Fault Detection and Isolation in Industrial Networks using Graph Convolutional Neural Networks

Hamed Khorasgani*
Industrial AI Lab
Hitachi America, Ltd. R&D
Santa Clara, CA, USA

Arman Hasanzadeh*
Industrial AI Lab
Hitachi America, Ltd. R&D
Santa Clara, CA, USA

Ahmed Farahat
Industrial AI Lab
Hitachi America, Ltd. R&D
Santa Clara, CA, USA

Chetan Gupta
Industrial AI Lab
Hitachi America, Ltd. R&D
Santa Clara, CA, USA

***Abstract*—Industrial networks represent large-scale systems that consist of several interacting components. In this paper, we present a solution for Fault Detection and Isolation (FDI) in industrial networks with same type of components (electrical, mechanical, etc.) such as power grids, and water supply networks. Traditional FDI algorithms are trained to detect and isolate faults on the level of a single component by considering features from this component and sometimes from nearby components. These algorithms are sub-optimal as they are independently applied to individual components without explicitly taking into consideration the dependency between the several components that co-exist in industrial network. The interaction between the components makes fault isolation challenging. An operation change or a fault in a component can affect the neighboring components. This negatively impact the diagnosis accuracy when we design an independent diagnoser for each component. On the other hand, designing a global diagnoser without considering the network structure can lead to overfitting which can degrade the diagnosis performance significantly specially when the training data is limited. Moreover, because of the large number of components in these systems, the single fault assumption may not stand. This increases the complexity of the problem. In order to solve this problem, we first model the industrial network as a weighted undirected graph structure. The graph structure represents the connected components. The weights quantify these connections. We then apply Graph Convolutional Neural Networks (GCNN) to detect and isolate faulty components in these systems. In the case study, we apply our solution to a simulated industrial network with 100 components. The case study shows that GCNN outperforms several baseline algorithms.**

***Index Terms*—Industrial networks, Fault detection and isolation, Graph convolutional neural networks.**

I. Introduction

Traditionally, machine learning based methods for fault diagnosis use system's measurements as the features to detect and isolate faults in each component [1]. However, in industrial networks such as power grids, it is not possible to detect and isolate faults by purely monitoring individual components. For example, low voltage in an electric station can be because of a fault in the station or high demand from the neighboring stations. Therefore, in many cases, the relationship between the components measurements are far more informative compared to the individual component measurements for fault detection and isolation. To capture these relationships, model-based fault detection and isolation techniques use system equations to extract analytical redundancies between component measurements [2], [3].

Unfortunately, for complex industrial networks, the system models are not easy to develop, and they keep on updating during the system life-cycle. Therefore, reliable models of these systems are not always available. Even when models are available, they are often incomplete and plagued by uncertainties in tracking system behavior. This can lead to high false positive or high false negative rates in model-based diagnosis [4]. An alternative approach is to apply a data-driven solution that in addition to the system measurements can also use the relationship between the components for fault diagnosis.

Convolutional Neural Networks (CNN) have significantly improved image classification performance. Unlike classical neural networks such as Multilayer Perceptron (MLP), CNNs can extract local relationships between input data by using localized convolutional filters. Convolutional filters extract identical features in different spatial locations. These stationary filters have led to breakthroughs in image and video classification tasks [5], [6]. Similarly, extracting local relationships between industrial variables in different components can improve fault detection and isolation.

Like images, industrial networks often represent a structure. For example, in an electric power system, a station voltage may be independent from a remote load, but highly correlated to a nearby generator. Unfortunately, CNNs perform well only for regular data structures such as images and cannot achieve the same performance for domains with irregular structure such as industrial networks. Recently, Graph Convolutional Neural Networks (GCNN) have been introduced to generalize CNN to some irregular or more generally non-Euclidean domains such as chemical molecules, and social networks [7], [8].

In this paper, we present a new solution in which we formulate fault detection and isolation in industrial networks as a node node classification problem over undirected graph, and use GCNN to solve this problem. We will show through a case study that GCNN can improve fault detection and isolation performance by capturing the relationship between related variables in the components of the industrial network.

The rest of this paper is organized as follows. Section II

*Hamed Khorasgani and Arman Hasanzadeh contributed equally to this work. Arman Hasanzadeh is currently with the department of electrical and computer engineering at Texas A&M University. This work was completed during his internship at Hitachi America, Ltd. R&D.

978-1-5386-8357-6/19/$31.00 ©2019 IEEE

defines industrial networks and fault detection and isolation in these systems. Section III formulates the problem of fault detection and isolation in industrial networks as a graph node classification problem. Section IV presents our solution for this challenging problem. Section V presents the case study and Section VI presents the conclusions of the paper.

II. Industrial Networks

In this section, we formally define the fault detection and isolation problem for industrial networks with several components.

Definition 1 (Component): A component, $c = (c_\theta, c_X)$ is a unit in the industrial network characterized by a set of parameters, c_θ and a set of time series measurements (signals), c_X.

The parameters define the characteristic of the component, the time series measurements represent the current state of the component. For example, we can consider an electric load in an electric power network as a component. For this component, electrical resistance and electrical inductance of the load define the characteristic of the load, and therefore, represent component parameters. However, the load's electric power and electric current are time varying variables that change based on the component operating state. These variables when they are measured represent time series signals of the component.

Definition 2 (Industrial Network): An industrial network $S = (C, W)$ is defined as a set of components, and a set of weights where the connection between component $c_i \in C$ and components $c_j \in C$ are represented by $w_{ij} \in W$.

As an example, an electric power system is an industrial network combined from several components such as electric loads, electric generators, etc. Note that the definition of connected components can be different based on the application. Weights can be binary variables where $w_{ij} = 1$ when c_i and c_j are connected and $w_{ij} = 0$ otherwise. The weights also can represent the degree of connection, such as distance, and correlation between components.

Assumption 1 (Homogeneous Components): In this paper, we assume that the components in industrial network have the same type and the same number of measurements.

For example, for an electric grid, we assume all the components have the same type of instrumentations such as voltmeters, and ammeters. Note that we do not assume the components have the same parameters.

Assumption 2 (Static Network): In this paper, we assume that the number of components in the network and their connection (connection weights) do not change during the network operation.

Assumption 3 (Bidirectional Connection): In this paper, we assume that if component A is connected to component B, component B is also connected to component A.

Bidirectional connection between the components is typically the case in industrial networks.

Definition 3 (Fault): An unpermitted deviation of a component parameters from standard condition is referred to as a fault.

Fault can occur in each component. Faults can put the operators at risk, disrupt the manufacturing processes and cost industries millions of dollars. Fault detection and isolation (FDI) can lead to early response from the operators and fault tolerant control units to minimize the damage. We define FDI in industrial networks as follows.

Definition 4 (FDI in industrial networks): Fault detection determines the occurrence of a fault and the fault occurrence time in one or more components in the network. In the next step, fault isolation determines the faulty components in the system.

In the next section, we formulate FDI in industrial networks problem as a node classification problem over undirected graphs.

III. Problem Formulation

To detect and isolate a faulty component, it is not enough to monitor the component's measurements. An unexpected behavior of a time series measurement could be because of a fault in the component or a fault in a related component. For example, a change in the voltage level of an electric station could be a sign of a fault (for example, a short circuit) in the station or a high demand in the neighboring stations. Therefore, using only the local measurements for fault detection and isolation can decrease the detection rates and increase false alarms.

On the other hand, using all the system measurements without considering the connection between the components can lead to overfitting in the training step which can also result in low detection rates and high false alarm rates in the application step. To capture the relationship between the components and develop more accurate fault detection and isolation solution for networks with several components, we formulate the problem of fault detection and isolation in these systems as a node classification problem in undirected graphs. In our model, each node represents a component. Measurements associated with each component represent the data defined on the node associated with that component.

The graph structure encodes pairwise relationship between the components; 1) the edges show connection between components, 2) the weight of an edge show the degree of connection between components. In this work, we assume the graph structure (the graph's edges, and the weight of each edge) are known. In many practical cases, this information can be provided by domain experts. It is also possible to learn the graph structure using operating data [9], [10]. In this paper, we assume the connections between the components are bidirectional (see Assumption 3). Therefore, we model the relationship between the components using undirected graphs.

In graph theory, the number of hops refers to the minimum number of nodes we must pass to get to the destination. k-hop neighborhood of each node are the set of nodes that are reachable from the node in maximum k hops. 1-hop neighbor components represent the immediate neighbors of each component. Typically, it is intuitive to consider 1-hop

neighbor components for fault detection and isolation. However, in many cases it is not enough to analyze 1-hop neighbor measurements and we must consider higher order neighbors for accurate fault detection and isolation. Our goal is to use the time series measurements along with the graph structure to classify each node (component) as normal or faulty. In the next section, we use GCNN to solve this challenging problem.

IV. Proposed Methodology

Most of Machine Learning (ML) algorithms are developed to operate on a vector space. In vector spaces, we can use distance metrics such as Euclidian distance to quantify the dissimilarity between the data points. Graphs represent much more complex data structure. Traditionally, graph embedding methods have been used to transfer graph data to the vector space [11], [12]. For systems with a few nodes and limited connections between higher order neighbors, graph embedding can be a simple solution. However, in general, graph embedding leads to information loss, and extra computational complexity. A more efficient approach is to adapt ML algorithms to the graph domain. Convolutional Neural Networks (CNNs) [5] use convolutional filters to extract features from images. Similarly, Graph CNNs (GCNNs) [7] use Graph Fourier Transform (GFT) to extract features from the graphs. In this section, we first review GFT and GCNN. Then, we apply GCNN for fault detection and isolation in industrial networks.

A. Graph Fourier Transform

Let W be a weighted adjacency matrix, which is a matrix representation of a weighted graph. When there is an edge between node i and node j, $w_{ij} \in W$ represents the weight on the edge between i and j. When there is no edge between i and j, $w_{ij} = 0$. The degree matrix of a graph D, is a diagonal matrix where an element on the main diagonal, d_{ii} is the weighted sum of all the edges connected to node i. A diagonal matrix is a square matrix in which the elements outside the main diagonal are all zero. The difference between weighted adjacency matrix and degree matrix of a graph is called the graph Laplacian matrix, L.

$$L = D - W, \tag{1}$$

where L is a real symmetric matrix and, therefore, it has real eigenvalues and orthogonal eigenvectors.

GFT of signal X over a graph with Laplacian matrix, L, $Z = GFT(X)$, is defined based on eigen-decomposition of L. Eigen-decomposition represents a square matrix in terms of its eigenvectors and eigenvalues. Consider U as the matrix whose columns represent eigen-vectors of L. More specifically, L can be written as

$$L = U\Lambda U^T, \tag{2}$$

where Λ is a diagonal matrix of eigenvalues. Graph Fourier transform of X is defined as follows [13], [14],

$$Z = U^T X. \tag{3}$$

We can use Inverse-GFT to calculate the original signal, X, from its GFT, $Z = GFT(X)$, as:

$$X = UZ. \tag{4}$$

Note that in this work, X represents the matrix of component measurements and Z is the GFT of the component measurements over the graph which represents the industrial network.

B. Graph Convolutional Neural Network

GCNN applies GFT to transfer the signals to graph Fourier domain using the U matrix. It then applies a filter in the graph Fourier domain to extract graph features. A k-hop localized filter uses any pair of nodes with shortest path distance less than k to extract the features. Finally, it applies Inverse-GFT to transfer features to the time domain. The extracted features in each graph layer can be presented as

$$\text{Inverse-GFT}(g_{\boldsymbol{\theta}} Z) = U g_{\boldsymbol{\theta}}(\Lambda) U^T X, \tag{5}$$

where graph filter $g_{\boldsymbol{\theta}}(\Lambda) \in R^{N \times N}$ is a diagonal matrix with parameters $\boldsymbol{\theta}$.

$$g_{\boldsymbol{\theta}}(\Lambda) = \begin{bmatrix} g_{\theta_1}(\lambda_1) & 0 & 0 & \dots & 0 \\ 0 & g_{\theta_2}(\lambda_2) & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & g_{\theta_N}(\lambda_N) \end{bmatrix}, \tag{6}$$

where λ_i is the i eigenvalue of L. GCNN uses data to learn graph Fourier domain filter parameters. A set of graph filters generate a graph convolutional layer in GCNN. Fig. 1 represents a graph convolutional layer in GCNN.

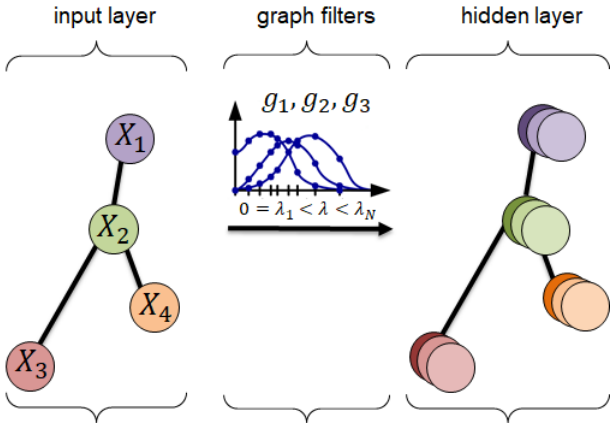

Fig. 1. A graph convolutional layer for a graph with 4 nodes (N=4). X_n represents graph signals in node n. g_l represents a filter to be learned. λ_i represents the i eigenvalue.

Matrix eigen-decomposition is computationally expensive. Chebyshev polynomial approximation can be used to extract graph features and learn the filter parameters without eigen-decomposition [7]. A Chebyshev approximation of a k-hop graph filter can be written as

$$X_f = \sum_{i=0}^{k} a_i T_i(\tilde{L}) X, \tag{7}$$

where $\tilde{L} = \frac{2}{\lambda_{max}} L - I_N$ with λ_{max} as maximum eigenvalue, $T_i(x) = 2xT_{i-1}(x) - T_{i-2}(x)$ with $T_1(x) = x$ and $T_0(x) = 1$, and X_f is the filtered graph signal, L is the graph Laplacian matrix and a_i for $i \in \{0 \dots k\}$ are trainable scalar coefficients. Using nonlinear activation functions, GCNN is capable of learning complex relationships between nodes signals.

C. Fault Diagnosis in Industrial Networks

In Section III, we formulated the fault detection and isolation problem in industrial networks as a node classification problem. Fig. 2 represents our approach for fault detection and isolation. X represents the signal measurements of component and is the input to the network. A k-hop graph convolutional layer extracts features using nodes with shortest path distance less than k. The network can have multiple layers for different k hop degrees to capture different levels of connections between the components. Finally, Y is the output of GCNN and represents the vector of node labels which can be normal or faulty.

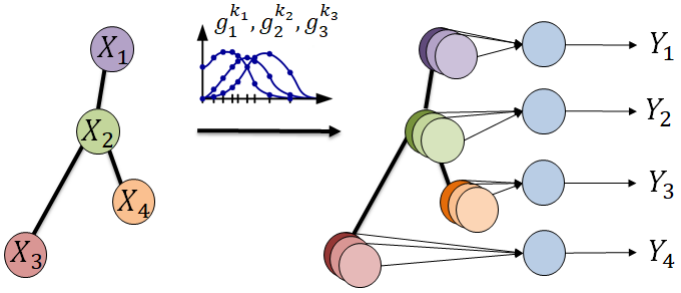

Fig. 2. A graph convolutional layer for a graph with 4 nodes (N=4). X_n represents graph signals in node n. g_l represents a filter to be learned. Y represents components labels (normal or faulty).

Graph convolutional filters capture the interactions between the components in normal and faulty operations. Learning these features, helps GCNN to distinguish normal interaction from fault propagations in the system. Moreover using the graph convolutional filter reduces the overfitting by restricting the model to learn relationship between k-hop connected components. Finally, unlike most FDI methods, our method does not make single fault assumption and can detect and isolate simultaneous faults in the system.

V. Case Study

We use a network of 100 connected water tanks to demonstrate the performance of our proposed method in fault detection and isolation in industrial networks.The water system network dataset is available at https://github.com/IndustrialNetwork/GraphDataset. Fig. 3 shows the structure of the network. We use a water distribution systems simulator to generate fault scenarios in this case study. The training dataset includes 50000 samples and the validation dataset has 10000 samples. The tanks in the network can be connected through pipelines. Fig. 4 shows a 5 connected tanks in the network. The flowrate between each two connected tanks is a nonlinear function of their pressures and distance between the tanks.

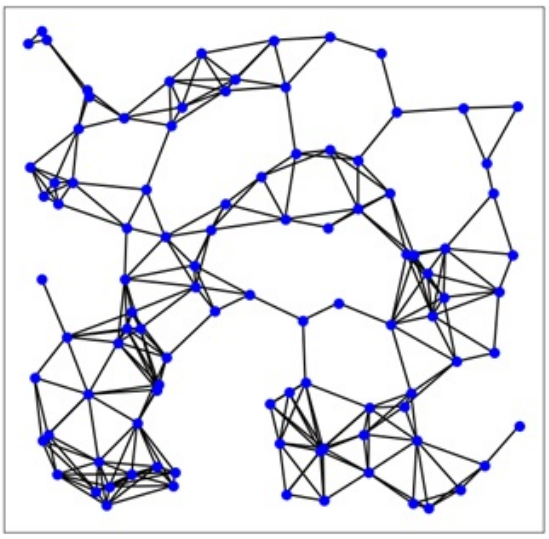

Fig. 3. Water tank network: an industrial network with 100 components.

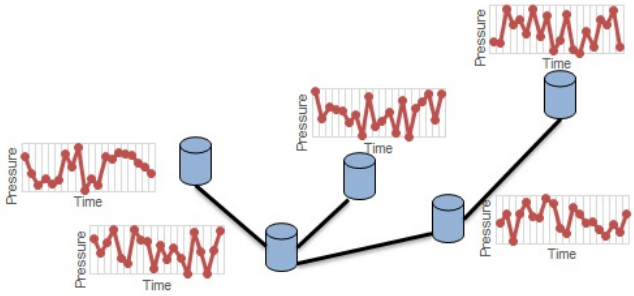

Fig. 4. A sample of 5 connected tanks in the network.

We consider each tank as a component in the network. The pressure in each tank is a measured signal in the component. When a tank's water level becomes less than a threshold, the tank switches to the refill mode. The refill mode continues untill the tank becomes full. A binary variable which represents the tank mode (refill or closed) is the second signal for each component. A fault in a tank represent a leakage. A leakage lasts for a random period before the operators identify and fix it. There is 5% noise in the measurements. When Tank i and Tank j are not connected by a pipe we consider $w_{ij} = 0$, and when Tank i and Tank k are connected by pipe we consider $w_{ik} = \frac{1}{\text{dis}_{ik}}$, where dis_{ik} is length of the pipe connecting Tank i to Tank k.

The network as shown in Fig. 3 represents a connected graph which means there is a path from any tank to any other tank in the network through the pipelines. This means a fault in a tank will affect every other tank in the network. Our goal is to design a fault diagnosis algorithm which detects leakages and isolates leaking tanks in the system. Accurate and timely

fault detection and isolation can reduce the leakage time and material waste. Typically, a leakage in a tank leads to loss of liquid and decrease in the pressure. However, a leakage is not the only factor in a tank pressure.

A decrease in pressure in the neighboring tanks can also increase the output flowrate and eventually decreases the tank pressure. Therefore, a tank pressure is also a function of leakage in other tanks. Fig. 5 represents pressure and leakage in a sample tank. The red lines represent leakages in the tank. On the vertical axis, 0 represents normal operation and 100 represents the leakage periods. The blue line represent pressure. As we can see the decrease in a tank pressure is not always because of a fault in the tank. This makes fault isolation problem very challenging. A fault in a tank can potentially change the pressure in all the tanks. Therefore, using only local measurements for fault detection and isolation can decrease the detection rates and increase false alarms.

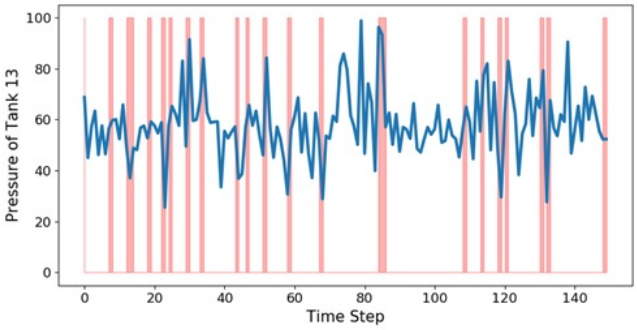

Fig. 5. Pressure (blue line) and fault (red line: normal =0, fault=100) in a tank.

Since a fault in a component can affect any other component, it may seem reasonable to use all the tanks signals for fault detection and isolation. However, in a system with several components, using all the components signals can result in overfitting, which can also result in low detection rates and high false alarm rates in the application step. To demonstrate this problem, we use a Fully Connected Neural Networks (NN) with 2 hidden layers and 500 nodes in each hidden layer to learn a fault diagnosis model which uses pressure measurements and refill status of all the tanks to detect and isolate the leaking tanks in the network. Fig.6 shows ROC curve for training and testing when we use a Fully Connected NN for fault detection and isolation in the case study. The area under the ROC curve (AUC) is a measure of how well the diagnosis algorithm can distinguish between normal and faulty components. As we can see from the curve, the training results (blue curve) is acceptable, however, because of overfitting the model does not perform well in the validation (red curve).

To capture the relationship between the components and develop more accurate fault detection and isolation solution for systems with several components, we formulate the problem as graph node classification problem and use a GCNN with 2 hidden layers (32 3-hop Chebyshev polynomial filters in the first hidden layer and 64 2-hop Chebyshev polynomial filters in the second hidden layer) to solve the problem. Fig.7 shows that GCNN can address the overfitting problem and perform equally well in the training and validation steps. Fig. 8 shows the diagnosis performance for Fully Connected with two different configurations, logistic regression, GCNNs with different number of layers and different localizations, and a local Support Vector Machine (SVM) classifier. TABLE I represents the parameters of each method. Note that the effect of overfitting is so significant to the extend that Univariate SVM which only uses component local measurements performs better than Fully Connected NNs.

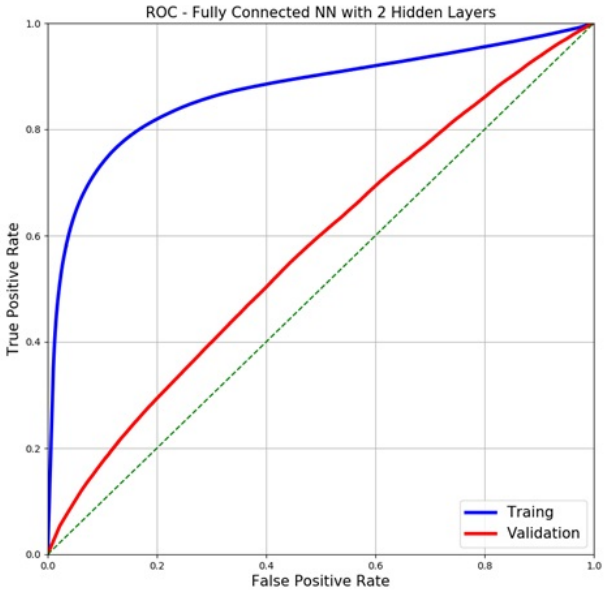

Fig. 6. ROC curve for Fully Connected Neural Networks (NN) with 2 hidden layers with tanh activation function in the hidden layers and sigmoid activation function in the output layer. Each hidden layer has 500 nodes and the output layer has 100 nodes. Input to the Fully Connected NN: 1- pressure of each tank, 2- refill status of each tank (200 input variables). Output of the Fully Connected NN: faulty tanks (100 variables).

The performance of GCNN can be improved by using different localizations. The proper level of localization depends on the network dynamic and degree of connection between k-hop neighborhoods. In general, selecting a lower value for k can prevent the network from learning higher order interactions in the graph. On the other hand, selecting a higher value for k can lead to overfitting. In this work, we adapt an ad hoc approach for selecting the optimal k value. We start with 1-hope localization and increase the k value until there is no improvement in the diagnosis performance. As shown in Fig. 8, 2-hop localization has much better performance than 1-hop localization in our water system network. However, compared to 2-hop localization, 3-hop localizations doe not significantly improve the diagnosis results.

TABLE I
NETWORK PARAMETERS

Baseline Method	Parameters		
	First Layer	*Second Layer*	*Output Layer*[a]
Fully Connected NN-1 Hidden Layer	500 nodes	-	100 nodes
Fully Connected NN-2 Hidden Layers	500 nodes	500 nodes	100 nodes
Multivariate Logistic Regression	-	-	100 nodes
GCNN-1 Hidden Layer 3-hop filter	32 3 hop filters	-	100 nodes
GCNN-1 Hidden Layer 1-hop filter	32 1 hop filters	-	100 nodes
GCNN-2 Hidden Layers 2/2-hop filter	32 2 hop filters	64 2 hop filters	100 nodes
GCNN-2 Hidden Layers 3/2-hop filter	32 3 hop filters	64 2 hop filters	100 nodes

[a]Last layer is fully connected.

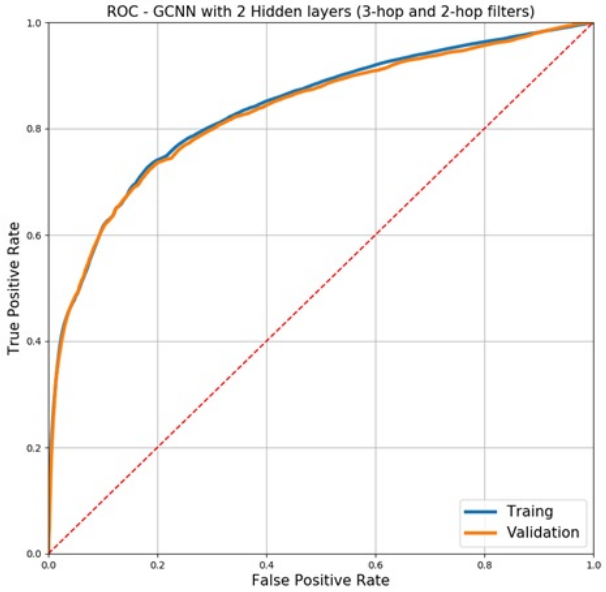

Fig. 7. ROC curve for GCNN with 2 hidden layers. The first hidden layer has 32 3-hop Chebyshev polynomial filters and the second hidden layer has 64 2-hop Chebyshev polynomial filters. The nonlinear activation function for each layer is relu. Input to the GCNN: 1- pressure of each tank , 2- refill status of each tank (200 variables). Output of the GCNN: faulty tanks (100 variables). The output layer is fully connected.

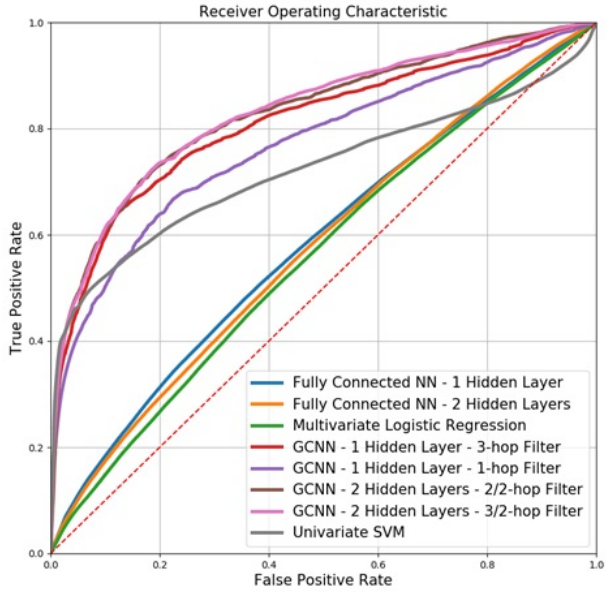

Fig. 8. Comparison of Fully Connected NN-1 Hidden Layer with 500 nodes in the hidden layer, Fully Connected NN-2 Hidden Layers with 500 nodes in each hidden layer, Multivariate logistic regression, GCNN-1 Hidden Layer with 32 3 hop Chebyshev polynomial filters, GCNN-1 Hidden Layers with 32 1 hop Chebyshev polynomial filters, GCNN-2 Hidden Layers with 32 2 hop Chebyshev polynomial filters in the first layer and 64 2 hop Chebyshev polynomial filters in the second layer, GCNN-2 Hidden Layers with 32 3 hop Chebyshev polynomial filters in the first layer and 64 2 hop Chebyshev polynomial filters in the second layer.

VI. CONCLUSIONS

In this paper, we developed a data-driven method to address FDI in large-scale industrial networks with several interacting components. Our solution improves fault detection and isolation by taking into account the connection between the components without using the system physical model. The connection can be physical connection such as pipelines and electric lines or can represent correlations between the components. Our solution does not make the single fault assumption which can be limiting for large-scale systems. The case study showed our method significantly outperforms several baseline solutions.

In future work, we will improve FDI performance by analyzing highly correlated components in industrial networks. The correlated components in the network can generate several communities in the network. The health condition of the components in each community are highly related to each other and a fault in each component can affect the entire community significantly. First, we investigate different solutions to detect communities in industrial networks. Then, we can apply a community-varying GCNNs [8], which learns a set of filters for each community to develop a community-varying diagnosis solution.

REFERENCES

[1] Hamed Khorasgani, Ahmed Farahat, Kosta Ristovski, Chetan Gupta, and Gautam Biswas. "A Framework for Unifying Model-based and Data-driven Fault Diagnosis." In PHM Society Conference, vol. 10, no. 1. 2018.

[2] Ragot, Jos, and Didier Maquin. "Fault measurement detection in an urban water supply network." Journal of Process Control 16, no. 9 (2006): 887-902.

[3] Ferrari, Riccardo MG, Thomas Parisini, and Marios M. Polycarpou. "Distributed fault detection and isolation of large-scale discrete-time nonlinear systems: An adaptive approximation approach." IEEE Transactions on Automatic Control 57, no. 2 (2012): 275-290.

[4] Hamed Khorasgani. "Model-and data-driven approaches to fault detection and isolation in complex systems." PhD diss., 2017.

[5] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." In Advances in neural information processing systems, pp. 1097-1105. 2012.

[6] Karpathy, Andrej, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. "Large-scale video classification with convolutional neural networks." In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 1725-1732. 2014.

[7] Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In Advances in Neural Information Processing Systems, 3837-3845.

[8] Zhou, Jie, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. "Graph Neural Networks: A Review of Methods and Applications."arXiv preprint arXiv:1812.08434(2018).

[9] Dong, Xiaowen, Dorina Thanou, Michael Rabbat, and Pascal Frossard. "Learning Graphs from Data: A Signal Representation Perspective." arXiv preprint arXiv:1806.00848 (2018).

[10] Dong, Xiaowen, Dorina Thanou, Pascal Frossard, and Pierre Vandergheynst. "Learning Laplacian matrix in smooth graph signal representations." IEEE Transactions on Signal Processing 64, no. 23 (2016): 6160-6173.

[11] Brand, Matthew. "Continuous nonlinear dimensionality reduction by kernel eigenmaps." In IJCAI, pp. 547-554. 2003.

[12] He, Xiaofei, Ming Ji, and Hujun Bao. "Graph Embedding with Constraints." In IJCAI, vol. 9, pp. 1065-1070. 2009.

[13] Sandryhaila, Aliaksei, and Jose MF Moura. "Discrete signal processing on graphs." IEEE transactions on signal processing 61.7 (2013): 1644-1656.

[14] Shuman, David I., et al. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains." IEEE Signal Processing Magazine 30.3 (2013): 83-98.