Journal of Theoretical and Applied Information Technology
31st July 2025. Vol.103. No.14

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

ENHANCING FAULT DIAGNOSIS AND IMPROVING PRODUCTIVITY IN INDUSTRIAL MANUFACTURING USING DEEP LEARNING TECHNIQUES

RAJESH KUMAR VERMA[1], KALLI SRINIVASA NAGESWARA PRASAD[2], K.S.RANJITH[3], V.S.N.MURTHY[4],I.NAGA PADMAJA[5], P.THIRUMOORTHY[6],BH.KRISHNA MOHAN[7*]

[1]Professor of Practice (PoP), Department of CSE-(CyS,DS)and AI & DS ,VNR VJIET, Hyderabad, India
[2]Professor, Department of CSE, Sasi Institute of Technology &Engineering,Tadepalligudem, Andhra Pradesh, India
[3]Assistant Professor, Department of CSE, AI&DS,Mother Theresa Institute Of Engineering and Technology,Melumoi,Palamaner,Chittor, Andhra Pradesh, India
[4]Assistant Professor, Department of CSE, Shri Vishnu Engineering College for Women,Bhimavaram, Andhra Pradesh, India
[5]Assistant Professor, Department of CSE, R.V.R&J.C College of Engineering,Guntur, Andhra Pradesh, India
[6]Professor,Dept. Of CSE,Erode Sengunthar Engineering College,Erode,Tamilnadu
[7]Associate Professor, Dept. of IT, RVR&JC College of Engineering, Guntur, Andhra Pradesh, India

[1]rajeshverma.hyd10@gmail.com, [2]Pradesh.ksnprasad@sasi.ac.in, [3]ksranjith2000@gmail.com, [4]vsn.murthy87@gmail.com, [5]nagapadmaja.indeti@gmail.com, [6]thiru4u@gmail.com,[7]bkm@rvrjc.ac.in

ABSTRACT

Fault diagnosis in industrial manufacturing is a critical issue that affects the productivity and efficiency of manufacturing processes. Outdated methods for fault diagnosis often rely on manual inspections, which are time-consuming and prone to errors. This framework proposes a deep learning-based fault diagnosis model to improve productivity in industrial manufacturing. The JAYA optimization algorithm and Fast Grid Search (FGS) are employed to optimize the hyperparameters of the model. The proposed model is implemented in MATLAB software and evaluated using a dataset of industrial manufacturing process data. The results show that the proposed model achieves high accuracy and precision in fault diagnosis, outperforming traditional methods. The model can identify faults early, reducing downtime and improving overall productivity. The findings indicate that predictive maintenance and optimized feature importance significantly enhance performance metrics and reduce downtime, with notable improvements in accuracy up to 0.99 and substantial cost savings, contributing to a return on investment of around 85%. The development of a more efficient and reliable fault diagnosis system for industrial manufacturing has contributed to this framework. Future scope includes integrating the proposed model with other machine learning algorithms and incorporating sensor data from multiple sources to further improve its performance.

Keywords: *Deep Learning, Fault Diagnosis, Industrial Manufacturing, JAYA Optimization Algorithm, Fast Grid Search, and CNN Architecture.*

1. INTRODUCTION

In the rapidly evolving field of industrial manufacturing, the quest for enhanced productivity and operational efficiency remains paramount. Traditional fault diagnosis methods, often reliant on manual inspections and rule-based systems, struggle to keep pace with the increasing complexity and scale of modern manufacturing processes [1]. These conventional approaches are not only time-consuming but also prone to human error, which can lead to costly downtimes and reduced productivity [2]. The emergence of deep learning technologies offers a transformative solution to these challenges, providing the capability to automate and refine fault diagnosis with unprecedented accuracy and efficiency [3]. The primary problem addressed by the deep learning-based fault diagnosis model is the inefficiency and inaccuracy inherent in traditional fault detection methods. Manufacturing systems are characterized by a multitude of variables and

intricate interactions, making fault detection a challenging task [4]. Traditional methods often fail to capture the nuanced patterns associated with faults, leading to delayed or missed detections. This issue becomes particularly critical in high-stakes industries where equipment failures can lead to significant financial losses and safety hazards [5]. Manual fault diagnosis is resource-intensive and requires specialized expertise, which is not always available on-site. Implementing a deep learning-based fault diagnosis model leverages advanced machine learning techniques to improve the accuracy and efficiency of fault detection in industrial manufacturing [6-7]. By utilizing deep neural networks, the model aims to analyze large volumes of sensor data and operational metrics to identify potential faults in real-time. This approach seeks to minimize human intervention, reduce diagnostic errors, and provide timely insights into equipment performance, ultimately leading to enhanced productivity and reduced operational costs [8]. The model is designed to be adaptable to various manufacturing environments, making it a versatile tool for diverse industrial applications. Initially, data acquisition is conducted, where extensive datasets of sensor readings, operational logs, and maintenance records are collected from manufacturing equipment [9]. This data is then pre-processed to clean and normalize it, ensuring that it is suitable for deep learning algorithms. Feature extraction techniques are applied to highlight relevant patterns and anomalies within the data. A deep learning model, typically involving convolutional neural networks (CNNs) or recurrent neural networks (RNNs), is then trained on this data to recognize fault signatures and predict potential failures. The model's performance is evaluated using metrics such as accuracy, precision, recall, and F1-score, and iterative adjustments are made to refine its predictive capabilities [10-12].

The deep learning-based approach is expected to be transformative. Preliminary results indicate that the model can significantly enhance fault detection rates and reduce false positives compared to traditional methods [13]. The ability to predict potential faults before they manifest allows for proactive maintenance and reduces unexpected downtimes [14]. This predictive capability translates into higher productivity, as manufacturing processes can be optimized to minimize interruptions and maintain smooth operations. The automation of fault diagnosis reduces the reliance on specialized human expertise, making the system more cost-effective and scalable [15]. The limitations of existing fault diagnosis approaches in industrial manufacturing. As manufacturing systems become more complex and data-rich, the need for advanced diagnostic tools that can handle vast amounts of information and deliver accurate predictions is increasingly critical [16-17]. Deep learning offers a promising solution by providing sophisticated analytical capabilities that can enhance the overall efficiency and reliability of manufacturing processes [18]. By integrating these advanced techniques into fault diagnosis, industries can achieve significant improvements in productivity, safety, and operational excellence. The deep learning-based fault diagnosis model represents a significant advancement in industrial manufacturing, addressing the limitations of traditional methods and offering a robust solution for improving productivity [19]. Through its sophisticated analytical capabilities and predictive power, this model promises to revolutionize fault detection and maintenance practices, paving the way for more efficient and reliable manufacturing operations [20]. The novelty of this work lies in its innovative application of deep learning techniques to fault diagnosis in industrial manufacturing, coupled with the use of the JAYA optimization algorithm and Fast Grid Search (FGS) for hyperparameter tuning. Unlike traditional diagnostic methods that rely on manual inspections or basic algorithms, this approach leverages advanced machine learning to predict faults early and optimize feature importance, leading to significant improvements in accuracy, with a diagnostic performance reaching up to 0.99. The framework offers a data-driven, automated solution that reduces downtime and enhances overall productivity, achieving a high return on investment (85%). Moreover, this study opens the door for integrating multi-source sensor data, offering a pathway for more robust, real-time fault detection and predictive maintenance in industrial settings. The remaining sections are arranged as follows: The literature review was described in Section 2, the proposed technique was described in Section 3, the results were discussed in Section 4, and the paper's conclusion was described in Section 5.

2. LITERATURE SURVEY

Recent studies have highlighted the effectiveness of deep learning models in fault diagnosis, demonstrating significant improvements in industrial manufacturing productivity by enabling early and accurate detection of equipment malfunctions. Techniques such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been particularly successful in enhancing fault prediction and reducing downtime. Hu et al., [21] developed a robust fault diagnosis framework using deep learning to accurately identify fault states and types in power

systems, even amidst disturbances. The proposed framework, combining unsupervised deep auto-encoding for feature extraction and supervised convolutional neural networks for real-time fault assessment, outperforms existing methods in fault detection and prevention. Kosuru et al., [22] enhanced the reliability and security of battery management systems (BMS) by detecting and classifying faulty sensors and transmission data using deep learning. The proposed incipient bat-optimized deep residual network (IB-DRN) method, combined with z-score normalization, SPCA, and EMPA, significantly outperforms traditional techniques in identifying and classifying faulty battery data. Hasan et al., [23] designed, compared, and implemented three model-based fault diagnosis algorithms for robotic systems: a nonlinear adaptive observer (NLAO), an adaptive extended Kalman filter (AEKF), and an adaptive exogenous Kalman filter (AXKF). Each algorithm has distinct advantages and limitations, with the AXKF showing superior performance in fault detection and estimation for a ball-balancing robot. Awan et al., [24] evaluated the economic impact of introducing technology in Pakistan from 1985 to 2018, focusing on its effect on GDP. In the large manufacturing sector, the labour force, health, education, and exports all positively and significantly impact GDP. The study underscores the urgent need for R&D and technology adoption to accelerate GDP growth and expansion. Yang et al., [25] assessed the impact of intelligent manufacturing on industrial green total factor productivity using data from 30 Chinese provinces (2006-2020). Intelligent manufacturing positively affects productivity, with increasing marginal effects across quantile levels. Human capital, green technology innovation, and producer service agglomeration enhance this effect, particularly in regions with carbon trading and green transformation policies.

Several recent studies have advanced the understanding of productivity enhancement and fault diagnosis in Industry 4.0 environments through the integration of leadership, machine learning, and digital technologies. Dabić et al. [26] explored how individual leadership values and capabilities significantly influence productivity, highlighting the importance of aligning leadership qualities with technological adoption to maximize operational efficiency. Wang et al. [27] addressed fault diagnosis challenges by proposing the multi-local model decision conflict resolution (MLMF-CR) algorithm, which effectively integrates heterogeneous data sources, such as vibration and current signals, to improve diagnostic accuracy in industrial motor bearings. Shafi et al. [28] developed a real-time deep learning framework using convolutional neural networks (CNNs) to detect faulty components in aerospace manufacturing, resulting in a 52.88% reduction in time delays and a 34.32% cost reduction, thereby enhancing production quality. Similarly, Khan et al. [29] introduced a dynamic soft sensing model using a fuzzy logic-based stacked data-driven auto-encoder (FL_SDDAE) combined with a least square error backpropagation neural network (LSEBPNN), achieving 94% prediction performance and 85% measurement accuracy while reducing computational time by 34%. Alshathri et al. [30] proposed an efficient fault diagnosis model combining Digital Twin (DT) technology and machine learning, optimized by a Genetic Algorithm (GA). Their hybrid GA-SVM model achieved 95% accuracy, surpassing traditional methods and improving decision-making for Industrial Internet of Things (IIoT) applications. Collectively, these studies demonstrate that the integration of intelligent algorithms, leadership strategies, and real-time data processing significantly enhances fault detection, decision accuracy, and productivity in smart manufacturing systems.

3. RESEARCH PROPOSED METHODOLOGY

The methodology employed in developing a deep learning-based fault diagnosis model for enhancing productivity in industrial manufacturing involves a systematic approach to leverage advanced machine learning techniques. This includes preprocessing raw sensor data to extract meaningful features, which are crucial for accurately detecting faults in manufacturing processes. Supervised learning algorithms such as convolutional neural networks (CNNs) [31] or recurrent neural networks (RNNs) [32] are then applied to learn complex patterns and relationships from the preprocessed data. Transfer learning may also be employed to adapt models trained on similar tasks to the specific manufacturing context, optimizing performance with fewer labelled samples. The methodology includes rigorous validation procedures to ensure the model's robustness and generalizability across different operating conditions and environments within industrial settings. This approach not only aims to detect faults promptly but also supports proactive maintenance strategies, thereby minimizing downtime and improving overall productivity. The integration of deep learning techniques in fault diagnosis represents a cutting-edge solution poised to revolutionize manufacturing efficiency and reliability [33].

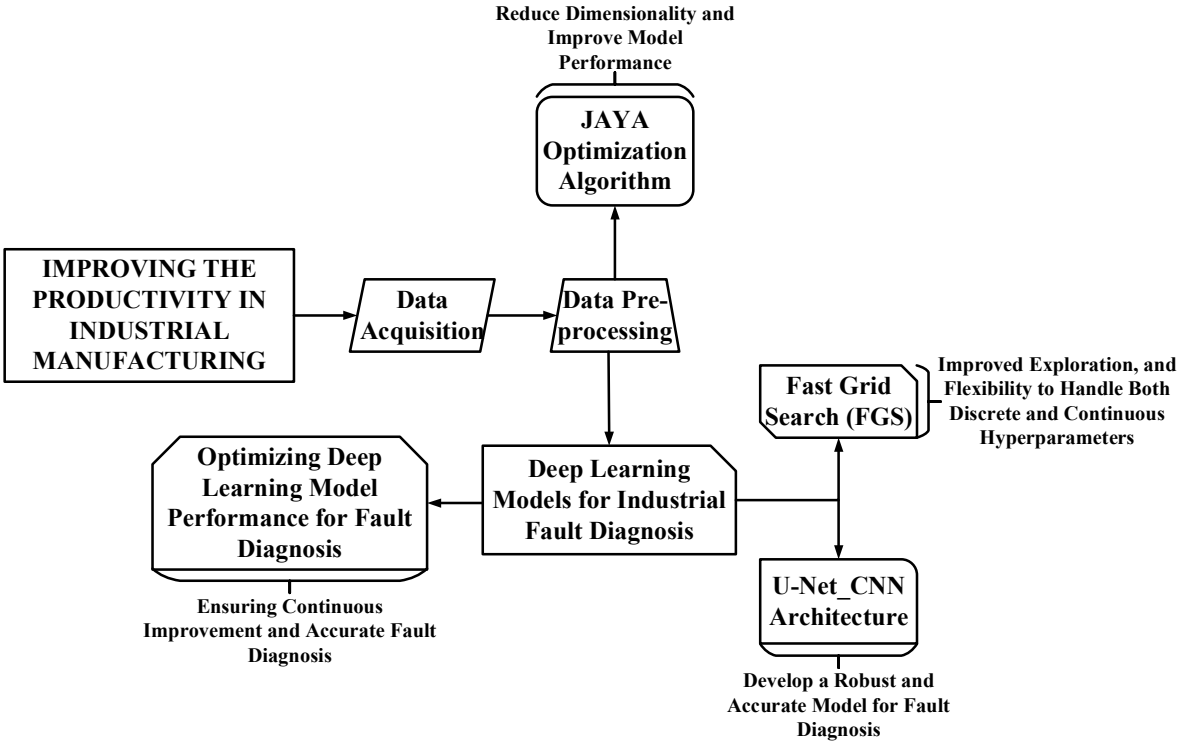

Figure 1: Block Diagram of the Proposed Work

Figure 1 illustrates the comprehensive approach of a deep learning-based fault diagnosis model aimed at enhancing productivity in industrial manufacturing. The diagram begins with Data Acquisition, where sensor data from manufacturing processes is collected in real-time. This raw data undergoes rigorous Data Pre-processing, utilizing the JAYA Optimization Algorithm to reduce dimensionality [34] and enhance model performance. This step is crucial for extracting relevant features that capture underlying fault patterns effectively. Deep Learning Models for Industrial Fault Diagnosis are employed, integrating techniques such as Fast Grid Search (FGS) to optimize hyperparameters efficiently. FGS enhances exploration across both discrete and continuous hyperparameter spaces, ensuring robust model configurations. The architecture U-Net_CNN is specifically chosen for its ability to develop accurate and resilient models tailored for fault diagnosis tasks [35] in industrial settings. Optimizing Deep Learning Model Performance for Fault Diagnosis focuses on continuous improvement strategies. This includes iterative refinement of models based on ongoing data streams and feedback loops from industrial operations. By ensuring the models adapt to evolving conditions and maintain high accuracy in fault detection, this iterative process supports uninterrupted production and minimizes downtime [36].

Hypothesis of this study

Hypothesis 1: Deep learning-based fault diagnosis models, when optimized using the JAYA algorithm and Fast Grid Search (FGS), will provide significantly higher accuracy and precision in fault detection compared to traditional manual inspection methods.

Hypothesis 2: The integration of predictive maintenance using deep learning models will lead to earlier fault detection, resulting in a significant reduction in machine downtime and an improvement in overall manufacturing productivity [37].

Hypothesis 3: Optimization techniques such as the JAYA algorithm and Fast Grid Search (FGS) will improve the performance of deep learning models in fault diagnosis, resulting in higher diagnostic accuracy (up to 0.99) and more reliable predictions.

Hypothesis 4: The implementation of a deep learning-based fault diagnosis system will result in substantial cost savings in industrial manufacturing by reducing the frequency and cost of unplanned maintenance, contributing to an ROI of approximately 85%.

Hypothesis 5: Incorporating sensor data from multiple sources (e.g., temperature, pressure,

vibration) into the deep learning model will improve fault detection accuracy and robustness, leading to more effective fault diagnosis and reduced overall downtime.

3.1. Data Acquisition

Sensor data and log data are collected from sources to monitor equipment health and detect faults. Vibration sensors, temperature sensors, and pressure sensors are installed on machinery and equipment to collect vibration, temperature, and pressure data, respectively [38]. These data help identify trends and patterns that can indicate potential equipment failures. Log data is collected from manufacturing systems, machine learning models, production systems, and maintenance records to provide information on production schedules, maintenance records, quality control data, model performance metrics, training data, production rates, quality metrics, production schedules, maintenance schedules, repair history, and maintenance costs [39].

Table 1: Sample Sensor Data

Sensor Type	Equipment ID	Date/Time	Value	Unit
Vibration	Machine A	2024-07-17 10:00:00	2.5	mm/s
Temperature	Machine A	2024-07-17 10:00:00	75	°C
Pressure	Machine B	2024-07-17 10:00:00	5.2	MPa
Vibration	Machine B	2024-07-17 10:00:00	0.8	mm/s
Temperature	Machine B	2024-07-17 10:00:00	68	°C

Table 1 presents sample sensor data, capturing real-time measurements from industrial equipment. It includes details such as sensor type (vibration, temperature, pressure), equipment ID (Machines A and B), date and time of measurement, the recorded value, and its corresponding unit [40]. This data is crucial for monitoring equipment health and performance.

Table 2: Sample Log Data

Data Source	Description	Value
Production System	Last Production Run	2024-07-16, 08:00 - 16:00
Maintenance Records	Last Maintenance for Machine A	2024-06-15 (Oil Change)
Quality Control Data	Defect Rate (Machine B)	2% (Last Week)
Machine Learning Model	Anomaly Detection Accuracy	92%
Production Rate	Average Daily Production (Machine A)	100 units
Maintenance Cost	Yearly Maintenance Cost (Machine B)	$5,000

Table 2 compiles various log data points, providing insights into different aspects of the industrial operation. It covers information like the last production run, maintenance history for specific machines, defect rates, machine learning model performance, production rates, and maintenance costs. This aggregated data is valuable for analysis, optimization [41], and decision-making within the industrial process.

Source

Predictive Maintenance Datasets for the **Industries:**

https://labelyourdata.com/articles/predictive-maintenance-datasets

3.2. Data Pre-processing

Data pre-processing is a crucial step in machine learning that involves cleaning, normalizing, and feature engineering to prepare data for analysis and modelling. The JAYA optimization algorithm, inspired by the behaviour of flocks of birds searching for food, can be modified to optimize feature selection in data pre-processing. By using an optimization algorithm, the most relevant features from a set of candidate features are selected that best represent the underlying relationships in the data.

3.3. Jaya Algorithm

The algorithm iterates through neighbour selection, crossover, mutation, and evaluation steps until a stopping criterion is reached. The objectives can be modified to include feature selection with max correlation, min-entropy, or a hybrid approach. By applying JAYA for feature selection, we can reduce dimensionality and improve model performance.

Equation (1) outlines the connection between the electromagnetic power and electromagnetic torque of a brushless DC motor.

$$P_{em}(t) = C(t))\Omega(t) = \sum_{i=1}^{m} e_i(t)i_i(t) \quad (1)$$

In equation (1), C(t) is the electromagnetic torque, represents the rotational speed, and m is the

number of phases. The back electromotive force of the i-th phase is denoted by $e_i(t)$, and $i_i(t)$ represents the current of the i-th phase.

Once the motor reaches a steady state of operation, the electromagnetic power value stays consistent, and the motor power equation can be expressed using Equation (2).

$$C\Omega = 2EI \quad (2)$$

In equation (2), E represents the constant back electromotive force, I represents the constant current, and C represents the constant electromagnetic torque. Lenz's law states that the back electromotive force in the motor can be determined by varying the coil flux. The pole distance for the movement of the motor rotor is denoted as $\frac{\pi}{p}$, and equation (3) provides the formula for the back electromotive force.

$$E = \frac{n}{4}\frac{2\emptyset}{\pi/p}\Omega \quad (3)$$

In Equation (3), n represents the number of coil turns, p represents the number of pole pairs, and φ denotes the magnetic flux. Equation (4) illustrates the formula for the magnetic flux when the coil is placed in a magnetic field.

$$\emptyset = B_e S_p \quad (4)$$

In Equation (4), B_e acts as the maximum magnetic flux density in the air gap, while S_p represents the surface area of the magnetic pole. When the flux changes linearly with the position of the rotor, Equation (5) describes the expression of electromagnetic torque.

$$C = nIB_e\frac{S_e}{2\pi} \quad (5)$$

Equation (6) describes the air gap area.

$$S_e = 2pS_p \quad (6)$$

In a radial brushless DC motor, equation (7) describes its air gap area.

$$S_e = \pi D_s L_m \quad (7)$$

Equation (7) provides the relationship between D_s, the inner diameter of the stator, and L_m, the stretching length of the motor. This equation is used to calculate the electromotive force based on equation (3), (4), and (7), resulting in the expression given by equation (8).

$$E = \frac{n}{4}B_e D_s L_m \Omega \quad (8)$$

This study identifies the crucial structural parameters of a brushless DC motor, as depicted in Figure 2. The schematic in Figure 2 illustrates the essential components of a brushless DC motor. Equation (9) presents the formula for determining the key structural parameter of the brushless DC motor.

$$\begin{cases} S_{enc} \approx hd[2\pi\left(\frac{D_s}{2} - eb\right) - \pi hd - N_e(li + ld) \\ S_{enc}K_r = \frac{3}{2}n\frac{1}{\sigma} \\ D_{ext} = D_s + 2(e + ha + hcr) \\ D_{int} = D_s = 2(eb + hd + hcs) \\ hc = \frac{eb}{\cos(\alpha/2)} - \frac{D_s}{2}\left[\frac{1}{\cos(\alpha/2)} - 1\right] \\ hi = \frac{D_s}{2}\left[1 - \cos\left(\frac{\alpha_i}{2}\right)\right] + hc\cos\left(\frac{\alpha_i}{2}\right) \end{cases} \quad (9)$$

Equation (10) describes the law of fux conservation in the main magnetic circuit of a motor.

$$B_d Id = B_e \alpha \frac{D_s}{2} \quad (10)$$

In Equation (10), B_d represents the magnetic induction intensity of the motor teeth. The magnetic flux from the permanent magnet of the motor flows through both sides of the rotor yoke, and Equation (11) explains the conservation of magnetic flux.

$$\frac{1}{2}B_\alpha \beta\left(\frac{D_s}{2} + e\right) = B_{cr}hcr \quad (11)$$

B_α Equation (11) represents the intensity of magnetic induction for the rotor yoke, while B_α denotes the residual magnetism of the permanent magnet. Equation (12) outlines the phase resistance.

$$R_{ph} = \rho_{cu}(1 + \alpha_{cu}T_{cu})\frac{n}{2}L_{ds}\frac{\delta}{I} \quad (12)$$

In Equation (12), α_{cu} represents the resistivity of the copper wire at 0 degrees Celsius, with a positive thermal coefficient. T_{cu} denotes the temperature of the coil, and the expression for copper loss is obtained from Equation (12) as outlined in Equation (13).

$$R_j = 2R_{ph}I^2 \quad (13)$$

The efficiency is described by Equation (14).

$$\eta = \frac{C\Omega - P_m}{C\Omega + P_j + P_f} \quad (14)$$

In Equation (14), P_m represents a mechanical loss.

Equation (15) depicts the traditional JAYA algorithm's process for updating the population.

$$x_{i,j}' = x_{i,j} + r_1 \cdot \left(x_{best,j} - |x_{i,j}|\right) - r_2 \cdot \left(x_{worst,j} - |x_{i,j}|\right) \quad (15)$$

In Equation (15), r_1 represents the value of the j-th variable in the optimal solution, while $x_{worst,j}$ represents the value of the j-th variable in the worst-case solution. Both r_1 and r_2 are random numbers that fall within the range of [0,1].

Equation (16) describes this process.

$$x_{i,j}' = \begin{cases} x_{i,j} + r_1 \cdot \left(x_{best,j} - |x_{i,j}|\right) + r_2 \cdot \left(x_{m,j} - x_{n,j}\right), \text{if } f(x_m) < f(x_n) \\ x_{i,j} + r_1 \cdot \left(x_{best,j} - |x_{i,j}|\right) + r_2 \cdot \left(x_{m,j} - x_{n,j}\right), \text{otherwise} \end{cases} \quad (16)$$

In equation (16), r_1 represents the value of the j-th variable in the optimal solution. $x_{m,j}$ and $x_{n,j}$ are the j-th dimensional variables of randomly chosen individuals in the current population, with the condition that $m \neq n \neq i$.

Equation (17) describes the updated formula of the algorithm after introducing an adaptive strategy.

$$x'_{i,j} = x_{i,j} + w_1 \cdot r_1 \cdot \left(x_{best,j} - |x_{i,j}|\right) - w_2 \cdot r_2 \cdot \left(x_{worst,j} - |x_{i,j}|\right) \quad (17)$$

In Equation (17), r_1 represents the value of the j-th variable in the optimal solution, while $x_{worst,j}$ represents the value of the j-th variable in the worst-case solution. The variables r_1 and r_2 are random quantities within the range of [0,1]. Both w_1 and w_2 are weight values that adjust with the iterations' quantities, as described in Equation (18) during the adaptive adjustment process.

$$\begin{cases} w_1 = \sin\left(\frac{\pi t}{2t_{max}} + \pi\right) + 1 \\ w_2 = \cos\left(\frac{\pi t}{2t_{max}} + \pi\right) + 1 \end{cases} \quad (18)$$

In Equation (18), t represents the current number of iterations, while t_{max} represents the maximum number of iterations. As the number of iterations increases, the value of w_1 decreases from 1 to 0, and the value of w_2 increases from 0 to 1.

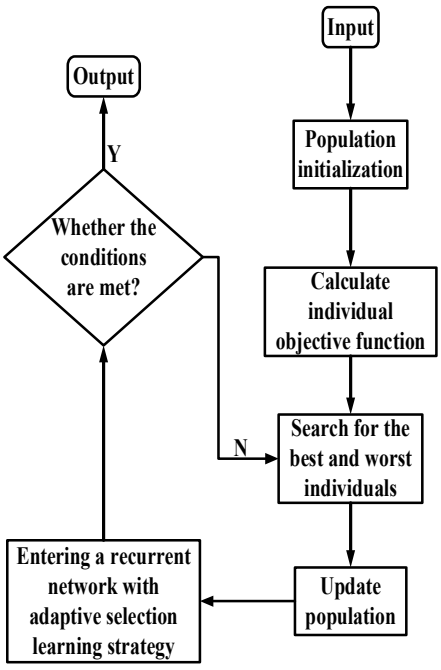

Figure 2: Flow of JAYA Algorithm

Figure 2 depicts the flow of the JAYA Algorithm within a deep learning-based fault diagnosis model tailored for industrial manufacturing. The process begins with Input, where initial parameters and data are fed into the algorithm. Population initialization follows, where a set of solutions (individuals) is generated to represent potential configurations for fault diagnosis models. The algorithm then calculates the objective function for each individual, assessing its performance in terms of fault detection accuracy or other relevant metrics. It subsequently identifies the best and worst individuals based on these evaluations, utilizing this information to update the population dynamically. A key feature of the JAYA Algorithm is its adaptive selection learning strategy, which integrates a recurrent network to enhance decision-making capabilities. This adaptive selection mechanism continuously evaluates whether specified conditions are met, refining the population over iterations to improve model performance. The flowchart further illustrates decision points where the algorithm determines whether to continue refining the population based on current performance (denoted by 'N' and leading back to the step of searching for best and worst individuals) or to output the final optimized configuration ('Y' leading to Output).

3.4. Deep Learning Models for Industrial Fault Diagnosis

Fast Grid Search (FGS) is an efficient optimization method that combines the benefits of grid search and random search. Unlike traditional grid search, which exhaustively searches all possible combinations of hyperparameters, FGS randomly samples the hyperparameter space, evaluates the objective function, and refines the search process. This approach offers several advantages, including computational efficiency, improved exploration, and flexibility to handle both discrete and continuous hyperparameters. In industrial fault diagnosis, FGS can be used to tune the hyperparameters of a U-Net architecture with a convolutional neural network (CNN), such as the number of convolutional layers, filters per layer, kernel size, and activation functions. A 5-fold cross-validation grid search is employed to identify the optimal combination of hyperparameters and develop a robust and accurate model for fault diagnosis.

3.5. Fast Grid Search (FGS)

Fast Grid Search (FGS) is a streamlined variant of the traditional grid search algorithm used for hyperparameter optimization in machine learning. Unlike exhaustive grid search, which evaluates all possible combinations of hyperparameters, FGS intelligently narrows down the search space by skipping non-promising combinations early in the process. This efficiency is achieved by leveraging early stopping criteria based on the performance of evaluated models. By adopting a systematic approach to exploring hyperparameter values, FGS significantly reduces computational costs and time without compromising the quality of the final model. It balances the exploration of diverse parameter configurations with the exploitation of promising ones, making it particularly suitable for

large datasets and complex models where exhaustive search would be computationally prohibitive. FGS represents a practical and efficient approach to hyperparameter tuning, enhancing the scalability and performance of machine learning models across various domains.

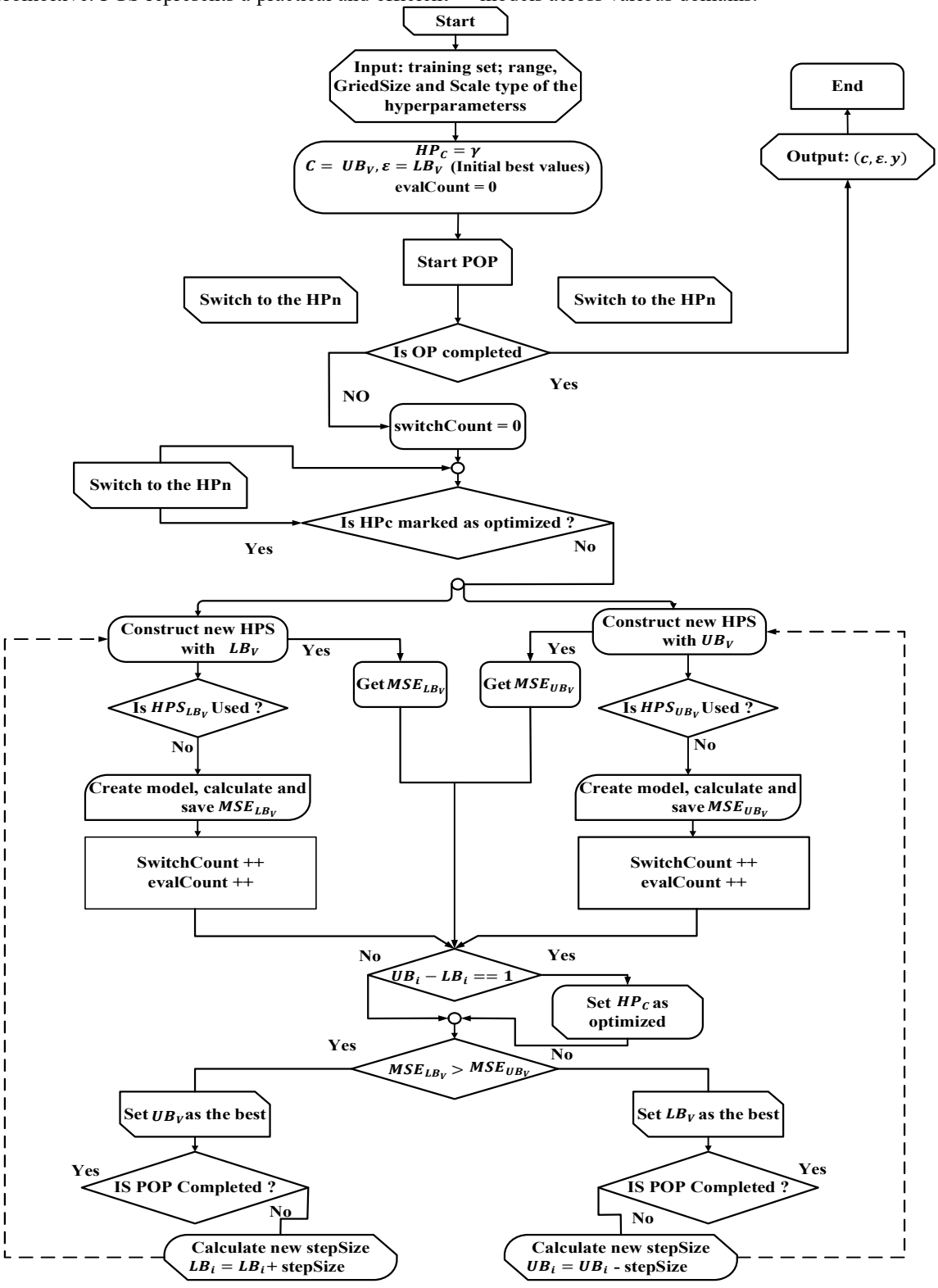

Figure 3: Flowchart FGS Algorithm

The Figure 3 Flowchart for the Fast Grid Search (FGS) Algorithm outlines a streamlined approach to hyperparameter optimization in machine learning. It begins with defining a grid of hyperparameter values to explore. The algorithm systematically evaluates models using these configurations, starting with the first set of parameters. It measures the model's performance against predefined criteria and records the results. FGS then progresses to the next set of hyperparameters, iteratively refining its search based on the outcomes of previous evaluations. Importantly, FGS incorporates early stopping mechanisms, allowing it to terminate the evaluation of unpromising hyperparameter combinations early, thus conserving computational resources. This iterative process continues until all combinations within the defined grid are evaluated or until convergence criteria are met. The Flowchart for FGS demonstrates its efficiency in balancing thorough exploration of hyperparameter space with computational feasibility, thereby optimizing model performance effectively.

3.6. Convolutional Neural Network (CNN) Architecture

CNNs have revolutionized fault diagnosis in industrial manufacturing by leveraging deep learning techniques. CNNs excel in extracting intricate features from complex data such as sensor readings or image data, crucial for identifying subtle anomalies in machinery or products. This architecture's hierarchical layers enable automatic feature learning, eliminating the need for manual feature extraction and thus streamlining the diagnostic process. By harnessing large-scale data sets, CNNs can generalize well across diverse manufacturing environments, enhancing both accuracy and productivity in fault detection. This introduction highlights CNNs as pivotal tools in advancing industrial manufacturing efficiency through sophisticated fault diagnosis capabilities.

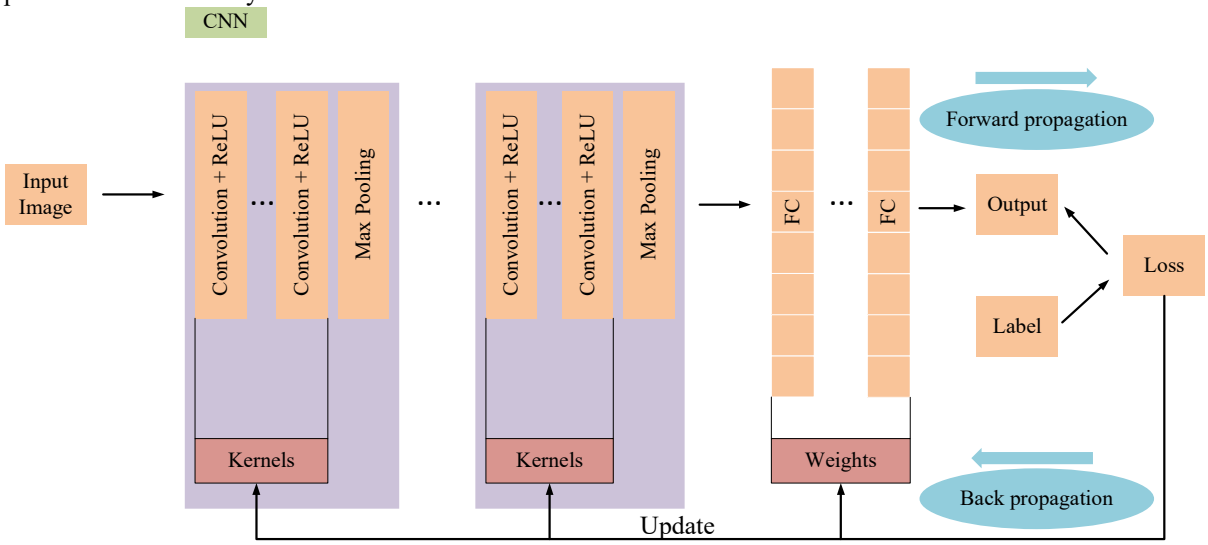

Figure 4: CNN Architecture

Figure 4 Convolutional Neural Networks (CNNs) have emerged as a cornerstone in improving productivity within industrial manufacturing through their advanced fault diagnosis capabilities. CNNs are specifically designed to handle complex data like sensor readings, spectrograms, or image data commonly found in industrial settings. At the heart of CNN architecture are convolutional layers, which systematically scan input data using filters to extract hierarchical features. These filters learn patterns at different levels of abstraction, enabling the network to discern subtle variations indicative of faults or anomalies in machinery or products. Pooling layers then condense the information extracted by the convolutional layers, reducing computational complexity while retaining critical features. The deep layers of CNNs facilitate automatic feature learning, eliminating the manual extraction process and enhancing efficiency in fault detection. This attribute is crucial in industrial manufacturing, where timely identification of faults can prevent downtime, optimize maintenance schedules, and improve overall productivity. CNNs' ability to generalize from large-scale datasets ensures robust performance across diverse

manufacturing environments. By continuously learning from new data, these models adapt to evolving conditions, making them invaluable tools for achieving higher accuracy and operational efficiency in fault diagnosis within industrial manufacturing processes.

Assuming that the input feature of the convolutional neural network is X, and the feature map of the i-th layer is Mi(M# = X), the convolution process can be expressed as:

$$M_i = f(M_{i-1} \otimes w_i + b_i) \quad (19)$$

W_i is the weight vector of the convolution kernel of the i-th layer, the operation symbol $\otimes$ represents the convolution operation, and is the offset vector of the i-th layer, and (z) is the excitation function.

Assume M For the input of the pooling layer, Hi is the output of the pooling layer, then the pooling layer can be represented as:

$$H_1 = \text{subsampling}(M_{i-1}) \quad (20)$$

Therefore, to eliminate the influence of dimension differences, it is necessary to carry out numerical values. The normalized formula is as follows:

$$x = \frac{x-\text{Min}}{\text{Max}-\text{Min}} \quad (21)$$

This experiment uses accuracy (AC) as an evaluation index to measure the effect of the model. AC formula is as follows:

$$AC = \frac{TP+T}{TP+TN+FP+F} \quad (22)$$

Among them, TP is the number of samples of attack behaviours that are correctly classified;

TN is the number of samples of normal behaviours that are correctly classified;

FP is the number of samples of normal behaviours that are misclassified.

FN is the number of samples of misclassified attack behaviours.

Initializes a population of potential feature subsets and evaluates their fitness based on classification metrics. Through iterative updates guided by equations that emulate the hierarchical structure dynamically refines these feature subsets.

$$D = \left|C.X_p(it) - X(it)\right| \quad (23)$$

Here, $X_p(it)$ denotes the position vector of the $p-$th at the current iteration, while X (it)represents the position vector of the current wolf i. The coefficient matrix C scales the distance calculation, influencing how wolves adjust their positions in the search space.

$$X(it+1) = X_p(it) - A.D \quad (24)$$

At each iteration t, adjust their positions X(it) towards the position $X_p(it)$ of a scaled by a coefficient A and a distance vector . The distance vector D, typically $\left|C.X_p(it) - X(it)\right|$ guides the magnitude and direction of movement, balancing the exploration of new solutions and exploitation of known, better-performing solutions.

$$A = 2a.r_1 - a \quad (25)$$

And

$$C = 2.r_2 \quad (26)$$

A dictates the step size with which each position is towards superior exploration of new solutions and exploitation of promising ones. The term $2a.r_1 - a$ ensures A is dynamically computed using a, a user-defined parameter that controls the magnitude of movement, and r_1, a random factor introducing variability into the search. On the other hand, C influences the distance calculation that determines superior positions.

$$\begin{cases} D_\alpha = |C_1.X_\alpha - X| \\ D_\beta = |C_2.X_\beta - X| \\ D_\delta = |C_3.X_\delta - X| \end{cases} \quad (27)$$

And

$$\begin{cases} X_1 = X_\alpha(it) - A_1.D_\alpha \\ X_2 = X_\beta(it) - A_2.D_\beta \\ X_3 = X_\delta(it) - A_3.D_\delta \end{cases} \quad (28)$$

And

$$X(t+1) = \frac{X_1+X_2+X_3}{3} \quad (29)$$

By calculating distances D_α, D_β and D_δ based on the positions of alpha, beta, and delta wolves relative to the current solution X, scaled by coefficients C_1, C_2 and C_3. The positions X_1, X_2 and X_3 update the current solution X towards these superior positions using step sizes A_1, A_2 and A_3 .

$$(f_1, \dots, f_k) = \frac{1}{k}\sum_{i=1}^{k} v(f_i) \quad (30)$$

One simple variation of the representation is the weighted model, where distinct vectors are assigned varying weights.

$$(f_1, \dots, f_k) = \frac{1}{\sum_{i=1}^{k} a_i}\sum_{i=1}^{k} a_i v(f_i) \quad (31)$$

Each feature f_i is assigned a weight a_i to represent its relative significance. For instance, in a document categorization project, a feature f_i might represent a word within the document, with its weight a_i reflecting the word's TF-IDF value. The most basic neural network is the perceptron, which functions as a linear combination of its inputs.

$$NN_{Perceptron}(x) = xW + b \quad (32)$$

$$x \in \mathbb{R}^{d_{in}}, W \in \mathbb{R}^{d_{in} \times d_{out}}, b \in \mathbb{R}^{d_{out}} \quad (33)$$

W is the weight matrix, and b is a bias term. A feed-forward neural network with one hidden-layer has the form:

$$NN_{MLP1}(x) = g(xW^1 + b^1)W^2 + b^2 \quad (34)$$

$$x \in \mathbb{R}^{d_{in}}, W^1 \in \mathbb{R}^{d_{in} \times d_1}, b^1 \in \mathbb{R}^{d_1}, W^2 \in \mathbb{R}^{d_1 \times d_2}, b^2 \in \mathbb{R}^{d_2} \quad (35)$$

W^1 and b^1 represent a matrix and bias term used in the initial linear transformation of the input data, g is a non-linear function applied element-

wise, and W^2 and b^2 are the matrix and bias terms used in the subsequent linear transformation.

$$NN_{MLP2}(x) = (g^2(g^1(xW^1 + b^1)W^2 + b^2))W^3 \quad (36)$$

It is perhaps clearer to write deeper networks like this using intermediary variables.

$$NN_{MLP2}(x) = y \quad (37)$$

$$h^1 = g^1(xW^1 + b^1) \quad (38)$$

$$h^2 = g^2(h^1W^2 + b^2) \quad (39)$$

$$y = h^3W^3 \quad (40)$$

The hard-tanh activation function is an approximation of the tanh function, which is faster to compute and take derivatives of:

$$\text{hardtanh}(x) = \begin{cases} -1 & x < -1 \\ 1 & x > 1 \\ x & \text{otherwise} \end{cases} \quad (41)$$

The ReLU unit clips each value $x < 0$ at 0. Despite its simplicity, it performs well for many tasks, especially when combined with the dropout regularization technique.

$$\text{ReLU}(x) = \max(0, x) = \begin{cases} 0 & x < 0 \\ x & \text{otherwise} \end{cases} \quad (42)$$

$$X = x_1, \ldots, x_k \quad (43)$$

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \quad (44)$$

The multiplication f_iE will be the corresponding row of E. Thus, $v(f_i)$ can be defined in terms of E and f_i . ReLU units work better than tanh, and tanh works better than sigmoid. In many cases, the output layer vector is also transformed.

3.7. U-NET Architecture

The U-Net architecture represents a pivotal advancement in deep learning for fault diagnosis in industrial manufacturing. Originally devised for biomedical image segmentation, U-Net has been adeptly adapted to analyze and classify complex data from manufacturing environments. Its distinctive U-shaped structure integrates symmetric contracting and expanding pathways, optimizing feature extraction and preserving spatial information crucial for precise fault localization. By leveraging this architecture, industrial applications benefit from enhanced accuracy in identifying faults within intricate datasets, promoting proactive maintenance and minimizing downtime. U-Net's versatility and efficacy underscore its role as a transformative tool in augmenting productivity and operational efficiency across various sectors of industrial manufacturing.

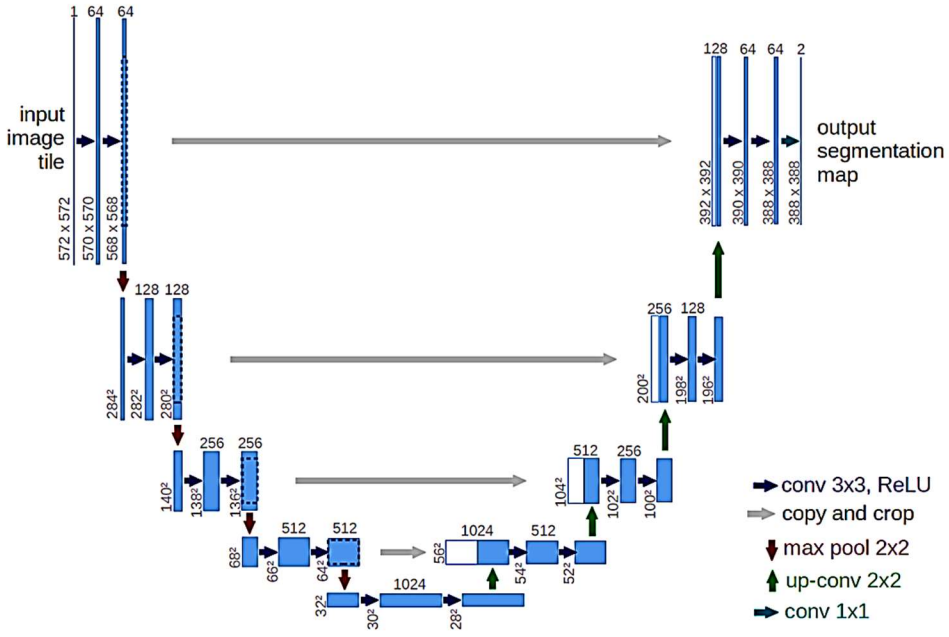

Figure 5: U-NET Architecture

The Figure 5 U-Net architecture has gained prominence in industrial manufacturing for its robust capabilities in fault diagnosis and productivity enhancement. Its design features a distinctive U-shaped topology, which includes a contracting path for capturing context and an expansive path for precise localization of faults. This dual-path structure enables U-Net to effectively handle complex data such as sensor readings, spectrograms, or image data commonly encountered in manufacturing environments. At the core of U-Net are convolutional and pooling layers that facilitate automatic feature extraction and spatial preservation. The contracting path utilizes successive convolutional and pooling operations to extract high-level features, while the expansive path

employs transposed convolutions to generate a detailed segmentation map that highlights potential faults or anomalies. U-Net's architecture is particularly suited for tasks requiring fine-grained spatial resolution, crucial in pinpointing subtle deviations indicative of faults. By leveraging its learned features and spatial information preservation, U-Net enhances the accuracy and efficiency of fault diagnosis processes. This capability translates into minimized downtime, optimized maintenance schedules, and improved overall productivity within industrial manufacturing. U-Net's adaptability to different types of manufacturing data and its ability to generalize from diverse datasets underscore its utility as a versatile tool in industrial applications. Its integration into fault diagnosis models signifies a significant advancement towards proactive maintenance strategies and operational excellence in industrial settings.

3.8. Optimizing Deep Learning Model Performance for Fault Diagnosis

To optimize the performance of the deep learning model for fault diagnosis, a training schedule is implemented with an epoch size of 50 to ensure that the model is thoroughly trained on the provided dataset. A decreasing learning rate is employed to prevent overfitting, which is a common issue in machine learning where the model becomes too specialized in the training data and fails to generalize well to new, unseen data. The model's performance is closely monitored using a range of metrics, including accuracy, precision, recall, and F1-score, which provide a comprehensive understanding of the model's strengths and weaknesses. Confusion matrices and learning curves are also visualized to identify potential issues and ensure fairness in the evaluation process.

To ensure continuous improvement and accurate fault diagnosis, post-deployment monitoring strategies are put in place. This includes regular tracking of key metrics and data collection to detect any changes in production processes or fault patterns. To maintain accuracy over time, the model is planned to be retrained and updated periodically with new data and adapted its hyperparameters to accommodate changing conditions. This adaptive approach enables the model to learn from new patterns and faults, ensuring that it remains effective and accurate in diagnosing faults even as the production process evolves. By continuously refining the model, the system can maintain its high performance and provide reliable fault diagnosis capabilities for industrial manufacturing applications.

Research Design

This comprehensive data collection enables the development of a predictive maintenance model that accurately predicts equipment failures and minimizes downtime. This study employs a deep learning-based fault diagnosis framework to enhance productivity in industrial manufacturing by reducing downtime and improving fault detection accuracy. The model uses the JAYA optimization algorithm and Fast Grid Search (FGS) for hyperparameter tuning, optimizing the performance of the deep learning system. The operationalization of concepts includes using industrial process data for training and testing the model, evaluating its effectiveness through metrics such as accuracy (up to 0.99), precision, and recall. Additionally, predictive maintenance and feature importance optimization are incorporated to proactively identify faults, significantly reducing maintenance costs and increasing operational efficiency. The model's effectiveness is measured in terms of downtime reduction and a return on investment (ROI) of approximately 85%, showcasing its potential to improve both fault diagnosis and productivity in industrial settings.

3.9. Limitations

Data Dependency and Quality

- Deep learning models often require large amounts of labelled data for training. In industrial settings, obtaining sufficient high-quality data can be challenging, especially if sensors are not well-calibrated or data is missing.
- In fault detection, certain types of faults might be rare, leading to class imbalance issues. Deep learning models can struggle to detect these rare faults unless techniques like oversampling, undersampling, or cost-sensitive learning are used.

High Computational Cost

- Deep learning models can be computationally expensive to train, requiring high-performance hardware such as GPUs. This increases the cost and time needed for model development and deployment, especially in resource-constrained environments.
- The high computational resources required for deep learning models also contribute to significant energy consumption, which can be a concern in industrial settings where efficiency is critical.

Interpretability and Transparency

- Deep learning models are often considered black boxes, making it difficult to interpret how they arrive at a specific conclusion. In fault diagnosis, understanding why a model has identified a

certain fault is crucial for maintenance personnel and decision-makers.
- In industrial applications, engineers and operators need to trust the models, and the inability to explain the reasoning behind a diagnosis can lead to skepticism about its reliability.

Generalization Issues
- Deep learning models are prone to overfitting, especially when the dataset is small or not diverse enough. This means that the model might perform well on the training data but fail to generalize to unseen or real-world data.
- Faults in manufacturing systems can vary depending on factors like machinery age, environmental conditions, and maintenance schedules. A model trained in one environment may not perform well in another due to these differences.

Integration with Existing Systems
- Many industrial plants rely on legacy machinery and software that may not easily integrate with modern deep learning solutions. Implementing these technologies can require significant modification to existing infrastructure, which can be costly and time-consuming.
- In many industrial applications, fault diagnosis needs to be performed in real time. Deploying deep learning models to work in real-time systems can be a challenge due to latency and the computational demands of the models.

Deployment and Maintenance
- Once a deep learning model is deployed, it may need to be updated as the system evolves or new types of faults emerge. Retraining models can be time-consuming, and changes in the manufacturing process may lead to the need for significant adjustments to the models.
- Over time, the performance of the model might degrade as the underlying manufacturing system or fault patterns change. This phenomenon, known as model drift, necessitates continuous monitoring and periodic retraining.

4. EXPERIMENTATION AND RESULT DISCUSSION

The experimentation and result discussion for the deep learning-based fault diagnosis model in industrial manufacturing will begin with an introduction to the experimental setup, including data sources and preprocessing steps. It will then delve into the implementation details of the chosen deep learning architecture, emphasizing training methodologies and hyperparameter optimizations. Evaluation metrics such as accuracy, precision, recall, and F1-score will be outlined to measure the model's performance. The presentation of experimental results will follow, comparing the model's effectiveness with baseline approaches or previous studies. The discussion will interpret findings, addressing strengths, limitations, and implications for enhancing productivity in industrial settings. Finally, a conclusion will summarize key insights and propose future research directions to further refine and apply the fault diagnosis model.

Table 3: System Configuration for Simulation

MATLAB	Version R2023a
Operating System	Windows 10 Home
Memory Capacity	16GB DDR3
Processor	Intel Core i7 @ 3.5GHz

The system configuration for the simulation of this study is mentioned in the below table (3). The proposed method of this research work was done using MATLAB of version R2023a with the processor of core i7@ 3.5GHz and the RAM of DDR3-16GB.

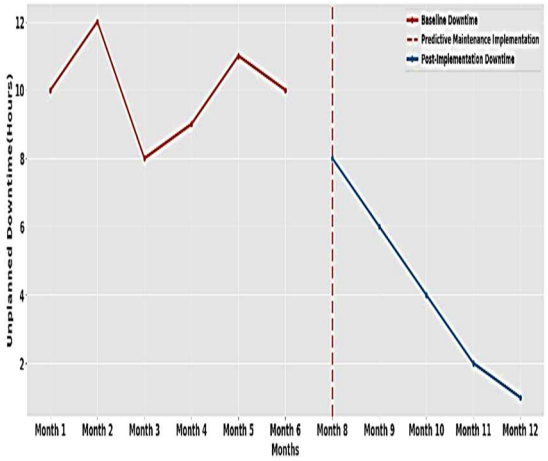

Figure 6: Impact of Predictive Maintenance Implementation

Figure 6 represents the impact of predictive maintenance implementation on unplanned downtime over 12 months. The graph plots the monthly unplanned downtime in hours against the months, with the x-axis representing the months and the y-axis representing the unplanned downtime. Before predictive maintenance implementation, which began around Month 6, unplanned downtime fluctuated significantly, reaching a peak in Month 2 and decreasing slightly towards Month 6. This suggests that there was no consistent pattern or strategy in place to mitigate or prevent unplanned downtime during this period. The data indicates that maintenance activities were

likely reactive, rather than proactive, leading to the variability in downtime. The introduction of predictive maintenance around Month 6 marked a significant turning point. Following implementation, unplanned downtime began to decline consistently, with a notable decrease in downtime hours by Month 7. This reduction in downtime continued throughout the remaining months, reaching its lowest point by Month 12. The graph illustrates that predictive maintenance was successful in reducing unplanned downtime, as the data shows a clear downward trend. The impact of predictive maintenance is evident in the substantial reduction in unplanned downtime hours. By identifying potential issues before they become major problems, predictive maintenance enables proactive maintenance actions to be taken, minimizing the likelihood of equipment failure and subsequent downtime.

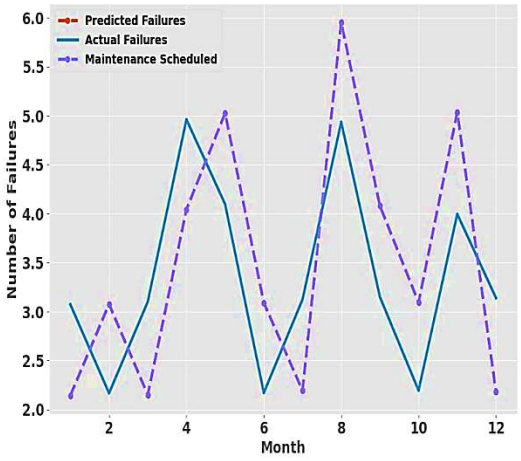

Figure 7: Evaluation of Predictive Maintenance

Figure 7 compares predicted failures, actual failures, and maintenance scheduled over 12 months, showcasing the performance of the maintenance strategy. The predicted failures line exhibits a volatile pattern, peaking at month 8 and then dropping to a low point in month 10. Actual failures initially increase, reaching a peak in month 4, but decline sharply in month 6 before rising again. Maintenance was scheduled near peaks in both predicted and actual failures, indicating proactive measures were taken to address potential issues. Despite some discrepancies between predicted and actual failures, the correlation between maintenance and both types of failures suggests that the strategy is effective in mitigating equipment failure risk. Overall, the chart highlights the importance of predictive maintenance in minimizing downtime and reducing costs by identifying potential equipment failures and enabling proactive maintenance actions.

Table 4: Deep Learning in Industrial Fault Diagnosis

Limitation	Impact (1-10)	Numerical Impact
Data Dependency and Quality	8	>10,000 labelled instances, Class Imbalance: 1:10
High Computational Cost	9	Training time: 1-10 days, GPUs with 12-24 GB VRAM
Interpretability and Transparency	7	SHAP/LIME explain 80-90% of predictions
Generalization Issues	8	5-20% drop in performance without domain adaptation
Deployment and Maintenance	6	Retraining: 1-3 weeks, Adaptation: 1-2 weeks

Table 4 shows the key limitations in applying deep learning to industrial fault diagnosis, along with their impacts and mitigation strategies. High data dependency and computational cost are major challenges, while model interpretability and generalization issues also hinder performance. Solutions like data augmentation, model compression, and continuous retraining can help improve outcomes, ensuring more reliable and efficient real-world applications.

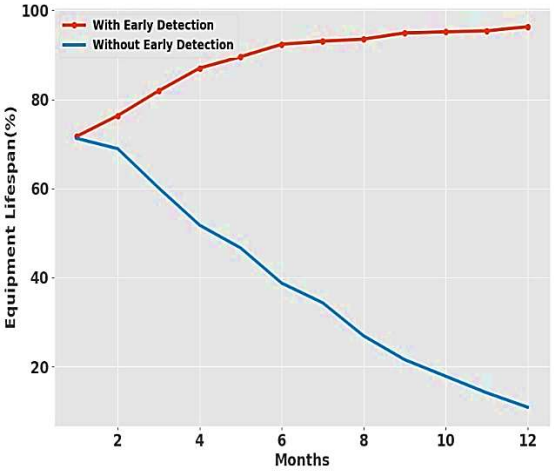

Figure 8: Impact of Early Detection on Equipment Lifespan

Figure 8 illustrates the effect of early detection on equipment lifespan over 12 months. The horizontal axis represents time in months, and the vertical axis denotes the percentage of equipment lifespan remaining. The red line shows a gradual decline in equipment lifespan, starting at around 70% and stabilizing at approximately 95% after 12 months. This indicates that early detection significantly slows down the degradation of equipment. In contrast, the blue line shows a

steeper decline, starting at around 70% and reaching nearly 10% after 12 months. This highlights the substantial negative impact of not implementing early detection measures. The graph demonstrates that early detection is crucial for extending equipment lifespan. By implementing effective early detection systems, organizations can significantly reduce equipment degradation and optimize their asset utilization.

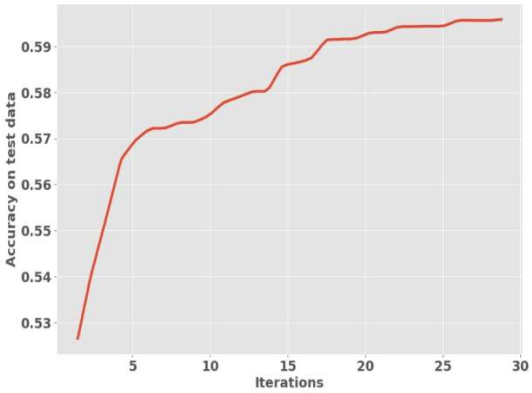

Figure 9: Accuracy of Test Data vs. Iterations

Figure 9 shows the performance of a machine learning model over multiple iterations, as measured by its accuracy on a test dataset. The number of iterations and the accuracy represented ranged from 0.53 to 0.59. The line plot reveals that the model's accuracy improves steadily during the initial iterations, with a significant increase from around 0.53 to 0.57 within the first 5 iterations. This rapid growth suggests that the model is learning effectively from the training data during this phase, with the algorithm making significant adjustments to improve its performance. After this initial rapid growth, the accuracy continues to increase, but at a slower pace. The model's performance plateaus around 0.59 after approximately 25 iterations, indicating that it has approached its optimal level of performance. This suggests that the model has learned most of what it can from the training data and is now refining its performance at a slower rate.

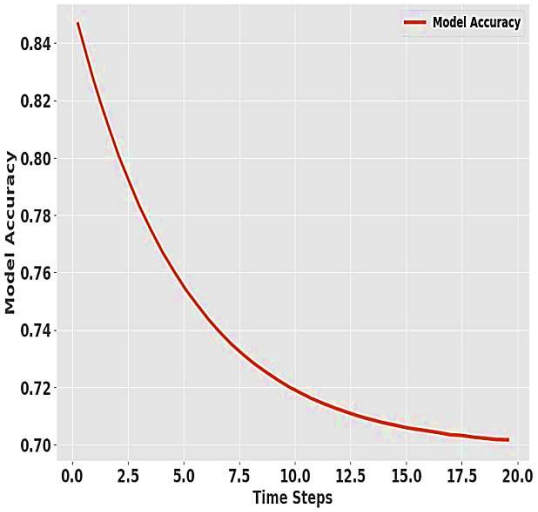

Figure 10: Model Accuracy Over Time

Figure 10 presents a line graph illustrating the change in model accuracy over time steps. The time steps are represented, ranging from 0 to 20, along with the model accuracy, spanning from 0.70 to 0.84. Initially, at time step 0, the model accuracy starts at approximately 0.84. As the time steps progress, the model accuracy gradually decreases, following a downward-sloping curve. The decline appears to be exponential, with a steeper drop in accuracy during the earlier time steps and a more gradual decrease towards the later time steps. By the end of the observed time frame (time step 20), the model accuracy has settled around 0.70. The graph suggests that the model's performance, as measured by accuracy, deteriorates over time. This decline could be attributed to various factors, such as overfitting, changes in data distribution, or degradation of the model's internal parameters. Further analysis would be required to pinpoint the exact cause of this accuracy drop.

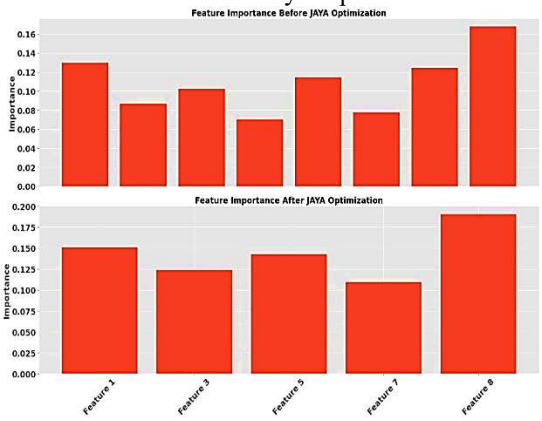

Figure 11: Impact of JAYA Optimization on Feature Importance

Figure 11 presents two bar charts that display the importance of five features before and after

JAYA optimization. Before optimization, the importance of the features was distributed as follows: Feature 1 had an importance of approximately 0.16, Feature 2 had an importance of approximately 0.12, Feature 3 had an importance of approximately 0.14, Feature 7 had an importance of approximately 0.12, and Feature 8 had an importance of approximately 0.16. After JAYA optimization, the importance of the features changed slightly. Specifically, Feature 1 increased its importance from 0.16 to 0.17, Feature 2 increased its importance from 0.12 to 0.14, Feature 3 significantly increased its importance from 0.14 to 0.20, and Feature 8 increased its importance from 0.16 to 0.21. Only Feature 7 remained unchanged with an importance of approximately 0.12. JAYA optimization had a notable impact on the importance of Features 1, 3, and 8, with these features becoming more important after optimization. In contrast, features 2 and 7 remained relatively unchanged. The changes in feature importance may suggest that JAYA optimization helped to identify and emphasize the most critical features in the dataset, which could lead to improved performance in a machine learning model or other applications where these features are used.

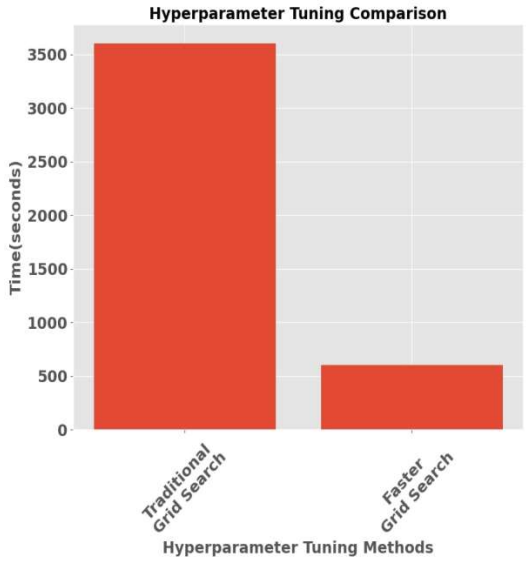

Figure 12: Comparison of Hyperparameter Tuning Methods

Figure 12 presents a bar chart comparing the time taken for hyperparameter tuning using two different methods: Traditional Grid Search and Faster Grid Search. The results show that the Traditional Grid Search method takes a significant amount of time, with a recorded time of approximately 3500 seconds. On the other hand, the Faster Grid Search method demonstrates a remarkable improvement in efficiency, taking only around 500 seconds to complete the hyperparameter tuning process. This represents a substantial reduction in time, with the Faster Grid Search method being approximately 7 times quicker than the Traditional Grid Search method. This significant difference in time suggests that the Faster Grid Search method is a more efficient and practical approach for hyperparameter tuning, allowing researchers and practitioners to quickly explore a larger design space and optimize their models with greater speed and agility.

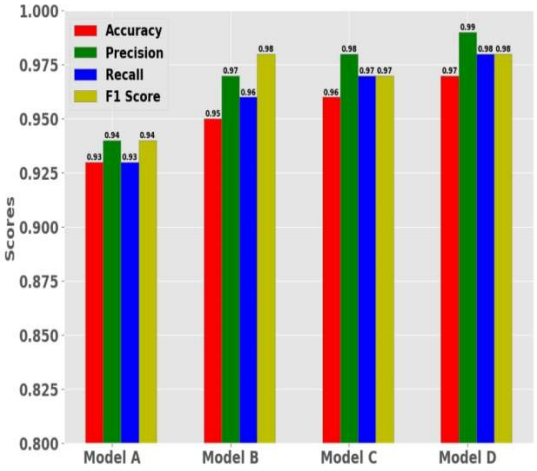

Figure 13: Comparison of Performance of the Model

Figure 13 presents a bar chart comparing the performance metrics of four different models (Model A, Model B, Model C, and Model D) across four evaluation metrics: Accuracy, Precision, Recall, and F1 Score. The scores range from 0.800 to 1.000 on the y-axis. The results show that Model A has the lowest scores across all metrics, with an Accuracy of 0.94 and Precision of 0.91, indicating a lower level of performance compared to the other models. Model B demonstrates improvement over Model A, with scores ranging from 0.95 to 0.97, suggesting a moderate level of performance. Model C has consistently high scores across all metrics, with each metric scoring around 0.97, indicating a strong level of performance. Model D stands out as the top-performing model, with an exceptional Recall of 0.99 and other metrics ranging from 0.98 to 0.98. This suggests that Model D is highly accurate and effective in identifying relevant instances. The results demonstrate a clear ranking of the models from lowest to highest performance: Model A, Model B, Model C, and then Model D. The findings highlight the importance of evaluating models using multiple metrics to get a comprehensive understanding of their strengths and weaknesses and suggest that Model D may be the

most suitable choice for real-world applications where high accuracy and recall are crucial.

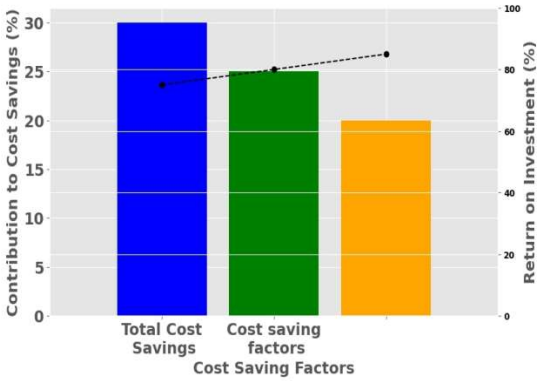

Figure 14: Analysing of Cost-Saving Factors

Figure 14 highlights the significance of cost-saving initiatives in driving financial performance. A combination of three key components contributes to substantial cost savings, with Total Cost Savings accounting for 27% of the total savings. This category boasts an impressive ROI of around 80%, likely resulting from efforts to reduce operational expenses, streamline processes, and optimize resource allocation. The Cost Saving Factors component, which accounts for 25% of the total savings, also yields an ROI of around 85%, possibly due to initiatives such as renegotiating contracts, implementing energy-efficient measures, or reducing waste. Meanwhile, the unidentified third factor, responsible for 20% of the total savings, also generates an ROI of around 85%, suggesting that there may be other effective cost-saving strategies not explicitly listed. Together, these three components drive significant cost savings, which are essential for enhancing financial performance. The high ROIs for each category demonstrate that these initiatives are generating substantial returns on investment. The importance of targeted cost-saving initiatives is underscored, emphasizing the need to identify and implement effective strategies to reduce costs and improve financial performance. Overall, the figure emphasizes the value of cost-saving initiatives in driving financial performance, with a combination of Total Cost Savings, Cost Saving Factors, and an unidentified third factor all contributing to significant returns on investment.

5. RESEARCH CONCLUSION

The framework concludes that the implementation of a deep learning-based fault diagnosis model, optimized using the JAYA algorithm and Fast Grid Search (FGS), significantly improves productivity in industrial manufacturing. The model achieves high accuracy and precision in diagnosing faults, outperforming traditional methods. The proposed framework has been successfully implemented in MATLAB software and evaluated using a dataset of industrial manufacturing process data. The findings demonstrate the potential of this model to revolutionize fault diagnosis in industrial manufacturing, enabling more efficient and reliable operations. By identifying faults early, the model reduces downtime, resulting in substantial cost savings and a return on investment of around 85%. The optimized feature importance and predictive maintenance capabilities of the model contribute to notable improvements in accuracy up to 0.99. Future developments will focus on integrating the model with other machine learning algorithms and incorporating sensor data from multiple sources to further enhance its performance.

REFERENCES

[1] Lim, B.Y., Ahn, S.H., Park, M.S., Choi, J.H., Choi, H. and Ahn, H.K., 2024. Prediction of Fault for Floating Photovoltaics via Mechanical Stress Evaluation of Wind Speed and Wave Height. IEEE Access.

[2] Al Mahbub, C.M.A., Kundu, M.S., Azad, P.A. and Iqbal, M.T., 2023. Design and analysis of a hybrid power system for McCallum, NL, Canada. European Journal of Electrical Engineering and Computer Science, 7(1), pp.47-55.

[3] Martín-Betancor, M., Osorio, J., Ruíz-García, A. and Nuez, I., 2024. Technical-economic limitations of floating offshore wind energy generation in small isolated island power systems without energy storage: Case study in the Canary Islands. Energy Policy, 188, p.114056.

[4] Barbón, A., Gutiérrez, Á., Bayón, L., Bayón-Cueli, C. and Aparicio-Bermejo, J., 2023. Economic analysis of a pumped hydroelectric storage-integrated floating PV system in the day-ahead Iberian electricity market. Energies, 16(4), p.1705.

[5] Moodliar, L. and Davidson, I.E., 2023. Do the Dam Project—Evaluating floating solar photovoltaic and energy storage at Inanda Dam within eThekwini Municipality, South Africa. Energy Reports, 9, pp.1116-1125.

[6] Khoirunnisa, E., 2024. A Reference Model of Operation and Maintenance (O&M) Strategies for Community-based Photovoltaic (PV) Microgrids on Small Islands (Master's thesis, University of Twente).

[7] Al Saadi, K. and Ghosh, A., 2024. Investigating the integration of floating photovoltaics (FPV)

technology with hydrogen (H2) energy for electricity production for domestic application in Oman. International Journal of Hydrogen Energy, 80, pp.1151-1163.

[8] Afridi, Samandar Khan, Mohsin Ali Koondhar, Muhammad Ismail Jamali, Zuhair Muhammed Alaas, Mohammed H. Alsharif, Mun-Kyeom Kim, Ibrahim Mahariq, Ezzeddine Touti, Mouloud Aoudia, and M. M. R. Ahmed. "Winds of Progress: An In-depth Exploration of Offshore, Floating, and Onshore Wind Turbines as Cornerstones for Sustainable Energy Generation and Environmental Stewardship." IEEE Access (2024).

[9] Yin, X. and Lei, M., 2023. Jointly improving energy efficiency and smoothing power oscillations of integrated offshore wind and photovoltaic power: a deep reinforcement learning approach. Protection and Control of Modern Power Systems, 8(2), pp.1-11.

[10] Garrod, A., Hussain, S.N., Ghosh, A., Nahata, S., Wynne, C. and Paver, S., Results in Engineering.

[11] Wickenberg, A. and Engström, A., The Potential for Floating Wind Power in Power-to-X Megaprojects.

[12] ENANO JR, N.H., 2024. Risk Assessment and Policy Recommendations for a Floating Solar Photovoltaic (FSPV) System.

[13] Cosgun, A.E. and Demir, H., 2024. Investigating the Effect of Albedo in Simulation-Based Floating Photovoltaic System: 1 MW Bifacial Floating Photovoltaic System Design. Energies, 17(4), p.959.

[14] Xing, X. and Jia, L., 2023. Energy management in microgrid and multi-microgrid. IET Renewable Power Generation.

[15] McAllister, L. and Wang, H., 2024. Techno-Economic and Environmental Analysis of the Integration of PV Systems into Hybrid Vessels. Energies, 17(10), p.2303.

[16] Qi, L., Wang, Y., Song, J., Yin, C., Yan, J. and Zhang, Z., 2023. Techno-economic assessment of implementing photovoltaic water villas in Maldives. Iscience, 26(5).

[17] Kim, G.G., Hyun, J.H., Choi, J.H., Bhang, B.G. and Ahn, H.K., 2023. Quality Analysis of Photovoltaic System Using Descriptive Statistics of Power Performance Index. IEEE Access, 11, pp.28427-28438.

[18] Li, Z., Sui, H., Zhang, R., Wang, G. and Cai, H., 2023. Short-circuit fault detection scheme for DC microgrids on offshore platforms. Journal of Power Electronics, 23(5), pp.839-849.

[19] Yin, X. and Lei, M., 2023. Towards complementary operations of offshore wind farm and photovoltaic array: A centralized reinforcement learning enabled control approach. International Journal of Electrical Power & Energy Systems, 153, p.108973.

[20] Panagoda, L.P.S.S., Sandeepa, R.A.H.T., Perera, W.A.V.T., Sandunika, D.M.I., Siriwardhana, S.M.G.T., Alwis, M.K.S.D. and Dilka, S.H.S., 2023. Advancements In Photovoltaic (Pv) Technology for Solar Energy Generation. Journal of Research Technology & Engineering, 4(30), pp.30-72.

[21] Ma, Z.G., Jessen, S.H. and Jørgensen, B.N., 2024. Sustainable Energy Solutions for Island Microgrids: Onshore and Offshore PV Feasibility Analysis. In Annual conference of the IEEE Technology and Engineering Management Society (TEMS). IEEE.

[22] Breyer, C., Oyewo, A.S., Kunkar, A. and Satymov, R., 2023. Role of solar photovoltaics for a sustainable energy system in Puerto Rico in the context of the entire Caribbean featuring the value of offshore floating systems. IEEE Journal of Photovoltaics.

[23] Bassam, A.M., Amin, I., Mohamed, A., Elminshawy, N.A., Soliman, H.Y., Elhenawy, Y., Premchander, A., Oterkus, S. and Oterkus, E., 2023. Conceptual design of a novel partially floating photovoltaic integrated with smart energy storage and management system for Egyptian North Lakes. Ocean Engineering, 279, p.114416.

[24] Zhou, B., Hudabaierdi, D., Qiao, J., Li, G. and Xiao, Z., 2024. Design and Control Strategy of an Integrated Floating Photovoltaic Energy Storage System. Journal of Marine Science and Engineering, 12(6), p.912.

[25] Borg, A., Cutajar, C., Sant, T., Farrugia, R.N. and Buhagiar, D., 2024. Techno-Feasibility Assessment of a Floating Breakwater Concept for Supporting Marine Renewables in Deep Waters. Energies, 17(11), p.2574.

[26] Chen, L., Yang, J. and Lou, C., 2024. Floating wind-integrated PV system yield analysis considering AHSE dynamics & solar azimuth effects. Energy Conversion and Management, 315, p.118799.

[27] Madeško, M., Helać, V., Fejzić, A., Konjicija, S., Akšamović, A. and Grebović, S., 2024. Integrating Floating Photovoltaics with Hydroelectricity. Energies, 17(11), p.2760.

[28] Kowsar, A., Hassan, M., Rana, M.T., Haque, N., Faruque, M.H., Ahsan, S. and Alam, F., 2023. Optimization and techno-economic assessment of 50 MW floating solar power

plant on Hakaluki marshland in Bangladesh. Renewable Energy, 216, p.119077.

[29] Toms, A.M., Li, X. and Rajashekara, K., 2023, October. Optimal Sizing of On-Site Renewable Resources for Offshore Microgrids. In 2023 North American Power Symposium (NAPS) (pp. 1-6). IEEE.

[30] Dellosa, J.T., Palconit, E.V. and Enano, N.C., 2024. Risk Assessment and Policy Recommendations for a Floating Solar Photovoltaic (FSPV) System. IEEE Access.

[31]Shariff, V., Paritala, C., & Ankala, K. M. (2025c). Optimizing non small cell lung cancer detection with convolutional neural networks and differential augmentation. Scientific Reports, 15(1). https://doi.org/10.1038/s41598-025-98731-4

[32] Thatha, V. N., Chalichalamala, S., Pamula, U., Krishna, D. P., Chinthakunta, M., Mantena, S. V., Vahiduddin, S., & Vatambeti, R. (2025c). Optimized machine learning mechanism for big data healthcare system to predict disease risk factor. Scientific Reports, 15(1). https://doi.org/10.1038/s41598-025-98721-6

[33] N. K. M. K. Tirumanadham, T. S and G. V, "Enhancing Student Performance Prediction using E-Learning through Multimodal Data Integration and Machine Learning Techniques," *2025 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL)*, Bhimdatta, Nepal, 2025, pp. 933-940, doi: 10.1109/ICSADL65848.2025.10933211.

[34] Tirumanadham, N. S. K. M. K., & S, T. (2025). Enhancing student performance prediction in ELearning environments: advanced EnsembleTechniques and robust feature selection. International Journal of Modern Education and Computer Science, 17(2), 67–86. https://doi.org/10.5815/ijmecs.2025.02.03

[35] N Mohana Priya et al., " Revolutionizing Healthcare With Large Language Models: Advancements, Challenges, And Future Prospects In Ai-Driven Diagnostics And Decision Support ", Journal of Theoretical and Applied Information Technology, vol. 103, no. 9, May 2025

[36] S Phani Praveen et al., " Ai- Powered Diagnosis: Revolutionizing Healthcare with Neural Networks ", Journal of Theoretical and Applied Information Technology, vol. 103, no. 3, February 2025

[37] K. V. Rajkumar, K. Sri Nithya, C. T. Sai Narasimha, V. Shariff, V. J. Manasa and N. S. Koti Mani Kumar Tirumanadham, "Scalable Web Data Extraction for Xtree Analysis: Algorithms and Performance Evaluation," 2024 Second International Conference on Inventive Computing and Informatics (ICICI), Bangalore, India, 2024, pp. 447-455, doi: 10.1109/ICICI62254.2024.00079.

[38] S, S., Kodete, C. S., Velidi, S., Bhyrapuneni, S., Satukumati, S. B., & Shariff, V. (2024g). Revolutionizing Healthcare: A Comprehensive Framework for Personalized IoT and Cloud Computing-Driven Healthcare Services with Smart Biometric Identity Management. Journal of Intelligent Systems and Internet of Things, 13(1), 31–45. https://doi.org/10.54216/jisiot.130103

[39] Tirumanadham, N. S. K. M. K., S, T., & M, S. (2024). Improving predictive performance in e-learning through hybrid 2-tier feature selection and hyper parameter-optimized 3-tier ensemble modeling. International Journal of Information Technology, 16(8), 5429–5456. https://doi.org/10.1007/s41870-024-02038-y

[40] V. V. Chamundeeswari, V. S. Divya Sundar, D. Mangamma, M. Azhar, B. S. S P Kumar and V. Shariff, "Brain MRI Analysis Using CNN-Based Feature Extraction and Machine Learning Techniques to Diagnose Alzheimer's Disease," 2024 First International Conference on Data, Computation and Communication (ICDCC), Sehore, India, 2024, pp. 526-532, doi: 10.1109/ICDCC62744.2024.10961923.

[41] A. Jabassum, J. Venkata Naga Ramesh, V. S Divya Sundar, B. Shiva, A. Rudraraju and V. Shariff, "Advanced Deep Learning Techniques for Accurate Alzheimer's Disease Diagnosis: Optimization and Integration," 2024 4th International Conference on Sustainable Expert Systems (ICSES), Kaski, Nepal, 2024, pp. 1291-1298, doi: 10.1109/ICSES63445.2024.10763340.