on a multi-class classification strategy, which can distinguish the type of fault among a set of previously detected incidents.

The learning-based FDII mechanism investigated in this paper uses measurement data instead of network traffic. Previously, research studies developed strategies using network data to find abnormal activities in ICSs. However, many reputationally destructive cyber attacks have occurred in critical infrastructure worldwide, which IT-based FDII systems failed to detect. The following describes the procedure for designing and testing a learning- and OT-based fault detection, isolation, and identification mechanism.

### *3.2. Methodology*

This subsection outlines the framework for designing a learning-based FDII mechanism using measurement data, including both off-line and on-line processes. We employed several machine learning algorithms, including Support Vector Machines (SVMs), Random Forest (RF), and Neural Networks (NNs), each chosen for their specific strengths in handling different aspects of the data. SVMs were selected for their robustness in high-dimensional spaces, RF for its ability to handle large datasets with high accuracy, and NNs for their ability to model complex non-linear relationships. The framework for designing a learning-based FDII mechanism using measurement data is shown in Figure 1. In this mechanism, there are two main processes, namely on-line and off-line processes. In the off-line process, an ML model is trained and tested for the FDII mechanism, which is implemented in the system for real-time applications. In the on-line process, the designed FDII system investigates the measurement data associated with each sampling period. If the FDII system detects any abnormal activity, it is reported to the security operation center for further action.

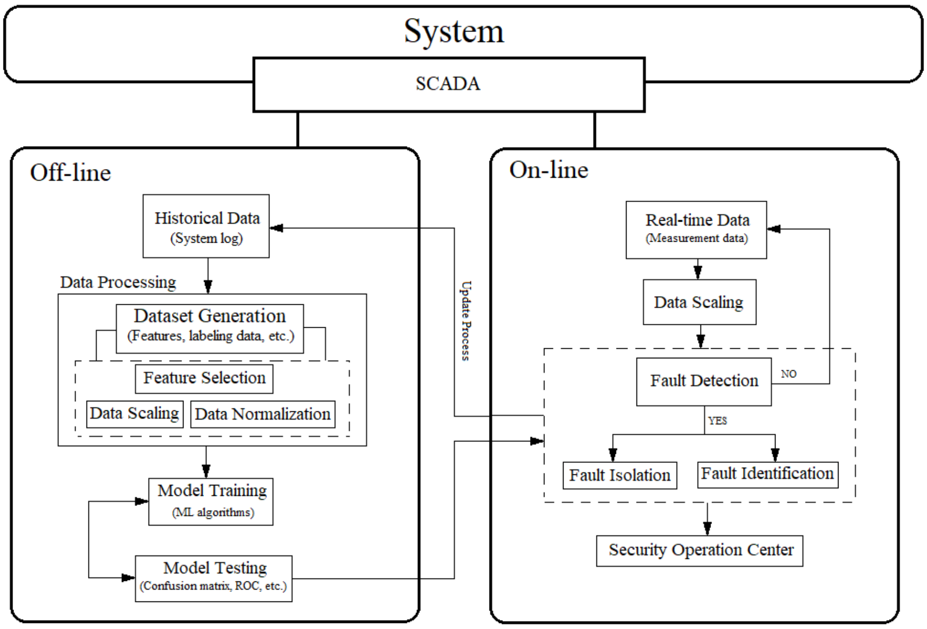

System
SCADA
Off-line
Historical Data (System log)
Data Processing
Dataset Generation (Features, labeling data, etc.)
Feature Selection
Data Scaling
Data Normalization
Model Training (ML algorithms)
Model Testing (Confusion matrix, ROC, etc.)
Update Process
On-line
Real-time Data (Measurement data)
Data Scaling
Fault Detection
NO
YES
Fault Isolation
Fault Identification
Security Operation Center

**Figure 1.** The framework of learning-based FDII design using measurement data.

### 3.2.1. Off-Line Process

This part explains the steps involved in the off-line process, such as data engineering, dataset generation, and ML model training and testing. The off-line process begins with data engineering, where raw sensor data are preprocessed to handle missing values, normalize features, and extract relevant features. Missing values were handled using techniques such as mean imputation for numerical features and mode imputation for categorical features. Feature normalization was performed using min–max scaling to ensure that all features contributed equally to the model training process. Feature selection was conducted using a combination of correlation analysis and feature importance scores from preliminary models. Highly correlated features were identified, and redundant features were removed to reduce multicollinearity and improve model performance. Additionally, domain knowledge was applied to ensure that only relevant features were included in the final dataset. To address the issue of imbalanced data, the Synthetic Minority Oversampling Technique (SMOTE) was employed. SMOTE generates synthetic samples for the minority class by interpolating between existing minority-class samples, thereby balancing the class distribution. This technique was crucial in ensuring that the machine learning models could learn effectively from both majority and minority classes. We generated a comprehensive dataset by simulating various fault conditions and normal operations. Each machine learning algorithm was then trained on this dataset using grid search for hyperparameter optimization. For instance, the SVM model was tuned with different kernel functions (linear, polynomial, and radial basis functions) to find the optimal settings, while the RF model's performance was optimized by adjusting the number of trees and maximum depth parameters. Neural networks were trained using a multi-layer perceptron architecture with backpropagation, adjusting the number of hidden layers and neurons to balance accuracy and computational efficiency. In the off-line process, a set of historical data from the SCADA system was exploited, and after data analysis, an ML model was trained and tested. In this paper, the OT-based data include measurements collected from the system's sensors during normal and abnormal performance. Data engineering consists of dataset generation, data labeling, feature selection, data scaling, and data normalization.

Data labeling involves determining the features and targets of the dataset, in which the features are measurement data and the target is a binary indicating normal or abnormal performance. Generally, the amount of abnormal data is far less than that of normal data, leading to an unbalanced dataset. This could lead to an overfitting problem in the trained ML model, which means the accuracy of prediction for unforeseen data would be lower than the desired value. To avoid the unbalanced dataset issue, two main techniques can be employed, namely over-sampling and under-sampling. Under-sampling methods balance the dataset by removing data from the majority class, while over-sampling methods add data to the minority class. Basically, removing data could lead to the loss of significant amounts data, and adding redundant data increases the risk of overfitting. In this paper, an over-sampling method, in which a set of new data is generated from the existing instances of the minority class, was used to address the unbalanced dataset problem. The SMOTE technique smoothly moves the points of a dataset from the minority class to a close neighborhood and generates new data. This minimizes the risk of overfitting by adding completely non-redundant data [19].

Feature selection finds the most significant features in a dataset. Methods such as filter, wrapper, embedded, and hybrid methods are employed for feature selection [20]. Due to the nature of the measurement data and the importance of keeping the original data untouched for further consideration, the proposed mechanism uses the filter method. In this method, the most correlated features are unified, and among the remaining features, those most correlated with the target are selected. The Pearson correlation technique was used in this study, and it can be described as

$$PC(k) = \frac{cov_{f_k,t}}{\sqrt{var_{f_k} \times var_t}} \tag{3}$$

where $PC(k)$ is the Pearson correlation for feature $k$, and $cov_{f_k,t}$ represents the covariance between the $k$th feature $(f)$ and target $(t)$. $var_{f_k}$, and $var_t$ are the variance of the $k$th feature and the target, respectively.

The data-scaling process is critical when the features have a wide range of values. In ICSs, measurement data values can vary from zero to very large absolute values. Therefore, it is necessary to scale data to prevent any possible overfitting problems in the training process. The most used data-scaling methods are min–max scaling, standardization (also called Z score), binarizing, and normalizing [21]. Since the min–max scaling method normalizes the data between zero and one, it may be an appropriate choice for the scaling of measurement data. The min–max scaler is described below.

$$f_k^{norm} = \frac{f_k - f_{min}}{f_{max} - f_{min}} \tag{4}$$

where $f_k^{norm}$ is the $k$th scaled feature, and $f_k$ indicates the $k$th feature. $f_{min}$ and $f_{max}$ are the minimum and the maximum values of the $k$th feature among all the sampling data, respectively.

The processed data, including the most significant, scaled, and normalized features, along with the determined labels, are used to train ML algorithms. In this step, the following models are trained and tested for the tasks of detection, isolation, and identification of faults. In the first step, a binary classification model is trained for fault detection using algorithms such as $k$-nearest neighbor (KNN), decision tree classifier (DTC), and random forest (RF). The second and third steps include training of multi-class classification algorithms, like multi-label KNN (MLKNN), support vector classifier (SVC), one-vs.-one classifier (OVOC), one-vs.-rest classifier (OVRC), and binary relevance classifier (BRC).

### 3.2.2. On-Line Process

This part describes the on-line process, where the designed FDII system investigates the measurement data for real-time fault detection, isolation, and identification. In the on-line process, real-time measurement data from the sensors are continuously fed into the trained machine learning models. The models classify the data in real time, identifying potential faults. The chosen algorithms allow for quick adaptation and provide probability scores for the presence of faults, enabling the system to isolate and identify the type of fault with high precision. The decision-making process integrates these probabilities with predefined thresholds to trigger alarms and initiate corrective actions. Additionally, continuous learning is implemented, where the system periodically updates the models using new data, ensuring that the FDII mechanism remains effective against evolving fault patterns and cyber-attack strategies. While the ML model is trained and successfully tested in the off-line process, it can be implemented in the system for real-time applications. In the on-line process, regarding the sampling time, the measurement data from the SCADA system are exploited. Then, the scaled data are tested by the fault detection mechanism; if they are flagged as abnormal, the fault isolation and identification mechanisms provide more detail about the location and nature of the fault. Lastly, a report consisting of all the details related to the incident is sent to the security operational center for any further action that may be required.

It should be mentioned that the off-line process may take longer than the on-line process. Indeed, the off-line process can take seconds to hours, while the on-line process requires less than a very small portion of a second, i.e., nanoseconds. This is due to the volume of the processing data in these two phases. While in the off-line step, a dataset including thousands of samples is process for the training of ML algorithms, in the on-line step, just one row of data is processed in each sampling period. The reduced amount of computation required in the learning-based method make its real-time application faster compared to model-based strategies, which deal with very complex mathematical formulas.

Since the ML algorithms in the FDII mechanism are trained for a specific set of data, the accuracy of these algorithms may decrease as a result of any changes in the system's dynamics over time. Therefore, the update process is in charge of tuning the trained ML models regarding changes in the system dynamics and new incidents log. In this process, the data related to the new dynamic system and recent incidents are added to the updated dataset, and the ML algorithms are trained and tested in the off-line process. In Section 4, the proposed framework is applied to the LFC system.

### *3.3. Evaluation Metrics*

To comprehensively assess the performance of the machine learning algorithms, we used several evaluation metrics, including precision, recall, F1 score, and area under the curve (AUC). Precision is defined as the ratio of true-positive predictions to the total number of positive predictions, indicating the accuracy of the positive predictions made by the model. Recall, also known as sensitivity, is the ratio of true-positive predictions to the total number of actual positives, reflecting the model's ability to identify all relevant instances. The F1 score, which is the harmonic mean of precision and recall, provides a single metric that balances both precision and recall. This is particularly useful in scenarios where an imbalanced dataset might skew the results of precision or recall alone [13]. The area under the curve (AUC) of the Receiver Operating Characteristic (ROC) is another critical metric used to evaluate the models. The AUC measures the ability of the model to distinguish between classes and is particularly useful for comparing the performance of different models. A higher AUC value indicates a better-performing model in terms of distinguishing between positive and negative classes. In our experiments, these metrics were calculated for each machine learning algorithm to provide a comprehensive evaluation of their performance. The results were then analyzed to determine the strengths and weaknesses of each algorithm in the context of fault detection, isolation, and identification. This detailed analysis allowed us to select the most effective models for the FDII system and ensure robust performance in real-world scenarios.

## 4. Case Study: Load Frequency Control

In this section, the proposed FDII mechanism is implemented on a simulated LFC system, as shown in Figure 2. In this model, an interconnected two-area smart grid with automatic gain control (AGC), in which all power plants in each area are modeled as an equivalent power plant, is mathematically simulated. Using the network connections, the two areas transmit the sensor data to the SCADA system, and the LFC sends back the control signals to maintain the frequency by adjusting the governor's incremental valve position. Since the LFC design is based on a feedback controller, the real-time FDII has a critical role in the reliability of the grid. If, for any reason, the LFC's dynamic states deviate from the desired values, the whole power grid is at risk of a breakdown.

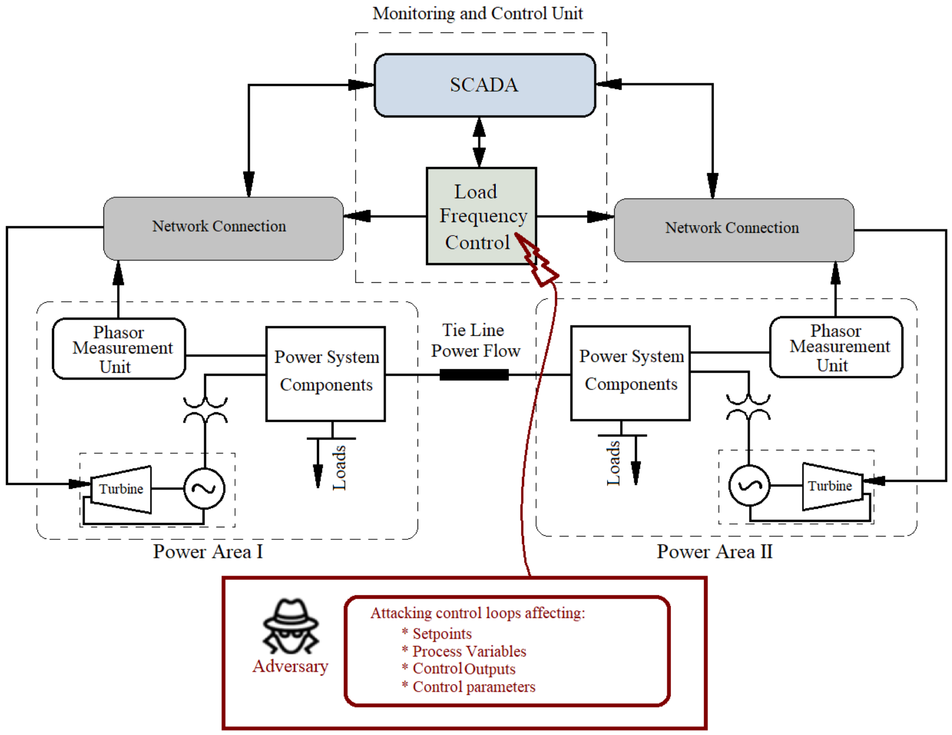


**Figure 2.** The LFC model of a two-area power system.

### *LFC Under Attack*

The concept of a false data injection attack was presented for the first time in the smart-grid domain [22]. Usually, attackers try to compromise sensor measurements in a stealthy way such that undetected deviations are counted in the calculations of state variables and values. Such attacks can have various types of functions, such as step, sine, ramp, etc. A combination of these functions may also be injected into the system simultaneously.

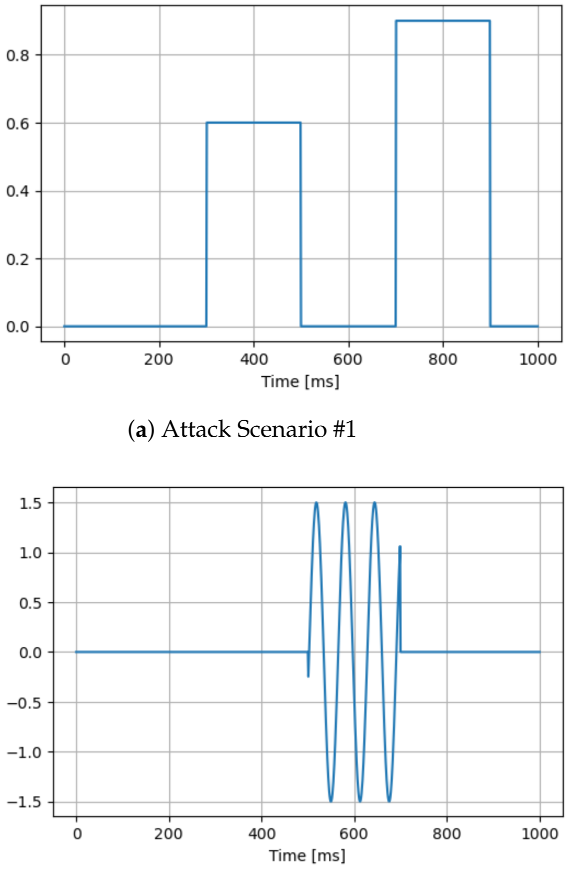
This paper considers four attack scenarios, as shown in Figure 3. The first attack scenario affects the third state variable of the LFC model, using a step function in two time periods of two seconds each. Specifically, the attacker injects a step increase, then another one in the state variable at predetermined times, causing abrupt changes that the system must counteract. This type of attack can lead to sudden deviations in frequency control as the system reacts to the unexpected jumps in sensor readings. The detailed impact of this attack is depicted in Figure 3a.

In the second scenario, a sine wave affects the third state variable of the LFC between seconds 5 and 7. This attack introduces a periodic fluctuation in the state variable, mimicking an oscillatory disturbance. Such an attack is designed to create persistent oscillations in the system's output, challenging the system's ability to maintain stable frequency control. The frequency and amplitude of the sine wave are chosen to match typical operational ranges, making the attack harder to detect. The resulting impact on the system is shown in Figure 3b.

The last two scenarios target the third and eighth state variables at the same time. In the third scenario, a combination of step and sine functions is used to attack the system. This complex attack involves injecting a step function into the third state variable while simultaneously introducing a sine wave into the eighth state variable. The combination of these functions aims to create a multi-faceted disturbance that tests the system's resilience to different types of simultaneous anomalies. Figure 3c illustrates the effects of this dual-function attack.

In the fourth scenario, a triangle function is injected into both the third and eighth state variables. The triangle function causes linear increases and decreases in the state variables, creating ramp-like disturbances that can be particularly challenging for the system to handle due to their continuous and changing nature. This attack scenario is designed to test the system's ability to respond to gradual but consistent changes in sensor readings. The impact of this triangular function attack is shown in Figure 3d.

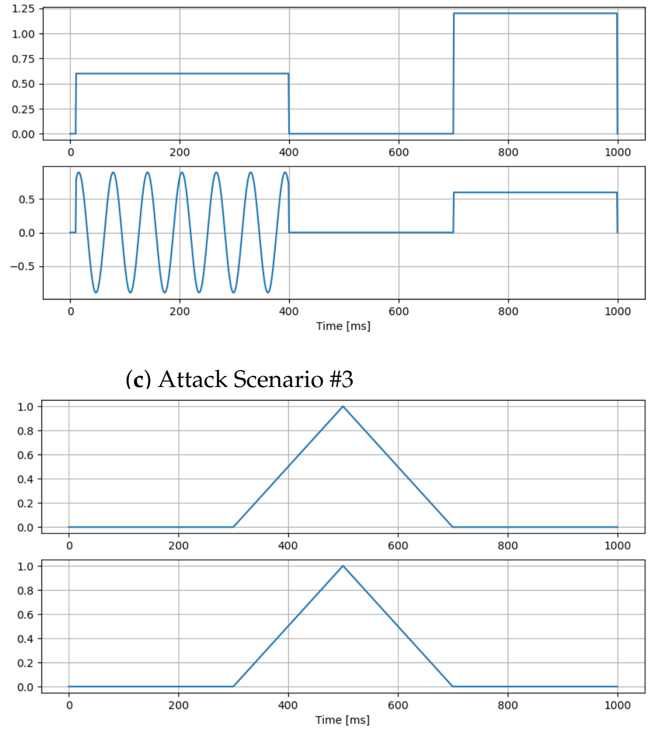
These attack scenarios were meticulously designed to cover a range of potential real-world threats to the LFC system, providing a comprehensive evaluation of the FDII mechanism's effectiveness. The included diagrams and charts illustrate the detailed processes and impacts of each attack scenario, enhancing the understanding of their potential consequences for the LFC system.



(**a**) Attack Scenario #1

(**b**) Attack Scenario #2

**Figure 3.** *Cont.*



(**c**) Attack Scenario #3

(**d**) Attack Scenario #4

**Figure 3.** False data injection attack scenarios.

## 5. Simulation Results and Discussion

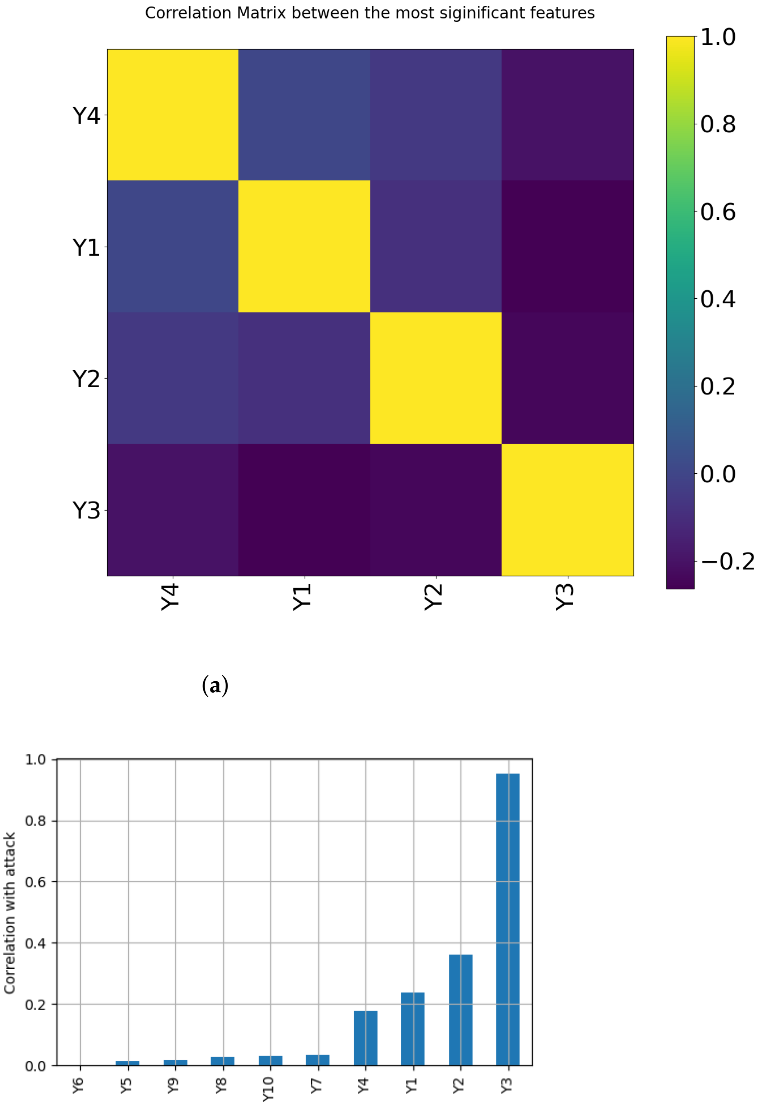
The proposed FDII mechanism is tested on the LFC model with the false data injection attack, and the performance of the FDII system based on the MIDS approach is evaluated in the following.

### 5.1. Fault Detection

Four types of false data, as described in Section 4, were injected into the third and eighth states of the LFC model. The system was run for 10 s for each type of fault to generate labeled datasets. After collection, the data were processed in the off-line phase. Then, four ML models were trained to detect each attack separately.

Data processing consists of balancing the normal and attack cases, scaling sensor values in a standard range, and selecting the most significant features. For data normalization, the SMOTE method was used. The data were balanced for the normal and abnormal cases. Then, the data were scaled with the min–max scaler, which scales all data in the range of zero to one. Finally, the most significant features were selected based on their correlation with the target and independence relative to other features. As an example, in the first attack scenario, the correlation of the together and between features with the target are shown in Figure 4. In this specific scenario of attack, the first four sensors (out of the ten sensors of the LFC model) were chosen in the feature selection process. These four sensors are the most correlated sensors to the step-function attack in the first scenario, as shown in Figure 4b. We selected features with a correlation of more than 85% to the target and less than 5% to each other.

In the training process, three different ML algorithms were employed. The performance of these algorithms in fault detection is compared in Table 1. The confusion matrices for the most accurate ML model in each scenario are shown in Figure 5. The AUC is illustrated in Figure 6. The AUC metric and the confusion matrix show that the RF algorithm achieves better performance in fault detection in the proposed LFC system. The results show that the FDII system was able to detect all four scenarios of attack. However, the accuracy of ML algorithms varies based on the nature of the attack. For instance, in the first scenario, all three algorithms performed perfectly, with an accuracy of 1.0, while in the other scenarios, the performances of KNN and DTC were degraded. Therefore, RF was the most accurate algorithm in all four scenarios. However, the DTC algorithm beat RF in computation time, and it is the fastest algorithm in the prediction process (on-line process). We should select the most fitted algorithm in terms of accuracy and computation time for the prediction process of different attack scenarios. In the third scenario, the DTC algorithm showed the same performance as the RF but with a lower computation time. Therefore, we selected the DTC algorithm rather than RF for this specific scenario of attack in the LFC system.



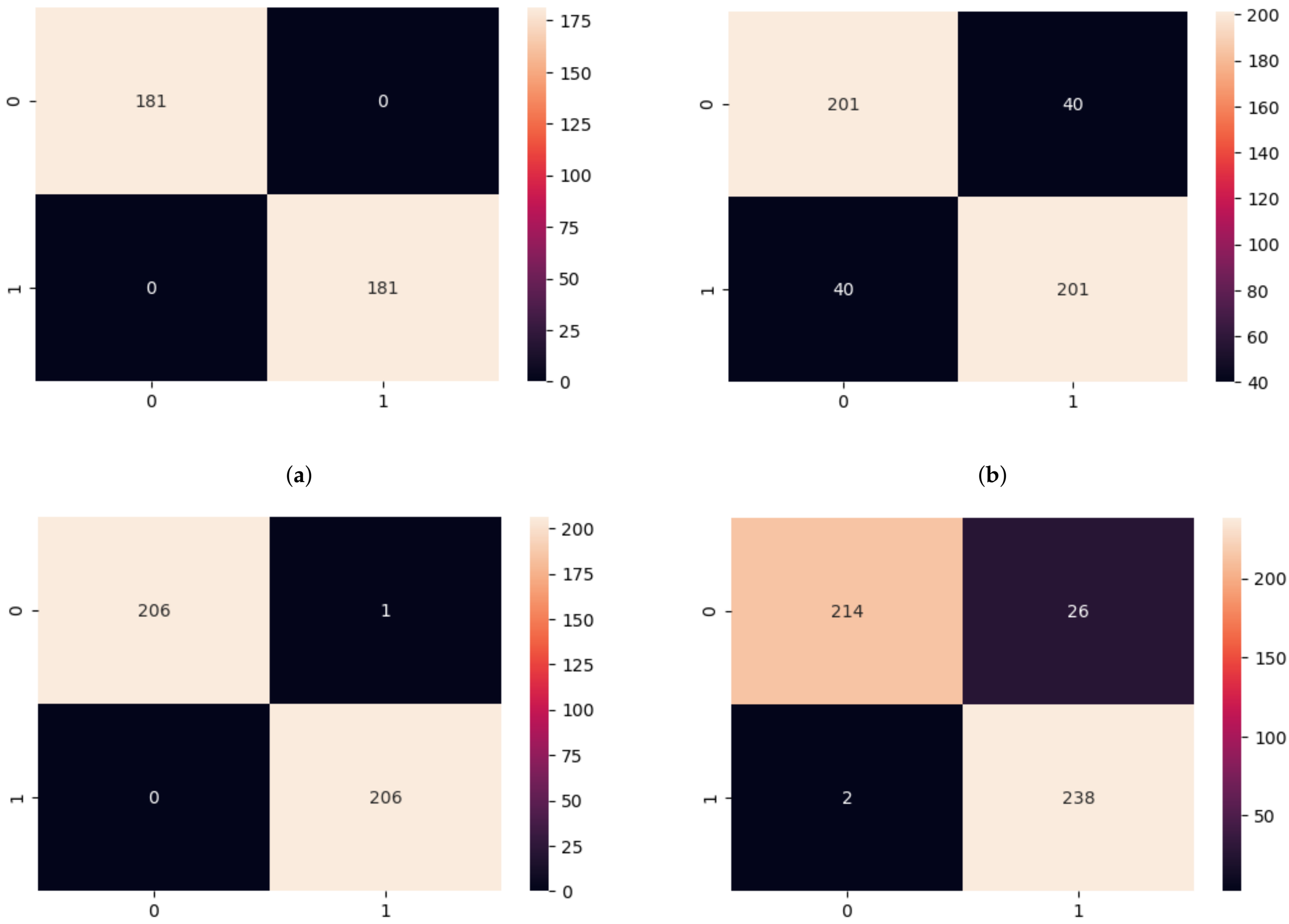
(**a**)

(**b**)

**Figure 4.** Feature selection based on the correlation method. (**a**) The correlation between the most significant features in the first scenario of attack. (**b**) The correlation between features and the target in the first scenario of attack.

**Table 1.** Fault detection performance using the MIDS mechanism.

| | **Attack No. 1** | | | **Attack No. 2** | | | **Attack No. 3** | | | **Attack No. 4** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **KNN** | **RF** | **DTC** | **KNN** | **RF** | **DTC** | **KNN** | **RF** | **DTC** | **KNN** | **RF** | **DTC** |
| Precision | 1.0 | 1.0 | 1.0 | 0.80 | 0.98 | 0.97 | 0.99 | 1.0 | 1.0 | 0.93 | 0.93 | 0.89 |
| Recall | 1.0 | 1.0 | 1.0 | 0.78 | 0.98 | 0.97 | 0.98 | 1.0 | 1.0 | 0.92 | 0.92 | 0.89 |
| F1 score | 1.0 | 1.0 | 1.0 | 0.78 | 0.98 | 0.97 | 0.99 | 1.0 | 1.0 | 0.92 | 0.92 | 0.89 |
| Accuracy | 1.0 | 1.0 | 1.0 | 0.78 | 0.98 | 0.97 | 0.99 | 1.0 | 1.0 | 0.92 | 0.92 | 0.89 |
| AUC | 1.0 | 1.0 | 1.0 | 0.78 | 0.98 | 0.97 | 0.99 | 1.0 | 1.0 | 0.92 | 0.92 | 0.89 |
| Training time [ms] | 0.92 | 23.73 | 2.11 | 1.06 | 32.41 | 5.14 | 2.35 | 27.60 | 31.15 | 2.44 | 26.76 | 11.01 |
| Prediction time [ms] | 19.87 | 5.21 | 0.75 | 23.42 | 6.26 | 0.31 | 31.15 | 5.68 | 1.02 | 27.74 | 8.34 | 1.01 |



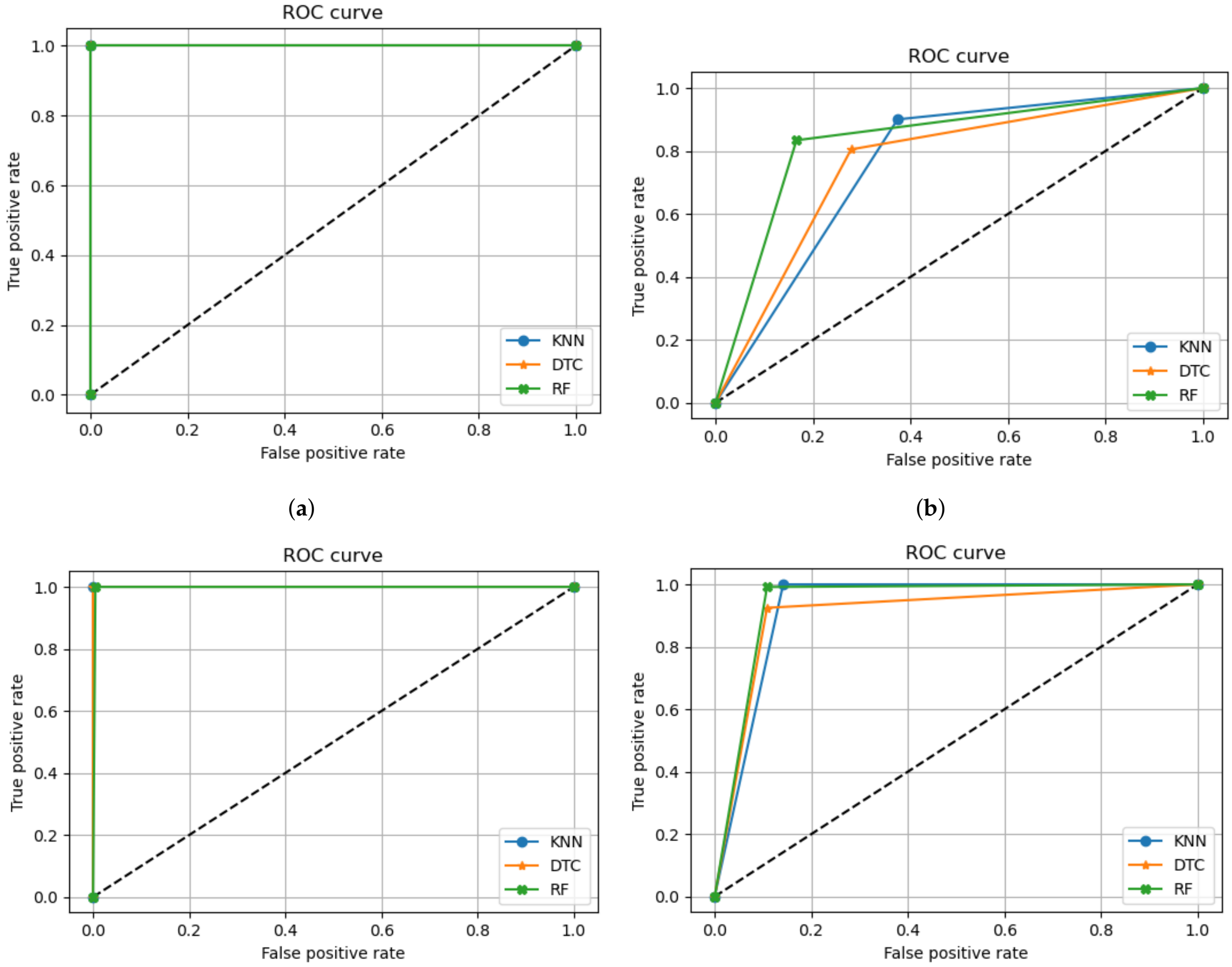
(**a**)

(**b**)

(**c**)

(**d**)

**Figure 5.** Fault detection confusion matrices. (**a**) Confusion matrix for fault detection of the RF model in scenario #1. (**b**) Confusion matrix for fault detection of the RF model in scenario #2. (**c**) Confusion matrix for fault detection of the RF model in scenario #3. (**d**) Confusion matrix for fault detection of the RF model in scenario #4.



(**a**) (**b**)

(**c**) (**d**)

**Figure 6.** Fault detection AUC. (**a**) The area under the curve of the RF model for scenario #1. (**b**) The area under the curve of the RF model for scenario #2. (**c**) The area under the curve of the RF model for scenario #3. (**d**) The area under the curve of the RF model for scenario #4.

### *5.2. Fault Isolation*

A step function was injected into the LFC system state variables for 1.5 s. This process was repeated ten times to attack all ten state variables separately. Therefore, a dataset including 100 s of the LFC operation was generated, consisting of attack data on all ten state variables of the LFC system. After data collection and data processing, ML model training and testing were performed. This dataset included abnormal data for different locations of attacks. The ML algorithms used in this process include RF (MOC), MLKNN, OVOC, OVRC, OCC, and BRC.

We evaluated the computational efficiency of various machine learning algorithms used for fault isolation by comparing their training and prediction times. Table 2 provides a summary of the average accuracy, training time, and prediction time for different multi-class algorithms.

**Table 2.** Comparison of ML models in fault isolation.

| Multi-class Algorithm | Average Accuracy | Training Time [s] | Prediction Time [s] |
|---|---|---|---|
| Multi-Output Classifier (MOC) | 1.0 | 4.67 | 0.98 |
| Multi-*k*-nearest neighbour (MLKNN) | 1.0 | 2.29 | 1.84 |
| One-Vs.-One Classifier (OVOC) | 0.99 | 0.128 | 0.026 |
| One-Vs.-Rest Classifier (OVRC) | 0.98 | 0.164 | 0.003 |
| Output-Code Classifier (OCC) | 0.97 | 0.85 | 0.007 |
| Binary Relevance Classifier (BRC) | 0.93 | 0.022 | 0.008 |

For fault isolation, the Multi-Output Classifier (MOC) exhibited the longest training time of 4.67 s, while the Binary Relevance Classifier (BRC) had the shortest training time of 0.022 s. The training times for the other algorithms varied, with Multi-k-nearest neighbor (MLKNN) taking 2.29 s and the One-Vs.-Rest Classifier (OVRC) taking 0.164 s. The prediction times also varied, with MOC taking 0.98 s and BRC demonstrating the fastest prediction time of 0.008 s. These results highlight the trade-offs between training time and prediction time for different algorithms.

The average accuracies of ML models in diagnosing the fault location and the computation time are shown in Table 2. The results show that the MOC and MLKNN algorithms achieved the highest accuracy in identifying the fault location. However, MOC performed faster in prediction than MLKNN. In the FDII mechanism, the computation time of the on-line process is critical and dependent on the prediction time of each row of data in the sampling period. Therefore, due to lower computation time, the trained model with the MOC algorithm would be chosen for the fault isolation process in this specific case. Moreover, based on Figure 7, the confusion matrix of the MOC algorithm indicates the high accuracy of this model in predicting the fault location during the off-line process.

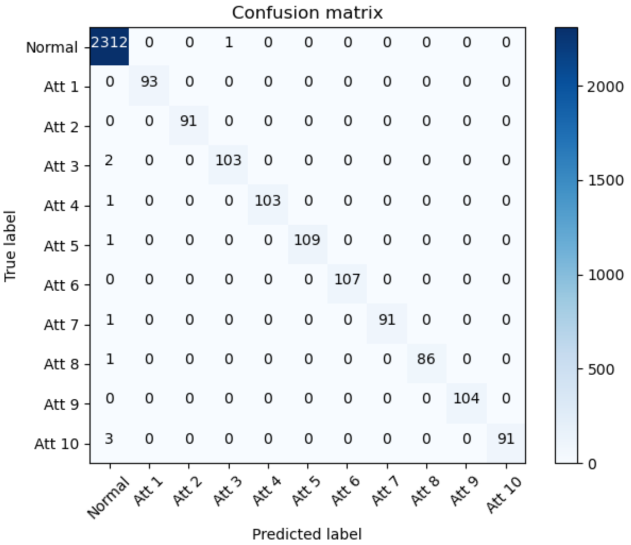


**Figure 7.** The confusion matrix for fault isolation in LFC.

### 5.3. Fault Identification

In this subsection, the four scenarios of attack shown in Figure 3 are injected into the LFC system, and the FDII performance in identifying the attack type is evaluated. The LFC system was run for 40 s, and every 10 s of operation, one of the mentioned attacks was injected into the system. The dataset generated based on system operation was processed using the proposed method, and the ML algorithms were trained and tested. Since fault identification is a multi-classification problem like the fault isolation problem, the ML algorithms used in both steps are the same. In Figure 8, the confusion matrix for fault isolation is shown. This matrix shows the accuracy of fault identification using the MOC algorithm. This algorithm was developed based on the RF strategy. Table 3 shows the performance of different ML algorithms in fault identification. According to the results, the MOC algorithm performed more accurately than other algorithms in fault identification. The average accuracy of the MOC model for fault identification in the LFC model was 98.6%, which is close to that of the MLKNN algorithm, at 98.2%. The MOC algorithm achieved a faster prediction time than the MLKNN algorithm, but for the training process, MLKNN was quicker.

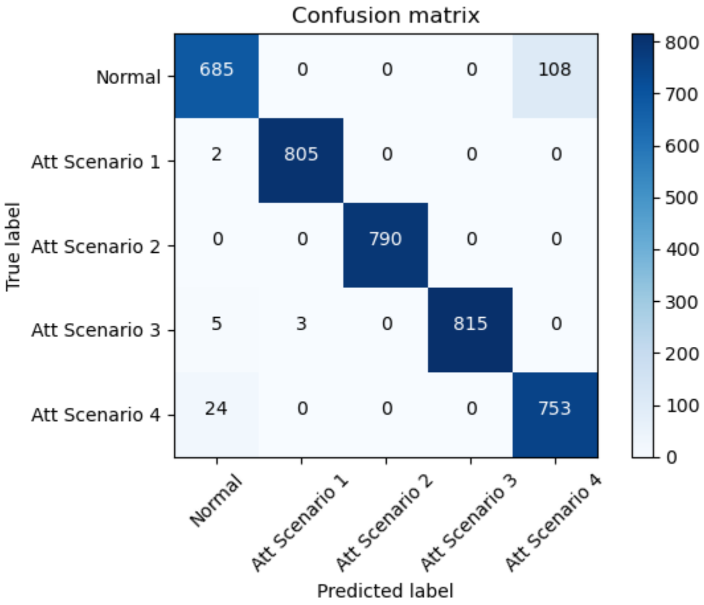


**Figure 8.** The confusion matrix for fault identification in LFC.

**Table 3.** Comparison of ML models in fault identification.

| Multi-class Algorithm | Average Accuracy | Training Time [s] | Prediction Time [s] |
|---|---|---|---|
| Multi-Output Classifier (MOC) | 0.986 | 4.31 | 1.38 |
| Multi-*k*-nearest neighbour (MLKNN) | 0.982 | 2.13 | 1.72 |
| One-Vs.-One Classifier (OVOC) | 0.85 | 0.096 | 0.006 |
| One-Vs.-Rest Classifier (OVRC) | 0.87 | 0.123 | 0.002 |
| Output-Code Classifier (OCC) | 0.80 | 0.297 | 0.003 |
| Binary Relevance Classifier (BRC) | 0.85 | 0.012 | 0.005 |

The computational efficiency of machine learning algorithms in fault identification was also evaluated. Table 3 provides a summary of the average accuracy, training time, and prediction time for different multi-class algorithms. Computational tasks were performed on a PC, equipped with an Intel Core i7 processor, 16 GB of RAM, and an NVIDIA GTX 1070 GPU. This setup provided sufficient processing power and memory to handle the data-intensive tasks and real-time detection required by the FDII system.

For fault identification, the multi-output classifier (MOC) had the longest training time of 4.31 s, while the binary relevance classifier (BRC) had the shortest training time of 0.012 s. The prediction times varied, with MOC taking 1.38 s and BRC demonstrating the fastest prediction time of 0.005 s. These metrics indicate the efficiency of each algorithm in real-time fault identification scenarios.

The suitability of machine learning algorithms for real-time applications in industrial control systems depends on their prediction times and the ability to quickly adapt to new data. Our analysis indicates that simpler algorithms like BRC and OVRC are particularly well-suited for real-time fault detection and isolation due to their low prediction times. Although more complex algorithms like MOC and MLKNN have longer training and prediction times, they offer higher accuracy and robustness, making them suitable for applications where accuracy is paramount.

To ensure real-time applicability, we recommend leveraging hardware acceleration techniques such as Graphics Processing Units (GPUs) and parallel processing to further reduce prediction times. Additionally, implementing incremental learning techniques can help models adapt to new data without requiring complete retraining, enhancing the system's responsiveness to evolving conditions.

In summary, while all the evaluated algorithms have their strengths, simpler models like BRC and OVRC are particularly advantageous for real-time fault detection and isolation in industrial control systems due to their balance of training and prediction efficiency. Future research should focus on optimizing these models for specific industrial contexts and exploring hybrid approaches that combine the strengths of different algorithms.

## 6. Practical Applicability of the Proposed FDII Mechanism

Implementing the proposed FDII mechanism in real-world industrial control systems involves several practical challenges and considerations. This section discusses the feasibility of deployment, potential limitations, and future research directions to enhance the system's applicability and robustness.

### *6.1. Challenges and Considerations*

One of the primary challenges in implementing the FDII mechanism is integration with existing ICSs. ICS environments are often characterized by legacy systems with proprietary protocols, which can complicate the integration process. Ensuring compatibility and seamless communication between the FDII mechanism and these legacy systems is crucial for effective deployment.

Another consideration is the computational requirements of the machine learning models. Real-time fault detection and isolation demand significant computational resources, which may not be readily available in all industrial settings. Deploying the FDII mechanism on edge devices or utilizing cloud computing solutions can help address this challenge, but it requires careful planning and resource allocation.

### *6.2. Potential Limitations*

The proposed FDII mechanism relies heavily on the quality and quantity of sensor data. Inaccurate or incomplete data can significantly impact the system's performance, leading to false positives or missed detections. Therefore, robust data validation and preprocessing techniques are essential to ensure the reliability of the input data.

Additionally, the system's performance may be affected by the dynamic nature of industrial processes. Changes in operational conditions such as varying load demands or equipment maintenance can introduce variability that the machine learning models must account for. Continuous model retraining and adaptation are necessary to maintain high detection accuracy under changing conditions.

## 7. Conclusions

This paper proposes a novel operational technology (OT)-based fault detection, isolation, and identification (FDII) mechanism tailored to smart grids, utilizing measurement data to effectively detect, isolate, and identify false data injection attacks. By considering four distinct types of cyber threats. This study comprehensively evaluated the performance of various ML algorithms in enhancing the security of the LFC model against these sophisticated attacks.

The FDII mechanism was structured into two primary phases, namely an off-line process and an on-line process. In the off-line phase, ML models were meticulously developed to diagnose the specifics of false data injection attacks, including their origin and characteristics. The on-line phase, in contrast, focuses on the real-time analysis of sampled measurement data collected from the SCADA system, enabling immediate responses to potential security breaches.

The results of simulations conducted using Python demonstrate the efficacy of the proposed FDII mechanism in safeguarding the LFC system against the considered types of attacks. These simulations underscore the potential of integrating advanced machine learning techniques with operational technology to enhance grid security.

In conclusion, the success of this OT-based FDII mechanism not only demonstrates its viability but also paves the way for more robust AI-driven security frameworks in critical infrastructure systems. As smart grids become increasingly integral to global energy networks, ensuring their resilience against cyber threats is paramount. Looking ahead, further research is necessary to explore the scalability of this FDII mechanism across different grid configurations and its adaptability to evolving cyber threat landscapes.

### *Future Research Directions*

Future research should focus on enhancing the adaptability and resilience of the FDII mechanism. Developing advanced machine learning models that can learn and adapt to new patterns with minimal supervision will be crucial. Techniques such as transfer learning and online learning can be explored to improve the system's ability to handle evolving threats and operational changes.

Another area of interest is the incorporation of advanced anomaly detection techniques such as deep learning models and ensemble methods to further enhance detection accuracy. Research on improving the interpretability and explainability of machine learning models will also be beneficial, enabling operators to better understand the system's decision-making process and take appropriate actions.

Finally, extensive field testing and pilot deployments in diverse industrial environments can provide valuable insights into the practical challenges and performance of the FDII mechanism. Collaboration with industry partners can facilitate these efforts, leading to more robust and scalable solutions for real-world applications.

In summary, while the proposed FDII mechanism shows promise for enhancing the security of industrial control systems, careful consideration of integration challenges, computational requirements, data quality, and dynamic operational conditions is necessary. Future research should focus on improving model adaptability, incorporating advanced detection techniques, and conducting extensive field testing to ensure practical applicability and robustness.

**Author Contributions:** Methodology, S.M. and K.K.Y.; Validation, S.M.; Data curation, S.M.; Writing—original draft, S.M.; Writing—review & editing, K.K.Y.; Supervision, K.K.Y.; Funding acquisition, S.M. and K.K.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was partially funded by the Dissertation Evidence Acquisition (DEA) fellowship, Spring 2022, Florida International University, FL, USA.

**Data Availability Statement:** The raw data supporting the conclusions of this article will be made available by the authors on request.

**Conflicts of Interest:** The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results

## References

1. Leppänen, S.; Ahmed, S.; Granqvist, R. Cyber Security Incident Report—Norsk Hydro. *Procedia Econ. Financ.* **2019**, 11. Available online: https://mycourses.aalto.fi/pluginfile.php/923542/mod_folder/content/0/Group%20CSS%20Norsk%20Hydro%202019.pdf (accessed on 2 August 2024) .
2. Abbaspour, A.; Sargolzaei, A.; Forouzannezhad, P.; Yen, K.K.; Sarwat, A.I. Resilient control design for load frequency control system under false data injection attacks. *IEEE Trans. Ind. Electron.* **2019**, *67*, 7951–7962. [CrossRef]

3. Li, Y.; Huang, R.; Ma, L. False data injection attack and defense method on load frequency control. *IEEE Internet Things J.* **2020**, *8*, 2910–2919. [CrossRef]
4. Qi, R.; Rasband, C.; Zheng, J.; Longoria, R. Detecting Cyber Attacks in Smart Grids Using Semi-Supervised Anomaly Detection and Deep Representation Learning. *Information* **2021**, *12*, 328. [CrossRef]
5. Alzubi, J.A. Bipolar fully recurrent deep structured neural learning based attack detection for securing industrial sensor networks. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4069. [CrossRef]
6. Sayghe, A.; Hu, Y.; Zografopoulos, I.; Liu, X.; Dutta, R.G.; Jin, Y.; Konstantinou, C. Survey of machine learning methods for detecting false data injection attacks in power systems. *IET Smart Grid* **2020**, *3*, 581–595. [CrossRef]
7. *IEEE Std 802.15.4-2020 (Revision of IEEE Std 802.15.4-2015)*; IEEE Standard for Low-Rate Wireless Networks. IEEE: Piscataway, NJ, USA, 2020; pp. 1–800. [CrossRef]
8. Jokar, P.; Leung, V.C.M. Intrusion detection and prevention for ZigBee-based home area networks in smart grids. *IEEE Trans. Smart Grid* **2016**, *9*, 1800–1811. [CrossRef]
9. Revathi, S.; Malathi, A. A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. *Int. J. Eng. Res. Technol. (IJERT)* **2013**, *2*, 1848–1853.
10. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
11. Aboelwafa, M.M.N.; Seddik, K.G.; Eldefrawy, M.H.; Gadallah, Y.; Gidlund, M. A machine-learning-based technique for false data injection attacks detection in industrial IoT. *IEEE Internet Things J.* **2020**, *7*, 8462–8471. [CrossRef]
12. Ramotsoela, D.T.; Hancke, G.P.; Abu-Mahfouz, A.M. Attack detection in water distribution systems using machine learning. *Hum.-Centric Comput. Inf. Sci.* **2019**, *9*, 13. [CrossRef]
13. Mokhtari, S.; Abbaspour, A.; Yen, K.K.; Sargolzaei, A. A machine learning approach for anomaly detection in industrial control systems based on measurement data. *Electronics* **2021**, *10*, 407. [CrossRef]
14. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015.
15. Maseer, Z.K.; Yusof, R.; Bahaman, N.; Mostafa, S.A.; Foozy, C.F.M. Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset. *IEEE Access* **2021**, *9*, 22351–22370. [CrossRef]
16. Kumar, A.; Choi, B.J. Benchmarking Machine Learning based Detection of Cyber Attacks for Critical Infrastructure. In Proceedings of the 2022 International Conference on Information Networking (ICOIN), Jeju-si, Republic of Korea, 12–15 January 2022; pp. 24–29.
17. Gheisarnejad, M.; Khooban, M.H. Secondary load frequency control for multi-microgrids: HiL real-time simulation. *Soft Comput.* **2019**, *23*, 5785–5798. [CrossRef]
18. Mokhtari, S.; Yen, K.K. Measurement data intrusion detection in industrial control systems based on unsupervised learning. *Appl. Comput. Intell.* **2021**, *1*, 61–74. [CrossRef]
19. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
20. Chandrashekar, G.; Sahin, F. A survey on feature selection methods. *Comput. Electr. Eng.* **2014**, *40*, 16–28. [CrossRef]
21. Zheng, A.; Casari, A. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*; O'Reilly Media, Inc.: Newton, MA, USA, 2018.
22. Ahmed, M.; Pathan, A.-S.K. False data injection attack (FDIA): An overview and new metrics for fair evaluation of its countermeasure. *Complex Adapt. Syst. Model.* **2020**, *8*, 4. [CrossRef]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.