Bad Data Detection and Identification Based on Graph Neural Network for Power System State Estimation

Ronald Kfouri, Rabih A. Jabr, *Fellow*, *IEEE*, and Izudin Džafić, *Senior Member*, *IEEE*

***Abstract*—Despite recent progress in solving the state estimation problem, its real-time performance remains challenged by the presence of bad data, increasing computational demands for detection and identification. A state estimator uses neighboring measurements to estimate the system states, similar to how a graph neural network (GNN) refines node embeddings (bus states) based on messages from neighboring nodes. This paper proposes a GNN-based framework that detects and identifies bad data before providing measurements to the state estimator. The framework incorporates grid topology, employs node and edge features, and exploits correlations of measurement data to enhance identification accuracy. Specifically, an edge-conditioned GNN is developed to transform graph-based features into categories that detect bad measurements and identify their sources. The generated dataset uses historical load profiles and includes conventional and synchrophasor measurements to emulate real-life applications. The proposed framework is tested on MATPOWER 6-bus and IEEE 14-, 30-, 118-, and 300-bus systems. The results demonstrate high accuracy and illustrate graph-learning patterns. Thus, operators can take preventive actions before the bad measurements propagate through the state estimator.**

***Index Terms*—Bad data detection, bad data identification, graph neural network, edge-conditioned convolution, machine learning, state estimation.**

I. Introduction

STATE estimation (SE) in power systems is important for the reliability, stability, and efficiency of the power grid. It is an algorithm that provides real-time operating conditions by estimating the voltage magnitude and phase angles based on noisy measurements and other data [1]. With the integration of renewable energy, the state estimator must deliver faster updates to accommodate the rapid changes under grid conditions.

However, the accuracy of SE depends on the quality of the measurement data. Bad data detection and identification (BDDI) techniques are usually employed to detect, identify, and filter bad data measurements, which are erroneous measurements caused by instrumentation errors or communication corruptions due to noise. BDDI techniques can be classified as geometric [2], [3] and statistical approaches [4], [5] or as post-estimation and pre-estimation BDDI based on their position in the SE chain. For post-estimation BDDI, filtering occurs after SE and may require several calls to the state estimator function. In pre-estimation BDDI, filtering happens before processing the data by the state estimator function.

Extensive work has been reported for post-estimation BDDI, using approaches like the χ^2-test [1], [4], [6] and residual-based methods [2], [7], [8]. Nevertheless, the weighted least squares (WLS) estimator cannot detect numerous gross errors, specifically if they appear in leverage points [6]. Also, although the weighted least absolute value (WLAV) estimator deals with gross errors, it fails to detect them at leverage points [9]. Moreover, post-estimation BDDI requires multiple iterations of SE, which increases the computational demand, making it less suitable for real-time applications where quick decision-making is required. For instance, waiting for SE to detect bad data introduces unnecessary delays with phasor measurement units (PMUs) that provide data at high refresh rates (50-60 frames per second) [10].

In contrast, pre-estimation filtering identifies bad data before feeding measurements to the state estimator, preventing error propagation. This approach analyzes all measurements, regardless of whether they belong to leverage points or not, and improves the computational efficiency of the state estimator. However, pre-estimation filtering has received less attention in the literature. For instance, [10] proposes to pre-filter bad data in linear SE using PMUs, while [11] detects bad data and topological errors using pre-estimation filtering. Other examples include [12] and [13], which use neural networks for early BDDI.

Traditional machine learning algorithms handle Euclidean data only and do not integrate power system physics. In deep neural networks, each neuron incorporates a nonlinear activation function and, together with similar neurons, forms a layer that is fully connected with the ones after it and before it [14]. This traditional architecture contains redundant

Manuscript received: March 4, 2025; revised: June 11, 2025; accepted: October 22, 2025. Date of CrossCheck: October 22, 2025. Date of online publication: November 12, 2025.

This work was supported by the American University of Beirut.

R. Kfouri and R. A. Jabr (corresponding author) are with the Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon (e-mail: rk245@aub.edu.lb; rabih.jabr@aub.edu.lb).

I. Džafić is with the Faculty of Electrical Engineering, University of Sarajevo, Sarajevo 71000, Bosnia and Herzegovina (e-mail: idzafic@etf.unsa.ba).

DOI: 10.35833/MPCE.2025.000178

parameters and connections, leading to over-parametrizational, which increases memory and computational costs and reduces model interpretability.

On the other hand, physics-informed machine learning approaches, particularly graph neural networks (GNNs), are a logical alternative for solving power system problems due to their graph-structured nature [15], [16]. GNNs demonstrate outstanding results for handling non-Euclidean data in fields like computer vision, chemistry, and recommendation systems [17]. GNNs store graph data as nodes and edges, and instead of treating nodes as isolated points, they allow them to exchange information with connected neighbors and edges through message passing and aggregation [18]. Different architectures, like graph convolutional networks (GCNs) [19], graph attention networks (GATs) [20], and edge-conditioned convolution (ECC) [21], modify how messages are aggregated, making GNNs adequate for tasks like node classification, edge prediction, and anomaly detection.

In power systems, GNNs incorporate the grid topology and node and edge measurements to solve a regression or classification problem. GNNs have been applied to diverse problems, including unit commitment [22]-[24], fault detection and localization [25], [26], and forecasting applications [27]-[29].

This paper presents a GNN-based framework to detect and identify bad data before processing them into a state estimator. Our framework consists of an edge-conditioned GNN (EC-GNN) that integrates the power system physics, its node and edge measurements, and their correlation to classify good and bad data and identify the location of bad data. The proposed EC-GNN model is based on the architectural design outlined in [21], the message-passing framework introduced in [30], and the activation function proposed in [20], combining the most advantageous elements of these foundational architectures. The contributions are outlined as follows.

1) A novel EC-GNN model is introduced, where edge and node features are concatenated and processed through a multi-layer perceptron (MLP), in contrast to existing models that rely on edge-to-matrix transformations for node updates, e. g., [19]. The proposed model offers more flexibility than traditional GCNs, relies on vector concatenation as a message-passing process and a mean aggregation that integrates all available features, and prioritizes simplicity and robustness, making it suitable for large-scale and noisy datasets.

2) A dataset generation process is presented. The dataset accounts for class imbalance and includes PMU and supervisory control and data acquisition (SCADA) measurements while implementing an encoding scheme that mitigates the imbalance problem.

3) An EC-GNN architecture is developed to improve GCN by explicitly incorporating edge features through vector concatenation, enabling more expressive message passing and enhanced modeling of power system behavior. This architecture avoids the limitations of spectral GCNs such as reliance on eigen decomposition and sensitivity to noise, by learning directly from local node and edge features, achieving greater robustness and scalability.

4) This paper demonstrates interpretability by integrating the t-distributed stochastic neighbor embedding (t-SNE) and GNNExplainer to analyze the learned representations and the decision-making process of the proposed EC-GNN model. The t-SNE visualization reveals that the proposed EC-GNN model produces well-separated and structured feature spaces aligned with class boundaries, while the GNNExplainer identifies the most influential node and edge features contributing to each prediction. Together, these tools provide information on the proposed EC-GNN reasoning and confirm that the model captures meaningful patterns in high-dimensional graph data.

The remainder of this paper is organized as follows. Section II provides the background. Section III introduces the proposed EC-GNN model. Section IV details the dataset generation process and encoding scheme employed in this paper. Section V presents the experimental setup and evaluation framework. Section VI presents and discusses the simulation results. Finally, Section VII concludes this work.

II. Background

A. AC SE

SE is an algorithm designed to compute the state vector of a power grid using noisy measurements and grid data as inputs [1]. The relationship between power flows and system states, which consist of voltage magnitudes and phase angles, is nonlinear and can be expressed as:

$$\boldsymbol{z} = g(\boldsymbol{x}) + \boldsymbol{e} \tag{1}$$

where $\boldsymbol{z} = [z_1, z_2, \ldots, z_m]$ is a vector of m measurements; $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$ is a vector of n system states; $g(\boldsymbol{x})$ is the function that captures the nonlinear dependency of the measurements on the system states; and $\boldsymbol{e}$ is a vector representing the measurement noise that usually follows a Gaussian distribution with zero-mean and a covariance matrix $\boldsymbol{R} = \text{diag}\{\sigma_1^2, \sigma_2^2, \ldots, \sigma_m^2\}$, and σ_i^2 is the variance of measurement z_i [6].

Measurements are obtained from SCADA systems and PMUs deployed throughout the grid. These measurements include complex phasor voltages, currents, power injections, and power flow measurements.

State variables are determined by solving the WLS optimization problem, which minimizes the weighted sum of squared residuals, formulated as:

$$J(\boldsymbol{x}) = \sum_{i=1}^{m} \frac{1}{R_{ii}} \left(z_i - g_i(\boldsymbol{x})\right)^2 \tag{2}$$

where R_{ii} is the variance of the i^{th} measurement z_i; and $g_i(\boldsymbol{x})$ is the i^{th} component of $g(\boldsymbol{x})$.

Usually, (2) is solved iteratively using the Gauss-Newton method [1], [6]. The objective function guarantees that measurements with smaller variances, equivalent to higher confidence, contribute more significantly to the estimation process.

B. BDDI

Bad data are erroneous measurements resulting from er-

rors that might occur during data transfer, unexpected interference, or malfunctioning measurement devices [6]. BDDI techniques are introduced to identify and eliminate such measurements. Most of these techniques rely on the residual, defined as the difference between the received measurement and its corresponding value as a function of the estimated state:

$$\boldsymbol{r}=\boldsymbol{z}-g\left(\hat{\boldsymbol{x}}\right) \tag{3}$$

where $\boldsymbol{r}$ is the residual vector; and $g\left(\hat{\boldsymbol{x}}\right)$ is the predicted measurement vector based on the estimated $\hat{\boldsymbol{x}}$.

The detection criterion is based on the condition $\|\boldsymbol{r}\|<\tau$, where τ is a predetermined threshold. If this condition is violated, it indicates the presence of at least one erroneous measurement. The threshold τ is determined using known error distributions and statistical testing such as the χ^2 test. By using the χ^2 distribution, τ is selected such that any residual exceeding this threshold has a low probability of occurrence under normal conditions, thereby identifying the bad measurement [1].

Moreover, measurement redundancy $(m>n)$ is employed to enhance reliability. This redundancy allows state estimators to detect and filter bad data. Some types of bad data such as unrealistically large power flows are detected a priori, while other types require a posteriori detection, depending on the SE algorithm. For instance, WLS-based SE identifies and eliminates bad data after the estimation process, but requires multiple calls to the estimator function [6].

III. EC-GNN Model

This section first introduces the ECC layer, which is the foundation of our model, and then details the proposed EC-GNN architecture. The proposed EC-GNN model requires the adjacency matrix, node features, and edge features as inputs to classify the graph into distinct categories. These categories indicate whether the input measurements are good or bad data while identifying the location of the bad measurements. The ECC layer employs message passing and graph pooling, inspired by the framework in [30], and incorporates the edge attributes into the aggregation process as in [21]. The proposed EC-GNN architecture is illustrated in Fig. 1. In ECC layers, the orange node indicates the node currently being updated, while the orange edges highlight its immediate neighbors whose features contribute to the message-passing step. The grey edges are the remaining connections in the graph that are not involved in that specific local convolution.

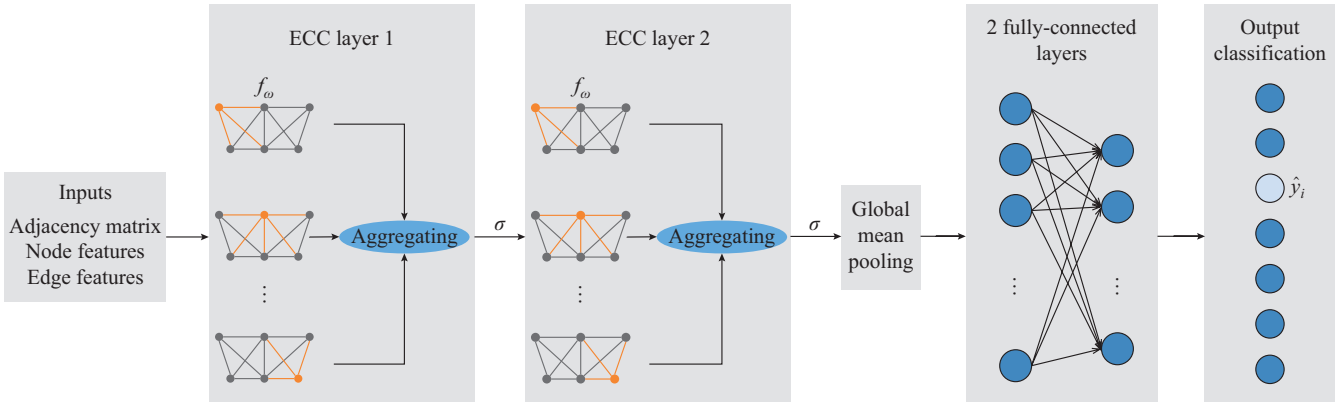

Fig. 1. Proposed EC-GNN architecture showing ECC layers followed by fully connected layers.

A. ECC Layer

The message-passing paradigm of a GNN can be naturally linked to the flow of information and electric power in power grids. A power system can be represented as a graph $\mathcal{G}$, defined as $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$ where $\mathcal{V}$ is a set of nodes, corresponding to the buses, with cardinality $|\mathcal{V}|=n$; and $\mathcal{E}$ is a set of edges, corresponding to transmission lines, with cardinality $|\mathcal{E}|=q$. The adjacency matrix $\boldsymbol{A}\in\mathbb{R}^{n\times n}$, whose entries depict the edges between node i and node j, is used to represent the graph [18].

Each node $i\in\mathcal{V}$ is associated with a feature vector $\boldsymbol{h}_i\in\mathbb{R}^{d_{\text{node}}}$, where $d_{\text{node}}=2$, corresponding to two features per node. Therefore, the entire node feature matrix $\boldsymbol{X}_{\mathcal{V}}\in\mathbb{R}^{n\times 2}$ contains all node features for the graph, which represent the evolving estimate of the power system state. In addition, each edge $(i,j)\in\mathcal{E}$ is associated with an edge feature vector $\boldsymbol{e}_{ij}\in\mathbb{R}^{d_{\text{edge}}}$, where $d_{\text{edge}}=2$, corresponding to two features per edge. Therefore, the entire edge feature matrix $\boldsymbol{X}_{\mathcal{E}}\in\mathbb{R}^{q\times 2}$ contains all edge features for the graph, which represent current phasors and power flow values.

The ECC layer follows the message-passing paradigm of any GNN that works by aggregating information from neighboring nodes and edges to update the state of each node [17]. This closely resembles how measurements propagate in a power grid. The general formulation of the updated feature of a node i, denoted as $\boldsymbol{h}_i^{(k+1)}$, is computed as:

$$\boldsymbol{h}_i^{(k+1)}=\sigma\left(\underset{j\in\mathcal{N}(i)\cup\{i\}}{\Psi}\left(f_\omega\left(\boldsymbol{h}_i^{(k)},\boldsymbol{h}_j^{(k)},\boldsymbol{e}_{ij}^{(k)}\right)\right)\right) \tag{4}$$

where $f_\omega(\cdot)$ is the message-passing function parametrized by learnable weights ω that computes the message from node j to node i; $\Psi(\cdot)$ aggregates messages from all neighbors and the node itself; $\mathcal{N}(i)$ is the set of neighbors of node i; k is the layer index; and $\sigma(\cdot)$ is a nonlinear activation function. The function $f_\omega(\cdot)$ emulates how PMU and SCADA measure-

ments propagate through the grid and influence neighboring buses, while $\Psi(\cdot)$ resembles how the state estimator refines predictions based on surrounding buses.

Based on the features of the source nodes, destination nodes, and the edges, the message from j to i $\boldsymbol{m}_{i\leftarrow j}^{(k)}$ is computed as:

$$\boldsymbol{m}_{i\leftarrow j}^{(k)}=f_{\omega}\left(\boldsymbol{h}_i^{(k)}\| \boldsymbol{h}_j^{(k)}\| \boldsymbol{e}_{ij}^{(k)}\right) \tag{5}$$

where $\|$ represents the concatenation. This message-passing formula is chosen to be an MLP, parametrized by learnable weights ω, following the framework introduced in [30]. The proposed EC-GNN architecture modifies the ECC approach from [21] by avoiding per-edge weight matrix generation. Instead, it adopts a concatenation-based message function, following the efficient and expressive design outlined in [30]. The concatenation combines the information from the source node, destination node, and edge. The MLP approximates nonlinear power system equations, learning how different variables interact. More specifically, the MLP is a feedforward fully connected neural network having two layers with the rectified linear unit (ReLU) as an activation function.

Self-loops are added to the graph so that features of node i contribute to its update, expressed by:

$$\boldsymbol{m}_{i\leftarrow i}^{(k)}=f_{\omega}\left(\boldsymbol{h}_i^{(k)}\| \boldsymbol{h}_i^{(k)}\| \boldsymbol{e}_{ii}^{(k)}\right) \tag{6}$$

where $\boldsymbol{e}_{ii}^{(k)}$ is initialized as a zero vector. The adjacency matrix thus becomes $\tilde{\boldsymbol{A}}=\boldsymbol{A}+\boldsymbol{I}_{n\times n}$, where $\boldsymbol{I}_{n\times n}$ is the identity matrix of dimensions $n\times n$ [18]. This step alleviates overfitting and guarantees that each node update explicitly depends on its initial state and not just neighboring measurements.

Then, the MLP processes this combined information to generate a message, which is a transformed feature vector. This message is aggregated by computing the mean over all neighbors, reducing the impact of noisy measurements by averaging multiple inputs. Also, the mean aggregation offers a computational advantage over the methods proposed in [20] and [21]. The aggregation is followed by a nonlinear activation function. Thus, (4) can be rewritten as:

$$\boldsymbol{h}_i^{(k+1)}=\sigma\left(\frac{1}{\left|\mathcal{N}(i)\right|}\sum_{j\in\mathcal{N}(i)\cup\{i\}}\boldsymbol{m}_{i\leftarrow j}^{(k)}\right) \tag{7}$$

where σ is chosen to be the LeakyReLU, with a negative coefficient of 0.01. ReLU sets all negative inputs to zero, which leads to nodes with zero gradients. LeakyReLU, on the other hand, allows small negative values via the 0.01 slope, ensuring that neurons will not shut down [20]. This is especially important for the application of GNN in power systems, because the adjacency matrix of large-scale grids is inherently sparse.

In this study, edge features are explicitly incorporated into the message-passing function. For each edge (i,j), the edge feature vector $\boldsymbol{e}_{ij}$ is concatenated with the source and destination node features, $\boldsymbol{h}_j$ and $\boldsymbol{h}_i$, respectively, as shown in (5). This combined vector is passed through a shared MLP that learns a nonlinear transformation conditioned on all three components. Thus, the edge features directly influence the message that node j sends to node i. However, in the ECC model [21], edge features are passed into a function that generates a weight matrix for each edge, which in turn is used to linearly transform the neighboring node features. Edge features are not concatenated, but instead used to dynamically generate transformation weights, which comes at the cost of greater computational demands. In the proposed EC-GNN model, edge features are integrated into the message-passing function following the formulation in [30]. This simplifies the architecture of the model while allowing edge features, i.e., power flows and currents, to influence the information passed between nodes.

B. EC-GNN Architecture

The proposed EC-GNN architecture consists of stacking multiple ECC layers followed by fully-connected layers to classify graphs based on node and edge features. The model requires the node feature matrix, edge feature matrix, and adjacency matrix as input. First, the input features are passed through the initial ECC layer, where the message-passing mechanism computes messages by concatenating each source node, destination node, and edge attribute and processing them by an MLP, as shown in (5) and (6). Then, based on (7), the updated node representations are obtained by aggregating the messages and applying the nonlinear LeakyReLU activation function.

Once ECC layer 1 processes the inputs, ECC layer 2 uses the updated features and applies another round of message-passing and aggregation steps. The output of ECC layer 2 represents the final node-level features. The updated features are then pooled into a single graph feature vector using a global mean pooling function, expressed by:

$$\boldsymbol{h}_{\mathcal{G}}=\frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\boldsymbol{h}_i^{(k+1)} \tag{8}$$

where $\boldsymbol{h}_{\mathcal{G}}\in\mathbb{R}^{d_{\text{graph}}}$ is the graph-level representation; and d_{graph} denotes the dimensionality of the learned embedding that summarizes the entire graph. Mean pooling is employed, rather than the maximum or minimum pooling, because it normalizes contributions from all neighbors equally, which is especially important when node degrees vary across graphs.

This graph-level embedding is then passed through two fully-connected layers. At this point, the graph representation $\boldsymbol{h}_{\mathcal{G}}$ is still high-dimensional and may contain redundant or noisy information. The first fully-connected layer applies a linear transformation to reduce the dimensionality and transform $\boldsymbol{h}_{\mathcal{G}}$ into a more structured representation. It is followed by a LeakyReLU activation function to introduce nonlinearity. The second fully-connected layer maps the reduced representation to the output space using a Softmax activation function to produce the classification result. The Softmax activation function is embedded in the loss function of the PyTorch library [31] for numerical stability and better gradient flow.

Over multiple layers, the proposed EC-GNN model learns an implicit representation of the power system states. It classifies the input graphs into distinct categories that dictate whether the input measurements are good or bad data, while identifying the location of the bad measurement. The proposed EC-GNN architecture exploits node and edge features to make informed decisions for graph-based classification.

One difference between the proposed EC-GNN architecture and the one presented in [21] lies in the message passing: the proposed architecture uses (5) and (6) to concatenate the edge and node features and then passes them through the MLP [30]. The study in [21] generates weights dynamically for each edge, allowing for more expressive interactions but adding computational complexity for large graphs [17], [32]. It aggregates messages using learned transformations applied to neighbors' features, which requires additional parameters and makes training more complex. In contrast, the proposed EC-GNN model directly concatenates edge and node features and processes them using a single MLP, simplifying computation while still capturing edge information.

In addition, the GCN presented in [19] is based on spectral methods that rely on the eigen decomposition of the graph Laplacian, which is computationally expensive. It treats edges indirectly through adjacency matrix augmentation, which is achieved by transforming edges into nodes, increasing computational complexity. In contrast, the proposed EC-GNN model uses explicit node and edge feature concatenation, offering a computationally efficient and scalable approach that is better suited to modeling power systems.

IV. Dataset Generation Process and Encoding Scheme

A. Dataset Generation Process

Due to data confidentiality, publicly available datasets are inaccessible for training the model. To address this limitation, the measurement data are simulated by solving the AC power flow problem for multiple test systems under different operating conditions. Historical load curves, which are sampled at 1-min intervals and vary based on weather conditions and load types, are employed to replicate realistic scenarios [13]. These load profiles are applied to a MATPOWER 6-bus demonstrative system and IEEE 14-, 30-, 118-, and 300-bus systems. The AC power flow equations are solved using MATPOWER 7.1 [33].

To synthesize measurement data, a metering configuration is applied after the AC power flow equations are solved. Specifically, PMUs are placed on odd-indexed nodes, and SCADA measurements are placed on even-indexed nodes. This approach creates a hybrid metering configuration that mirrors redundant measurement availability in realistic scenarios where PMUs and SCADA systems coexist. Also, the proposed EC-GNN model is not dependent on this specific setup and generalizes to other sensor placements.

Then, Gaussian noise with zero mean is introduced to mimic measurement uncertainty. The measurement standard deviations are shown in Table I. These values are based on established standards and literature to ensure realism in the simulated measurements. In particular, for PMU measurements, the voltage magnitude error of 0.5% and phase angle error of 0.1° align with the performance requirements specified in IEEE/IEC 60255-118-1:2018 standard [34], which requires a total vector error of less than 1% under steady-state conditions. For SCADA measurements, the assumed standard deviations of 2% for both power injection and flow measurements are consistent with the typical error ranges reported in industry practices and previous studies [35]-[37], which analyzed realistic noise and measurement corruption in power system metering devices. This design ensures that, despite being synthetic, the dataset retains strong practical relevance, supporting the generalizability of the proposed method to real-world settings. Next, the data are shuffled to eliminate seasonality and ensure that the training process is not biased toward temporal patterns.

TABLE I
Measurement Standard Deviations and Perturbation Factor for Dataset Generation Process

System	Measurement	Standard deviation	Perturbation factor (%)
SCADA	Power injection	2%	±50
	Power flow	2%	±50
PMU	Voltage	0.5%	±30
	Current	0.5%	±70
	Phase angle	0.1°	±30

In addition, rather than normalizing the data, we convert the power, voltages, and currents to per-unit (p. u.) values and use the sine of the angles. The goal is to standardize the data relative to the system base values. Using per-unit values avoids dependence on dataset-specific statistics and prevents data leakage. Since the task is BDDI, where bad measurements manifest as outliers, normalization or standardization are deliberately avoided because they compress or hide these anomalies. Statistical normalization is skewed by outliers, which can shift or compress them, reducing the model sensitivity to the types of deviations that we aim to detect. The per-unit system, in contrast, scales based on fixed physical bases rather than dynamic dataset statistics, allowing anomalies to stand out in the input data.

While voltage values are typically close to 1 p. u., power and current values can vary over wider ranges depending on loading conditions and system configuration. Rather than treating these variations as a problem, the proposed EC-GNN model is designed to exploit them. Each feature plays a different role in the graph structure. Voltages are used as node features, whereas power flows and currents are used as edge features. This representation keeps relative relationships intact while mitigating the impact of extreme scale disparities.

Most importantly, when deployed in the real world, the model is expected to operate on a single row of real-time measurements. In such scenarios, there is no access to the full dataset to compute global means, variances, or scaling factors, making statistical normalization infeasible and inconsistent with how the model would be used operationally. In contrast, the per-unit system requires only fixed base values, which are known a priori and remain constant.

B. Dataset Encoding Scheme

Each pair of measurements in the dataset is assigned a binary label of 1 as a first step. To simulate the presence of bad data, additional noise is introduced into specific features

by applying predefined perturbation factor, as shown in Table I. These perturbation factors randomly select one pair of correlated measurements, e.g., active and reactive power at a node, and corrupt both measurements together to emulate realistic bad data scenarios. This corruption is applied at the underlying raw signal level such as voltage, current, or phase angle, causing both derived quantities (P_i and Q_i) to be simultaneously affected. Specifically, a perturbation factor (e.g., ±50%) and additive Gaussian noise are used to emulate realistic variability. The label for this pair is then changed from 1 (good data) to 0 (bad data), marking the presence of bad measurements. For each measurement type, 5000 instances are selected to be corrupted.

The rationale for corrupting a pair of measurements is based on the inherent correlation between measurements [38]. For example, active power P_i and reactive power Q_i at a given node i are correlated rather than independent. These power values are not measured directly but are derived from raw measurements, i. e., voltage magnitude, current magnitude, and phase angle. Consequently, errors originating in these fundamental measurements propagate into the computed active and reactive power values, introducing statistical correlations among measurement errors [38].

While grouping correlated measurements such as P_i and Q_i is based on their shared physical origin, there are still some theoretical risks worth noting. For example, if only one output is affected by communication errors, jointly labeling both measurements as bad data could also reduce interpretability by not pinpointing the exact faulty measurement. However, these risks are mitigated because all corrupted measurements are introduced at the raw voltage/current signal level, ensuring that both P_i and Q_i are affected together in a physically consistent way [38].

Given this correlation, treating P_i and Q_i as a single label is both physically meaningful and computationally efficient. If an error occurs in P_i or Q_i, the root cause is associated with the shared measurement unit at node i. Thus, the labeling is simplified, reducing redundancy while preserving critical information about measurement integrity. In addition, this approach lowers the volume of data required to train the proposed EC-GNN model, halving the number of output labels. This accelerates the training and testing of the proposed EC-GNN model, improving its scalability and making it more efficient for larger-scale power grids.

C. Demonstration Example

For the MATPOWER 6-bus system, we generate 1051200 data instances (2 years × 365 days/year × 24 hours/day × 60 data instances/hour). Figure 2 shows the measurement distribution for the MATPOWER 6-bus system. We note that zero-indexing is used to be consistent with graph theory conventions. There are 12 node features: active and reactive power injection measurements at nodes 0, 2, and 4, represented by P_0, Q_0, P_2, Q_2, and P_4, Q_4, and voltage magnitude and angle measurements at nodes 1, 3, and 5, represented by V_1, θ_1, V_3, θ_3, and V_5, θ_5. There are also 22 edge features: active and reactive power flow measurements from nodes 0 to 4 and 2 to 4, represented by P_{04}, Q_{04}, and P_{24}, Q_{24}, and branch current magnitude and angle measurements, represented by I_{ij} and ϕ_{ij}. This results in $12+22=34$ total input features.

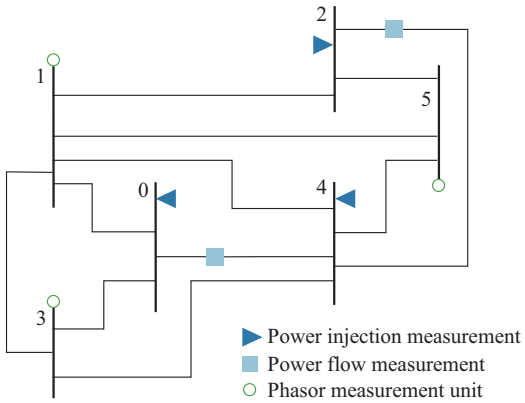

Fig. 2. Measurement distribution of MATPOWER 6-bus system.

As mentioned earlier, 5000 data points per measurement type are corrupted. For the MATPOWER 6-bus system, this results in $5000\times34=170000$ bad data points. This is equivalent to 0.48% of the data being bad data (label 0) and 99.52% of the data being good (label 1). Using this binary encoding technique results in a largely imbalanced dataset, leading to bias in the training and testing of the GNN.

To overcome this challenge, the dataset is encoded differently. That is, labels are encoded for multi-label classification, as shown in Table II. In this encoding scheme:

1) Each feature (node or edge) is represented as a binary value, where 1 indicates that the measurement is good; and 0 indicates that the measurement is bad.

2) Multiple features define a specific class, making it a multi-label classification problem. The Class column in Table II is a label that corresponds to a combination of binary feature states.

This results in 16.18% bad data and 83.82% good data. In other words, there are 18 output classes, encoded as digits ranging from 0 to 17, where the digit 0 means that all the measurements are good, and the digits from 1 to 17 represent the presence of one bad measurement and its location. For example, class 2 indicates that either V_1 or θ_1 is bad, i.e., once flagged, the operator should inspect the measurement at node 1 to diagnose and fix the problem.

D. Principal Component Analysis (PCA)

PCA is employed to reduce the dimensionality of measurement pairs, accelerating the training process. PCA is a statistical method that transforms the data into a lower-dimensional space based on covariance. The first step is to standardize the dataset, compute a covariance matrix that measures the relationships between features, and calculate its eigenvectors and eigenvalues. The eigenvectors represent the directions (principal components) along which the data vary the most, and the eigenvalues represent the amount of variance explained by each eigenvector [14].

TABLE II
LABEL ENCODING FOR NODE AND EDGE FEATURES OF MATPOWER 6-BUS SYSTEM

Feature	Data affected	Class	Implication
Node feature	None	0	All data are good
	P_0, Q_0	1	P_0 or Q_0 is bad
	V_1, θ_1	2	V_1 or θ_1 is bad
	P_2, Q_2	3	P_2 or Q_2 is bad
	V_3, θ_3	4	V_3 or θ_3 is bad
	P_4, Q_4	5	P_4 or Q_4 is bad
	V_5, θ_5	6	V_5 or θ_5 is bad
Edge feature	I_{10}, ϕ_{10}	7	I_{10} or ϕ_{10} is bad
	I_{30}, ϕ_{30}	8	I_{30} or ϕ_{30} is bad
	P_{04}, Q_{04}	9	P_{04} or Q_{04} is bad
	I_{12}, ϕ_{12}	10	I_{12} or ϕ_{12} is bad
	I_{13}, ϕ_{13}	11	I_{13} or ϕ_{13} is bad
	I_{14}, ϕ_{14}	12	I_{14} or ϕ_{14} is bad
	I_{15}, ϕ_{15}	13	I_{15} or ϕ_{15} is bad
	P_{24}, Q_{24}	14	P_{24} or Q_{24} is bad
	I_{52}, ϕ_{52}	15	I_{52} or ϕ_{52} is bad
	I_{34}, ϕ_{34}	16	I_{34} or ϕ_{34} is bad
	I_{54}, ϕ_{54}	17	I_{54} or ϕ_{54} is bad

In this paper, PCA is applied to physically meaningful pairs of electrical quantities, i.e., (P_i, Q_i) or (V_i, θ_i) for node features, and (P_{ij}, Q_{ij}) or (I_{ij}, ϕ_{ij}) for edge features. Each measurement pair is reduced to a principal component that captures the variance within each pair while discarding less relevant information. This reduction exploits the strong correlation often found in power system measurements due to underlying physical laws (Kirchhoff's and Ohm's laws).

For the MATPOWER 6-bus system, PCA is applied on each pair of input features. This results in 6 node features and 11 edge features, rather than 12 and 22, respectively. Meanwhile, the outputs and other hyperparameters remain unchanged.

The upstream effect of applying PCA is a transformation of original feature vectors into a new orthogonal basis where each axis (principal component) captures the maximum variance. Also, by compressing input features, it suppresses noise and redundancy without discarding relevant information. Although these principal components do not correspond to physically interpretable quantities, they preserve the overall variance and structure of the data. As a result, the proposed EC-GNN model continues to receive sufficient discriminatory information while operating in a compact feature space. Importantly, since PCA is applied consistently across all training and test data, the statistical relationships between samples are preserved throughout the pipeline. Overall, PCA minimizes the dataset dimensionality without sacrificing essential information. This method improves computational efficiency, reduces noise sensitivity, and enhances the scalability of the proposed EC-GNN model.

V. EXPERIMENTAL SETUP AND EVALUATION FRAMEWORK

This section outlines the pipeline to train the proposed EC-GNN model. Standard performance metrics are then presented, followed by the interpretability methods used to interpret the model predictions. Finally, the robustness of the model is evaluated under dynamic grid conditions through systematic topological perturbations.

A. Training Pipeline of Proposed EC-GNN Model

To train the proposed EC-GNN model, the dataset, generated in Section IV, is split into 80% allocated for training the proposed EC-GNN model and 20% for testing it. From the training dataset, 20% is allocated as a validation dataset. This validation dataset prevents overfitting during the training phase and helps monitor the model performance on unseen data [14].

Grid search is performed over key hyperparameters, exploring architectures with 2 and 3 hidden ECC layers and neuron counts of 32, 64, 128, and 256, respectively. This configuration is selected because GNNs with more than 3 layers lead to over-smoothing, which reduces their ability to capture meaningful features [39]. Therefore, the proposed EC-GNN model is built using 2 hidden ECC layers, while the number of neurons depends on the grid size. This selection balances model complexity and performance.

The training process spans 100 epochs and employs the Adam optimizer with a learning rate of 0.001. Categorical cross-entropy is used as the loss function. During each epoch, training is conducted in mini-batches, where the model generates predictions $\hat{\boldsymbol{y}}$ based on the current parameters. The loss is computed by comparing the predictions with the actual labels, followed by gradient calculation through backpropagation. The optimizer then updates the model parameters to minimize the loss.

B. Performance Metrics

To evaluate the performance of the proposed EC-GNN model, typical evaluation metrics are employed. The testing accuracy is examined to check the performance of the proposed EC-GNN model on previously unseen data, expressed by:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

where *TP*, *TN*, *FP*, and *FN* denote true positive, true negative, false positive, and false negative, respectively. Furthermore, the F1 score is calculated, defined as:

$$F_1 = \frac{2TP}{2TP + FP + FN} \tag{10}$$

The confusion matrix is also evaluated, which summarizes the prediction results by comparing the true values with the predicted values.

In addition, the Matthews correlation coefficient (MCC) is computed. The MCC considers true and false positives and negatives, providing a balanced measure even when the classes are imbalanced [40], which is the case of the dataset.

C. Interpretability

To interpret the learned behavior of the trained EC-GNN

model, GNNExplainer is implemented to provide post-hoc explanations. GNNExplainer is a model-agnostic method that identifies the most influential edge and node features that contribute to predictions [41]. The method learns a mask over the input graph that highlights the substructure most relevant to the model decision. The explanation is defined as an optimization problem that seeks a subgraph $\mathcal{G}_S$ and a subset of features X_S, which maximally preserves the model predictive power. Specifically, GNNExplainer maximizes the mutual information (MI) between the model predictions Y and the candidate explanation $\left(\mathcal{G}_S, X_S\right)$, expressed as:

$$\max_{\mathcal{G}_S} MI\left(Y, \left(\mathcal{G}_S, X_S\right)\right) = \mathcal{H}(Y) - \mathcal{H}\left(Y \middle| \mathcal{G} = \mathcal{G}_S, X = X_S\right) \quad (11)$$

where $\mathcal{H}(\cdot)$ denotes the information entropy.

Although GNNExplainer is originally designed only for graphs with node features, its use in this study is extended to simulate a node mask and an edge mask, allowing the interpretation of edge and node features on model predictions. For evaluation, the pre-trained EC-GNN model is tested using 20000 random graph samples from the testing dataset. For each of the 20000 graphs, an explanation instance is generated. Then, the node feature masks are extracted directly from the output of the explainer and averaged over all instances to obtain global importance scores across the node features. Similarly, edge feature importance is computed by combining the learned edge-level mask with the corresponding edge attributes and averaging over all instances to obtain global scores for the edge features. The resulting importance scores are normalized to the $[0, 1]$ range.

Moreover, t-SNE is employed to visualize the dataset before and after the training phase. The t-SNE is a well-known dimensionality reduction algorithm for visualizing high-dimensional data. It minimizes the Kullback-Leibler divergence between the distributions in high- and low-dimensional spaces through gradient descent. The t-SNE helps uncover complex data structures and identify relationships within the data. Moreover, it aids in visually distinguishing different classes by clustering them in the reduced-dimensional space [42].

D. Structural Robustness Testing via Topological Perturbations

GNNs learn representations that preserve similarity across structurally related graphs. As a result, they can generalize well under small topological perturbations such as minor edge additions or deletions. This robustness is facilitated by their message-passing structure and embedding-based design, which inherently maps similar graph structures to similar latent spaces [43], [44].

To evaluate the robustness of the proposed EC-GNN model to dynamic grid topologies, the trained EC-GNN model is tested under topological perturbations that simulate practical conditions. Specifically, the grid is modified by adding 1 to 5 new edges to emulate grid reconfiguration due to fault restoration, topology extensions, or line switching. For each test graph, k edges are randomly generated, where $k \in \{1, 2, 3, 4, 5\}$, such that the new edges do not exist in the original topology. Each new edge is assigned a randomly sampled edge feature vector with the same dimensionality as the original edge feature. The rest of the edges, nodes, and their features remain unchanged. The model is evaluated using the modified structure over 10 independent trials for each value of k, where a different set of edges is randomly added. The trials are tested on the trained EC-GNN model (without retaining it on the new data), and the accuracy, F1 score, and MCC are reported and compared against the original metrics.

In addition, to simulate line outages and disconnections due to faults, maintenance, or sensor or communication link failures, an analogous experiment is conducted where 1 to 5 randomly selected edges are removed from each graph. For each value of $k \in \{1, 2, 3, 4, 5\}$, random k edges and their corresponding features are removed from the grid. The remaining edge and node features are left intact. The EC-GNN model is also evaluated for 10 independent trials, and the same performance metrics are computed and averaged after each trial to report the mean.

Both perturbation strategies assist in analyzing the topological robustness of the proposed EC-GNN model. They also help to evaluate its ability to generalize under dynamic grid conditions, measure its sensitivity to topological precision, and quantify the trade-off between structural fidelity and predictive reliability.

VI. Results and Discussion

This section presents the simulation results and discusses them. Simulations were conducted using MATPOWER 7.1 [33], running on MATLAB R2023b, for creating the dataset and Python for building the GNN. Libraries like PyTorch [31] and PyTorch Geometric [45] were used to develop and evaluate the GNN. Simulation is performed on a high-performance computing cluster, equipped with two NVIDIA Tesla V100-PCIe GPUs, with 32 GB of memory each, an 8-core AMD CPU, and 128 GB of RAM.

A. MATPOWER 6-bus System

For the MATPOWER 6-bus system, the node and edge features are fed into the proposed EC-GNN model with 2 hidden layers, each having 64 neurons. Each node has 2 features, and each edge has 2 features. The measurement distribution and the labels are explained in Section IV-C and shown in Fig. 2 and Table II. The MATPOWER 6-bus system is transformed into a graph using the following adjacency matrix:

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} \quad (12)$$

After training, the proposed EC-GNN model is tested on unseen data (the testing dataset), and the testing accuracy is 99.81% with an F1 score of 0.9981 and an MCC of 0.9933, as shown in Table III. The high accuracy indicates that the

proposed EC-GNN model generalizes well to new data and is robust to measurement noise, making it reliable for real-world applications. Furthermore, the high F1 score confirms that the model identifies erroneous measurements while minimizing false alarms (FPs) and missed detections (FNs). The high MCC indicates a strong correlation between the predicted and actual classifications and shows that the model is not biased toward the majority class. That is, the proposed EC-GNN model is learning from minority class instances despite the imbalanced dataset. This is noteworthy as MCC is a more expressive metric in such cases than accuracy alone.

TABLE III
EVALUATION METRICS FOR MATPOWER 6-BUS SYSTEM

Metric	Value	
	Full dataset	With PCA
Number of neurons per layer	64	64
Accuracy (%)	99.81	98.95
F1 score	0.9981	0.9891
MCC	0.9933	0.9642
Time (ms)	1.8257	1.7127

Moreover, the deployment time of the proposed EC-GNN model is evaluated. After training the EC-GNN model, the inference process is repeated 100 times to ensure that the inference time measurement is stable and not affected by outliers or initialization overhead. The average inference time for one data instance is 1.8257 ms, as presented in Table III. This low latency is suitable for real-time inference, especially when measurements are updated every few milliseconds. This also contrasts with traditional BDDI techniques that rely on multiple iterations, introducing significant delays when identifying bad data.

To gain insights into the decision-making process of the proposed EC-GNN model, GNNExplainer is applied to the trained model over 20000 test graph instances of the MATPOWER 6-bus system. The aim is to quantify the relative contribution of each node and edge feature to the model classification outcomes. The average importance scores are visualized in Supplementary Material A Figs. SA1 and SA2 for node and edge features, respectively.

As shown in Supplementary Material A Fig. SA1, the node features with the highest importance scores are Q_2, P_0, and Q_0 with scores of 0.543, 0.538, and 0.531, respectively. The edge-level importance scores, shown in Supplementary Material A Fig. SA2, reveal a similar non-uniform contribution across features. The most influential edge features are Q_{24}, I_{13}, and I_{54} with scores of 0.430, 0.375, and 0.370, respectively.

These importance scores offer an additional layer of interpretability, illustrating how the proposed EC-GNN model internally differentiates between normal and anomalous measurements. The results reinforce the ability to uncover hidden dependencies among network topology, feature space, and classification output. This is a step toward more interpretable and trustworthy graph-based models for power systems.

To further analyze the results, we visualize the data before and after training the proposed EC-GNN model using t-SNE, as presented in Supplementary Material A Figs. SA3 and SA4. Only two classes are visualized for clarity. The t-SNE reduces the 18-dimensional output space into a 2D representation, helping to inspect the model behavior. The pre-training data, presented in Supplementary Material A Fig. SA3, show that the class labels are highly interspersed. Class 0 (blue), representing good measurements, dominates the plot and is randomly mixed with other classes (1-17). This implies that the initial feature space lacks a meaningful structure for classification.

After training, the data distribution undergoes a significant transformation. The data points form a well-clustered feature space, with class 0 being a compact and distinct region, as shown in Supplementary Material A Fig. SA4. Also, class 0 still dominates the plot, confirming that the power system operates mostly under normal conditions, and bad data occurrence is rare. In addition, classes 1-17, representing bad data, are separated, demonstrating that the model has learned discriminative features that capture the underlying patterns in the data, enabling accurate classification. Few misclassified points overlap between clusters, but their limited presence suggests strong generalization rather than overfitting.

In addition, the confusion matrix is visualized in Supplementary Material A Fig. SA5. The rows represent the actual (true) classes, while the columns represent the predicted classes. Each cell (i,j) contains the number of instances where the true class is i but predicted as j. The diagonal entries with $i=j$ represent correct classifications, i.e., where the predicted class matches the true class.

In the matrix, most data instances are on the diagonal, confirming that the proposed EC-GNN model accurately classifies good measurements (class 0) and bad data (classes 1-17). The large values in the diagonal elements also demonstrate strong generalization across all classes, reinforcing the quantitative metrics (accuracy, F1 score, and MCC). Moreover, few misclassifications exist, corresponding to borderline cases where bad measurements exhibit slight deviations that resemble good data. These findings also align with the t-SNE visualization, where certain bad data points lie close to the boundary of good data. Together, these visualizations provide a granular view of the model performance, supporting it as a robust and reliable solution for early BDDI in power grids.

B. Benchmarking Proposed EC-GNN Model Against Established GNN Models

To compare the proposed EC-GNN model with state-of-the-art GNN models, GCN, GAT, and ECC models were implemented on the MATPOWER 6-bus system, following the algorithms presented in [19]-[21], respectively.

As summarized in Table IV, the proposed EC-GNN model achieves the highest accuracy, outperforming GCN, GAT, and ECC models. Unlike GCN model, which relies on Laplacian eigen decomposition, making it computationally expensive and sensitive to noise, the proposed EC-GNN model avoids spectral operations and directly incorporates edge features with node features, making it computationally efficient

and more robust to noise and topological perturbations. Although GAT model improves on GCN model through an attention mechanism, it does not incorporate edge features and requires high computational resources. Furthermore, ECC model incorporates edge features into the message-passing process through dynamically generated filters, increasing computational cost. For the MATPOWER 6-bus system, MLP of the ECC model outputs a 64×64 matrix, during message passing, resulting in 4096 parameters per feature. In contrast, the proposed EC-GNN model directly concatenates edge features with node features through a shared MLP that outputs a 64-dimensional vector, avoiding the need for large matrices and their multiplications. Thus, despite similar accuracies, the proposed EC-GNN model requires significantly less training time (252 min) than ECC model (373 min), highlighting its computational efficiency and suitability for real-time or large-scale deployment.

TABLE IV
PERFORMANCE COMPARISON ACROSS DIFFERENT MODELS

Model	Message passing	Training time (min)	Accuracy (%)	Robustness
GCN* [19]	Spectral-based	269	87.46	Limited robustness due to spectral methods
GAT* [20]	Attention on node pairs	243	87.55	Sensitive to noise, as attention scores might overfit to noisy neighbors
ECC [21]	Edge-conditioned matrix	373	98.15	Moderately robust; performance depends on quality of edge features and learned filters
Proposed EC-GNN	MLP on $\left[\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{e}_{ij}\right]$	252	99.81	High robustness due to direct feature integration and avoidance of learned filters

Note: Items marked with * do not handle edge features.

C. Other Test Systems

To evaluate the scalability of the proposed EC-GNN model, it is tested on larger power systems. For the IEEE 14-bus system, which has 14 nodes and 20 edges, 5000 data points per measurement type are corrupted, resulting in $5000 \times 68 = 340000$ bad data points (0.48% of the dataset). After applying the encoding scheme, the dataset has 32.34% bad data and 67.66% good data. The accuracy, F1 score, and MCC are 98.27%, 0.9777, and 0.9668, respectively, as detailed in Table V.

TABLE V
PERFORMANCE METRICS FOR LARGER-SCALE POWER SYSTEMS

System	Data processing method	Number of neurons per layer	Accuracy (%)	F1 score	MCC	Time (ms)
IEEE 14-bus	Full dataset	64	98.27	0.9777	0.9668	2.1982
	With PCA	64	95.56	0.9476	0.9335	1.9761
IEEE 30-bus	Full dataset	128	96.94	0.9576	0.9448	2.4228
	With PCA	128	93.21	0.9151	0.9061	2.1078
IEEE 118-bus	Full dataset	128	96.83	0.9607	0.9672	2.4861
	With PCA	128	94.11	0.9203	0.9381	1.8801
IEEE 300-bus	Full dataset	128	96.51	0.9570	0.9484	2.7017
	With PCA	128	94.02	0.9355	0.9388	1.9281

For the IEEE 30-, 118-, and 300-bus systems, the datasets are scaled to 2.1, 8.4, and 21 million data instances, respectively, to maintain a similar data imbalance. After encoding, the datasets contain 33.77% bad data for the IEEE 30-bus system, 36.15% bad data for the IEEE 118-bus system, and 33.82% bad data for the IEEE 300-bus system. The performance metrics and inference time for the full dataset are provided in Table V.

Notably, even for the IEEE 300-bus system, the inference time remains remarkably low at 2.7017 ms. This demonstrates the robustness and efficiency of the proposed EC-GNN model, even when handling more complex networks. It also highlights the scalability of the proposed EC-GNN model, making it a feasible solution for real-time applications regardless of grid size. This ensures that the proposed EC-GNN model can be integrated into power grids, supporting real-time monitoring, control, and decision-making.

D. Effect of PCA

The application of PCA yields a 50% reduction in the number of input features. Despite this dimensionality reduction, the proposed EC-GNN model maintains high performance across all test systems. For example, for the MATPOWER 6-bus system, the proposed EC-GNN model achieves an accuracy of 98.95%, an F1 score of 0.9891, and an MCC of 0.9642, compared to 99.81%, 0.9981, and 0.9933 without PCA, as summarized in Table III. For larger-scale test systems, the performance metrics of the reduced dataset are comparable with the ones for the full dataset, as demonstrated in Table V. This small drop in performance is acceptable given the considerable gains in efficiency.

One of the clearest advantages of PCA preprocessing is

shorter training time and lower computational load. Reducing input dimensionality translates into reducing memory consumption during training and inference, which is an advantage in embedded or edge-computing environments. The ability of the proposed EC-GNN model to operate effectively even with compressed inputs proves that it does not overly depend on high-dimensional data. Furthermore, PCA reduces training time by 35%-40%, with larger reductions for more complex grids. This makes the method scalable and suitable for large-scale power systems with limited computational resources. In addition, by projecting the data onto principal components, PCA reducts the noise and low-variance fluctuations, which may otherwise hinder the model's robustness and generalization.

PCA operates in an unsupervised manner and does not require label information, making it compatible with various preprocessing stages. Moreover, as measurement pairs in power systems are often highly correlated due to the physics of power flow, this form of dimensionality reduction is physically meaningful and does not discard critical information. Overall, PCA improves scalability and enhances the practicality of deploying the proposed EC-GNN model in real-world systems, offering a trade-off between input complexity and computational efficiency without sacrificing classification performance.

E. Performance Under Topological Perturbations

To assess the structural robustness of the proposed EC-GNN model, two experiments are conducted to modify the topology of the IEEE 118-bus system, as explained in Section V-D. The proposed EC-GNN model is trained on the original IEEE 118-bus topology and then evaluated on modified versions to assess generalization. In the first experiment, edges are added incrementally to the graph, while in the second experiment, edges are removed to simulate line outages or sensor failures. The accuracy, F1 score, and MCC are averaged across 10 independent trials for each perturbation level, ranging from 1 to 5 modified edges. The results are summarized in Table VI.

TABLE VI
PERFORMANCE OF PROPOSED EC-GNN MODEL UNDER EDGE PERTURBATIONS IN IEEE 118-BUS SYSTEM

Scenario	Number of modified edges	Accuracy(%)	F1 Score	MCC
Baseline	0	96.83	0.9607	0.9672
Adding edges	1	95.97	0.9520	0.9581
	2	95.06	0.9430	0.9486
	3	94.16	0.9343	0.9392
	4	93.28	0.9258	0.9299
	5	92.40	0.9174	0.9208
Removing edges	1	96.42	0.9565	0.9628
	2	95.96	0.9519	0.9580
	3	95.50	0.9473	0.9531
	4	95.05	0.9429	0.9485
	5	94.59	0.9383	0.9436

As a baseline, the model achieves an accuracy of 96.83%, an F1 score of 0.9607, and an MCC of 0.9672 when tested on the original topology. After the addition of new edges, the proposed EC-GNN model demonstrates graceful degradation. Specifically, the accuracy decreases gradually from 95.97% with one added edge to 92.40% with five added edges. Similar declines are observed in the F1 score (from 0.9520 to 0.9174) and MCC (from 0.9581 to 0.9208).

In contrast, edge removal tests simulate the impact of missing data due to faults, sensor loss, or intentional disconnection. The proposed EC-GNN model presents higher resilience to edge removals than to edge additions. For example, the accuracy with one removed edge (96.42%) remains very close to the baseline, and even with five removed edges, the proposed EC-GNN model retains an accuracy of 94.59%, an F1 score of 0.9383, and an MCC of 0.9436. This gradual reduction indicates that the proposed EC-GNN model is not overfitting to a fixed topology and does not excessively rely on a small set of edges. Moreover, the proposed EC-GNN model has learned structural patterns that are meaningful even with minor dynamic behavior, due to redundancy in the message-passing mechanism.

These findings confirm that the proposed EC-GNN model is inherently robust to minor topological perturbations. It also generalizes well under moderate structural uncertainty, making it a suitable solution for real-time and fault-tolerant applications in power systems with evolving or partially known topologies.

F. Comparative Evaluation of EC-GNN-based BDDI Method with Other BDDI Methods

The approach in [10] uses a discrete Kalman filter (DKF) for bad data detection and filtering in PMU-based linear state estimators. The approach applies dynamic thresholds based on statistical confidence intervals to identify anomalies. These are then classified as either bad data or system dynamics (e. g., faults) using spatial correlation heuristics across neighboring PMUs. However, it has three limitations. First, it depends on the manual tuning of statistical thresholds and parameters, which are sensitive to noisy conditions, whereas the proposed EC-GNN model is trained end-to-end without manual calibrations. Second, in contrast to the iterative nature and physical assumptions of DKF-based filtering, the proposed EC-GNN-based method achieves faster and more accurate identification without requiring a detailed physical model. Third, the DKF method lacks quantified performance metrics such as F1 score or MCC, making it difficult to evaluate generalization and robustness.

The approach in [11] utilizes a Kalman filter with a random walk process model to filter out bad data and detect topology errors in wide-area PMU-based systems prior to SE. However, the approach often misclassifies slow or subtle system dynamics as bad data, especially during post-contingency transients. Additionally, although the approach in [11] detects bad data, it does not identify the specific erroneous measurement, unlike the proposed EC-GNN-based method.

VII. CONCLUSION

A GNN-based framework is proposed in this paper, which

consists of an EC-GNN model to identify bad data in power system measurements before SE. The proposed framework shifts the computational burden to an offline stage, enabling real-time deployment with minimal latency. Given the rare occurrence of bad data in power systems, the generated dataset reflects this imbalance. It is encoded using a multi-label encoding scheme to overcome the imbalance problem by exploiting the correlations of measurement data. Furthermore, the proposed EC-GNN model uses grid topology and node and edge features to accurately classify good and bad data by integrating the message passing and aggregation paradigms. Both functions are computationally efficient and reduce sensitivity to noise and outliers. Analyzing different performance metrics, the proposed framework demonstrates that integrating node and edge features leads to highly accurate graph-based learning, achieving real-time anomaly identification with minimal false alarms. Importantly, the proposed EC-GNN model demonstrates robustness to small variations in grid topology, maintaining high classification performance under edge additions and removals. GNNExplainer and t-SNE are also implemented to interpret and visualize the learning of the model. These methods offer insights into how the model differentiates between normal and bad measurements and act as a step toward more interpretable and trustworthy graph-based models for power systems.

Future research aims to explore dynamic GNN architectures such as those presented in [46] and [47], which are explicitly designed to adapt to evolving grid structures and capture time-varying topological behavior.

REFERENCES

[1] A. Abur and A. G. Expósito, *Power System State Estimation: Theory and Implementation*. Boca Raton: CRC Press, 2004.

[2] K. A. Clements and P. W. Davis, "Multiple bad data detectability and identifiability: a geometric approach," *IEEE Transactions on Power Delivery*, vol. 1, no. 3, pp. 355-360, Jul. 1986.

[3] A. L. Monteiro, E. M. M. Nogueira, E. M. Lourenço *et al*., "Bad data processing in fast-decoupled state estimation via geometric approach," *Journal of Control*, *Automation and Electrical Systems*, vol. 36, no. 1, pp. 134-146, Feb. 2025.

[4] E. Handschin, F. C. Schweppe, J. Kohlas *et al*., "Bad data analysis for power system state estimation," *IEEE Transactions on Power Apparatus and Systems*, vol. 94, no. 2, pp. 329-337, Mar. 1975.

[5] R. Baldick, K. A. Clements, Z. Pinjo-Dzigal *et al*., "Implementing nonquadratic objective functions for state estimation and bad data rejection," *IEEE Transactions on Power Systems*, vol. 12, no. 1, pp. 376-382, Feb. 1997.

[6] A. Monticelli, *State Estimation in Electric Power Systems: A Generalized Approach*. New York: Springer Science & Business Media, 2012.

[7] H. J. Koglin, T. Neisius, G. Beiβler *et al*., "Bad data detection and identification," *International Journal of Electrical Power & Energy Systems*, vol. 12, no. 2, pp. 94-103, Apr. 1990.

[8] Z. Yang, H. Liu, T. Bi *et al*., "Bad data detection algorithm for PMU based on spectral clustering," *Journal of Modern Power Systems and Clean Energy*, vol. 8, no. 3, pp. 473-483, May 2020.

[9] N. G. Bretas, S. A. Piereti, A. S. Bretas *et al*., "A geometrical view for multiple gross errors detection, identification, and correction in power system state estimation," *IEEE Transactions on Power Systems*, vol. 28, no. 3, pp. 2128-2135, Aug. 2013.

[10] M. Pignati, L. Zanni, S. Sarri *et al*., "A pre-estimation filtering process of bad data for linear power systems state estimators using PMUs," in *Proceedings of 2014 Power Systems Computation Conference*, Wroclaw, Poland, Aug. 2014, pp. 1-8.

[11] J. Glarbo, H. Jóhansson, J. Ostergaard *et al*., "Detecting topological errors with pre-estimation filtering of bad data in wide-area measurements," in *Proceedings of 2017 IEEE Manchester PowerTech*, Manchester, UK, Jun. 2017, pp. 1-6.

[12] H. Salehfar and R. Zhao, "A neural network preestimation filter for bad-data detection and identification in power system state estimation," *Electric Power Systems Research*, vol. 34, no. 2, pp. 127-134, Aug. 1995.

[13] A. Akagic and I. Džafić, "Enhancing smart grid resilience with deep learning anomaly detection prior to state estimation," *Engineering Applications of Artificial Intelligence*, vol. 127, p. 107368, Jan. 2024.

[14] S. Haykin, *Neural Networks and Learning Machines.* Noida, India: Pearson Education India, 2009.

[15] W. Liao, B. Bak-Jensen, J. R. Pillai *et al*., "A review of graph neural networks and their applications in power systems," *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 2, pp. 345-360, Aug. 2021.

[16] B. Huang and J. Wang, "Applications of physics-informed neural networks in power systems – a review," *IEEE Transactions on Power Systems*, vol. 38, no. 1, pp. 572-588, Mar. 2022.

[17] Z. Wu, S. Pan, F. Chen *et al*., "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4-24, Jan. 2021.

[18] W. L. Hamilton, *Graph Representation Learning*. San Rafael: Morgan & Claypool Publishers, 2020.

[19] T. N. Kipf and M. Welling. (2017, Feb.). Semi-supervised classification with graph convolutional networks. [Online]. Available: https://arxiv.org/abs/1609.02907

[20] P. Veličković, G. Cucurull, A. Casanova *et al.* (2018, Feb.). Graph attention networks. [Online]. Available: https://arxiv.org/abs/1710.10903

[21] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, USA, Jul. 2017, pp. 29-38.

[22] A. V. Ramesh and X. Li, "Spatio-temporal deep learning-assisted reduced security-constrained unit commitment," *IEEE Transactions on Power Systems*, vol. 39, no. 2, pp. 4735-4746, Mar. 2024.

[23] L. Gao, L. Wei, S. Cui *et al*., "A topology-guided high-quality solution learning framework for security-constraint unit commitment based on graph convolutional network," *International Journal of Electrical Power & Energy Systems*, vol. 164, p. 110322, Mar. 2025.

[24] L. Wei, X. Ai, J. Fang *et al*., "Data-augmentation acceleration framework by graph neural network for near-optimal unit commitment," *Applied Energy*, vol. 377, p. 124332, Jan. 2025.

[25] A. Pandey and S. R. Mohanty, "Graph convolutional network based fault detection and identification for low-voltage DC microgrid," *Journal of Modern Power Systems and Clean Energy*, vol. 11, no. 3, pp. 917-926, May 2023.

[26] J. Hu, W. Hu, J. Chen *et al*., "Fault location and classification for distribution systems based on deep graph learning methods," *Journal of Modern Power Systems and Clean Energy*, vol. 11, no. 1, pp. 35-51, Jan. 2023.

[27] X. Dong, Y. Sun, Y. Li *et al*., "Spatio-temporal convolutional network based power forecasting of multiple wind farms," *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 2, pp. 388-398, Mar. 2022.

[28] J. Shi, W. Zhang, Y. Bao *et al*., "Load forecasting of electric vehicle charging stations: attention based spatiotemporal multi-graph convolutional networks," *IEEE Transactions on Smart Grid*, vol. 15, no. 3, pp. 3016-3027, May 2024.

[29] W. Liao, S. Wang, B. Bak-Jensen *et al*., "Ultra-short-term interval prediction of wind power based on graph neural network and improved bootstrap technique," *Journal of Modern Power Systems and Clean Energy*, vol. 11, no. 4, pp. 1100-1114, Jul. 2023.

[30] J. Gilmer, S. S. Schoenholz, P. F. Riley *et al.*, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, Aug. 2017, pp. 1263-1272.

[31] A. Paszke, S. Gross, F. Massa *et al.*, "PyTorch: an imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems*, vol. 32, pp. 8026-8037, Dec. 2019.

[32] P. W. Battaglia, J. B. Hamrick, V. Bapst *et al.* (2018, Oct.). Relational inductive biases, deep learning, and graph networks. [Online]. Available: https://arxiv.org/abs/1806.01261

[33] R. Zimmerman and C. Murillo-Sanchez. (2024, Aug.). MATPOWER (version 7.1). [Online]. Available: https://matpower.org

[34] *Measuring Relays and Protection Equipment – Part 118-1: Synchrophasor for Power Systems – Measurements*, IEEE/IEC 60255-118-1: 2018 Std., 2018.

[35] G. Valverde, S. Chakrabarti, E. Kyriakides *et al*., "A constrained for-

mulation for hybrid state estimation," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 1102-1109, Aug. 2011.
[36] I. Džafić, R. A. Jabr, and T. Hrnjić, "Hybrid state estimation in complex variables," *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5288-5296, Jan. 2018.
[37] I. Džafić and R. A. Jabr, "Real-time equality-constrained hybrid state estimation in complex variables," *International Journal of Electrical Power & Energy Systems*, vol. 117, p. 105634, May 2020.
[38] E. Caro, A. J. Conejo, and R. Minguez, "Power system state estimation considering measurement dependencies," *IEEE Transactions on Power Systems*, vol. 24, no. 4, pp. 1875-1885, Nov. 2009.
[39] T. K. Rusch, M. M. Bronstein, and S. Mishra. (2023, Mar.). A survey on oversmoothing in graph neural networks. [Online]. Available: https://arxiv.org/abs/2303.10993
[40] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, p. 6, Jan. 2020.
[41] Z. Ying, D. Bourgeois, J. You *et al.*, "GNNExplainer: generating explanations for graph neural networks," *Advances in Neural Information Processing Systems*, vol. 32, pp. 9244-9255, Dec. 2019.
[42] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 11, pp. 2579-2605, Nov. 2008.
[43] K. Xu, W. Hu, J. Leskovec *et al.* (2019, Feb.). How powerful are graph neural networks? [Online]. Available: https://arxiv. org/abs/1810.00826
[44] F. Gama, J. Bruna, and A. Ribeiro, "Stability properties of graph neural networks," *IEEE Transactions on Signal Processing*, vol. 68, pp. 5680-5695, Sept. 2020.
[45] M. Fey and J. E. Lenssen. (2019, Apr.). Fast graph representation learning with pytorch geometric. [Online]. Available: https://arxiv. org/abs/1903.02428
[46] A. Pareja, G. Domeniconi, J. Chen *et al*., "EvolveGCN: evolving graph convolutional networks for dynamic graphs," in *Proceedings of the AAAI Conference on Artificial Intelligenc*e, New York, USA, Feb. 2020, pp. 5363-5370.
[47] Y. Wang, Y. Sun, Z. Liu *et al*., "Dynamic graph CNN for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1-12, Oct. 2019.

Ronald Kfouri received the B. E. degree in electrical engineering and the M.S. degree in computer engineering from the Lebanese American University, Byblos, Lebanon, in 2015 and 2023, respectively. From 2016 to 2018, he worked as an Energy Engineer at a private company in Lebanon, where he was responsible for the engineering, implementation, and commissioning of photovoltaic systems. His research interests include application of artificial intelligence in power systems, power system planning, operation, and control, and integration of blockchain and emerging technologies in smart grids.

Rabih A. Jabr received the B.E. degree in electrical engineering (with high distinction) from the American University of Beirut, Beirut, Lebanon, in 1997, and the Ph. D. degree in electrical engineering from Imperial College London, London, U. K., in 2000. He is currently a Professor in the Department of Electrical and Computer Engineering at the American University of Beirut. His research interests include mathematical optimization, power system analysis, and computing.

Izudin Džafić received the Ph.D. degree from the University of Zagreb, Zagreb, Croatia, in 2002. From 2002 to 2014, he was with Siemens AG in Nuremberg, Germany, where he held the positions of Head of the Department and Chief Product Owner (CPO) for Distribution Network Analysis (DNA) R&D. He holds eight US and international patents. He is currently a Professor in the Department of Automatic Control and Electronics, Faculty of Electrical Engineering, at the University of Sarajevo, Sarajevo, Bosnia and Herzegovina. His research interests include power system modeling, development, and application of fast computing to power system simulation.