BEYKOZ ÜNİVERSİTESİ

GRADUATE PROGRAMS INSTITUTE

DEEP LEARNING AND MACHINE LEARNING METHODS FOR DETECTING FALSE DATA INJECTION CYBER-ATTACKS ON SMART GRID PHASOR MEASUREMENT UNITS

Khalid GOUN

MASTER'S THESIS

İstanbul, August 2024

DEEP LEARNING AND MACHINE LEARNING METHODS FOR DETECTING FALSE DATA INJECTION CYBER-ATTACKS ON SMART GRID PHASOR MEASUREMENT UNITS

Khalid GOUN

MASTER'S THESIS

COMPUTER ENGINEERING DEPARTMENT

COMPUTER ENGINEERING MASTER'S PROGRAM WITH THESIS

MASTER'S THESIS ADVISOR

Asst. Prof. Dr. Özlem Feyza ERKAN

MASTER'S THESIS JURY MEMBERS

Asst. Prof. Dr. İnal Begüm TURNA DEMİREL

Asst. Prof. Dr. Nedim MUZOĞLU

CONTENTS

ÖZET

AKILLI ŞEBEKE FAZÖR ÖLÇÜM ÜNİTELERİNDE SAHTE VERİ ENJEKSİYONU SİBER SALDIRILARININ TESPİTİ İÇİN DERİN ÖĞRENME VE MAKİNE ÖĞRENMESİ YÖNTEMLERİ

Akıllı şebekeler, normal güç sistemlerine kıyasla çeşitli avantajlar sunar. Ancak özellikle veri bütünlüğü ve güvenliği açısından önemli siber güvenlik zorlukları da sunarlar. Fazör Ölçüm Birimleri (PMU), senkronize, yüksek çözünürlüklü ölçümler aracılığıyla şebekenin gerçek zamanlı izlenmesini ve yönetimini sağlayarak bu zorlukların ele alınmasında önemli bir rol oynar ve böylece güvenilirliğini artırır. Fakat bilgisayar korsanları hayati öneme sahip Fazör Ölçüm Birim bilgilerini manipüle ederek bazı avantajlar elde edebilirler. PMU'lara yanlış veri girişi yapılmasına dayanan yanlış veri enjeksiyon saldırıları, tüm güç şebekesinin verimliliğini ve güvenilirliğini tehlikeye atabileceğinden ciddi bir tehdit oluşturur. Bu nedenle, akıllı şebekelerdeki güvenli operasyonlar etkili yanlış veri tespiti ve azaltmaya bağlıdır. Bu tez çalışmasında, akıllı şebekelerdeki yanlış veri saldırılarını tanımak için hem makine öğrenimi hem de derin öğrenme yöntemleriyle desteklenen kapsamlı bir çözüm geliştirilmesi hedeflenmiştir. Bu çözüm, güç sistemi saldırısı etiketli veriler kullanılarak geliştirilen denetimli makine öğrenimi modellerinin kullanılmasını içerir. Rastgele Ormanlar ve Ekstra Ağaçlar gibi ağaç tabanlı teknikler öncelikle PMU verilerini doğal veya riskli olay olarak ikili bir sınıflandırma kullanılır. Daha sonra olaysız, doğal olay veya aktif saldırıyı belirlemek için çok sınıflı sınıflandırma yapılır. Bunun yanı sıra, özellikle zaman serisi verileri için tasarlanmış ve ince zamansal kalıpları etkili bir şekilde tespit edip yakalayabilen Evrişimsel Sinir Ağları (CNN), Uzun Kısa Süreli Bellek Ağları (LSTM) ve CNN-LSTM hibritleri gibi derin öğrenme mimarilerini incelenmiştir. Bu araştırmanın birincil amacı, gerçek PMU verileri üzerinde en son makine ve derin öğrenme tekniklerini kapsamlı bir şekilde değerlendirmek, saldırı tespit doğruluğunda önemli bir iyileştirmeye yol açmak ve nihayetinde kritik enerji tedarik altyapısının güvenliğini ve dayanıklılığını korumaktır.

Anahtar Kelimeler: Yanlış Veri Enjeksiyonu, Siber Saldırılar, Fazör Ölçüm Birimleri, Akıllı Şebekeler, Derin Öğrenme.

Tarih:05.08.2024

ABSTRACT

DEEP LEARNING AND MACHINE LEARNING METHODS FOR DETECTING FALSE DATA INJECTION CYBER-ATTACKS ON SMART GRID PHASOR MEASUREMENT UNITS

Compared to regular power systems, smart grids offer various advantages. However, they also present significant cybersecurity challenges, particularly in terms of data integrity and security. Phasor Measurement Units (PMUs) play a crucial role in addressing these challenges by providing real-time monitoring and management of the grid through synchronized, high-resolution measurements, thus improving its reliability. Unfortunately, hackers can exploit PMUs by manipulating the vital information they provide. False data injection attacks, which rely on incorrect data input into PMUs, pose a serious threat as they can compromise the efficiency and reliability of the entire power grid. Therefore, secure operations in smart grids depend on effective false data detection and mitigation. In this research, we focus on developing a comprehensive solution powered by both machine learning and deep learning methods to recognize false data intrusion attacks on smart grids. This involves utilizing supervised machine learning models trained on power system attack-labeled data. Tree-based techniques such as Random Forests and Extra Trees are initially employed for binary classification, categorizing PMU data as either natural or compromised. Subsequently, multi-class classification is performed to identify no events, natural events, or active attacks. Furthermore, we explore deep learning architectures like Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and CNN-LSTM hybrids, which are specifically designed for time-series data and can effectively detect and capture subtle temporal patterns. The primary objective of this research is to thoroughly evaluate the state-of-the-art machine and deep learning techniques on real PMU data, leading to a significant improvement in attack detection accuracy and ultimately protecting the security and resilience of critical energy supply infrastructure.

Keywords: False Data Injection, Cyber-attacks, Phasor Measurement Units, Smart Grids, Deep Learning.

Date: 05.08.2024

SYMBOLS

Measurement-related

x_{mal}: State variable estimation using malicious sensor measurements

τ: Threshold value used in bad data detection

a: Attack vector injected by the attacker

c: Estimation error injected by the attacker

H: Measurement matrix representing the system's topology and line impedance

m: Number of measurements

n: Number of buses or locations within the power system

r: Measurement residual

x: Estimated state vector obtained through state estimation techniques

z: Actual measurements collected directly from sensors without preprocessing

z_a: Measurement vector containing malicious measurements

Waveform-related

ϕ: Phase angle

w: Angular frequency

X(t): Sinusoidal waveform at time (t)

X_i: Imaginary component of the phasor

X_m: RMS magnitude of the waveform

X_r: Real component of the phasor

E(t): Energy consumption at time (t)

I(t): Total information flow at time (t)

$I_{C\rightarrow U}(t)$: Information flow from consumers to utility at time (t)

$I_{U\rightarrow C}(t)$: Information flow from utility to consumers at time (t)

L(t+k): Load forecast at time (t+k)

Machine learning-related

tree $(x; t)$: Prediction of decision tree t for input x

$f(x)$: Prediction of the Random Forest for input x

T: Set of decision trees in the forest

$T_i(x)$: represents the prediction of the $i-th$ decision in extra Trees Classifier

N: the number of trees in the extra Trees Classifier

Neural network-related

$\hat{y}$: Output or classification result of the neural network model

$a^{(l+1)}j$: Activation of neuron (j) in layer (l+1) of a neural network

$b^{(l)}j$: Bias term for neuron (j) in layer (l)

f: Activation function applied to the weighted sum of inputs

S: Representation of spatial features extracted by CNN

$Y_{i,j}$: Output of the pooling layer at position ((i,j))

LSTM-related

$\tilde{c}_t$: Candidate cell state in LSTM

c_t: Cell state in LSTM at time (t)

f_t: Forget gate activation in LSTM

h_t: Hidden state in LSTM at time (t)

i_t: Input gate activation in LSTM

o_t: Output gate activation in LSTM

Power System Framework-related

BR1 − *BR*4: Circuit breakers

L1, L2: Transmission Lines

P1, P 2: Power Generators, primary sources of electrical power

R1 − R4: Intelligent Electronic Device (IED)

ABBREVIATIONS

ADC Analog-to-Digital Conversion
AMI Advanced Metering Infrastructure
ANN Artificial Neural Network
CDBN Conditional Deep Belief Network
CT Current Transformer
DFT Discrete Fourier Transform
DoS Denial of Service
EMS Energy Management Systems
ETC Extra Trees Classifier
FDIA False Data Injection
GAE Graph Autoencoder
GPS Global Positioning System
GRU Gated Recurrent Unit
HMI Human-Machine Interface
HTI Hypothesis Testing Identification
IED Intelligent Electronic Devices
LNRT Largest Normalized Residual Test
ORNL Oak Ridge National Laboratory
PCA Principal Component Analysis
PDC Phasor Data Concentrator
PMU Phasor Measurement Unit
PT Potential Transformer
RF Random Forest
ROCOF Rate of Change of Frequency
RTU Remote Terminal Unit
SCADA Supervisory Control and Data Acquisition
SCOPF Security-Constrained Optimal Power Flow
SLG Single Line-to-Ground
SMOTE Synthetic Minority Over-sampling Technique
WAMS Wide-Area Monitoring System

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

The transition of conventional power networks into smart grids has introduced a new era of efficiency and connectivity, revolutionizing the way we distribute and manage electricity. However, amid these advancements, the transition to smart grids has also brought to light a number of unprecedented challenges, particularly concerning data security and integrity. False data injection emerges as one of the most critical threats among these issues, posing a significant risk to the reliability and efficiency of the entire power distribution network. In response to this pressing concern, this thesis aims to propose sophisticated machine and deep learning models specifically designed for detecting false data injection attacks within smart grids. These models will leverage data collected from Phasor Measurement Units (PMUs).

Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTMs) are two of the state-of-the-art architectures known for their ability to interpret complex datasets. CNNs are renowned for their proficiency in extracting spatial features, which is particularly valuable for detecting anomalies in measurements across the grid, as they can identify patterns in data that occur in close proximity to each other. LSTMs, on the other hand, can provide valuable insights into the dynamic behavior of the power system, as they are capable of capable of capturing temporal dependencies and patterns over time. The CNN-LSTM hybrid model combines these two architectures in a complementary manner to achieve a more comprehensive analysis of both temporal and spatial dimensions, resulting in improved precision and effectiveness in false data injection detection in smart grids.

The research for this thesis begins with a careful examination of the dataset and a binary classification task to distinguish between natural and attack events. From this base, the development of a more complex system with a three-class categorization is described that discerns between attack events, no events, and natural events. Extensive comparison of model performance is done using a variety of machine learning models, such as the Random Forest Classifier and Extra Trees Classifier, in conjunction with a couple of preprocessing techniques: under/over/SMOTE sampling and feature selection.

Furthermore, the thesis discusses deep learning techniques that begin with an in-depth exploration of the architecture of CNN and LSTM, its interaction with different preprocessing techniques, to provide the best possible performance. This study is then extended to a hybrid methodology CNN + LSTM aiming to extract spatial-temporal features from smart grid data.

The model, through recurred optimization and fine tuning, tries to identify false data injection attacks by achieving remarkable accuracy and dependability.

In conclusion, this thesis will significantly contribute to the field by providing reliable and efficient method for identifying false data injection attacks on smart grids. This research will enhance the security and reliability of future energy infrastructure by applying cutting-edge deep and machine learning techniques to smart grids.

2 BACKGROUND

The digital age has transformed the renewable energy industry. Smart grids are gradually replacing traditional power grids that have provided electricity supply over long periods. These modern power systems can potentially change everything about electricity production, distribution and utilization due to their advanced information communication technology.

A crucial component of these intelligent grids is the Phasor Measurement Unit (PMU). The time-synchronized, high-resolution power wave measurements provided by PMUs are essential for monitoring and controlling the dynamic properties in power systems.

Moving to smart grids are confronted by numerous problems just like any other technological transformation. A primary concern is ensuring the accuracy and security of the information involved. Given the critical role of data in the proper functioning of smart grids, any alteration of data could lead to catastrophic consequences affecting total power delivering system effectiveness and reliability.

Among the various threats and risks faced by smart grids, False Data Injection (FDI) attacks are particularly concerning. These attacks involve the introduction of false data into the system, which can lead to incorrect decision making, inefficient operations, and even the total collapse of the grid. The insertion of false data into the system poses a significant risk, potentially compromising the stability and reliability of smart grid operations.

This chapter focuses on the intricacies of smart grids, manipulated data issues, the challenges posed by manipulated data, and the need for robust detection systems. Furthermore, we examine machine learning algorithms such as Random Forest and Extra Trees Classifier, deep learning methods like Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), and hybrid models that combine both, and how they can enhance smart grid security. We will also conduct an extensive literature review. Therefore, the objective of our study is to establish a comprehensive foundation for the development of sophisticated deep and machine learning models targeting false data injection detection.

2.1 ELECTRIC POWER GRID OVERVIEW

In this section, we will examine the traditional electric power grid configuration and how it changes into a smarter grid architecture. We will also delve into such significant components such as Supervisory Control and Data Acquisition (SCADA), Energy Management Systems (EMS), and Advanced Metering Infrastructure (AMI), highlighting how they contribute to enhanced grid resilience, efficiency, and reliability.

2.1.1 Traditional Electric Power Grid

A power system is a collection of electricity-producing stations, transmission lines, transformers, and distributors/relays required by various consumers: commercial enterprises large and small, factories, and residential units. Currently, power generation plants are located at a distance from consumers, connected to grids via very long-distance transmission lines. Power flows through these lines, reaching distribution points after undergoing switching and processing, and ultimately reaching end users through communication channels. The term "electric grid" refers to the unidirectional flow of power from generators to final consumers. Over several decades, the discovery that enabled different utilities to interconnect in order to enhance the overall reliability of the power system by compensating for the unexpected power cuts and breakages from power sources such as generators and transmission lines, was made.

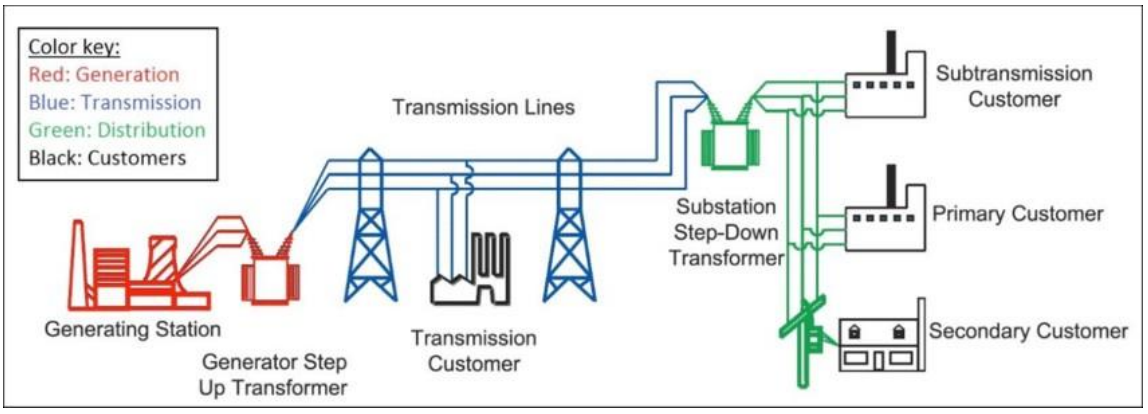

Figure 2.1: Overview of typical electric power grid

For an electric grid to function properly, there must be proper synchronization of power distribution, transmission, and generation. The current state of the electric grid can be classified into four main components: generation, transmission, distribution, and customers. These components can be visualized in Figure 2.1, adapted from Koščo, Tauš, and Šimon (2019).

To produce electricity, hydroelectric dams, coal-fired power plants, wind and solar farms are used as sources of energy. Electric generators are often located away from cities due to

security, legal concerns, and the desire to promote economic efficiency, among other factors. As a result, electrical grids must utilize power lines to ensure the distribution of electricity.

Transferring power from transmission lines to consumers is referred to as distribution. A typical electrical distribution system begins at transmission substations and ends at customer meters (Bosich et al., 2023). It consists of medium-voltage power lines (below 50 kV), substations, and transformers. A substation is composed of a busbar for dividing the electricity into separate areas, relays, circuit breakers, and step-down transformers. Circuit breakers are used to isolate the substation from the power grid or from various distribution lines when necessary. Different areas can receive electricity from the same transmission substation at varying voltages, and the power may be further reduced in multiple steps to approach 7200 V.

Every customer site utilizes a transformer to reduce the voltage. The connection between a house's energy meters and a transformer is achieved using two cables, each carrying 120 volts. Both types of electrical systems, 120-volt and 240-volt consumption appliances can thus be used here due to the 180° out of phase nature of their generated 240-volts.

2.1.2 Smart Grid

The lack of automated analysis and situational awareness in the existing electric power infrastructure renders it outdated and ill-suited for the twenty-first century's rapidly growing electricity consumption. For instance, over the past 20 years, both the demand and consumption of electricity have increased by 2.5% annually in the United States (U.S. Department of Energy). Moreover, the transportation and electrical sectors are significant contributors to greenhouse gas emissions and global climate change.

Smart grids are equipped with advanced electricity networks, featuring sensors, communication systems, and control units for the purpose of maintaining power distribution sustainability and efficiency.

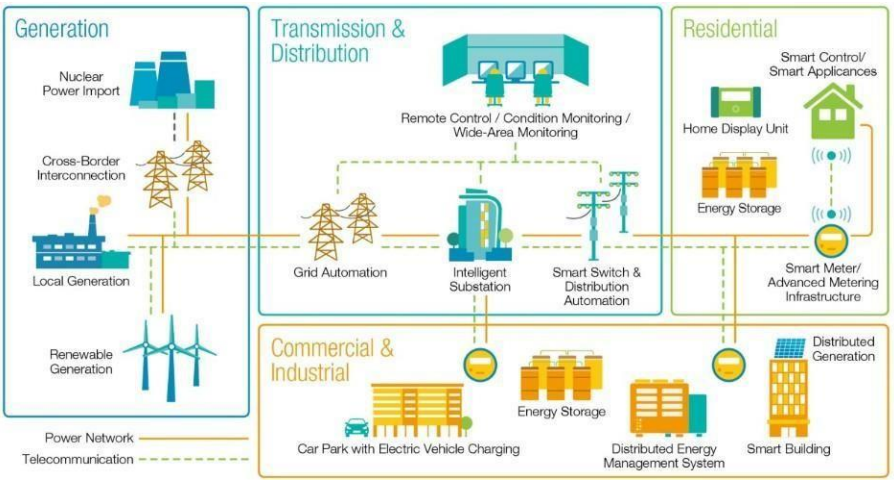

Figure 2.2 : Smart Grid Technology overview

Furthermore, Figure 2.2 illustrates how automated control and state-of-the-art communication technology enable the integration of alternative and renewable energy sources including plug-in hybrid electric vehicles, solar electricity, wind energy, biomass power generation, tidal power, small hydro power plants (Gungor et al., 2010).

A two-way communication system that facilitates information exchange between power companies and customers is the foundation of smart grids. This bidirectional connection enables real-time monitoring, control, and optimization of the electrical system, helping utilities adapt quickly to demand variations, integrate renewable energy sources, and implement demand-side management plans (Rezgui et al., 2012).

The following formula describes the bidirectional transfer of data in smart grids:

$$I(t) = IU\,ßC(t) + IC ßU\,(t) \qquad (2.\ 1)$$

where $I(t)$ is the time of total information flow, $I_UC(t)$ is the information flow from service provider to consumers, and $I_CU(t)$is the information flow from utility consumers.

Sophisticated networks of power may include internets including progressed metering techniques (AMI), supervisory control and data acquisition systems (SCADA), and energy management systems (EMS) in order to work towards better productivity, resilience and sustainability.

2.1.3 Advanced Metering Infrastructure

To enable two-way communication between utilities and customers, AMI is a technology that integrates data management systems, communication networks, and smart meters.

Smart meters are advanced types of electric meters which are capable of collecting data on a regular basis and sharing it, providing comprehensive information about energy usage for both utilities and consumers. AMI systems offer deeper insights into energy consumption while at the same time making it possible for utilities to charge for energy used with great accuracy based on Automatic Meter Reading (AMR), Remote Meter Monitoring and Control functions of electricity meters (Rathor, S. K., Saxena, D. (2020)).

The Figure 2.3 illustrates the architecture of an AMI system, demonstrating how smart meters connect home devices to a network management system. This system transmits energy usage data to utility companies, enabling accurate billing and effective energy management (Chalmers, C., Hurst, W., Mackay, M., & Fergus, P. (2015)).

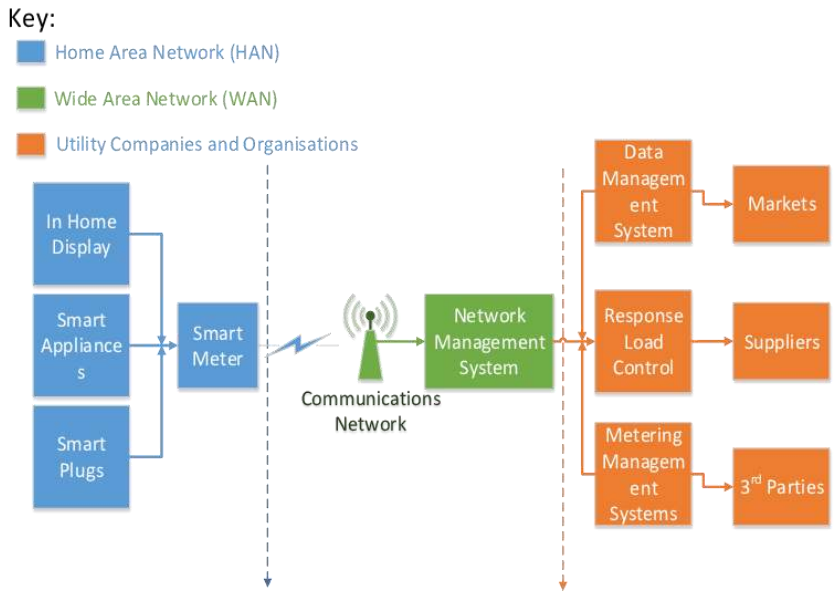

Figure 2.3: Advanced Metering Infrastructure (AMI)

Smart meter data can be displayed in the form of time series. Time series are systematic recordings of energy consumed (Fang et al., 2012). This frequency can be as high as once per hour or even once every 15 minutes. The next time series data can be expressed as

$$E(t) = [E(t1), E(t2), E(t3), \ldots, E(tn)] \quad (2.2)$$

where $E(t)$ represents the energy consumption at time t, and (t_1, t_2, t_3...t_n) the time intervals at which energy consumed is recorded.

2.1.4 SCADA Systems

SCADA systems play a very critical role in smart grids, allowing central monitoring and management of electricity systems. These systems have the capability of collecting information from various sensors, meters, control devices located at different points of the grid, such as power plants, overhead transmission lines, and substations in real-time (Gungor et al., 2013).

By processing and analyzing this data, SCADA systems allow operators to monitor the grid's operational status and remedy faults or disturbances even from remote locations. Grid operators gain a better insight into the power system through the use of SCADA systems, which enhances reliability, efficiency, and situational awareness within the electrical grid, as depicted in Figure 2.4 (Chalmers, C., Hurst, W., Mackay, M., & Fergus, P. (2015)).

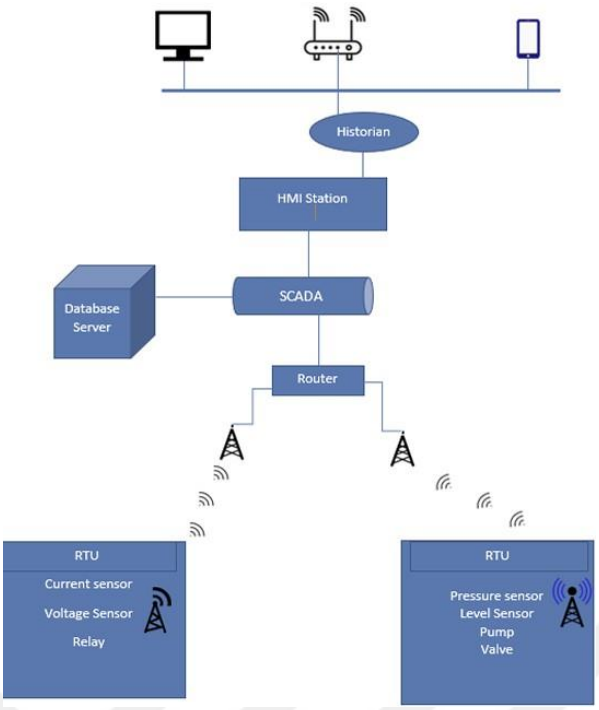

Figure 2.4: The Supervisory Control and Data Acquisition Architecture

The primary components of a SCADA system are:

Remote Terminal Units (RTUs): These devices connect measurement units or sensors dispersed throughout the grid, convert their signals into digital data, and transmit the data to the supervisory system. Additionally, they receive digital orders from the supervisory system and transmit them to the sensors.

Human-Machine Interface (HMI): This enables a human operator to monitor and interact with the grid's status by displaying data in a comprehensible format. Operators can use the HMI to request data from the data acquisition server.

Software Service Historian: This component uses all time-stamped information and events to create a database, which is then used to visually display power trends via HMI.

Supervisory System (Control Center): This system transmits control directives to the system via communication infrastructure and collects data regarding the state of various grid participants. Through these subsystems, SCADA gathers data from the RTUs and then forms and combines it to facilitate decision-making functions for the supervisory system, such as adjusting RTUs. The historian can also receive data in order to enable trends and other analytical audits.

Early SCADA systems, which were typically restricted to open networks like the Internet, were less distributed and more isolated, leaving the system open to various cyber

security threats. SCADA should therefore have security measures that allow its operations do not endanger society. Similarly, in the existing Similarly, analytical instruments like state estimators are crucial in modern SCADA systems (Ymeri et al., 2022).

However, the process of state estimation is susceptible to both cyber and physical security threats due to the predominantly unencrypted data transfer, making it easier for hackers to exploit. This vulnerability can enable significant attacks, such as FDI attacks.

In SCADA systems, the state (x) is estimated by solving a set of nonlinear equations based on the collected measurements (z). The following is its mathematical expression:

$$z = h(x) + e \tag{2.3}$$

where z denotes measurement vector, which can represent current, voltage or power while x represents the state vector, such as bus voltages and phase angles. The nonlinear mapping $h(x)$ links state variables with measurements, indicating potential measurement errors.

2.1.5 Energy Management Systems

Energy Management Systems (EMS) software may be used to optimize power system performance and operation. By integrating various data sources, such as market data, SCADA systems, and weather predictions, these systems enable decision-making processes for electricity generation, transmission, and distribution. EMS applications often utilize functions such as load forecasting, unit commitment and economic dispatch, contingency analysis and security-constrained optimal power flow (SCOPF). EMS leverage advanced algorithms and optimization strategies to facilitate the control of energy resources by the utility, reduce operating costs, and ensuring safe and reliable supply (Saldaña- Gonzále et al., 2020).

The load forecasting feature of EMS systems predicts future power usage by analyzing past data and various variables, including weather conditions, seasonal changes, and economic indicators (Miceli, R. (2013)). Regression analysis can be used to formulate this problem, where the load at time $t+k$ is estimated using historical load data and other relevant characteristics.

$$L(t + k) = f(L(t), L(t - 1), \ldots, L(t - n), X(t), X(t - 1), \ldots, X(t - m)) \tag{2.4}$$

where $L(t)$ represents the load at time t, $X(t)$ denotes the set of additional features (e.g., temperature, day of the week) at time t, and f is the regression function learned from historical data.

2.1.6 Conclusion

These technologies work together to allow the smart grid's enhanced functions. With AMI systems, customers can communicate with utilities and access precise information about power

consumption. SCADA systems supervise and control operations, while EMS applications optimize their functioning to ensure that electrical energy is continuously supplied reliably and efficiently. The use of these technologies enables improved grid management, enhanced awareness of conditions both inside and outside of the grid, as well as the incorporation of various distributed energy resources like energy storage solutions and renewable energy (Zhou et al., 2020).

2.2 PHASOR MEASUREMENT UNITS

Enabling real-time control and monitoring of smart grid systems are essential advanced measuring tools known as Phasor Measurement Units (PMUs). PMUs provide high precision as well as accuracy time-synchronized measurements of voltage and current phasors, frequency and other relevant power system characteristics. Consequently, a Wide Area Monitoring System WAMS is created by methodically placing this equipment at various points in the power supply. (Phadke et al., (2017)).

This section covers the key components and processes involved in PMU operation, from input signal acquisition to output signal generation, including data transmission and processing.

2.2.1 PMU Functionality and Operational Processes in Smart Grids

As shown in Figure 2.5 (Lal, M. D, & Varadarajan, R. (2023)), the following subsections provide a detailed explanation of how PMUs function and operate in modern power grid systems. The underlying concept is the representation of sinusoidal waveforms as phasors. PMU measurements are Charles's Steinmetz concept, where the complex angle and complex modulus represent the phase and amplitude of the cosine wave, respectively at a specific point in time. According to the mathematician, a sinusoidal waveform is represented as follows:

$$X(t) = X_m \cos(\omega(t) + \phi) \tag{2. 5}$$

Refer to Equation (2.5), the phasor form can be rewritten as:

$$X = \frac{X_m}{\sqrt{2}} e^{j\phi} = \frac{X_m}{\sqrt{2}} (\cos\phi + j \sin\phi) = X_r + jX_i \tag{2. 6}$$

where

- $\frac{X_m}{\sqrt{2}}$ is the waveform's RMS magnitude.
- X_r and X_i are the complex number's corresponding real and imaginary.

Current Transformers (CTs) and Potential Transformers (PTs) are used to measure current and voltage levels within the power system, respectively, providing input to PMUs. To ensure accuracy,

these instruments reduce high-potential impulses to tolerable levels. In addition, anti-aliasing filters are employed by PMUs for the removal of any high frequency components existing in analog measurements before further processing. This filtering protects against aliasing since it ensures that the sampled signals accurately capture forms (Lal, M. D., & Varadarajan, R. (2023)).

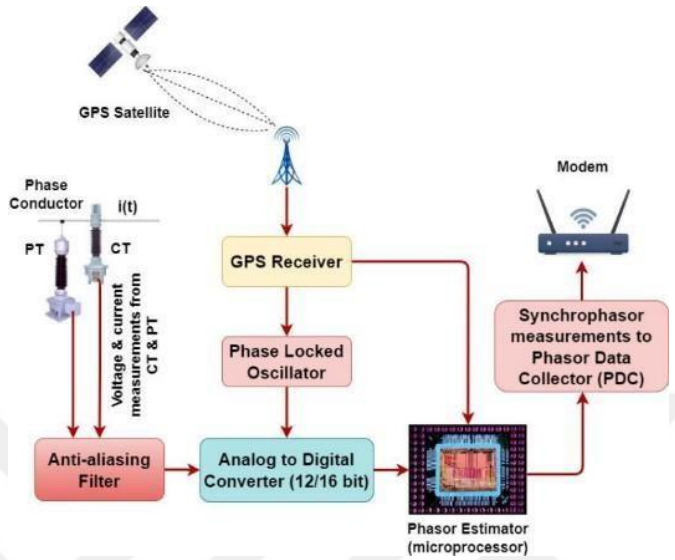

Figure 2.5: Functional Block Diagram of PMU

By utilizing the Global Positioning System (GPS) or other time synchronization sources, PMUs achieve precision time-stamping within a few microseconds. This precise temporal synchronization allows one to perform coherent and coordinated measurements at a large number of points thus making it easy to completely describe the dynamic behavior of the power system.

PMUs convert filtered analog signals to digital format by means of an analog-to-digital conversion process, referred to as ADC. By processing these converted signals, accurate phasor estimation can be achieved under conditions of synchronization with GPS-based timing marks. However, observing the entire grid at once is difficult due to the absence of GPS.

The phase estimator, which is essentially a microprocessor, uses sophisticated signal processing algorithms to estimate the phasor values from the sampled volt- age and current waveforms. These methods include the Discrete Fourier Transform (DFT) and other digital filtering techniques. Because PMUs sample at rates generally between 48 and 256 samples per cycle, effective phasor estimation is possible even when transient events or harmonic distortions are present. The PMU output includes local estimations as well as monitoring of the frequency and rate of change of frequency (ROCOF).

The phase estimator, essentially a microprocessor, makes use of sophisticated signal processing calculations to estimate phasor values from sampled voltage and current waveforms. These methods include the Discrete Fourier Transform (DFT) and other digital filtering

techniques. Since PMUs usually sample at between 48 and 256 samples per cycle, then it means that we can have exact estimates of phasors in presence of transient phenomena or harmonic disturbances. The PMU generates real-time estimates and tracks the frequency and its rate of change.

2.2.2 Phasor Data Concentrator

PMUs transmit phasor measurements to a Phasor Data Concentrator (PDC) through communication networks that use specialized or secure communication channels, as illustrated in Figure 2.6. Receive values of incoming phasor data for analysis on quality and accuracy can be examined by the PDC. These values aid in analyzing the status of a power system in real-time (Mohanta et al., 2016). Super PDCs collect remote PDC and PMU phasor data for centralized viewing. Access to data gathered is enabled by a centralized database which is connected to the Super PDC. Following this data collection and processing phase, power state estimation is the subsequent critical step in monitoring power systems (Sundararajan, A., Khan, T., Moghadasi, A., & Sarwat, A. I., 2019). State estimators are essential for the estimation of the power system state variables, as implied by their name. The comprehensive analysis of the system's status results in greater precision in evaluating the power system.

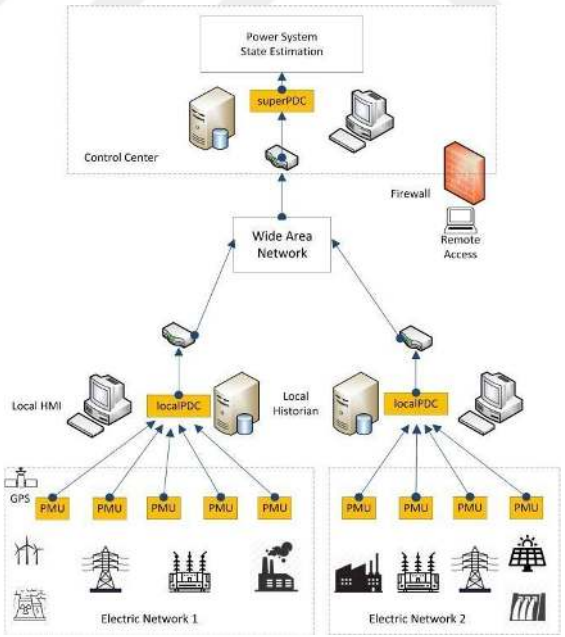

Figure 2.6: Phasor Data Concentrator Architecture

2.2.3 The Vital Role of Phasor Measurement Units

PMUs are vital if you want to avoid blackouts which are more likely to occur during power shortages. Voltage instability is caused by imbalances within the three main components of

energy flow within the power network. When machines are operating, it is essential to ensure they run in step with each other. This smooths grid operation as connected machines can synchronize as needed by sharing both phase angles and frequency (Ding et al., 2021).

Furthermore, PMUs enable the extensive scope of oversight which advantageously accrues to power systems operation engineers where they are able to benefit from extensive monitoring. This allows engineers to make informed decisions to prevent disasters, ensure service reliability and network resilience through continuous provision of point on real-time data from multiple locations. For these reasons, PMUs are critical for maintaining grid stability and preventing catastrophic events (Wang, W., Lu, Z. (2013)).

2.2.4 The Strategic Placement of Phasor Measurement Units

In the context of a conventional utility power system, which consists of the distribution, consumption, transmission, and generating sectors, each part has a specific but interconnected function. The transmission system is one of the most notable of these; enabling large-scale power transfer over great distances by operating at high voltages, typically between 138 and 1000 kV. Despite its vital role, the transmission network faces significant risks; blackouts, for example, can pose serious operational threats (Phadke et al., 2008).

As illustrated in Figure 2.7, phasor measurement units (PMUs) represent a significant technological advancement in power systems, primarily applied at the transmission level. Even though many electric utilities have already integrated PMUs into their grid infrastructures at high voltage substations, monitoring of high voltage transmission lines is also part of the implementation roadmap. However, the distribution processes have distinct features and operational contexts that require complementary measurement methods, if an efficient monitoring instrument has to be made to serve their purposes.

As previously discussed, the strategic deployment of PMUs at the terminals of transmission lines offers a fundamental approach of monitoring voltage and power. Such operational principle is all-inclusive and makes it more expeditious to acquire pertinent information about system dynamics in order to enable electric utilities proactively manage their activities with a view towards enhancing them (Zanni, L. (2017)).

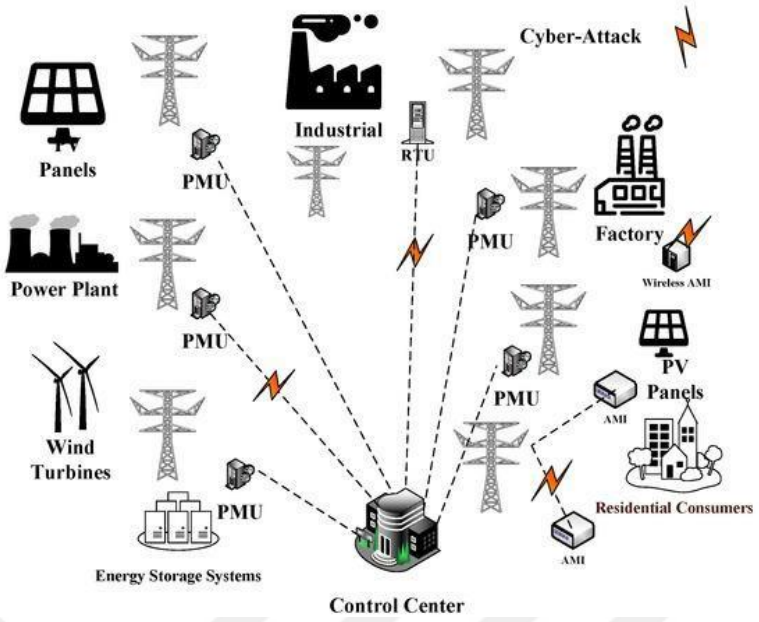

Figure 2.7: Strategic Placement of PMUs in Transmission Networks

2.2.5 PMU's advantages

Now that we have a better understanding of phasor measuring units (PMUs), it is worthwhile to examine their advantages over the conventional measurement technologies.

1. **Wide-area visibility:** PMUs provide a synchronized, system-wide overview of the power grid, enabling the monitoring and analysis of transient disturbances that extend over different areas.

2. **High-resolution data:** The continuous PMU data flow helps to monitor power system status in real-time, making easier and rapid detection and reaction to abnormal conditions and emergencies.

3. **Real-time monitoring:** The constant flow of PMU data makes it possible to track the status of the power system in real-time, which makes it easier to identify and react quickly to anomalous circumstances or emergencies.

4. **Improved situational awareness:** PMUs provide grid operators with comprehensive information about the grid's status, enabling better decision-making and control activities.

5. **Advanced applications:** PMU data help carrying out a number of advanced tasks such as energy structure simulation, event analysis, dynamic stability assessment, and state estimation (Saldaña-Gonzále et al., 2020).

Essentially, PMUs are crucial for modernizing power grid operations because they

provide accurate, timely, and thorough data for control, analysis, and monitoring. Due to the associated benefits, the grid has become more resilient, efficient, and reliable as it is dealing with changing operational issues and dynamic grid conditions.

2.2.6 Conclusion

However, the deployment of phasor measurement units has also introduced new challenges, including security against cyber threats, communication issues and data issues such as malicious data injection. The vast amount of detailed data obtained by PMUs is very large hence, adequate processing capacity and storage facilities are required to accommodate the overwhelming information flow (Biswal et al., 2023). Potential vulnerabilities in communication networks and PMU devices could allow malicious actors to tamper with or inundate them with false data, compromising the data measurement reliability and accuracy. To ensure the confidentiality, integrity, and consistency of PMU data, especially during critical grid operations, robust cybersecurity measures are essential (Mohanta et al., 2016).

2.3 FALSE DATA INJECTION ATTACKS

Incorporating state-of-the-art monitoring and control technologies has significantly enhanced the performance and reliability of electricity grids. However, the increased interconnectivity has introduced security issues, as there is now a greater risk of False Data Injection (FDI) attacks originating from this scenario. These FDI attacks aim to introduce false or manipulated measurements into the sensor data sets collected by equipment such as PMUs, compromising the data integrity within the power grid (Liu et al., 2011).

This section covers a wide range of topics related to FDIA in power systems, including various attack categories targeting PMUs, potential attack targets, the potential impacts of an attack, and the limitations of conventional detection methods.

2.3.1 False Data Injection Attacks in Power Systems

Attackers who use FDI aim to disrupt the normal operation of the power grid by injecting false or corrupted data into measurements made by sensors, such as PMUs. To mitigate these risks is essential for ensuring the safety and reliability of electric power infrastructure (G. Hug and J. A. Giampapa, 2012).

PMU security flaws can be categorized into four main attack classes:

1. **Interruption:** Attacks known as denial of service (DoS) assaults, which employ system resources to deny access to authorized users, or physical attacks, such cutting a cable, can cause interruptions in communications be- tween PMUs and

other networked devices.

2. **Interception:** Attacks, such as man-in-the-middle attacks or passive attacks like packet analysis, that include snooping on information being transferred between PMUs and other devices.
3. **Modification:** Attacks that tamper with or modify data while it's being transported between PMUs and other devices, like injecting malicious code into network traffic or changing data while it's being transmitted.
4. **Fabrication:** Attacks involving the creation or insertion of false information into the network, such as spoofing attacks that generate falsified PMUs or other devices, or attacks that inject false data into the network.

Because false data assaults modify data inside the grid without triggering immediate alerts, they are difficult to identify. False data injection is a sophisticated threat that seeks to undermine information integrity, in contrast to overt assaults like physical damage or denial of service attacks (Liu et al., 2011). A successful false data injection can have severe consequences, including erroneous decisions, operational disruptions, and potentially even physical infrastructure damage. Attackers can significantly compromise grid dependability by focusing on data integrity and taking advantage of flaws in smart grid monitoring and control systems (Liang et al., 2017).

2.3.2 Impact of FDI Attacks on Smart Grids

The reliable and safe running of smart grids may seriously and extensively hinder due to False Data Injection (FDI) assaults. Attackers can lead the energy management system (EMS) and control systems into error changing the state estimation process, which is a crucial part of power system monitoring and control, possibly leading to a number of negative outcomes (Liang et al., 2017):

1. **Incorrect Operational Decisions:** FDI assaults can lead to the abuse of EMS working capabilities, emanating from inadequate load shedding quality, insufficient generation dispatch or switching errors, which causes inappropriate application. In turn, the system can become more unreliable, more expensive to correct or have potential damage to equipment; all of which may result in operational failure (Deng et al., 2017).
2. **Line Overloads and Equipment Damage:** An FDI attack can damage the state estimation process so that it cannot identify or accurately forecast situations of equipment stress or line overloads hence leading to unobserved overloading of

transformers, transmission lines and other major equipment that can cause cascade failures or irreparable impairment (Ashok et al., 2018).

3. **Grid Destabilization and Blackouts:** By creating large inaccuracies in the state estimation process, FDI assaults have the potential to disrupt the whole power system in extreme situations. The control systems may get infected with these flaws, resulting in improper control actions that might lead to voltage or frequency instability and consequently widespread blackouts or power outages (X. Liu and Z. Li, 2019).
4. **Situational Awareness Compromise:** On the integrity of the measured data, which may compromise the situational awareness of the grid operators. If not detected and reacted promptly, this could intensify these challenges (Stoustrup et al., 2019).
5. **Cascading Effects:** Aside from the local power system, foreign direct investment attacks could have impact on other interdependent critical infrastructure. This may include water utilities, transportation networks and communication networks among others and may lead to cascading failures and wide spread disruptions.

Furthermore, FDI intrusions pose a significant threat due to their ability to operate undetected, evading traditional bad data detection systems and altering reality systems for extended periods. Consequently, the electricity system is at risk of permanent and lasting damage in terms of both operation and infrastructure being interfered with (Kosut et al., 2011). To mitigate the impacts trended upon by the FDI attacks, there is need for a multi-modal strategy which entails robust cyber security measures, advanced detection and mitigation technologies (such as machine and deep learning approaches) and resilient system design. The primary objective of this ongoing study is to develop effective guard mechanisms for these sophisticated malwares so that they can enhance both resilience and safety for a smart grid (Y. Zou et al., 2016).

2.3.3 Attack Target Layers in Smart Grids

By manipulating sensors, controllers, and the smart grid remote control units, attackers provide incentives for false data that has an impact on power flow calculations, control decisions, among other things inside a system that may lead to equipment failure on the grid and consequently total power network shutdown in severe scenarios. This could also result in significant financial losses and jeopardize the network's integrity. Figure 2.8 depicts an FDIA and its architecture.

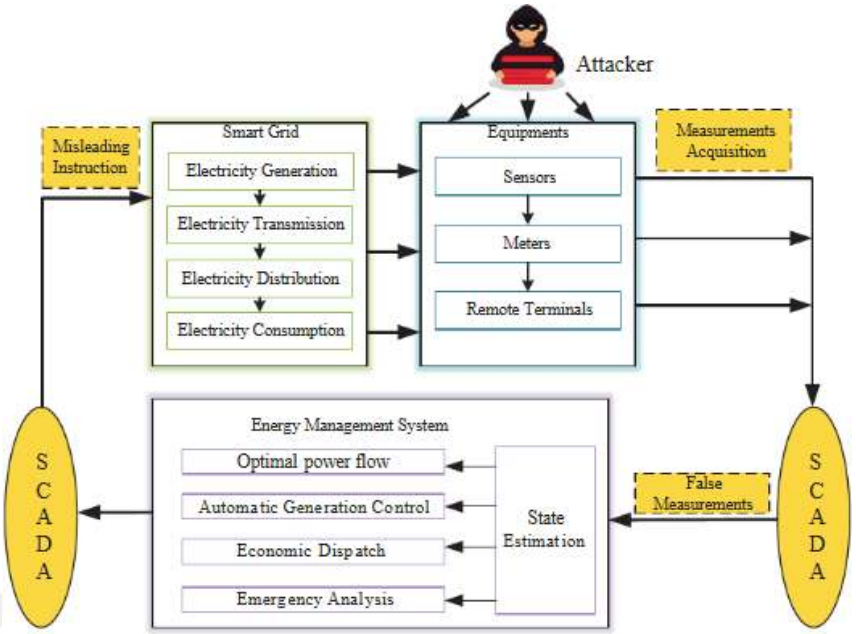

Figure 2.8: Attack Target Layers in Smart Grids

The measurement layer might be attacked in the following ways:

1. **Equipment Layer:** Consisting of sensors, meters and remote terminals, this stratum is responsible for collection and transmission of measurements from several points of smart grid infrastructure. In order to manipulate or inject false data directly at the source, the attacker may focus on these devices.
2. **Measurement Acquisition:** The equipment layer's measurements are attained and evaluated by the EMS. Attackers are capable of trying to break this layer in 2 ways: the introduction of false data, manipulation result measurement during processing or transmission.
3. **Energy Management System (EMS):** The EMS is important in doing things like ensuring that production is controlled automatically, calculating the best ways through which energy flows as well as determining what could happen if things go wrong with regards to power usage. For these purposes, a considerable amount of precision measurements and evaluations of conditions are required. Therefore, efficiently asserting control over the process in this level can be catastrophic apart from its financial implications due to possible erroneous functioning of the system following wrong state predictions and inappropriate control responses.
4. **Control and Data Acquisition (SCADA):** The SCADA system monitors and coordinates smart grid operations. A SCADA system breach allows at- tackers to conduct illegitimate control operations based on a modified system state triggered by an FDI attack if indirectly associated with such attacks.

2.3.4 Traditional Bad Data Detection in Power Systems and Their Limitations

Techniques like the hypothesis testing identification (HTI) approach and the largest normalized residual test (LNRT) are conventional methods of identifying and eliminating gross measurement errors in power systems so as to improve the accuracy of their state estimation (Yang et al., 2013). Nevertheless, with the injection of spurious figures, effective complex FDI attacks can occur irrespective of the use of these methods meant to detect poor information.

The basic idea behind LNRT approach is to use the leftover from the state estimation problem to separate out any bad data and is founded on the assumption that measurement error can be described by Gaussian distribution. However, FDI attacks might be able to generate attack paths positioned on the null space of measurement matrix which may entail small residuals qualifying corrupt data detection test.

On the other hand, the HTI approach employs a hypothesis testing process to help repeatedly locate and exclude some few faults in the data it receives. It assumes that the proportion of faults is small in comparison to the number of measurements taken which may not always hold, especially when there are multiple sensors or meters attacked simultaneously as part of a coordinated FDI attack. As a result of these constraints, advanced methods have been developed to detect and prevent FDI attacks in smart grids, by such techniques as machine learning and deep learning.

2.3.5 Methodology of Traditional Bad Data Detection

A methodical comparison between the actual measurement collected from sensors and the measurements predicted by a mathematical model based on the assumed state of the system (Hx) is used in bad data detection. The measurement residual *r*, which is determined by the difference between these two sets of measurements, is as follows:

$$\mathrm{r} = \mathrm{z} - \mathrm{H}\hat{x} \qquad (2.\ 7)$$

comparing the residual magnitude with a threshold value (τ), ((z - Hx^ > τ)), deviations beyond the threshold may be identified as possibly erroneous data. This can be done by assessing the residual magnitude using techniques such the (|z − Hx^|) L2 norm (Jawhar, J. (2020)).

where

- z: represents the vector of original measurements.
- H: is a m × n Measurement matrix representing the system's topology and line impedances.
- m: stands for the quantity of measures.

- n: represents the number of buses or locations within the power system.
- x: Estimated state vector obtained through state estimation techniques.
- r: Measurement residual, indicating the difference between actual and estimated measurements.
- Threshold (τ): This is a predefined value that represents the maximum allowable difference between the actual measurement and the predicted measurement.

The threshold test, which compares the measurement residuals magnitude with a predetermined threshold, is an essential part of BDD. When the residual surpasses this threshold ((z - Hx^ > τ)), it indicates the possibility of possibly false measurements.

2.3.6 Bypassing Bad Data Detection

FDI Attacks target the flaws of the state estimation technique especially when an attacker knows the system matrix H linking measurements from various sensors with state variables such as voltage magnitudes on particular buses. If someone has data about H already, then they may insert incorrect values into state estimation without setting off any alarms (Jawhar, J. (2020)).

Scenario

- Assumption: The attacker is already aware of the system matrix H, which contains details on the topology and line impedance of the power system.
- The measurement vector containing malicious measurements is given by:

$$za = z + a \quad (2.8)$$

where the initial measurement vector, denoted by z = (z1...zm), and the attack vector, denoted by a = (a_1,..., a_m), which might be any non-zero random vector.

When an attacker hacks the meter and substitutes the initial measurement z with $z + a$, where a stand for the attack vector, the state variable's estimation using the malicious sensor measurement z_a is as follows:

$$\hat{x}mal = \hat{x} + c \quad (2.9)$$

where c estimation error of the attacker is made equal to 'a'. When attacker's attack vector 'a' is made equal to system matrix H multiplied by attackers's estimation error c, L2 norm of vector of measurements with malicious data becomes equal to that of original measurements (Jawhar, J. (2020)):

$$||(za - H\hat{x}\,mal)|| = ||(z - H\hat{x}||) \quad (2.10)$$

The following equations explain how this is possible:

$$||(za - Hx\hat{}mal)|| = ||z + a - H(x\hat{} + c)|| = ||z + a - Hx\hat{} - Hc||) \quad (2.11)$$

Substituting a = Hc, we get:

$$||(za - Hx\hat{}mal)|| = ||(z - Hx\hat{})|| \leq \tau \quad (2.12)$$

Thus, the False Data Injection Attack can easily bypass the bad data detection test without getting detected.

2.4.8 Conclusion

To sum up, false data injection attacks seriously threaten reliability and safety of power systems, leading to the wrong decisions, interruptions and physical damage to infrastructure that have been made because of altering grid data by attackers. Attackers exploit vulnerabilities in state estimate algorithms as well as measurement infrastructure to manipulate grid information in such a way that it can mislead controlling and surveillance systems.

2.4 MACHINE LEARNING METHODS FOR FALSE DATA INJECTION DETECTION

A supervised learning model works by learning from data that has labels or known outcomes. It uses this labeled data to make predictions or decisions about new, incoming data. On the other hand, unsupervised algorithms do not make use of any labels offered by anyone; instead, what they do is to direct their attention towards finding underlying structures present in every single point set available (Alloghani et al., 2020). Moreover, this will keep operations stable.

Detecting false data injection attacks in smart grids requires important use of supervised learning approaches since they can learn on the labeled data representing false data injection attack and normal conditions. It is only after being taught that these algorithms will label any other new data other than the training ones as either abnormal or normal (maybe represent an attack). Random Forest and Extra Trees Classifier stand out as two of the most popular supervised learning approaches, a more elaborate discussion of which follows below.

2.4.1 Random Forest

The Random Forest method models entire sets of data using a collection of decision trees. This collection of trees forms the core of the Random Forest model. During the training phase, it creates numerous decision trees and outputs the mean prediction or the mode of classes for each tree. (Rigatti, S. J. (2017)).

Construction of Decision Trees

Every tree in Random Forest is built from the preliminary training sample. And then, for each node to test, provide a random sample of k features from the feature-set so that each decision tree may choose the best feature to use a mathematical criterion (typically the Gini Index) to segregate the data.

Feature Importance Calculation

During the creation of the Random Forest, features are selected by calculating how much each feature improves the model. This is done by measuring the reduction in error or the Gini Index for each feature, which then determines the importance of that feature.

Information Gain Calculation

The decision criteria used will be Information Gain. The formula for information gain is:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \quad (2.13)$$

where

- S is the set of all samples
- A is an attribute
- Sv is the subset of S for which attribute A has value v
- Entropy(S) is the entropy of set S

Entropy Calculation

The formula for calculating the entropy is:

$$Entropy(S) = -\sum_{i=1}^{c} p_i \log_2 p_i \quad (2.14)$$

where c is the number of unique class labels and p_i is the proportion of rows with output label i. There are many benefits of Random Forests for False Data Injection (FDI) Detection, including resistance against noise, scalability to massive datasets, and ability to capture complex feature dependencies. The application of Random Forests can be used for more secure smart grids, and can point out false data injections via their specific trends by combining predictions across multiple decision trees.

2.4.2 Extra Trees Classifier

The structure of the Extra Trees Classifier is based on decision trees, one of the ensemble's learning methods. It builds a large number of trees (decisions) by training and outputs a class that forms the mean prediction (regression) or mode of the classes (classification) of individual trees. Building Decision Trees, the original training sample is used to build each decision tree in the Extra Trees classifier. The decision trees then have to choose the optimum feature to split

the data based on some mathematical criteria, usually the Gini Index, given a random sampling of k features from the feature-set at each test node.

1. **Random Selection of Splitting Value:** The key unique feature of additional trees is the aspect of having a splitting value for a feature being randomly selected by the forest algorithm. The algorithm selects a split value randomly instead of using Gini or entropy to split the dataset and computing a locally optimal value. Thus, the trees lead to diversity and no correlation.

2. **Feature Importance Calculation:** In order to select features employing the previously mentioned forest structure, during the forest's development, the normalized overall diminution in the mathematical requirements used for determining which features to split (or, if the Gini Index is utilized during the construction of the forest, its value) is calculated for each feature. The feature's Gini Importance is the name given to this value.

3. **Information Gain Calculation:** The decision criteria used will be Information Gain. The formula for Information Gain is:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \quad (2.\ 15)$$

Entropy Calculation

The formula for calculating the entropy is:

$$Entropy(S) = -\sum_{i=1}^{c} p_i \log_2 p_i \quad (2.\ 16)$$

where c is the number of unique class labels and p_i is the proportion of rows with output label i.

Key Features

The Extra Trees Classifier offers several key features that distinguish it from traditional Random Forests:

- Random split points: Instead of finding the perfect split for each node using optimization algorithm, extra trees randomly select split points. Noise resistance grows if more fluctuations are incorporated into the decision tree.
- Fast Training Times: Extra Trees can be trained more quickly than Random Forests because of the random split point selection, especially for larger datasets. Because of the decreased processing weight, it is a desirable option for applications that need

rapid model deployment.

- Ensemble Learning: Extra Trees Classifier as well as Random Forests involves ensemble learning because it combines predictions from a number of decision trees classifiers. Therefore, this ensemble technique enhances both robust- ness and generalization performance that allow it to be employed in difficult classification problems.
- Parallelization: Multiple processor core parallelization is made possible by Extra Trees' separate decision tree generation mechanism. The technique is more salable because of its parallel processing capacity, which also shortens training periods.

Anomaly, intrusion, and false data injection attack detection are just a few of the uses for the Extra Trees Classifier in smart grid security scenarios. Analysis of complicated datasets gathered from smart grid sensors and devices is made possible by its capacity to handle high-dimensional data and tolerate noisy characteristics.

2.4.3 Conclusion

Random Forests and Extra Trees Classifier offer protection against malicious data manipulations on smart power grids through applying artificial intelligence methods such as machine learning (artificial neural networks etc.) to them which makes it possible not only acquiring knowledge about abnormalities detection but also decreasing the risk level (by looking back at its history; this helps isolate core aspects that formed their basis)

2.5 DEEP LEARNING FOR FDIA DETECTION

Deep learning focuses on the use of deep neural networks that are one of the many types of models in machine learning, in detecting False Data Injection Attacks (FDIA) on smart grids, with promising results. Due to the automatic pattern and representation extraction in deep learning models, they are perfect for that difficult job. Among others, several architectures for deep learning have been studied specifically for the purpose of FDIA identification and include convolutional neural networks (CNN), Long Short-Term Memory (LSTM), and their variations (Sa'ad et al., 2023).

2.5.1 Convolutional Neural Networks (CNNs)

Convolutional neural networks are helpful when assessing spatial interconnectivities in power measurements because they can extract local and spatial features much better. They can also detect the presence of FDIA by identifying anomalies or patterns in measurement data

(Ayad et al., 2018).

In a CNN model, the input data progresses through multiple layers. Each layer aims to extract complex traits on a stepwise basis within advance, before the final classification stage. CNNs consist of two main components: convolutional layers and pooling layers. When it comes to the filters that are used for extracting specific features from input data in convolutional layers, pooling layers are responsible for reducing data dimensionality by downsampling convolutional layer's output. The output is unfolded and entered into a series of dense layers where it is used to classify or regress representations that are generated from a large number of convolutions and pooling stages as shown in Figure 2.9 (Phung, V. H., & Rhee, E. J. (2019)).

Let x be an input measurement vector $x \in R^n$, where n is the number of measurements. One possible representation of the convolutional operation is as follows.

$$a_j^{(l+1)} = f\left(\sum_i x_i * w_{ij}^{(l)} + b_j^{(l)}\right) \quad (2.16)$$

- $a^{(l+1)}_j$ represents the activation of neuron j in layer $l + 1$.
- f is the activation function applied to the weighted sum of inputs.
- $x_i * w^{(l)} + b^{(l)}$ represents the weighted sum of inputs to neuron j in layer l.
- x_i is the output of neuron i in the previous layer.
- $w^{(l)}$ is the weight connecting neuron i in layer l to neuron j in layer $l + 1$.
- $b^{(l)}$ is the bias term for neuron j in layer l.

In the context of CNNs, this equation represents the output of a neuron after applying convolutional filters to the input data x. The operation $x_i * w(l)$ denotes the convolution between the input data x and the filter $w(l)$, followed by the addition of bias term $b(l)$. The activation function f is typically a nonlinear function such as ReLU (Rectified Linear Unit) applied element-wise to the result of the convolutional operation.

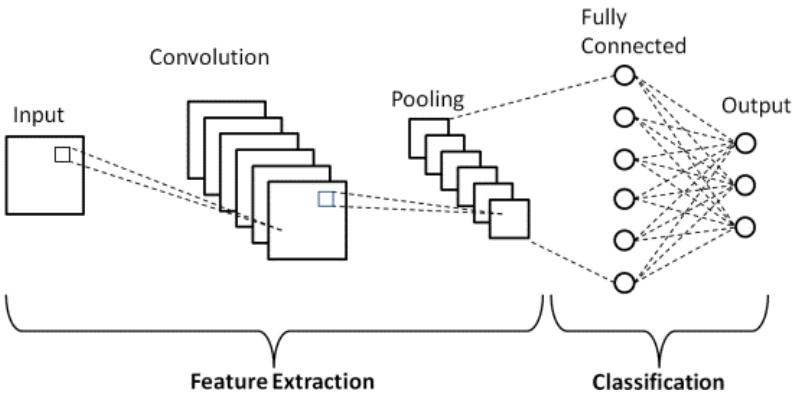

Figure 2.9: CNN model architecture

CNNs require pooling layers to downsample the output of convolutional layers while preserving essential features. Dimensionality reduction in data helps maintain essential information for labeling during classification and subsequently optimizes computational performance.

Typically, pooling layers carry out operations like average or maximum pooling. The maximum value within each area is substituted for the output value in each region of the input feature map during the maximum pooling process. On the other hand, average pooling determines the average value for each region in the input feature map (Schmidhuber, J. (2015)).

Let us denote the output of the convolution layer as X, where X represents a feature map that is $m \times n$ matrix. By way of illustration, in max pooling, output of pooling layer Y can be calculated as:

$$Y_{i,j} = \max_{p,q}\left(X_{p,q}\right) \tag{2. 18}$$

where i and j index the rows and columns of the pooled feature map, and p and q index the corresponding regions in the input feature map X.

Similarly, for average pooling:

$$Y_{i,j} = \frac{1}{k \times l} \sum_{p=i}^{i+k-1} \sum_{q=j}^{j+l-1} X_{p,q} \tag{2.19}$$

where k and l represent the size of the pooling window.

The final layers of a CNN are called fully connected layers, or dense layers. They use the features extracted from the preceding layers to conduct classification or regression tasks, which allow them to discriminate between normal and attacked scenarios.

The output of the fully connected layer can be expressed as:

$$a^{(l+1)} = \mathrm{f}\left(w^{(l)} * x^{(l)} + b^{(l)}\right) \tag{2.20}$$

where

- $a^{(l+1)}$ represents the activation of neurons in layer $l + 1$.
- f is the activation function applied to the weighted sum of inputs.
- $x^{(l)}$ is the output of neurons in layer l.
- $b^{(l)}$ is the bias vector for neurons in the layer l.

2.5.2 Long Short-Term Memory (LSTM) Networks

LSTMs have found widespread applications in natural language processing, anomaly detection, time-series prediction, and other tasks involving sequential dependence because they are particularly excellent at modeling sequences, which was designed to address the vanishing gradient problem encountered in traditional RNNs. This new type of RNN architecture is known as Long Short-Term Memory (LSTM) networks (Van Houdtet al., 2020).

A collection of gates that control the information flow and memory cells make up LSTM networks as illustrated in Figure 2.10 (Chevalier, G. (2018)). An LSTM cell's essential components are as follows:

1. **Forget Gate**: Controls the extent to which previous cell state information should be retained or forgotten. It takes as input the previous cell state c_{t-1} and the current input x_t, and it outputs a forget gate activation f_t between 0 and 1.
2. **Input Gate**: The input gate (i_t) and the candidate cell state $\tilde{c}_t$ determine how much new data should be retained in the cell state. The input gate decides which values to update, while the candidate cell state generates potential new values.
3. **Candidate Cell State**: The candidate cell state $\tilde{c}_t$ is calculated from the prior cell state c_{t-1} and the current input x_t using a calculation involving hyperbolic tangent (tanh) function.
4. **Cell State Update**: The update to cell state c_t is the result of various combinations including forgetting gate output f_t, as well as input gate output i_t coupled to candidate cell $\tilde{c}_t$.
5. **Output Gate**: Controls the extent to which the cell state should influence the output. It takes as input the previous cell state c_{t-1}, the current input x_t, and the current hidden state h_t, and outputs an output gate activation o_t between 0 and 1.

LSTM cell functioning at time step t is determined by the following equations:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \textit{ Forget gate} \quad (2.21)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \textit{ Input gate} \quad (2.22)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \textit{ Cell state} \quad (2.23)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \textit{ Output gate} \quad (2.24)$$

$$h_t = o_t \cdot \tanh(c_t) \textit{ Hidden state} \quad (2.25)$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \textit{ Candidate cell state} \quad (2.26)$$

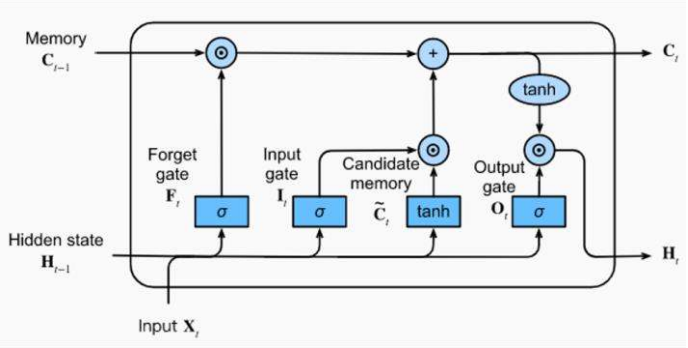

Figure 2.10: Architecture of a LSTM Unit

where, W_f, W_i, W_c, W_{ho} are weight matrices, b_f, b_i, b_c, b_o are bias vectors, σ is the activation function of the sigmoid and $[h_{t-1}, x_t]$ represents the concatenation of the previous hidden state h_{t-1} and the current input x_t.

In light of gate activations, the LSTMs update the cell state and hidden state respectively so it can learn long range dependencies from the observation data. Considering the fact that FDIA attacks can take time, this can be used to detect compressive sensing.

2.5.3 Hybrid Architectures

LSTM networks and CNN are often combined in hybrid designs for FDIA detection in order to take advantage of the temporal and spatial information found in power system data. While the CNN component focuses on spatial characteristics, the LSTM component captures temporal relationships (Aksan et al., 2023).

The LSTM network captures long-term dependencies in the measurement data by selectively updating the hidden state and the cell state using gate activations. Equations (3.21) -(3.26) govern the key LSTM cell equations. The LSTM processes the input data step by step. At each time step, when the gates are activated, it updates the hidden state and the cell state. The final hidden state stored in h_T can possess temporal dependencies of data. The output of the LSTM network, h_T, can be reshaped and fed into a CNN to extract spatial features. Spatial characteristics can be extracted from the CNN using the LSTM's output.

The output sequence of the LSTM network is represented as **H**, where **H** = **H** = [h_1, h_2,...,h_T]. This sequence can be reshaped into a tensor of shape (*T*, W, H, C). The LSTM net's output sequence is usually represented by **H** where **H** = [h_1, h_2...h_T]. It is possible to transform this sequence into some tensor (*T*, W, H, C) where:

- *T* represents the number of time steps (length in terms of sequence).
- W and H denote width and height for an element located on a feature map.
- *C* refers to the quantity of channels accounting for that in the feature space.

The tensor H is reformatted and then fed forward into CNN for the purpose of extracting spatial features. FCNN stands for convolutional layers of this CNN whereas f_p stands for its pooling layers. S signifies CNN's output which captures the spatial characteristics derived from the final LSTM output h_T. The process can be mathematically represented by:

$$S = Pl\big(FCNN(H)\big) \tag{2.27}$$

In Equation (2.27), the FCNN first takes in the rearranged LSTM output, followed by the pooling layers (P_l), which provide further processing and extract the spatial characteristics represented by S.

Once the CNN has retrieved the spatial characteristics, they can be classified by passing through a fully connected layer, which is represented by f_F *C*. Assume that the output vector of the fully connected layer is represented as y^:

$$\hat{y} = fFC(S) \tag{2.28}$$

The expected outcome or classification result, in such an instance would be represented by y^, which is dependent on the spatial features as extracted from the LSTM output h_T and passed through the CNN and fully connected layer.

2.5.4 Conclusion

Even though deep learning methods have shown good results in the identification of FDIA, yet there are certain problems that hit back such as needs for huge varied training sets and computational complexity as well as the interpretability of deep learning models.

2.6 LITERATURE REVIEW

FDIA detection has gained significant attention in the field of smart grid security due to its potential to compromise the reliability and integrity of power systems. To overcome this difficulty, several strategies that make use of deep and machine learning models have been introduced. In this literature review, an overview of current research and its contributions to deep learning-based FDIA detection in smart grids is provided.

In recent years, the increasing deployment of smart grids in power systems has highlighted the pressing necessity for effective security mechanisms aimed at averting false data injection attacks (FDIAs). In their work, Keçeci et al. (2023) propose a mixed deep neural network structure together with federated learning-based techniques which offer a decentralized manner in which to detect these attacks without compromising user privacy. According to comprehensive simulations that evaluate a range of electric power grid system topologies, they've successfully localized FDIAs with their technique. By making use of federated learning

that is achieved when training models over decentralized data sources without affecting their privacy, Keçeci et al.'s model is able to ensure a robust detection performance while preserving data privacy.

Following this, a recent technique by Takiddin et al. (2023) employs graph auto-encoders (GAE) for detecting anomalies with high detection rates across various system topologies, efficiently capturing spatiotemporal information about power networks. This approach finds compact representation for complex power system topologies using GAE. Their methodology enables the accurate identification irregularities which may be a sign of data injection attack frauds are being enabled by their methodology. This demonstrates adaptability and efficiency in various system topologies making it suitable for real-world applications ensuring smart grid security.

Similarly, according to Yang et al. (2023), they were able to increase the speed at which they could identify FDIA by using a combined detection model that was composed of principal component analysis (PCA) and CNN leading to increased precision. By merging PCA for dimensionality reduction and CNNs for feature extraction in their model, spatial patterns within power system data is well understood. This enabled them to accurately identify of anomalous behavior indicative of false data injection attacks. Their method demonstrates how useful it is to use complementary approaches to improve detection performance in smart grid settings.

Sikri and Singh (2023) came up with a way of spotting false data employing recurrent neural networks. The technique includes the use of entities referred to as EMSs resembling neural networks in form but vary greatly in purpose for detecting spurious data that have been inserted in electric power system collection with fraudulent intent in an attempt to raise accuracy levels. The data they presented is valuable because it offers information about where false news manipulation lies from a space and time perspective. This will improve the state of emergency preparation at operation level for grid connected devices and how threats can be avoided.

Regev et al., (2022) made a hybrid model of an AI-based anomaly detection for detecting abnormalities in PMUs which is significant because it makes it possible for the model to pinpoint all anomalies in both measurements from PMUs and measurement errors. Therefore, it allows the introduction of CNNs and LSTMs for detecting deviations within PMUs measurements as well as its abnormalities. This later extends to detecting likely cases where false data has been inserted before its discovery. This enables you to spot likely malicious attempts to inject false data at an earlier stage. augments the security posture of power systems by providing real-time surveillance, as well as detection capabilities, which reduce the adverse

consequences of attacks by foreign organizations on grid operations.

In their work, Basumallik et al. (2019) has introduced a data filter that uses Convolutional Neural Network (CNN) to enhance the protection of Phasor Measurement Unit (PMU) based State Estimators. The subject is focused on recognizing unusual data trends using many PMU voltage and current datasets based on a CNN-based filter with Nadam update and cross-entropy loss. This method stands out due to its uniqueness which involves exclusion of false data by the use of multivariate time series in the process of creating features before state estimate operations start thus adding an additional layer of security.

The method is successful at using many sources of data together via attention mechanisms, thus allowing for thorough examination as well as detection of false information planted into smart grid systems during these attacks. Wu et al. (2023) introduce an efficient multi source self-attention data fusion approach for FDIA recognition that demonstrates improved detection accuracy and resistance against different possible assault scenarios on FDIA recognition.

Moreover, they suggest a false data injection attack detection technique based on CNN-GRU, which notably enhances security for power grid systems. This approach makes it possible to accurately identify malicious data injections by combining convolutional neural networks (CNNs) and gated recurrent units (GRUs) for studying the spatial and temporal features of power system data. Moreover, enhancing scalability and robustness using deep learning methods on the detection technique shields power grid systems against constantly changing cyber threats.

This study by Xu, Li and Sun (2021), introduces a new cell type that is more powerful than other methods including Support Vector Machines (SVM), or Artificial Neural Networks (ANN) for detecting FDIA disease condition, with the support of deep learning-based techniques on Conditional Deep Belief Networks (CDBN) model. They make an acceptable distinction between normal and abnormal behavior of power systems by utilizing the layer structure and bottom-up strategy involved in CDBNs, improving detection of false data injection attacks. This approach can be applied successfully in today's real-world smart grids.

Xu, Li, and Sun (2021) have also proposed a technique for detecting attacks through machines in which they apply random forest classifiers thus achieving an impressive level of matching across various IEEE node systems. Ensemble learning techniques are employed driving detection tools that are able to pick out complicated patterns characteristic of false metering incidents hence being able to identify abnormal activities in power systems without any false information. The efficiency and adaptability of their suggested approach in improving the security of smart grid infrastructures are highlighted by the performance that has been

shown in a variety of system configurations.

Last but not least, Tuyizere and Ihabwikuzo (2023) bring to light the useful applications of their model in the support of electrical grid operators; they propose a machine learning oriented approach for identifying possible cyber-attacks, as well as those leading to power system disruptions. By using machine learning algorithms, their methodology can enable the detection of anomalies and cyber threats in live streaming power system data hence promoting proactive measures against them by enhancing initiative preventive measures and improvement for the entire system. Integrating their method into operational frameworks can help to make smart grid infrastructure more secure and reliable by giving system operators the information they need just in time.

All these studies emphasize the use of advanced machine and deep learning techniques in order to make smart grids more secure against FDIAs as well as cyber threats which is a promising way of improving the security and resilience of power systems as they continue to face changed cyber-physical threats.

2.7 CONCLUSION

Since the advent of the digital age, the energy sector has undergone significant transformation due to the emergence of smart grids. These advanced power systems have the potential to revolutionize power generation, transmission, distribution and usage through the application of cutting-edge information and communication technology. PMUs are very important in the operation of smart grids, providing time-synchronized, high-resolution measurements of electrical waveforms, which are essential for observing and managing dynamic power system responses.

Smart grid adoption presents challenges, particularly in data security and integrity, as incorrect decisions and system failures can be occasioned by compromised data. This makes FDI a major threat to the intelligence and effectiveness of smart grids. Therefore, reliable detection systems are indispensable for thwarting FDI attacks on smart grids.

This chapter has explored the complicated nature of smart grids, brought attention to the problem of manipulated data, and emphasized the need for efficient detection systems. Furthermore, we have examined how deep and machine learning methods, particularly Random Forest, Extra Trees Classifier, LSTMs and CNNs, could improve smart grid security. As suggested in this thesis, our goal in creating this comprehensive foundation is to facilitate the creation of advanced deep and machine learning models for FDI detection.

3 METHODOLOGY

This chapter outlines the comprehensive approach taken in this research to address the critical problem of identifying false data injection attacks in smart grids. The attractiveness of smart grids as attack targets has grown due to their use of PMUs for high-resolution real-time measurement. This approach aims to counter such threats and enhance safety and reliability of essential power distribution systems.

3.1 PROBLEM STATEMENT

PMUs are essential components of today's smart electricity networks since they provide time coordinated, high-quality electrical measurements such as voltages or currents. This valuable information enables us to observe what is happening in real-world operations, enhancing the reliability of our power systems. In contrast, since PMUs are very important in attacking the power system (false data injection attackers) the monitored data (PMU data) is highly critical for context-specific cyber-attacks (Ding et al., 2021).

PMUs can be compromised through physical or cyberattacks to tamper with measurements so that operational systems and management centers get false data. By releasing incorrect information into PMUs, grid operators may be misled, potentially leading to erroneous control measures or, in severe cases, widespread blackouts. For intelligent power transmission systems to maintain security, stability, and sustainability, preventing such attacks is essential.

We provide a complete methodology that employs machine learning and deep learning techniques to address the problem of identifying false data injection attacks in smart grids. Our strategy will utilize supervised learning models trained on labeled Power System Attack datasets from Oak Ridge National Laboratory and Mississippi State University. These datasets will allow for thorough training and evaluation of our proposed models since they provide realistic simulations of grid operations under various scenarios.

Initially, we will conduct binary classification analyses to differentiate between PMU data that is natural and compromised that has been the targeted by false injection attacks. Based on these results, we will expand our study to include multi-class classification settings, where three classes will be distinguished: no events, natural events, and active attack events, utilizing the capabilities of powerful machine learning models such as Random Forests and Extra Trees Classifiers.

Furthermore, in the context of time-series data, which is the subject of our investigation, there are architectures for deep learning such as CNNs and hybrid models which incorporate CNNs and LSTM networks. The techniques from deep learning have been said to potentially be able to exploit intricate temporal relations together with complex characteristics in the PMU data streams.

The ultimate objective of this research is to perform an extensive examination through cutting-edge machine learning and deep learning techniques on real PMU data. By significantly improving the accuracy of attack detection, we aim to enhance the security and resilience of critical energy distribution systems.

3.2 EXPLORATION OF POWER SYSTEM ATTACK DATASETS

The dataset used in this study, developed jointly by Mississippi State University and Oak Ridge National Laboratory (ORNL), comprises measurement datasets that include measurements related to normal, disturbance, control, and cyber-attack behaviors in a simulated power system environment. Measurements in the dataset include synchro phasor measurements and data logs from Snort, a simulated control panel, and relays.

This subsection will present the power system attack datasets, which involve system setup, data set attributes and classification tasks. It serves as a foundation to comprehend the data and how it can be applied in identifying data injection attacks.

3.2.1 Power System Framework Configuration

A schematic layout of the power system framework setup utilized to generate data is shown in Figure 3.1 Phasor Measurement Units (PMUs), control panels, relays, and network traffic monitoring tools (Snort) are the main sources of measurement data (Wang et al., (2019)).

The system's essential elements are:

- **Power Generators (P1 and P2):** Primary sources of electrical power.
- **(IEDs)(R1-R4):** Intelligent Electronic Devices capable of controlling the switching of breakers (BR1-BR4) on or off.
- **BR1-BR4:** Circuit breakers that can be tripped by the IEDs or manually overridden by operators.
- **Transmission Lines:**
  - **Line One:** Spans from breaker one (BR1) to breaker two (BR2).
  - **Line Two:** Spans from breaker three (BR3) to breaker four (BR4).

- **Distance Protection Scheme:** Implemented within the IEDs to trip breakers upon detecting faults, regardless of their validity.
- **Phasor Data Concentrator (PDC):** Responsible for storing, displaying, and recording historical phasor data.
- **System Operation**
  - Each IED (R1-R4) is responsible for controlling a specific breaker (BR1-BR4) automatically.
  - The distance protection scheme triggers breaker operation upon fault detection, without internal validation.
  - Operators have the capability to manually issue commands to trip breakers BR1-BR4 when necessary, such as during maintenance activities.

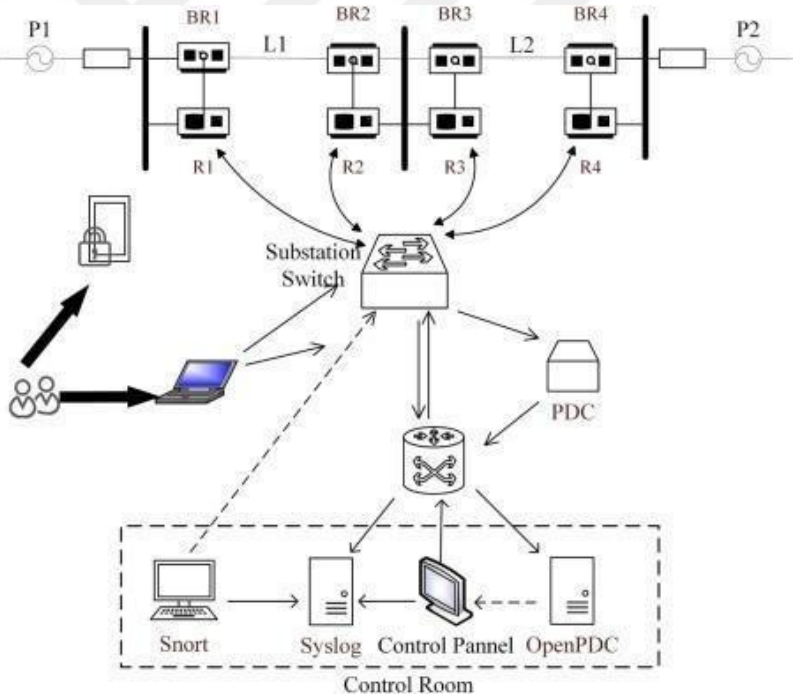

Figure 3.1: Power system framework configuration

3.2.2 Dataset Features

The dataset includes 128 features, as shown in Table 3.1. Each PMU produces twenty-nine different measurements. A synchro phasor is another name for PMU, which is a device in a power system that acts as an accurate clock so that researchers can observe electric waves with precise timing on grids. Four PMUs formed the basis of the dataset used here, each with 29 measurements, resulting in 116 PMU measurement columns. Each column has a naming convention called "R#-Signal Reference," which denotes the type of measurement from a

particular PMU by "R#." code. Below is a list of the signal references along with their corresponding descriptions. For example, "R1-PA1:VH" represents the Phase A voltage phase angle recorded by PMU R1.

Table 3.1 Description of Features

Feature	Feature Description
PA1: VH - PA3: VH	Phase A - C Voltage Phase Angle PM1:
V - PM3: V	Phase A - C Voltage Phase Magnitude
PA4: IH - PA6: IH	Phase A - C Current Phase Angle
PM4: I - PM6: I	Phase A - C Current Phase Magnitude
PA7: VH - PM9: VH	Pos. - Neg. - Zero Voltage Phase Angle
PM7: V - PM9: V	Pos. - Neg. - Zero Voltage Phase Magnitude
PA10: VH - PA12: VH Pos.	Neg. - Zero Current Phase Angle
PM10: V - PM12: V	Pos. - Neg. - Zero Current Phase Magnitude F
	Frequency for relays
DF	Frequency Delta (dF/dt) for relays
PA: Z	Appearance Impedance for relays
PA: ZH	Appearance Impedance Angle for relays
S	Status Flag for relays

The control panel logs, snort alerts, and relay logs from the four combined PMU/relay devices are contained in the 12 extra columns that follow the 116 PMU measurement columns. The marker column, which is the last column in the dataset, contains details on the scenario that each data instance in the dataset represents.

3.2.3 Classification Tasks

The dataset was randomly sampled and divided into binary and multi-class classification tasks.

- **Binary Classification:** Distinguishing between Natural and Attack Events.
- **Multi-Class Classification:** Differentiating among No Events, Natural Events, and Attack Events.

Table 3.2 Description of Scenarios

Scenario No.	Description	Type
41	Normal operation load changes	No Events
1–6	SLG faults	Natural Events
13, 14	Line maintenance	
7–12	Data injection	Attack Events
15–20	Remote tripping command injection	
21–30, 35–40	Relay setting change	

The scenarios included in the dataset are summarized in Table 3.2 and are explained in more detail below:

- **Single Line-to-Ground (SLG) Faults:** These include anomalous voltage, current, and frequency caused by simulated short circuits at different points along the transmission lines.
- **Line Maintenance:** Scenarios in which, for maintenance reasons, one or more relays on a particular line are disabled.
- **False Data Injection Attacks:** Attacks that modify phase angles and produce erroneous sensor data in order to obstruct the power system status estimation process. By altering variables like voltage, current, and sequence components at various points along the transmission lines, the dataset simulates these attacks.
- **Remote Tripping Command Injection Attacks:** Unexpected relay trip directives that target one or more relays are delivered over the communications network from the computer of the attacker.
- **Relay Setting Change Attacks:** attacks that alter the configuration of the relay's distance protection system, making it less responsive to legitimate problems.

3.2.4 Conclusion

Mississippi State University and Oak Ridge National Laboratory (ORNL) provide the Power System Attack Datasets. These datasets mimic real-world power systems' intricate structure since it comprises numerous scenarios ranging from standard operations to organic difficulties and diverse forms of cyberattack behaviors. Its design imitates actual grid complexity through having various types of scenarios such as normal functioning modes; environmental challenges and cyberattacks.

The comprehensive set of functions, including network traffic data, logs from the control panel and data received from various PMUs, provides ample opportunities to develop and evaluate models for recognizing anomalies. To support the in-depth study of suggested approaches, there are binary as well as multi-class tasks differing in difficulty level starting with telling original from disrupted events and up to recognizing several types of events more precisely.

In general, Power System Attack Datasets are a valuable test-bed that enables researchers and practitioners to advance the field of false data injection attack detection. This enhances the safety and resilience of modern-day smart grids.

3.3 BINARY CLASSIFICATION: NATURAL VS. ATTACK

In this section, we explore the challenges and methodologies involved in classifying events within the power system dataset into distinct categories. We will initially address binary classification which is the central question of distinguishing Natural and Attack events in our dataset, Subsequently, we will extend this analysis to a multi-class classification task that includes the No events category. To ensure suitability and cleanliness of our data for classification purposes, we conduct a rigorous initial preprocessing process. These include dealing with any missing values as well as infinite ones while encoding all categorical attributes in a format that can be used during machine learning among other steps.

After the preparation stages, we explore a variety of preprocessing methods designed to improve the classification process. These strategies include undersampling, oversampling, and the Synthetic Minority Oversampling Technique (SMOTE) for addressing the effects of class imbalance and feature selection for determining the most relevant features.

3.3.1 Data Integrity and Preprocessing

We establish the essence of our machine learning case study by ensuring the integrity and precision of our dataset. This involves checking the data for missing entries to identify any anomalies or empty spaces. Subsequently, we verify the presence of infinity values and ensure that all true values are suitable for performing accurate and numerically stable calculations. We also focus on properly encoding class labels, which is an important step in getting our dataset ready for classification tasks. Our goal in carefully preprocessing the data is to build a solid and reliable basis for our ensuing studies and model training.

Handling Missing Values

To ensure data dependability and integrity, it is important to meticulously examine the dataset for missing. Our approach included systematically identifying and rectifying missing data throughout the dataset, making it more suitable for further preparation and analysis.

1. Techniques for Addressing Missing Values

Applying the isnull() function to a data frame determines whether or not there are missing values in the data set. When it is called, this function creates another data Frame whose shape is identical to that of the original; each cell in the frame takes either true or false indicating whether its corresponding cell in the data frame contains NaN (meaning there is missing information). We use isnull() to construct a mask containing only True values where there is missing data's existence. The technique makes it possible to handle missing data better and for us to understand how clean the data is. We use this method to determine the number of missing values in each column of a DataFrame. We apply the sum() function, which, by default, combines disparate columns in a DataFrame, yielding a single series that displays the null values for all columns.

2. Assessment of Missing Data

During the investigation of the dataset for missing values, each of the selected scenarios' columns was examined thoroughly. One promising thing discovered in the study was that there were no absence values for any of the columns in any scenario. This indicates a well-curated dataset with accurate and inclusive observations. The dataset can thus be considered stable, making it suitable for further cleaning operations if necessary.

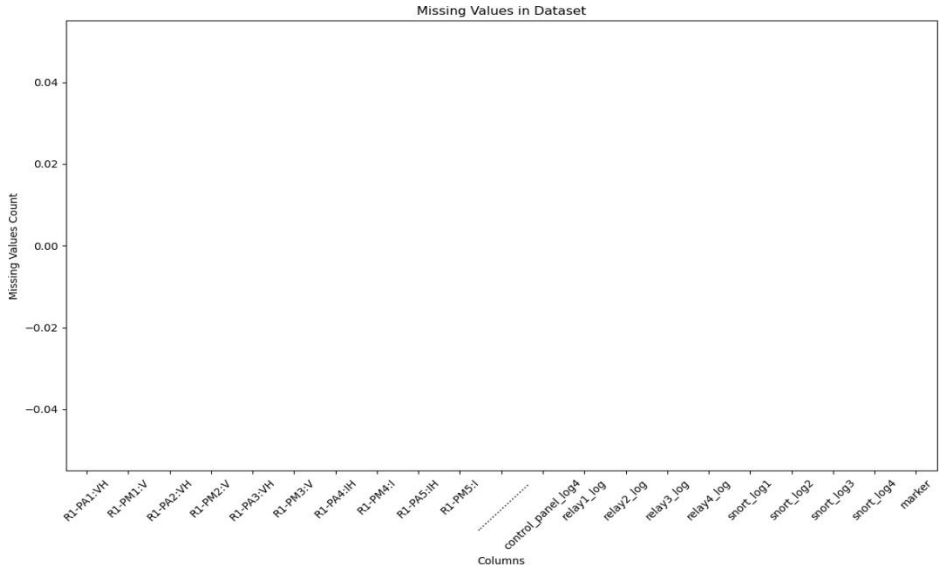

Figure 3.2: Missing Values Count

The missing value detection method was implemented, and a visualization that shows how no missing values are distributed across various columns was created as shown in Figure 3.2. We are assured by this thorough evaluation that the dataset is reliable and ready for further processing in handling infinite values.

Handling Infinite Values

One way to ensure research integrity is by addressing all irregularities or inconsistencies in the dataset as it is prepared for analysis. In machine learning models, infinity is commonly recognized as an abnormal value that can interfere with accuracy. The following outlines how we detect and handle infinity in datasets.

1. **Identification of Infinite Values**

Initially, each column was systematically scanned to identify the presence of infinite values such that we could determine which columns were most affected and develop a feasible solution. A careful examination of the data revealed that columns [R1-PA: Z, R2-PA: Z, R3-PA: Z, R4-PA: Z] contained infinite values.

2. **Consideration of Impedance Measurements**

The infinite values affecting the columns represent the apparent impedance (PA: Z) of each relay. Impedance measurement is vital for identifying power system malfunctions and taking preventive measure. For this purpose, Intelligent Electronic Devices (IEDs) trip breakers using distance protection method if impedance falls below a certain level. As a consequence, assigning a value of zero to infinite impedance measurements would change the base structure of the datasets, hence interfering with the accuracy for future analysis.

3. **Techniques for Handling Infinity Values**

We utilized the SimpleImputer technique with the maximum strategy to address infinite values without compromising data integrity. In this approach, missing values defined as NaN (Not a Number) are substituted by the largest observed value for each column, effectively handling infinity values. NaN values can be identified using the isnull() method in Python, which generates a DataFrame with Boolean values indicating whether each element is NaN. By applying the any() function to this Boolean table, we can examine individual rows for any signs of missing data.

4. Visualization of Infinity Values

To gain a better understanding of the columns affected by infinite values we created a box plot representation. For these columns, the box plot can help to evaluate how data is distributed and identify extreme or outlier values with infinite values in different columns in this instance.

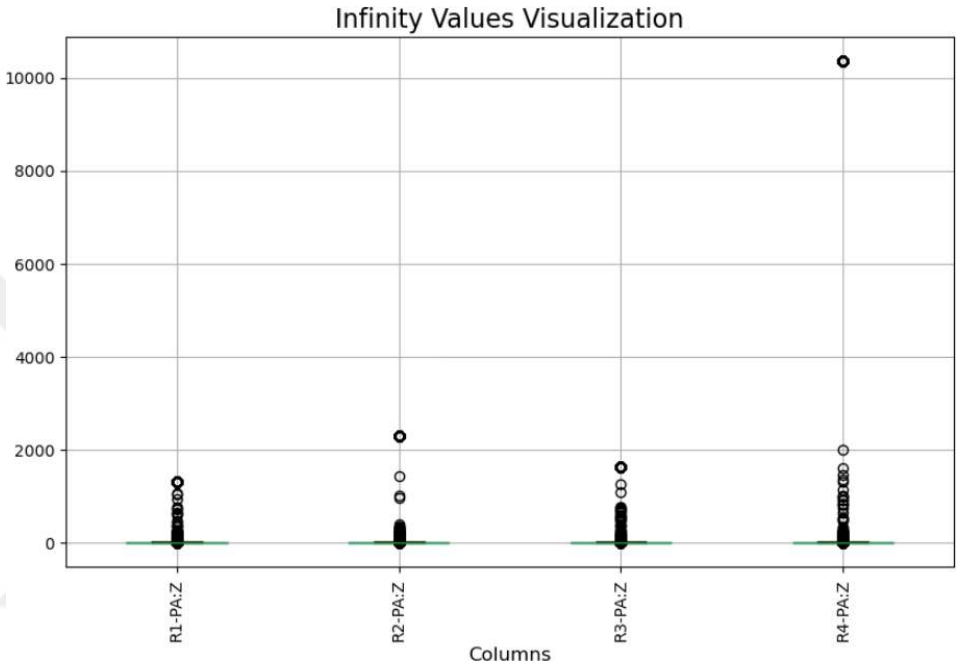

Figure 3.3: Infinity Values Visualization

Figure 3.3 displays a box plot illustration for the [R1-PA: Z, R2-PA: Z, R3-PA: Z, R4-PA: Z] columns following the substitution of infinity values with their respective maximum values. In summary, the box plot displays the distribution of data inside each column, and the replaced infinity values, which are now the maximum values in those columns, are represented by the outliers, or dots.

Label Encoding of Categorical Features

A common technique for transforming categorical data into a format that machine learning algorithms can understand is label encoding. By encoding categorical variables into numerical labels, we enable efficient data processing and analysis. Using this approach, a unique integer label is given to each distinct category in the categorical feature. Converted a categorical attribute referred to

as a "marker" to encode its label. The "marker" column had two distinct levels: "Natural" and "Attack." As depicted in Figure 3.4, these labels comprised the "marker" column in its raw state before encoding.

R1-PM4:I	R1-PA5:IH	R1-PM5:I	...	control_panel_log4	relay1_log	relay2_log	relay3_log	relay4_log	snort_log1	snort_log2	snort_log3	snort_log4	marker
605.91099	-57.003571	626.78553	...	0	0	0	0	0	0	0	0	0	Natural
483.59351	-50.947407	500.98896	...	0	0	0	0	0	0	0	0	0	Natural
483.59351	-50.913030	500.98896	...	0	0	0	0	0	0	0	0	0	Natural
482.86107	-50.437475	499.15786	...	0	0	0	0	0	0	0	0	0	Natural
484.50906	-50.013486	497.69298	...	0	0	0	0	0	0	0	0	0	Natural
...	...	...	...	...	...	...	...	...	...	...	...	...	...
455.57768	166.977090	472.79002	...	0	0	0	0	0	0	0	0	0	Attack
459.42299	166.667693	471.69136	...	0	0	0	0	0	0	0	0	0	Attack
459.60610	166.633316	471.50825	...	0	0	0	0	0	0	0	0	0	Attack
462.53586	166.467158	471.87447	...	0	0	0	0	0	0	0	0	0	Attack
463.08519	166.415592	471.50825	...	0	0	0	0	0	0	0	0	0	Attack

Figure 3.4: "Marker" Column Before Encoding

1. Techniques Used for Label Encoding

LabelEncoder class found in the library scikit-learn facilitates the encoding of labels. In order to fit on the ‘marker’ column values and then transform them into numbers rather than the original label values, LabelEncoder class uses the fit − transform() method. Every different category in the column is given a unique integer value by the LabelEncoder, which starts at 0. In this instance, as seen in Figure 3.5, LabelEncoder assigns 0 to "Natural" and 1 to "Attack" since the "marker" column has two categories: "Natural" and "Attack."

R1-PM4:I	R1-PA5:IH	R1-PM5:I	...	control_panel_log4	relay1_log	relay2_log	relay3_log	relay4_log	snort_log1	snort_log2	snort_log3	snort_log4	marker
605.91099	-57.003571	626.78553	...	0	0	0	0	0	0	0	0	0	1
483.59351	-50.947407	500.98896	...	0	0	0	0	0	0	0	0	0	1
483.59351	-50.913030	500.98896	...	0	0	0	0	0	0	0	0	0	1
482.86107	-50.437475	499.15786	...	0	0	0	0	0	0	0	0	0	1
484.50906	-50.013486	497.69298	...	0	0	0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
455.57768	166.977090	472.79002	...	0	0	0	0	0	0	0	0	0	0
459.42299	166.667693	471.69136	...	0	0	0	0	0	0	0	0	0	0
459.60610	166.633316	471.50825	...	0	0	0	0	0	0	0	0	0	0
462.53586	166.467158	471.87447	...	0	0	0	0	0	0	0	0	0	0
463.08519	166.415592	471.50825	...	0	0	0	0	0	0	0	0	0	0

Figure 3.5: "Marker" Column After Encoding

Data Visualization and Class Distribution

Examine our dataset’s class distribution and develop a data visualization for clarification on the characteristics it has. The analysis about dealing with imbalanced samples in the dataset will guide us on how we should go about this; for successful machine learning modeling.

1. Class Distribution

The dataset is divided into two classes: "Attack" and "Natural". The following is the distribution of classes:

- Attack: 55,663 instances
- Natural: 22,714 instances

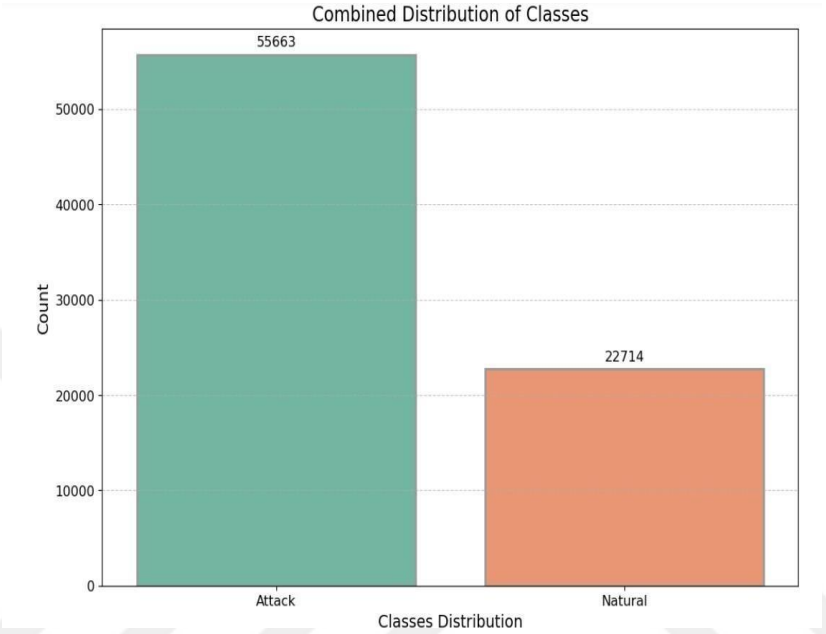

Figure 3.6: Distribution of Classes

The distribution of the classes in this dataset is quite unbalanced as shown in Figure 3.6, with the “Attack” class being more than twice the size of the “Natural” class. The problem with class imbalances is that with many machine learning models there is already an implicit bias towards the majority class, hence making the minority class to be inappropriately modeled resulting in poor performance.

The "Natural" class requires the smart grid system to have examples of valid or typical data measurements, as noted by Kamble et al. (2007). In turn, authenticating and classifying these instances of normal performance instances during false data injection attacks helps preserve the reliability and integrity of grid operations while minimizing the occurrence of false positives that may cause downtime or trigger unnecessary security alerts.

To keep consistency in the smart grid system it remains imperative that the class “Natural” keeps all data observing examples which are genuine or normal like ones. Detecting and labeling such instances of standard behaviors during

false data insertion attacks helps in keeping the grid operations running and monitoring for reliable information without causing too many unnecessary alarms that may disrupt power supply they protect.

We will conduct a comprehensive comparison of training machine learning algorithms on the original unbalanced dataset, and also the under sampled, oversampled and the SMOTE datasets, to determine how effective these approaches are. This method will enable us to assess model performance using original data along with sampling strategies impacts.

3.3.2 Baseline Model on Imbalanced Dataset

We must first train machine learning models based on imbalanced datasets to establish a ground level before applying any sampling techniques; this method will enable understanding about how they perform in cases where there exists a class imbalance as well provide standards upon which to measure impact of SMOTE, under sampling and oversampling techniques. We can determine what complexities and limitations the models face when managing class imbalances through scrutiny of training data belonging to an imbalanced dataset. This will help us effectively utilize sampling methods and demonstrate the potential benefits of addressing class imbalance.

Model training

Two machine learning models, the Extra Trees Classifier and Random Forest Classifier, were trained on the original imbalanced dataset to establish a baseline performance.

1. **Extra Trees Classifier**

For the Extra Trees Classifier, the following steps were taken:

- We initiated by separating the features (X) and the target variable (y) from the original Data-Frame.
- Employing train_test_split() from scikit_learn, we split the data set into training and testing sets with 20% reserved for testing and a random state of 21 for reproducibility.
- An ExtraTreesClassifier is loaded from the scikit_learn library.
- The training set (X_train and y_train) was subjected to the

ExtraTreesClassifier, the model was trained using the *fit* method.

- Predictions were made on the test data (X_test) using the *predict* method, and the predicted values were stored in y_pred.
- The performance of the model was evaluated using the following metrics: accuracy score, classification report, confusion matrix.

2. Random Forest Classifier

A similar process was followed for the Random Forest Classifier. A Random Forest Classifier loaded from the scikit_learn library was initialized with 100 estimators.

- The training set (X_train and y_train) was subjected to the ExtraTreesClassifier, the model was trained using the fit method.
- Predictions were made on the test data (X_test) using the *predict* method, and the predicted values were stored in y_pred.
- The performance of the model was evaluated using accuracy score, classification report, confusion matrix.

The baseline models were trained on the imbalanced dataset to assess their performance and limitations in handling class imbalance. In the next sections, we will explore and utilize undersampling, oversampling, and SMOTE strategies to overcome this problem and improve the models' ability to accurately identify instances from both classes.

3.3.3 Undersampling to Address Class Imbalance

We exploited the Random undersampling strategy from the imbalanced-learn library to deal with the problem of class imbalance and enhance the performance of machine learning algorithms on the minority class. In simple terms, with under- sampling the majority class instances are reduced to match those from the minority class hence making the class distribution balanced.

The undersampling process was carried out as follows:

- We initiated by separating the features (*X*) and the target variable (*y*) from the original Data-Frame.
- Employing train_test_split() from scikit- learn, we split the data set into

training and testing sets with 20% reserved for testing and a random state of 21 for reproducibility.

- The training set (*X*_train and *y*_train) was subjected to the RandomUnderSampler of the imbalanced-learn library with sampling_strategy ='auto', whereby the minority class is automatically identified and subjected to under-sampling so that it matches the number of instances in the majority class.
- The fit_resample() method of the RandomUnderSampler object was used to perform the under-sampling and obtain the resampled features (*X*_resampled) and target variable (*y*_resampled).

- Finally, a new Data-Frame (resampled_df) was created by concatenating the resampled features and target variable, preserving the original column names. The training set has a balanced set of classes after applying the random under sampling method, as seen in Figure 3.7. That illustrates both classes after under- sampling, which is for both classes (Attack "0" and Normal "1") having an equal number of instances which is 18,143 in this case.

Figure 3.7: Under-sampled Dataset Distribution

As a follow-up to under sampled data (resampled_df), we will now be able to construct Extra Trees and Random Forest Classifier models. Evaluate them according to which they excel in an equally distributed category. The training is

conducted in a similar process as explained in Baseline Model on Imbalanced Dataset. The outcomes of the under sampled data will be discussed in the next chapter Findings and Evaluation.

3.3.4 Oversampling to Address Class Imbalance

From the Unbalanced Learning package, we performed the method Random sampling to handle the class imbalance issue and at the same time enhance the efficiency of machine learning models in the under-represented category. This implies that there needs to be an increase in the number of instances of the minority class to reach the same number of instances of the majority class if there are to be any successful attempts at class distribution balancing through oversampling.

The oversampling process was carried out as follows:

- We initiated by separating the features (*X*) and the target variable (*y*) from the original Data-Frame.
- Employing train_test_split() from *scikit learn*, we split the data set into training and testing sets with 20% reserved for testing and a random state of 21 for reproducibility.
- The training set (X_train and y_train) was subjected to the RandomUnderSampler of the imbalanced-learn library with sampling_strategy =′ auto′ for parameters automatically identified the minority class and performed oversampling on that class to match the majority class size. The randomstate = 21 ensured consistent sampling across different runs.
- The fit_resample() method of the RandomOverSampler object was used to perform the oversampling and obtain the resampled features (X_resampled) and target variable (y_resampled).
- A new DataFrame (resampled_df) was created by concatenating the resampled features and target variable, preserving the original column names.

After applying the Random Over-sampling technique, the class distribution in the training set was balanced, as illustrated in Figure 3.8, It shows the combined distribution of classes after over-sampling, where both classes (Attack "0" and

Normal "1") have an equal count of 44,558 instances.

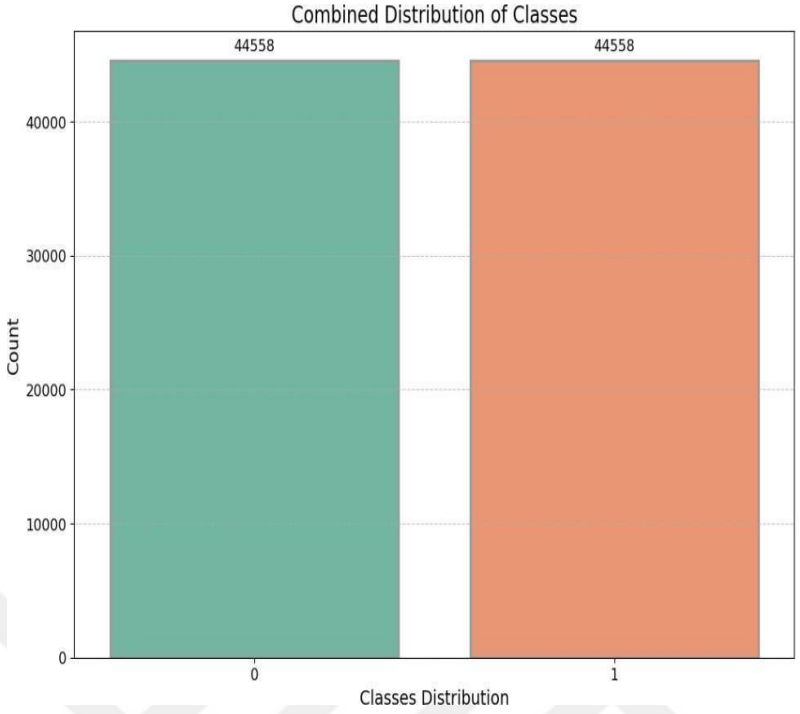

Figure 3.8: Over-sampled Dataset Distribution

We trained the machine learning models on the over-sampled dataset (resampled_df), The training conducted in a similar process as explained in Baseline Model on Imbalanced Dataset. and assessed their performance on the balanced class distribution. The outcomes of training and testing the Random Forest and Extra Trees classifiers on the over-sampled dataset will be discussed in the next chapter Findings and Evaluation.

3.3.5 Synthetic Minority Oversampling Technique to Address Class Imbalance

We utilized the SMOTE from the imbalanced-learn package to enhance the performance of machine learning models on the minority class and solve the issue of class imbalance. SMOTE is a sophisticated oversampling method that creates artificial minority class instances by comparing feature spaces of already-existing minority class examples. The SMOTE process was carried out as follows:

- The features (*X*) and target variable (*y*) were separated from the original Data-Frame
- The SMOTE object from the imbalanced learn library was applied to the training set (X_train and y_train). The sampling strategy =′auto′ parameter automatically identified the minority class and performed

oversampling on that class to match the majority class size. The random-state = 21 ensured consistent sampling across different runs.

- The fit_resample() method of the SMOTE object was used to perform the oversampling and obtain the resampled features (X_resampled) and target variable (y_resampled).

Figure 3.9 shows the combined distribution of classes after applying SMOTE, where both classes ("0" and "1") have an equal count, effectively balancing the class distribution.

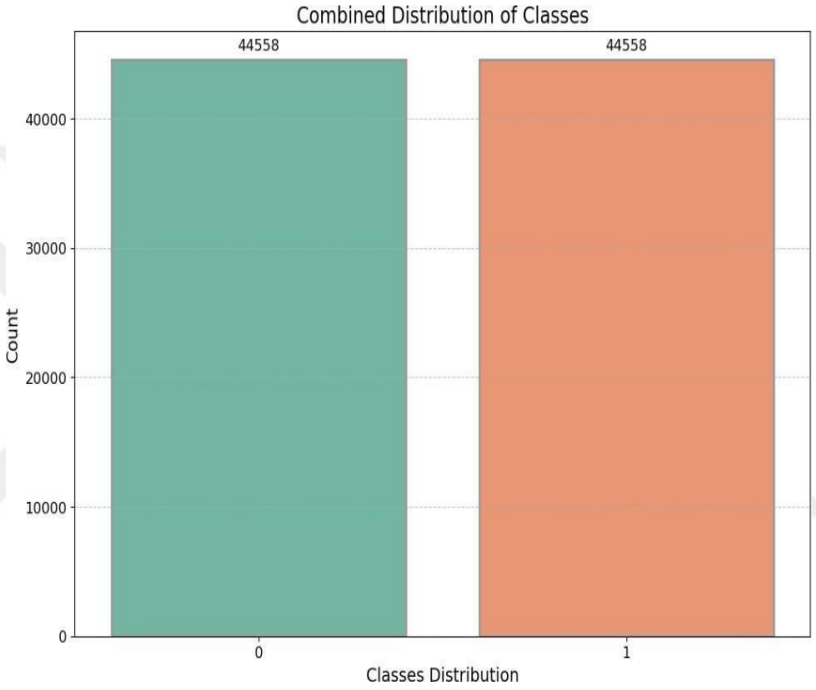

Figure 3.9: SMOTE Dataset Distribution

3.3.6 Feature Selection

A significant improvement in the performance and understanding of our Machine Learning Model is achieved by identifying attributes in the data set that have the most impact as features. This objective was achieved through the application of an intrinsic property of the Random Forest algorithm that allows for an evaluation of the importance of every feature during training through it.

Resampled dataset resampled_df, was obtained using Synthetic Minority Over-sampling Technique (SMOTE) in order to address the issues of class imbalance during the selection of features.

Random Forest Classifier for Feature Importance

We used the Random Forest Classifier as a tool to identify the most suitable features for our data set. Random Forest method is beneficial because it ranks all features on the basis of their individual contribution towards enhancing model prediction accuracy thereby making it the best option for such tasks to be performed. Here are some details about the techniques employed:

1. Separate Features and Target

The resampled dataset (resampled_df) was split into features (X) and the target variable (y), which is the 'marker' column indicating whether an instance belongs to the 'Attack' or 'Natural' class.

2. Fit the Random Forest Classifier

In training, the classifying algorithm builds decision trees to help it understand how the features are related to the target variable. At this point, the method learns these relationships using X and y as input parameters to random forest algorithms.

3. Get Feature Importance

For the purpose of predicting target variable importance, after training Random Forest Classifier, feature_importances method applied in order for retrieving the importance of all individual features used by this model. Importance score will be given representing how much that particular feature contributes towards the prediction power of the whole model.

4. Create DataFrame for Feature Importances

The classifier gives a data frame labeled feature_importance_df. Two columns: "Feature," that enlists all the feature names, and "Importance," which gives the rank of each feature.

5. Sort Features by Importance:

The features in feature_importance_df are sorted according to importance scores in descending order, for improved visualization and analysis of the key predictors of the target variable as demonstrated in Figure 3.10

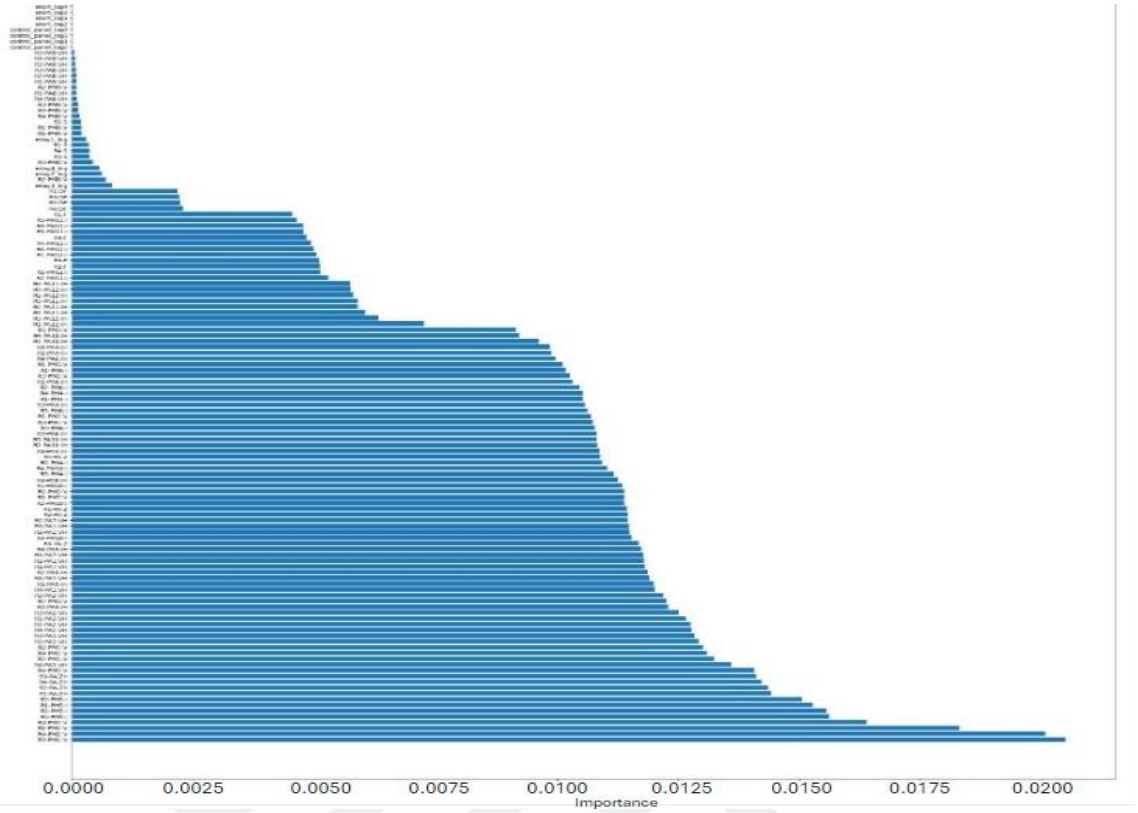

Figure 3.10: Ranking of Feature Importance

Filtering Features Based on Importance Threshold

Determining a threshold for choosing the most significant features came next, after the extraction of feature importance from the Random Forest Classifier and their decreasing order of significance. This threshold value is the lowest value of relevance that a characteristic can have in order to be considered significant for the classification task. we detail the process of filtering features based on threshold, as follows:

6. **Setting the Threshold for Feature Importance**

When selecting important features and balancing the need for relevance with dimensionality reduction, this study is based on the distribution pattern of importance scores. The threshold of 0.002 was determined by analyzing the distribution of these scores, where features with scores above this value are considered important and retained. This threshold was chosen based on statistical analysis of the importance score distribution and validation experiments to ensure that it effectively separates relevant features from less important ones, thus balancing model complexity and performance.

7. **Filtering Features Based on the Threshold:**

Then only features with scores higher than or equal to the predetermined threshold

are kept after features are filtered based on their significance ratings.

8. Filtering the Original Data:

Sequentially, these important features were condensed in the new Data- Frame X_filtered that was created through a process of selecting only those features considered. Resultant the filtered frame contains only the selected features which have been identified as very important in the classification process.

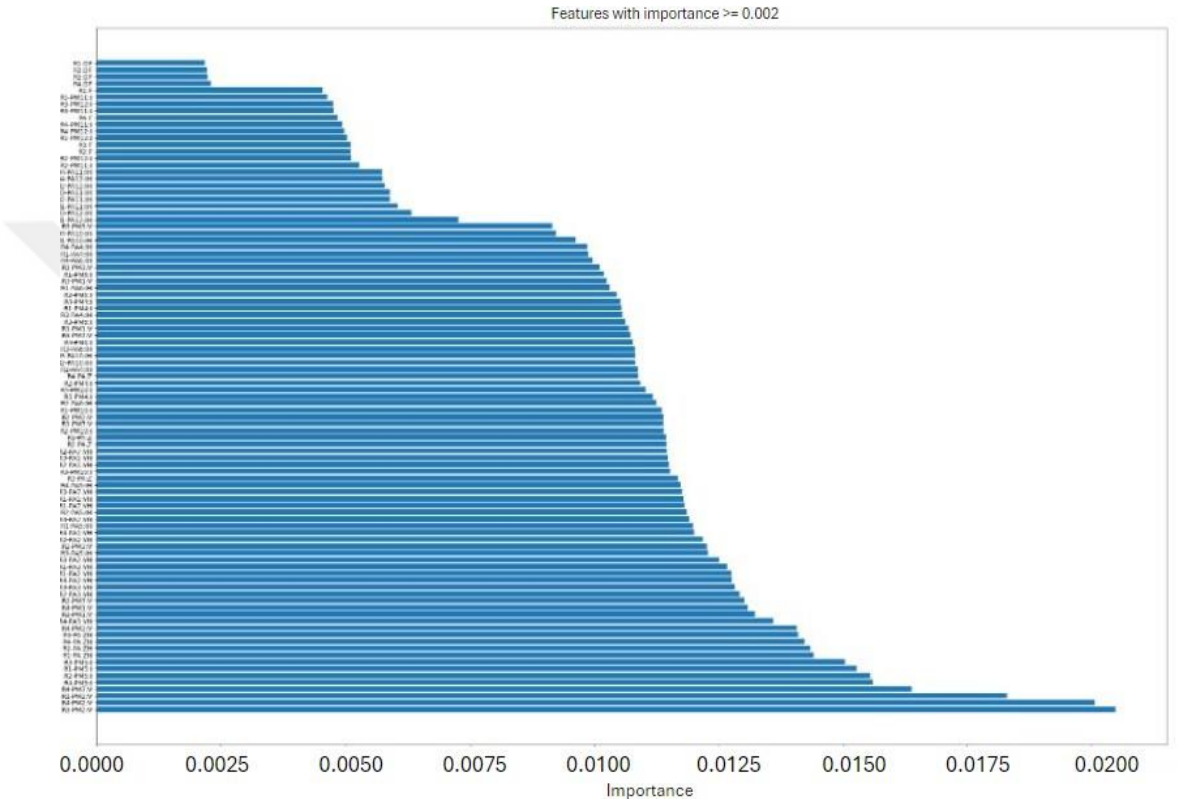

Figure 3.11: Distribution of Feature Importance After Applying the Threshold

To enhance the computational efficiency, interpretability and performance of the model, the features were filtered based on the significance threshold following which the dataset's dimensionality was reduced. In the next step the ExtraTreesClassifier will be trained using the selected features from X_filtered.

3.3.7 Conclusion

The binary category assignment of distinguishing among Natural and Attack activities in our dataset has been tackled in-depth in this subsection. To ensure the data is suitable for machine learning techniques, we began with careful information preprocessing methods such addressing infinite values, handling missing values, and encoding categorical features.

Next, we explored a variety of preprocessing techniques with the intention of enhancing the classification process. These strategies included undersampling, oversampling, and SMOTE for addressing the consequences of class imbalance, and feature selection for identifying the most relevant characteristics within the dataset.

3.4 MULTI-CLASS CLASSIFICATION: NATURAL, NO EVENTS, AND ATTACK EVENTS

Guided by the results obtained from the binary classification task, we go one step further to extend our work to the problem of multi-class classification distinguishing between the three classes: Natural events, No events, and Attack events. In the case of this multi-class classification problem, a model is being developed that can classify the three types of events, improving the capability of the smart grid towards attack detection and reaction. In the following section, we will be focusing on initial preprocessing steps of the data, such as checking missing values, handling infinite values, and encoding of labels for the three classes. Further, we visualize data and the class distribution, apply SMOTE to handle multi-class imbalance, feature selection, and train the model using the Extra Trees Classifier. The application of these techniques will therefore yield a very strong multi-class classification model with high accuracy in distinguishing between Natural, No Events, and Attack events, hence increasing the security of the overall system and reliability.

3.4.1 Initial Data Preprocessing

Handling Missing Values

Following the same rigorous preprocessing steps, we applied in the binary classification task, we began by checking the dataset for missing values. Using the isnull() function from pandas, we checked every column for entries that were null or NaN. In Figure 3.12, we clearly see that there are no missing values across any of the columns of the multi-class dataset.

The absence of missing data implies that the dataset is well curated, accurate and inclusive in its observations. In this scenario where there are no missing values, we would proceed to handling infinite values if any as the next preprocessing step before proceeding with analysis and modeling.

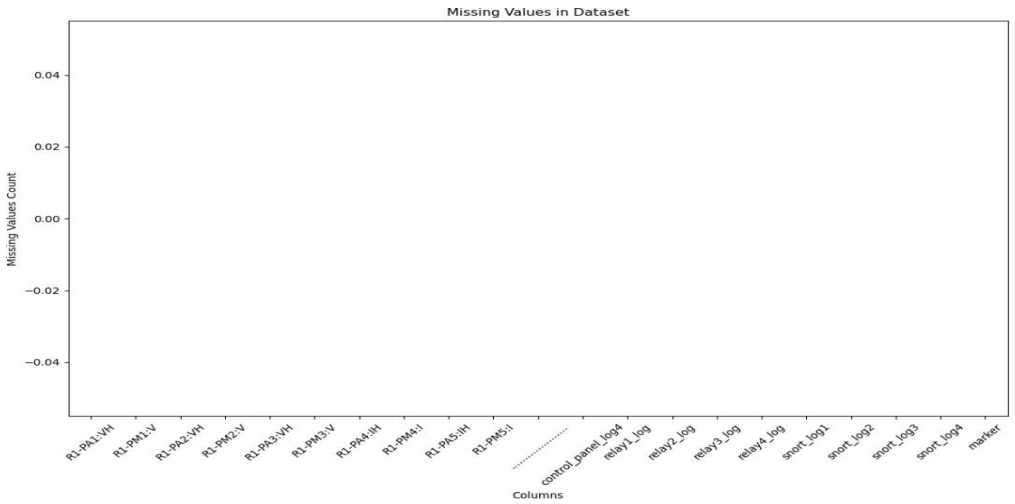

Figure 3.12: Missing Values Count in the multi-class dataset

Handling Infinite Values

Following the confirmation that the dataset contains no missing values, we went ahead to handle infinite values, as their existence may lead to decreasing the accuracy and stability of machine learning models. In order to find whether there were negative or positive infinite values in every column of the dataset, we employed masked arrays from NumPy. What was established were those columns which had infinite figures, after which we could now continue by dealing with them differently using the SimpleImputer function available on scikit-learn library by replacing these values with the maximum observed value in the respective column. Thus, the original data structure remained untouched without introducing any biasing but only eliminating the badness that would have emanated from those infinite values on data analysis and modeling.

Label encoding process for the multi-class dataset

Next, we will handle missing and infinite values. we wish to encode the categorical 'marker' column, which represents three classes: Natural, No Events, and Attack. Unlike in the case of a binary classification with two classes, in the current case, we face a multi-class problem, and label encoding is important to represent the classes in a manner that machine learning algorithms can understand numerically.

We applied the technique of label encoding, whereby a unique numerical label is assigned to every different category within the 'marker' column. This will actually transform the categorical data into a form ready for processing by most machine learning models.

R1-PM4:I	R1-PA5:IH	R1-PM5:I	...	relay1_log	relay2_log	relay3_log	relay4_log	snort_log1	snort_log2	snort_log3	snort_log4	marker	marker_encoded
276.86232	-123.174467	282.35562	...	0	0	0	0	0	0	0	0	NoEvents	2
278.69342	-123.409380	282.90495	...	0	0	0	0	0	0	0	0	NoEvents	2
279.24275	-123.443757	282.90495	...	0	0	0	0	0	0	0	0	NoEvents	2
280.15830	-123.581267	282.90495	...	0	0	0	0	0	0	0	0	NoEvents	2
255.62156	-121.129008	257.08644	...	0	0	0	0	0	0	0	0	NoEvents	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
265.50950	172.718127	264.04462	...	0	0	0	0	0	0	0	0	Attack	0
265.50950	172.712398	264.04462	...	0	0	0	0	0	0	0	0	Attack	0
265.32639	172.603536	264.04462	...	0	0	0	0	0	0	0	0	Attack	0
411.81439	49.721277	444.59108	...	0	0	0	0	0	0	0	0	Natural	1
436.16802	42.524928	466.93050	...	0	0	0	0	0	0	0	0	Natural	1

Figure 3.13: 'Marker' column before and after encoding

In Figure 3.13, we can observe the content of the "marker" column as it was before encoding, with classes shown as textual labels – "NoEvents", "Attack", and "Natural". Further, the label encoding has been implemented leading to creation of one more column titled "markerencoded" whereby each class gets its unique numerical value; 2 is assigned to "NoEvents", 0 goes as "Attack" and 1 stands for "Natural".

3.4.2 Data Visualization and Class Distribution

Under such circumstances, we begin our analysis of the nature of our multi-class dataset by visualizing the distribution of the three classes: Natural, No Events, and Attack events. Figure 3.14 shows a bar plot displaying their combined distribution.

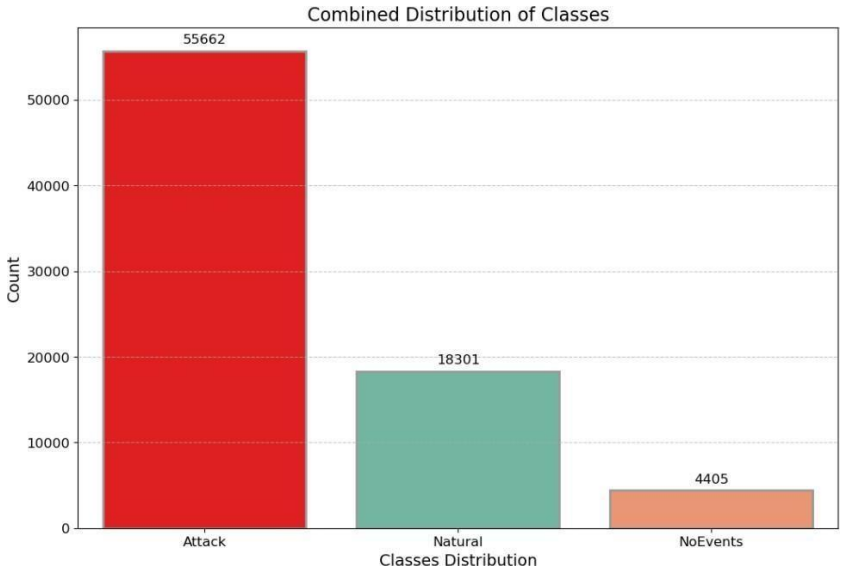

Figure 3.14: Distribution of Multi-Classes

We can see from the figure that the dataset is seriously imbalanced. The "Attack" class has the most instances with 55,662, followed by the "Natural" class with 18,301 instances, and the "NoEvents" class as the minority with only 4,405 instances.

3.4.3 SMOTE for Multi-Class Imbalance

We apply Synthetic Minority Oversampling Technique to reduce the effects of class imbalance and enhance the model's ability to learn from discriminative patterns of minority classes. The advanced oversampling method generates synthetic instances for minority classes, interpolating between examples in the feature space.

The SMOTE process was carried out as follows:

- The data, after feature separation, were split into an input variable (X) and a target variable (Y). Later, the training set was fed into the SMOTE algorithm with the 'sampling_strategy' parameter set to 'auto'; automatically, it determined the underrepresented class and oversampled it.
- The result will be the resampled_df DataFrame, containing the oversampled cases, where the class distribution now is reasonably balanced across all three classes, as shown in Figure 3.15. Each class now has the same number of instances 44,584 such that the machine learning model is able to capture meaningful patterns from all classes without being unduly influenced by the majority class.

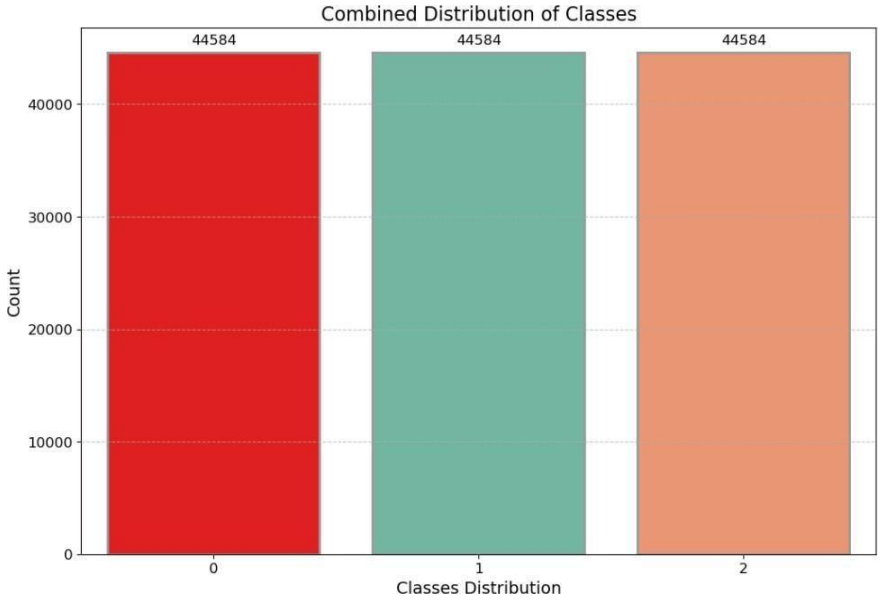

Figure 3.15: SMOTE Multi-Class Dataset Distribution

With the help of SMOTE, classes are now well balanced, so the dataset is ready to

train and test a multi-class classification model. Feature selection is now necessary. Usually, feature selection deals with choosing features most related to making model performance.

3.4.4 Feature Selection

Having addressed class imbalance with SMOTE, we sought to identify the most pertinent features for our multi-class classification task. Picking out features is a very important factor in improving how the model works, making it easier to understand and decreasing computational complexity.

We used a Random Forest Classifier to rank the features based on importance scores. We then fit the Random Forest model on resampled data with SMOTE after which we extracted the feature importance through the attribute feature- importance.

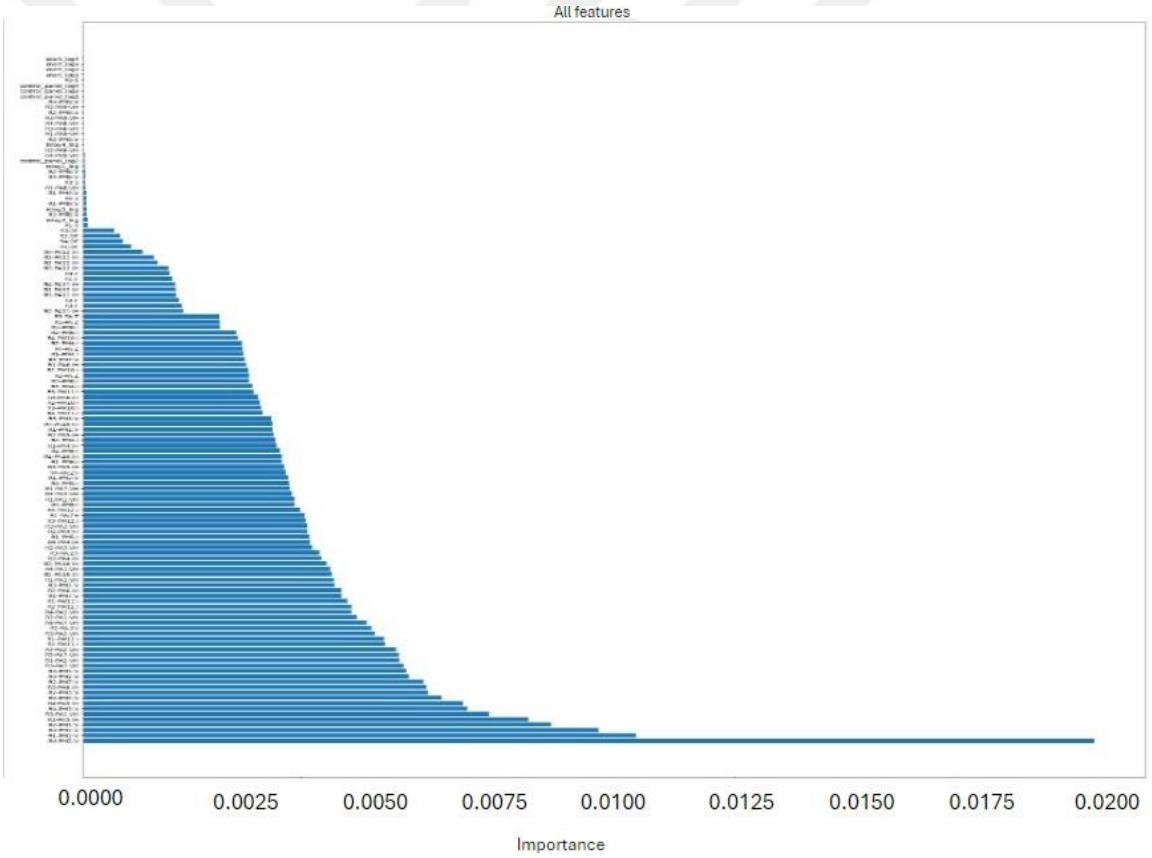

Figure 3.16: Ranking of Feature Importance in The Multi-class dataset

Figure 3.16 plots the feature importance scores in descending order. From this plot, we can observe the relative importance of each feature and get a hint about the candidates to be selected or dropped.

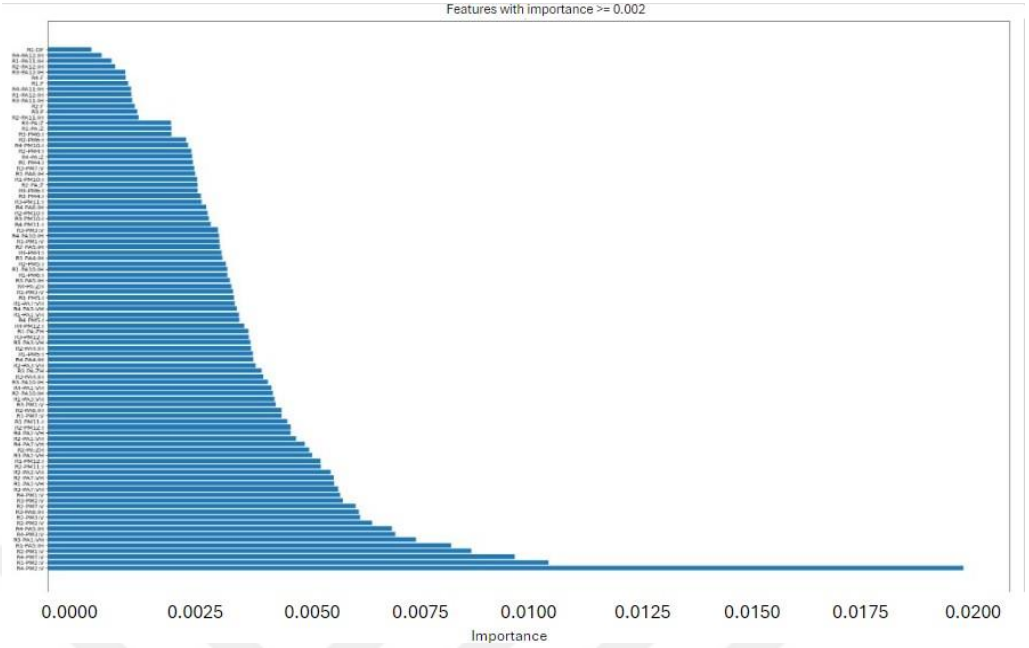

Figure 3.17: Feature Importance Distribution after applying the threshold

Figure 3.17 shows the threshold applied with the distribution of feature importance. The chosen features are clear and well-classified with less important features filtered out. Keeping only the most useful features makes the model more efficient for the catching of underlying patterns effectively and reduces the risk of over-fitting and enhances computational efficiency.

3.4.5 Conclusion

In the following section, the major steps for the proper classification of the dataset related to the power systems are presented. Later, work is done to preprocess this data, in order to make the data ready for machine-learning, which takes care of the infinities, the missing values, and the encoding of the categorical attributes. We handled firstly the problem of class imbalance in this part about under-sampling, over-sampling, and the SMOTE method. On the part of feature selection, we identified the most pertinent characteristics of our models. Therefore, these preprocessing methods increase the accuracies and robustness of both binary and multi- classification tasks, resulting in the developed much securer and efficient power system monitoring framework.

3.5 DEEP LEARNING CASE STUDY

In this section, we discuss how we apply deep learning techniques to our power system attack dataset. We will begin with CNNs, LSTMs, and then transition to hybrid models incorporating CNN and LSTM networks. We will employ the same preprocessing techniques used in the other sections, including SMOTE and feature selection using the Random Forest Classifier, with some additional preprocessing.

3.5.1 Data Preprocessing

In addition, apart from these already-discussed preprocessing steps, which involved handling missing values, infinite values, label encoding, applying SMOTE for class imbalance, and feature selection using a Random Forest Classifier, we did two more preprocessing steps to make the data ready for the CNN model:

1. **Clipping Data:**

Purpose: This is a method of clipping data to a range specifically tailored to handle outliers, which could be extreme and then degrade the performance of a model. The outliers may cause models to learn incorrect patterns that result in poor generalization on unseen data.

Implementation: Feature values are clipped into a range from -1e+10 to 1e+10. We used the clip () function from NumPy. This would allow values far from this range to be set to the nearest boundary value within such a range, thus making the data consistent.

2. **Standard Scaling:**

Purpose: The StandardScaler then is applied to do the standardization of feature values by removing the mean and scaling to unit variance, this transformation will allow each of the features to contribute by the same amount to the model features with large ranges that do not dominate the learning process.

Implementation: The StandardScaler from sklearn.preprocessing module is applied to the features. The transformation will scale the data so that its distribution will be at a mean of zero and a standard deviation of one. In many deep learning models, the standardization of features is very important, since this can assure the process of gradient descent to converge more efficiently.

3.5.2 Convolutional Neural Network (CNN)

After preprocessing, we define a CNN architecture tailored for the multi-class classification task. CNNs are particularly effective in extracting spatial patterns from data. In our case, we reshape the features to fit the input shape of the first 2D convolution layer in the CNN, treating each feature as a separate channel. Using 2D convolutions, even with 1D data, allows the model to apply convolutional filters across these multiple channels. This approach helps capture complex interactions between features that a purely 1D convolution might miss. The 2D convolutional technique enables the model to leverage established CNN methods for learning intricate feature representations and provides flexibility for adapting to future multi-dimensional data. This method effectively enhances the model's ability to detect and represent patterns in the data.

Model Architecture

The model summary provides an overview of the layers, their output shapes, and the number of parameters in the model as illustrated in Figure 3.18.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_10 (Conv2D)           (None, 1, 1, 32)          2912

flatten_18 (Flatten)         (None, 32)                0

dense_30 (Dense)             (None, 64)                2112

dense_31 (Dense)             (None, 512)               33280

dense_32 (Dense)             (None, 3)                 1539

=================================================================
Total params: 39,843
Trainable params: 39,843
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.18: CNN model summary the model consists of the following layers

- **Convolutional Layer**: The first layer is a 2D convolutional layer composed of 32 filters using ReLU activation with a kernel size of 1 × 1, Such a layer is used for the depiction of local patterns in the data.

- **Flatten Layers**: These layers convert 2D outputs that come from the convolutional layer into a 1D feature vector, so that it can be fed into dense layers.

- **Dense Layers**: These layers are composed of fully connected nodes. The first dense layer contains 64 units, which are ReLU activated. Thereafter, there is another

dense layer of size 512. The last dense layer contains units as many as the classes; this has the softmax activation function that provides the class probability distribution as output.

Model Training

• **Compilation:** The CNN model was compiled with a loss function of sparse_ categorical_crossentropy, which is proper for multi-class classification tasks where the labels are integer encoded. In this model training it uses an Adam optimizer and an additional metric of accuracy.

• **Training:** Training the model will run for 100 epochs, using a batch size of 32. Data will then be split into trains and validation and performance will be monitored on both these sets.

After training, the CNN model yields high accuracy on both the training and validation sets. The following are the last results:

- Final Training Accuracy: 98,76%
- Final Training Loss: 0.0322
- Final Validation Accuracy: 97,79%
- Final Validation Loss: 0.0857

These results in a way point out that the model performs well on the validation set, which is a good indication of the generalization to unseen data. We visualize the training and validation accuracy and loss over epochs for an understanding of the performance of the CNN model. As illustrated in the Figures 3.19, 3.20.

Accuracy Plot:

- **Description:** This plot demonstrates how the training and validation accuracy change over the 100 epochs. On the x-axis, it represents the number of the epoch, while on the y-axis, it shows the accuracy values between 0 and 1.
- **Analysis:** Both the training and validation accuracy curves, at the end of this training, seem to level out around an accuracy for the training dataset of about 98% and an accuracy for the validation dataset of about 97%. High accuracy values, besides the fact that the curves are leveling out, indicate good performance on the train and validation sets.

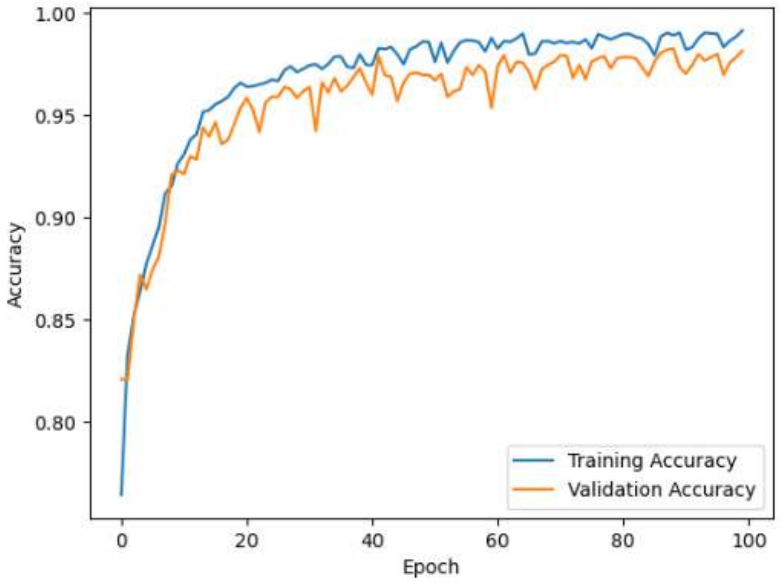

Figure 3.19: CNN Training and Validation Accuracy Curves

- Loss Plot:

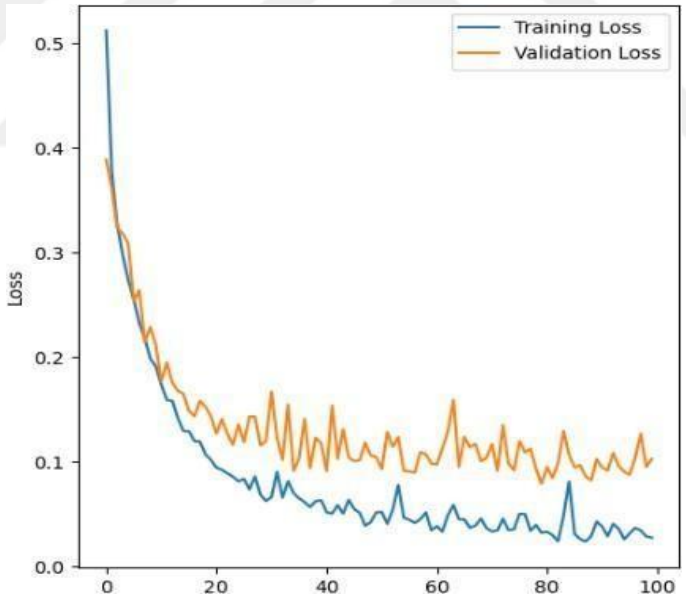

Figure 3.20: CNN Training and Validation Loss Curves

- **Description:** This graph displays how training and validation loss varies as we go through 100 different epochs. On the x-axis a number representing each epoch appears while on the y-axis are the corresponding losses measurements.
- **Analysis:** The two losses in training and validation decrease as time goes by, but that of the training falls faster. This means that the model still works well even on unseen testing data because it sustains its performance.

Next, we explore in detail some of the more advanced deep learning architectures, including the Long Short-Term Memory (LSTM) networks and hybrid CNN-LSTM models, to further enhance the ability to recognize temporal patterns for this time- series data.

3.5.3 Long Short-Term Memory (LSTM)

In this stage, we explore the architecture and the training performance of the Long Short-Term Memory Network model in classifying events in smart-grid data into either Natural, No Events, or Attacks. For this, the architecture of LSTMs inherently fits very well for sequential data and hence captures long-term dependencies that are very important to detect anomalies and classify possible attacks in time-series data from smart-grid systems.

Model Architecture

The model summary provides an overview of the layers, their output shapes, and the number of parameters in the model as illustrated in Figure 3.21.

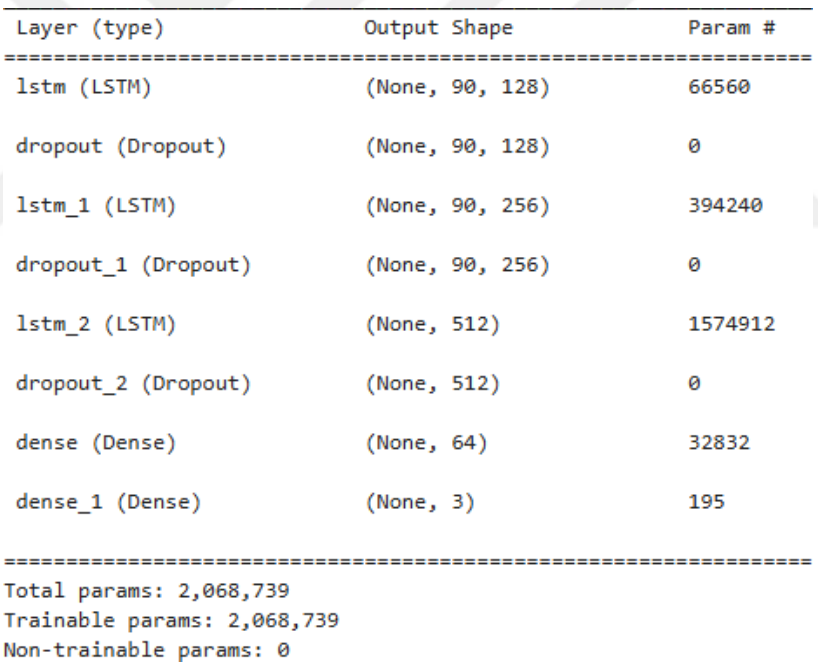

Figure 3.21: LSTM Model Architecture

The model here symbolizes three LSTM layers stacked with 128, 256, and 512 units, respectively. The input layer consists of an LSTM unit with 128 nodes. It is designed to take an input of shape (None, 90), where None is the batch size, and 90 is the sequence length. The output shape is (None, 90, 128), where 90 is the sequence length and 128 is the unit size within that LSTM unit. Each of the LSTM layers is followed, for regularization, by a Dropout layer that helps to reduce overfitting in smart grid data analysis. These LSTM layers are followed by two fully connected Dense layers with 64

units and the final output layer consisting of 3 units corresponding to the three classes of Natural, No Events, and Attacks.

Model Training

- **Compilation:** The LSTM model was compiled with a loss function of sparse_ categorical_crossentropy, which is proper for multi-class classification tasks where the labels are integer encoded. In this model training it uses an Adam optimizer and an additional metric of accuracy.

- **Training:** Training the model will run for 100 epochs, using a batch size of 32. The smart grid will then be split into trains and validation and performance will be monitored on both these sets.

After training, the LSTM model yields high accuracy on both the training and validation sets. The following are the results of training:

- Final Training Accuracy: 97,29%
- Final Training Loss: 0.0767
- Final Validation Loss: 0.1230

We visualize the training and validation accuracy and loss over epochs for an understanding of the performance of the CNN model. As illustrated in the Figures 3.22 and 3.23.

Accuracy Plot:

- **Description:** This plot demonstrates how the training and validation accuracy change over the 100 epochs. On the x-axis, it represents the number of the epoch, while on the y-axis, it shows the accuracy values between 0 and 1.

- **Analysis:** The training accuracy curve is steep in the early epochs, indicating that the model is effectively learning from the training data achieving around 97% accuracy in the training dataset. The curve for validation accuracy by achieving around 95% is more or less similar, though it comes with some lower values and fluctuations, as expected when evaluating on unseen smart grid data.

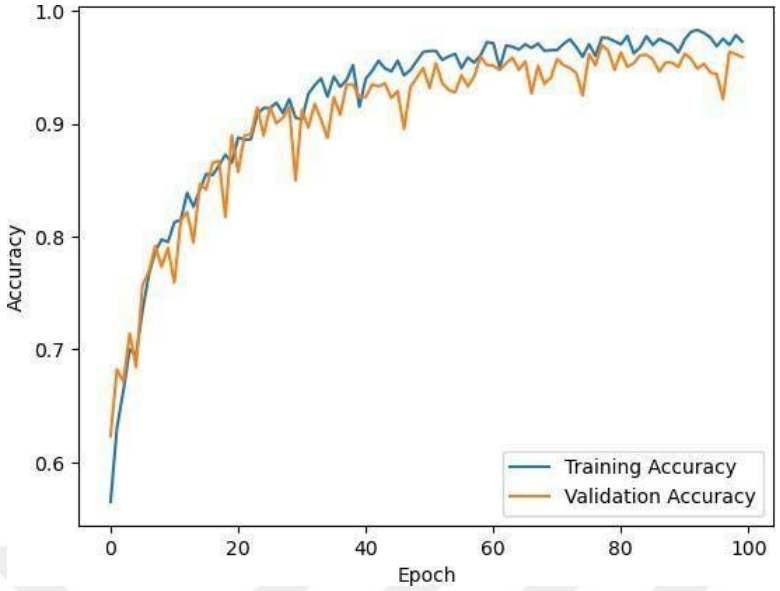

Figure 3.22: LSTM Training and Validation Accuracy Curves

Loss Plot:

- **Description:** This graph displays how training and validation loss varies as we go through 100 different periods (epochs). On the x-axis a number representing each epoch appears while on the y-axis are the corresponding losses measurements.
- **Analysis:** The training loss plot follows a smooth, progressive decline, indicating that the model is able to decrease the loss function on the training data. The validation loss plot fluctuates more yet follows the same general trend thus showing a fairly well-fitted model without overfitting.

Lastly, hybrid models, combining powers of CNNs and LSTMs, CNN-LSTM architectures, will also be investigated. Such hybrid models will have the potential to learn both local, spatial patterns and long-term temporal dependencies within the data produced by the smart grid, therefore enhancing the false data detection capability and offering deep insights into the underlying events and anomalies.

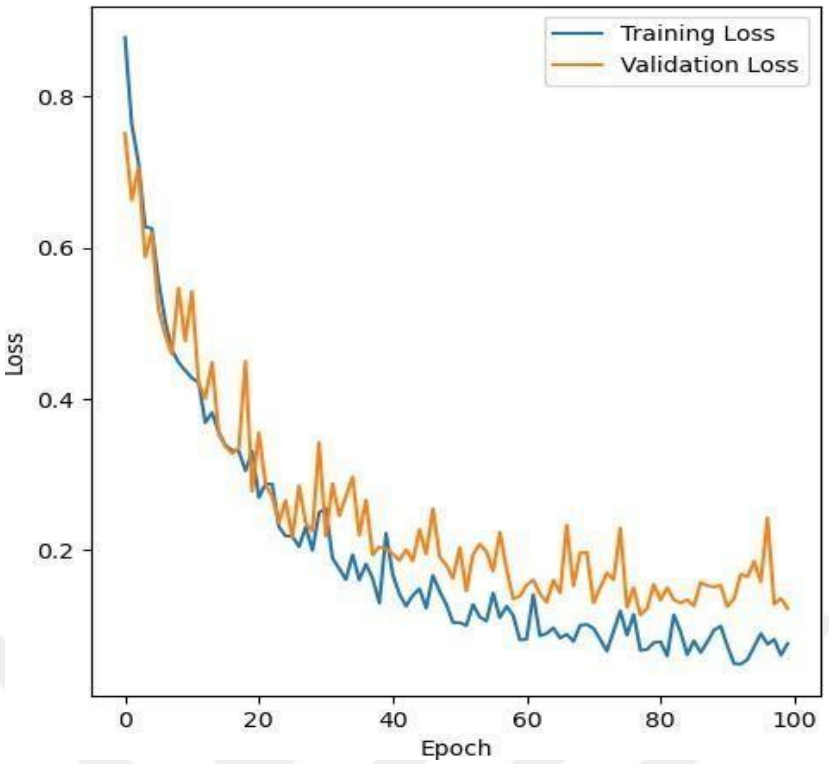

Figure 3.23: LSTM Training and Validation Loss Curves

3.5.4 Hybrid Model CNN-LSTM

The contribution of this subsection is to develop a hybrid model integrating the merits of Convolutional Neural Networks and Long Short-Term Memory Networks for the false data detection problem in smart grids. The proposed hybrid architecture integrates the strength of CNNs in local spatial pattern extraction and the abilities of LSTMs to model long-term temporal dependencies for the analysis of complex time series data generated from smart grid systems.

Model Architecture:

The proposed hybrid model architecture is comprised of two prime constituents include:

- The CNN layers that extract the spatial features from the input data.
- The LSTM layers to capture the temporal or sequential features in the data.

The model summary provides an overview of the layers, their output shapes, and the number of parameters in the model as illustrated in Figure 3.24.

The first layer consists of 32 filters of size (1*1) with an activation function of ReLU in 2D convolution. Low-level spatial features in the input data can be detected by this layer, whose shape has been reshaped to be (batch_size 1*1 input_features). After

flattening the output from this convolutional layer is then passed through a dense layer with units 64 that has activation function ReLU.

One more dense layer of 512 units with ReLU activation, helping in preparing the output coming out from the layers of CNN to feed into the layers of LSTM. Next, this is followed by a flatten operation. The resulting output is reshaped to match the input shape required by the LSTM layers of (batch_size, time_steps, features).

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 1, 1, 32)	2912
flatten (Flatten)	(None, 32)	0
dense (Dense)	(None, 64)	2112
dense_1 (Dense)	(None, 512)	33280
reshape (Reshape)	(None, 1, 512)	0
lstm (LSTM)	(None, 1, 64)	147712
dropout (Dropout)	(None, 1, 64)	0
lstm_1 (LSTM)	(None, 1, 128)	98816
dropout_1 (Dropout)	(None, 1, 128)	0
lstm_2 (LSTM)	(None, 256)	394240
dropout_2 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 32)	8224
dense_3 (Dense)	(None, 32)	1056
dense_4 (Dense)	(None, 3)	99

Total params: 688,451
Trainable params: 688,451
Non-trainable params: 0

Figure 3.24: CNN-LSTM Model Architecture

Three LSTM layers with 64, 128 and 256 units; the return_sequences attribute must be set to true for the first two LSTM layers such that they return a full sequence of output for every time step. their outputs go into the next LSTM layer; while for the last LSTM layer, we set the return_sequences as False in order to only get an output from the last time step.

Added after every LSTM layer dropout layer with a rate of 0.2, to help in the reduction of overfitting. Next, the last LSTM layer feeds its output to a dense layer of 32 units using ReLU activation, followed by another dense layer of 32 units using ReLU activation with L2 regularization with a weight decay of 0.001, serving as another hidden layer but with the applied regularization to the kernel weights. The L2 regularization

avoids overfitting by adding a penalty term to the loss function that leads to smaller weights and lesser complexity in the model.

Afterwards, a dense layer is used on this output whose number of units 3 is determined by the number of unique classes in the target labels, Natural, No Events, and Attacks, with the final classification probabilities obtained through a softmax activation function.

Model Training

- **Compilation:** The CNN-LSTM model was compiled with a loss function of sparse_ categorical_crossentropy. In this model training it uses an Adam optimizer and an additional metric of accuracy.
- **Training:** Training the model will run for 100 epochs, using a batch size of 16. The smart grid will then be split into trains and validation and performance will be monitored on both these sets.

After training, the CNN-LSTM model yields high accuracy on both the training and validation sets. The following are the results of training:

- Final Training Accuracy: 98,18%
- Final Training Loss:0.0543
- Final Validation Accuracy: 97,36%
- Final Validation Loss: 0.1283

We visualize the training and validation accuracy and loss over epochs for an understanding of the performance of the CNN model as illustrated in Figures 3.25 and 3.26.

- Accuracy Plot:
- **Description:** This plot demonstrates how the training and validation accuracy change over the 100 epochs. On the x-axis, it represents the number of the epoch, while on the y-axis, it shows the accuracy values between 0 and 1.
- **Analysis:** The blue plot of the training accuracy rises very steeply in the first epochs and eventually flattens out to converge to approximately 0.98 toward the end of training. In orange, the validation accuracy curve follows a similar pattern but now plateaus at a slightly lower value.

That means it is fitting the training data very well and, as expected, exhibits a slight gap in generalization regarding the validation data.

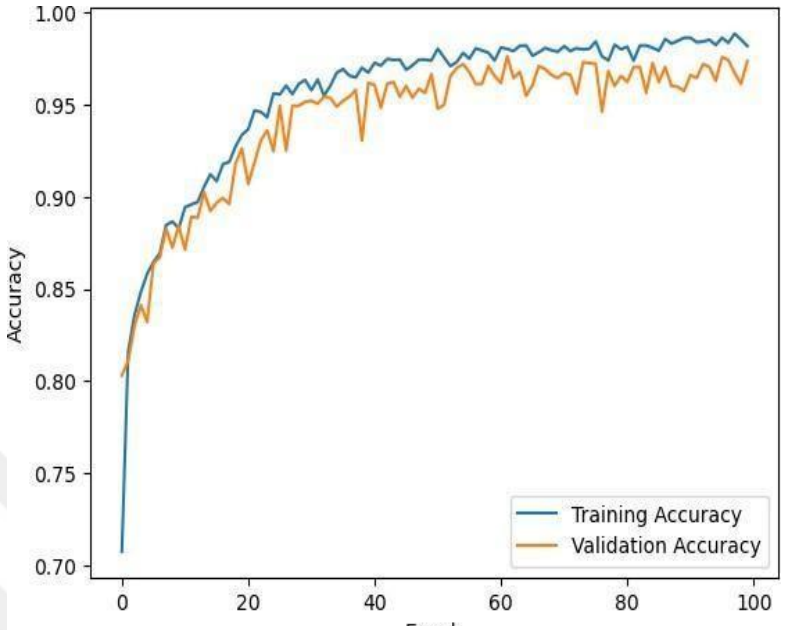

Figure 3.25: CNN-LSTM Training and Validation Accuracy Curves

Loss Plot:

- **Description:** This graph displays how training and validation loss varies as we go through 100 different periods (epochs).
- **Analysis:** Training loss decreases rapidly in the first few epochs and then continues to decrease slowly; the final value for the training loss is at about 0.05. Validation loss generally trends similarly to training loss, but the plateau occurs at a higher value of about 0.13 again, suggesting a relatively small generalization gap.

The last training and validation metrics correspond to the tendency reflected in the curves. The model depicts high accuracy in training equal to 0.9818 and a low loss in training equal to 0.0543, reflecting excellent fitting on the training data. The validation accuracy of 0.9736 and the loss of 0.1283 are a bit worse than the training metrics but still point to good generalization performance.

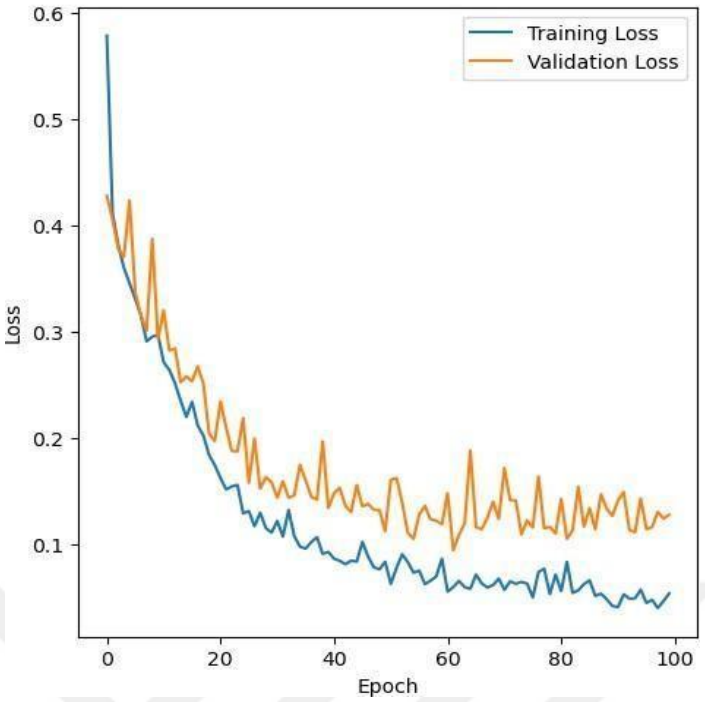

Figure 3.26: CNN-LSTM Training and Validation Loss Curves

3.5.5 Conclusion

CNN, LSTM networks, and Hybrid CNN-LSTM in handling the attack on the power system dataset. Data preprocessing methods include imputation of missing, infinite, label encoding, handling class imbalance with SMOTE, feature selection using a Random Forest Classifier, data clipping for handling outliers, and standard scaling for feature normalization.

In general, in the training, the deep learning models for the dataset of this power system attack, they represented high approximate values and carried low loss values. It showed the ability of such models in the successful detection and classification of false data injections and other forms of anomalies in the smart grid environment. This research alludes to appreciable potentials that techniques of deep learning hold for the smart grid systems' security and dependability.

The findings and the evaluation of our models shall be shown in the next chapter, where a comprehensive test set performance analysis will take place.

4 FINDINGS

This chapter presents the results obtained from the evaluation of traditional machine learning-based models Extra Trees classifier and Random Forest to the state-of-the- art deep learning models, Convolutional Neural Networks and Long Short-Term Memory Networks, along with a hybrid CNN-LSTM architecture for the detection of cyber-attacks via false data injection on smart grid phasor measurement units.

However, we delve further to find out the influence of pre-processing techniques on the performance of the models. The results were analyzed based on accuracy, precision, recall, F1-score and confusion matrices, including the ability to handle class imbalances and locate instances of the minority class, and an overview table comparing the performances of the different models.

4.1 MACHINE LEARNING CASE STUDY FINDINGS

4.1.1 Initial Model Performance on Binary Unbalanced Dataset

Two machine learning models were trained on the original imbalanced dataset, to establish a ground level before applying any sampling techniques, and this is the results obtained from testing the Extra Trees Classifier and the Random Forest Classifier.

Extra Trees Classifier

The Extra Trees Classifier was trained on the imbalanced dataset, and the following results were obtained:

- Score: 0.94
- Precision Score: 0.94
- Recall Score: 0.91
- F1-Score: 0.93

Classification Report:

Table 4.1 ETC Classification Report on Imbalanced Dataset

Class	Precision	Recall	F1-Score
0	0.93	0.98	0.96
1	0.94	0.85	0.89

Confusion Matrix:

As Figure 4.2 illustrate:

- The top-left cell represents the number of true negatives (TN), meaning the number of observations correctly predicted as class 0.
- The top-right cell represents the number of false positives (FP), meaning the number of observations incorrectly predicted as class 1 when they are actually class 0.
- The bottom-left cell represents the number of false negatives (FN), meaning the number of observations incorrectly predicted as class 0 when they are actually class 1.
- The bottom-right cell represents the number of true positives (TP), meaning the number of observations correctly predicted as class 1.

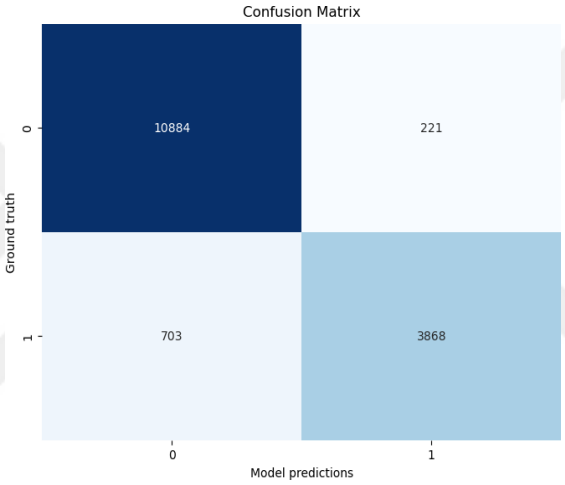

Figure 4.1: ETC Confusion Matrix on Imbalanced Dataset

Random Forest Classifier

The Random Forest Classifier was also trained on the imbalanced dataset, yielding the following results:

- Accuracy Score: 0.92
- Precision Score: 0.93
- Recall Score: 0.89
- F1-Score: 0.90

Classification Report

table 4.2 RF Classification Report on Imbalanced Dataset

Class	Precision	Recall	F1-Score
0	0.92	0.98	0.95
1	0.94	0.79	0.86

Confusion Matrix

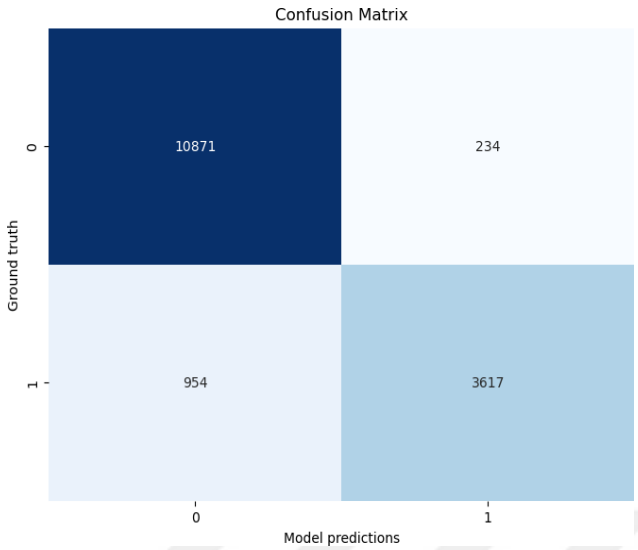

Figure 4.2: RF Confusion Matrix on Imbalanced Dataset

4.1.1.2 Observations and Limitations

The findings show that in the unbalanced data set, the Random Forest Classifier and the Extra Trees Classifier both obtained good accuracy ratings. Nevertheless, it is evident from a closer look at the confusion matrices and classification reports that the models had trouble correctly classifying cases from the minority class ("Natural"). With 0.91 for the Extra Trees Classifier and 0.89 for the Random Forest Classifier, the models' recall scores for the minority class which indicate their capacity to detect true positives were comparatively low. This suggests that many examples belonging to the "Natural" class were mistakenly assigned to the "Attack" class, Additionally, the confusion matrices show a higher proportion of false positives.

4.1.2 Performance on Binary Under-sampled Dataset

We trained the machine learning models on the undersampled dataset after implementing the Random Undersampling approach in order to assess their performance on the balanced class distribution. The outcomes of testing the Random Forest and Extra Trees classifiers on the undersampled data are shown in the following parts.

Extra Trees Classifier

The Extra Trees Classifier was trained on the undersampled dataset, and the following results were obtained:

- Accuracy Score: 0.90

- Precision Score: 0.88
- Recall Score: 0.91
- F1-Score: 0.89

Classification Report

Table 4.3 (ETC) Classification Report on Under-sampled Dataset

Class	Precision	Recall	F1-Score
0	0.97	0.90	0.93
1	0.79	0.92	0.85

The Extra Trees Classifier achieved a precision of 0.96 for the majority class "0" and 0.80 for the minority class "1" as presented in the classification report. Further- more, recall scores were 0.91 for both classes meaning both classes were correctly identified by the model almost equally. The F1-score, which is the harmonic mean of precision and recall, supports that the majority class had a higher value (0.93) while the minority class had a lower one (0.85). Hence, the precision-recall balance was slightly better for the majority class than for the minority one for this model.

Confusion Matrix

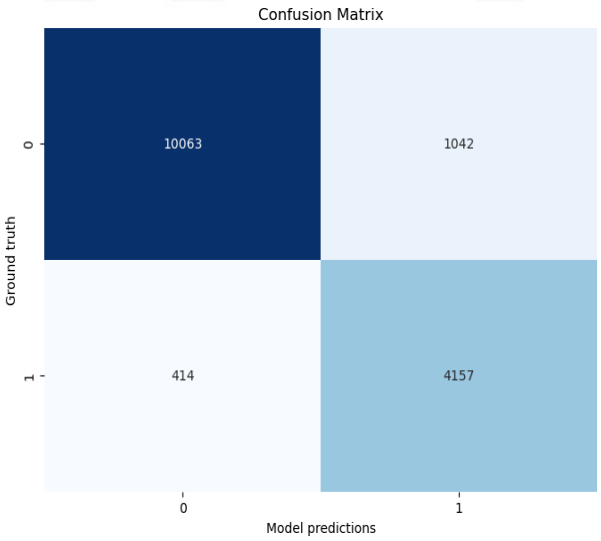

Figure 4.3: ETC Confusion Matrix on Under sampled Dataset

Random Forest Classifier

The Random Forest Classifier was also trained on the imbalanced dataset, yielding the following results:

- Accuracy Score: 0.89

- Precision Score: 0.86
- Recall Score: 0.90
- F1-Score: 0.87

Classification Report

Table 4.4 (RF) Classification Report on Undersampled Dataset

Class	Precision	Recall	F1-Score
0	0.96	0.88	0.92
1	0.76	0.91	0.83

The Extra Trees Classifier outperformed the Random Forest Classifier marginally on the resampled data. but the precision score associated with the minority class ("1") was 0.86, a value below that of the Extra Trees Classifier which measured 0.80 for the same class as well. Both classes had a recall score of 0.90, meaning they could accurately identify samples in these classes. Nonetheless, the minority class ("1") recorded a lower F1-score of 0.87 than that of the same class in Extra Trees Classifier at 0.85.

Confusion Matrix

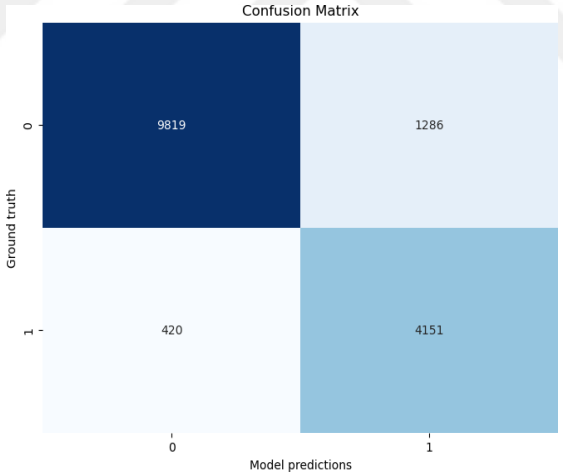

Figure 4.4: RF Confusion Matrix on Under-sampled Dataset

Observations and Limitations

A significant improvement for minority class ("1") compared to original unbalanced data set was found through Extra Trees Classifier and Random Forest Classifier on undersampled data. The balancing class distribution that resulted from under-sampling enabled these models to identify minority class instances better. Balanced performance on both classes is preferable in situations where correctly recognizing instances from the minority class is critical, even though the total accuracy scores may be lower than those achieved on the imbalanced dataset.

4.1.3 Performance on Binary Over-sampled Dataset

Extra Trees Classifier

The Extra Trees Classifier was trained on the over-sampled dataset, and the following results were obtained:

- Accuracy Score: 0.93
- Precision Score: 0.94
- Recall Score: 0.91
- F1-Score: 0.92

The classification report provides a breakdown of the model's performance for each class as Table 4.5 illustrates.

Table 4.5 (ETC) Classification Report on Over-sampled Dataset

Class	Precision	Recall	F 1-Score
0	0.94	0.98	0.96
1	0.95	0.84	0.89

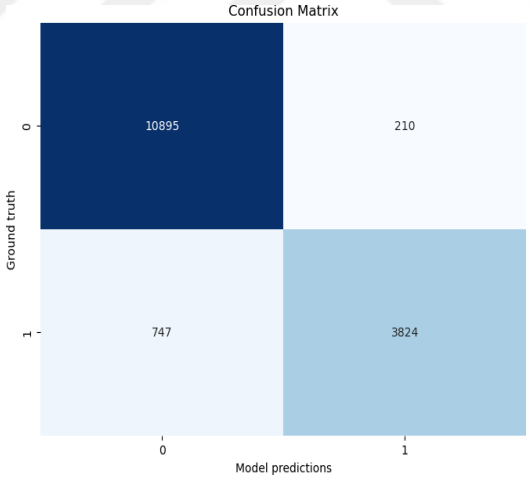

Figure 4.5: (ETC) Confusion Matrix on Over-sampled Dataset

Confusion Matrix

Random Forest Classifier

The Random Forest Classifier was also trained on the oversampled dataset, yielding the following results:

- Accuracy Score: 0.93
- Precision Score: 0.93
- Recall Score: 0.90

- F1-Score: 0.91

Classification Report:

Table 4.6 (RF) Classification Report on Over-sampled Dataset

Class	Precision	Recall	F1-Score
0	0.93	0.97	0.95
1	0.92	0.84	0.87

Confusion Matrix:

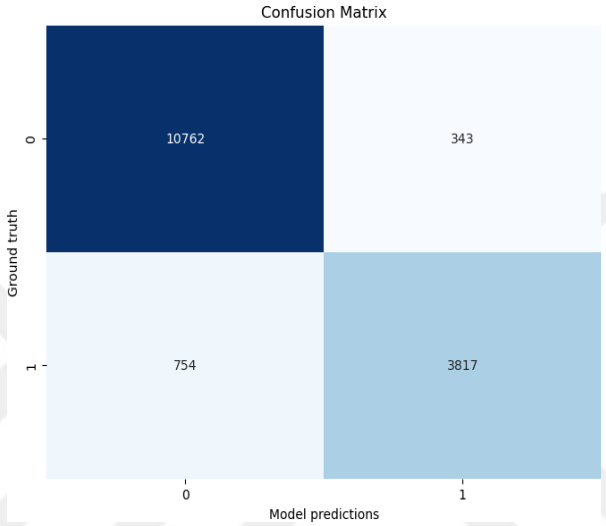

Figure 4.5: (RF) Confusion Matrix on Over-sampled Dataset

Observations and Improvements

The performance of the Random Forest Classifier and the Extra Trees Classifier for the minority class (Natural "1") on the oversampled dataset was much better than that of the undersampled and initially unbalanced datasets. The models were able to identify instances from the minority class more effectively because of the balanced class distribution that oversampling produced. This is important because it helps the smart grid system detect possible false data injection attempts. Balanced performance on both classes is preferable in situations where correctly recognizing instances from the minority class is critical, even though the overall accuracy scores are similar to those found on the imbalanced dataset.

4.1.4 Performance on Binary (Synthetic Minority Over-sampling Technique) Dataset

We trained machine learning models on the balanced class distribution and assessed their performance using the SMOTE over-sampled dataset (resampled_df). The outcomes of training the Random Forest and Extra Trees classifiers on the SMOTE over-sampled dataset are shown in the

following parts.

Extra Trees Classifier

The Extra Trees Classifier was trained on the SMOTE over-sampled dataset, and the following results were obtained:

- Accuracy Score: 0.94
- Precision Score: 0.94
- Recall Score: 0.92
- F1-Score: 0.92

The classification report provides a breakdown of the model's performance for each class as Table 4.7 illustrates

Table 4.7 (ETC) Classification Report on SMOTE Dataset

Class	Precision	Recall	F1-Score
0	0.95	0.96	0.95
1	0.94	0.87	0.89

Confusion Matrix

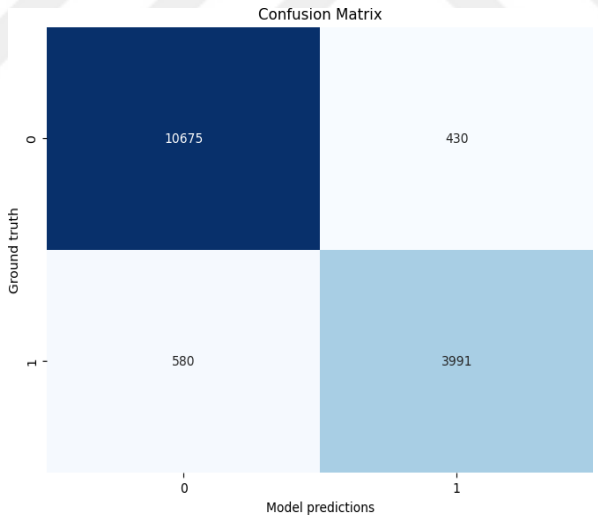

Figure 4.6: (ETC) Confusion Matrix on SMOTE Dataset

Random Forest Classifier

The Random Forest Classifier was also trained on the SMOTE oversampled dataset, yielding the following results:

- Accuracy Score: 0.90
- Precision Score: 0.89

- Recall Score: 0.87
- F1-Score: 0.88

Classification Report

Table 4.8 (RF) Classification Report on SMOTE Dataset

Class	Precision	Recall	F1-Score
0	0.91	0.95	0.93
1	0.87	0.78	0.83

Confusion Matrix

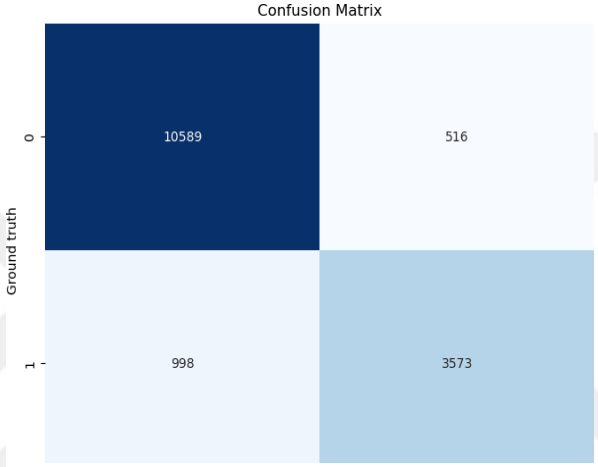

Figure 4.7: (RF) Confusion Matrix on SMOTE Dataset

Observations and Improvements

The performance of the Random Forest Classifier and the Extra Trees Classifier increased significantly for the minority class ("1") on the SMOTE oversampled dataset when compared to the results obtained on the undersampled, randomly oversampled and initially unbalanced datasets. The models were able to identify instances from the minority class more effectively because of the balanced class distribution that SMOTE helped to establish. This is important because it helps to properly detect possible false data injection assaults in the smart grid system.

The Extra Trees Classifier outperformed the Random Forest Classifier in the SMOTE oversampled dataset by recording lower numbers of misclassified examples, but higher accuracy, recall and F1-scores for both the classes.

4.1.5 Evaluation of SMOTE and Feature Selection Techniques

Following a SMOTE and feature selection procedure that identified the most significant among the features in the dataset, the ExtraTreesClassifier was trained on a filtered set of properties introduced as X_filtered and the target variables.

Extra Trees Classifier

Several measures were used to assess the effectiveness of the Extra Trees Classifier, including F1-score, accuracy, precision, and recall. The following outcomes were achieved as presented in Table 4.9.

Classification Report

Table 4.9 (ETC) Classification Report on SMOTE and Feature Selected Dataset

Class	Precision	Recall	F1-Score
0	0.94	0.95	0.94
1	0.95	0.94	0.94

The ExtraTreesClassifier succeeded to classify the samples with respect to the model's overall accuracy of 0.94%. Moreover, this high accuracy implies that the model can effectively differentiate between the two classes "Attack" and "Natural".

Observations and Improvements

Additional information on the model's performance for every class found in the classification report. The accuracy and recall for the majority class (0) were 0.94 and 0.95 respectively. This shows an acceptable trade-off between avoiding false positives and correctly spotting true positives. Similarly, the minority class (1) had accuracy and recall scores of 0.95 and 0.94 respectively thereby indicating that it had the capability to pick out instances of the minority class with high correctness while maintaining a low percentage of false positive rate.

4.1.6 Comparative Analysis

We tested how effectively the dataset's intrinsic class imbalance, based on a ratio of about 10 times more "Attack" instances compared to "Natural" instances can be addressed by four different preprocessing techniques: Random Under-sampling, Random Over-sampling, the Synthetic Minority Over-sampling Technique (SMOTE) and Feature selection, model training and testing was conducted with Random Forest and Extra Trees classifiers.

"Attack" class was significantly favored as a result of an unbalanced data set used to train classifiers without any preprocessing. The recall of the most common class in the Random Forest and Extra Trees classifiers was high 0.98. On the other hand, when it comes to the 'Natural' class that is a minority, the recall was significantly less, with rates varying from 0.85 and 0.79. This actually made most crucial occurrences of the 'Natural' class get wrong classification; an indication that these classifiers prefer most dominant classes.

We used Random undersampling method to realize balance which entails randomly removing some samples from one class while keeping others as they were. This strategy on one hand lowered overall accuracy 0.91 and 0.89. Even though recall for the minority class went up by 0.91. There was a cost to performance when instances belonging to the majority class were eliminated. Consequently, some vital information got lost.

On the contrary, the recall for the minority class increased while keeping fairly high precision for the majority class through the Random Oversampling technique in which instances from the minority class were randomly duplicated. Nonetheless, the total accuracy remained lower as opposed to other techniques perhaps due to overfitting that resulted from duplicated instances.

Out of the algorithms that were tested, SMOTE was found to be the most successful preprocessing method. It not only increased recall of the minority class to a range of 0.87 and 0.78, but also kept a high level of precision for the majority class (between 0.95 and 0.91). The best accuracy was obtained from employing this technique together with an Extra Trees Classifier.

When trained on a SMOTE filtered set of features after feature selection process the Extra Trees Classifier outperformed all other methods with respect to overall accuracy. It outperformed the Random Forest Classifier and Extra Trees Classifier on unbalanced data both in terms of accuracies and in performance also on over-sampled and under-sampled preprocessed datasets.

These exciting findings suggest that the Extra Trees Classifier combined with the Synthetic Minority Oversampling Technique and a feature selection procedure is an effective system that can be used in detecting controllers that input false data leading to security threats within important smart systems like increased general security as well as reliability.

Table 4.10 Performance summary table for Binary classification

Classifier	Dataset	Accuracy	P (0)	P (1)	R (0)	R (1)	F1 (0)	F1(1)
Extra Trees	Imbalanced	0.91	0.90	0.87	0.96	0.87	0.95	0.89
Random Forest	Imbalanced	0.92	0.93	0.89	0.98	0.79	0.95	0.86
Extra Trees	Under-sampled	0.91	0.92	0.88	0.95	0.86	0.94	0.87
Random Forest	Under-sampled	0.90	0.90	0.87	0.95	0.85	0.92	0.86
Extra Trees	Over-sampled	0.93	0.94	0.95	0.98	0.84	0.96	0.89
Random Forest	Over-sampled	0.93	0.93	0.92	0.97	0.84	0.95	0.87
Extra Trees	SMOTE	0.94	0.95	0.94	0.96	0.87	0.95	0.89
Random Forest	SMOTE	0.90	0.91	0.87	0.95	0.87	0.93	0.83
Extra Trees	Feature selected	0.94	0.94	0.95	0.95	0.94	0.94	0.94

Table 4.10 summarizes performance measures for the various models where:

- P (0): Precision of class (0) attack.
- P (1): Precision of class (1) natural.
- R: Recall
- F1: F1-score

Eventually, our research uncovered the way machine learning algorithms provided proper results only if combined with preprocessing techniques, SMOTE and feature selection. A combination of these methods leads to reducing the class imbalance problem; thus, the models can extract the most discriminative patterns from the data and finally increase the smart grids' ability to spot false data injection attempts.

4.1.7 Performance Evaluation of Multi-Class Classification

We train the ExtraTreesClassifier, it is evaluated on the feature-selected and SMOTE-resampled dataset with regard to a number of performance metrics, such as accuracy, classification report, and confusion matrix. The overall accuracy on the test set came out to be 0.9420 for the ExtraTreesClassifier, which indicates strong predictive performance. Further detail in understanding model performance for each class is captured in the classification report as illustrated below.

- Accuracy Score: 0.96
- Precision Score: 0.95
- Recall Score: 0.95
- F1-Score: 0.95

Classification Report

Table 4.11 (ETC) Classification Report on Multi-Class Classification

Class	Precision	Recall	F1-Score
0	0.94	0.93	0.94
1	0.94	0.94	0.94
2	0.98	0.99	0.98

Confusion Matrix

The Confusion Matrix, as seen in Figure 4.9, gives in detail the information regarding the predictions from the model. In class 0, it correctly predicted 8441 cases and mispredicted 490 cases belonging to class 1 and 38 cases belonging to class 2. Class 1 had its cases correctly predicted to be 8419; however, its mis predicted 428 as class 0 and 24 as class 2 cases. Class 2 had its cases correctly predicted to be 8875; however, its mispredicted 25 as belonging to class 0 and 11 as belonging to class 1.

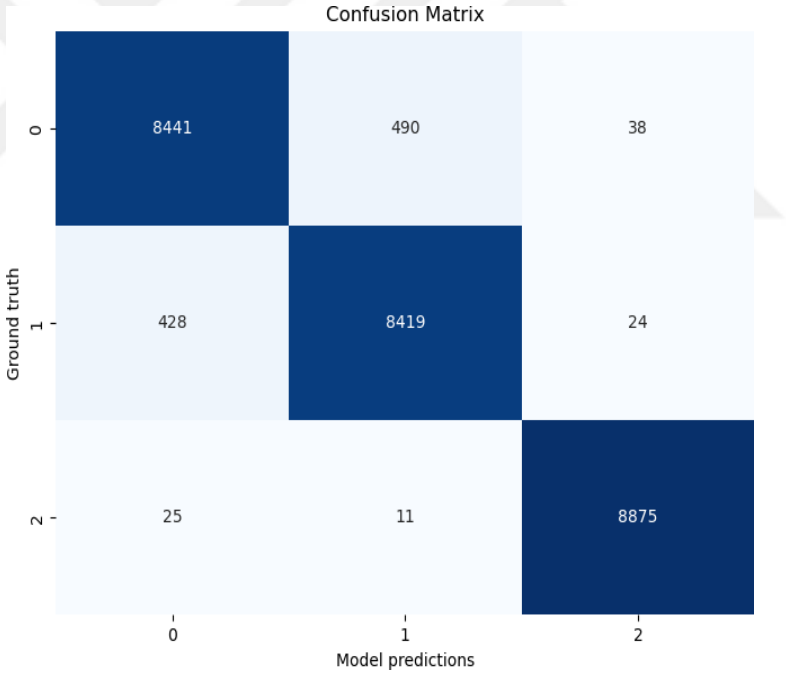

Figure 4.8: (ETC) Confusion Matrix on Multi-Class Classification

Overall, by balancing the dataset with SMOTE, feature selection, and using the ExtraTreesClassifier, promising results are achieved toward the exact classification of Natural, No Events, and Attack events in a multi-class scenario. All of these techniques enhanced the model's ability toward threat detection and enhancement of security and reliability in the smart grid system as a whole.

4.3 DEEP LEARNING CASE STUDY FINDINGS

The most impressive achievements in the detection of false data injection assaults on smart grid phasor measurement units have been found in the various deep learning models examined, including Convolutional Neural Networks, Long Short-Term Memory networks, and hybrid CNN-LSTM model.

4.3.1 CNN Results and Evaluation

The achieved accuracy was 97.36% on the test dataset during the test process of the CNN model. Besides, the evaluation of the performance of the model on the test set was further done with other metrics: precision, recall, and F1-score computed for each class.

The classification report given in Table 4.12 provides detailed metrics for each class.

Table 4.12 CNN Classification Report

Class	Precision	Recall	F1-Score
0	0.97	0.95	0.96
1	0.96	0.97	0.96
2	0.99	0.98	0.98

The results have shown that the architecture of CNN has been effective in capturing those relevant patterns and features from smart grid data, based on which events could be classified as Natural, No Events, and Attacks accurately. The extra preprocessing steps undertaken clipping and standardization added to the stability in training and convergence of the model.

4.3.2 LSTM Results and Evaluation

The achieved accuracy was 95.91% on the test dataset during the test process of the LSTM model. Besides, the evaluation of the performance of the model on the test set was further done with other metrics: precision, recall, and F1-score computed for each class.

The classification report provides detailed metrics for each class as illustrated in Table 4.13.

Table 4.13 LSTM Classification Report

Class	Precision	Recall	F1-Score
0	0.95	0.92	0.94
1	0.94	0.95	0.95
2	0.98	0.97	0.97

This performance classifying smart-grid events based on the LSTM model truly

impresses in that regard; it records a test accuracy of 95.91% with high precision, high recall, and F1-score of all the classes. Such high precision and high recall of every class balance each other; it evidences that the model can fit classes naturally, no events, and attacks without creating false positives or false negatives. However, it is obvious that the LSTM model, in contrast to the CNN model, demonstrates slightly lower accuracy, recall, and F1-scores. The differences are minimal indicating both models do very well on this task.

4.3.3 Hybrid CNN-LSTM Results and Evaluation

The performance of the CNN-LSTM trained model was tested through the test set, which has not been seen before in the training. Hence, the prediction of the model on the test set was made and various evaluation metrics were calculated, which can explain the effectiveness of the proposed model in detecting false data injection in smart grids.

The achieved accuracy was 97.58% on the test dataset during the test process of the CNN-LSTM model. The classification report provides detailed metrics for each class as illustrated in Table 4.14.

Table 4.14 CNN-LSTM Classification Report

Class	Precision	Recall	F1-Score
0	0.98	0.95	0.96
1	0.96	0.98	0.97
2	0.99	1.00	1.00

Besides the test accuracy, this is a good classification report. These results show that the CNN-LSTM model is learned and detects false data injection in the smart grids with a good margin. The model detects instances of class 2 with a good accuracy, recall, and F1-score of 1.00%; class 0 and class 1 also show quite good results.

On the whole, the CNN-LSTM model was able to show its robustness in detection capabilities when it comes to false data injection scenarios, hence its being a promising approach to the enhancement of security and reliability of smart grid systems.

4.3.4 Comparative Analysis

This subsection compares the performances of different deep learning models. Convolutional Neural Networks and Long Short-Term Memory, along with the hybrid CNN-LSTM model, have shown excellent performance in detecting false data injection attacks on smart grid phasor measurement units. Although all models were performing appreciably well, this comparative study brings out the strengths and weaknesses of each.

Table 4.15 Performance summary table for deep learning models

Model	Accuracy	Precision (Range)	Recall (Range)	F1-Score (Range)
CNN	97.36%	0.96 - 0.99	0.95 - 0.98	0.96 - 0.98
LSTM	95.91%	0.94 - 0.98	0.92 - 0.97	0.94 - 0.97
CNN-LSM	97.58%	0.96 - 0.99	0.95 - 1.00	0.96 - 1.00

Convolutional Neural Network (CNN)

- The CNN model has an accuracy of 97.36% for the test dataset.
- Precisions, recalls, and F1 scores were pretty high for all three classes: Natural, No Events, and Attacks from 0.96 to 0.99 in range.
- This CNN architecture showed good ability in capturing important spatial patterns and features of the smart grids' data, hence leading to a high accuracy in classifying events.
- Clipping and standardization were among the preprocessing techniques that helped stabilize the model, leading to better convergence of the model during training.

Long Short-Term Memory (LSTM)

- Additionally, the precision of the model within the test data set with LSTM was at 95.91%, which is slightly lower than that obtained with the CNN model.
- All the classes have high precision, recall, and F1-scores, but slightly lower than the CNN model.
- The LSTM network architecture captures long-term dependencies over the time series data that play a very important role in determining anomalies and attacks for smart grid systems.
- Nevertheless, the fact that the LSTM model demonstrated performance was only a little worse than the CNN model, but confirmed the effectiveness of working with sequential data.

Hybrid CNN-LSTM Model

- The results point out that the hybrid CNN-LSTM model obtained the highest accuracy of 97.57% among all the models, clearly outperforming both CNN and LSTM models on the test dataset.
- The precisions, recalls, and all the three classes' F1-scores were great. Class 2 received an F1 score of 1.00.

- Following on from that, the hybrid model combined the two CNN and LSTM architectures' strengths and resulted in the effective capture of both spatial patterns and temporal dependencies that occur within the smart grid data.
- The hybrid approach has shown the robustness of such a hybrid approach in false data injection scenarios, which indicates the strength in its enhancement of security and reliability for the smart grid.

All the three deep learning approaches, though all very promising, showed that the hybrid approach of CNN-LSTM was the more effective way to detect false data injection attacks on a smart grid. Through the features learned by both the CNN and LSTMs, this hybrid model can effectively learn and derive spatial and temporal features, hence better detection capabilities for such hybrid models.

4.4 CONCLUSION

This chapter has presented in detail the evaluation of different machine learning and deep learning models used for determining performance in detecting false data injection in smart grids. The main findings of this assessment are:

- Model Performance: The CNN-LSTM hybrid model came out best in all cases with an accuracy score of 97.58% in precision, recall, and F1 score. This would further imply that the model can learn complex patterns and dependencies from datasets.
- Impact of Preprocessing: Feature selection and use of feature scaling this improved the stability and performance of the model. In a nutshell, the SMOTE application in balancing the dataset is most certainly needed such that it can further generalize well across the classes.
- Comparative Insights: The Extra Trees Classifier model was proper, having dis- played good accuracy at 94.20% in multi classification, although lower than the accuracy in the deep learning models. The CNN model showed an accuracy of 97.36%, while the accuracy for the LSTM model was 95.91%. The effectiveness of the deep approaches has been shown in relation to this aspect.

In conclusion, this chapter has provided adequate illustrations of the ability of deep learning models, The CNN model achieved excellent results, attaining impressive precisions, recalls and F1-scores in all classes. The LSTM model performed well too though with slightly

lesser but still remarkable results relative to CNN, combining the two CNN and LSTM in a hybrid model yielded better outcomes. Furthermore, the effect of preprocessing on model performance was explicitly explained, which includes SMOTE, feature selection and scaling.

The results prove that deep learning techniques, and in this case, a hybrid CNN- LSTM, can be effectively applied to the detection of FDI attacks on smart grid PMUs. In general, such models enhance cyber-security and promote smart grid resilience by precisely identifying and mitigating such attacks to guarantee integrity and reliability in a power grid.

5 DISCUSSION

The major objective of this research was to evaluate the performance of various models in machine learning and deep learning for the detection of false data injection attacks in smart grids. The research sought to determine which model and which preprocessing techniques would give it the greatest accuracy and reliability in such important infrastructures. Key results outlined in the previous chapter included outstanding performance with CNN-LSTM, the importance of preprocessing techniques like feature selection and SMOTE, and comparative results for several other models, such as Extra Trees Classifier, CNN, and LSTM. Besides, a number of experiments were conducted in the machine learning binary classification task by means of Random Forest and Extra Trees Classification for class imbalance, after which it was concluded that the best approach is SMOTE in association with feature selection. This preprocessing strategy was further applied to deep learning models, along with clipping and standard scaling.

This chapter discusses the practical implications of these findings and later delves into possible challenges and limitations in its real-world application and further research for future work.

5.1 INTERPRETATION OF KEY FINDINGS

The evaluation of machine learning and deep learning models yielded significant insights into their effectiveness for FDIA detection:

- **Superior Performance of the CNN-LSTM Hybrid Model:**

Among all models applied, CNN-LSTM showed the highest accuracy of 97.58%. One of the possible reasons for this outstanding performance is related to the spatial features extracted by the convolutional layers and how that retains the temporal dependencies of the LSTM layers. We validated the proposed model performance concerning other hybrid models based on empirical evidence. Indeed, recent studies about complex tasks show that a hybrid model often outperforms a single-architecture model (Regev et al., 2022; Yang et al., 2023).

- **Effectiveness of Preprocessing Techniques:**

The preprocessing steps, in the form of feature selection, scaling, and the application of SMOTE, greatly improved the performance of the models. Several experiments were run with Random Forest and Extra Trees Classification, out of which it was found that SMOTE with feature selection gave the best results in the case of class imbalance. This preprocessing strategy improved the stability and performance of the models and was thus followed in the deep learning experiments with further steps of clipping and standard scaling. Such preprocessing techniques are important in the real world, considering the imbalanced nature and noisy data normally associated with the data (Xu, Li, & Sun, 2021).

- **Comparative Performance of Models:**

Even though Extra Trees Classifier shows 94.20% accuracy, it is marginally surpassed by deep learning models. The best accuracy is shown by the CNN model at 97.36%, and the LSTM model followed closely at 95.91%. The comparison reveals the effectiveness of deep learning approaches, especially in com- plex patterns in large datasets typical for smart grid environments (Keçeci et al., 2023; Takiddin et al., 2023).

These findings collectively address the research questions in showing that hybrid deep-learning models, especially the combination of CNN and LSTM, are much better in terms of performance when detecting FDIA.

5.2 PRACTICAL IMPLICATIONS

The findings from this study have several practical implications for the deployment of FDIA detection systems in smart grids:

- **Enhanced Detection Capabilities**

The superior performance of the CNN-LSTM hybrid model indicates great potential for real-world use in smart-grid systems. Precisely detecting FDIAs, this model will enhance both security and reliability against attacks that may cause operational disruption, huge damages, and severe consequences.

- **Preprocessing and Model Training**

The most important to mention about this preprocessing is feature selection, scaling, and using the SMOTE method to oversample the minority class. Therefore, the meticulous preparations of the dataset aim at building very effective systems for FDIA detection when the dataset is sufficiently balanced and informative. The results are an improvement in both model performance and generalizability.

- **Early Warning Systems and Real-time Monitoring**

The CNN-LSTM hybrid model and all pre-processing techniques can be integrated into the smart grid monitoring and control system. In other words, this can be set up as a real-time anomaly detection module that constantly analyzes streams of data from different sensors and measurement units scattered all over the grid. This integration would enable early detection of possible FDIAs so that mitigation and response actions can be pursued in time for the purpose of minimizing any impact on the operation of the grid and ensuring no interruption of power.

5.3 POTENTIAL CHALLENGES AND LIMITATIONS

While the results are promising, there are a number of challenges and limitations which need to be addressed while deploying these models in real smart grid environments.

- **Computational Resources and Infrastructure:**

Training and deployment of such deep learning models as CNN-LSTM may be computationally expensive and require substantial computational infrastructure. For this reason, some utilities or grid operators may be challenged to pursue them, specifically with smaller resource bases or legacy infrastructure. Doing so in an efficient and scalable manner across distributed smart grid systems might require substantial hardware and software infrastructure investments.

- **Evolving Cyber Threats and Adversarial Attacks:**

Cyber threats towards the smart grid domain are advancing daily, and the adversaries might come up with sophisticated methodologies to either bypass or mislead the FDIA detection system. Therefore, it is a matter of great concern whether the implemented CNN-LSTM model is robust under the outcomes of new threats and potential adversarial attacks. This must be done by updating and consistent retraining of this model.

- **Integration and Interoperability:**

The direct integration of the CNN-LSTM model and preprocessing techniques into the already existing smart grid monitoring and control systems raises some concerns related to interoperability, data exchange protocols, and compatibility with the legacy systems. Seamless integration and effective communication between the FDIA detection system and other components of the smart grid infrastructure are necessary in view of the timely detection and mitigation of potential threats.

5.4 RECOMMENDATIONS FOR FUTURE RESEARCH

Even though this work contributes a lot to the field of FDIA detection in smart grids, several investigation and research could be further explored:

- **Real-world Deployment and Validation:**

While the performance of the models can be tested with the help of this simulation study, there is a considerable need to deploy and validate the models in accurate real operation grid environments. A pilot or field test might reveal essential challenges of practical concern with the performance and scalability of the solution under real-world conditions.

- **Integration with Existing Smart Grid Infrastructure:**

Future research must be directed to seamless integration of the proposed solution with the existing smart-grid monitoring and control systems. The development of standardized interfaces, data exchange protocols, and compatibility measures would enable practical deployment and interoperability with legacy systems in a way that assures timely detection and mitigation of threats.

- **Scalability and Computational Optimization:**

This would then lead to researching CNN-LSTM scalability and computational optimization techniques in deep learning models for practical deployment in resource-constrained environments. This could be brought about by researching model compression, probably carding quantization, or even distributed training strategies for better computational efficiency aiming for broader deployment.

The findings from this chapter, and the implications derived in the evaluation of machine learning and deep learning models for false data injection attack detection in smart grids, point toward the superiority of performance for the CNN-LSTM hybrid model, the critical role of preprocessing techniques, and some of the practical applications. The challenges of deploying these models in real-world settings have been discussed. Solutions to challenges identified and future research directions can lead to significant development in the direction of effective and robust FDIA detection systems for smart grids.

6 CONCLUSION

This research focused on the evaluation of the performance of different machine learning and deep learning models specifically designed for FDIA detection, using data harvested from Phasor Measurement Units. The motivation behind this research is the enhancement of security and reliability in regard to critical infrastructures in smart grids for identifying actual FDIA that may cause operational disruption with severe consequences.

In this regard, through extensive experimentation and evaluation, the superiority of the CNN-LSTM hybrid model for the detection of FDIAs has been unraveled. This hybrid model, developed by integrating the strengths of CNN and LSTM, showed better performance with an accuracy of 97.58%. The CNN layers of the model extracted spatial features from the data of the smart grid effectively, and the LSTM layers extracted long-term temporal dependencies to enable the model to learn complex patterns and classify events correctly as natural, no events, or attacks.

Preprocessing techniques such as feature selection, scaling, and the use of Synthetic Minority Oversampling Technique, further proved to be critical in improving the performances of the models and the generalization of the results. Such techniques addressed common problems of class imbalance and noisy data in real smart grid environments.

Comparative analysis of different models, such as the Extra Trees Classifier, CNN, and LSTM, provided insights into the respective strength and weakness of each of these models. Though generally the deep learning models outperformed traditional machine learning models, the hybrid model of CNN-LSTM came out as the most appropriate solution for FDIA detection.

From a practical standpoint, these precious results, obtained from the proposed research, contribute to deploying FDIA detection systems in smart grids. The hybrid coupling model of CNN-LSTM with the best performance and preprocessing techniques, in a way, enhances detection capability with early warning systems and real-time monitoring of the potential FDIA. This, in turn, means that the impact of the attacks may be reduced while keeping the power supply continuous.

However, some of the potential issues or limitations with respect to deploying these models within real-world smart grid environments should not be ignored. These include problems of

computational resource requirements, evolving cyber threats, and adversarial attacks and integration and interoperability issues that may arise with regards to existing smart grid infrastructure.

Addressing these challenges and further developing FDIA detection in smart grids, several future research directions have been proposed: real-world deployment and validation of models, their smooth integration with the already existing infrastructure of smart grids, and research in the field of scalability and computational optimization techniques for deep learning models.

In the final analysis, this research work made tremendous contributions toward the subject area of FDIA detection in smart grids through the illustration of the effectiveness of the CNN-LSTM hybrid model and preprocessing techniques. The findings and recommendations of this study point to further research for enhancement in the security and reliability of smart grid systems, which are critical infrastructures, to be resilient against such cyber threats.

REFERENCES

[1] Aksan, F., Li, Y., Suresh, V., Janik, P. (2023). CNN-LSTM vs. LSTM-CNN to Predict Power Flow Direction: A Case Study of the High-Voltage Subnet of Northeast Germany. Sensors, 23(2), 901.

[2] Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A., Aljaaf, A. J. (2020). A systematic review on supervised and unsupervised machine learning algorithms for data science. Supervised and unsupervised learning for data science, 3-21.

[3] Ashok, A., Govindarasu, M., & Ajjarapu, V. (2018). Online detection of stealthy false data injection attacks in power system state estimation. IEEE Transactions on Smart Grid, 9(3), 1636-1646.

[4] Ayad, A., Farag, H. E., Youssef, A., & El-Saadany, E. F. (2018). Detection of false data injection attacks in smart grids using recurrent neural networks. In 2018 IEEE power energy society innovative smart grid technologies conference (ISGT) (pp. 1-5). IEEE.

[5] Basumallik, S., Ma, R., & Eftekharnejad, S. (2019). Packet-data anomaly detection in PMU-based state estimator using convolutional neural network. International Journal of Electrical Power Energy Systems, 107, 690-702.

[6] Biswal, C., Sahu, B. K., Mishra, M., & Rout, P. K. (2023). Real-time grid monitoring and protection: A comprehensive survey on the advantages of phasor measurement units. Energies, 16(10), 4054.

[7] Chalmers, C., Hurst, W., Mackay, M., & Fergus, P. (2015, October). Smart health monitoring using the advanced metering infrastructure. In 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (pp. 2297-2302). IEEE.

[8] Chevalier, G. (2018). LARNN: Linear attention recurrent neural network. arXiv preprint arXiv:1808.05578.

[9] Deng, R., Xiao, G., & Lu, R. (2017). Defending against false data injection attacks on power system state estimation. IEEE Transactions on Industrial Informatics, 13(1), 198-207.

[10] Ding, W., Xu, M., Huang, Y., Zhao, P., & Song, F. (2021). Cyber attacks on PMU placement in a smart grid: Characterization and optimization. Reliability Engineering System Safety, 212, 107586.

[11] Ding, Y., Ma, K., Pu, T., Wang, X., Li, R., & Zhang, D. (2021). A deep learning-based classification scheme for false data injection attack detection in power system. Electronics, 10(12), 1459.

[12] G. Hug & J. A. Giampapa. (2012). Vulnerability assessment of AC state

estimation with respect to false data injection cyber-attacks. IEEE Transactions on Smart Grid, 3(3), 1362-1370.

[13] He, Y., Li, L., Qian, H., & Yao, S. (2022, December). CNN-GRU based false data injection attack detection method for power grid. In 2022 2nd International Conference on Electrical Engineering and Control Science (IC2ECS) (pp. 408-411). IEEE.

[14] Jawhar, J. (2020). Stealthy attack detection using convex optimization-based RPCA algorithm. Electric Power Systems Research, 187, 106418.

[15] Keçeci, C., Davis, K. R., & Serpedin, E. (2023). Federated learning-based distributed localization of false data injection attacks on smart grids. arXiv preprint arXiv:2306.10420.

[16] Koščo, J., Tauš, P., & Šimon, P. (2019). Negative impacts of photovoltaic electric station operation for distribution of electrical energy in Sobrance. Power, 6, 8.

[17] Lal, M. D., & Varadarajan, R. (2023). A review of machine learning approaches in synchrophasor technology. IEEE Access.

[18] Liu, X., & Li, Z. (2019). False data attack models, impact analyses, and defense strategies for power system state estimation. IEEE Transactions on Smart Grid, 10(4), 4324-4334.

[19] O. Kosut, L. Jia, R. J. Thomas, & L. Tong. (2011). Malicious data attacks on the smart grid. IEEE Transactions on Smart Grid, 2(4), 645-658.

[20] Phadke, A. G., & Thorp, J. S. (2008). Synchronized phasor measurements and their applications. Springer Science Business Media.

[21] Phadke, A. G., & Thorp, J. S. (2017). Synchronized phasor measurements and their applications. Springer.

[22] Phung, V. H., & Rhee, E. J. (2019). A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. Applied Sciences, 9(21), 4500.

[23] Regev, Y. A., Vassdal, H., Halden, U., Catak, F. O., & Cali, U. (2022, November). Hybrid AI-based anomaly detection model using phasor measurement unit data. In 2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain Beyond (iGETblockchain) (pp. 1-6). IEEE.

[24] Rigatti, S. J. (2017). Random forest. Journal of Insurance Medicine, 47(1), 31-39.

[25] Sa'ad, H. H. Y., Ali, A. M. Z., & Al-Hmadi, A. (2023). False Data Injection Attacks (FDIA) detection by deep learning techniques in smart grids: Survey. Alrazi University Journal of Computer Science and Technology, 1(1).

[26] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural

networks, 61, 85-117.

[27] Sikri, A., & Singh, A. (2023, April). Classification of false data injection attacks in smart grid using multi-label deep learning. In 2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE) (pp. 1-6). IEEE.

[28] Stoustrup, J., Annaswamy, A., Chakrabortty, A., & Qu, Z. (2019). Smart grid control.

[29] Sundararajan, A., Khan, T., Moghadasi, A., & Sarwat, A. I. (2019). Survey on synchrophasor data quality and cybersecurity challenges, and evaluation of their interdependencies. Journal of Modern Power Systems and Clean Energy, 7(3), 449-467.

[30] Takiddin, A., Atat, R., Ismail, M., Boyaci, O., Davis, K. R., & Serpedin, E. (2023). Generalized graph neural network-based detection of false data injection attacks in smart grids. IEEE Transactions on Emerging Topics in Computational Intelligence.

[31] Tian, J., Wang, B., Wang, Z., Cao, K., Li, J., & Ozay, M. (2021). Joint adversarial example and false data injection attacks for state estimation in power systems. IEEE Transactions on Cybernetics, 52(12), 13699-13713.

[32] Tuyizere, D., & Ihabwikuzo, R. (2023). Machine learning to detect cyber-attacks and discriminate the types of power system disturbances. arXiv preprint arXiv:2307.03323.

[33] Van Houdt, G., Mosquera, C., & Nápoles, G. (2020). A review on the long short-term memory model. Artificial Intelligence Review, 53(8), 5929-5955.

[34] Wang, W., & Lu, Z. (2013). Cyber security in the smart grid: Survey and challenges. Computer networks, 57(5), 1344-1371.

[35] Wu, Y., Wang, Q., Guo, N., Tian, Y., Li, F., & Su, X. (2023). Efficient multi-source self-attention data fusion for FDIA detection in smart grid. Symmetry, 15(5), 1019.

[36] Xu, L., Li, X., & Sun, Y. (2021). Detection of false data injection attacks in smart grid based on machine learning. In Advances in Artificial Intelligence and Security: 7th International Conference, ICAIS 2021, Dublin, Ireland, July 19-23, 2021, Proceedings, Part III 7 (pp. 191-203). Springer International Publishing.

[37] Yang, H., Cao, R., Pan, H., & Jin, J. (2023, May). Deep learning-based hybrid detection model for false data injection attacks in smart grid. In 2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems (ICPS) (pp. 1-7). IEEE.

[38] Yang, Q., Yang, J., Yu, W., An, D., Zhang, N., & Zhao, W. (2013). On false data-injection attacks against power system state estimation: Modeling and countermeasures. IEEE Transactions on Parallel and Distributed Systems, 25(3),

717-729.

[39] Y. Liu, P. Ning, & M. K. Reiter. (2011). False data injection attacks against state estimation in electric power grids. ACM Transactions on Information and System Security, 14(1), 1-33.

[40] Y. Zou, J. Zhu, X. Wang, & L. Hanzo. (2016). A survey on wireless security: Technical challenges, recent advances, and future trends. Proceedings of the IEEE, 104(9), 1727-1765.

[41] Zanni, L. (2017). Power-system state estimation based on PMUs: Static and dynamic approaches-from theory to real implementation (No. 7665). EPFL.

[42] Zhang, G., Gao, W., Li, Y., Guo, X., Hu, P., & Zhu, J. (2023). Detection of false data injection attacks in a smart grid based on WLS and an adaptive interpolation extended Kalman filter. Energies, 16(20), 7203.

BIOGRAPHY

Khalid Goun is a Master's student in Computer Engineering at Beykoz University in Istanbul, Turkey. He is passionate about the intersection of technology and data, with a focus on developing expertise in Cybersecurity, Deep and Machine Learning. Khalid's academic journey is marked by his commitment to continual learning and hands-on experience in these fields.

Khalid earned a University Diploma of Technology in Electrical Engineering and Industrial Data Processing from Mohammed V University in Rabat, Morocco. He then completed his Bachelor's degree in Diagnostics and Maintenance of Automotive Onboard Electronic Systems at the same institution. These studies provided Khalid with a solid foundation in technology and engineering, which he has further developed in his graduate work.

Khalid has also pursued certifications in Data Analytics, Machine Learning, and Cybersecurity from renowned institutions like Amazon Web Services, Duke University, and Google. His skills include cybersecurity, machine learning, deep learning, and interpersonal skills.

In recognition of his academic excellence, Khalid received the Director's Honor Certificate for achieving a high GPA during the Fall 2022-2023 semester. He aspires to leverage his skills and knowledge to make significant contributions to the fields of cybersecurity, deep and machine learning.