Journal of Theoretical and Applied Information Technology
15th May 2024. Vol.102. No 9

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

INTEGRATING AI FOR ENHANCED FAULT DETECTION IN INDUSTRIAL SYSTEMS: EVALUATING MACHINE AND DEEP LEARNING APPROACHES IN THE INDUSTRY 4.0 AND IIOT ERA

YOUSSEF ZERGUIT[1], YOUNES HAMMOUDI[2], MOSTAFA DERRHI[3]

[1,3]LTI Research Laboratory, ENSAT, Abdelmalek Essaadi University, Tangier, Morocco

[2]LAMON2E Research Laboratory, FSO, Mohammed First University, Oujda, Morocco

E-mail : [1]youssef.zerguit@etu.uae.ac.ma, [2]y.hammoudi@ump.ac.ma, [3]m.derrhi@uae.ac.ma

ABSTRACT

In the transformative landscape of Industry 4.0, the integration of Artificial Intelligence (AI) within the Industrial Internet of Things (IIoT) has emerged as a cornerstone for advancing operational efficiency and reliability. This paper explores the application of various AI methodologies, including both machine learning and deep learning approaches, to enhance fault detection in industrial systems, particularly focusing on three-phase electrical systems. Utilizing an integrated system architecture comprising Power Monitoring Units (PMUs) and advanced computational units, we implement and evaluate a suite of AI models such as Support Vector Machines (SVM), Decision Trees, K-Nearest Neighbors (KNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) networks. Our comprehensive analysis reveals the nuanced capabilities and performance metrics of these models in the context of real-time fault detection, thereby providing pivotal insights for deploying AI-driven diagnostics in industrial settings.

Keywords: *Three-Phase Systems, Industry 4.0, Industrial Internet Of Things (Iiot), Artificial Intelligence (Ai), Machine Learning, Deep Learning, Support Vector Machines (Svm), Decision Trees, K-Nearest Neighbors (Knn), Artificial Neural Networks (Ann), Convolutional Neural Networks (Cnn), Recurrent Neural Networks (Rnn), Long Short-Term Memory (Lstm), Gated Recurrent Units (Gru).*

1. INTRODUCTION

The advent of Industry 4.0 marks a pivotal shift in manufacturing paradigms, characterized by the integration of advanced digital technologies and automation within industrial ecosystems [1]. This transformative era is propelled by the convergence of cyber-physical systems, the Internet of Things (IoT), and the Industrial Internet of Things (IIoT), heralding unprecedented levels of operational efficiency, flexibility, and automation in manufacturing processes [2].

Central to the Industry 4.0 revolution is the IIoT, which embodies a network of interconnected devices, sensors, and systems designed to communicate and exchange data [3]. This interconnectedness facilitates real-time monitoring and decentralized decision-making, enabling predictive maintenance, enhanced operational efficiency, and minimized downtime [4]. The IIoT serves as the backbone of smart factories, where machines and systems autonomously communicate and cooperate with each other and with humans in real-time [5].

Three-phase electrical systems, the cornerstone of industrial operations, are undergoing significant advancements in the context of Industry 4.0 [6]. These systems are essential for their efficiency in power delivery and their role in driving various industrial machinery and processes. The reliability and seamless performance of three-phase systems are paramount, given their critical function in the industrial sector. As such, the detection and diagnosis of faults within these systems are crucial to averting operational disruptions and ensuring sustained productivity [7].

In this digital era, Artificial Intelligence (AI) emerges as a transformative force, enhancing the capabilities for fault detection and diagnosis in industrial systems [8]. Machine Learning (ML) algorithms, including Support Vector Machines (SVM), Decision Trees, and K-Nearest Neighbors

(KNN), lay the groundwork for pattern recognition and classification, which are vital for identifying anomalies and irregularities within operational processes [9].

Moreover, the evolution of Deep Learning (DL) has introduced advanced neural network architectures such as Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory networks (LSTM), and Gated Recurrent Units (GRU) [10]. These models are adept at analyzing complex, high-dimensional data prevalent in industrial settings, capturing temporal and spatial dependencies, and offering superior accuracy in fault detection and predictive maintenance.

The synergy of AI technologies and the IIoT framework enables a nuanced, real-time analysis of data from three-phase systems, facilitating the early detection of faults and the prediction of potential failures before they manifest. This proactive approach to maintenance underscores the Industry 4.0 vision, emphasizing the integration of interconnectedness, automation, and data analytics to achieve unparalleled efficiency and reliability in industrial operations.

Despite these advancements, effectively detecting and diagnosing faults in three-phase electrical systems remains a significant challenge, crucial for maintaining the efficiency and integrity of industrial operations. The complexity and variability of these systems necessitate advanced solutions that can adapt to dynamic operational conditions. This study addresses the gap by exploring how AI can be integrated within the IIoT framework to enhance fault detection in three-phase systems, thereby contributing to the robustness and sustainability of Industry 4.0 environments.

The research is motivated by the need to bridge this gap, aiming to enhance the accuracy and efficiency of fault detection mechanisms. Despite the technological strides in Industry 4.0, reliable fault detection and diagnosis within three-phase electrical systems remain challenging, underscoring the need for innovative solutions that enhance the precision and speed of these processes in industrial environments. The study will explore how AI methodologies, when integrated with the IIoT framework, can improve the accuracy and efficiency of fault detection in three-phase electrical systems and will examine the comparative advantages and limitations of different AI models in the context of fault detection and diagnosis in these systems, aligning with the operational demands of Industry 4.0.

In conclusion, the integration of IIoT, AI, and advanced digital technologies marks a significant step in the transformative journey of the manufacturing sector towards Industry 4.0 and beyond to Industry 5.0. This paradigm shift not only redefines the operational landscape of industries but also anticipates an era of enhanced human-machine collaboration, customization, and sustainability.

2. BACKGROUND

2.1 Industry 4.0

Industry 4.0, the fourth industrial revolution, integrates advanced digital technologies into manufacturing, heralding the era of smart factories. These factories leverage cyber-physical systems and the Internet of Things (IoT) to create networks of machines and systems that communicate and cooperate in real time, enabling advanced automation, data exchange, and intelligent decision-making. This integration facilitates the creation of a digital twin of the manufacturing process, allowing for simulation, analysis, and optimization in a virtual environment [11], [12].

2.2 Industrial Internet of Things (IIoT)

The IIoT is an extension of the IoT focused on industrial applications, involving a network of intelligent devices connected to form systems that monitor, collect, exchange, and analyze data. Each device in the IIoT is equipped with sensors and actuators that interact with the physical world and the cyber-physical systems. This data is then used to optimize operations, predict maintenance needs, and increase overall efficiency through machine learning algorithms and real-time data analytics [13], [14].

2.3 Three-Phase Electrical Systems

Three-phase electrical systems are widely used in industrial settings due to their ability to deliver consistent, high power with greater efficiency than single-phase systems. They consist of three alternating currents of the same frequency and amplitude, each phase offset by 120 degrees. This configuration allows for a more balanced load, reducing the size and cost of wiring and electrical equipment. In Industry 4.0, these systems are crucial for powering machinery and are often monitored using advanced sensor technology to predict and prevent failures [15], [16].

2.4 Support Vector Machines (SVM)

SVMs are a set of supervised learning methods used for classification, regression, and outliers detection. In an SVM model, each data item

is plotted as a point in n-dimensional space (with n being the number of features), with the value of each feature being the value of a particular coordinate. The SVM algorithm then constructs a hyperplane in this multidimensional space that best separates the different classes. The optimal hyperplane is the one with the largest margin between the two classes, with support vectors being the data points closest to the hyperplane [17].

2.5 Decision Trees (DT)

Decision Trees are a non-parametric supervised learning method used for classification and regression. A decision tree is built from a root node and involves partitioning the data into subsets that contain instances with similar values (homogeneous). DTs use entropy and information gain to decide which feature (and value) to split on at each step. The goal is to create branches that lead to nodes with the highest possible purity, meaning the nodes are as homogeneous as possible [18].

2.6 K-Nearest Neighbors (KNN)

The KNN algorithm assumes that similar things exist in close proximity, known as the proximity assumption. In KNN, data points are classified by a majority vote of their neighbors, with the data point being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). KNN works by finding the distances between a query and all the examples in the data, selecting the specified number of examples (k) closest to the query, then voting for the most frequent label (in classification) or averaging the labels (in regression) [19].

2.7 Artificial Neural Networks (ANN)

ANNs are computing systems inspired by the biological neural networks that constitute animal brains [20]. An ANN is composed of a large number of interconnected processing elements (neurons) working in unison to solve specific problems. ANNs are configured for specific applications, such as pattern recognition or data classification, through a learning process involving adjustments to the synaptic connections between neurons [21].

2.8 Convolutional Neural Networks (CNN)

CNNs are a class of deep neural networks, most commonly applied to analyzing visual imagery. They are composed of multiple layers of neurons that process pieces of the input image, called receptive fields [22]. The layers of a CNN typically include convolutional layers, pooling layers, and fully connected layers. Convolutional layers apply a convolution operation to the input, passing the result to the next layer [23]. This process allows the network to build a hierarchy of concepts, from simple to complex, facilitating complex image recognition tasks [24].

2.9 Recurrent Neural Networks (RNN)

RNNs are a class of neural networks where connections between nodes form a directed graph along a temporal sequence, allowing it to exhibit temporal dynamic behavior [25]. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition [26].

2.10 Long Short-Term Memory (LSTM)

LSTMs are a special kind of RNN capable of learning long-term dependencies. They were introduced to avoid the long-term dependency problem, allowing the network to learn when to forget previous hidden states and when to update hidden states given new information [27]. LSTMs are particularly useful for modeling sequences and time-series data, where the timing and order of events are critical [28].

2.11 Gated Recurrent Units (GRU)

GRUs are a gating mechanism in recurrent neural networks, introduced as a variation of the LSTM with a simplified structure. Like LSTMs, GRUs are designed to help capture dependencies for sequences of data [27]. However, GRUs combine the forget and input gates into a single "update gate" and merge the cell state and hidden state, resulting in a more straightforward model that can perform on par with LSTM on certain datasets but with fewer parameters [29].

3. RELATED WORKS

In the evolving landscape of Industry 4.0, the symbiosis of Artificial Intelligence (AI) and Machine Learning (ML) with the Industrial Internet of Things (IIoT) has marked a transformative epoch in the domain of intelligent systems. Particularly, the incorporation of AI and ML algorithms into predictive maintenance and fault detection mechanisms stands at the forefront of scholarly exploration, especially within the intricate web of three-phase systems. This segment aims to amalgamate insights from seminal research, highlighting the innovative methodologies and strategies employed to harness the power of AI and ML in preempting and identifying faults. The discourse extends beyond the mere application of these technologies, venturing into the realm of IIoT-

driven industrial practices, where the integration of smart, connected devices elevates the potential for system efficiency and reliability. Through a comprehensive review of the existing literature, this section endeavors to elucidate the pivotal role of AI and ML in redefining fault detection paradigms, thereby setting a solid foundation for our in-depth analysis within the dynamic IIoT ecosystem.

Almasoudi et al. [30] integrated artificial intelligence in modern power grids, employing advanced hybrid machine learning models such as CNN-RNN, CNN-GRU, and CNN-LSTM for fault prediction and detection. Their approach demonstrated the potential of these models in enhancing grid resilience by enabling faster and more accurate fault identification and remediation, showcasing a significant improvement in the efficiency and reliability of power distribution systems.

Pietrzak and Wolkiewicz [31] applied continuous wavelet transform (CWT) and artificial intelligence techniques for diagnosing PMSM stator winding faults. By utilizing CWT for signal processing and applying various machine learning algorithms for fault classification, they highlighted the effectiveness of these algorithms in accurately classifying stator winding faults, which is crucial for the maintenance and reliability of PMSM systems.

Farrar et al. [32] provided a comprehensive review of the application of AI and ML in grid-connected wind turbine control systems. Their review underscored the critical role of AI and ML algorithms in optimizing wind farm power generation and designing efficient control schemes, thereby enhancing the performance and reliability of wind turbines and contributing to sustainable energy solutions.

Polenghi et al. [33] proposed a framework for fault detection in collaborative robots using a modular structure of unsupervised AI algorithms. This framework was shown to effectively handle different trajectories and notify operators of cobots' unhealthy states, enabling first-level maintenance and enhancing operational efficiency, which is vital for the automation and flexibility of manufacturing processes.

Albertin et al. [34] presented a real-time novelty recognition framework using machine learning models to automate anomaly detection in factory systems. Their framework, which utilized a sensor fusion approach and optimized software system, enhanced computation scalability and response time for novelty detection, offering a viable solution for predictive maintenance and ensuring continuous operation in Industry 4.0 environments.

Preethi et al. [35] introduced a supervised classification method based on the Logit boosting algorithm for fault detection in three-phase induction motors. By analyzing rotor current and vibration signals to extract features and using the Logit boosting algorithm for fault classification, their method demonstrated superior performance in identifying and classifying motor faults, providing a reliable tool for early detection and maintenance of induction motors.

Hong et al. [36] employed convolutional neural networks for intelligent fault detection in power systems. By transforming fault data into images for CNN processing, their approach was able to classify fault types and locations under various conditions, significantly reducing the complexity and effort required for fault analysis and enhancing the safety and reliability of power systems.

Badr et al. [37] introduced a non-invasive CNN-based approach for fault diagnosis in three-phase induction motors, focusing on stator-related faults. Their method, which uses external measurements to feed into CNNs, demonstrated high accuracy in diagnosing faults, offering a cost-effective and efficient solution for motor maintenance and contributing to the advancement of diagnostic tools in the field of electrical engineering.

Building on the diverse contributions of previous works, which have underscored the potential of AI-based solutions in fault detection across various electrical systems and industries, our study aims to delve deeper within the context of Industry 4.0, particularly through the Industrial Internet of Things (IIoT) framework. Unlike prior research that often concentrated on specific AI models or fault types, we conduct a comprehensive analysis of a broad spectrum of machine learning and AI models. This includes Support Vector Machines (SVM), Decision Trees, K-Nearest Neighbors (KNN), Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRU). Our research is distinctive in its inclusivity, considering any type of three-phase load and even accommodating single-phase scenarios, thus extending the applicability of our findings across a broader range of industrial settings.

Moreover, our solution addresses a multiclass classification problem, testing on an extensive array of faults, specifically 57 different types, which surpasses the scope of fault types considered in previous studies. This comprehensive fault coverage ensures a more robust and versatile fault detection system. Additionally, our approach

uniquely focuses on electrical parameters, 47 in total, as its input, which streamlines the data acquisition process and enhances the model's applicability and ease of integration into existing industrial systems. While previous studies have laid a solid foundation by demonstrating the effectiveness of AI and ML in fault detection, they often limit their focus to specific fault types or aspects of electrical systems. Our study seeks to overcome these limitations by providing a holistic and inclusive solution that not only covers a broader spectrum of fault types but also supports a wider variety of load conditions within the IIoT framework of Industry 4.0. This holistic approach not only advances fault detection technologies but also aligns with the evolving needs of modern industrial environments, where versatility, comprehensiveness, and seamless integration are paramount.

4. PROPOSED METHODOLOGY

4.1 System Architecture

Our integrated system architecture is meticulously engineered to augment fault detection capabilities within three-phase electrical systems. At the core of this architecture are Power Monitoring Units (PMUs), strategically deployed ahead of each load to ensure comprehensive monitoring. These PMUs are interconnected with a Main Control Unit (MCU), which, in turn, is linked to a Main Power Monitoring Unit (MPMU). This configuration offers a holistic view of the electrical system's status, as illustrated in Figure 1. The architecture's design is pivotal for precise fault identification at both individual load levels and system-wide.

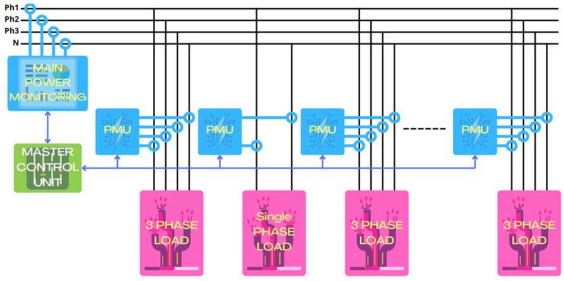

Figure 1: Schematic Diagram of System Architecture with PMUs and MCU Integration

The PMUs, embodying advanced Industrial Internet of Things (IIoT) technology, are adept at measuring a wide array of electrical parameters, including voltage, current, power factor, and the spectrum of power components (active, reactive, and apparent). Embedded within these units are AI fault detection models (AI-FDMs) that operate in real-time, powered by Jetson Nano boards. The selection of Jetson Nano boards as the computational backbone is attributed to their capacity to handle the demands of AI models efficiently, thereby facilitating real-time analytics.

Data acquisition is twofold: firstly, the PMUs process information locally via the embedded AI-FDM, ensuring immediate fault detection. Simultaneously, this data is transmitted to the MCU, fostering an environment for extended analysis and model refinement. An additional layer of analysis is provided by the AI-FDM within the MCU, which also processes data from the MPMU. This layered approach not only assures rapid fault identification but also enhances the system's learning capabilities, promoting continuous improvement through the iterative analysis of aggregated data.

4.2 Data Handling and Preprocessing

The PMUs are tasked with the continuous monitoring and collection of data pivotal for the accurate detection of faults within the system. This collected data is subjected to a series of preprocessing steps, each designed to enhance its quality and suitability for analysis by AI models:

- Filtering: Initial data processing involves the removal of noise and extraneous information. This step is crucial for refining the data, thereby improving the precision and reliability of the model's outputs.
- Normalization: Given the heterogeneity of data sources, normalization procedures are applied to standardize data ranges. This consistency is vital for the effective training of AI models, ensuring uniform interpretation of input values across diverse datasets.
- Feature Extraction: This critical phase involves identifying and isolating features strongly indicative of various fault conditions. The effectiveness of the AI models in fault detection and prediction hinges on the quality and relevance of these extracted features.

For real-time fault detection, AI models are deployed directly onto the Jetson Nano boards integrated within the PMUs. This deployment strategy ensures rapid fault identification with minimal latency, enabling the system to take swift corrective actions. This localized processing not only bolsters the system's operational reliability but also its overall safety.

The models chosen for deployment encompass a broad spectrum of machine learning and deep learning techniques, including Support Vector Machines (SVM), Decision Trees, K-Nearest

Neighbors (KNN), Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Unit (GRU) networks. This diverse array of models allows for a comprehensive approach to fault detection, accommodating the varied and complex nature of potential electrical system faults.

4.3 Model's Inputs and Outputs

4.3.1 Inputs

The proposed system is designed to analyze and interpret various electrical parameters (47 in total) to detect and classify power system anomalies. The inputs to the model are meticulously chosen to capture the comprehensive state of the electrical system across three phases. These inputs include:

- Frequency: The fundamental frequency of the electrical system, which is a critical parameter for AC power systems.
- Timestamp: The specific time at which the data is recorded, allowing for temporal analysis of the electrical parameters.
- For each of the three phases, the following parameters are recorded:
  - Maximum Voltage (Vmax): The peak voltage observed, providing insights into potential overvoltage conditions.
  - Root Mean Square (RMS) Voltage (Vrms): A measure of the effective voltage level, indicative of the power content in the voltage signal.
  - Voltage Angle Theta (Theta): The phase angle of the voltage, important for power factor and phase shift analyses.
  - Voltage Waveform (Vwave): The shape of the voltage signal, useful for identifying distortions and harmonics.
  - Load Resistance (Zr)** and **Load Reactance (Zx): The resistive and reactive components of the load impedance, respectively, affecting the system's power consumption and phase angles.
  - Maximum Current (Imax) and RMS Current (Irms): Peak and effective current levels, crucial for assessing load demand and potential overcurrent situations.
  - Current Angle Phi (Phi): The phase angle of the current relative to the voltage, important for calculating real and reactive power.
  - Current and Power Waveforms (Iwave, Pwave): The shapes of the current and power signals, respectively, which help in identifying anomalies like harmonics and transients.
  - Active (P), Reactive (Q), and Apparent (S) Power: These parameters provide a comprehensive view of the power dynamics in each phase, essential for energy management and fault diagnosis.

Additionally, the total active power (Pt), total reactive power (Qt), and total apparent power (St) are computed to provide a holistic view of the power system's performance.

4.3.2 Outputs

The system's outputs (58 in total) are designed to categorize the operational status of the electrical system into various fault and non-fault conditions. These outputs include:

- Normal Operation (Good/ok): Indicating the system is functioning within the expected operational parameters.
- Voltage Anomalies: Including single, double, and triple undervoltage (suv, duv, tuv) and overvoltage (sov, dov, tov) faults, alongside voltage magnitude (svmi, dvmi, tvmi) and phase angle (svpi, dvpi, tvpi) imbalances.
- Harmonic Distortions: Classified into single, double, and triple harmonic distortion faults (shd, dhd, thd), reflecting the presence of non-fundamental frequency components in the electrical signals.
- Arc Faults: Identified as single, double, and triple arc faults (sarc, darc, tarc), indicative of high-energy discharges that can lead to equipment damage or fire hazards.
- Line Faults: Including line to ground (slg, dlg, tlg), line to line (sll, dll, tll), and open circuit faults (soc, doc, toc), crucial for pinpointing issues in the transmission infrastructure.
- Current Imbalances: Highlighting single, double, and triple current magnitude (scmi, dcmi, tcmi) and phase angle imbalances (scpi, dcpi, tcpi), essential for load balancing and system stability.
- Overcurrent Faults: Detected as single, double, and triple overcurrent conditions (sovc, dovc, tovc), important for protecting equipment from excessive current flows.
- Machine Faults: Including rotor (srot, drot, trot), stator (ssta, dsta, tsta), and bearing faults (sber, dber, tber) in electrical machines, which can lead to efficiency losses and equipment failures.
- Insulation, Thermal, and Resonance Conditions: Covering single, double, and triple insulation failures (sins, dins, tins), thermal overloads (stro, dtro, ttro), and resonance conditions (sres,

dres, tres), each indicative of specific issues affecting the system's reliability and safety.

These outputs are crucial for real-time monitoring, fault detection, and preventive maintenance strategies in modern electrical systems.

4.4 Data Collection and Dataset Composition

4.4.1 Data Collection Methodology

Our approach to data collection was event-driven, activated by the occurrence of faults within the electrical system. This strategy ensured that our data acquisition was highly targeted, capturing critical events that are most relevant for fault detection and analysis. Upon fault detection, our specialized equipment was immediately engaged to record the pertinent electrical parameters at a high fidelity.

4.4.2 Sampling Strategy

The chosen sampling rate of 50Ksps (50,000 samples per second) was pivotal for our analysis. Operating in a three-phase system with a nominal frequency of 50Hz, this sampling rate allowed us to achieve an approximate resolution of 100 samples per cycle. This high-resolution data capture was instrumental in identifying subtle disturbances, harmonics, and other anomalies indicative of various fault conditions, providing a rich dataset for in-depth analysis.

4.4.3 Dataset Composition

Following the filtration of the captured data and its segmentation into training and testing subsets, we achieved the following dataset composition:

- Training Dataset: This dataset, spanning 10 hours, 6 minutes, and 41 seconds of system operation, comprises a total of 17,262,000 samples. It represents a comprehensive collection of fault events and normal operational states, serving as the foundation for training our AI models.
- Testing Dataset: The testing dataset, which extends over 2 hours, 31 minutes, and 32 seconds, includes 4,299,000 samples. This dataset is crucial for evaluating the models' performance, ensuring they can accurately identify and classify faults under varied conditions.

4.4.4 Fault Type Distribution

A detailed breakdown of the occurrence frequency of each fault type is available in Table 2, which covers both the test and training sets. Figure 2 visually maps out these frequencies, making it simpler to compare the different fault types. Moreover, Figure 3 contrasts the instances when the system was functioning correctly, marked as 'OK', with the times various faults were detected, offering a clear perspective on how common faults are relative to normal operation.

Fault Type	Test Counts	Train Counts
suv	78000	342000
duv	66500	334500
tuv	69500	268000
sov	87000	283500
dov	70500	302000
tov	78000	288500
svmi	73000	290000
dvmi	47000	319500
tvmi	80500	270000
svpi	79000	314500
dvpi	67500	289000
tvpi	75500	318000
shd	4831	12007
dhd	8236	34888
thd	8270	40146
sarc	3902	14251
darc	7130	30604
tarc	11126	39034
slg	69500	275500
dlg	69000	292000
tlg	81000	287500
sll	79000	307500
dll	85000	307000
tll	75000	328500
soc	86500	280000
doc	69000	293500
toc	51000	304500
scmi	77500	341500
dcmi	66000	308500
tcmi	68000	306500
scpi	94000	325000
dcpi	60000	278000
tcpi	60000	283000
sovc	89500	298500
dovc	87500	331000
tovc	75000	323500
srot	84500	311000
drot	66500	319500
trot	77500	321000
ssta	67000	324000
dsta	60500	330000
tsta	91500	292000
sinc	4252	17184
dinc	10907	26311
tinc	7153	39234
sber	3935	14271
dber	6821	28832
tber	10678	40863
sins	4354	15500
dins	4798	26442
tins	14210	42533
stro	89500	293500
dtro	65000	270000
ttro	85000	313000
sres	24733	107204
dres	29142	163242
tres	41586	198521
ok	1191436	4448433

Table 1:Frequency Distribution of Fault Types in Training and Test Sets

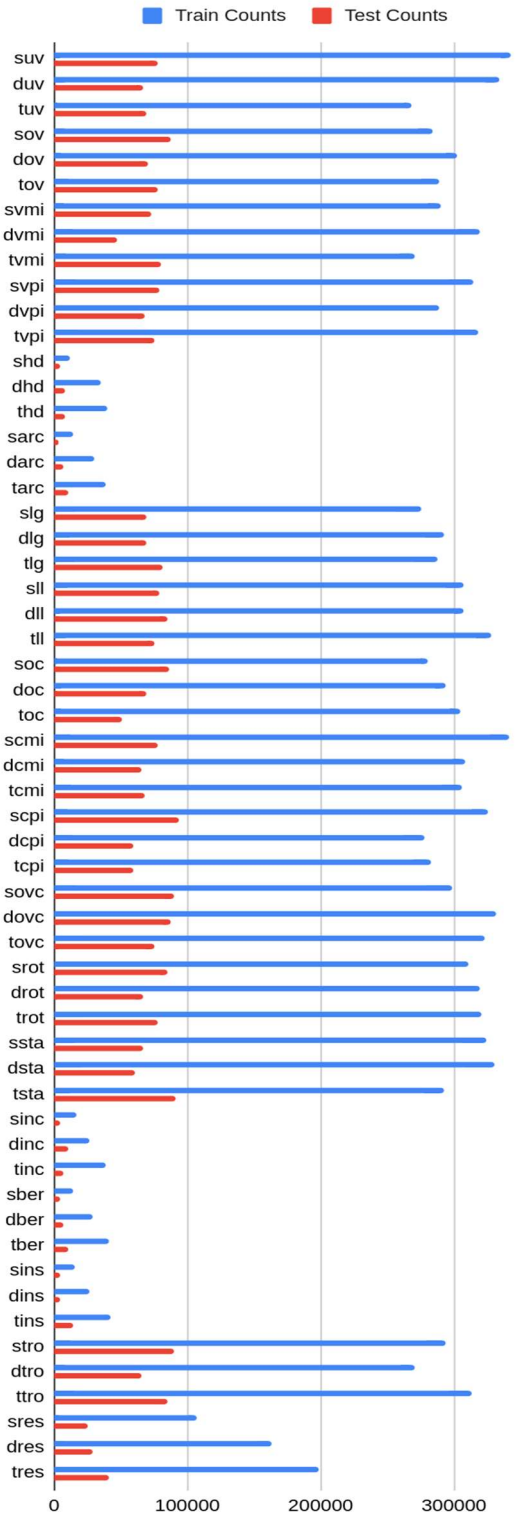

Figure 2: Distribution of Fault Type Occurrences in Training and Test Sets

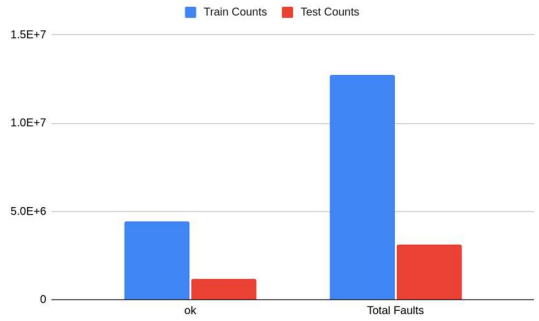

Figure 3: Comparative Analysis of Fault and Non-Fault Occurrences in Training and Test Sets

4.5 Model's Implementation

4.5.1 SVM

I chose the RBF kernel for the SVM because it excels at handling the non-linear complexities inherent in electrical fault diagnosis. The RBF kernel's adaptability makes it particularly effective for parsing the nuanced relationships within the multivariate data of three-phase electrical systems. This kernel's strength lies in its ability to transform the data into a higher-dimensional space, thereby uncovering decision boundaries not visible in the original feature space. Such a transformation is crucial for accurately classifying the subtle distinctions between different types of electrical faults. To optimize the model, I experimented with various gamma values [0.0001, 0.001, 0.01, 0.1, 1, 10], fine-tuning the model's sensitivity to the data's complexity and ensuring the best balance between bias and variance. This meticulous calibration, combined with the RBF kernel's proven track record in complex classification tasks, solidified its selection for achieving high accuracy in fault diagnosis.

4.5.2 Decision Trees

I opted for Decision Trees with the Gini impurity criterion due to their transparent and straightforward decision-making process, which is particularly beneficial for the intricate task of diagnosing electrical faults in three-phase systems. The Gini criterion, known for its efficiency in measuring the purity of a node, serves as an excellent tool for handling the diverse and categorical nature of electrical fault data. To further refine the model and adapt it to the complexity of the electrical fault patterns, I experimented with various maximum depths [8, 16, 32, 64, 128, 256]. These depths were carefully chosen to explore the trade-off between model simplicity and the ability to capture detailed

fault characteristics without overfitting. By adjusting the max_depth parameter, I was able to control the tree's growth, ensuring it was deep enough to learn the nuanced distinctions between different fault types, yet restrained to prevent learning the noise in the data. This methodical approach to tuning the Decision Tree, coupled with the intuitive nature of the Gini criterion, formed the cornerstone of my strategy to achieve a model that is both accurate and interpretable.

4.5.3 KNN

I selected the K-Nearest Neighbors (KNN) algorithm for its intuitive approach to classifying electrical faults by leveraging the similarity of data points in the feature space. Understanding that the proximity of data points in a multidimensional space often signifies similarity in characteristics, I found KNN to be particularly apt for distinguishing between the various types of faults in three-phase electrical systems based on their feature signatures. To fine-tune the model and ensure it captures the intricate patterns of the data without succumbing to noise, I experimented with different numbers of neighbors [8, 16, 32, 64, 128, 256]. This range allowed me to assess the model's performance across a spectrum of neighborhood sizes, from more localized to more generalized perspectives. By adjusting the n_neighbors parameter, I aimed to strike an optimal balance where the model is sensitive enough to identify subtle fault distinctions but robust against the irregularities and variability inherent in the data. This careful calibration of KNN, guided by the principle of similarity in the feature space, underpinned my strategy to develop a model that is both precise in fault classification and resilient to overfitting.

4.5.4 Neural Networks

In the structured design of neural network architectures for the study, encompassing ANN, CNN, RNN, LSTM, and GRU, a standardized approach was rigorously employed to ensure consistency and methodological clarity across various models. Each model's architecture initiates with an input layer, meticulously configured to align with the dimensions of the input data, thereby laying a robust foundation for the data processing pipeline. This phase is succeeded by model-specific hidden layers, each thoughtfully engineered to capture the unique temporal or spatial patterns present in the dataset, as elaborated in the architectural framework and illustrated in Figure 4.

Following the implementation of these specialized hidden layers, a unified architectural backbone is adopted across all models. This includes a Flatten layer, facilitating a seamless transition into dense layer processing, a crucial step depicted in Figure 4. The architecture is enhanced by integrating activation functions into the dense layers, which boosts the model's capacity to capture complex, non-linear relationships. Specifically, the first dense layer introduces non-linearity by employing a number of units as defined in (1), thereby aiding in the representation of a wide range of features. The following dense layer, which maintains the number of units as specified in (2), also includes a ReLU activation function, further refining the network's learning capabilities.

$$Units = 2 * 2^{\lceil log_2((\text{number\_of\_classes})\rceil} \quad (1)$$

$$Units = 2^{\lceil log_2((\text{number\_of\_classes})\rceil} \quad (2)$$

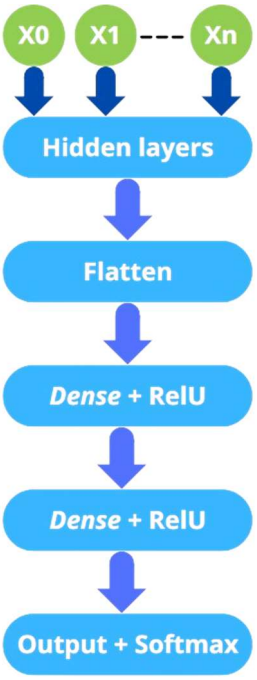

Figure 4: Core Architecture of Standardized Neural Networks for AI Models

The architectural sequence reaches its culmination with a softmax output layer, aptly designated as "output". This layer is specifically designed to provide a probabilistic distribution over the various classes, thereby enabling precise multi-class classification. To optimize the models for multi-class classification tasks, "categorical_crossentropy" was chosen as the loss function for its effectiveness in handling multiple classes by comparing the predicted probabilities with the actual class distribution. The "adam"

optimizer was selected for its adaptive learning rate capabilities, making it well-suited for a wide range of data and network architectures, thereby enhancing the training process's efficiency and convergence.

This cohesive and methodologically sound architectural blueprint, as detailed in the layer configuration, activation layers, loss function, and optimizer choice, and visually encapsulated in Figure 4, emphasizes a commitment to upholding a consistent framework across the array of neural network models under consideration. The explicit mention of "categorical_crossentropy" and "adam" ensures a clear understanding of the training dynamics, contributing to the models' transparency and reproducibility. This structured approach facilitates a detailed assessment of each model's distinct features and efficacy, offering a robust and versatile framework tailored to the analytical objectives at hand.

4.5.4.1 ANN

In the ANN model's hidden layer configuration, a dense layer with ReLU activation was employed. The model's adaptability and learning depth were tested by varying the number of units in this layer, specifically exploring configurations with 8, 16, 32, 128, and 256 units. This range was chosen to assess the network's performance from simpler to more complex structures, aiming to find an optimal setup that captures intricate data patterns efficiently while avoiding overfitting, thus ensuring precise classification and robust generalization.

4.5.4.2 CNN,RNN,LSTM,GRU

In our comprehensive exploration, we delved into the intricacies of neural network architectures including CNN, RNN, LSTM, GRU, and their hybrid variations, particularly those that integrate CNN with RNN, LSTM, and GRU elements. Our primary aim was to identify the optimal configurations that delicately balance model complexity with the ability to generalize effectively. This balance is pivotal for proficient pattern recognition and ensuring stable performance across diverse datasets. To achieve this, we systematically tested various model depths and unit values, examining how these parameters influence the models' learning capabilities and overall efficacy.

4.5.4.2.1 CNN Architectures

Delving into the realm of Convolutional Neural Networks, our study meticulously explored the capabilities of CNNs to process and analyze spatial data. This foundational exploration served as a critical step in understanding how CNNs, with their unique architectural design, effectively capture spatial features and patterns within data, setting the stage for more complex model configurations.

- CNN_1 Configuration: Employs a single Conv1D layer, experimenting with different units (8, 16, 32, 128, 256) to gauge the initial spatial feature extraction capabilities. This is followed by a MaxPooling layer to diminish dimensionality and highlight key features, thereby streamlining the feature selection process (see Figure 5).
- CNN_2 Configuration: Builds on the CNN_1 setup by stacking two such layers sequentially, using units (8,16), (16,32), (32,64), (64,128), and (128,256). This structure is intended to refine the depth of feature extraction, enhancing the model's analytical depth (illustrated in Figure 6).
- CNN_3 Configuration: Introduces a third layer to the CNN_1 sequence, employing unit combinations (8,16,32), (16,32,64), (32,64,128), and (64,128,256). This approach aims to explore the advantages of increased depth in hierarchical feature learning (depicted in Figure 7).

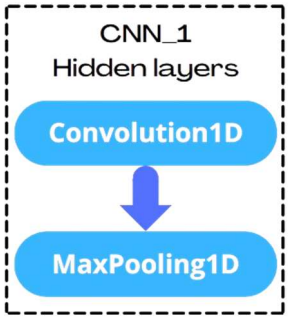

Figure 5: Hidden Layer Configuration of CNN_1 Model

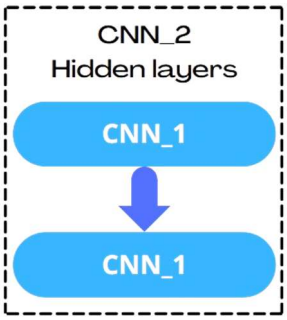

Figure 6: Hidden Layer Configuration of CNN_2 Model

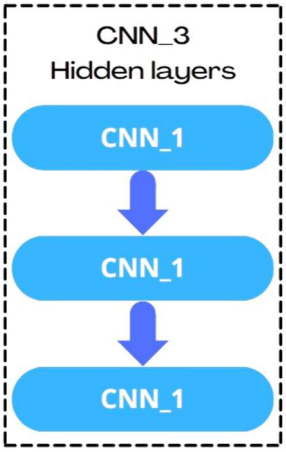

Figure 7: Hidden Layer Configuration of CNN_3 Model

4.5.4.2.2 RNN, LSTM, and GRU Models

The focus shifts towards understanding the management of temporal dependencies, essential for processing sequential data. The configurations explored include:

- (RNN/LSTM/GRU)_1: Incorporates a single recurrent layer with units (8, 16, 32, 128, 256) and a 50% dropout rate to evaluate the basic ability to capture temporal patterns and prevent overfitting, thereby enhancing the model's generalization capacity (see Figure 8).
- (RNN/LSTM/GRU)_2: Consists of two (RNN/LSTM/GRU)_1 layers stacked sequentially, with units set as pairs (8,16), (16,32), (32,64), (64,128), (128,256). This configuration seeks to assess how additional layers improve the model's capability to capture more intricate temporal relationships (illustrated in Figure 9).
- (RNN/LSTM/GRU)_3: Assembles three (RNN/LSTM/GRU)_1 layers sequentially, with unit combinations (8,16,32), (16,32,64), (32,64,128), (64,128,256). This setup aims to delve into the network's potential for advanced temporal pattern recognition and long-term dependency modeling (depicted in Figure 10).

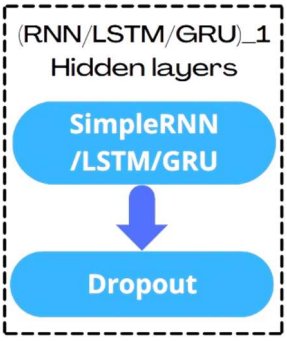

Figure 8: Hidden Layer Configuration in Single-Layer RNN/LSTM/GRU Models

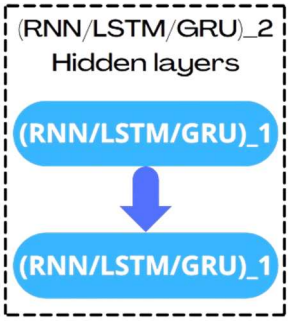

Figure 9: Hidden Layer Configuration in Dual-Layer RNN/LSTM/GRU Models

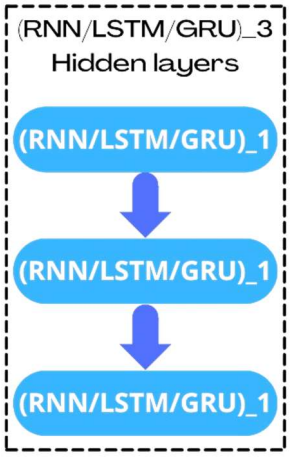

Figure 10: Hidden Layer Configuration in Triple-Layer RNN/LSTM/GRU Models

4.5.4.2.3 Hybrid CNN_(RNN/LSTM/GRU) Models

The fusion of CNN architectures with recurrent neural models (RNN, LSTM, GRU) marks a significant advancement in our quest to harness the full potential of neural networks for spatial-temporal data analysis. These hybrid models are designed to capture the best of both worlds: the CNN's adeptness at processing spatial information and the sequential

data handling capabilities of RNN-based models. By intertwining these architectures, we aim to create a versatile framework capable of deciphering complex patterns that are not only spatially but also temporally significant. This innovative approach promises to extend the applicability of neural networks to a broader spectrum of challenges, especially those involving intricate spatial-temporal dynamics.

- CNN_(RNN/LSTM/GRU)_1: This hybrid model initiates with the complete CNN_1 structure, focusing on spatial feature extraction, and is followed by an (RNN/LSTM/GRU)_1 layer dedicated to temporal modeling. The objective is to synergize the spatial analysis proficiency of CNNs with the temporal sequencing strengths of RNN-based models, offering a holistic solution for analyzing spatial-temporal data (refer to Figure 11).
- CNN_(RNN/LSTM/GRU)_2: The model's depth and analytical capabilities are further augmented by combining two CNN_(RNN/LSTM/GRU)_1 units. This enhancement seeks to enrich both the spatial and temporal dimensions of data analysis, providing a more nuanced understanding of the underlying patterns (illustrated in Figure 12).
- CNN_(RNN/LSTM/GRU)_3: By integrating three CNN_(RNN/LSTM/GRU)_1 units, this configuration substantially increases the model's complexity, pushing the boundaries of spatial-temporal feature extraction and recognition. This ambitious setup is designed to tackle the most challenging spatial-temporal analysis tasks, offering unparalleled insights into the data (depicted in Figure 13).

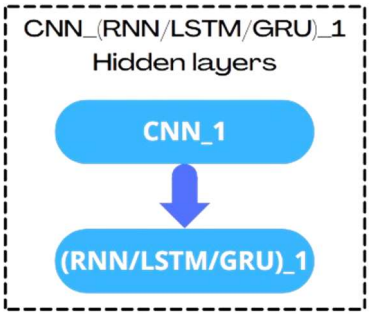

Figure 11: Hidden Layer Configuration in Single-Layer CNN_(RNN/LSTM/GRU) Models

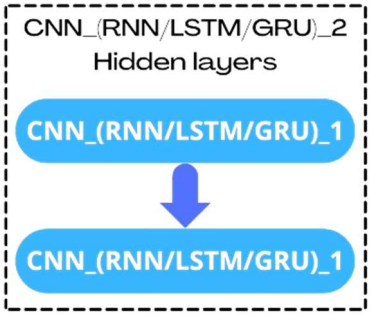

Figure 12: Hidden Layer Configuration in Dual-Layer CNN_(RNN/LSTM/GRU) Models

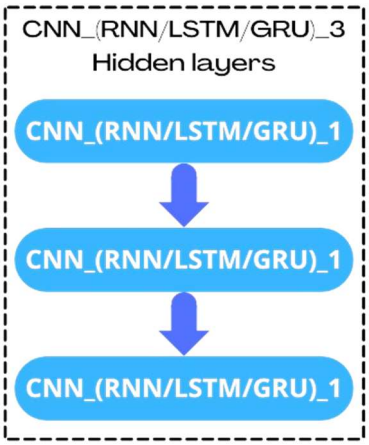

Figure 13: Hidden Layer Configuration in Triple-Layer CNN_(RNN/LSTM/GRU) Models

Through this extensive exploration, we leveraged the distinct strengths of CNN and RNN-based architectures, charting new territories in complex dataset analysis that necessitates an intricate understanding of both spatial and temporal dimensions. This endeavor not only sheds light on the optimal configurations for various neural network models but also sets the stage for future innovations in neural network applications and optimization.

5. RESULTS AND DISCUSSION

In this section, we delve into the empirical findings from our application of various AI methodologies to fault detection in three-phase electrical systems. Through a comparative analysis, we assess the performance of each model in identifying and classifying faults, offering insights into their practical implications for industrial applications.

Our analysis is intricately linked to the overarching objectives of this research, aiming to

enhance the accuracy and efficiency of fault detection mechanisms in industrial settings. The results demonstrate that AI models, particularly those employing deep learning techniques like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), significantly improve the detection and diagnosis of faults. This empirical evidence underscores the potential of AI to revolutionize fault detection processes, aligning with our initial research objectives of integrating advanced AI methodologies to improve operational reliability and efficiency in the context of Industry 4.0.

5.1 Hardware characteristics

The outcomes of our study were achieved on a server equipped with specific hardware specifications:

- CPU: AMD Ryzen 9 5900X (1 socket, 12 cores, 24 threads).
- RAM: 128 GB.
- GPU: 4 * NVidia's GeForce RTX 3080 with Cuda v12.0.

In our experiments, we employed Keras, an open-source Python library for deep learning, running on TensorFlow version 2.15.0 as the backend engine. Additionally, we utilized Scikit-learn version 1.3.2 for traditional machine-learning models and metric calculations.

5.2 Evaluating the models

To evaluate our models, we generated a wide array of metrics, including Accuracy, Precision, Recall, F1 Score, Class Precision, Class Recall, Class F1 scores, and the confusion matrix for each. However, for the purpose of comparison across the diverse set of models in this study, we will concentrate on using only Accuracy, Precision, Recall, and F1 Score. These core metrics provide a comprehensive and efficient framework for assessing and comparing model performance, offering insights into the models' overall effectiveness and their balance between precision and recall, which are pivotal for the nuanced task of multi-class fault classification.

In summarizing the extensive analysis across various neural network models and configurations, we observed the following key insights:

5.2.1 SVM model:

The SVM model's performance varies with gamma values, as illustrated in Table 2. Lower gammas lead to underfitting with high precision but low recall, resulting in a low F1-score. Slight increases in gamma marginally improve accuracy, precision, and recall. Gamma 0.01 offers balanced performance with significantly improved metrics. However, higher gammas induce overfitting, sacrificing accuracy for increased recall. Gamma 0.01 emerges as the optimal choice, striking a balance between precision, recall, and accuracy. Therefore, meticulous gamma parameter tuning is crucial for optimizing the SVM model's performance on the dataset, ensuring effective classification without compromising generalization.

Table 2: SVM Model Performance Analysis with Varied Gamma Values

Gamma	Accuracy	Precision	Recall	F1-score
0.0001	0.374	0.864	0.169	0.120
0.001	0.402	0.889	0.237	0.138
0.01	0.448	0.789	0.330	0.290
0.1	0.308	0.684	0.334	0.266
1	0.138	0.668	0.261	0.149
10	0.104	0.876	0.092	0.092

5.2.2 Decision Tree model:

The Decision Tree model's performance varies with tree depths, as outlined in Table 3. At depth 8, the model exhibits high precision but low recall, indicating conservative positive case predictions, resulting in a low F1-score despite high accuracy. Increasing to depth 16 improves accuracy, precision, and recall slightly, capturing more data complexities. Depth 32 shows decreased precision but improved recall, yielding a higher F1-score. Depth 64 further enhances recall, resulting in the highest F1-score among tested depths. Depths 128 and 256 plateau in performance, suggesting diminishing returns. The optimal balance between precision and recall, evident in the highest F1-score, is at depth 64. Careful tree depth tuning is vital for optimizing model performance on this dataset.

Table 3: Decision Tree Model Evaluation Across Different Depths

Depth	Accuracy	Precision	Recall	F1-score
8	0.540	0.970	0.266	0.262
16	0.613	0.963	0.343	0.337
32	0.588	0.658	0.369	0.658
64	0.512	0.425	0.382	0.717
128	0.482	0.430	0.379	0.701
256	0.482	0.430	0.379	0.701

5.2.3 KNN model:

The performance of the K-Nearest Neighbors (KNN) model is influenced by the number of neighbors, as demonstrated in Table 4. With 8 neighbors, moderate accuracy and precision are achieved, alongside a relatively low recall. The F1-score indicates a balanced precision-recall trade-

off. Increasing to 16 neighbors maintains similar accuracy and slightly improves precision, yet the F1-score decreases slightly. Doubling to 32 neighbors leads to decreased accuracy but increased precision, resulting in a lower F1-score. Further increases to 64 and 128 neighbors show diminishing returns, with declining recall and fluctuating F1-scores. At 256 neighbors, there's a slight increase in accuracy and precision, but the F1-score remains lower. Smaller neighborhoods exhibit better precision-recall balance, suggesting their effectiveness in this dataset.

Table 4: KNN Model Performance with Varying Number of Neighbors

Depth	Accuracy	Precision	Recall	F1-score
8	0.540	0.970	0.266	0.262
16	0.613	0.963	0.343	0.337
32	0.588	0.658	0.369	0.658
64	0.512	0.425	0.382	0.717
128	0.482	0.430	0.379	0.701
256	0.482	0.430	0.379	0.701

5.2.4 ANN model:

The ANN model's performance varies with the number of units, as outlined in Table 5. With 8 units, moderate accuracy and precision are observed, along with relatively low recall. Increasing to 16 units improves all metrics, indicating better capture of dataset nuances. At 32 units, slight improvements in accuracy and precision are noted, with a marginal increase in the F1-score. Jumping to 64 units notably boosts recall, though precision decreases slightly, resulting in a significant F1-score improvement. Surprisingly, 128 units yield decreased metrics, suggesting potential overfitting. However, at 256 units, the model achieves peak performance, raising concerns of overfitting. Overall, the ANN model's performance improves with complexity, but careful tuning is vital to prevent overfitting.

Table 5: ANN Model Performance Across Different Unit Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.655	0.784	0.411	0.531
16	0.686	0.780	0.458	0.579
32	0.689	0.793	0.458	0.581
64	0.709	0.722	0.496	0.642
128	0.557	0.583	0.357	0.590
256	0.790	0.817	0.595	0.659

5.2.5 CNN models:

The CNN_1 model reveals diverse performance metrics across various configurations, as shown in Table 6. With 8 units, it demonstrates moderate accuracy and precision, while increasing to 16 units shows slight accuracy improvements but a decrease in precision. However, at 32 units, there's a notable drop in accuracy and recall. Surprisingly, 128 units yield significant improvements in all metrics, indicating the best performance among CNN_1 setups.

The CNN_2 model, detailed in Table 7, showcases enhancements as complexity increases. Ranging from [8,16] to [128,256] units, there's a consistent trend of increasing accuracy, precision, and recall. Notably, configurations like [128,256] yield the highest performance, indicating a better balance and overall effectiveness.

In the CNN_3 model, illustrated in Table 8, performance improves with deeper layers. Progressing from [8,16,32] to [64,128,256] units, notable enhancements in accuracy, precision, and recall are observed. However, fluctuations in the F1-score suggest varying balance levels. Nonetheless, the [64,128,256] setup achieves remarkable accuracy and precision, coupled with a significant recall boost, indicating robust performance.

The CNN models exhibit diverse performance trends across different configurations and depths. As complexity increases, so does the overall effectiveness, with deeper layers and larger units generally leading to better balance and performance. Notably, configurations like [128,256] for CNN_2 and [64,128,256] for CNN_3 demonstrate superior performance, showcasing the importance of complexity in capturing dataset nuances. However, potential considerations for overfitting and computational costs should be carefully evaluated when opting for deeper and more intricate models.

Table 6: CNN_1 Model Performance with Various Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.687	0.762	0.453	0.591
16	0.690	0.671	0.456	0.649
32	0.647	0.728	0.396	0.584
64	0.579	0.565	0.368	0.635
128	0.775	0.835	0.590	0.616
256	0.664	0.638	0.419	0.662

Table 7: CNN_2 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16]	0.685	0.712	0.451	0.641
[16,32]	0.704	0.692	0.480	0.642
[32,64]	0.738	0.754	0.531	0.617
[64,128]	0.667	0.649	0.427	0.686
[128,256]	0.858	0.831	0.667	0.760

Table 8: CNN_3 Model Performance with Varied Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16,32]	0.678	0.764	0.439	0.561
[16,32,64]	0.698	0.695	0.467	0.626
[32,64,128]	0.718	0.749	0.500	0.580
[64,128,256]	0.800	0.847	0.607	0.634

5.2.6 RNN models:

The RNN models demonstrate varied performance across different configurations and depths. Starting with RNN_1, showcased in Table 9, we observe that 8 units exhibit moderate accuracy and high precision, while 16 units show improved accuracy and recall. However, at 32 units, there's a decrease in accuracy, precision, and recall. Substantial improvements are observed at 64 and 128 units, with optimal performance seen at 256 units.

Moving to RNN_2, illustrated in Table 10, configurations such as [8,16] units display lower accuracy and recall compared to RNN_1 setups, with relatively high precision but a low F1-score. Further configurations show varying degrees of improvement, with [128,256] units achieving the highest metrics across the board.

Similarly, we explore RNN_3, detailed in Table 11, where we observe notable performance fluctuations. While [8,16,32] units exhibit the lowest accuracy and recall, subsequent configurations demonstrate improvements. Notably, [64,128,256] units show the highest F1-score among RNN_3 setups.

Overall, increasing the complexity of RNN models generally enhances performance, with notable improvements in accuracy, precision, recall, and F1-scores observed in larger unit configurations. However, optimal configuration selection requires careful tuning and validation to prevent overfitting and ensure robust generalization to new data.

Table 9: RNN_1 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.673	0.843	0.430	0.501
16	0.717	0.812	0.497	0.535
32	0.590	0.611	0.370	0.568
64	0.814	0.862	0.623	0.650
128	0.927	0.943	0.741	0.758
256	0.944	0.959	0.777	0.786

Table 10: RNN_2 Model Performance with Various Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16]	0.578	0.756	0.350	0.424
[16,32]	0.523	0.564	0.357	0.610
[32,64]	0.866	0.903	0.676	0.697
[64,128]	0.923	0.943	0.755	0.764
[128,256]	0.973	0.976	0.806	0.817

Table 11: RNN_3 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16,32]	0.473	0.550	0.312	0.490
[16,32,64]	0.500	0.548	0.354	0.559
[32,64,128]	0.430	0.482	0.302	0.591
[64,128,256]	0.457	0.459	0.334	0.619

5.2.7 LSTM models:

The LSTM models exhibit diverse performance across different configurations and depths. In LSTM_1 (Table 12), with 8 units, there's moderate accuracy and high precision but lower recall, suggesting a simplistic approach. Progressing to 16 units shows slight improvements in accuracy, precision, and recall, indicating a more balanced model. However, at 32 units, there's a drop in accuracy and precision, possibly due to overfitting, while 64 units exhibit a further decrease in metrics. Nonetheless, 128 units show improved accuracy and precision.

Moving to LSTM_2 (Table 13), configurations such as [8,16] units display lower accuracy and recall compared to LSTM_1, while [16,32] units show improvements in accuracy and recall. However, setups like [32,64] units exhibit a significant drop in accuracy and precision. On the other hand, [64,128] units demonstrate notable improvements across all metrics.

In LSTM_3 (Table 14), configurations vary considerably. For instance, [8,16,32] units show the lowest accuracy and recall, while [64,128,256] units achieve the highest accuracy and precision. However, increasing complexity doesn't consistently enhance performance, as evidenced by fluctuating F1-scores. Despite this, LSTM_2 configurations seem to strike a balance, particularly the [64,128] units setup.

Overall, while increased complexity tends to improve LSTM model performance, further complexity doesn't always yield better results and may introduce overfitting. Thus, careful tuning is necessary to find the optimal configuration that balances precision, recall, and overall accuracy.

Table 12: LSTM_1 Model Performance with Varied Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.677	0.786	0.430	0.521
16	0.706	0.747	0.476	0.543
32	0.581	0.573	0.360	0.594
64	0.561	0.562	0.363	0.587
128	0.710	0.750	0.478	0.564

256	0.630	0.796	0.377	0.476

Table 13: LSTM_2 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16]	0.607	0.744	0.359	0.439
[16,32]	0.684	0.714	0.450	0.517
[32,64]	0.516	0.505	0.334	0.607
[64,128]	0.794	0.847	0.584	0.613
[128,256]	0.651	0.783	0.400	0.538

Table 14: LSTM_3 Model Performance with Various Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16,32]	0.466	0.721	0.218	0.319
[16,32,64]	0.215	0.766	0.062	0.216
[32,64,128]	0.575	0.891	0.324	0.368
[64,128,256]	0.648	0.941	0.395	0.394

5.2.8 GRU models:

The GRU models exhibit varied performance across configurations and depths. In GRU_1 (Table 15), starting with 8 units, there's moderate accuracy and high precision but lower recall, suggesting a precise yet limited model. Progressing to 16 units shows a slight decrease in accuracy and precision, but a better balance between precision and recall. At 32 units, there are significant improvements in accuracy, precision, and recall, indicating effective pattern capture. Subsequently, 64 units display substantial enhancements in all metrics, reflecting excellent model performance, while 128 units exhibit near-perfect accuracy and precision.

In GRU_2 (Table 16), configurations like [8,16] units show lower accuracy and precision compared to GRU_1, suggesting limited effectiveness. Further configurations exhibit nuanced changes, with [32,64] units showcasing improvements in balance and overall performance. However, setups like [128,256] units indicate a complex interplay between precision and recall, despite lower overall metrics.

In GRU_3 (Table 17), configurations vary significantly. For instance, [8,16,32] units show the lowest accuracy and precision, while [64,128,256] units achieve the highest accuracy and precision. Nonetheless, increasing complexity doesn't consistently enhance performance, as evidenced by varying F1-scores. Despite this, GRU_1 configurations, particularly with 256 units, demonstrate the best balance between precision, recall, and overall accuracy.

Overall, GRU models benefit from increased complexity up to a point, as seen in GRU_1 configurations. However, further complexity doesn't always yield better performance and may introduce overfitting, necessitating careful tuning and validation for generalization to new data.

Table 15: GRU_1 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.676	0.810	0.434	0.557
16	0.576	0.582	0.361	0.611
32	0.785	0.836	0.578	0.629
64	0.900	0.896	0.713	0.769
128	0.954	0.951	0.777	0.809
256	0.978	0.981	0.817	0.827

Table 16: GRU_2 Model Performance with Varied Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16]	0.610	0.612	0.393	0.560
[16,32]	0.542	0.543	0.353	0.599
[32,64]	0.807	0.834	0.630	0.651
[64,128]	0.736	0.733	0.585	0.653
[128,256]	0.674	0.660	0.430	0.657

Table 17: GRU_3 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16,32]	0.342	0.402	0.241	0.468
[16,32,64]	0.621	0.837	0.370	0.463
[32,64,128]	0.609	0.819	0.355	0.444
[64,128,256]	0.669	0.752	0.424	0.582

5.2.9 CNN_RNN models:

The CNN-RNN hybrid models exhibit varied performance across different configurations and depths. In CNN_RNN_1 (Table 18), starting with 8 units, there's moderate accuracy and very high precision but lower recall, suggesting a model that's precise but not as effective in identifying all positive cases. Progressing to 16 units shows improvements in accuracy and recall, with a slight decrease in precision but a higher F1-score, indicating a better balance. At 32 units, significant enhancements in all metrics reflect excellent model performance, while 128 units achieve near-perfect accuracy and precision.

In CNN_RNN_2 (Table 19), configurations like [8,16] units exhibit lower accuracy and recall compared to CNN_RNN_1, suggesting limited effectiveness. Further configurations show nuanced changes, with [32,64] units indicating potential issues with model overfitting or complexity. However, setups like [128,256] units demonstrate a significant jump in all metrics, suggesting an optimal balance between complexity and performance.

In CNN_RNN_3 (Table 20), configurations vary in performance. For instance, [8,16,32] units show moderate accuracy and high precision but low

recall, while [64,128,256] units exhibit further decreases in accuracy and precision, with potential overfitting issues. Despite this, CNN_RNN_1 and CNN_RNN_2 configurations, particularly with larger unit sizes, demonstrate improved performance.

Overall, CNN-RNN hybrid models tend to improve in performance with increased complexity up to a point, particularly seen in CNN_RNN_1 and CNN_RNN_2 configurations. However, further complexity doesn't consistently lead to better performance and might introduce overfitting, emphasizing the importance of careful tuning and validation. The optimal configuration appears to be within the CNN_RNN_2 range, especially the [128,256] units configuration.

Table 18: CNN_RNN_1 Model Performance with Various Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.656	0.895	0.409	0.462
16	0.706	0.818	0.466	0.546
32	0.793	0.838	0.593	0.614
64	0.878	0.908	0.686	0.706
128	0.958	0.969	0.772	0.788
256	0.968	0.983	0.798	0.810

Table 19: CNN_RNN_2 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16]	0.599	0.874	0.349	0.408
[16,32]	0.614	0.689	0.396	0.593
[32,64]	0.517	0.587	0.328	0.525
[64,128]	0.496	0.573	0.319	0.497
[128,256]	0.958	0.973	0.754	0.758

Table 20: CNN_RNN_3 Model Performance with Varied Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16,32]	0.578	0.806	0.329	0.403
[16,32,64]	0.556	0.594	0.345	0.627
[32,64,128]	0.464	0.587	0.326	0.538
[64,128,256]	0.442	0.583	0.308	0.600

5.2.10 CNN_LSTM model:

The CNN-LSTM hybrid models showcase performance variations across different configurations and depths. Starting with CNN_LSTM_1 (Table 21), configurations like 8 units display moderate accuracy with high precision but lower recall, suggesting a significant number of missed detections. Progressing to 16 units shows improved accuracy and recall with slightly lower precision, indicating a better balance between identifying and classifying positive cases. At 32 units, there's a drop in accuracy and precision but an increase in the F1-score, suggesting improved balance despite lower overall performance.

In CNN_LSTM_2 (Table 22), configurations like [8,16] units exhibit lower accuracy and recall compared to most CNN_LSTM_1 setups, indicating less effectiveness in capturing relevant cases. Further configurations show nuanced changes, with [32,64] units indicating potential improvements in identifying positive cases. However, setups like [128,256] units demonstrate a drop in all metrics, suggesting potential issues with overfitting or model complexity.

In CNN_LSTM_3 (Table 23), configurations vary significantly, with setups like [8,16,32] units showing the lowest accuracy and recall, while [64,128,256] units exhibit further decreases in accuracy and precision. Despite this, CNN_LSTM_1 configurations, particularly with larger unit sizes, indicate improved performance, suggesting more complex models can better capture the dataset's spatial-temporal dynamics.

Overall, CNN-LSTM hybrid models tend to show improved performance with increased complexity up to a certain point, as seen in CNN_LSTM_1 configurations. However, further complexity doesn't consistently lead to better performance and might introduce issues like overfitting. The optimal configuration appears to be within the CNN_LSTM_1 range, especially the 256 units setup, offering a good balance between precision, recall, and overall accuracy.

Table 21: CNN_LSTM_1 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.660	0.874	0.411	0.448
16	0.737	0.815	0.514	0.544
32	0.564	0.599	0.373	0.574
64	0.813	0.863	0.609	0.636
128	0.975	0.988	0.802	0.815
256	0.978	0.980	0.826	0.833

Table 22: CNN_LSTM_2 Model Performance with Various Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16]	0.518	0.750	0.279	0.405
[16,32]	0.466	0.515	0.291	0.480
[32,64]	0.441	0.577	0.305	0.542
[64,128]	0.362	0.504	0.254	0.523
[128,256]	0.410	0.538	0.294	0.531

Table 23: CNN_LSTM_3 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16,32]	0.306	0.565	0.081	0.334
[16,32,64]	0.411	0.463	0.222	0.460
[32,64,128]	0.256	0.365	0.114	0.516

[64,128,256]	0.245	0.428	0.168	0.496

5.2.11 CNN_GRU model:

The CNN_GRU hybrid models display varying performance across configurations and depths. In CNN_GRU_1 (Table 24), starting with 8 units, there's moderate accuracy and high precision but relatively low recall, suggesting precision at the expense of identifying all relevant cases. Progressing to 16 units shows a decrease in accuracy and precision, with a similar F1-score, indicating a less optimal balance. At 32 units, there's further decline in accuracy and precision but an increase in the F1-score, suggesting an improvement in balance despite lower overall performance.

In CNN_GRU_2 (Table 25), configurations like [8,16] units exhibit lower accuracy and recall compared to most CNN_GRU_1 setups, indicating less effectiveness in capturing relevant cases. Further configurations show nuanced changes, with [32,64] units indicating potential improvements in identifying positive cases. However, setups like [128,256] units demonstrate a drop in all metrics, suggesting potential issues with overfitting or model complexity.

In CNN_GRU_3 (Table 26), configurations vary significantly, with setups like [8,16,32] units showing the lowest accuracy and recall, while [32,64,128] units exhibit improvements in accuracy, precision, and recall. Despite this, CNN_GRU_1 configurations, particularly with larger unit sizes, indicate improved performance, suggesting more complex models can better capture the dataset's spatial-temporal dynamics.

Overall, CNN-GRU hybrid models tend to show improved performance with increased complexity up to a certain point, as seen in CNN_GRU_1 configurations. However, further complexity doesn't consistently lead to better performance and might introduce issues such as overfitting. The optimal configuration appears to be within the CNN_GRU_1 range, especially the 256 units setup, offering a good balance between precision, recall, and overall accuracy. Careful tuning and validation are essential to prevent overfitting and ensure the model generalizes well to new data.

Table 24: CNN_GRU_1 Model Performance with Varied Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
8	0.679	0.888	0.434	0.488
16	0.597	0.720	0.366	0.493
32	0.562	0.573	0.358	0.599
64	0.859	0.907	0.661	0.692
128	0.948	0.962	0.764	0.780
256	0.979	0.971	0.818	0.828

Table 25: CNN_GRU_2 Model Performance Across Different Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16]	0.555	0.912	0.300	0.324
[16,32]	0.541	0.572	0.364	0.522
[32,64]	0.473	0.575	0.307	0.546
[64,128]	0.479	0.549	0.333	0.543
[128,256]	0.453	0.580	0.323	0.561

Table 26: CNN_GRU_3 Model Performance with Various Architectural Configurations

Units	Accuracy	Precision	Recall	F1-score
[8,16,32]	0.329	0.581	0.164	0.354
[16,32,64]	0.315	0.334	0.131	0.465
[32,64,128]	0.662	0.845	0.420	0.504
[64,128,256]	0.360	0.459	0.258	0.522

5.2.12 Best Performing Models:

Drawing insights from Table 27, which presents the 'Best' configurations for each model type and is visually represented in Figure 14 for clearer understanding, we observe distinct performance characteristics across a spectrum of machine learning models:

5.2.12.1 Traditional Models:

- SVM (Gamma 0.01): This configuration shows moderate accuracy and high precision, indicating a strong ability to correctly identify positive cases. However, the low recall and F1-score suggest a cautious approach in classification, highlighting a potential area for improvement in capturing all positive instances.
- DT (Depth 64): With relatively lower accuracy and precision but a higher recall than SVM, this depth setting indicates a broader identification of positive cases, albeit at the cost of increased false positives. The higher F1-score compared to SVM suggests a better balance between precision and recall.
- KNN (8 Neighbors): This model offers balanced performance with moderate accuracy, precision, and recall. The F1-score indicates a reasonable trade-off between precision and recall, making it suitable for applications where a moderate level of accuracy is acceptable.
- ANN (256 Units): Demonstrating high accuracy and precision alongside moderate recall, this configuration showcases strong capabilities in pattern recognition and classification. The significant F1-score emphasizes the model's balanced performance, making it a competitive option for complex tasks.

5.2.12.2 Advanced Neural Networks:

- CNN ([128,256] Units): Exhibiting high accuracy, precision, and recall, this configuration underscores the CNN's excellent spatial feature extraction and classification capabilities. The high F1-score suggests an effective balance between precision and recall, making it particularly suitable for tasks involving spatial data.
- RNN ([128,256] Units): This model stands out with superior accuracy, precision, and recall, highlighting its exceptional ability to capture temporal sequences and dependencies. The elevated F1-score accentuates its efficacy in sequence prediction tasks.
- LSTM ([64,128] Units): With high accuracy and the highest precision among all models, coupled with moderate recall, this configuration reveals LSTM's strength in handling long-term dependencies. The commendable F1-score indicates a well-balanced model suitable for complex sequential tasks.
- GRU (256 Units): Achieving the highest accuracy and F1-score, along with very high precision and recall, this model configuration showcases its efficiency in temporal pattern recognition. Its performance suggests a slight edge over LSTM in efficiency, making it highly suitable for sequence modeling tasks.

5.2.12.3 Hybrid Models:

- CNN_RNN (256 Units): Very high accuracy, precision, recall, and F1-score in this configuration indicate a robust capability to analyze spatial-temporal data, making it ideal for applications involving both spatial and temporal dynamics
- CNN_LSTM (256 Units): Achieving top marks in accuracy, precision, recall, and the highest F1-score, this model highlights the synergy between CNN's spatial analysis and LSTM's temporal processing capabilities. It's particularly effective for complex spatial-temporal challenges.
- CNN_GRU (256 Units): With the highest accuracy and strong performance across all metrics, this configuration combines CNN's spatial feature extraction with GRU's efficient temporal modeling. It stands out for spatial-temporal analysis tasks, especially where computational efficiency is paramount.

Table 27 indicates that the CNN_LSTM hybrid model with 256 units is the most optimal, given its unparalleled balance and performance in handling datasets with both spatial and temporal dimensions. This model is the preferred choice for tackling complex spatial-temporal analysis tasks, offering the best of both worlds in terms of spatial and temporal data processing capabilities.

Table 27: Comparison of the Best Model Configurations Across Different Model Types

Models	Config	Accuracy	Precision	Recall	F1-score
SVM	0.01	0.448	0.789	0.330	0.290
DT	64	0.512	0.425	0.382	0.717
KNN	8	0.483	0.455	0.338	0.583
ANN	256	0.790	0.817	0.595	0.659
CNN	[128,256]	0.858	0.831	0.667	0.760
RNN	[128,256]	0.973	0.976	0.806	0.817
LSTM	[64,128]	0.794	0.847	0.584	0.613
GRU	256	0.978	0.981	0.817	0.827
CNN_RNN	256	0.968	0.983	0.798	0.810
CNN_LSTM	256	0.978	0.980	0.826	0.833
CNN_GRU	256	0.979	0.971	0.818	0.828

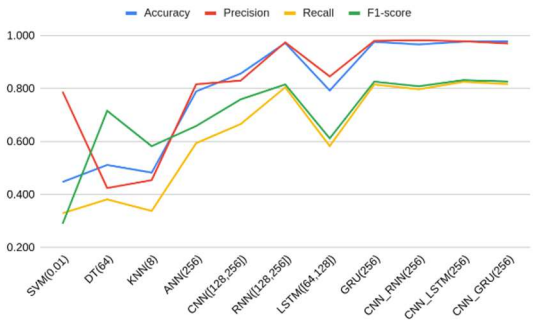

Figure 14: Visual Representation of Optimal Model Configurations and Their Performance Metrics

In our study, while the integration of AI with IIoT for fault detection in three-phase systems has shown promising results, there are limitations that warrant further discussion. The complexity of real-world industrial environments, with their myriad operational variables and conditions, poses significant challenges to the universal applicability of our findings. Additionally, the scope of our research was constrained by the available data and the specific types of faults analyzed, which may not encompass the full spectrum of anomalies encountered in industrial settings.

Furthermore, the computational demands of advanced AI models necessitate a balance between accuracy and real-time processing capabilities, which could limit their deployment in resource-constrained environments. These limitations highlight the need for ongoing research to refine AI methodologies, enhance their adaptability to diverse industrial contexts, and optimize their computational efficiency.

Addressing these open issues will require a multifaceted approach, including the development of

more robust datasets that capture a wider range of fault scenarios, the exploration of scalable AI models that maintain high accuracy without excessive computational costs, and the implementation of adaptive algorithms that can learn from new data in dynamic industrial environments.

6. CONCLUSION

Building on the diverse contributions of previous works, which have underscored the potential of AI-based solutions in fault detection across various electrical systems and industries, our study aims to delve deeper within the context of Industry 4.0, particularly through the Industrial Internet of Things (IIoT) framework. Unlike prior research that often concentrated on specific AI models or fault types, we conduct a comprehensive analysis of a broad spectrum of machine learning and AI models. This includes Support Vector Machines (SVM), Decision Trees, K-Nearest Neighbors (KNN), Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRU). Our research is distinctive in its inclusivity, considering any type of three-phase load and even accommodating single-phase scenarios, thus extending the applicability of our findings across a broader range of industrial settings.

Moreover, our solution addresses a multiclass classification problem, testing on an extensive array of faults, specifically 57 different types, which surpasses the scope of fault types considered in previous studies. This comprehensive fault coverage ensures a more robust and versatile fault detection system. Additionally, our approach uniquely focuses on electrical parameters, 47 in total, as its input, which streamlines the data acquisition process and enhances the model's applicability and ease of integration into existing industrial systems. While previous studies have laid a solid foundation by demonstrating the effectiveness of AI and ML in fault detection, they often limit their focus to specific fault types or aspects of electrical systems. Our study seeks to overcome these limitations by providing a holistic and inclusive solution that not only covers a broader spectrum of fault types but also supports a wider variety of load conditions within the IIoT framework of Industry 4.0. This holistic approach not only advances fault detection technologies but also aligns with the evolving needs of modern industrial environments, where versatility, comprehensiveness, and seamless integration are paramount.

REFERENCES:

[1] F. Correa, "Integrating Industry 4.0 Associated Technologies into Automated and Traditional Construction," Oct. 2020, doi: 10.22260/ISARC2020/0041.

[2] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Trans. Ind. Informatics*, vol. 14, no. 11, pp. 4724–4734, Nov. 2018, doi: 10.1109/TII.2018.2852491.

[3] H. M. Cheng *et al.*, "Error analysis of the three-phase electrical energy calculation method in the case of voltage-loss failure," *Metrol. Meas. Syst.*, vol. 26, no. 3, pp. 505–516, Jan. 2019, doi: 10.24425/mms.2019.129575.

[4] H. R. Chi, C. K. Wu, N. F. Huang, K. F. Tsang, and A. Radwan, "A Survey of Network Automation for Industrial Internet-of-Things Toward Industry 5.0," *IEEE Trans. Ind. Informatics*, vol. 19, no. 2, pp. 2065–2077, Feb. 2023, doi: 10.1109/TII.2022.3215231.

[5] G. Li, J. Wu, S. Li, W. Yang, and C. Li, "Multitentacle Federated Learning Over Software-Defined Industrial Internet of Things Against Adaptive Poisoning Attacks," *IEEE Trans. Ind. Informatics*, vol. 19, no. 2, pp. 1260–1269, Feb. 2023, doi: 10.1109/TII.2022.3173996.

[6] C. Song, S. Liu, G. Han, P. Zeng, H. Yu, and Q. Zheng, "Edge-Intelligence-Based Condition Monitoring of Beam Pumping Units Under Heavy Noise in Industrial Internet of Things for Industry 4.0," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3037–3046, Feb. 2023, doi: 10.1109/JIOT.2022.3141382.

[7] J. Long, W. Liang, K. C. Li, Y. Wei, and M. D. Marino, "A Regularized Cross-Layer Ladder Network for Intrusion Detection in Industrial Internet of Things," *IEEE Trans. Ind. Informatics*, vol. 19, no. 2, pp. 1747–1755, Feb. 2023, doi: 10.1109/TII.2022.3204034.

[8] O. Aouedi, K. Piamrat, G. Muller, and K. Singh, "Federated Semisupervised Learning for Attack Detection in Industrial Internet of Things," *IEEE Trans. Ind. Informatics*, vol. 19, no. 1, pp. 286–295, Jan. 2023, doi:

10.1109/TII.2022.3156642.

[9] L. Ma, X. Wang, X. Wang, L. Wang, Y. Shi, and M. Huang, "TCDA: Truthful Combinatorial Double Auctions for Mobile Edge Computing in Industrial Internet of Things," *IEEE Trans. Mob. Comput.*, vol. 21, no. 11, pp. 4125–4138, Nov. 2022, doi: 10.1109/TMC.2021.3064314.

[10] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE Trans. Ind. Informatics*, vol. 14, no. 8, pp. 3690–3700, Aug. 2018, doi: 10.1109/TII.2017.2786307.

[11] A. G. Frank, L. S. Dalenogare, and N. F. Ayala, "Industry 4.0 technologies: Implementation patterns in manufacturing companies," *Int. J. Prod. Econ.*, vol. 210, pp. 15–26, Apr. 2019, doi: 10.1016/J.IJPE.2019.01.004.

[12] Y. Zerguit, Y. Hammoudi, I. Idrissi, and M. Derrhi, "A RELAY-BASED AUTOMATIC BALANCING SYSTEM FOR THREE-PHASE LOADS IN INDUSTRY 4.0," *J. Theor. Appl. Inf. Technol.*, vol. 101, no. 5, pp. 2012–2026, Mar. 2023.

[13] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," *J. Ind. Inf. Integr.*, vol. 6, pp. 1–10, Jun. 2017, doi: 10.1016/J.JII.2017.04.005.

[14] Y. Zerguit, Y. Hammoudi, and M. Derrhi, "IMPLEMENTING ADVANCED POWER QUALITY AND EFFICIENCY SOLUTIONS IN INDUSTRY 4.0: THE ROLE OF DYNAMIC LOAD BALANCING AND POWER FACTOR CORRECTION," *J. Theor. Appl. Inf. Technol.*, vol. 102, no. 2, pp. 460–479, Jan. 2024.

[15] J. M. Guerrero, M. Chandorkar, T. L. Lee, and P. C. Loh, "Advanced control architectures for intelligent microgridspart i: Decentralized and hierarchical control," *IEEE Trans. Ind. Electron.*, vol. 60, no. 4, pp. 1254–1262, 2013, doi: 10.1109/TIE.2012.2194969.

[16] Y. Zerguit, Y. Hammoudi, and M. Derrhi, "ENERGY EFFICIENCY IN INDUSTRY 4.0: IMPLEMENTING REAL-TIME POWER FACTOR CORRECTION IN SMART MANUFACTURING," *J. Theor. Appl. Inf. Technol.*, vol. 101, no. 1, pp. 359–373, Jan. 2024.

[17] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Futur. Gener. Comput. Syst.*, vol. 100, pp. 779–796, Nov. 2019, doi: 10.1016/J.FUTURE.2019.05.041.

[18] N. J. Tierney, F. A. Harden, M. J. Harden, and K. L. Mengersen, "Using decision trees to understand structure in missing data," *BMJ Open*, vol. 5, no. 6, p. e007450, Jan. 2015, doi: 10.1136/bmjopen-2014-007450.

[19] L. Y. Hu, M. W. Huang, S. W. Ke, and C. F. Tsai, "The distance function effect on k-nearest neighbor classification for medical datasets," *Springerplus*, vol. 5, no. 1, pp. 1–9, Dec. 2016, doi: 10.1186/S40064-016-2941-7/FIGURES/8.

[20] S. Vieira, W. H. L. Pinaya, and A. Mechelli, "Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications," *Neurosci. Biobehav. Rev.*, vol. 74, pp. 58–75, Mar. 2017, doi: 10.1016/j.neubiorev.2017.01.002.

[21] N. Ouerdi, I. Elfarissi, A. Azizi, and M. Azizi, "Artificial neural network-based methodology for vulnerabilities detection in EMV cards," *Proc. 2015 11th Int. Conf. Inf. Assur. Secur. IAS 2015*, pp. 85–90, Jun. 2016, doi: 10.1109/ISIAS.2015.7492750.

[22] R. Vinayakumar, K. P. Soman, and P. Poornachandrany, "Applying convolutional neural network for network intrusion detection," *2017 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2017*, vol. 2017, pp. 1222–1228, Nov. 2017, doi: 10.1109/ICACCI.2017.8126009.

[23] W. Tao, W. Zhang, C. Hu, and C. Hu, "A Network Intrusion Detection Model Based on Convolutional Neural Network," *Adv. Intell. Syst. Comput.*, vol. 895, pp. 771–783, Jan. 2020, doi: 10.1007/978-3-030-16946-6_63.

[24] B. Jang, M. Kim, G. Harerimana, S. U. Kang, and J. W. Kim, "Bi-LSTM model to increase accuracy in text classification: Combining word2vec CNN and attention mechanism," *Appl. Sci.*, vol. 10, no. 17, p. 5841, Sep. 2020, doi: 10.3390/app10175841.

[25] X. Ni, C. Shi, Y. Ma, L. Liu, and J. He, "Research on Fault Prediction Method of Electronic Equipment Based on Bi-LSTM[基于Bi-LSTM的电子装备故障预测方法研究]," *Aero Weapon.*, vol. 29, no. 6, pp. 102–110, Dec. 2022, doi:

10.12132/ISSN.1673-5048.2021.0234.

[26] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," *Phys. D Nonlinear Phenom.*, vol. 404, p. 132306, Mar. 2020, doi: 10.1016/j.physd.2019.132306.

[27] S. Yang, X. Yu, and Y. Zhou, "LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example," *Proc. - 2020 Int. Work. Electron. Commun. Artif. Intell. IWECAI 2020*, pp. 98–101, Jun. 2020, doi: 10.1109/IWECAI50956.2020.00027.

[28] J. Chen and X. Ran, "Deep Learning with Edge Computing: A Review," *Proc. IEEE*, vol. 107, no. 8, pp. 1655–1674, Aug. 2019, doi: 10.1109/JPROC.2019.2921977.

[29] N. Zhai, P. Yao, and X. Zhou, "Multivariate time series forecast in industrial process based on XGBoost and GRU," *Pages 1397 - 1400*, p. 2020, doi: 10.1109/ITAIC49862.2020.9338878.

[30] F. M. Almasoudi, "Enhancing Power Grid Resilience through Real-Time Fault Detection and Remediation Using Advanced Hybrid Machine Learning Models," *Sustain.*, vol. 15, no. 10, p. 8348, May 2023, doi: 10.3390/su15108348.

[31] P. Pietrzak and M. Wolkiewicz, "Fault Diagnosis of PMSM Stator Winding Based on Continuous Wavelet Transform Analysis of Stator Phase Current Signal and Selected Artificial Intelligence Techniques," *Electron.*, vol. 12, no. 7, p. 1543, Apr. 2023, doi: 10.3390/electronics12071543.

[32] N. O. Farrar, M. H. Ali, and D. Dasgupta, "Artificial Intelligence and Machine Learning in Grid Connected Wind Turbine Control Systems: A Comprehensive Review," *Energies 2023, Vol. 16, Page 1530*, vol. 16, no. 3, p. 1530, Feb. 2023, doi: 10.3390/EN16031530.

[33] A. Polenghi, L. Cattaneo, and M. Macchi, "A framework for fault detection and diagnostics of articulated collaborative robots based on hybrid series modelling of Artificial Intelligence algorithms," *J. Intell. Manuf.*, 2023, doi: 10.1007/s10845-023-02076-6.

[34] U. Albertin, G. Pedone, M. Brossa, G. Squillero, and M. Chiaberge, "A Real-Time Novelty Recognition Framework Based on Machine Learning for Fault Detection," *Algorithms*, vol. 16, no. 2, p. 61, Feb. 2023, doi: 10.3390/a16020061.

[35] I. Preethi, S. Suryaprakash, and M. Mathankumar, "A State-of-Art Approach on Fault Detection in Three Phase Induction Motor using AI Techniques," *Proc. - 5th Int. Conf. Comput. Methodol. Commun. ICCMC 2021*, pp. 567–573, Apr. 2021, doi: 10.1109/ICCMC51019.2021.9418444.

[36] J. Hong, Y. H. Kim, H. Nhung-Nguyen, J. Kwon, and H. Lee, "Deep-Learning Based Fault Events Analysis in Power Systems," *Energies*, vol. 15, no. 15, p. 5539, Aug. 2022, doi: 10.3390/en15155539.

[37] B. E. A. Badr, I. Altawil, M. Almomani, M. Al-Saadi, and M. Alkhurainej, "Fault Diagnosis of Three-Phase Induction Motors Using Convolutional Neural Networks," *Math. Model. Eng. Probl.*, vol. 10, no. 5, pp. 1727–1736, Oct. 2023, doi: 10.18280/mmep.100523.