Graph Neural Network based Root Cause Analysis Using Multivariate Time-series KPIs for Wireless Networks

Chia-Cheng Yen,† Wenting Sun,*† Hakimeh Purmehdi,† Won Park,†
Kunal Rajan Deshmukh,† Nishank Thakrar,† Omar Nassef,‡ Adam Jacobs†
†Ericsson, Inc, Santa Clara, CA, USA
Email: {jerry.yen, wenting.sun, hakimeh.purmehdi, won.park,
kunal.rajan.deshmukh, nishank.thakrar, adam.jacobs}@ericsson.com
‡Department of Engineering, King's College London, London, UK
Email: omar.nassef@kcl.ac.uk

***Abstract*—Due to the rapid adoption of 5G networks and the increasing number of devices and base stations (gNBs) connected to it, manually identifying malfunctioning machine or devices that causes other part of the networks to fail also becomes more challenging. Furthermore, data collected from the networks are not always sufficient. To overcome these two issues, we proposed a novel root cause analysis (RCA) framework that integrates graph neural networks (GNNs) with graph structure learning (GSL) to infer hidden dependencies from available data. The learned dependencies are the graph structure utilized to predict the root cause machine or devices. We found that despite the fact that the data is often incomplete, the GSL model can infer fairly accurate hidden dependencies from data with a large number of nodes and generate informative graph representation for GNNs to identify the root cause. Our experimental results showed that higher accuracy of identifying a root cause and victim nodes can be achieved when the number of nodes in an environment is increased.**

***Index Terms*—5G networks, anomaly detection, fault localization, root cause analysis, graph neural networks**

I. Introduction

5G NR significantly improves the flexibility by enabling many sub-carrier spacing (SCS) configurations which can support a variety of use-cases (e.g., massive IoT, MBSFN, eMBB, and uRLLC). According to Cisco annual Internet report [1], by 2023, the overall global mobile subscribers will increase to 5.7 billion (over 70% of human population) and the 5G capable mobile devices will grow to 1.4 billion (over 10% of all global mobile devices). Due to this development, it is expected that the cost and complexity of network operations and management will increase in the near future. To increase fault tolerance and reduce human intervention, involving AI-enabled root cause analysis (RCA) to automatically detect faults, trace a chain of failures and then, identify potential issues and possibly fix them automatically has become essential for the next generation of networks.

*Corresponding Author

Well-formulated network key performance indicators (KPIs) can be used as benchmarks by which optimal network performance can be determined. Tracking performance against KPIs helps telecommunication operators make proactive decisions to ensure agreed service levels are met. KPIs also provide quantifiable measures against which fact-based decisions around infrastructure investment, performance and demand can be made. KPI data can also be used for RF planning, radio channel measurements and modelling, feasibility studies and formulation of appropriate regulatory policies for wireless communication systems [2]. Unfortunately, there have not been many effective approaches to explore root causes with KPI data Moreover, we found that data managed by operators are usually in aggregated format which provides insufficient knowledge and leads to missing information to represent the network thoroughly. This issue makes inferring useful insight difficult. For example, key contributing factors such as relationship dependencies among individual nodes are usually missing in the data.

To address the above two challenges, we propose a graph neural networks (GNNs) based RCA framework leveraging KPI data and graph structure learning (GSL) model to recognize dependencies and potential root causes. In each iteration, every node aggregates data from its neighboring nodes and also propagates the representation of data aggregated to the neighboring nodes to generate embeddings that can be projected onto the latent space. During the training, GNNs learn how to map nodes' features to the latent space by fine-tuning neural parameters. The goal is to minimize the loss between the predicted root cause nodes and the observed ones. The proposed algorithm using GNNs is capable of 1) minimizing the level of human intervention by parameterizing turnable features, 2) reducing dependency on priori knowledge by using KPI data and node embeddings, and 3) achieving fairly accurate prediction of what would happen to a network by gathering more participant nodes.

Our contributions include 1) filling the gap of insufficient

study on RCA with KPI data and 2) discovering hidden relationships missing in the datasets. We exploit KPIs combined with domain knowledge to infer hidden information and investigate the possibilities of using KPI data to successfully identify root causes in the network for further mitigation measures. We believe the outcomes of this work can answer the important question of how to extract useful information from incomplete data and lead us toward future self-learning and self-healing networks that require minimum human intervention to address the requirements from increasingly complex networks.

The rest of this paper is organized as follows. In Section II, we briefly introduce the concept of causal analysis, its unique distinction from typical association relationships. We also summarized the major trends of research in this area. In Section III, we formulate the RCA problem as a classification problem and apply cross-entropy as the objective function. Section IV illustrates the proposed framework using GNN-based approach with GSL that we have taken to solve the RCA problem in large complex networks. The simulated experimentation result is presented in Section V. We conclude with Section VI by highlighting the contributions and providing recommendations for future work.

II. Related Work

RCA refers to the capabilities to precisely synthesizing the status of the system for human beings to make decisions based on carefully analyzed system behavior. The core concept behind RCA is causality. Typical standard statistical analysis attempts to infer associations among variables, estimate beliefs or probabilities of past and future events as well as update those probabilities in light of new evidence or new measurements. Causality analysis, on the other hand, goes one step further (see Fig. 1). Its aim is to infer not only beliefs or probabilities under static conditions, but also the dynamics of beliefs under changing conditions; for example, changes induced by treatments or external interventions [3]. [4] subsumes and unifies the approaches to causation, and provides a coherent mathematical foundation for the analysis of causes and counterfactuals.

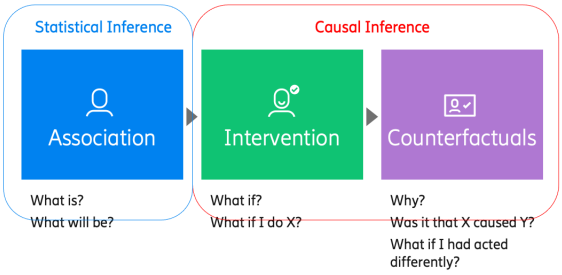


Fig. 1: Illustration of statistical inference vs. causal inference

A large amount of algorithms and techniques has been proposed to perform RCA for decades but we will only briefly summarize some main categories of technologies in this section.

A. Tree-based Approaches

Decision trees are often used as a popular tool to make machine learning decision interpretable for human, which creates possibilities for them to be used for RCA. However, often times, they will have to be combined with domain knowledge/intuition or heuristics to provide some approximation towards the real root causes. Chen *et al.* [5] proposed MinEntropy algorithm as a Gain function for decision trees in their fault diagnosis system. Using heuristic techniques by prioritizing features to explain large number of failures, instead of building a decision forest using random selection, Singh *et al.* [6] also used entropy to select better-performing nodes. Even though tree-based techniques are useful in building an explainable model, authors observed a trade-off between explainability and classification accuracy, when more decision trees were used to build decision forest, they observed some decline in accuracy.

B. Causal Discovery

Causal discovery methods can be used to distinguish direct from indirect dependencies and common drivers among multiple time series. Using the relationship between conditional independence and causality in Bayesian graphs (faithfulness assumption), Runge *et al.* [7] proposed the PCMCI method to carry out causal discovery, which is referred to as the PC method [8] adapted to time series. The strengths of PCMCI is its high dimensionality, flexiblity in choosing different independent tests (both linear or non-linear), and transparency in how causal links are excluded, which occurs when independence is measured for two features given any subset of conditions. The drawback, however, is the limitation of markov equivalence classes. PCMCI has been later improved to include contemporaneous links [9] and latent variables [10].

C. Probabilistic Graphical Models

Probabilistic graphical models map the conditional dependency structure between random variables in terms of probabilistic models [11]. A framework to create probabilistic Bayesian models was proposed in [12] where it adopted different layer types to model the manufacturing process. The creation of the model requires expert knowledge to correctly identify each of the nodes and layers. Dynamic Bayesian models have been adopted in multiple works due to its ability to capture the root causes of the manufacturing process such as in telecommunication [13], [14], semiconductors [15] and chemical processes [16].

Probabilistic graph models are not just limited to Bayesian networks. In [17], directed graphic models can be used to represent the probabilistic variables between the variables in a system where linear regression was used for parameter estimation and penalised least square function was used for structure learning. On the other hand, Markov models can also be used to represent non-directed graphs where casual

effects are not always blatant [18]. Alternatively, work in [19], adopted a spatiotemporal graphical modeling approach utilising symbolic dynamics to represent the casual relationship between nodes.

D. Graph Neural Network Models

GNNs have been a promising research topic and widely applied to many scientific fields in recent years. Different from CNNs that can only be applied to data with Euclidean structures, e.g., an image consisted of pixels on regular two-dimensional grids or a signal composed of sequential digits in one-dimensional space, GNNs offer the flexibility needed to process an arbitrary number of neighboring nodes for each node and thus, have particularly impressive performance on handling the data with non-Euclidean structures [20] [21] [22], such as social networks, three-dimensional images, and telecommunication networks. With the capability of operating aggregation and propagation on irregular data structures, GNN-based solutions can be applied to perform RCA over a network of nodes. In this kind of problems, collected data which are used to be features of nodes can be properly combined with a graph used to describe possible dependencies among nodes to generate graph-structured data. Then, GNN models can fine-tune neural parameters to learn the optimal embeddings that achieve node-level prediction. Recently, GNN-based solutions to RCA over networks have drawn much more attention. He *et al.* [23] proposed a GNN-based method using alarm data to locate devices with abnormal statuses in a telecommunication network. However, GNN models particularly rely on dependencies (the graph structure) and dependency information is usually missing or not provided in collected data, which increases the difficulty of applying GNN models.

III. Problem Formulation

For future 5G networks (as shown in Fig. 2), a wide variety of services will be supported. Each gNB can concurrently serve multiple users for different applications using dedicated data bearers. However, managing the variety of the devices and services will be challenging. Locating the node that is experiencing problems and causing propagated various symptoms in a large network is extremely difficult. This is essentially a RCA problem for the networks based on incomplete information.

We formulate the problem of RCA as a classification problem and concentrate on learning meaningful embeddings from time-series KPI data to improve accuracy of locating root causes. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote an input graph with a set of nodes $\mathcal{V}$ and an adjacent matrix $\mathcal{E} = \{e_{vu} \mid \forall v, u \in \mathcal{V}\} \in R^{\mathcal{N}\times\mathcal{N}}$ where $\mathcal{N}$ is the number of nodes and $e_{vu} = 1$ indicates $(v, u) \in \mathcal{E}$; otherwise, $(v, u) \notin \mathcal{E}$. Let $\mathcal{F} = \{f_v(r) \mid v \in \mathcal{V}, r \in \mathcal{T}\} \in R^{\mathcal{D}\times\mathcal{T}}$ denote a set of $\mathcal{D}$ dimensional time-series KPI data over realization $\mathcal{T}$ where $f_v(r) \in R^{\mathcal{D}}$ represents the set of KPIs pertaining to the node v at realization r. An example of a input graph is shown in Fig. 5 where gNBs can be considered as nodes, each of which is associated with

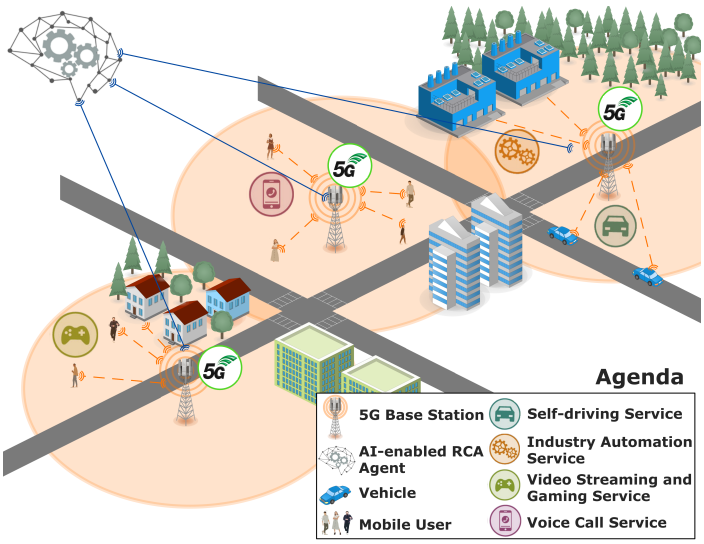


Fig. 2: An overview of deploying an AI-enabled agent over a 5G wireless environment where gNBs provide different types of services using dedicated data bearers for mobile users. KPI data associated with individual gNBs are collected by the agent to analyze/diagnose potential root causes that would lead to a critical crisis.

multivariate time-series KPI data as features reflecting its performance, and edges which connect any pair of two nodes represent relationship dependencies among them. The time-series KPI data of nodes are collected from the 5G networks and utilized as inputs to train a RCA model to identify potential root causes that would happen in the near future. Given an observation of nodes $\mathcal{X}_r = \{f_{v_1}(r), f_{v_2}(r), \cdots, f_{v_n}(r)\}$ at realization r, the goal is to predict a sequence of labels $\hat{\mathcal{Y}}_r = \{\hat{y}_{v_1}(r), \hat{y}_{v_2}(r), \cdots, \hat{y}_{v_n}(r)\}$ that minimize the loss function given by

$$\mathcal{L} = -\frac{1}{\mathcal{N}}\frac{1}{\mathcal{T}}\sum_{r\in\mathcal{T}}\sum_{v\in\mathcal{V}}\mathcal{Y}_r \log g(\mathcal{X}, \mathcal{A}) \tag{1}$$

$\mathcal{Y}_r$ is the ground truth indicating if the nodes are the potential root causes or victim nodes or functioning nodes at the r-th realization, $\mathcal{A}$ is an adjacency matrix indicating if any pair of nodes are connected or not, and $g(\cdot)$ is a given GNN model. Two GNN models, graph convolutional networks (GCNs) and graph attention networks (GATs) are considered in this work and we will elaborate them more in Section IV-C.

IV. The Proposed Framework

To provide useful insights into the issues and root causes of 5G networks, an analysis of the performance data (KPIs) associated with the individual gNBs located in an area and for a given duration of time can be carried out to reflects its utility. The KPIs pertaining to a gNB at a given time-period (realization) can be represented as a feature, which an AI-enabled agent can learn from, to predict labels that signifies which nodes are potential root causes using deep

learning models. We propose a RCA framework consists of four main stages: 1) Input Data representation, 2) Graph Structure Construction, 3) Classification, and 4) Root Cause Evaluation as shown in Fig. 3.

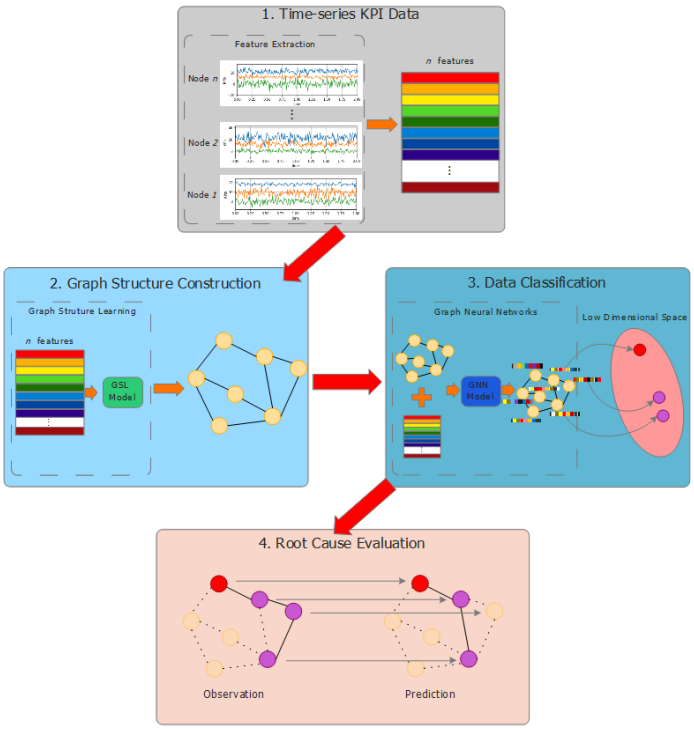


Fig. 3: The proposed framework for root cause analysis.

A. Input Data

Deep learning highly relies on features of input data to learn meaningful embeddings during the entire training process. In this work, base stations are referred to as the nodes and the KPI data associated with nodes are the features. Each node's feature is represented by a vector of KPIs including reference signal received power (RSRP), reference signal received quality (RSRQ), received signal strength indicator (RSSI), signal-to-interference-plus-noise ratio (SINR), and throughput. RSRP and RSRQ measure signal level and quality for modern 5G networks, e.g., in 5G networks, user equipments (UEs) are non-stationary and more likely to move frequently around an area. While being served by a particular node, these UEs measure signal strength and signal quality of neighboring nodes and select the strongest one as the next base station for hand-over before exceeding the service range of the current node. RSSI reveals how well a signal from a base station can be received by a UE. It can be an indicator to determine whether the signal power is strong enough to build a stable wireless connection. SINR measures quality of a wireless connection. Throughput refers to a data rate, namely, how many bits can be delivered to a user per second. For example, 5G is capable of delivering up to tens of Gigabits-per-second (Gbps). These features are aggregated from time-series KPI data in a 5G wireless environment. KPI data are collected from our proposed simulations on which we will elaborated more in Section V-A. To the best of our knowledge, KPI data have not yet been considered and exploited to train models for RCA.

B. Graph Structure Construction

The features extracted from KPI data are leveraged to explore relationship dependencies among nodes in the same geographical area. The graph structure learning (GSL) model proposed by [24] is applied to a given set of nodes with features to learn a weighted adjacency matrix $\mathcal{A}$ which predicts relationship dependencies among the nodes in close proximity to each other. The GSL model can be described as follows:

$$\begin{aligned} Z_1 &= \Theta_1 \cdot \mathcal{X}_r \\ Z_2 &= \Theta_2 \cdot \mathcal{X}_r \\ Z_1^{'} &= tanh(\alpha \cdot Z_1) \\ Z_2^{'} &= tanh(\alpha \cdot Z_2) \\ \mathcal{A} &= ReLU(tanh(\alpha \cdot (Z_1^{'} Z_2^{'} - Z_2^{'} Z_1^{'}))) \end{aligned} \tag{2}$$

where $\mathcal{X}_r$ is a set of nodes' features represented by the KPIs at realization r, Θ_1 and Θ_2 denote the neural parameters for linear layers 1 and 2, respectively, and α is a control variable for the saturation rate of the activation function. The above model is the procedure $graph\_constructor$ at line 5 in **Algorithm 1**. It feeds nodes' features into neural networks to generate the adjacency matrix $\mathcal{A}$ with weights indicating connection likelihood among nodes. From these weights, each node only selects its top k highest connections (weights) as its neighboring nodes to predict relationship dependencies. An example of predicted dependencies among 7 nodes is shown in Fig. 4. The predicted dependencies (edges) are utilized to build an input graph as shown in Fig. 5 for data classification in the next step. The nodes' features combined with edge information is the graph-structured input data upon which the proposed GNN algorithm explores different possible causes and analyzes them if some parts of the networks go wrong. The constructed graph-structured input data using KPI vectors and predicted dependencies is self-contained and informative enough to train a RCA model.

C. Classification

We formulate a root cause identification problem as a node-level classification problem in this work. Nodes are classified into three groups, potential root causes (nodes that are failing and causing other nodes to fail), victim nodes (nodes that are failing and caused by root cause nodes) and functioning nodes (nodes that are not affected and are performing normally), based on their performance which is represented by a vector of KPIs (feature). We adopt GNNs to predict root cause nodes over 5G networks.

The proposed RCA algorithm takes the graph data obtained from the stage 2 (Fig. 3) as inputs and employs GNNs to learn embeddings for nodes over the input graph. Within GNNs,

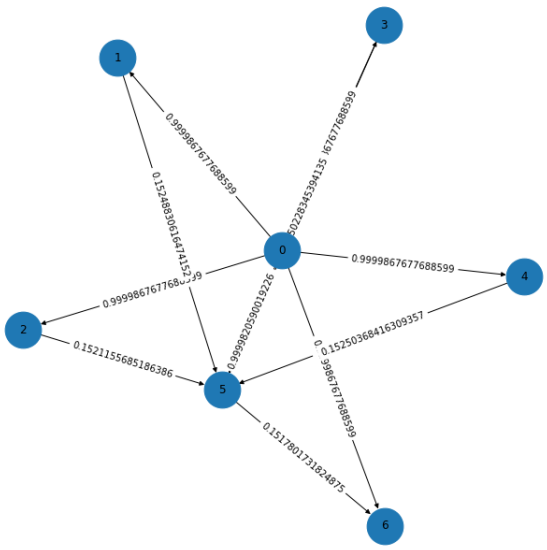


Fig. 4: An example shows the predicted dependencies among 7 nodes for constructing an input graph. Weight denotes connection likelihood among nodes and are used to select neighboring nodes for each node.

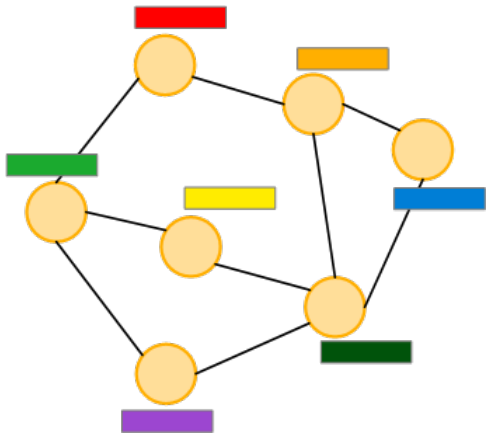


Fig. 5: An illustration shows a input graph where yellow circles denote gNBs (nodes), edges represent relationship dependencies among nodes, and each bar is a vector of KPIs (a feature) associated with an individual node.

for each iteration, a feature of each node is propagated to its neighboring nodes to generate embeddings for all the nodes in the graph. The features from neighboring nodes of a node are aggregated and fed into a transformer neural network (NN) that transforms input features into an embedding by tunable neural weights within the NN's hidden layers. Then, the new embeddings generated in the current iteration will be the features in the next iteration. The aggregation and propagation continue until all nodes get updated by their new embeddings. Throughout the entire training process, GNNs fine-tune the weights that output optimal embeddings which can be mapped onto latent space for separating the potential root causes (sources) and the other victim nodes (symptoms). The detailed implementation can be found in **Algorithm 1**.

1) Graph Convolutional Networks: Graph convolutional networks (GCNs) [25] were proposed to handle graph-structured data where the number of neighboring nodes for each node is usually arbitrary. GCNs allow a flexible kernel to process these arbitrary numbers of neighboring nodes and aggregates features of nodes to generate embeddings. The propagation rule is given by

$$\begin{aligned} g(\mathcal{X}^{(l)}, \mathcal{A}) &= \sigma(D^{-\frac{1}{2}}\mathcal{A}D^{-\frac{1}{2}}\mathcal{X}^{(l)}W^{(l)}) \\ &= \mathcal{X}^{(l+1)} \end{aligned} \tag{3}$$

where $W^{(l)}$ is the matrix of neural weights for the l-th neural layer, σ is a non-linear activation function, $ReLU$, and $\mathcal{X}$ and $\mathcal{A}$ are features of nodes and an adjacency matrix, respectively, as mentioned in Section III.

2) Graph Attention Networks: Graph attention networks (GATs) [26] are an advanced version of the GNN models that incorporates the concept of attention into the GNN models and they are defined by

$$g' = \sigma(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k W^k f_{v_i}) \tag{4}$$

where K is the number of independent attention mechanisms that have been performed if multi-head is considered, $\mathcal{N}_i$ is the neighboring nodes of node i, W^k is the matrix of neural weights for the k-th attention mechanism, and α_{ij}^k are normalized attention coefficients which are calculated by the the k-th attention mechanism to make the coefficients easily comparable across nodes. The attention mechanism assigns different weighting values (coefficients) to different neighboring nodes to express different level of importance of one node to another node. The formula of calculating attention coefficients is given by

$$e_{ij} = a(Wf_{v_i}, Wf_{v_j}) \tag{5}$$

where a is a single-layer feedforward neural network that maps high dimensional features into real numbers. And the normalized form is defined by

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k\in\mathcal{N}_i} exp(e_{ki})} \tag{6}$$

V. Experimentation Results

A. Simulation Environment

In this study, we used a scenario in which a node (a base station) fails and this impacts the network. Tracking this event to find the root cause of the change is the goal of our study. In this simulation, it is assumed that a combination of 4G LTE and 5G NR RANs are randomly distributed over an area; the size of this area is assumed to 3Km by 3Km. This simulation area and the distributed nodes are illustrated in Fig. 6, where

Algorithm 1 GNN-based RCA Algorithm (GNN-RCA)

Input:
Number of nodes n, size of neighboring nodes c, dimension of features $\mathcal{D}$, input features of nodes $\mathcal{X}$, ground truth labels $\mathcal{Y}$.
Initialize:
GCNet [25], GATNet [26], optimizer
model = GCNet() or GATNet()

for each episode $e \in \mathcal{E}_{\mathcal{P}}$ **do**
  for each realization $r \in \mathcal{T}$ **do**
    $\mathcal{X}_r \leftarrow \{f_{v_1}(r), f_{v_2}(r), \cdots, f_{v_n}(r)\}$;
    $\mathcal{Y}_r \leftarrow \{y_{v_1}, y_{v_2}, \cdots, y_{v_n}\}$;
    $\mathcal{A} \leftarrow$ graph_constructor(n, c, $\mathcal{D}$, $\mathcal{X}_r$) [24]
    $\mathcal{E}_i \leftarrow$ where($\mathcal{A} > 0$)
    $\mathcal{E}_w \leftarrow \mathcal{A}$[nonzero($\mathcal{A} > 0$)]
    graph = Data($\mathcal{X}_r$, $\mathcal{Y}_r$, $\mathcal{E}_i$, $\mathcal{E}_w$)
    model.train()
    optimizer.zero_grad()
    output ← model(graph)
    loss ← F.nll_loss(output[train_idx], graph.y[train_idx])
    loss.backward()
    optimizer.step()
    model.eval()
    pred = model(graph).max()
    correct = pred[graph.test_idx].eq(graph.y[graph.test_idx]).sum())
    accuracy = correct / graph.test_idx.sum()
  end for
end for
return accuracy

LTE RAN also known as eNB is an integrated entity, while each 5G RAN includes central unit (CU), distribution unit (DU) and radio unit (RU). We assumed various types of RU and eNB can perform together in this area: in our simulation we assumed macro and pico transmitters, the features of each of these RANs are listed in Table I.

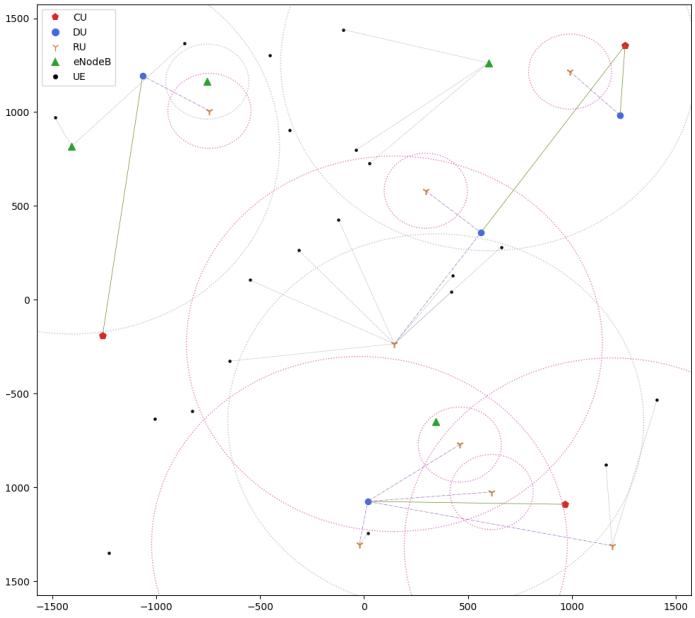


Fig. 6: The simulated layout for root cause analysis

At each realization of this network configuration, 20 multi-antenna users (UEs) are distributed randomly in the area; each UE is equipped with 2 antennas. After association of the UEs with the nearest RU or eNB, all the signal and interference channels are formed based on the path loss and Reighly fading channel models. At each node, we consider block-diagonal precoding for multi-user MIMO transmissions and water filling algorithm to satisfy the power-constraint transmission. Without loss of generality, we assume random user scheduling for those nodes in which the number of requesting UEs are more than the available resources on the node. There are some KPIs that the simulator collects for each UE at this stage which include reference signal received power (RSRP), reference signal received quality (RSRQ), received signal strength indicator (RSSI), signal-to-interference-plus-noise ratio (SINR), and throughput.

TABLE I: Characteristics of RAN Types for both 5G NR and 4G LTE

Type	No. of Antennas	Transmission Power	Path Loss	Cell Radius
macro	6	36dBm	2	1000m
pico	4	30dBm	2	200m

At each realization, one of the transmitters randomly picked up to be turned off; aka failed RU or failed eNB. Thus, all the UEs connected to this node, should be transferred (known as handover mechanism) to adjacent nodes where first, the UE is under their coverage and second, the node has available resources to allocate to this UE. If in a realization, there is at least one UE where its original base station is failed, and no neighboring nodes could serve them, then its ongoing call will be stopped. The simulator raises a fail-flag for this situation, and it is required to investigate the root cause of this failure. Moreover, by handing over any call, the topology of the network in terms of the signal and interference changes which will affect the measured KPIs. Thus, after all possible handovers are finalized, then the KPIs are updated. We simulated this scenario for 80000 realizations and evaluate the root cause of the failing calls within the network.

B. Performance Comparison

In our experiments, we applied GNN models to the simulated dataset to identify potential root cause nodes from 80000 realizations, each of which has a different number of nodes. To make experimental results clearly visualized, we further split them into three datasets with different numbers of nodes, namely, 3 sets of realizations for experimental environments with 7, 8, and 9 nodes. In every training process, GCNs and GATs only focused on a set of realizations with the same number of nodes. The average training loss, average accuracy, and training time of applying GCNs and GATs to different environments are summarized in Table II.

The detail of how GCNs and GATs perform during training is shown in Fig. 7 where the loss is indicated by the red line, and the performance of GCNs and GATs are indicated by blue line and the green line, respectively. Based on our experimental results, we found that increasing the number of nodes in an environment substantially improves learning performance; otherwise, decreasing it demotes the performance. We assume that the underlying reason for this is twofold: 1) the simulated

datasets do not provide satisfactory information for the GSL model (Section IV-B) to infer an appropriate graph structure which 2) later influence performance of GNN models because the predicted graph structure is not accurate enough and the number of features is not great enough. In contrast, an adequate number of nodes provide plenty features for GSL to pick up a graph structure that accurately describe relationship dependencies over a graph. Thus, the GNN models can rely on the accurate graph structure and perform aggregation and propagation using these features as well as learn from more features during training. These contributing factors (accurate graph structure and more features) lead the models to better performance.

Our hypothesis can be validated by the experiments. In Fig. 7a and 7d, the average accuracy of using GCNs is only 0.54 and that of using GATs is even lower, 0.45. The learning curves for both of them are fluctuating and do not seem to be converged. One of the reasons that lead to unstable learning curves is because the data we obtained from the simulation do not contain sufficient information for the GSL model to precisely interpret hidden dependencies from them. But, since there is no ground truth for the dependencies, this is the optimal way to obtain relationship dependencies among nodes. On the other hand, as we increased the number of nodes in the environment, we observed that the learning curves gradually became stable and converged. In Table II, higher accuracy values can be achieved when applying GCNs and GATs to environments with 8 and 9 nodes. Even if the predicted edge knowledge is not accurate enough, this drawback can be improved if more feature data from more participant nodes can be included in the training process.

Furthermore, we observed that the performance of using GATs is worse than using GCNs with respect to the average training loss and average accuracy in Table II and Fig. 7. The explanation is that the attention mechanism is operated on predicted dependencies (edges) and it generates comparatively inaccurate attention coefficients which lead to poor performance. GATs rely on attention coefficients that describe different level of importance of one node to another node in order to train models. Inaccurate attention coefficients directly influence weighting values assigned to neighboring nodes of a node. As a result, the features of some important neighboring nodes are weighted by lower attention but the features of other less important neighboring nodes are weighted by higher attention.

TABLE II: Training performance of GCNs and GATs on different numbers of nodes

Environment	Model	Average Training Loss	Average Accuracy	Training Time
9 Nodes	GCN	0.833971	0.722237	0.577800s
	GAT	1.445496	0.698092	0.899881s
8 Nodes	GCN	0.864629	0.635255	0.577004s
	GAT	1.161185	0.611330	0.940887s
7 Nodes	GCN	0.991822	0.541025	0.575542s
	GAT	1.377784	0.454060	0.929919s

VI. Conclusion and Future Work

As the rapid development of wireless technologies and advanced use-cases supported by them increase, in the foreseeable future, our wireless networks will be inevitably filled with an astronomical amount of devices and base stations, which makes manual diagnosis unrealistic. Self-diagnosing and self-healing is the only way to manage the complexity while maintaining it with reasonable cost. RCA is the very first step towards these goals. This work demonstrates the feasibility of applying GNN to the RCA problem combined with KPI data. Our proposed framework using GNN-based approach and the GSL model achieves better performance if more participant nodes can be included. We found that when more information is aggregated from more participant nodes, the GSL model is capable of generating fairly accurate graph structure on which the GNN model can operate. That is to say, the edge information obtained by the GSL model can substantially assist in locating root causes and should be included to enhance the performance of the proposed RCA algorithm. We have also studied the difference between GCNs and GATs and compared their performance. In the future work, we will add a refinement process to further boost the classification results and explore path propagation to improve prediction accuracy of the root causes.

References

[1] "Cisco visual networking index: global mobile data traffic forecast update, 2018–2023," *Cisco white paper*, 2020.

[2] A. L. Imoize, K. Orolu, and A. A.-A. Atayero, "Analysis of key performance indicators of a 4g lte network based on experimental data obtained from a densely populated smart city," *Data in Brief*, vol. 29, p. 105304, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340920301980

[3] J. Pearl, "Causal inference in statistics: An overview," *Statistics Surveys*, vol. 3, pp. 96–146, 01 2009.

[4] ——, *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2000.

[5] M. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan, and E. Brewer, "Failure diagnosis using decision trees," in *International Conference on Autonomic Computing, 2004. Proceedings.* IEEE, 2004, pp. 36–43.

[6] S. Singh, H. S. Subramania, S. W. Holland, and J. T. Davis, "Decision forest for root cause analysis of intermittent faults," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1818–1827, 2012.

[7] J. Runge, P. Nowack, M. Kretschmer, S. Flaxman, and D. Sejdinovic, "Detecting and quantifying causal associations in large nonlinear time series datasets," *Science Advances*, vol. 5, no. 11, p. eaau4996, Nov 2019. [Online]. Available: http://dx.doi.org/10.1126/sciadv.aau4996

[8] T. D. Le, T. Hoang, J. Li, L. Liu, H. Liu, and S. Hu, "A fast pc algorithm for high dimensional causal discovery with multi-core pcs," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 16, no. 5, p. 1483–1495, Sep 2019. [Online]. Available: http://dx.doi.org/10.1109/TCBB.2016.2591526

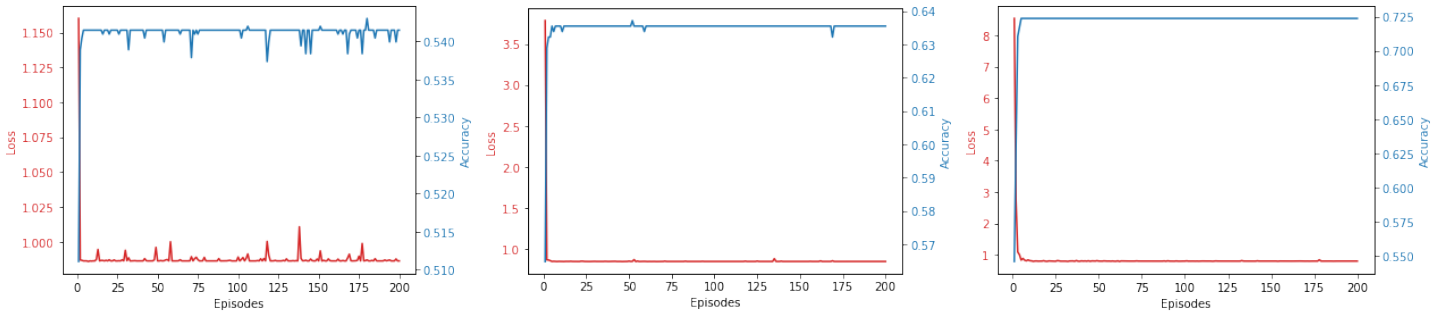
[9] J. Runge, "Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets," 2020.

[10] A. Gerhardus and J. Runge, "High-recall causal discovery for autocorrelated time series with latent confounders," 2021.

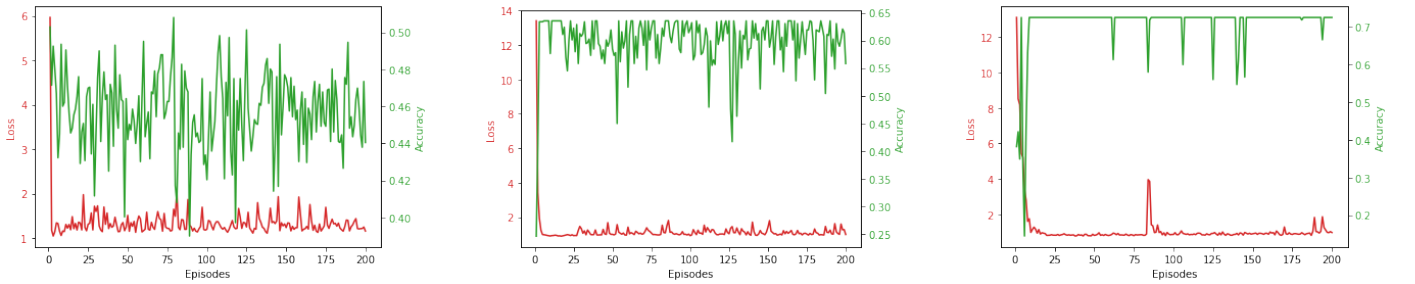
[11] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[12] A. Lokrantz, E. Gustavsson, and M. Jirstrand, "Root cause analysis of failures and quality deviations in manufacturing using machine learning," *Procedia CIRP*, vol. 72, pp. 1057–1062, 2018.

[13] F. J. M. Velasco, "A bayesian network approach to diagnosing the root cause of failure from trouble tickets," *Artif. Intell. Res.*, vol. 1, pp. 75–85, 2012.



(a) $\mathcal{N} = 7$, $m = 4$, $\mathcal{D} = 5$ (b) $\mathcal{N} = 8$, $m = 5$, $\mathcal{D} = 5$ (c) $\mathcal{N} = 9$, $m = 5$, $\mathcal{D} = 5$



(d) $\mathcal{N} = 7$, $m = 4$, $\mathcal{D} = 5$ (e) $\mathcal{N} = 8$, $m = 5$, $\mathcal{D} = 5$ (f) $\mathcal{N} = 9$, $m = 5$, $\mathcal{D} = 5$

Fig. 7: Performance comparison of GCN and GAT on environments with different number of nodes where $\mathcal{N}$ denotes the number of nodes in an environment, m represents number of nodes for training, and $\mathcal{D}$ is the feature dimension.

[14] G. Lee, "Capri: a common architecture for autonomous, distributed diagnosis of internet faults using probabilistic relational models," in *the First Workshop on Hot Topics in Autonomic Computing (HotAC I) in conjunction with the 3rd IEEE International Conference on Autonomic Computing (ICAC-06)*. Citeseer, 2006.

[15] S. Jha, W. Li, and S. A. Seshia, "Localizing transient faults using dynamic bayesian networks," in *2009 IEEE International High Level Design Validation and Test Workshop*. IEEE, 2009, pp. 82–87.

[16] J. Yu and M. M. Rashid, "A novel dynamic bayesian network-based networked process monitoring approach for fault detection, propagation identification, and root cause diagnosis," *AIChE Journal*, vol. 59, no. 7, pp. 2348–2365, 2013.

[17] A. M. E. Gómez, K. Paynabar, and M. Pacella, "Functional directed graphical models and applications in root-cause analysis and diagnosis," *Journal of Quality Technology*, pp. 1–17, 2020.

[18] F. Salfner and M. Malek, "Using hidden semi-markov models for effective online failure prediction," in *2007 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007)*. IEEE, 2007, pp. 161–174.

[19] C. Liu, K. G. Lore, Z. Jiang, and S. Sarkar, "Root-cause analysis for time-series anomalies via spatiotemporal graphical modeling in distributed complex systems," 2018.

[20] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[21] Z. Guo, K. Yu, Y. Li, G. Srivastava, and J. C.-W. Lin, "Deep learning-embedded social internet of things for ambiguity-aware social recommendations," *IEEE Transactions on Network Science and Engineering*, 2021.

[22] Y. Wu, H.-N. Dai, and H. Tang, "Graph neural networks for anomaly detection in industrial internet of things," *IEEE Internet of Things Journal*, 2021.

[23] J. He and H. Zhao, "Fault diagnosis and location based on graph neural network in telecom networks," in *2020 International Conference on Networking and Network Applications (NaNA)*. IEEE, 2020, pp. 304–309.

[24] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 753–763.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[26] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.