Chapter 2

A Graph Convolutional Network Based Fault Identification for Low Voltage DC Microgrid

DC microgrid is a solution in low voltage levels that integrates RES, loads, and storage devices through power electronic converters. DC microgrids offer several benefits over AC systems, such as fewer conversion stages for contemporary DC electronic devices, greater power transfer capability, improved efficiency, and reduced cable losses since the skin effect is not present [108]. Besides several advantages of the DC microgrids to that of AC, it possess challenges associated with protection. Therefore, the protection of DC microgrids requires innovative solutions that can rapidly detect, classify, and isolate faults while accommodating bidirectional power flow, variable fault current contributions, and converter-based interfacing. Advanced fault detection techniques such as those based on artificial intelligence, signal processing, and graph-based models are being actively researched to address these challenges [24]. A robust and adaptive protection scheme is critical to ensure the safe and reliable operation of future DC microgrids.

2.1 Indroduction

In this chapter, a GCN-based fault diagnosis algorithm is presented for a low-voltage DC microgrid. GCN is an extension of the CNN model, which utilizes the network topology's explicit spatial information and measurement data to identify a fault. The

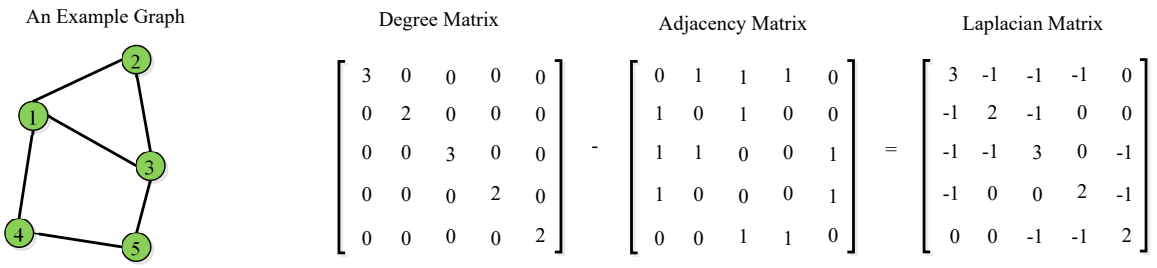


Figure 2.1: Illustration of graph laplacian matrix.

fault identification task is formulated as a node classification problem over the undirected graph, and GCN is used as a solution. It has a more substantial feature extraction ability even in the presence of noise and bad data. The adjacency matrix for GCN is developed by considering the network topology as an inherent graph. The bus voltage and line current samples during the fault are considered as the node attributes. The performance of the proposed method under different operating conditions are compared with various machine learning techniques such as CNN, SVM, and FCN. The performance of the proposed method is found to have improved fault detection and isolation ability.

2.2 Introduction to graph convolutional network

2.2.1 Mathematical Notation of a Graph

A basic graph can be expressed in the following way:

$$G = G(V, E) \tag{2.1}$$

where V is the set of nodes and E is the set of edges. For a node $v_i \in V$, the value of $e_{jk} = (v_j, v_k) \in E$ represents an edge between v_j and v_k. Typically, it is common to represent a graph through adjacency matrix $A \in R^{N \times N}$ where N is the number of nodes i.e. $N = |V|$. The elements of adjacency matrix A_{jk} represents presence of an edge between nodes v_j and v_k. In general,

$$A(j,k) = \begin{cases} 1, & \text{if } (v_j, v_k \in E \text{ and } j \neq k; \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

In practice, a graph may have a node feature matrix, often known as node attributes X $\in \mathcal{R}^{N \times f}$ where f denotes the node feature vector's dimension. The degree matrix

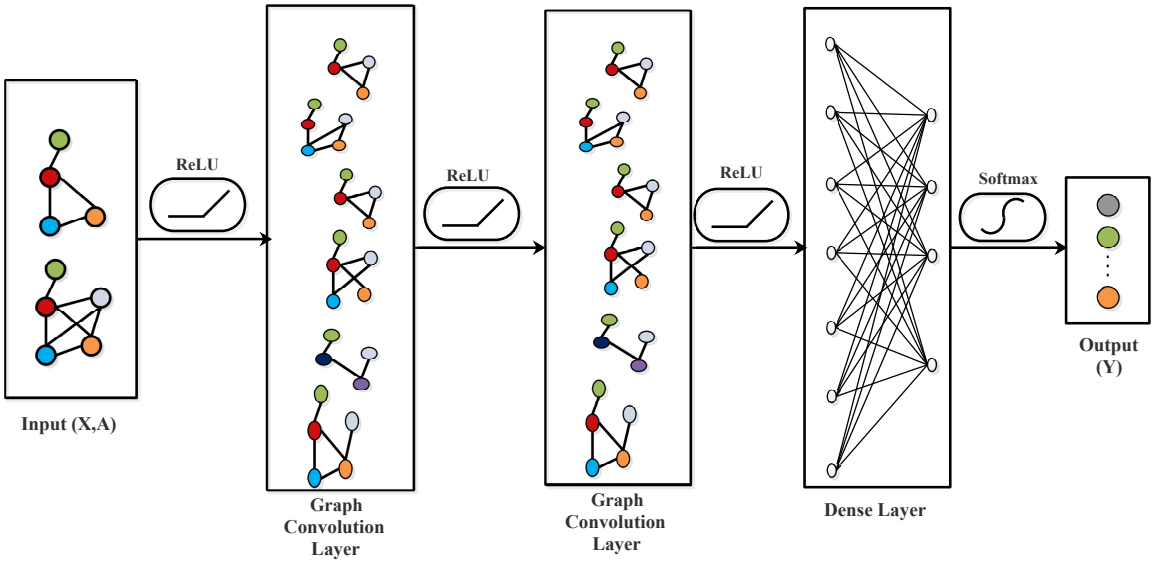


Figure 2.2: The structure of GCN model.

$D \in \mathcal{R}^{N\times N}$ is a diagonal matrix that can be calculated as $D_{jj} = \sum_{j=1}^{N} A_{jk}$. The adjacency and degree matrix of a typical graph is shown in Fig. 2.1.

2.2.2 Graph Convolutional Networks

GCN can be thought of as a CNN extension. The original derivation of graph convolutional network was made on the fundamentals of graph theory in association with convolutional theorem having an intention to application in the data processing. Throughout the consistent enhancement and optimization of the GCN, it becomes easier to understand the concept. A typical GCN structure is shown in Fig. 2.2.

The Graph convolutional network was proposed by [109], whose one layer of operation is given by:

$$Z = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}XW) = \sigma\left(\hat{A}XW\right) \tag{2.3}$$

where $\hat{A}$ is the self normalized adjacency matrix, $\hat{A} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$ and W is trainable weight matrix. The adjacency matrix is normalized to keep the scale of the eigenvector unaltered after multiplication. $\widetilde{A}$ is the adjacency matrix with self-loop. $\widetilde{A} = A + I$ makes each node consider the eigenvector of self and every other node in the graph. σ is the activation function, e.g. Relu function. Similar to convolutional neural networks, GCN uses graph fourier transform (GFT) for feature extraction from the graph.

2.2.3 Graph Fourier Transform

Consider the undirected graph $G = (V, \varepsilon, A)$ in which V represents the set of vertices, ε represents the set of edges, and, and A represents the adjacency matrix. Here $|V| = n$, number of nodes in the graph. The eigen decomposition of the normalized graph Laplacian matrix L_n is used to calculate the GFT of a signal X over a graph G. The Laplacian of a signal at a given point can be considered as a measure of how different is the signal at from its neighbors. For a graph with adjacency matrix A and degree matrix D, the unnormalized graph Laplacian L_u is given as:

$$L_u = D - A \tag{2.4}$$

where degree matrix D is a diagonal matrix whose diagonal element D_{jj} represents the weighted sum of all the edges connected with j^{th} node. The normalized graph Laplacian then becomes:

$$L_n = D^{-\frac{1}{2}} L_u D^{-\frac{1}{2}} = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \tag{2.5}$$

L_n is a symmetric matrix that has real eigenvalues and orthogonal eigenvectors. Eigen decomposition of L_n represented as $L_n = U \Lambda U^T$, where $U = (u_1, u_2, ..., u_n)$ are the orthonormal eigenvectors of L_n, and $\Lambda = (\lambda_1, \lambda_2, ..., \lambda_n)$ is a diagonal matrix with non-negative eigen values. GFT of X is defined as [110]:

$$Z = U^T X \tag{2.6}$$

The original signal X can be computed by using inverse-GFT as:

$$X = UZ \tag{2.7}$$

Convolution on a graph in the spectral domain can be done in the same way it is done on discrete euclidean spaces with the use of the fourier transform. To put it another way, the following is the spectral convolution of the two signals g and f:

$$g * f = U((U^T g) o (U^T f)) = U \; diag\,(\hat{g}_1, \hat{g}_2, ..., \hat{g}_n,)\, U^T f \tag{2.8}$$

where o indicates the elementwise multiplication of two vectors.

2.3 Graph convolutional network for fault identification

This section begins with a concise overview of the fault location task in the DC microgrid, emphasizing its significance in ensuring operational reliability and minimizing system downtime. Following this, the concept of spectral graph convolution is introduced, highlighting its mathematical foundation and relevance to processing graph-structured data such as power network topologies. The discussion then moves to the generation of test cases for a low-voltage DC microgrid, which are designed to reflect various physical and environmental operating conditions, including load fluctuations, converter behavior, and the intermittent nature of renewable energy sources. These scenarios are essential for evaluating the robustness and generalizability of the proposed approach. Finally, the application of a GCN for fault identification within the DC microgrid is presented. This includes the construction of the graph model, feature representation, and the classification framework used to detect and localize faults with high accuracy.

2.3.1 Formulation of Fault Identification Problem

In this work, the fault identification problem is formulated as node classification, where each node belongs to a particular class. The adjacency matrix is formulated by considering the common bus between the cables in the physical microgrid. If cable 'j' and cable 'k' have a common bus in the microgrid, then $A_{jk} = 1$; otherwise, $A_{jk} = 0$ in the adjacency matrix. Training data is generated by operating the case in different scenarios, including changing the connected load by different values and adding and removing different DGs to fault at different cables with variation in fault resistances and fault locations. It is assumed that the measurement of voltage of each bus and current through each cable is available. Thus, we have access to all these measurements. A data sample from measurements can be represented as $X \in \mathcal{R}^{n_0 \times f_0}$, where n_0 is the number of observations and f_0 is the number of measured parameters. Using a data sample matrix X_i, as a preliminary step, the faulted line can be obtained by $y_i = f(X_i)$, where f is the specific model for fault classification.

Table 2.1: Rating of the DC microgrid components.

Components	**Ratings**
DC Grid Voltage	600 V
Base Power	500 kW
Grid VSC	500 kW
Solar Panel	Vmp = 54.7 V, Imp = 5.58 A at STC
PV Converter	250 kW
Diesel Generator	400 kVA
Battery	220 V, 0.65 kAh
Battery DC-DC Converter	250 kW
Filter Capacitance	20 mF
Cable Length	0.75 km to 1.5 km
DC Load	0 - 500 kW

2.3.2 System Description and Generation of Test Cases

PSCAD/EMTDC simulation is used for modeling of DC microgrid and generation of test cases. The basic configuration of the DC microgrid network is extracted from the test system proposed in [14], [111], and the system parameter is given in Table 2.1. DC microgrid is simulated in two modes, namely grid-connected mode and islanded mode. The network consists of a PV system (250 kWp) that operates under maximum power point tracking (MPPT) mode to deliver maximum power to the system. A battery energy storage system with a bidirectional DC-DC converter is simulated as a storage unit. Charging discharging control for the energy storage system is implemented as per state of charge (SOC) control, allowing the battery to charge when SOC is less than 40% and block charging when it crosses the 95% level. The algorithm permits discharging when SOC lies between 40% to 95% and blocks discharging when it reaches below 40% level. A 400 kVA diesel generator with AC/DC converter is simulated, which works as a local generating unit. A 500 KVA bi-directional AC-DC converter integrates the DC microgrid with the AC utility grid. During normal operation, the voltage source converter (VSC) controls the DC voltage of the grid by balancing active power in grid-connected mode. A variable (0-500 kW) DC load is connected to the system with a DC-DC converter. Frequency-dependent phase model of underground cable is considered as lines in the system. The

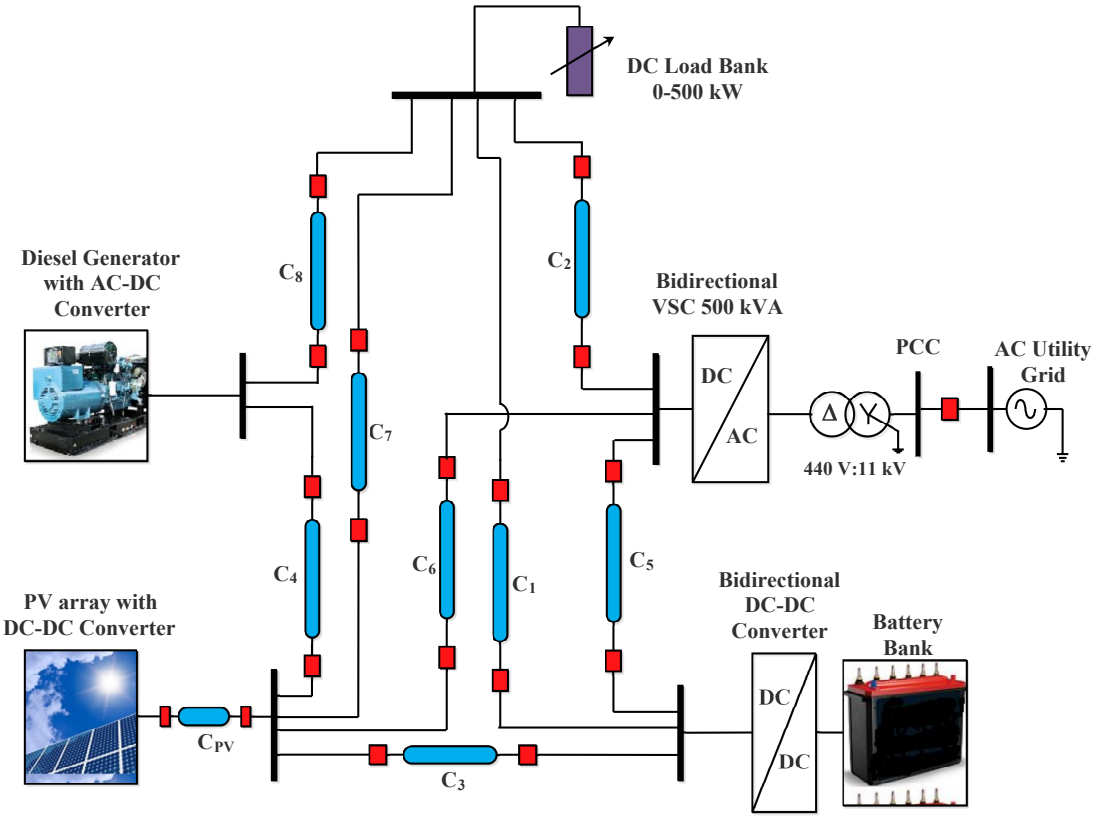


Figure 2.3: Schematic of the DC microgrid under consideration.

core conductor resistivity is 2.0×10^{-8} Ωm and the sheath resistivity is 30×10^{-8} Ωm [55].

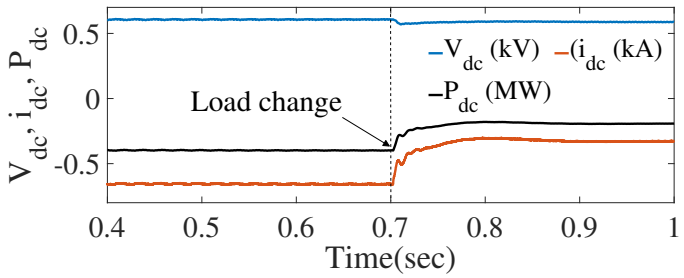
For the system depicted in Fig. 2.3, various faults are simulated for five different levels of fault resistances in each cable. Addition and removal of loads in steps of 25% in the range of 0 to 100% in grid-connected and islanded modes are also simulated to validate the performance of the proposed method during normal switching. The impact of the load variations on voltage, current at the point of common coupling (PCC), and power transfer from/to the utility grid is presented in Fig. 2.4, while Fig 2.5 shows the variation in fault current contribution from the utility grid (I_{dc}) for various DC cable faults.

Addition and removal of DGs were also incorporated while generating the test cases. Each case is simulated for nine sets of solar irradiance and temperature values for the PV system. In this way, we have a total of 2727 cases in grid-connected mode (Table 2.2) and 2655 cases in islanded mode operation of the DC microgrid.

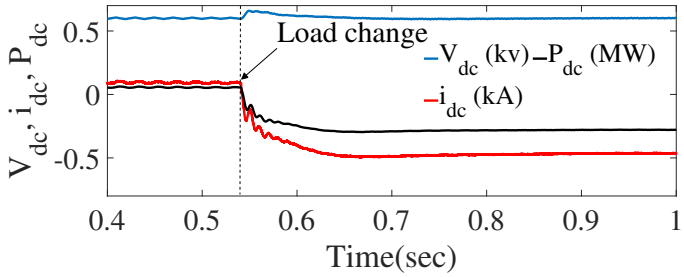
The fault inception time is 0.6 sec, and the fault duration is 0.05 sec. The PSCAD model of microgrid has a sampling frequency of 20 kHz, which means 1000 fault sam-

Table 2.2: Various test cases generated on DC microgrid network in grid connected mode.

Disturbance Event	**Parameter Variation**	**Number of Cases**
Events of normal Switching	Load addition, 0-100% in four steps	4
	Load removal, 0-100% in four steps	4
	Simultaneous DG (Diesel Generator/ Battery/PV) addition (3) with varying load (4).	3×4 = 12
	Simultaneous DG (Diesel Generator/ Battery/PV) Removal (3) with varying load (4).	3×4 = 12
Normal	No Variation	1
Fault Events	Three different faults (P-N, P-G, N-G) at two different locations with 5 different fault resistance (0 -15 Ω) in nine different cables	3×2 × 5 × 9 = 270
All these cases for 9 different sets of temperature and solar irradiance values		
Total cases generated for Grid-connected mode		(270 + 12 + 12 + 4+ 4 + 1) × 9 = 2727



(a)



(b)

Figure 2.4: Variation of voltage, current, and power at PCC with (a) addition of 50% load (b) reduction of 50% load.

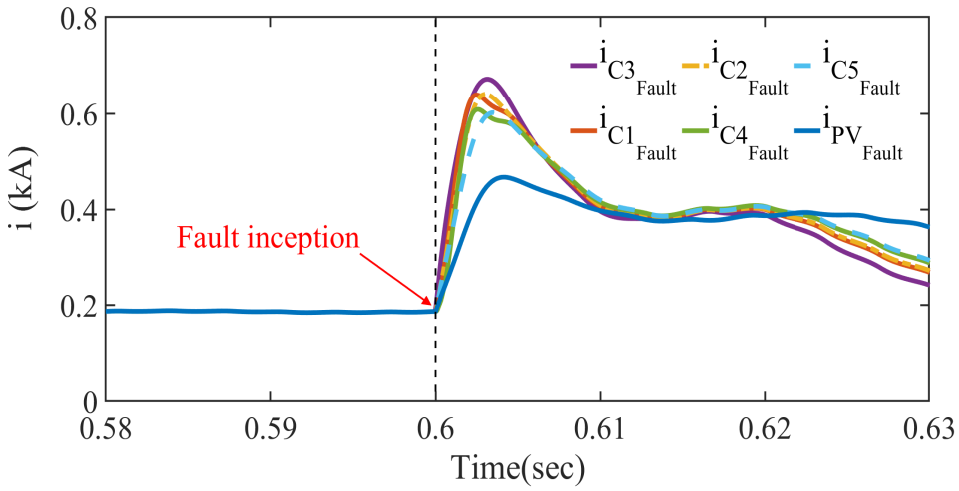


fig6

Figure 2.5: DC current variations at PCC with different cable faults.

ple values will be generated in the fault period of 0.05 sec. Each cable fault case was simulated with 9 sets of different values of solar irradiance and temperature and with 5 different values of fault resistance (R_f). Figure 2.6 shows the fault current response of PV cable pole-to-pole fault (PPF) for different fault resistance. Three fault types, namely positive pole-to-ground fault (PGF), negative pole-to-ground fault (NGF), and PPF were simulated in both the grid connected mode and islanded mode. After down sampling the data set, it is further divided into training and testing data in the ratio of 7:3.

Before being fed into GCN, each feature vector is normalized using min-max normalization (2.9) to correspond to the range [0,1] because the model's performance may be negatively impacted by the wide disparity in the numerical values of the feature vectors.

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2.9}$$

2.3.3 GCN for Fault Line Identification

Finally, the graph convolutional network is applied to the fault location task as described by the flow chart shown in Fig. 2.7. The adjacency matrix, unnormalized and normalized graph laplacian, is represented in Fig. 2.8. We utilize commonly used GCN for graph convolution operation in the form of feature transfer and aggregation through a self-normalized adjacency matrix. The first GCN layer takes node attribute matrix X and adjacency matrix $\hat{A}$ as inputs and multiplies both to transfer and aggregate the feature of adjacent nodes. Finally, the first GCN layer (Z^1) output is produced on which all nodes

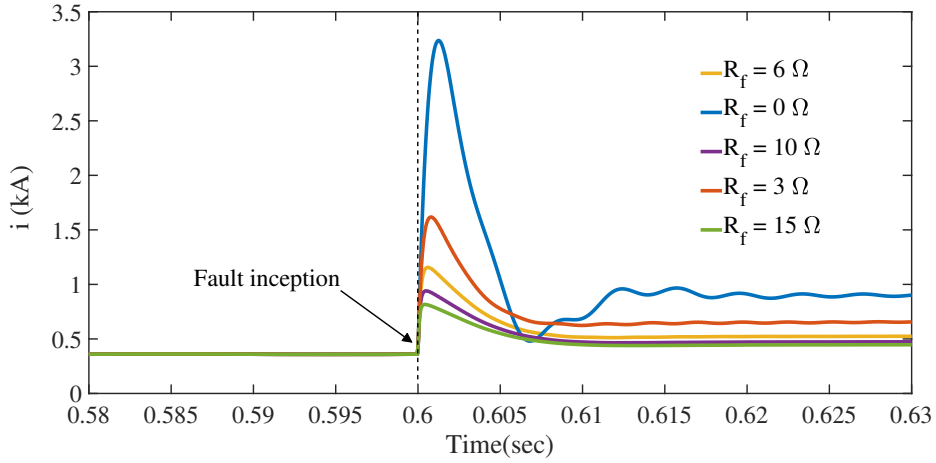


Figure 2.6: PV cable fault current variations with fault resistance during pole-to-pole fault.

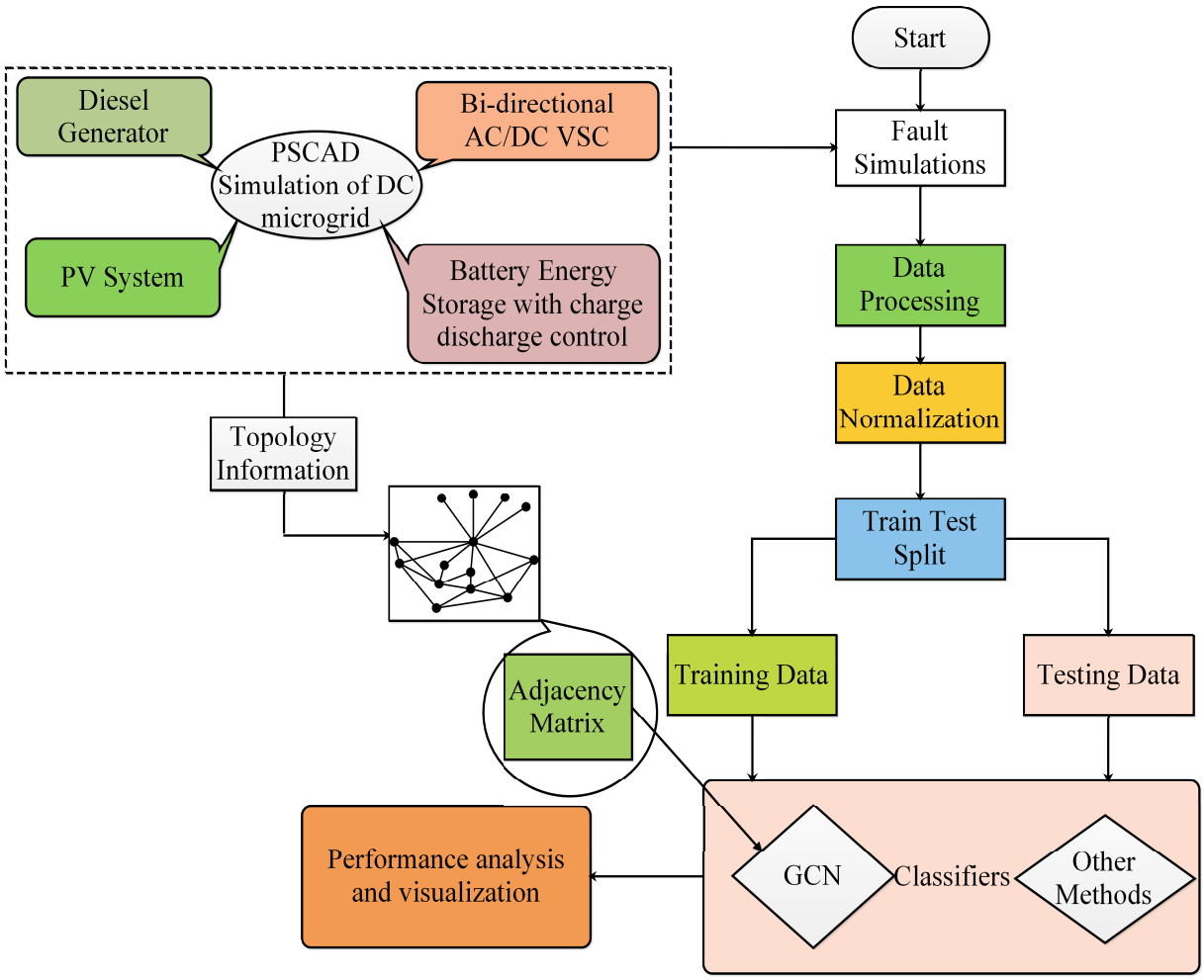
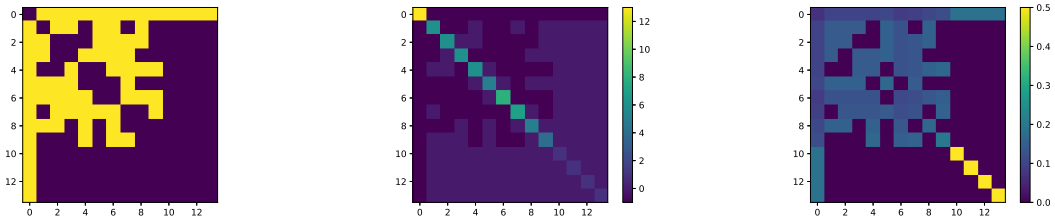


Figure 2.7: Workflow of the proposed method.



(a) (b) (c)

Figure 2.8: Processing of graph laplacian for GCN (a) adjacency matrix (b) unnormalized laplacian (c) normalized laplacian.

contain first-order neighborhood information.

It might be deduced that the output neurons of k^{th} GCN layers can express k-order neighborhood information. In this way, the hidden layers of GCN provide more prior information for the model training, so the hidden layer neurons have more extraordinary feature extraction ability after training. The output of the last graph convolution layer is flattened into a vector, which is then fed to the fully connected layer, which uses the softmax activation function to produce output.

The hyperparameter selection of GCN is made by taking [112] as a reference that states that the hidden layer in a graph convolutional network is usually set to 2 or 3. After testing and comparing the effects of different layers, we choose a model with two hidden layers. The number of hidden neurons is finalized to be 70 for each of the two layers. Rectified linear unit (ReLu) is selected as an activation function for each hidden layer which is defined as:

$$ReLu(x) = max(0, x) \tag{2.10}$$

Cross entropy error is usually preferred as a loss function for multiclass classification problems because it calculates the loss through a simple derivative and has a fast convergence rate [113] the expression is as follows:

$$CE\,(p, q) = -\sum_{i=1}^{C} p_i\ log\,(q_i) \tag{2.11}$$

where C represents the number of categories, p_i is the true positive value, and q_i is the predicted value. Adam optimizer is considered due to its fast convergence speed, small memory requirements, and high learning efficiency.

2.4 Results and Discussion

The performance of the proposed method is validated for different operating conditions, including normal switching and faults. This section represents the results of the proposed fault detection scheme for different fault conditions and other disturbance events.

2.4.1 Performance Evaluation and Comparison with Previous Methods

To verify the effectiveness of the proposed technique, the model is tested under different situations and operating modes, and the performance is illustrated in Fig. 2.9 with the confusion matrix (CM). A CM is a table used to evaluate the performance of a classification model. CM gives the count of true positive (TP), true negative (TN), false positive (FP), and false negative (FN), which means:

- TP: A label is correctly predicted and belongs to the original class.
- TN: A label is correctly predicted but does not belong to the original class.
- FP: A label is predicted as positive but does not belong to the original class.
- FN: A label is predicted as negative but belongs to the original class.

The overall classification accuracy is found to be 99.32% (Table 2.3), which shows that the proposed technique can classify faults and disturbances with higher accuracy. However, the average accuracy may be unable to provide a complete analysis of the model's performance. Therefore, the classification performance was further evaluated with the F1-score to investigate how the classifier acted for specific fault classes. The F1-score, a function of recall/sensitivity and precision, is regarded as ideal when it equals one and as the worst when it equals zero. The precision, also referred to as the positive predictive value, is defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{2.12}$$

Another metric, recall, which is known as the true positive rate or the sensitivity of the classifier, can be defined as:

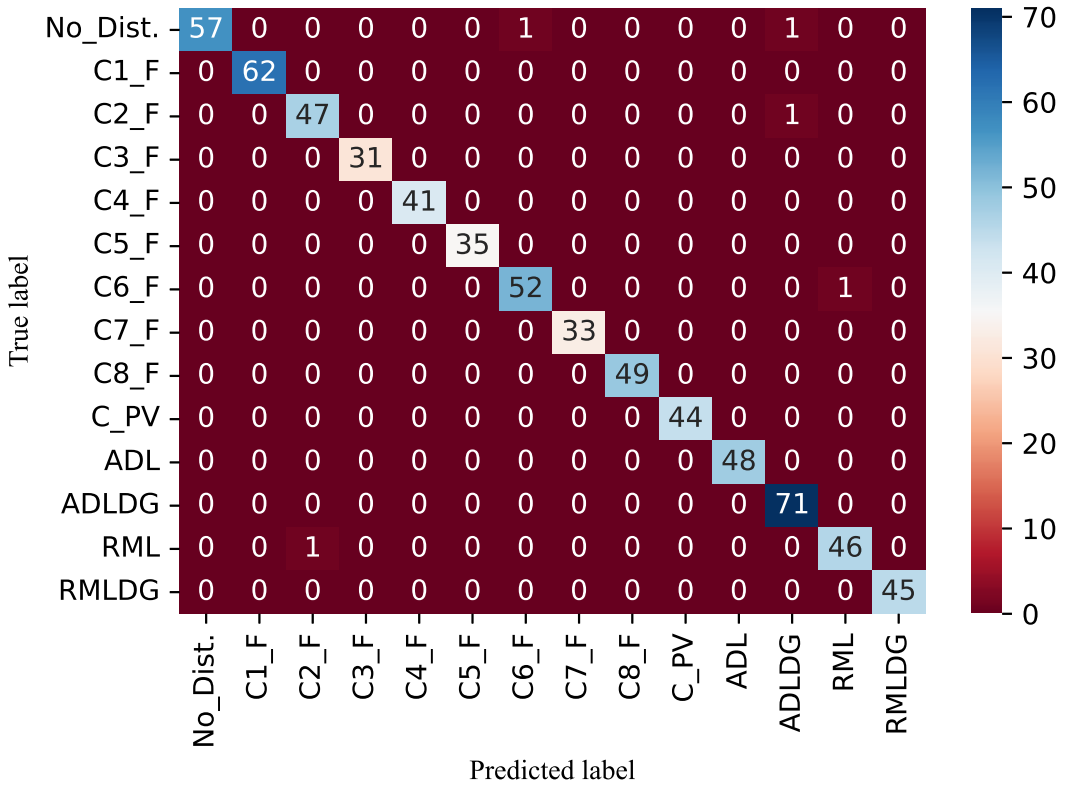


Figure 2.9: Confusion matrix of GCN based classifier in grid connected mode.

$$Recall/Sensitivity = \frac{TP}{TP + FN} \tag{2.13}$$

F1-score which takes precision and recall into account, is obtained as:

$$F1 - score = \frac{2 * (Precision * Recall)}{Precision + Recall} \tag{2.14}$$

The average (Macro) value of precision and F1-score for the proposed method is found to be 99.37% and 99.34% respectively.

Further, the proposed method is compared with CNN, SVM, and FCN-based classifiers, and it is observed that the performance of the proposed method outperforms all.

CNN architecture has 3 convolutional layers followed by dropout layers and two dense layers. The hyperparameter of CNN architecture is finalized by a random search algorithm with Keras tuner [114]. After making five trials in the keras tuner the parameter is finalized to be 128 neurons in the first hidden layer and 64 neurons in both the second and third CNN layer. The first dense layer has 48 neurons followed by 14 neurons in the last dense layer. Adam optimizer with a learning rate of 0.001 is selected. The kernel size

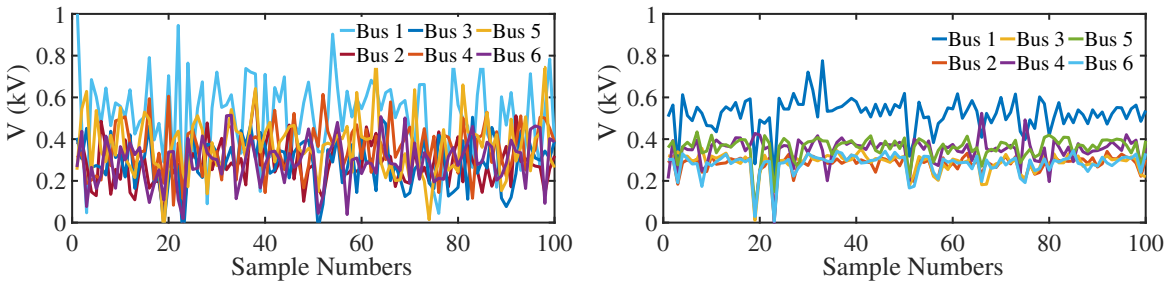
Table 2.3: Results of the proposed method for different cable faults in grid connected mode.

Fault Type	**Accuracy (%)**	**Recall (%)**
No Disturbance	96.61	96.61
Cable-1 Fault	100	100
Cable-2 Fault	97.92	97.92
Cable-3 Fault	100	100
Cable-4 Fault	100	100
Cable-5 Fault	100	100
Cable-6 Fault	98.11	98.11
Cable-7 Fault	100	100
Cable-8 Fault	100	100
PV Cable Fault	100	100
Addition of Load	100	100
Simultaneous addition of Load with DG	100	100
Removal of Load	97.87	97.87
Simultaneous removal of load and DG	100	100
Average	**99.32**	**99.32**

Table 2.4: Performance comparison of different methods for standard Fault conditions.

Methods	**Accuracy (%)**	**Precision (%)**	**F1 Score (%)**
FCN	96.61	96.74	96.62
SVM	90.74	89.53	89.50
CNN	97.92	97.96	97.93
GCN	99.32	99.37	99.34

is chosen to be 3. A polynomial function is selected as a kernel in SVM, and 5-fold cross-validation with grid search is used to optimize the parameter values. At last, γ is set to 1 and the value of C is found to be 100. At last, the fully connected neural network with three hidden layers with fully connected neurons is implemented to compare the efficacy



(a) (b)

Figure 2.10: Data with different SNRs (a) SNR = 10 dB (b) SNR = 25 dB.

of the proposed method. After continuous tuning and testing, the number of neurons in each of the three hidden layers is selected to be 80, 320, and 48 respectively.

2.4.2 Performance of the Proposed Method with Bad Data

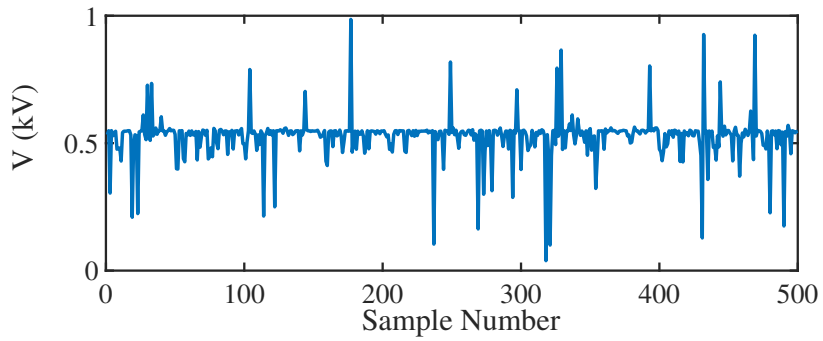
The performance of the proposed model is also analyzed with the corrupted data in the measurement. The measurement data are corrupted as follows:

1. Inaccurate measurement is modeled by randomly modifying the standard measurement data. The modification is done by multiplying 2 % of each standard measurement data sample with a random number ranging from 0.75 to 1.25.

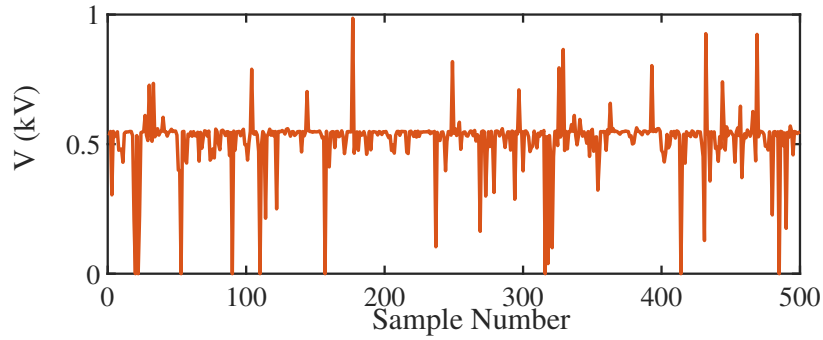
2. Secondly, the effect of data loss is tested by arbitrarily discarding the measurement data points. The number of samples loosed is set as 2 % of the total samples.

3. Further the performance of the proposed method is also against noise. The fault data samples are subjected to Gaussian noise with signal-to-noise ratio (SNR) of 10 dB, and 25 dB, as shown in Fig. 2.10, while the model's remaining parameters remain unchanged.

These bad data are added to the original samples which are further divided into training and testing data in the ratio of 7:3. The proposed method classification accuracy with bad data is depicted in Table 2.5. Figure 2.11 shows the voltage sample of bus 1 with and without the presence of bad data. It is obvious that after adding bad data the waveform of the fault data becomes more complicated. The average classification accuracy of the proposed method is still achieved as 97.53% as shown in Table 2.5.



(a)

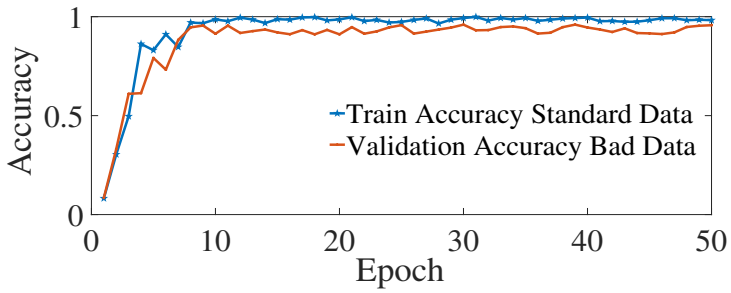


(b)

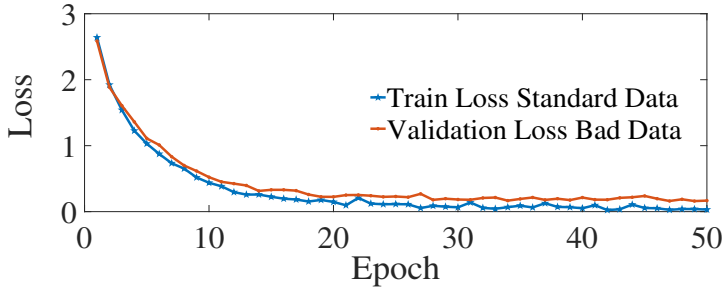
Figure 2.11: Samples of bus 1 voltage during fault, (a) standard data and (b) bad data).

Table 2.5: Accuracy (%) of the proposed method with standard and bad data.

Fault Type	**Accuracy (%)with Standard Data**	**Accuracy(%)with Bad Data**
No Disturbance	96.61	95.28
Cable-1 Fault	100	98.05
Cable-2 Fault	97.92	95.94
Cable-3 Fault	100	97.12
Cable-4 Fault	100	98.20
Cable-5 Fault	100	98.90
Cable-6 Fault	98.11	97.34
Cable-7 Fault	100	97.24
Cable-8 Fault	100	98.38
PV Cable Fault	100	98.18
Addition of Load	100	98.50
Simultaneous addition of Load with DG	100	98.70
Removal of Load	97.87	95.67
Simultaneous removal of load and DG	100	97.87
Average	**99.32**	**97.53**



(a)



(b)

Figure 2.12: Training and validation curve (a) accuracy (b) loss.

Table 2.6: Accuracy of different methods with the presence of noise.

Name of Methods	**Classification Accuracy (%)**	
	10 dB SNR	**25 dB SNR**
FCN	80.57	88.94
SVM	78.42	84.85
CNN	87.27	92.05
GCN	91.43	96.26

Figure 2.12 represents the accuracy and loss curve training and validation curve with standard data and bad data and show that the model performs better even with the presence of bad data. Table 2.6 shows the classification accuracies of different methods under the presence of noise. When the SNR is 40 dB, the proposed approach has an average accuracy of 98.69 percent. Hence the classification performance of the proposed methods is quite encouraging even in presence of noise with SNR above 25 dB.

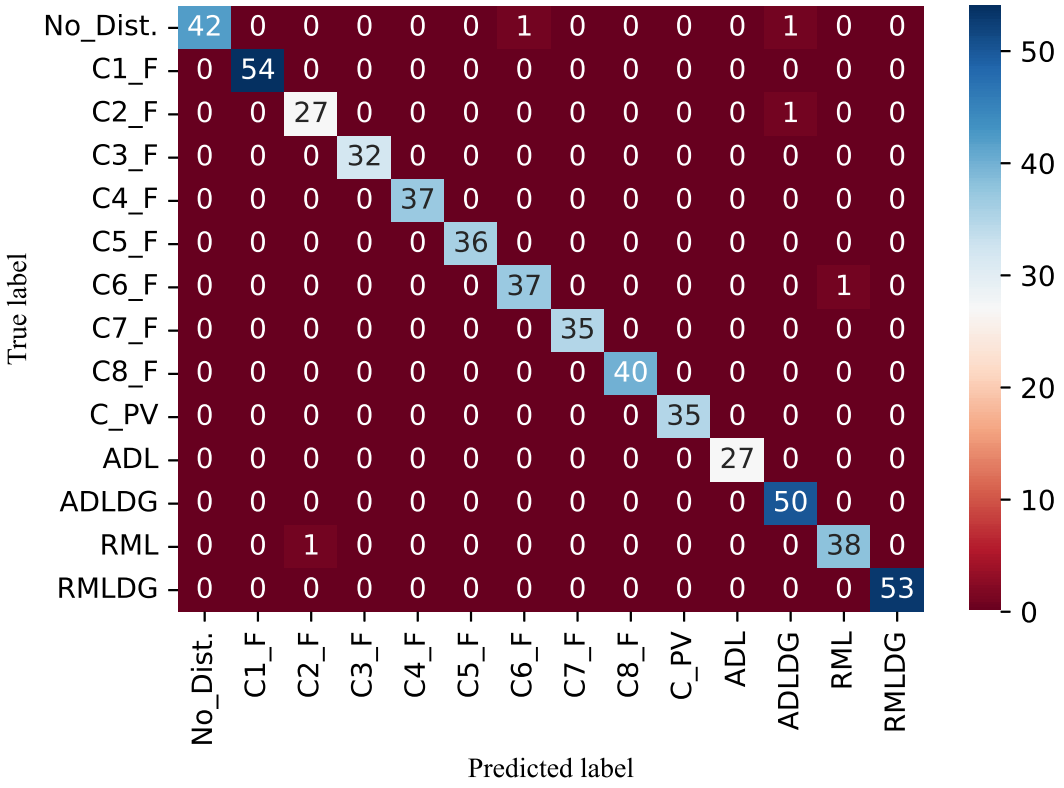


Figure 2.13: Confusion matrix for GCN-based classifier in the islanded mode.

2.4.3 Fault Identification in Islanded Mode

In addition to grid-connected mode, the proposed fault classification approach is also evaluated under islanded mode operation of the DC microgrid test system. Various fault scenarios are simulated to reflect realistic operating conditions in the absence of grid support. The resulting fault data samples undergo preprocessing to ensure consistency and suitability for input into the classification model. The classification accuracy for different types of faults and non-fault disturbances is summarized in Table 2.7. As illustrated in Fig. 2.13, the GCN-based fault detection method demonstrates strong performance even under islanded mode conditions, confirming its robustness and effectiveness across different operational modes of the DC microgrid.

2.5 Discussion

Although GCN offers several advantages for fault identification in DC microgrids, particularly due to its ability to model complex topologies and capture spatial dependen-

Table 2.7: Accuracy of the proposed method for different cable faults in islanded mode.

Fault Type	**Accuracy (%)**	**Total number of test cases**
No Disturbance	100	9
Cable-1 Fault	100	270
Cable-2 Fault	100	270
Cable-3 Fault	100	270
Cable-4 Fault	97.37	270
Cable-5 Fault	100	270
Cable-6 Fault	100	270
Cable-7 Fault	97.22	270
Cable-8 Fault	100	270
PV Cable Fault	97.22	270
Addition of Load	100	27
Simultaneous addition of Load with DG	98.04	81
Removal of Load	100	27
Simultaneous removal of load and DG	98.15	81
Average	**99.09**	**Total 2655 cases**

cies among interconnected components, its flexibility is somewhat limited. GCNs are well-suited for handling the irregular data structures typical of power networks and can process multi-source information, making them ideal for large-scale and distributed systems. However, a key limitation lies in the method's dependency on the adjacency matrix, which directly reflects the underlying grid topology. While the fault detection approach leverages spatiotemporal information from the power network to enhance classification performance, any changes in the system topology require a complete reformulation of the adjacency matrix and subsequent retraining of the model. This sensitivity to topology changes can hinder real-time adaptability and scalability in dynamic grid environments.

2.6 Summary

A GCN-based fault detection in DC microgrids is provided. Considering the electrical power network as an inherent graph, the proposed method utilizes spatial information from the test system to formulate the fault identification problem as node classification. First, the nodes and edges of the graph are defined in terms of the microgrid topology. After that, subsequent inclusion of the network topology is made so that the fault data samples

have both temporal and spatial information. The method provides better knowledge for the classification task and improves the classifier's performance. The fault data set is simulated considering various situations such as variations in temperature and irradiance, fault resistance, and fault distance. The experimental results show that the proposed method distinguishes different types of disturbances, including faults, with high precision. The proposed method is also tested with the existence of bad data and noise in fault data samples and shows better performance.

The proposed fault identification method supersedes the various machine learning-based algorithms and can be practically implemented. However, the incorporation of dynamic graph can further improve the performance of the proposed method in practical applications where frequent changes in network topology persist.