MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Distribution system fault location analysis using graph neural network with node and link attributes

Sun, Hongbo; Kawano, Shunsuke; Nikovski, Daniel N.; Takano, Tomihiro; Mori, Kazuyuki

TR2021-130 October 21, 2021

Abstract

This paper presents a graph neural network based fault location method for distribution systems, in which both link attributes and node attributes are considered. The proposed method integrates multiple measurements at different buses with branch parameters at different branches as inputs of the GNN, and transforms fault locations on branches into output features of corresponding connected nodes for the faulted branch. Besides the system topology that can be naturally considered by the GNN, the branch parameters and related regulation and energization statuses are explicitly taken into account as link attributes. Numerical examples are provided to demonstrate the usage of the proposed method.

IEEE PES Innovative Smart Grid Technologies Conference - Europe (ISGT Europe) 2021

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Distribution Fault Location Using Graph Neural Network with Both Node and Link Attributes

Hongbo Sun
Data Analytics Group
Mitsubishi Electric Research Labs
Cambridge, MA 02139, USA
hongbosun@merl.com

Shunsuke Kawano
Advanced R&D Technology Center
Mitsubishi Electric Corporation
Amagasaki 661-8661, Japan
Kawano.Shunsuke@dn.MitsubishiElectric.co.jp

Daniel Nikovski
Data Analytics Group
Mitsubishi Electric Research Labs
Cambridge, MA 02139, USA
nikovski@merl.com

Tomihiro Takano
Advanced R&D Technology Center
Mitsubishi Electric Corporation
Amagasaki 661-8661, Japan
Takano.Tomihiro@df.MitsubishiElectric.co.jp

Kazuyuki Mori
Advanced R&D Technology Center
Mitsubishi Electric Corporation
Amagasaki 661-8661, Japan
mori.kazuyuki@ab.mitsubishielectric.co.jp

***Abstract*—This paper proposes a graph neural network (GNN) based approach to locate fault spots within the power distribution systems. The GNN model is extended from graph convolutional networks (GCNs), and includes several graph processing layers and followed by several full connected layers. The graph processing layers incorporated with node and link attributes are used to map system topology, bus measurements and branch parameters into hidden node embeddings, and full connected layers are used to relevant fault locations to node embeddings. The node attributes of the graph include measured phase voltage and current measurements, and branch impedance, admittance and regulation parameters are integrated into link attributes of the graph. The fault locations are represented as output features of nodes for the graph, in which only terminal buses of faulted branch have non-zero values corresponding to faulted phases. The developed approach is applicable to both short-circuit faults and ground faults. Numerical examples are given to demonstrate the effectiveness of the proposed method. The test results showed that the faulted section can be estimated with high percentage of accuracy, and more importantly there is no need for re-training in case of topology changes.**

Keywords— distribution systems, fault detection and location, graph neural network, ground fault, link attributes, node attributes, short-circuit fault

I. INTRODUCTION

Power distribution systems are constantly under the threat of ground and short circuit faults that would cause power outages. In order to enhance the operation quality and reliability of distribution systems, system operators have to deal with outages in a timely manner. Thus, it is of paramount importance to accurately locate and quickly clear faults immediately after the occurrence, so that quick restoration can be achieved.

Existing fault location techniques in the literature can be divided into several categories, namely, impedance-based methods[1]-[3], traveling wave-based methods[4]-[5], and machine learning-based methods[6]-[10]. Impedance-based fault location methods [1]-[3] use voltage and current measurements to estimate fault impedance and fault location. However, the accuracy of impedance-based methods can be affected by factors including fault type, unbalanced loads, heterogeneity of overhead lines, and measurement errors. Traveling wave-based methods[4]-[5] use observation of original and reflected waves generated by a fault. In general, however, traveling wave-based methods require high sampling rates and communication overhead of measurement devices. Machine learning models[6]-[10] are leveraged for fault location in distribution systems, such as artificial neural network, support vector and others. Recent advances in the field of machine learning, especially deep learning, have gained extensive attentions from both academia and industry, such as convolutional neural networks and graph convolutional networks. However, most of machine learning based approaches solely base on measurements to learn the relationships between fault location and measurements, and ignore the impacts of system topology and branch parameters and regulations on the fault behaviors.

In this paper, a graph neural network (GNN) based framework for fault location in distribution systems is proposed, in which both link attributes and node attributes are taken into account. The proposed method integrates multiple measurements at different buses while considering the impacts of system topology and branch parameters as well. The measurements at buses and impedance/admittance and regulation parameters on branches are modeled as node and link attributes in the GNN model, respectively.

Conventional graph convolutional networks (GCNs) can be used to achieve fault detection and location task by using its strong capability in aggregating local neighborhood information for individual graph nodes. Those GCNs can effectively leverage low-rank proximities and node features of the graph, however attributes that graph links may carry are commonly ignored, as almost all of these models simplify graph links into binary or scalar values describing node connectedness to identify neighborships and their influence if weighted in the local neighborhoods. For example, [10] proposed a GCN based approach to locate faults at buses using voltage and currents at buses, and the weighted adjacency matrix is constructed based on the physical distance between the nodes. However, in real-world scenarios, a link between a pair of nodes carries a lot more information than a simple indicator of neighborship. It represents a concrete relationship between two entities, usually with concrete attributes. For example, a line branch connected with two buses may have different impedance and admittances Therefore, revisiting these link attributes, which are generally ignored in current GCNs, allows us to recover the exact relationships between nodes.

This paper has mapped the fault detection and location problem into a non-linear regression problem and solved through an extended GCN model with both node and link attributes, that is a GCN-LASE model (i.e. a GCN with Link Attributes and Sampling Estimation) [11]. This extended GCN model takes both node and link attributes as inputs, in

which links are reverted to concrete relationships between entities with discretional attributes. The proposed method has been tested on a sample distribution system. The results showed high percentage of the method accuracy. The results also showed that there is no need for re-training in case of topology changes which commonly required by machine learning based fault location methods.

II. Formulation of Distribution Fault Location Task

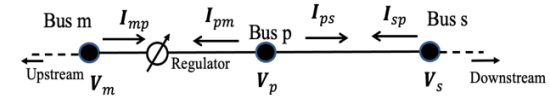
In this paper, the distribution system is modeled as a graph, the buses and branches of the distribution system is regarded as the nodes and links of the graph. The measurements at the buses are regarded as node features, and the parameters of the branches as link features. The fault locations are modeled as output features of nodes. Therefore, the fault location task can be formulated as a multiple non-linear regression problem. More specifically, given a matrix of sample node features $\boldsymbol{X}^{(s)}$, and a matrix of sample branch features $\boldsymbol{Y}^{(s)}$, the matrix/vector of sample fault location $\boldsymbol{Z}^{(s)}$, is obtained by $\boldsymbol{Z}^{(s)} = \boldsymbol{f}(\boldsymbol{X}^{(s)}, \boldsymbol{Y}^{(s)})$, where $\boldsymbol{f}$ is a specific faulty location regression model, s is the index for the sample fault event. The fault location vector $\boldsymbol{Z}^{(s)}$ defines the indications of faulty spot for all buses, in which only the terminal buses for fault branches are set with non-zero values on faulted phases in which the non-zero values are related to the distance between the fault spot and corresponding bus. A fault is correctly located if $\left|\widetilde{\boldsymbol{Z}}^{(s)} - \boldsymbol{Z}^{(s)}\right| \leq \varepsilon$, where $\boldsymbol{Z}^{(s)}$ indicates the true fault location, and $\widetilde{\boldsymbol{Z}}^{(s)}$ is the estimated fault location corresponding to $\boldsymbol{X}^{(s)}$ and $\boldsymbol{Y}^{(s)}$, and ε is given error tolerance.

Assumed that the voltage and current phasor measurements are available at measured buses, such as buses connected to loads and distributed generations. That is, for a given measured bus in a grounded distribution system, we have access to its three-phase voltage and current phasors ($\left|V_p^x\right|, \angle V_p^x, \left|I_p^x\right|, \angle I_p^x, x \in \{a,b,c\}$) $\in \mathbb{R}^{12}$,where $|\cdot|, \angle\cdot$ stands for an absolute value and a phase angle of a complex number, V_p^x and I_p^x denotes phase voltage and phase injection current at phase x of bus p. Values corresponding to unmeasured phases are set to zero. A data sample of measurements from the distribution system can then be represented as $\boldsymbol{X} \in \mathbb{R}^{n\times 12}$, where n is the number of buses. For a given measured bus in a ungrounded distribution system, we have access to its three-phase-to-phase voltage and current phasors, and zero-sequence voltage phasor, ($\left|V_p^{xy}\right|, \angle V_p^{xy}, xy \in \{ab,bc,ca\}; \left|V_p^0\right|, \angle V_p^0; \left|I_p^x\right|, \angle I_p^x, x \in \{a,b,c\}$) $\in \mathbb{R}^{14}$,where, V_p^{xy} denotes phase-to-phase voltage between phase x and y of bus p. A data sample of measurements from the ungrounded distribution system can then be represented as $\boldsymbol{X} \in \mathbb{R}^{n\times 14}$, where n is the number of buses to be considered.

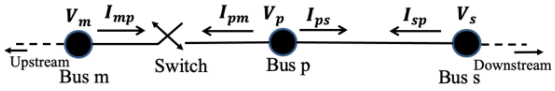
The distribution system has branches with various types, such as distribution line, transformer, breaker or switch, voltage regulator. In order to use a uniform data set representing various types of branches, we use the real and imaginary parts of equivalent nodal admittance matrix $\boldsymbol{Y}^{eqv}$ to represent branch features of graph neural network. For a branch, $\boldsymbol{Y}^{eqv}$ relates its branch currents flowing into two terminals to bus voltages at two terminal buses for the branch. The parameters for a given branch between bus p and bus s are related to $(Y_{pp}^{xy}, Y_{ps}^{xy}, Y_{sp}^{xy}, Y_{ss}^{xy}, xy \in \{aa,ab,ac,ba,bb,bc,ca,cb,cc\} \in \mathbb{R}^{72}$ and the branch parameters of the system can be represented as $\boldsymbol{Y} \in \mathbb{R}^{m\times 72}$, where m is the number of branches to be considered. If dividing into 4 sub-matrices, only diagonals of sub-matrices used, the number of branch parameters are reduced to 24 for each branch, $m \times 24$ for the system, and $\boldsymbol{Y} \in \mathbb{R}^{m\times 24}$.

The branches can be categorized into impedance-based branches, and zero-impedance branches. The impedance-based branches include distribution lines, and transformers. The equivalent nodal conductance and susceptance matrices is determined according to a series impedance matrix and a shunt admittance matrix for a distribution line branch, and transformer tap ratios, series impedances and winding connection for a transformer branch.

The zero-impedance branches include voltage regulators, feeder breakers, and switches, as shown in Fig. 1. For a zero-impedance branch between bus m and bus p, it is merged into a downstream distribution line between bus p and bus s as a new branch between bus m and bus s to be modeled [12]. Meanwhile, the measurements at bus p are ignored.



(A). Regulator branch



(B). Switch branch

Fig. 1. Modeling of zero-impedance branch

The equivalent nodal admittance matrix for the branch combined a regulator with a downstream distribution line is determined according to a set of regulation ratios of the regulator and a series impedance matrix and a shunt admittance matrix of the distribution line as shown in (1a) if voltage amplifying is given by phase voltages, and (1b) if amplifying is given by phase-to-phase voltages:

$$\boldsymbol{Y}^{eqv} = \begin{bmatrix} -\boldsymbol{A}_{I_{mp}}\boldsymbol{Y}_{pp}\boldsymbol{A}_{V_{pm}} & -\boldsymbol{A}_{I_{mp}}\boldsymbol{Y}_{ps} \\ \boldsymbol{Y}_{sp}\boldsymbol{A}_{V_{pm}} & \boldsymbol{Y}_{ss} \end{bmatrix} \quad (1a)$$

$$\boldsymbol{Y}^{eqv} = \begin{bmatrix} -\boldsymbol{A}_{I_{mp}}\boldsymbol{Y}_{pp}\boldsymbol{C}_V^{PL}\boldsymbol{A}_{V_{pm}}^{LL}\boldsymbol{C}_V^{LP} & -\boldsymbol{A}_{I_{mp}}\boldsymbol{Y}_{ps} \\ \boldsymbol{Y}_{sp}\boldsymbol{C}_V^{PL}\boldsymbol{A}_{V_{pm}}^{LL}\boldsymbol{C}_V^{LP} & \boldsymbol{Y}_{ss} \end{bmatrix} \quad (1b)$$

where $\boldsymbol{A}_{I_{mp}}$ is a current amplifying factor matrix for the regulator defined as $\boldsymbol{I}_{mp}=\boldsymbol{A}_{I_{mp}}\boldsymbol{I}_{pm}$, and $\boldsymbol{A}_{V_{pm}}$ is a voltage amplifying factor matrix for the regulator defined as $\boldsymbol{V}_p=\boldsymbol{A}_{V_{pm}}\boldsymbol{V}_m$. $\boldsymbol{A}_{V_{pm}}^{\boldsymbol{LL}}$ is a voltage amplifying matrices of the voltage regulator given in terms of phase-to-phase voltages. $\boldsymbol{I}_{mp}$ and $\boldsymbol{I}_{pm}$ are the phase currents entering through bus m and bus p, and $\boldsymbol{V}_m$ and $\boldsymbol{V}_p$ are the phase voltages of bus m and bus p, respectively. Those amplifying factor matrices are determined by the winding connection and tap positions for the voltage regulator. $\boldsymbol{Y}_{pp}$ and $\boldsymbol{Y}_{ss}$ are the self-admittance matrices at bus p and bus s for the line, and $\boldsymbol{Y}_{ps}$ and $\boldsymbol{Y}_{sp}$ are the mutual admittance matrices between bus p and bus s, respectively. $\boldsymbol{C}_V^{LP}$ is a voltage conversion factor matrix that converted phase-to-ground voltages into phase-to-phase voltages, $\boldsymbol{C}_V^{LP} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix}$. $\boldsymbol{C}_V^{PL}$ is a voltage conversion factor matrix that converted phase-to-phase voltages into phase-to-ground voltages, $\boldsymbol{C}_V^{PL} = \begin{bmatrix} 1/3 & 0 & -1/3 \\ -1/3 & 1/3 & 0 \\ 0 & -1/3 & 1/3 \end{bmatrix}$. For a switch/breaker between bus m and bus p connected with a line branch between bus p and bus s, the nodal admittance matrix

for the equivalent branch between bus m and bus s is calculated as:

$$\boldsymbol{Y}^{eqv} = \begin{bmatrix} \boldsymbol{S}_{mp}\boldsymbol{Y}_{pp} & \boldsymbol{S}_{mp}\boldsymbol{Y}_{ps} \\ \boldsymbol{S}_{mp}\boldsymbol{Y}_{sp} & (\boldsymbol{I}-\boldsymbol{S}_{mp})\boldsymbol{Y}_{sp}+\boldsymbol{Y}_{ss} \end{bmatrix} \quad (2)$$

where $\boldsymbol{S}_{mp}$ is a phase energized status matrix for the switch branch between bus m and bus p and represented by a diagonal matrix in which each element S_{mp}^{x} stands for the energized status for phase x, $x \in \{a, b, c\}$, and S_{mp}^{x} equals to 1 if energized, otherwise equals to zero.

The node features, and branch features are normalized before using for facilitating migrations to other systems with different topologies.

Fig. 2 illustrates representing fault locations using node output features for buses in the distribution system, in which each bus has one independent feature for each phase. Each bus has a row vector with dimensions of 1×3 to define the fault location related information, in which each phase of the bus has a corresponding element to describe whether there is a fault in this phase, and how far from this bus. Only terminal buses of a branch having a fault have non-zero output features residing at faulted phases of the buses. Any bus p has 3 output features one phase each, $\left(o_p^x, x \in \{a, b, c\}\right) \in \mathbb{R}^3$, and the output features of the system can be represented as $\boldsymbol{Z} \in \mathbb{R}^{n\times 3}$, where n is the number of buses to be considered. In Fig. 2, a fault occurs on the branch between bus p and bus s. The terminal buses of faulted branch, i.e., bus p and bus s, have non-zero elements in their output feature vectors. For all other buses, for examples, bus m and bus t, the values of output features are set as zero. For the terminal buses of faulted branch, only the elements corresponding to faulted phases are set with non-zero values in their output feature vectors. The magnitudes of output features for fault phases on the terminal buses of faulted branch are determined based on the relative distances from the fault spot to the terminal buses. For example, in Fig.2, a phase A to ground fault occurred at the line between bus p and s. d_p and d_s are the distances from the fault location to bus p and bus s, respectively. Therefore, the output features for bus p are set as $\left[\frac{d_s}{d_s+d_p} \quad 0 \quad 0\right]$,and the output features for bus s are $\left[\frac{d_p}{d_s+d_p} \quad 0 \quad 0\right]$. Only the output features corresponding to fault phase A have non-zero values.

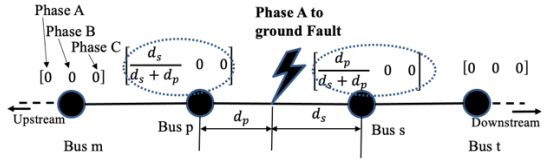


Fig. 2. Modeling of fault locations as node output features

For a known fault event, given the fault branch, fault location and fault phases, we can determine output features for all buses accordingly. Therefore, a full set of output features, and node features and branch features for the event can be obtained and served as a training sample for learning a relationship between output features and node and branch features.

For an unknown fault event, if we can estimate the output features with given node and branch features. Then we can determine the fault branch, fault location and fault phases accordingly. The fault branch is determined as a branch that has a maximum sum of non-zero output features of terminal buses. The fault phases are the phases of one of terminal buses of the faulted branch that have non-zero output features greater than a pre-determined threshold. The fault location is determined by using a ratio of distance from fault spot to the upstream bus over length of the branch. Taken a fault branch between bus p and bus s as an example, s_{ph} is the set of faulted phases, and bus p is the upstream bus. The ratio of distance between fault spot to upstream bus p over length of the branch, α_p is determined as:

$$\alpha_p = \frac{\sum_{x\in s_{ph}}(1-\hat{o}_p^x+\hat{o}_s^x)}{2\|s_{ph}\|} \quad (3)$$

wherein $\|\cdot\|$ is the cardinality of a set, $\hat{o}_p^x$ and $\hat{o}_s^x$ are the estimated output features corresponding to faulted phase x of bus p and bus s, respectively.

For a given distribution system, normal and faulty cases are simulated for each branch in the system to generate the training and test datasets used for training and evaluating the fault location models. The types of faults include single phase to ground fault, double phase to ground fault, phase to phase fault, and three phase to ground fault, and phase-to-phase-to-phase fault. The different fault locations for each branch, different fault resistances for each fault, and different load levels for the system are simulated. The voltage and current phasors are measured during the fault.

III. Graph Neural Network With Link Attributes

In this paper, the graph neural network (GNN) configured as shown in Fig. 3 is used to map the relationship between the fault locations with bus measurements and branch parameters of the distribution system. The GNN includes several graph processing layers followed by several fully-connected dense layers. The graph processing layers with combined node and link attributes are used to map system topology, bus measurements and branch parameters into hidden node embeddings, and full-connected dense layers are used to relevant fault locations to hidden node embeddings. The inputs $\boldsymbol{X}$ and $\boldsymbol{Y}$ is passed through $\boldsymbol{L_g}$ graph processing layers and $\boldsymbol{L_d}$ fully-connected dense layers followed by nonlinear activation functions (i.e. sigmoid functions). The weights and biases for graph processing layers and full-connected layers are optimized by minimizing a loss function defined as the Squared Error loss of output features. The Adam optimizer is used to train the model.

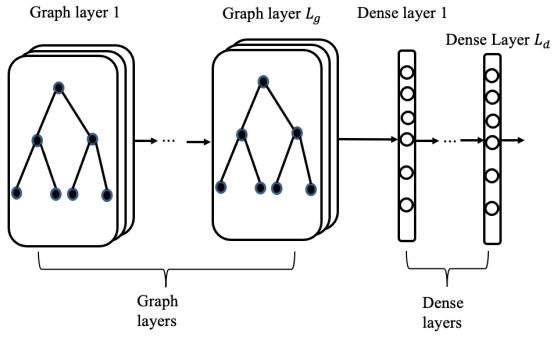


Fig. 3. The configuration for a graph neural network

The GNN model used here is an extended Graph Convolutional Network (GCN) model. The extended GCN [11] model used takes both node and link attributes as inputs. Specifically, each node has a set of attributes to describe its features using a row vector, and each branch also has a set of its own attributes to describe its features as a row vector. For adequately capturing the interactions between link and node attributes, the neighbor features of nodes can be defined as a tensor product of link attributes and corresponding node attributes.

Suppose an undirected weighted graph $G = (\boldsymbol{V}, \boldsymbol{E})$ is used to describe a distribution system, where $\boldsymbol{V}$ is the set of

nodes, $\boldsymbol{E}$ is the set of links. A neighbor can be described as an ordered pair, containing a neighbor node and the link connecting it to the central node, i.e. (node, link). In order to capture the interactions within a neighbor and adequately incorporate link attributes into node hidden representations, the associated neighbor feature is defined using their tensor product, instead of simply adding or concatenating node and link attributes together. Tensor product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is calculated as $\boldsymbol{a}\boldsymbol{b}^T$ with shape $d_a \times d_b$, and d_a and d_b are the lengths of $\boldsymbol{a}$ and $\boldsymbol{b}$. The calculated tensor contains all bilinear combinations of the two attributes, and serves as a fully conjoined feature. Formally, for the central node u connected to node v by a link $e_{u,v}$, the corresponding neighbor feature is defined as:

$$f\left(\left(v, e_{u,v}\right)\right) \coloneqq f(v) \otimes f\left(e_{u,v}\right) \tag{4}$$

where u and v are nodes in G, $e_{u,v}$ is a link from node u to node v. $\otimes$ stands for the operation of tensor product. $f(\cdot)$ is the feature of a node, a link or a pair, $\left(v, e_{u,v}\right)$ is a neighbor of node u, i,e, a pair of node v and link $e_{u,v}$.

Instead of directly using the tensor as inputs that leads to unacceptably high dimensionalities and heavy redundancies, graph kernels for so-defined neighbor features are used. For two neighbors, $\left(v, e_{\cdot,v}\right)$ and $\left(w, e_{\cdot,w}\right)$,. the neighbor kernel is defined as the inner product of the neighbor tensors, i.e.

$$\begin{aligned}\mathcal{K}\left(\left(v, e_{\cdot,v}\right), \left(w, e_{\cdot,w}\right)\right) &\coloneqq \langle f\left(\left(v, e_{\cdot,v}\right)\right), f\left(\left(w, e_{\cdot,w}\right)\right)\rangle \\ &= \langle f(v), f(w)\rangle \cdot \langle f\left(e_{\cdot,v}\right), f\left(e_{\cdot,w}\right)\rangle\end{aligned} \tag{5}$$

$\langle\cdot,\cdot\rangle$ stands for the operation of inner product. Based on the neighbor kernel, a kernel of two l-hop neighborhoods with central node u and u' can be defined as

$$\mathcal{K}_N^{(l)}(u,u') = \begin{cases} \langle f(u), f(u')\rangle & l = 0 \\ \langle f(u), f(u')\rangle \cdot \lambda \cdot \sum_{v\in N(u)} \sum_{v'\in N(u')} \mathcal{K}_N^{(l-1)}(v,v') \cdot \langle f\left(e_{u,v}\right), f\left(e_{u',v'}\right)\rangle & l > 0 \end{cases} \tag{6}$$

by regarding the lower-hop kernel, $\mathcal{K}_N^{(l-1)}(v,v')$, as the inner product of the $(l-1)$-th hidden representations of v and v'. $\lambda \in \{0,1\}$ is a decay factor. $N(u)$ is the set of neighbor nodes of u. By recursively applying the neighborhood kernel, a neural architecture for graphs with both node and link attributes, GCN-LASE (i.e. GCN with Link Attributes and Sampling Estimation) can be defined as a graph processing layer as Fig. 4.

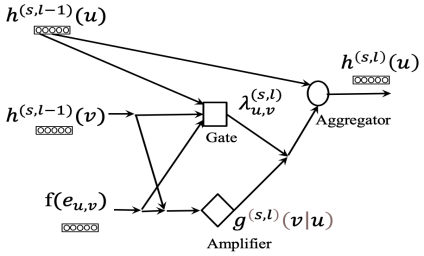


Fig. 4. The architecture for a GCN-LASE graph layer

Fig. 4 is a schematic illustrating an architecture of the graph processing layer included in the graph neural network. For layer l with corresponding sample s, the forward propagation calculations for the graph processing layer include three components, namely a gate, an amplifier, and an aggregator. The gate $\lambda_{u,v}^{(s,l)}$ evaluates v's influence in u's neighborhood. The amplifier $g^{(s,l)}(v|u)$ amplifies the node attributes using link information element-wisely. The aggregator $h^{(s,l)}(u)$ sums up neighbor embeddings and combines them with the central node embedding.

By uniting the depth and breadth convolution of nodes, and referring to the neighborhood aggregation concept in Graph-SAGE [13], a LASE-SAGE [11] architecture for the given graph processing layer l ($l = 1, \dots, L_g$), using sample s can be defined as:

$$\begin{aligned}\lambda_{u,v}^{(s,l)} = \sigma\Big(&\boldsymbol{W}_{G_{node}}^{(l)} h^{(s,l)}(u) + \boldsymbol{W}_{G_{link}}^{(l)} f^{(s)}\left(e_{u,v}\right) + \\ &\boldsymbol{W}_{G_{neighbor}}^{(l)} h^{(s,l)}(v) + b_G^{(l)}\Big)\end{aligned} \tag{7}$$

$$h^{(s,0)}(u) = f_{node}^{(s)}(u) \tag{8}$$

$$g^{(s,l)}(v|u) = h^{(s,l-1)}(v) \odot sigmoid\left(\boldsymbol{W}_{A_{link}}^{(l)} f_{link}^{(s)}\left(e_{u,v}\right)\right) \tag{9}$$

$$g^{(s,l)}\left(N(u)\right) = \textstyle\sum_{v\in N(u)} \lambda_{u,v}^{(s,l)} g^{(s,l)}(v|u) \tag{10}$$

$$\begin{aligned}h^{(s,l)}(u) = \sigma\Big(&\boldsymbol{W}_{A_{node}}^{(l)} h^{(s,l-1)}(u) \oplus \\ &\boldsymbol{W}_{A_{neighbor}}^{(l)} g^{(s,l)}\left(N(u)\right) + b_A^{(l)}\Big)\end{aligned} \tag{11}$$

where, $\odot$ is the operation of element-wise product, and $\oplus$ is the operation of concatenating input vectors. $\sigma(\cdot)$ is a nonlinear activation function. The action function is a sigmoid function. $h^{(s,l)}(u)$ is the hidden representation of node u in layer l. $\boldsymbol{W}_{G_{node}}^{(l)}$, $\boldsymbol{W}_{G_{link}}^{(l)}$, $\boldsymbol{W}_{G_{neighbor}}^{(l)}$ and $\boldsymbol{W}_{A_{node}}^{(l)}$, $\boldsymbol{W}_{A_{link}}^{(l)}$, $\boldsymbol{W}_{A_{neighbor}}^{(l)}$ are the weight parameters in the graph neural network. $b_G^{(l)}$ and $b_A^{(l)}$ are the biases in the network.

For each layer l, the above calculation is executed ($L_g - \mathrm{l} + 1$) times with different depth/hop for neighborhood. One hop represents a tracing from one link in the graph to the next.

The prediction layers are used to generate output features based on hidden node embeddings, and for the first layer, it takes the hidden node embeddings at last graph layer as input. The output of last prediction layer represents the faut location estimation results. The calculations for prediction layer k ($k = 1, \dots, L_d$) include:

$$o_u^{(s,0)} = h^{(s,L_g)}(u) \tag{12}$$

$$o_u^{(s,k)} = \sigma\left(\boldsymbol{W}_{P_{node}}^{(k)} o_u^{(s,k-1)} + b_P^{(k)}\right) \tag{13}$$

where $\boldsymbol{W}_{P_{node}}^{(k)}$ and $b_P^{(k)}$ are the weights and bias parameters of prediction layers.

Taken a graph neural network with two graph processing layers and one prediction layers as example, the dimensions of weights and biases for the first graph layer $\boldsymbol{W}_{G_{node}}^{(1)}$/ $\boldsymbol{W}_{A_{node}}^{(1)}$, $\boldsymbol{W}_{G_{link}}^{(1)}$/$\boldsymbol{W}_{A_{link}}^{(1)}$, $\boldsymbol{W}_{G_{neighbor}}^{(1)}$/$\boldsymbol{W}_{A_{neighbor}}^{(1)}$ are ($n_{out}^{(1)}$, m_{node}), ($n_{out}^{(1)}$, m_{link}), ($n_{out}^{(1)}$, m_{node}),and the dimension of $b_G^{(1)}/b_A^{(1)}$ is $n_{out}^{(1)}$, wherein $n_{out}^{(1)}$ is the pre-determined number of node embeddings for first graph processing layer, m_{node} and m_{link} are the numbers of node attributes and branch attributes respectively. The dimensions of weights and biases for the second graph layer $\boldsymbol{W}_{G_{node}}^{(2)}$/$\boldsymbol{W}_{A_{node}}^{(2)}$, $\boldsymbol{W}_{G_{link}}^{(2)}$/ $\boldsymbol{W}_{A_{link}}^{(2)}$, $\boldsymbol{W}_{G_{neighbor}}^{(2)}$/$\boldsymbol{W}_{A_{neighbor}}^{(2)}$ are ($n_{out}^{(2)}$, $2n_{out}^{(1)}$), ($n_{out}^{(2)}$, m_{link}), ($n_{out}^{(2)}$, $2n_{out}^{(1)}$),and the dimension of $b_G^{(2)}/b_A^{(2)}$ is $n_{out}^{(2)}$, and $n_{out}^{(2)}$ is the pre-determined number of node embeddings for second graph processing layer. The dimensions of weights and biases for the prediction layer $\boldsymbol{W}_{P_{node}}^{(1)}$ are (m_{out}, $2n_{out}^{(2)}$), and the dimension of $b_P^{(1)}$ is m_{out}, and m_{out} is the number of output features for fault location. The weights and biases are determined by minimizing a loss function to measure the differences between calculated outputs of last prediction layer and target outputs for a set of

training fault location scenarios. Wherein a squared error loss function is used when the fault location regression model is used:

$$\text{Loss} = \sum_{s=1}^{S} \sum_{u} \sum_{m=1}^{m_{out}} \left(o_{u,m}^{(s,L_d)} - \hat{o}_{u,m}^{(s,L_d)} \right)^2 \quad (14)$$

where S is the total number of sample fault events, $\hat{o}_{u,m}^{(s,L_d)}$ is the true value for m-th output features of node u for sample fault event s, $o_{u,m}^{(s,L_d)}$ is the prediction value for m-th output features of node u for sample fault event s.

Similar to conventional GCN, GCN-LASE also faces challenge for scalability, that is calculating the convolutions demands a recursively expanded neighborhood. For nodes with high degrees, it will quickly cover a large portion of the graph. To control batch scales, the Monte Carlo method is leveraged to estimate the summed neighborhood information by sampling a fixed number of neighbors. The summed neighborhood information is formulated as:

$$g^{(s,l)}\big(N(u)\big) = \sum_{v \in N(u)} \lambda_{u,v}^{(s,l)} g^{(s,l)}(v|u)\, \mathbb{E}_{p^{(s,l)}(\cdot|u)} \left[\frac{\lambda_{u,v}^{(s,l)} g^{(s,l)}(v|u)}{p^{(s,l)}(\cdot|u)} \right] \quad (15)$$

where $p^{(l)}(\cdot\,|u)$ denotes the sampling probabilities in $N(u)$. We then approximate $g^{(l)}\big(N(u)\big)$ through estimating the expectation. As the sampling process is always unbiased, we look for the optimal probabilities that minimize the estimation variance. According to the derivations of importance sampling, the sampling probabilities can be determined to minimize sampling variation as:

$$p^{(s,l)}(v|u) = \frac{\lambda_{u,v}^{(s,l)} \left\| g^{(s,l)}(v|u) \right\|_2}{\sum_{v \in N(u)} \lambda_{u,v}^{(s,l)} \left\| g^{(s,l)}(v|u) \right\|_2} \quad (16)$$

where $\| \cdot \|$ is the L2-norm of the vector.

Evaluating the sampling probabilities batch-wisely can be rather inefficient. Considered that the network parameters do not dramatically vary from batch to batch, a tradeoff can be made between variance and efficiency by controlling the interval of calculating the optimal distribution. That is, the sampling probabilities for all training nodes are calculated every e batches. Although the calculation may be time-consuming, the batch-averaged time cost will be reduced to $1/e$.

To make training time manageable, the set of nodes to be trained is divided into number of batches, and each batch has a fixed number of nodes. To reduce computation burden, a fixed number of neighbor samples is considered for each node by randomly chosen from all neighbors of the node under study.

Algorithm 1: Node sampling for graph processing layers

Input: Graph $G(V, E)$
Number of graph processing layer L_g
Minibatch for node, B
Neighborhood sampling function, $N^{(l,k)}(u)$
Output: Set of nodes for generating representation $B^{(l,k)}$
1: for $l = 1, \dots, L_g$ do
2: $B^{(l,0)} \leftarrow B$
3: for $k = 1, \dots, L_g - l$ do
4: for $u \in B^{(l,k-1)}$ do
5: $B^{(l,k)} \leftarrow B^{(l,k-1)} \cup N^{(l,k)}(u)$
6: end for
7: end for
8: end for

Algorithm 1 gives a procedure for sampling all nodes needed for each hop of each graph processing layer. Minibatch for node, B contains nodes that we want to generate representations for. $N^{(l,k)}$ denotes a deterministic function which specifies a random sample of a node's neighborhood with given number, and we index this function by l and k to denote the fact that the random samples are independent across iterations over l and k. Each set $B^{(l,k)}$ contains the nodes that are needed to compute the representations of nodes at layer l with search depth , k.

Algorithm 2 gives a procedure for minibatch forward propagation for each depth of each graph processing layer. At each search depth, nodes aggregate information from their local neighbors with weighted by link attributes, and as this process iterates, nodes incrementally gain more and more information from further reaches of the graph.

Algorithm 2: Forward propogation for graph processing layers

Input: Graph $G(V, E)$
Number of graph processing layer L_g
Minibatch for node, B
Neighborhood sampling function, $N^{(l,k)}(u)$
Set of nodes for generating representation $B^{(l,k)}$
Output: Representations for nodes $h^{(s,l)}(u),\ u \in B$
1: set $h^{(s,0)}(u)$ using (8)
2: for $s = 1, \dots, S$
3: for $l = 1, \dots, L_g$ do
4: for $k = 1, \dots, L_g - l$ do
5: for $u \in B^{(l,k-1)}$ do
6: generate a given number of neighbors of u, $N(u)$
7: calcuate $\lambda_{u,v}^{(s,l)}$ using (7), $v \in N(u)$
8: calculate $g^{(s,l)}(v|u)$ using (9), $v \in N(u)$
9: calculate $p^{(s,l)}(v|u)$ using (16), $v \in N(u)$
10 calculate $g^{(s,l)}\big(N(u)\big)$ using (15)
11: calculate $h^{(s,l)}(u)$ using (11)
12: end for
13: end for
14: end for
15: end for

IV. Numerical Examples

The proposed method has been tested on a sample distribution system as shown in Fig. 5. The sample system is an ungrounded system at 6.6 kV, and fed by one equivalent source from a substation. It has 12 buses, 1 transformers, 10 distribution lines, and 8 loads. The length of each line is given in the figure. The transformer windings use Delta/Delta connections, and the loads are Delta-connected constant PQ loads. An earthed voltage transformer (EVT) is connected to the secondary side of the transformer at substation. The system has been measured at buses 2-12, and then the system is modeled as a graph with 11 nodes and 10 links at most. The system has two switches, including a normally closed switch SW-1, and a normally open switch SW-2.

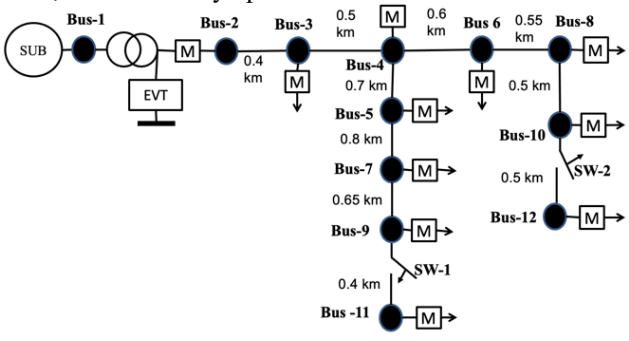


Fig. 5. A sample distribution system

The sample fault events are generated by choosing 5 fault spots on each distribution line and assigning each spot with different type of ground and short circuit types. The spots are uniformly located along the line, in which 4 of them are used

for training, and 1 of them s used for testing. The GNN is trained using the normal topology, i.e., SW-1 is closed and SW-2 is open.

Due to space limitation, only test results on one type of ground faults and one type of short circuit faults are given in this section. Table I lists the test results for phase A to ground faults and phase B to phase C short circuit faults. There are 4 different scenarios. Scenario I is a base scenario under normal topology and the GNN uses both node and line attributes. Scenario II also use the normal topology but the GNN uses node attributes only (link attributes are set as identity). Scenarios III and IV corresponds to the scenarios that the system has changed its topology. Switches SW-1 and SW-2 are closed in scenario III, and open in scenario IV. Moreover, the scenarios III and IV use the same GNN that trained for scenario I.

TABLE I. TEST RESULTS FOR ESTIMATED FAULT LOCATIONS ON PHASE A TO GROUND, AND PHASE B TO PHASE C FAULTS

Faulted Line	Faulted Phases	Test Scenario			
		I	II	III	IV
2-3	A	2-3	((4-5))	2-3	2-3
	BC	2-3	((((((7-9))))))	(3-4)	(3-4)
3-4	A	3-4	(4-6)	3-4	3-4
	BC	3-4	3-4	3-4	3-4
4-5	A	4-5	4-5	(3-4)	(((8-10)))
	BC	4-5	4-5	(5-7)	(5-7)
4-6	A	(3-4)	((((9-11))))	4-6	(4-5)
	BC	4-6	((((9-11))))	(((10-12)))	(6-8)
5-7	A	5-7	5-7	5-7	(4-5)
	BC	5-7	(7-9)	5-7	(7-9)
6-8	A	(4-6)	((3-4))	6-8	(8-10)
	BC	6-8	((4-5))	(8-10)	((4-5))
7-9	A	(5-7)	(9-11)	7-9	7-9
	BC	7-9	7-9	(9-11)	(5-7)
8-10	A	8-10	((4-6))	8-10	(6-8)
	BC	8-10	((((5-7))))	(6-8)	8-10
9-11	A	(7-9)	(((((((8-10)))))))	(((4-5)))	
	BC	9-11	(7-9)	9-11	
10-12	A			(8-10)	
	BC			10-12	
Estimation Accuracy (within 0-1 hops)		100%	50%	90%	88%

Table I gives the actual fault line and estimated fault line for each test case. If estimated fault lines are different than actual ones, the estimates will be given with pairs of parentheses, and the number of parentheses denotes the number of hops between the estimated and actual lines. For example, the estimates with single pair of parentheses indicates the estimated line is within 1-hop distance from actual line, that is the estimated line is directly connected to the actual faulted line. The related elements for branches not energized left in blank in the table.

As shown in Table I, using the proposed GNN model, the fault lines can be accurately estimated within 1 hop distance from the actual ones for 100%, 90% and 88% of test cases of scenarios I, III and IV, respectively. However, only 50% of test cases for scenario II are estimated with 1-hop accuracy.

Results listed in Table I showed that the proposed approach can be used to predict fault locations for both ground faults and short circuit faults. As indicated by scenario I, it can estimate the faulted section with 1-hop distance accuracy. Compared results for scenarios I and II, we can see the estimation accuracy is significantly improved by adding link attributes to the GNN besides node attributes. Moreover, as demonstrated by scenarios III and IV, the approach does not need re-training but still maintaining reasonable estimation accuracy for most cases when the topology of system is changed.

V. CONCLUSIONS

This paper has proposed a GNN based approach to locate fault spots within the distribution systems. The extended GCN model is used and configured with graph processing layers and full connected layers. The proposed method uses graph processing layers with node and link attributes to map system topology, bus measurements and branch parameters into hidden node embeddings, and full connected layers to relevant fault locations to node embeddings. The node attributes of the graph include measured phase voltage and current measurements. The link attributes of the graph integrate branch impedance, admittance and regulation parameters together. The fault locations are represented as output features of nodes. The test results showed that the developed approach can be used to predict fault locations for both ground faults and short circuit faults. It can accurately estimate the faulted section within 1-hop distance. More importantly, the approach does not need re-training while maintaining reasonable estimation accuracy for most of cases when the topology of system is changed.

Future work may be focused on effective link attribute selection, more efficient training algorithm, and graph network configuration optimization.

REFERENCES

[1] Y. Liao, “Generalized fault-location methods for overhead electric distribution systems,” IEEE Transactions on Power Delivery, vol. 26, no. 1, pp. 53–64, Jan. 2011.

[2] R. Krishnathevar and E. E. Ngu, “Generalized impedance-based fault location for distribution systems,” IEEE Transactions on Power Delivery, vol. 27, no. 1, pp. 449–451, Jan. 2012.

[3] S. Das, N. Karnik, and S. Santoso, “Distribution fault-locatingalgorithms using current only,” IEEE Transactions on Power Delivery, vol. 27, no. 3, pp. 1144–1153, July 2012.

[4] D. W. Thomas, R. J. Carvalho, and E. T. Pereira, “Fault location in distribution systems based on traveling waves,” in 2003 IEEE Bologna Power Tech Conference Proceedings, vol. 2, 2003, pp.242–246.

[5] S. Shi, A. Lei, X. He, S. Mirsaeidi, and X. Dong, “Travelling waves-based fault location scheme for feeders in power distribution network,” The Journal of Engineering, vol. 2018, no. 15,pp. 1326–1329, Oct. 2018.

[6] D. Thukaram, H. Khincha, and H. Vijaynarasimha, “Artificial neural network and support vector machine approach for locating faults in radial distribution systems,” IEEE Transactions on Power Delivery, vol. 20, no. 2, pp. 710–721, Apr. 2005.

[7] J. Mora-Florez, V. Barrera-Nunez, and G. Carrillo-Caicedo,“Fault location in power distribution systems using a learning algorithm for multivariable data analysis,” IEEE Transactions on Power Delivery, vol. 22, no. 3, pp. 1715–1721, July 2007.

[8] Y. Aslan and Y. E. Yagan, “Artificial neural-network-based fault location for power distribution lines using the frequency spectra of fault data,” Electrical Engineering, vol. 99, no. 1, pp. 301–311, Mar. 2017.

[9] Z. S. Hosseini, M. Mahoor, and A. Khodaei, “AMI-Enabled distribution network line outage identification via multi-label SVM,” IEEE Transactions on Smart Grid, vol. 9, no. 5, pp. 5470–5472, Sept. 2018.

[10] K. Chen, J. Hu, Y. Zhang, Z. Yu and J. He, "Fault Location in Power Distribution Systems via Deep Graph Convolutional Networks," in *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 1, pp. 119-131, Jan. 2020, doi: 10.1109/JSAC.2019.2951964.

[11] Z. Li , L. Zhang and G. Song,”GCN-LASE: Towards Adequately Incorporating Link Attributes in Graph Convolutional Networks”, Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19) , August 10-16, 2019, Macao, China.

[12] H. Sun, D. Nikovski, T. Takano, Y. Kojima and T. Ohno, "Line fault analysis of ungrounded distribution systems," 2013 North American Power Symposium (NAPS), Manhattan, KS, USA, 2013, pp. 1-6, doi: 10.1109/NAPS.2013.6666851.

[13] G. Nikolentzos, M. Vazirgiannis, “Random Walk Graph Neural Networks”, 34th Conference on Neural Information Processing Systems (NeurIPS 2020), December 6-12, 2020, Vancouver, Canada.