Fault Location in Power Distribution Systems via Graph Convolutional Networks*

Kunjin Chen*, Jun Hu*, Yu Zhang†, Zhanqing Yu*, Jinliang He*

*Tsinghua University, †UC Santa Cruz

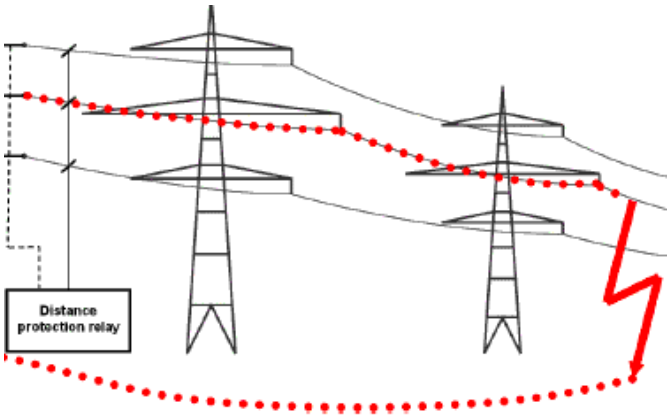
Funding ACK: National Key Research and Development Program of China Grant-2018YFB0904603, NSFC Grant-51720105004, State Grid Corporation of China Grant-5202011600UJ, Faculty Research Grant (FRG) of UC Santa Cruz, Hellman Fellowship

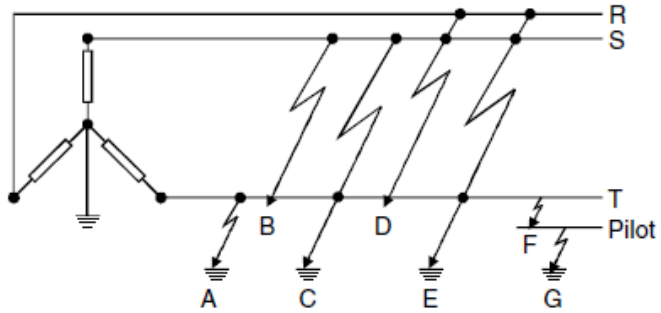
IEEE Journal on Selected Areas in Communications, vol. 38, no. 1, pp. 119–131, Jan. 2020.
https://ieeexplore.ieee.org/document/8892483

Overview

Fault Location in Distribution Systems

- Power distribution systems are constantly under the threat of short-circuit faults that would cause power outages.



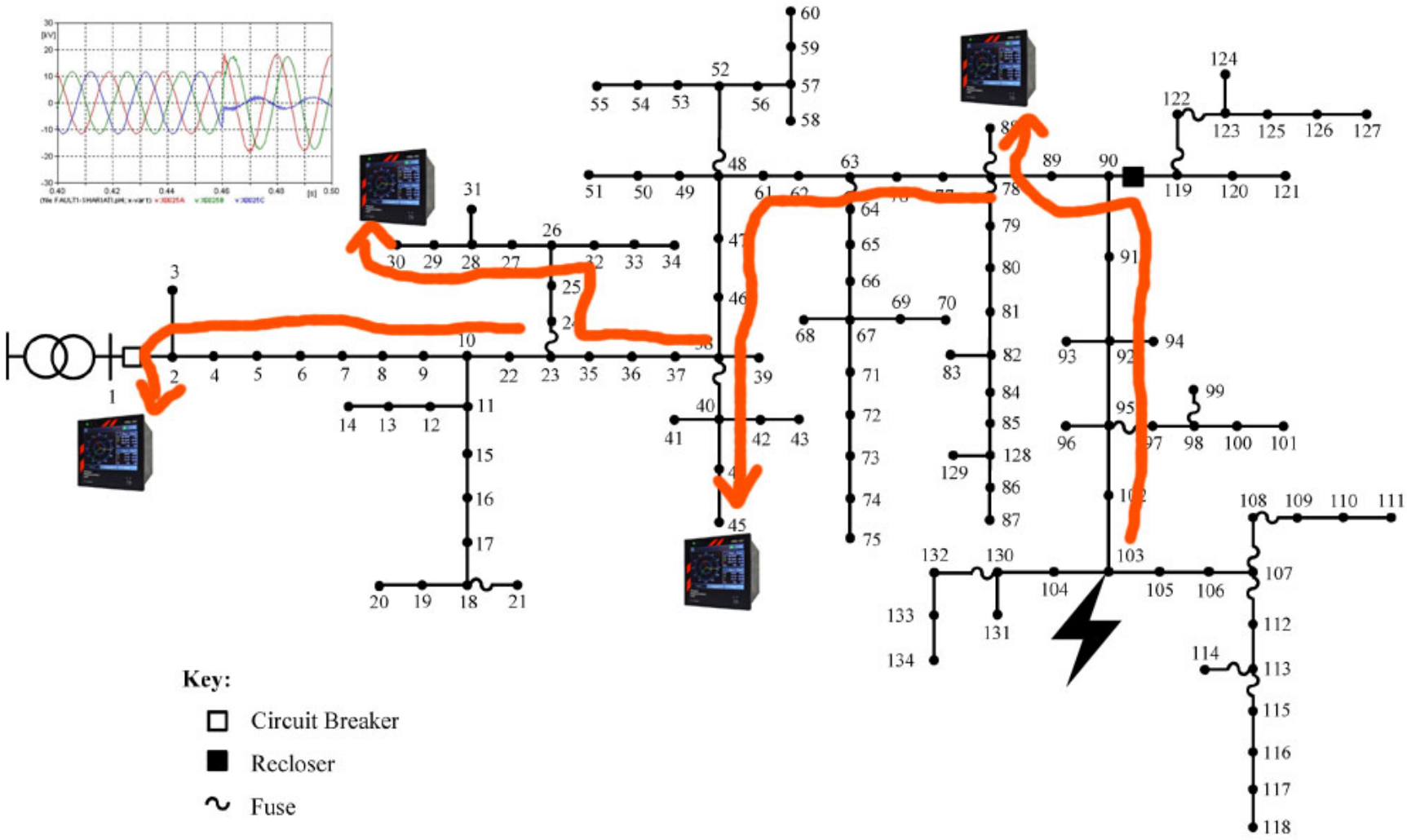


- System operators have to deal with outages timely to achieve high reliability.

- Goal: Accurately locate faults after the occurrence, so that quick restoration can be achieved.

Fault Location in Distribution Systems

Locating a fault in a distribution system:



Motivation

Existing methods:

- Methods based on impedance, voltage sag, traveling wave, and classical machine learning algorithms.
- They work fine theoretically, but are easily affected by noise, missing data, topology changes, etc.

The starting point of this work:

- Measuring phasors of voltage and current at lots of nodes becomes possible.
- Find a proper machine learning model that can use measurements from plenty of sensors to locate the fault in a distribution grid.

Problem Statement

We formulate the task of fault location as a classification problem.

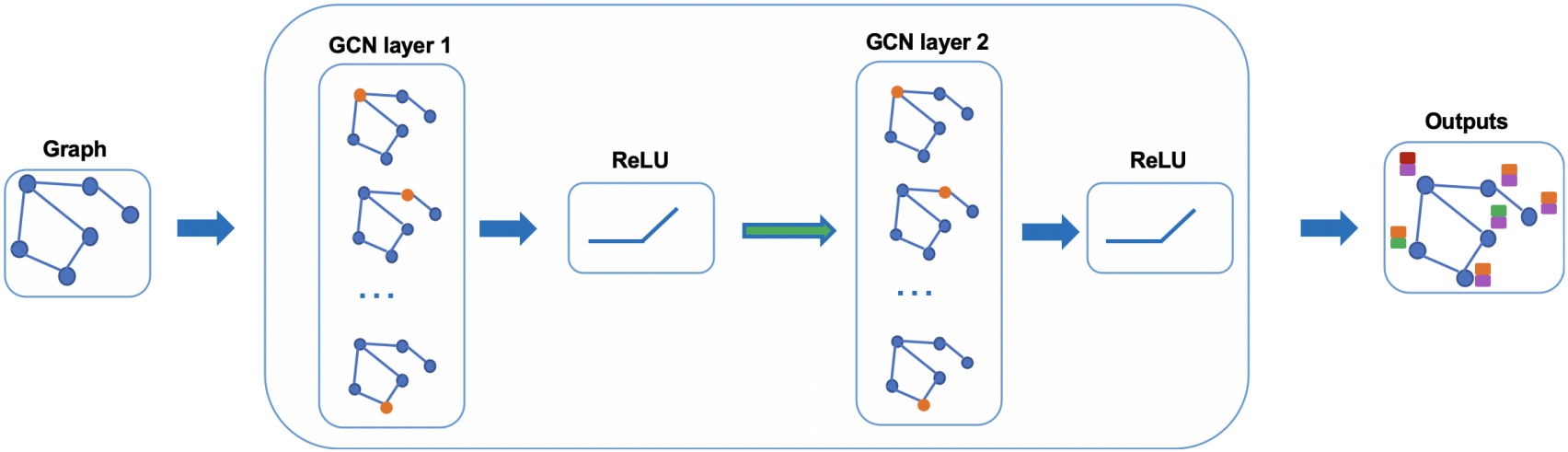
- For a given measured bus, assume that we have access to its three-phase voltage and current phasors:

$$(V_1, \theta_1^V, V_2, \theta_2^V, V_3, \theta_3^V, I_1, \theta_1^I, I_2, \theta_2^I, I_3, \theta_3^I) \in \mathbb{R}^{12}.$$

- A data sample of measurements: $\mathbf{X} \in \mathbb{R}^{n_o \times 12}$, where n_o is the number of observed buses. Values for unmeasured phases are set to zero.

- Given a data sample $\mathbf{X}_i$, the faulty bus $\tilde{y}_i = f(\mathbf{X}_i)$, where f is a classification model. A fault is correctly located if $\tilde{y}_i = y_i$, where y_i indicates the true faulty bus.

Graph Convolutional Networks (GCN)



- Graph is a natural representation of a power network: nodes for buses; edges for power lines.
- GCNs is a powerful predictive tool: leverages the information contained in the data and the relationships between data.
- In each GCN layer, the normalized graph structure is multiplied by the node properties and weights, and then passes through an activation function.

Figure: https://bit.ly/3h5IAz0

Spectral Graph Theory

- Consider an undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, the unnormalized graph Laplacian is $\mathbf{L}_u = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is the degree matrix and $\mathbf{W}$ is the weighted adjacency matrix.
- Normalized graph Laplacian is given as:

$$\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{L}_u\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2} \succeq \mathbf{0} \tag{1}$$

- Eigendecomposition: $\mathbf{L} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top$, where $\mathbf{\Phi} = (\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n)$ are orthonormal eigenvectors of $\mathbf{L}$, and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ collects ordered eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.
- $\lambda_1 = 0$ with the eigenvector $\boldsymbol{\phi}_1 = \frac{1}{\sqrt{n}}\mathbf{1}$.
- Algebraic multiplicity of $\lambda_1 = \dim(\mathrm{null}(\mathbf{L}))$ = the number of connected components of the graph.

Spectral Convolution on Graphs

- For a signal $\mathbf{f} \in \mathbb{R}^n$ on the vertices of graph $\mathcal{G}$ (a scalar for each vertex), the graph Fourier transform (GFT) is performed as $\hat{\mathbf{f}} = \mathbf{\Phi}^\top \mathbf{f}$ while the inverse GFT is $\mathbf{f} = \mathbf{\Phi}\hat{\mathbf{f}}$.
- The spectral convolution of two signals $\mathbf{g}$ and $\mathbf{f}$ is defined as

$$\mathbf{g} * \mathbf{f} := \underbrace{\mathbf{\Phi}}_{\text{IGFT}} \Big(\underbrace{(\mathbf{\Phi}^\top \mathbf{g})}_{\text{GFT}} \circ \underbrace{(\mathbf{\Phi}^\top \mathbf{f})}_{\text{GFT}} \Big) = \mathbf{\Phi}\, \mathrm{diag}(\hat{g}_1, \ldots, \hat{g}_n) \mathbf{\Phi}^\top \mathbf{f},$$

- Convolution of signal $\mathbf{f}$ with a filter $\mathbf{B} = \mathrm{diag}(\boldsymbol{\beta})$ is defined as:

$$\boxed{\mathbf{B} * \mathbf{f} = \mathbf{\Phi} \mathbf{B} \mathbf{\Phi}^\top \mathbf{f}.}$$

- The above filtering may not be spatially localized and is computationally expensive.
- Localized filters are able to extract features from small areas of interest instead of the whole input.

ChebNet

- Using filters that are smooth in spectral domain can bypass such an issue. e.g., polynomial filters represented in the Chebyshev basis, which stabilizes the training of the filters.

$$h_{\boldsymbol{\alpha}}(\tilde{\mathbf{\Lambda}}) = \sum_{k=0}^{K} \alpha_k \, T_k(\tilde{\mathbf{\Lambda}}), \tag{2}$$

  where $\{\alpha_k\}$ are learnable coefficients; eigenvalues of $\tilde{\mathbf{\Lambda}} = 2\lambda_n^{-1}\mathbf{\Lambda} - \mathbf{I}$ are rescaled frequency in the interval $[-1, 1]$.

- The Chebyshev polynomials are recursively defined as

$$T_k(x) = 2x\, T_{k-1}(x) - T_{k-2}(x), \tag{3}$$

  with $T_0 = 1,\ T_1 = x$.

ChebNet (Con't)

- Since $\mathbf{L}^k = \mathbf{\Phi}\mathbf{\Lambda}^k\mathbf{\Phi}^\top$, the filtering becomes

$$\mathbf{\Phi} h_{\boldsymbol{\alpha}}(\mathbf{\Lambda})\mathbf{\Phi}^\top \mathbf{f} = h_{\boldsymbol{\alpha}}(\mathbf{L})\mathbf{f} = \sum_{k=0}^{K} \alpha_k \, T_k(\tilde{\mathbf{L}})\mathbf{f},$$

  where $\tilde{\mathbf{L}} = 2\lambda_n^{-1}\mathbf{L} - \mathbf{I}$.

- Define $\mathbf{d}_0 = \mathbf{f}$ and $\mathbf{d}_1 = \tilde{\mathbf{L}}\mathbf{f}$, we have the recursive update

$$\mathbf{d}_k = 2\tilde{\mathbf{L}}\mathbf{d}_{k-1} - \mathbf{d}_{k-2}.$$

- Consider the filtering operation

$$\boxed{h_{\boldsymbol{\alpha}}(\mathbf{L})\mathbf{f} = [\mathbf{d}_0, \cdots, \mathbf{d}_K]\boldsymbol{\alpha}}$$

  It has a complexity of $\mathcal{O}(K|\mathcal{E}|)$ thanks to the sparsity of $\mathbf{L}$.

- Because of the Kth order truncation, the filter is K-hop localized w.r.t. the connections embodied in $\mathbf{L}$; i.e., it depends only on nodes that are at maximum K steps away from the central node.

The GCN Model

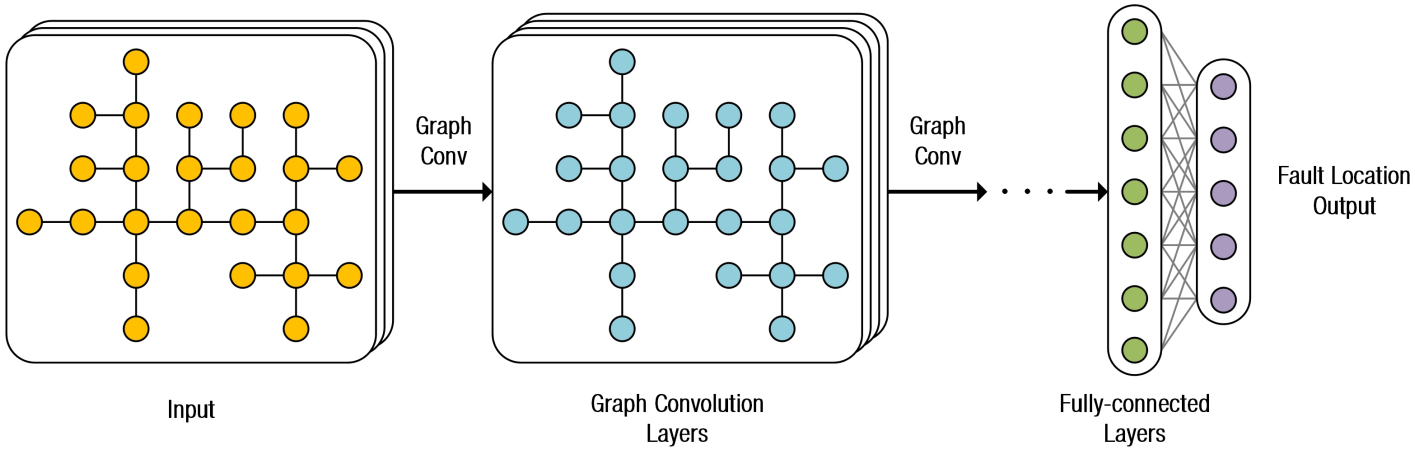


Figure: The input $\mathbf{X}$ passes through L_c graph convolution layers and L_f fully-connected layers followed by the softmax operator.

- The jth feature map of a graph convolution layer is

$$\mathbf{y}_j = \sum_{i=1}^{N_{in}} h_{\boldsymbol{\alpha}_{i,j}}(\mathbf{L})\mathbf{x}_i, \tag{4}$$

where $\mathbf{x}_i \in \mathbb{R}^n$ is the ith input feature map, $\boldsymbol{\alpha}_{i,j} \in \mathbb{R}^K$ contains the trainable coefficients, and N_{in} is the number of filters of the previous layer.

Calculation of the Graph Laplacian

- Find the distance matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$. The entry $\mathbf{S}_{ij}$ is the length of the shortest path between bus i and bus j.

- Ascending-sort (from left to right) and keep the smallest K_n values in each row of $\mathbf{S}$ to obtain $\tilde{\mathbf{S}} \in \mathbb{R}^{n \times K_n}$ and calculate the normalization factor $\sigma_S = \sum_i \tilde{\mathbf{S}}_{iK_n}/n$.

- Calculate $\tilde{\mathbf{W}}_{ij} = e^{-\tilde{\mathbf{S}}_{ij}^2/\sigma_S^2}$. Obtain the weighted adjacency matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ by restoring the positional correspondence of $\tilde{\mathbf{W}}_{ij}$ to bus i and bus j.

- Calculate the Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$.

The IEEE 123 Bus Test Case

- Three types of faults: single phase to ground, two phase to ground, and two phase short-circuit.
- Training and test data samples are simulated using OpenDSS.

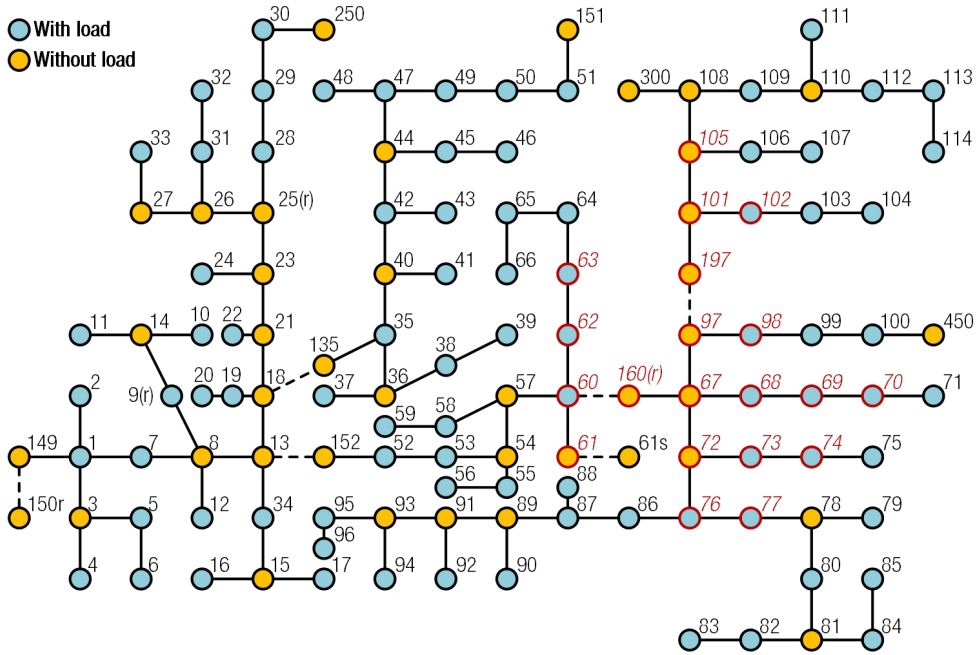
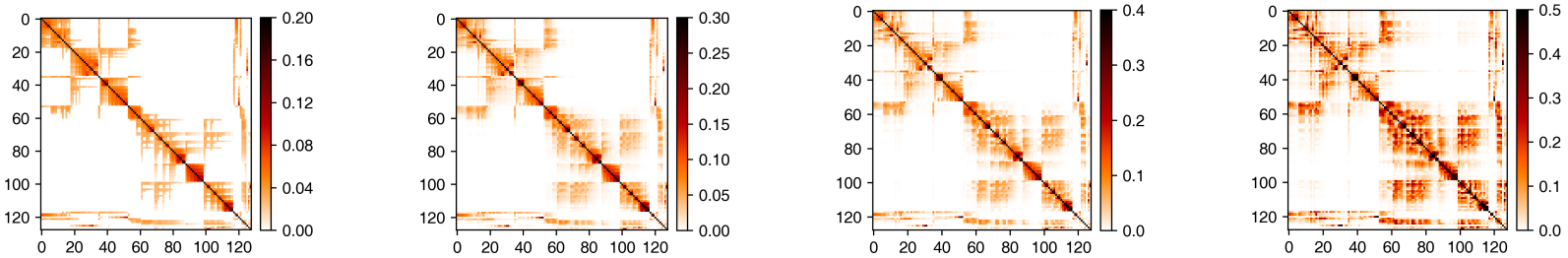


Figure: Dashed lines: normally closed switches; (r): with regulators. The red color numbers denote the 20 buses that are closest (in distance) to bus 67.

Implementation Details

- A total of 119 labels for the classification task.
- 20 data samples for each type of fault at each bus.
- 3 graph convolution layers, each with 256 filters.
- 2 fully-connected layers (512 & 256 hidden units), dropout rate $= 0.5$.
- $K_n = 20, K = [3, 4, 5]$.
- Adam optimizer, 400 epochs, mini-batch size $= 32$.

Visualization of $\mathbf{L}^m$: Locality of the Spectral Filters



(a) (b) (c) (d)

Figure: Visualization of the absolute values in $\mathbf{L}^m$ when $K_n = 20$, (a) $m = 1$, (b) $m = 3$, (c) $m = 5$, and (d) $m = 10$. When $m = 5$, the support of filters becomes the whole graph.

- The size of filters grows fast with the increase of m.
- Relatively large absolute values are mainly limited to entries corresponding to closer buses.
- Polynomials of $\mathbf{L}$ represent the filters. Higher-order terms facilitate the filters to explore more nodes.
- The locality of filters are ensured when choose K_n properly.

Classification Accuracy

3 types of data modifications are considered in the test set:

- Add Gaussian noise (SNR = 45 dB).
- Data loss of buses: randomly drop the data of N_{drop} buses (i.e., set the measured values to 0) per data sample.
- Random data loss for measured data: Each measurement at all buses is replaced by 0 with a probability P_{loss}.

TABLE II
FAULT LOCATION ACCURACIES OF THE MODELS UNDER VARIOUS MEASUREMENT MODIFICATIONS

Model	Noise (I)	Bus (II)	Random (III)	I + II	I + III	II + III	I + II + III
PCA + SVM	89.13 / 97.30	58.73 / 79.97	61.43 / 81.78	57.76 / 79.26	60.33 / 81.09	45.44 / 69.64	44.87 / 69.20
PCA + RF	85.94 / 96.77	53.82 / 67.62	58.57 / 74.07	52.15 / 66.66	56.94 / 73.23	40.55 / 55.84	40.05 / 55.64
FCNN	85.72 / 95.95	62.61 / 82.93	69.40 / 88.09	61.24 / 82.47	69.51 / 87.96	53.54 / 76.42	54.12 / 76.83
GCN	**97.10 / 99.72**	**92.67 / 97.44**	**89.09 / 96.67**	**90.76 / 97.83**	**87.70 / 96.31**	**83.55 / 94.51**	**80.63 / 93.86**

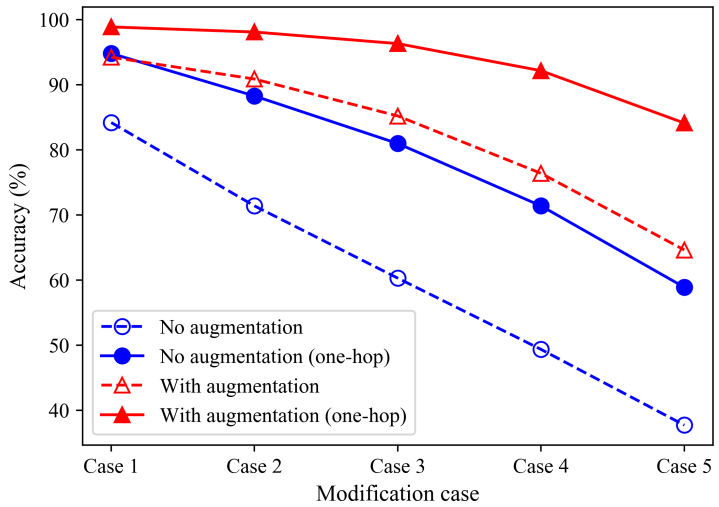
Classification Accuracy (Cont'd)

- Add Gaussian noise to training data:

TABLE III
FAULT LOCATION ACCURACIES OF THE MODELS UNDER VARIOUS MEASUREMENT MODIFICATIONS WHEN TRAINED WITH NOISY DATA

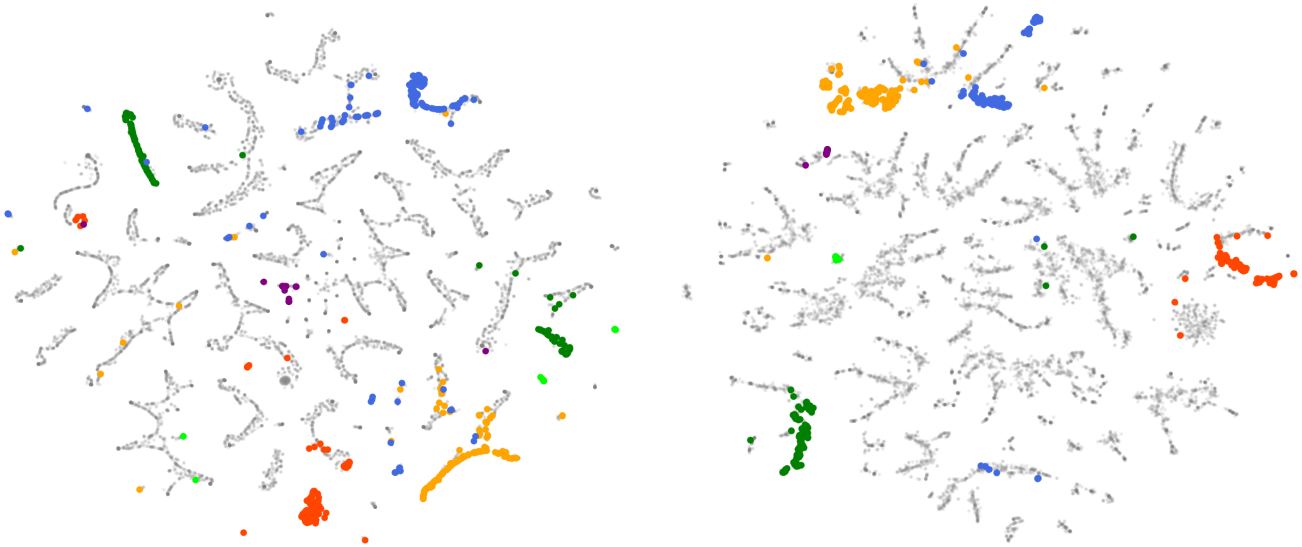
Model	Noise	Noise + Bus	Noise + Random	All Combined
PCA + SVM	85.70 / 96.21	55.98 / 77.74	58.00 / 80.17	44.12 / 68.51
PCA + RF	86.51 / 97.55	64.11 / 81.61	66.12 / 84.90	52.34 / 72.13
FCNN	86.95 / 97.19	61.95 / 82.58	70.32 / 88.52	53.98 / 76.55
GCN	**97.52 / 99.73**	**92.67 / 98.26**	**88.76 / 96.44**	**84.53 / 94.77**

- Add data augmentation to training data (applying random modifications to training data samples):



Visualization of Transformed Data

- Visualizing the test data processed by PCA and t-distributed stochastic neighbor embedding (t-SNE).



(a) (b)

Figure: Visualization of hidden features of test data added with all three types of modifications using t-SNE with two components: (a) the FCNN model, and (b) the GCN model.

Thank You!