Graph-variational convolutional autoencoder-based fault detection and diagnosis for photovoltaic arrays.

ARIFEEN, M., PETROVSKI, A., HASAN, M.J., NOMAN, K., NAVID, W.U. and HARUNA, A.

2024

machines

MDPI

Article

Graph-Variational Convolutional Autoencoder-Based Fault Detection and Diagnosis for Photovoltaic Arrays

Murshedul Arifeen [1,*], Andrei Petrovski [2], Md Junayed Hasan [3], Khandaker Noman [4,*], Wasib Ul Navid [4] and Auwal Haruna [5]

[1] School of Computing, Engineering and Technology, Robert Gordon University, Aberdeen AB10 7AQ, UK
[2] Faculty of Secure Information Technologies, ITMO University, St. Petersburg 197101, Russia; a.petrovski@rgu.ac.uk
[3] Dataxense, Aberdeen AB11 5RG, UK; j.hasan@dataxense.co.uk
[4] School of Civil Aviation, Northwestern Polytechnical University, Xi'an 710129, China; navid.wasib32@mail.nwpu.edu.cn
[5] School of Aeronautics, Northwestern Polytechnical University, Xi'an 710129, China; mag@nwpu.edu.cn
* Correspondence: m.arifeen@rgu.ac.uk (M.A.); khandakernoman93@nwpu.edu.cn (K.N.)

Abstract: Solar energy is a critical renewable energy source, with solar arrays or photovoltaic systems widely used to convert solar energy into electrical energy. However, solar array systems can develop faults and may exhibit poor performance. Diagnosing and resolving faults within these systems promptly is crucial to ensure reliability and efficiency in energy generation. Autoencoders and their variants have gained popularity in recent studies for detecting and diagnosing faults in solar arrays. However, traditional autoencoder models often struggle to capture the spatial and temporal relationships present in photovoltaic sensor data. This paper introduces a deep learning model that combines a graph convolutional network with a variational autoencoder to diagnose faults in solar arrays. The graph convolutional network effectively learns from spatial and temporal sensor data, significantly improving fault detection performance. We evaluated the proposed deep learning model on a recently published solar array dataset for an integrated power probability table mode. The experimental results show that the model achieves a fault detection rate exceeding 95% and outperforms the conventional autoencoder models. We also identified faulty components by analyzing the model's reconstruction error for each feature, and we validated the analysis through the Kolmogorov–Smirnov test and noise injection techniques.

Keywords: solar array; photovoltaic array; fault detection; fault diagnosis; graph convolutional network; variational autoencoder

check for updates

Citation: Arifeen, M.; Petrovski, A.; Hasan, M.J.; Noman, K.; Navid, W.U.; Haruna, A. Graph-Variational Convolutional Autoencoder-Based Fault Detection and Diagnosis for Photovoltaic Arrays. *Machines* **2024**, *12*, 894. https://doi.org/10.3390/machines12120894

Academic Editor: Davide Astolfi

Received: 30 October 2024
Revised: 27 November 2024
Accepted: 4 December 2024
Published: 6 December 2024

1. Introduction

Solar arrays, or Photovoltaic (PV) grids, are essential for sustainable energy production, addressing environmental and economic concerns [1]. They harness sunlight to generate electricity, providing a renewable and clean energy source that reduces reliance on fossil fuels and minimizes greenhouse gas emissions [2]. Solar arrays contribute to energy security by diversifying the energy supply, and they can be deployed at various scales, from residential rooftops to large solar farms. Additionally, they promote energy independence and have the potential to reduce electricity costs over time significantly [3]. Their implementation supports the global shift to a low-carbon economy and helps alleviate the impacts of climate change [4]. However, issues with solar arrays or PV grids can significantly impact their efficiency, safety, and overall performance. Common problems include the following: shading, which can reduce the output of individual panels and the entire system; micro-cracks or hotspots, which can degrade panels over time; and connection problems, such as loose or corroded wiring, which can disrupt the flow of electricity [5]. Inverter faults, which convert the direct current (DC) output of solar panels to alternating current (AC) for grid use, can also lead to substantial energy losses [6]. Thus, monitoring and diagnostic systems are crucial for the early detection and resolution of these faults to maintain optimal operation and extend the lifespan of solar installations. Regular maintenance and inspection are also essential to ensure reliability, prevent potential hazards, and maximize energy production from PV grids.

There are two different methods for detecting faults within a Solar or PV system. These are first-principal or model-driven and data-driven methods [7,8]. The former involves the establishment of a physical model or mathematical model, while the latter relies on control system's sensor/IoT data [8]. Data-driven models can be implemented using either shallow learning techniques or deep learning (DL) techniques, both of which provide reliable and efficient solutions for fault detection when compared to model-driven approaches [9,10]. DL is particularly effective at analyzing complex supervisory control and data acquisition (SCADA) data. Within the scope of DL, there are two primary approaches to fault detection: forecasting-based techniques and reconstruction-based techniques [11]. This paper, however, exclusively centers on reconstruction-based techniques, providing a detailed and comprehensive understanding of this specific approach.

Models based on an autoencoder (AE) [12–14], including long short-term memory (LSTM) and convolutional neural network (CNN) models, as well as variational, sparse, and denoising types, are commonly used models for fault detection in different applications [15,16]. For example, stacked AE-based models are used for solar array fault diagnosis in [17,18]. However, the stacked AE-based models have multiple hidden layers, which can lead to overfitting. Seghiour et al. [19] combined an AE and a feed-forward neural network to diagnose and classify faults in solar PV arrays. Ayobami et al. [20] proposed Variational AE (VAE) and spread spectrum techniques to detect, isolate, and characterize faults in PV systems. LSTM AE-based models with wavelet packet-transformed PV signals are proposed in [21,22]. Similarly, AE and its variants are applied to fault diagnosis for solar arrays in [23–25]. However, the AE-based models often struggle to capture SCADA data's spatial and temporal relationships [26,27].

To gain comprehensive insights, it is essential to consider these relationships together [27]. Graph neural networks (GNNs) excel at learning spatiotemporal data due to their unique features, such as permutation invariance and local connectivity [11]. Researchers have employed GNNs, particularly graph convolutional networks (GCNs), to develop spatiotemporal autoencoders for fault identification, achieving better results than traditional AEs across various process monitoring applications [26,28–31]. However, GCN-enabled VAE is not adequately investigated for fault diagnosis in industrial applications. VAE offer several advantages over conventional AE models. For example, VAE incorporate a regularization term (the Kullback–Leibler divergence) in their loss function, encouraging the latent space to conform to a standard normal distribution [32,33]. This helps prevent overfitting and ensures a more uniform latent space [32]. The latent representations learned through VAE are often more informative and disentangled than those obtained via traditional AE. This can improve performance in downstream tasks such as clustering, classification, or anomaly detection.

This study uses the GCN model to develop a VAE to detect faults in PV microgrids within renewable energy applications. The GCN's capacity to learn spatiotemporal features renders it exceptionally well suited for SCADA data's intricate nonlinear spatiotemporal modeling. Moreover, skip connections are added between the models hidden layers to mitigate the loss of information for the encoding–decoding process. Figure 1 shows the proposed fault detection and diagnosis framework for the GCN-based VAE model. The major contributions of this paper are outlined as follows.

- A VAE model based on GCN layers is developed to detect faults in solar array applications. The GCN VAE model effectively learns the spatiotemporal characteristics of SCADA sensor data, outperforming conventional AE models.

- We use the convolutional filter and boxplot algorithm to enhance the data quality. The convolutional filter reduces noise, while the boxplot removes outliers from the noisy sensor data.
- The performance of the GCN-VAE model is evaluated based on a solar array dataset and compared with the conventional AE models
- The false alarm rate (FAR) and fault detection rate (FDR) show that the proposed GCN-based VAE outperforms conventional AE models.
- Also, individual feature reconstruction errors are analyzed for diagnosing the root cause of a fault.

Section 2 explains the fault detection and diagnosis process using the GCN-based VAE model. We first provide a brief overview of the GCN and then demonstrate how it is integrated with the VAE to create a DL model for fault detection. Finally, we discuss the fault detection indicators and the diagnosis process. Section 3 presents the experimental procedure, including data processing, model training, testing, and results. Finally, Section 4 concludes the paper.

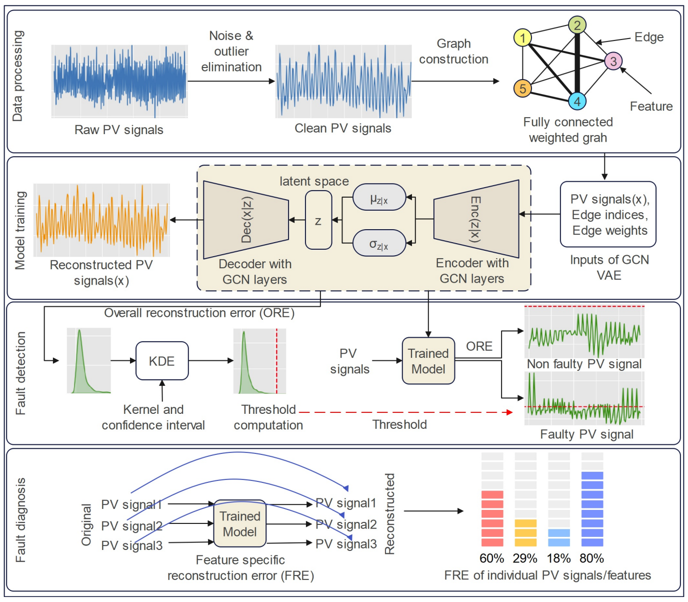


Figure 1. Overall framework of the fault detection and diagnosis model for solar/PV array grids. The raw solar array signals are first de-noised and cleaned using the data processing module. Then, the graph attributes are generated from the clean PV signals. The clean dataset and the graph attributes are used to train the proposed model. Finally, a fault detection indicator is constructed, and the diagnosis process is illustrated.

2. GCN VAE-Based Fault Detection and Diagnosis

This section discusses GCN and GCN-based VAE for detecting faults in PV or solar array systems. Then, it is shown how the reconstruction error of the GCN VAE can be used to identify faulty components (features) within the larger system.

2.1. Graph Convolutional Network

We can define a graph as $G = (V, E, A)$, which contains a set of nodes, V, $|V| = N$, a set of edges, E, $|E| = M$, and an adjacency matrix, A. The graph's adjacency matrix, $A \in R^{N \times N}$, expresses the relationships of weights and edges among the graph nodes, V. Therefore, if we find an edge between node $v_i \in V$ and $v_j \in V$, then these two nodes are neighbors $(i \neq j)$, and the corresponding entry, $A(i, j)$ in A, refers to the weights of the edge. We can use various techniques, such as Euclidean similarity, a correlation matrix, or cosine similarity, to compute the weights of the edges [13]. On the other hand, the entries of the adjacency matrix, A, of an unweighted graph can be set to $(i, j) = 1$.

Graph convolution networks use spectral and spatial approaches, with spectral methods based on graph signal processing and defining the convolution operator in the spectral domain. Spectral methods involve transforming a graph signal, x, using the graph Fourier transform, $\Psi(.)$, conducting a convolution operation, and then transforming back the resulting signal using the inverse graph Fourier transform $\Psi^{-1}(.)$ [34,35]. Mathematically,

$$\Psi^{-1}(\Psi(x)) = Ux \tag{1}$$

where U is the normalized graph laplacian matrix $L = I_N - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$; D, A are the degree and adjacency matrix, respectively. Then, according to the convolution operation,

$$\begin{aligned} v \otimes x &= \Psi^{-1}(\Psi(v) \otimes \Psi(x)) \\ &= U(U^T v \otimes U^T x) \end{aligned} \tag{2}$$

where $U^T v$ is the spectral domain filter. The principal function of the spectral method can be defined by simplifying the filter through a learnable diagonal matrix v_w. Defferard et al. [36] approximated the v_w in their proposed Chebynet model with Chebyshev polynomials $Y_k(x)$ up to K^{th} order. The ChebNet model operation can be described as

$$v_w \otimes x = \sum_{k=0}^{K} w_k Y_k(\tilde{L})x; \tilde{L} = \frac{2}{\lambda_{max}}L - I_N \tag{3}$$

where λ_{max} is the largest eigenvalue of L. However, in this work, GCN proposed by Kipf [37] is used in constructing the AE model. Kipf simplified the convolution operation of ChebNet by setting $K = 1$ and $\lambda_{max} = 2$ to overcome the overfitting problem. Using these assumptions, the operation of GCN can be defined as

$$\begin{aligned} v_w \otimes x &= w_0 x + w_1(L - I_N)x \\ &= w_0 x - w_1 D^{-\frac{1}{2}}AD^{-\frac{1}{2}}x \\ &= w(I_N + D^{-\frac{1}{2}}AD^{-\frac{1}{2}})x; (w_0 = -w_1) \end{aligned} \tag{4}$$

Kipf further introduced the renormalization trick to overcome the exploding/vanishing gradient problem by updating $\tilde{A} = A + I_N$ and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. Finally, the convolution operation of GCN can be defined as

$$H = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}XW \tag{5}$$

where X is the input data matrix, W is the model parameters, and H is the resulting convolution matrix. The GCN based on the spectral method is used in this paper to construct the fault diagnosis model.

2.2. Graph Convolutional Network-Based Variational Autoencoder

This work develops a graph convolutional variational autoencoder (GCVAE) model that combines GCN and VAE to learn spatiotemporal time series data representations. First, the time series data are represented as a graph by defining nodes, computing edges, and edge weights. Then, the GCN-based variational encoder is used to encode the graph-structured spatiotemporal time series data. Each feature in the time series data is considered a node (V) in the graph, and a fully connected weighted graph ($A_{ij} \neq 1$) is assumed, meaning that each feature (V) is connected to all the other features. The edge weights (e_w) are calculated based on the cosine similarity metric. The model consists of a multi-layer GCN-based variational encoder and decoder architecture. The encoder $f_{\alpha_{gae}}(z|x, e_i, e_W)$ generates the latent representation of the time series data by taking into account the edges (e_i) and edge weights (e_w). On the other hand, the decoder $g_{\beta_{gae}}(x^{'}|z, e_i, e_W)$ generates reconstructed data $x^{'}$ based on the latent representation z along with the edges (e_i), and the edge weights (e_w). The output of the l_e^{th} encoder layer can be written as a function of the previous layer output and the adjacency matrix A, the edge indices (e_i), and the edge weights (e_w) as

$$\begin{aligned} H^{l_e+1} &= f_{\alpha_{gae}}(H^{l_e}, A) \\ &= \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{l_e}, W^{l_e}) \end{aligned} \tag{6}$$

where σ, W_e^l, and H_e^l are the activation function, weight matrix, and the encoded representation of layer l. On the contrary, the l_d^{th} layer of the decoder can be mathematically represented as

$$\begin{aligned} H^{l_d+1} &= g_{\beta_{gae}}(H^{l_d}, A) \\ &= \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{l_d}, W^{l_d}) \end{aligned} \tag{7}$$

Similarly to the VAE, the GCVAE also optimizes the encoder and decoder parameters α_{ae} and β_{ae} by minimizing the below loss function through the backpropagation algorithm.

$$\zeta_{vae}(\alpha_{vae}, \beta_{vae}) = \sum_{t=1}^{T}\Big[D_{KL}(f_{\alpha_{vae}}(z_t|x_t)||p_{\beta_{vae}}(z_t)) + (x_t - x_t^{'})^2\Big] \tag{8}$$

Since the GCN model is capable of learning the spatiotemporal relation among the features, the GCN-based VAE can learn the spatiotemporal data more effectively than other conventional AE models.

2.3. Graph-Convolutional Variational Autoencoder for Fault Detection

Microgrid operation produces significant data, which SCADA systems gather and store. Nevertheless, most of these data arise from normal operational circumstances, and faulty data are infrequent and occasionally inaccessible. This section endeavors to develop a standard reference model utilizing this normal data (fault-free data). Through this reference model, testing data can be assessed to recognize and identify potential faults. Therefore, in this section, a GCN VAE-based fault detection model is presented that learns the non-linear spatiotemporal SCADA data from the normal operation of the PV panel. The model learns the latent representation of the spatiotemporal SCADA data by reconstructing it, which makes it more robust than other AE-based models.

Unlike other AE-based detection methods, a GCN VAE detection model cannot be trained directly on multivariate sensor data to comprehensively understand their intrinsic representation and the spatial and temporal connections between them. Multivariate time-series sensor data needs to be converted into graph-structure data. The previous subsection explains the process of transforming the time series data into graph-structured data. Let $X_n = \{x^i\}_{i=1}^N \in \mathbb{R}^{M\times N}$ be the normal multivariate SCADA data from the PV panel, where M is the number of features (sensors), and N is the number of data samples. The dataset X is split into training, validation, and testing data as $X_n = \{X_t, X_v, X_d\}$. Then, for training, the graph attributes are generated from X_t for the adjacency matrix A, i.e., the edges and weights of each edge. A fully connected graph is considered for this work, meaning that all the nodes are connected with each other, and finally, the model computes the edge weights using the cosine similarity metric. The cosine similarity between two nodes or features, M_1 and M_2, can be defined as follows:

$$CS(M_1, M_2) = \frac{M_1 \cdot M_2}{||M_1||||M_2||} \tag{9}$$

where $M_1 \cdot M_2 = \sum_i^N m_1^i m_2^i$, and $||M_1|| = \sqrt{M_1 \cdot M_2}$

After the graph attributes are generated, the GCN VAE model is trained by passing the data, X_t, and the adjacency matrix, A, to minimize the loss function defined in Equation (8) for learning the optimal encoder and decoder parameters. Only the normal train data split, X_t, is used to train the model. Upon the completion of training, the model excels in precisely reconstructing normal data, while faulty samples from the faulty data, X_f, produce noticeable spikes in reconstruction error. Therefore, a reconstruction error-based fault detection indicator can detect possible faults in the testing data [38].

2.4. Fault Detection Indicator Construction

Squared prediction error (SPE) residuals, $E \in \mathbb{R}^{N \times 1}$, are used here to represent the reconstruction error for all the samples, $x \in X$. The SPE is computed on the validation data, X_v, to determine the fault indicator threshold as follows:

$$E = (X_v - \tilde{X_v})^2; X_v \in \mathbb{R}^{M \times N} \tag{10}$$

Then, the threshold or control limit of $E \in R^{N \times 1}$ is computed using the statistical properties of its distributions. However, in this case, the distribution is unknown in advance; therefore, it is needed to estimate the probability density functions of E through a non-parametric kernel density estimation (KDE) technique. KDE is a widely accepted and proven methodology for estimating the probability density function (PDF), and it has demonstrated remarkable success in process monitoring and fault detection. We used the KDE method to compute the threshold. The estimated PDF of error points, say $r_i \in E$, $i = 1, 2, ..., N$ at point r, can be mathematically defined as [13]

$$p(r) = \frac{1}{N\Omega} \sum_{i=1}^{N} \kappa\left(\frac{r - r_i}{\Omega}\right) \tag{11}$$

where $\kappa(.)$ is the kernel function, and Ω is the bandwidth. Here, the Gaussian kernel function $\kappa(u) = \frac{e^{-\frac{u^2}{2}}}{\sqrt{2\pi}}$ and the Silverman bandwidth are used. Finally, we can compute the threshold (γ) of the error vector E using the estimated PDF for a given confidence value, α, by solving the following equation [38]:

$$P(r < \gamma) = \int_{-\infty}^{\gamma} p(r)\gamma(r) = \alpha \tag{12}$$

In the online monitoring phase, any data sample that exceeds the threshold (γ) is classified as a faulty sample.

2.5. Fault-Diagnosis Process

Fault diagnosis is a process that helps accurately identify the root cause of problems, enabling corrective action. It is also known as "fault isolation", which differentiates it from fault detection. In this work, diagnosis aims to identify the sensor responsible for the system failure. To achieve this, the reconstruction error is analyzed for individual features that contribute more than the others to each type of fault in the overall reconstruction error.

3. Experiments and Results

To conduct the experiment for evaluating the proposed fault detection and diagnosis model, a solar/PV array microgrid dataset from a recent article [6] was considered. The overview of the proposed framework is presented in Figure 1. We first de-noised and removed outliers from the raw data. Then, from the cleaned data, we generated the graph attributes and trained the model. The KDE is used to determine the threshold from the trained model's reconstruction error. Finally, the individual reconstruction error for each feature was used to identify the faulty components in the diagnosis phase.

3.1. Dataset Explanation

In a laboratory environment, the authors of [6] implemented a PV microgrid system to collect both normal and faulty operational data from Integrated Power Probability

Table (IPPT) mode. The data encompass various sensor signals, including time, PV array current, voltage, DC voltage, three-phase current measurements, three-phase voltage measurements, current magnitude, current frequency, voltage magnitude, and voltage frequency. Deliberate faults were introduced, such as inverter faults (Fault 1), feedback sensor faults (Fault 2), grid anomalies (Fault 3), PV array mismatches (Fault 4 (10% to 20%), and Fault 5 (15%)), as well as controller faults (Fault 6) and converter faults (Fault 7). Table 1 shows the statistical summary of the dataset.

Table 1. Summary of the IPPT mode PV panel dataset.

Features	Mean	std	Min	25%	50%	75%	Max
Ipv	1.502544	0.078136	0.604523	1.448395	1.498535	1.554352	2.06427
Vpv	101.1912	0.317784	96.45386	101.0437	101.1902	101.3489	103.6133
Vdc	142.9183	0.712972	127.4414	142.3828	142.9688	143.2617	149.1211
ia	−0.01325	0.348352	−0.62525	−0.3634	−0.01428	0.341552	0.576537
ib	0.002302	0.350489	−0.61096	−0.34912	0.013428	0.349121	0.584106
ic	−0.01702	0.335469	−0.59668	−0.34827	−0.03943	0.323115	0.564814
va	0.750142	109.7158	−160.818	−108.888	0.855865	110.4187	159.866
vb	0.665057	109.7159	−160.818	−108.972	0.687103	110.3283	159.866
vc	0.633947	109.7023	−158.918	−109.047	0.622813	110.26	159.2312
Iabc	0.486386	0.029416	0.206126	0.467845	0.482093	0.498496	1
If	50.01225	0.187179	48.96146	49.92325	50.00496	50.08717	51.01998
Vabc	154.9138	5.767691	1	154.8725	155.1658	155.3576	155.7785
Vf	50.00029	0.011106	49.88067	49.9966	50.00013	50.00352	50.19678

Figure 2 demonstrates the use of the Kolmogorov–Smirnov (KS) test in analyzing the IPPT dataset of fault type 1 and the standard set. The KS test is employed here to understand the statistical difference (cumulative distribution function (CDF)) between each feature from the standard and faulty part of the dataset. The KS test is a non-parametric statistical test used to compare a sample distribution with a reference probability distribution or to compare two sample distributions. It evaluates the null hypothesis that the samples are drawn from the same distribution without assuming any specific form for the distribution [39]. The test assesses the maximum difference between the empirical distribution function of the sample and the cumulative distribution function of the reference distribution or between the empirical distribution functions of two samples. A significant KS test result indicates that the distributions differ significantly. It is a valuable tool for testing assumptions about data distributions and detecting deviations from expected patterns.

Based on the KS test, the Vpv feature displays considerably greater differences between standard and faulty parts of the dataset compared to other features. Table 2 presents the numerical results of the KS test for all fault types. It is evident from Table 2 that the Vpv feature is responsible for making the solar array defective across all fault types (marked as bold).

Table 2. KS test for quantifying difference in CDFs of non-faulty and faulty features.

Features	Fault 1	Fault 2	Fault 3	Fault 4	Fault 5	Fault 6	Fault 7
Ipv	0.79	0.55	0.35	1.00	0.68	0.22	0.48
Vpv	**1.00**	**0.92**	**0.99**	**1.00**	**0.97**	0.90	**1.00**
Vdc	0.55	0.34	0.68	0.70	**0.67**	**0.84**	0.44
Ia	0.07	0.14	0.05	0.07	0.34	0.07	0.06
Ib	0.14	0.04	0.07	0.09	0.37	0.07	0.02
Ic	0.10	0.28	0.06	0.10	0.29	0.05	0.06
Va	0.06	0.01	0.01	0.02	0.09	0.01	0.02
Vb	0.01	0.01	0.01	0.01	0.11	0.01	0.02
Vc	0.05	0.01	0.01	0.01	0.05	0.01	0.02

Table 2. *Cont.*

Features	Fault 1	Fault 2	Fault 3	Fault 4	Fault 5	Fault 6	Fault 7
Iabc	0.66	0.55	0.19	0.52	0.67	0.12	0.26
If	0.05	0.43	0.03	0.03	0.12	0.03	0.04
Vabc	0.09	0.17	0.30	0.21	0.13	0.53	0.52
Vf	0.04	0.38	0.01	0.03	0.09	0.08	0.07

The boldfaced numbers represent a significant difference in the CDF for the Vpv feature than other features.

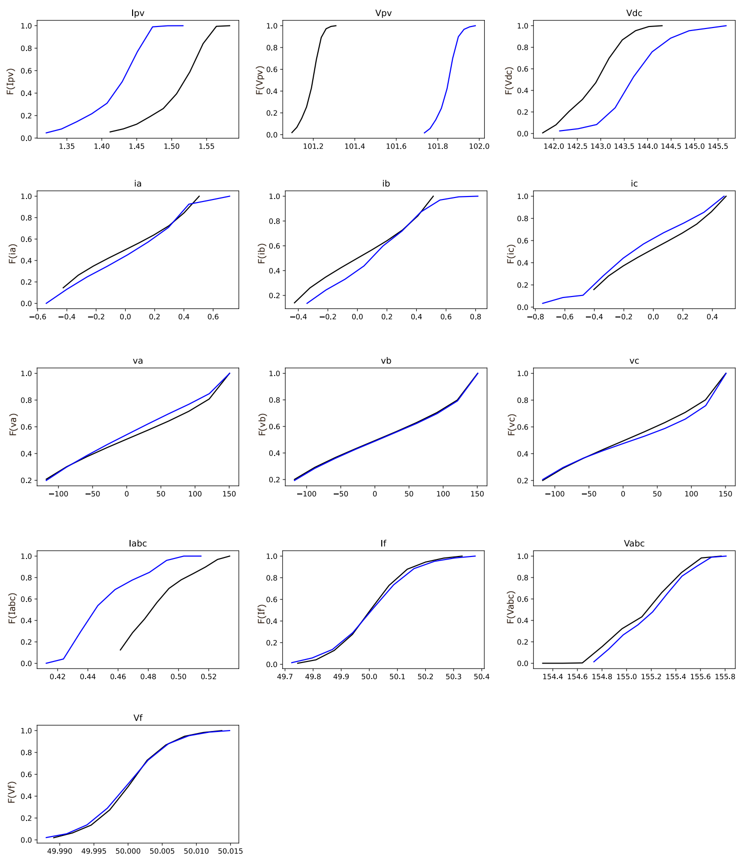


Figure 2. Cumulative density function (CDF) plot for normal data (fault-free) and faulty data (Fault 1). The black curve shows the CDF of the features from the normal data, while the blue curve shows the CDF of the features from the Fault 1.

3.2. Data Preprocessing

The proposed fault detection and diagnosis framework's data preprocessing module comprises a convolution filter-based noise-reduction method and a boxplot-based outlier-removal technique.

3.2.1. Convolution Filter-Based Noise Reduction

In this work, a convolution filter [40,41] is applied to reduce the effect of noise from each feature. Convolutional filter-based signal smoothing is a method that aims to reduce noise and highlight essential features within a signal. This technique works by sliding a filter (kernel) across the targeted signal and conducting element-wise multiplications and summations. The filter coefficients can be of different types, such as Gaussian or moving-average filters, which determine the extent of the smoothing effect. Convolutional filters can reduce the impact of high-frequency noise while maintaining the low-frequency components. This filtering technique has widespread usage for enhancing data quality in domains such as computer vision, signal processing, time-series sensor data analysis, etc. Therefore, it improves the interpretation and analysis of the underlying dataset.

Moreover, the adaptability and efficiency of this technique have made it suitable for real-time applications with large-size datasets.

3.2.2. Boxplot for Outlier Removing

A boxplot is a straightforward and effective method for identifying and eliminating outliers from a dataset [42]. A boxplot provides a five-number summary of a data set [43]: the minimum value (Min), the maximum value (Max), the first quartile ($Q1$), the median value (the second quartile ($Q2$)), and the third quartile ($Q3$). This method produces the $(Max - Min)$ range and the interquartile range $(IQR(Q3 - Q1))$; using these ranges, a boundary can be established to distinguish outliers from standard data samples. Mathematically, we can say that any data points above $(Q3 + 1.5 \times IQR)$ and below $(Q1 - 1.5 \times IQR)$ are outliers, and then we can remove those data points from the original data series [13].

Therefore, the noisy components from each feature of the dataset are removed using the convolution filter algorithm with different window sizes. The resulting features were then subjected to the boxplot method to remove any outliers from training, validation, and testing data. Figure 3 shows the effect of applying the convolutional filter and boxplot on the Ipv feature of the dataset.

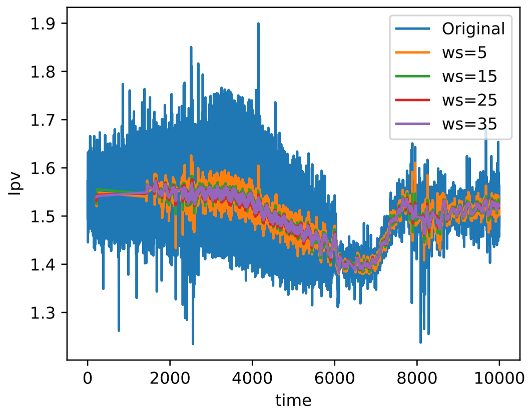


Figure 3. The effect of applying the convolution filter for different window sizes (ws = 5, 15, 25, 35) and the boxplot to reduce noise from the raw Ipv signal. We can see that increasing the window size reduces the noise from the original Ipv signal.

3.3. Model Training and Testing

The primary step for training the proposed DL model is to normalize the dataset using Python's StandardScaler function (https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html accessed 21 August 2024). The StandardScaler function ($z = \frac{x-\mu}{\sigma}$, where μ is the mean and σ is the standard deviation) is first applied to the training data (fault-free dataset) to normalize it; then, the mean and standard deviation of the fault-free training data is used to normalize the validation data (fault-free) and test data (faulty data). After that, the edge index and weights of the graph are generated based on the cosine similarity metric. Finally, the processed training data, its edge indices, and edge weights are transferred to the models for training. The first 7000 data points of the fault-free training data are utilized to train the model, and an additional 2000 data points of the fault-free training data are marked for validation. Subsequently, the trained model was tested using 1000 data points for each type of fault. The hyperparameters were chosen through a trial and error approach, wherein the loss function was carefully monitored to identify the parameters that yielded the lowest reconstruction error. The lowest reconstruction error was found for the Adam Optimiser, which has a learning rate of 0.001 and 3000 epochs. Also, experiments were conducted for two different window sizes of the convolutional filter and three different hidden layer dimensions.

3.4. Threshold Selection

Once the model has been trained, the validation dataset is evaluated and used to compute the SPE error [13]. Next, the threshold is calculated from the SPE using the Gaussian KDE estimation method, following the Equation (12) for a confidence value of $\alpha = 0.99$. In the test set, any data points exceeding the threshold are classified as faulty, while those below the threshold are considered non-fault samples.

3.5. Results

The fault detection rate and false alarm rate are used to evaluate the model's performance on fault detection and false alarm suppression. These critical metrics are instrumental in assessing the efficacy of fault detection systems. The FDR indicates the proportion of faults correctly identified through the system, thereby illustrating its ability to recognize genuine issues. A DL model with high FDR ensures the reliability, safety, and timely detection of most faults in the underlying cyber-physical systems. On the contrary, the FAR measures the percentage of non-faulty conditions inaccurately predicted as faults. These evaluation metrics reflect the DL model's precision in differentiating the normal states from the faulty states. A high FAR can result in unwanted human interventions, increased maintenance costs, and reduced confidence in the system's accuracy. Striking a balance between FDR and FAR is essential; an optimal fault detection system maximizes FDR while minimizing FAR, enabling efficient and dependable operation with minimal false alarms. The FDR and FAR are defined mathematically as follows [13]

$$FDR = \frac{\lambda}{\Lambda} \tag{13}$$

Here, λ indicates the number of detected faults, while Λ refers to the total count of faulty samples.

$$FAR = \frac{\psi}{\Psi} \tag{14}$$

Here, ψ indicates the number of detected normal samples, while Ψ refers to the total count of normal data samples.

Table 3 outlines the FDR about different fault types for the GCN VAE when the number of nodes in the bottleneck layer (or latent dimension) is varied (for 1, 2, and 3 nodes in the bottleneck layer). This table indicates that the GCN VAE demonstrates optimal fault detection performance when two nodes/neurons are in the bottleneck layer, as evidenced by a decline in performance for Fault 3 and Fault 5 when the latent dimensions were 1 and 3. However, the model's performance remains relatively consistent across all three bottleneck layers for other fault types. Furthermore, the table also presents the FDR after introducing skip connections between the layers of the GCN VAE. Without skip connections, the FDR decreases to 9 and 0 for Fault 3 and Fault 6, respectively, and consequently fails to detect Fault 5 correctly (32% FDR for fault type 5). Conversely, with skip connections, the model can detect all fault types with an FDR exceeding 95%.

In this work, the FDR performance of the GCN VAE is evaluated alongside other conventional autoencoder-based diagnosis models such as Feedforward AE, LSTM-AE, and LSTM VAE. Table 4 shows this comparison results. We conducted the comparison using two different window sizes for the convolutional filter. For the convolution filter's window size of 25, the GCN VAE achieved a similar FDR to AE, LSTM AE, and LSTM VAE for Fault 1, Fault 2, Fault 4, and Fault 7. However, the GCN VAE outperformed the other models in detecting Fault 3 and Fault 6. It is worth noting that the window size of 25 for the convolution filter couldn't effectively eliminate noise from the signals. Consequently, none of the models, including GCN VAE, could achieve significant performance for Fault 5. In the context of a window size of 35, the GCN VAE exhibited superior overall performance compared to other baseline models. Specifically, the GCN VAE achieved a fault detection rate of over 95% across all types of faults. For Fault 1, Fault 2, Fault 3, and Fault 4, all models demonstrated similar detection performance, with the exception of the LSTM VAE, which did not perform well for Fault 3. Additionally, for Fault 5, the GCN VAE outperformed all other models with a 95% detection rate. For Fault 6 and Fault 7, again, the GCN VAE outperformed other models with 99% FDR. Figure 4 illustrates the comparison results of Tables 3 and 4 graphically.

Table 3. Fault detection rate (%) of GCN VAE for different hidden layer sizes.

Fault Types	hd 3	hd 2	hd 1	With Skip	Without Skip
Fault 1	99.98	99.98	99.98	99.98	99.98
Fault 2	99.97	99.97	99.97	99.97	98.5
Fault 3	95.4	98.00	94.9	98.00	9
Fault 4	99.99	99.99	99.99	99.99	99.99
Fault 5	94.9	95.00	56.5	95.00	32.4
Fault 6	99.9	99.50	99.6	99.50	0
Fault 7	99.98	99.98	99.98	99.98	99.5

Table 4. Comparison of fault detection rate (%) of GCN VAE and baseline models.

Faults	Window Size 35				Window Size 25			
	GCNVAE	AE	LSTMAE	LSTMVAE	GCNVAE	AE	LSTMAE	LSTMVAE
Fault 1	**99.98**	99.97	99.98	99.98	99.98	99.97	99.98	99.97
Fault 2	**99.97**	99.97	99.97	99.96	99.98	99.96	99.97	99.97
Fault 3	**98.00**	97.40	94.36	81.08	**94.30**	85.5	76.91	74.78
Fault 4	**99.99**	99.98	99.99	99.97	99.98	99.98	99.96	99.97
Fault 5	**95.00**	88.50	71.60	63.88	57.90	55.8	53.48	**59.69**
Fault 6	**99.50**	86.10	96.42	87.76	**99.98**	76.4	87.99	89.13
Fault 7	**99.98**	99.98	99.99	99.99	99.97	99.96	99.97	99.96

The boldfaced values show best FDR for the corresponding model.

The FAR metric is also considered to evaluate the proposed models' performance. The GCN VAE achieved 0.3% and 0.1% false alarm rates for convolutional window sizes of 25 and 35, respectively. However, the LSTM AE showed 0.15% and 0.18%; LSTM VAE showed 0.09% and 0.07%; and finally, the feed-forward AE model gained 1.05% and 1.55% FAR for convolution window sizes 25 and 35, respectively. Although LSTM VAE showed good FAR compared to the GCN VAE, still, the FAR for GCN VAE is less than 1%.

We validated the model's capability to identify the faulty sensor or feature-causing system malfunctions. In order to do so, we intentionally introduced noise to each feature individually while keeping the others unchanged. Subsequently, we computed the reconstruction error for each feature. The results are displayed in Table 5, showing the reconstruction error for each feature of the GCN VAE with a noise mean of 1.50 and a standard deviation of 3.50. The initial row of the table displays the reconstruction errors for each feature in the absence of added noise. The table clearly indicates that modifying the Ipv feature leads to a higher reconstruction error compared to the other features, which remain unchanged. This pattern also holds when noisy components are added to other features. Furthermore, the reconstruction error of the noisy features is notably higher than that of the features without noise. Similarly, Table 6 shows the effect of individual reconstruction error of the features with and without noise for a noise mean of 2.50 and a standard deviation of 4.50. From these tables, we can observe that the reconstruction errors for the features are increased from Table 5 to Table 6 with the increase in noise density.

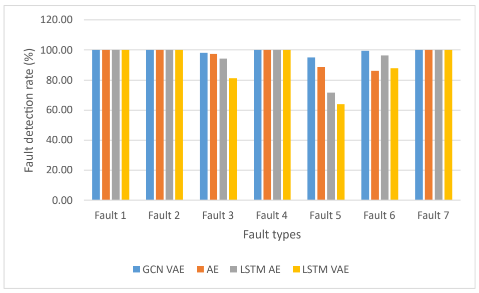
Table 5. Feature-specific reconstruction error for model validation ($\mu = 1.50$ and $\sigma = 3.50$).

Features	Ipv	Vpv	Vdc	Ia	Ib	Ic	Va	Vb	Vc	Iabc	If	Vabc	Vf
WN	0.03	0.03	0.05	0.01	0.01	0.01	0.02	0.01	0.01	0.05	0.02	0.02	0.02
Ipv	**3.58**	0.05	0.41	0.08	0.27	0.25	1.36	0.33	0.43	0.43	0.13	0.26	0.16
Vpv	0.24	**3.05**	0.04	1.38	1.09	1.60	0.16	5.34	0.21	0.08	0.35	0.17	0.07
Vdc	0.13	0.10	**7.54**	0.08	0.81	0.14	0.35	0.12	0.17	1.84	0.12	0.12	0.07
Ia	0.16	0.14	0.15	**12.71**	1.12	0.44	0.40	2.08	3.19	0.04	0.11	0.21	0.16

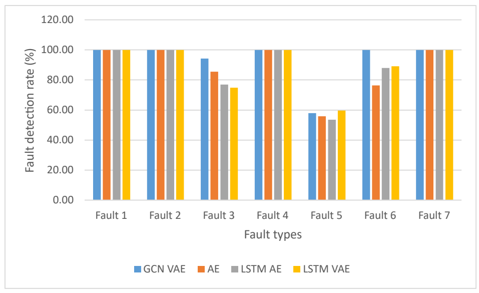
Table 5. *Cont.*

Features	Ipv	Vpv	Vdc	Ia	Ib	Ic	Va	Vb	Vc	Iabc	If	Vabc	Vf
Ib	0.26	0.13	0.10	0.39	**5.72**	0.38	1.07	0.92	0.45	0.06	0.10	0.18	0.17
Ic	0.17	0.02	0.05	0.77	1.01	**14.16**	0.27	0.39	1.66	0.05	0.06	0.44	0.14
Va	0.21	0.18	0.14	0.32	0.54	0.26	**18.30**	2.61	3.01	0.14	0.10	0.18	0.16
Vb	0.18	0.10	0.04	0.33	2.81	0.19	0.92	**9.58**	0.43	0.07	0.07	0.26	0.15
Vc	0.19	0.02	0.06	0.83	1.34	0.12	0.32	0.45	**7.84**	0.03	0.05	0.33	0.13
Iabc	0.46	0.15	1.15	0.26	0.42	0.14	0.52	0.24	0.19	**3.93**	0.27	0.59	0.13
If	0.06	0.26	0.04	0.27	0.17	0.30	1.02	0.32	0.08	0.05	**1.86**	0.19	0.15
Vabc	0.21	0.07	0.05	0.59	1.33	2.03	0.64	0.18	0.19	0.21	0.11	**3.21**	0.12
Vf	0.12	0.10	0.30	1.97	0.99	0.24	0.29	0.18	0.20	0.11	0.13	0.17	**2.36**

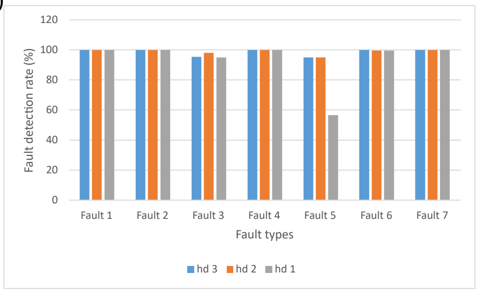
WN—without noise; the boldfaced numbers show the effect of adding noise to the corresponding features.



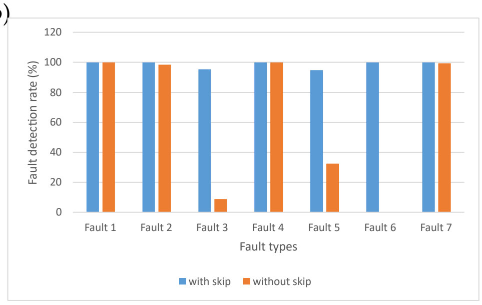
(**a**)



(**b**)



(**c**)



(**d**)

Figure 4. (**a**) FDR comparison of GCN VAE with other AE models for convolutional window size 35, (**b**) FDR comparison of GCN VAE with other AE models for convolutional window size 25, (**c**) FDR comparison of GCN VAE for three hidden dimensions, and (**d**) FDR comparison of GCN VAE for skip connections.

Table 6. Feature-specific reconstruction error for model validation ($\mu = 2.50$ and $\sigma = 4.50$).

Features	Ipv	Vpv	Vdc	Ia	Ib	Ic	Va	Vb	Vc	Iabc	If	Vabc	Vf
WN	0.03	0.03	0.05	0.01	0.01	0.01	0.02	0.01	0.01	0.05	0.02	0.02	0.02
Ipv	**7.90**	0.07	0.57	0.10	0.38	0.39	2.02	0.46	0.66	0.56	0.20	0.33	0.26
Vpv	0.41	**7.19**	0.06	1.74	1.61	2.42	0.26	7.23	0.33	0.13	0.53	0.27	0.09
Vdc	0.15	0.11	**13.77**	0.10	1.14	0.19	0.52	0.15	0.26	2.57	0.16	0.15	0.09
Ia	0.24	0.28	0.22	**22.80**	1.44	0.66	0.53	2.72	4.06	0.06	0.18	0.30	0.25
Ib	0.39	0.25	0.14	0.64	**12.45**	0.54	1.49	1.14	0.62	0.12	0.15	0.27	0.27
Ic	0.28	0.03	0.08	1.16	1.15	**24.50**	0.41	0.58	2.09	0.09	0.10	0.67	0.21
Va	0.33	0.35	0.24	0.44	0.67	0.38	**30.72**	3.40	3.77	0.22	0.16	0.26	0.25
Vb	0.27	0.20	0.07	0.54	3.29	0.27	1.28	**18.33**	0.59	0.13	0.12	0.42	0.24
Vc	0.32	0.03	0.10	1.25	1.55	0.19	0.48	0.67	**15.56**	0.04	0.08	0.50	0.21
Iabc	0.65	0.20	1.82	0.27	0.68	0.18	0.74	0.31	0.29	**8.15**	0.39	0.88	0.19
If	0.09	0.37	0.07	0.31	0.22	0.46	1.51	0.50	0.10	0.08	**4.86**	0.29	0.23
Vabc	0.34	0.11	0.06	0.80	1.73	3.02	1.01	0.25	0.32	0.36	0.17	**7.46**	0.20
Vf	0.16	0.16	0.43	2.83	1.39	0.39	0.41	0.26	0.24	0.15	0.22	0.25	**5.98**

WN–without noise; the boldfaced numbers show the effect of adding noise to the corresponding features.

Based on our validation process, we can confidently utilize the GCN VAE model to detect faulty components in the PV array. Table 7 presents the individual feature reconstruction errors for the test data of faulty and non-faulty parts. It is apparent from Table 7 that the feature Vpv is responsible for causing faults in the PV array for all faults except Fault 6, while Vdc causes the system to be faulty under Fault 6. However, Vdc also contributes to fault types 5 and 3. This analysis is further supported by Figure 2, where the KS test is used to identify the faulty features in the original dataset. Figure 5 graphically illustrates the individual reconstruction error for the GCN VAE. This figure illustrates that the feature Vpv contributes to making the solar array faulty for fault types 1, 2, 3, 4, 5, and 7, while Vdc contributes to fault types 3, 5, and 6.

Table 7. Faulty feature identification using trained GCN VAE model.

Features	NFT	Fault 1	Fault 2	Fault 3	Fault 4	Fault 5	Fault 6	Fault 7
Ipv	0.01	2.51	0.71	0.71	144.90	0.12	0.55	0.26
Vpv	0.02	**148.21**	**56.76**	**3.17**	**391,027.38**	**1.82**	0.28	**31.76**
Vdc	0.03	4.19	2.26	**2.10**	2.41	**1.73**	**4.28**	0.98
Ia	0.03	2.84	1.23	0.13	3.69	0.06	0.21	0.26
Ib	0.03	2.26	0.92	0.13	6.47	0.08	0.17	0.29
Ic	0.01	0.27	0.58	0.04	1.27	0.02	0.05	0.12
Va	0.02	2.44	1.44	0.22	3.56	0.09	0.19	0.64
Vb	0.02	1.39	1.14	0.13	5.29	0.06	0.07	0.29
Vc	0.01	0.25	0.60	0.04	1.46	0.02	0.06	0.20
Iabc	0.04	2.92	3.86	1.09	3.48	0.62	0.10	2.45
If	0.01	2.90	10.32	0.51	1.41	0.45	0.08	1.52
Vabc	0.04	0.10	1.42	0.12	3.61	0.14	0.47	2.26
Vf	0.04	1.57	30.24	0.41	1.09	0.17	0.23	0.28

The boldfaced values show the features with the highest reconstruction error for different fault types.

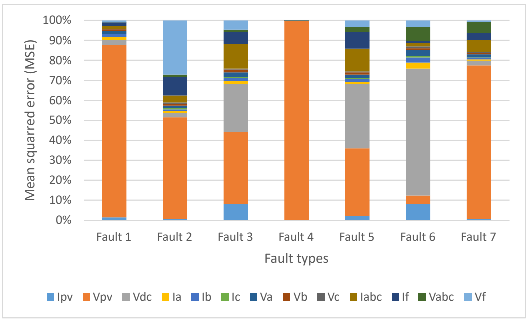


Figure 5. Fault diagnosis outcome after the reconstruction error from each individual feature is analyzed. The figure shows that the Vpv feature is responsible for fault types 1, 2, 3, 4, 5, and 7, while Vdc is responsible for fault types 3, 5, and 6.

4. Conclusions

Solar arrays are a critical renewable energy production system that reduces reliance on fossil fuels. However, the lack of proper maintenance and monitoring may reduce the overall efficiency and performance of these systems. Faults are one of the crucial indicators for a degrading solar array. In this study, we proposed a DL-based fault detection and diagnosis (identifying faulty components) process for the monitoring and timely detection of potential faults. A graph convolutional network model-based variational autoencoder is proposed for serving the purpose of detecting and identifying potential faults. The developed model effectively learns the spatial and temporal behavior of features, which enhances its performance in detecting faults compared to the LSTM-based autoencoder and VAE models. The conventional AE models cannot explicitly learn the spatial and temporal behavior of the features, which makes it difficult for these models to learn the features properly. The proposed model was evaluated using a recently published dataset for the IPPT mode PV array, and the experimental results demonstrate its superior performance, with a fault detection rate exceeding 95%. Furthermore, in identifying faulty components within the large PV array, the individual reconstruction errors for each feature are considered and validated through KS test and noise injection. In our future work, we will analyze the models' diagnosis performance by injecting noise into multiple features.

Author Contributions: M.A.: conceptualization, methodology, writing—original draft, and funding acquisition. A.P.: software (https://www.anaconda.com/ accessed on 13 July 2024), formal analysis, visualization, and investigation. M.J.H.: validation and writing—review and editing. K.N.: conceptualization, methodology, supervision, funding acquisition, and writing—review and editing. W.U.N.: formal analysis. A.H.: validation and formal analysis. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to express their sincere appreciation to their respective institutions for providing the resources necessary to carry out this research. We are grateful to our colleagues and mentors for their encouragement and constructive feedback throughout the project.

Conflicts of Interest: Author Md Junayed Hasan was employed by the company Dataxense. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Maka, A.O.; Alabid, J.M. Solar energy technology and its roles in sustainable development. *Clean Energy* **2022**, *6*, 476–483. [CrossRef]
2. Rodwell, J. Harness the Power of the Sun with Our Solar PV Panel Installations. Available online: https://www.jem-energy.co.uk/articles/2024/5/7/harness-the-power-of-the-sun-with-our-solar-pv-panel-installations#:~:text=By%20harnessing%20sunlight%20to%20generate,is%20improving%20all%20the%20time (accessed on 15 August 2024).
3. Anonymous. Energy Independence with Solar Power. Available online: https://saveenergyuk.co.uk/energy-independence-with-solar-power/#:~:text=The%20benefits%20of%20solar%20power,footprint%2C%20helping%20combat%20climate%20change (accessed on 15 August 2024).
4. Wang, F.; Harindintwali, J.D.; Yuan, Z.; Wang, M.; Wang, F.; Li, S.; Yin, Z.; Huang, L.; Fu, Y.; Li, L.; et al. Technologies and perspectives for achieving carbon neutrality. *Innovation* **2021**, *2*, 100180. [CrossRef] [PubMed]
5. Lozanova, S. Common Solar Panel Defects: Solar Panel Discoloration & Delamination. Available online: https://www.greenlancer.com/post/common-solar-panel-defects (accessed on 15 August 2024).
6. Bakdi, A.; Bounoua, W.; Guichi, A.; Mekhilef, S. Real-time fault detection in PV systems under MPPT using PMU and high-frequency multi-sensor data through online PCA-KDE-based multivariate KL divergence. *Int. J. Electr. Power Energy Syst.* **2021**, *125*, 106457. [CrossRef]
7. Yu, X.; Tang, B.; Zhang, K. Fault diagnosis of wind turbine gearbox using a novel method of fast deep graph convolutional networks. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–14. [CrossRef]
8. Chen, Z.; Xu, J.; Peng, T.; Yang, C. Graph convolutional network-based method for fault diagnosis using a hybrid of measurement and prior knowledge. *IEEE Trans. Cybern.* **2021**, *52*, 9157–9169. [CrossRef]
9. Cen, J.; Yang, Z.; Liu, X.; Xiong, J.; Chen, H. A review of data-driven machinery fault diagnosis using machine learning algorithms. *J. Vib. Eng. Technol.* **2022**, *10*, 2481–2507. [CrossRef]
10. Sahu, A.R.; Palei, S.K.; Mishra, A. Data-driven fault diagnosis approaches for industrial equipment: A review. *Expert Syst.* **2024**, *41*, e13360. [CrossRef]
11. Jiang, G.; Li, W.; Fan, W.; He, Q.; Xie, P. TempGNN: A Temperature-Based Graph Neural Network Model for System-Level Monitoring of Wind Turbines with SCADA Data. *IEEE Sens. J.* **2022**, *22*, 22894–22907. [CrossRef]
12. Arifeen, M.; Ghosh, T.; Islam, R.; Ashiquzzaman, A.; Yoon, J.; Kim, J. Autoencoder based Consensus Mechanism for Blockchain-enabled Industrial Internet of Things. *Internet Things* **2022**, *19*, 100575. [CrossRef]
13. Arifeen, M.; Petrovski, A. Temporal Graph Convolutional Autoencoder Based Fault Detection for Renewable Energy Applications. In Proceedings of the 2024 IEEE 7th International Conference on Industrial Cyber-Physical Systems (ICPS), St. Louis, MO, USA, 12–15 May 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1–6.
14. Arifeen, M.; Petrovski, A. Bayesian Optimized Autoencoder for Predictive Maintenance of Smart Packaging Machines. In Proceedings of the 2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems (ICPS), Wuhan, China, 8–11 May 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1–6.
15. Qian, J.; Song, Z.; Yao, Y.; Zhu, Z.; Zhang, X. A review on autoencoder based representation learning for fault detection and diagnosis in industrial processes. *Chemom. Intell. Lab. Syst.* **2022**, *231*, 104711. [CrossRef]
16. Habib, M.A.; Hasan, M.J.; Kim, J.M. A Lightweight Deep Learning-Based Approach for Concrete Crack Characterization Using Acoustic Emission Signals. *IEEE Access* **2021**, *9*, 104029–104050. [CrossRef]
17. Liu, Y.; Ding, K.; Zhang, J.; Li, Y.; Yang, Z.; Zheng, W.; Chen, X. Fault diagnosis approach for photovoltaic array based on the stacked auto-encoder and clustering with IV curves. *Energy Convers. Manag.* **2021**, *245*, 114603. [CrossRef]

18. Chen, W.; Jiang, Z.; Pei, T.; Zhang, X. A Fault Diagnosis Method for Photovoltaic Arrays Based on Dropout Optimized Stack Autoencoders. In Proceedings of the 2022 IEEE 6th Conference on Energy Internet and Energy System Integration (EI2), Chengdu, China, 11–13 November 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 78–83.
19. Seghiour, A.; Abbas, H.A.; Chouder, A.; Rabhi, A. Deep learning method based on autoencoder neural network applied to faults detection and diagnosis of photovoltaic system. *Simul. Model. Pract. Theory* **2023**, *123*, 102704. [CrossRef]
20. Edun, A.S.; LaFlamme, C.; Kingston, S.R.; Furse, C.M.; Scarpulla, M.A.; Harley, J.B. Anomaly detection of disconnects using SSTDR and variational autoencoders. *IEEE Sens. J.* **2022**, *22*, 3484–3492. [CrossRef]
21. Alrifaey, M.; Lim, W.H.; Ang, C.K.; Natarajan, E.; Solihin, M.I.; Juhari, M.R.M.; Tiang, S.S. Hybrid deep learning model for fault detection and classification of grid-connected photovoltaic system. *IEEE Access* **2022**, *10*, 13852–13869. [CrossRef]
22. Hu, W.; Dong, Z.; Huang, X.; Gao, Y.; Zhang, Z.; Hao, J. Photovoltaic inverter anomaly detection method based on LSTM serial depth autoencoder. *J. Phys. Conf. Ser.* **2023**, *2474*, 012026. [CrossRef]
23. Rao, S.; Muniraju, G.; Tepedelenlioglu, C.; Srinivasan, D.; Tamizhmani, G.; Spanias, A. Dropout and pruned neural networks for fault classification in photovoltaic arrays. *IEEE Access* **2021**, *9*, 120034–120042. [CrossRef]
24. Petrovski, A.; Arifeen, M.M.; Hasan, M.J.; Zeeshan, A. Exponential Degradation Model Based Remaining Life Prediction for Tools of Milling Machine. In Proceedings of the Intelligent Information Technologies for Industry-IITI24, Harbin, China, 1–7 November 2024.
25. Hasan, M.J.; Arifeen, M.M.; Sohaib, M.; Rohan, A.; Kannan, S. Enhancing Gas-Pipeline Monitoring with Graph Neural Networks: A New Approach for Acoustic Emission Analysis under Variable Pressure Conditions. In Proceedings of the British Institute of Non-Destructive Testing (BINDT)—CM 2024, Oxford, UK, 18–20 July 2024.
26. Miele, E.S.; Bonacina, F.; Corsini, A. Deep anomaly detection in horizontal axis wind turbines using graph convolutional autoencoders for multivariate time series. *Energy AI* **2022**, *8*, 100145. [CrossRef]
27. Liu, J.; Wang, X.; Xie, F.; Wu, S.; Li, D. Condition monitoring of wind turbines with the implementation of spatio-temporal graph neural network. *Eng. Appl. Artif. Intell.* **2023**, *121*, 106000. [CrossRef]
28. Liu, L.; Zhao, H.; Hu, Z. Graph dynamic autoencoder for fault detection. *Chem. Eng. Sci.* **2022**, *254*, 117637. [CrossRef]
29. Wu, W.; Song, C.; Zhao, J.; Wang, G. Knowledge-Enhanced Distributed Graph Autoencoder for Multiunit Industrial Plant-Wide Process Monitoring. *IEEE Trans. Ind. Inform.* **2023**, *20*, 1871–1883. [CrossRef]
30. Li, T.; Suna, C.; Yan, R.; Chen, X.; Fink, O. A Novel Unsupervised Graph Wavelet Autoencoder for Mechanical System Fault Detection. *arXiv* **2023**, arXiv:2307.10676. [CrossRef]
31. Goswami, U.; Rani, J.; Kodamana, H.; Kumar, S.; Tamboli, P.K. Fault detection and isolation of multi-variate time series data using spectral weighted graph auto-encoders. *J. Frankl. Inst.* **2023**, *360*, 6783–6803. [CrossRef]
32. Liu, Y.; Yang, Z.; Yu, Z.; Liu, Z.; Liu, D.; Lin, H.; Li, M.; Ma, S.; Avdeev, M.; Shi, S. Generative artificial intelligence and its applications in materials science: Current situation and future perspectives. *J. Mater.* **2023**, *9*, 798–816. [CrossRef]
33. Wei, R.; Garcia, C.; El-Sayed, A.; Peterson, V.; Mahmood, A. Variations in variational autoencoders-a comparative evaluation. *IEEE Access* **2020**, *8*, 153651–153670. [CrossRef]
34. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]
35. Ullah, I.; Manzo, M.; Shah, M.; Madden, M.G. Graph convolutional networks: Analysis, improvements and results. *Appl. Intell.* **2022**, *52*, 9033–9044. [CrossRef]
36. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *arXiv* **2016**, arXiv:1606.09375
37. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
38. Jiang, G.; Xie, P.; He, H.; Yan, J. Wind turbine fault detection using a denoising autoencoder with temporal information. *IEEE/ASME Trans. Mechatronics* **2017**, *23*, 89–100. [CrossRef]
39. Berger, V.W.; Zhou, Y. Kolmogorov–Smirnov test: Overview. In *Wiley Statsref: Statistics Reference Online*; Wiley: Hoboken, NJ, USA, 2014.
40. Smith, K. Convolution and Filtering in the Time Domain. Available online: https://kls2177.github.io/Climate-and-Geophysical-Data-Analysis/chapters/Week6/convolution_filters_in_time.html (accessed on 21 August 2024).
41. Bilogur, A. Denoising Algorithms. Available online: https://www.kaggle.com/code/residentmario/denoising-algorithms (accessed on 21 August 2024).
42. Laurikkala, J.; Juhola, M.; Kentala, E.; Lavrac, N.; Miksch, S.; Kavsek, B. Informal identification of outliers in medical data. In Proceedings of the Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology, Berlin, Germany, 20–25 August 2000; Volume 1, pp. 20–24.
43. Smiti, A. A critical overview of outlier detection methods. *Comput. Sci. Rev.* **2020**, *38*, 100306. [CrossRef]