PowerGraph: A power grid benchmark dataset for graph neural networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Public Graph Neural Networks (GNN) benchmark datasets facilitate the use of GNN and enhance GNN applicability to diverse disciplines. The community currently lacks public datasets of electrical power grids for GNN applications. Indeed, GNNs have the potential to capture complex power grid phenomena over alternative machine learning techniques. Power grids are complex engineered networks that are naturally amenable to graph representations. Therefore, GNN have the potential for capturing the behavior of power grids over alternative machine learning techniques. To this aim, we develop a graph dataset for cascading failure events, which are the major cause of blackouts in electric power grids. Historical blackout datasets are scarce and incomplete. The assessment of vulnerability and the identification of critical components are usually conducted via computationally expensive offline simulations of cascading failures. Instead, we propose the use of machine learning models for the online detection of cascading failures leveraging the knowledge of the system state at the onset of the cascade. We develop PowerGraph, a graph dataset modeling cascading failures in power grids, designed for two purposes, namely, i) training GNN models for different graph-level tasks including multi-class classification, binary classification, and regression, and ii) explaining GNN models. The dataset generated via a physics-based cascading failure model ensures the generality of the operating and environmental conditions by spanning diverse failure scenarios. In addition, we foster the use of the dataset to benchmark GNN explainability methods by assigning ground-truth edge-level explanations. PowerGraph helps the development of better GNN models for graph-level tasks and explainability, critical in many domains ranging from chemistry to biology, where the systems and processes can be described as graphs. The dataset is available at https://figshare.com/articles/dataset/PowerGraph/22820534 and the code at https://anonymous.4open.science/r/PowerGraph/.

1 Introduction

The lack of public Graph Neural Network (GNN) datasets for power grid applications has motivated the development of a new graph dataset. Power grid stability is crucial to modern society, and, therefore, power grids are designed to be robust under failures of different nature. Under particular conditions, however, the failure of critical components can trigger cascading outages. In the worst case, cascading failures spread into the full blackout of the power grid [6, 26]. The complete understanding of complex events as cascading failures is therefore of uttermost importance. Such events are rare and historical data is scarce, therefore, we must rely on simulating cascading failures via computer

Submitted to the 37th Conference on Neural Information Processing Systems (NeurIPS 2023) Track on Datasets and Benchmarks. Do not distribute.

models. The established traditional approach for cascading failure analysis is a quasi-steady state model, such as the OPA model [12], the Manchester model [47], and the Cascades model [22]. These models assess how the power grid responds after an outage is introduced in the grid. In fact, they simulate the complex behavior of the systemic responses and how a chain of successive failures (cascade) propagates in the grid. Since such tools are computationally intensive, they cannot be used by power grid operators for online detection of cascading failure nor for probabilistic risk analysis employing sequential Monte Carlo.

The shortage of historical blackout data and the high computational cost of current methods to simulate cascading failures in power grids highlight the need for machine learning models that can detect cascading failures in almost real-time. Power grid operators, specifically transmission system operators (TSO), will greatly benefit from an online tool able to estimate the potential of cascading failures under given operating conditions of the power grid. The research community has presented new methods that employ machine learning algorithms for the online prediction of cascading failures. The proposed methods often do not generalize for diverse sets of failures [1, 4]. They are trained with datasets created with cascading failure models that often rely on the direct current (DC) power flow approximation [38], less accurate than the alternate-current (AC) power flow. In addition to these limitations, the authors are not aware of publicly available datasets on the subject.

Within the realm of machine learning algorithms, GNN are convenient and powerful machine learning algorithms to model power grid phenomena, since graphs allow an intuitive representation of power grids. In [37], the authors introduce how GNN have been employed for various applications in the field of power systems. Our paper focuses on fault scenario application, but we plan to extend it to power flow calculation in the future. On this topic, the authors of [59] provide a review of GNN for power flow models in the distribution systems. The work in [54] shows that a GNN outperforms a feed-forward neural network in predicting cascading failures in power grids. To produce a large and complete dataset, we use Cascades [22], an alternate-current (AC) physics-based cascading failure model. The model simulates the evolution of the triggering failures yielding the final demand not served (DNS) to the customers. We produce a power grid GNN dataset comprising a large set of diverse power grid states. The power grid state represents the pre-outage operating condition, which is linked to the initial triggering outage (one or more failed elements), referred to as the outage list. Each power grid state is represented as a graph, to which we assign a graph-level label according to the results of the physics-based model. The dataset is generated to suit different graph-level tasks, including multi-class classification, binary classification, and regression.

The presented graph property prediction dataset fills a gap according to the OGB taxonomy for graph dataset [30, 29]. Graph datasets are classified according to their task, domain, and scale. The task is at the node-, link-, or graph- level; the scale is small, medium, or large; and the domain is nature, society, or information. Our dataset comprises a collection of power grid datasets, which are designed for graph-level tasks, and their size ranges from small to medium [21]. Moreover, all the datasets in PowerGraph have the same number of features per node, and therefore, they can be utilized as one combined dataset to train GNN models. Table 1 reports the total number of graphs per power grid, the number of buses and branches in the grid, the number of loading conditions, and the number of outage lists simulated. The dataset fits the society domain, where no public GNN graph property prediction datasets are available [30], see Appendix A.1.

Table 1: Parameters of the AC physics-based cascading failure model for the selected four test power grids. A bus is defined as a node where a line or several lines are connected and may also include loads and generators in a power system. Transmission lines and transformers are defined as branches.

Test system	**# Bus**	**# Branch**	**# Loading conditions** $n_{load\ cond}$	**# Outage lists** $n_{outage\ lists}$	**# Graphs** N
IEEE24	24	38	300	43	12900
UK	29	99	300	132	39600
IEEE39	39	46	300	55	16500
IEEE118	118	186	300	250	75000

Other relevant GNN datasets for graph property prediction are the TU collection [44] and the MoleculeNET [58] dataset. Their application is natural science, particularly molecular graphs, i.e., molecules are represented as graphs to predict certain chemical properties. Publicly available power grid datasets such as the Electricity Grid Simulated (EGS) datasets [15], the PSML [64], and the Simbench dataset [43] are not targeted to machine learning on graphs. In addition, both the EGS and PSML provide data for very small power grids, with 4 and 13 nodes respectively. Instead, Simbench focuses only on power system analysis in the German distribution and transmission grid, and the dataset is not designed for machine learning on graphs. In [46], the authors present new datasets of dynamic stability of synthetic power grids. They found that their GNN models, which primarily use emphasizes node regression, can predict highly non-linear targets from topological information. On the other hand, PowerGraph, which uses graph-level tasks, does not address dynamic stability and relies on established real-world-based power grid models to predict the development of cascading failures. Overall, the dataset we provide fills a gap in the domain of GNN datasets for graph-level tasks [30] and is the only publicly available GNN dataset for power grids.

Besides benchmarking GNN models, the dataset is intended to be used for explainability methods. Therefore, we assign ground-truth edge explanations using the insights provided by the physics-based cascading failure model. As explanations, we consider the branches that have failed after the initial trigger, i.e., the cascading stage. In the field of explainability for GNN, there is to the best of our knowledge no existing real-world dataset with reliable ground-truth explanations [2]. There have been recent attempts to create a synthetic graph data generator producing a variety of benchmark datasets that mimic real-world data and are accompanied by ground-truth explanations [2], as well as to provide atom-wise and bond-wise feature attribution for chemical datasets [28, 32]. However, none of these attempts provides real world data with empirical explanations. Here, we propose a real world dataset for GNN graph level tasks that has clear ground-truth explanations obtained from physic-based simulations.

This work provides a large-scale graph dataset to enable the prediction of cascading failures in electric power grids. The PowerGraph dataset comprises the IEEE24 [17], IEEE39 [18], IEEE118 [16] and UK transmission system [45]. These test power systems have been specifically selected due to their representation of real-world-based power grids, encompassing a diverse range of scales, topologies, and operational characteristics. Moreover, they offer comprehensive data with all the necessary information required for conducting cascading failure analysis. With PowerGraph, we make GNN more accessible for critical infrastructures such as power grids and facilitate the online detection of cascading failures. Our contributions are the following:

- We provide a data-driven method for the online detection of severe cascading failure events in power grids.
- We make the dataset public in a viable format (PyTorch Geometric), allowing the GNN community to test architectures for graph-level applications.
- The dataset includes several graph-level tasks: binary classification, multi-class classification, and regression.
- We provide explanatory edge masks, allowing the improvement of GNN explainability methods for graph-level applications.

The rest of the paper is organized as follows: Section 2 describes the physics-based model used to simulate cascading failure scenarios; Section 3 outlines the structure of the graph datasets; Section 4 reports the benchmark experiments of the different datasets; Section 5 describes the method used to benchmark explainability methods; and Section 6 concludes the article with a final discussion.

2 Physics-based model of cascading failures

We employ the established Cascades model [22, 24] for cascading failure simulations to produce the GNN datasets. Indeed, its application to the Western Electricity Coordinating Council (WECC) power grid demonstrates that Cascades can generate a distribution of blackouts that is consistent with the historical blackout data [35]. Cascades is a steady-steady state model with the objective to simulate the power grid response under unplanned failures in the grid. For that purpose, the model simulates

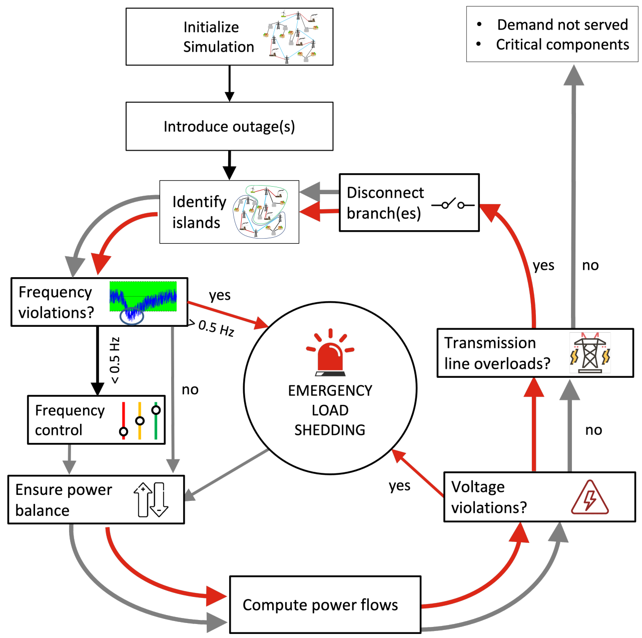

Figure 1: Workflow of the Cascades [23] model, used to simulate cascading failures in power grids. Separate runs of Cascades are performed for the different test power grids namely, IEEE24, IEEE39, UK, and IEEE118.

the power system's automatic and manual responses after such failures. Initially, all components are in service and there are no overloads in the grid. The system is in a steady-state operation with the demand supplied by the available generators, which produce power according to AC- optimal power flow (OPF) conditions [10]. The simulation begins with the introduction of single or multiple initial failures. Then, Cascades simulates the post-outage evolution of the power grid, i.e., identifies islands, performs frequency control, under-frequency load shedding, under-voltage load shedding, AC power flows, checks for overloads, and disconnects overloaded components. The model returns two main results: the demand not served (DNS) in MW and the number of branches tripped after the initial triggering failure. The simulation is performed for a set of power demands sampled from a yearly load curve. For each season of the year, an equal number of loading conditions are randomly sampled. We use a Monte-Carlo simulation to probabilistically generate outages of transmission branches (lines and transformers). We define the number of loading conditions and the size of the outage list. Therefore, we are able to simulate a large number of scenarios and thus create large datasets. Each scenario generated is a power grid state, and therefore, becomes an instance of the dataset. For each combination of loading condition and element in the outage list, we simulate the cascading failure, identify the terminal state of the power grid, quantify the demand not served, and list the tripped elements. Figure 1 shows the structure of the Cascades model [23].

3 PowerGraph benchmark for graph-level predictions and explainability

The PowerGraph dataset is obtained by processing the results of the Cascades model. Because we work with graph-level tasks, the dataset is a collection of N attributed graphs $\mathcal{G} = \{G_1, G_2, .., G_N\}$. Each input graph reflects a unique pre-outage operating condition of the system and one set of single/multiple outages. Therefore, the total number of graphs N per power grid equals to $n_{load\ cond} * n_{outage\ lists}$. Finally, each graph is assigned an output label corresponding to the chosen task. An attributed graph is defined $G = (\mathcal{V}, \mathcal{E}, \mathbf{V}, \mathbf{E})$, where $\mathcal{V}$ is the set of nodes (bus) and $\mathcal{E}$ is the set of edges (branches), $\mathbf{V} \in \mathbb{R}^{|\mathcal{V}| \times t}$ is the node feature matrix, with $|\mathcal{V}|$ nodes and t features per node and $\mathbf{E} \in \mathbb{R}^{|\mathcal{E}| \times s}$ is the edge feature matrix, with $|\mathcal{E}|$ edges and s features per edge. Finally, the graph connectivity information is encoded in COO format [20]. We assign three bus-level features and four branch-level features. Each feature quantity is normalized using mean normalization. The input features are:

Bus:

- Net active power at bus i, $P_{i,net} = P_{i,gen} - P_{i,load}, P \in \mathbb{R}^{n_{bus} \times 1}$, where $P_{i,gen}$ and $P_{i,load}$ are the active generation and load, respectively.
- Net apparent power at bus i, $S_{i,net} = S_{i,gen} - S_{i,load}, S \in \mathbb{R}^{n_{bus} \times 1}$, where $S_{i,gen}$ and $S_{i,load}$ are the apparent generation and load, respectively.
- Voltage magnitude at bus i, $V_i \in \mathbb{R}^{n_{bus} \times 1}$, where n_{bus} is the number of buses in the power grid.

Branch:

- Active power flow $P_{i,j}$
- Reactive power flow $Q_{i,j}$
- Line reactance $X_{i,j}$
- Line rating $lr_{i,j}$.

Figure 2 displays an instance of the PowerGraph dataset. Each graph represents a state of the power grid associated with a loading condition and an outage (single or multiple failures). Since each outage is associated with disconnected branches, we remove the respective branches from the adjacency matrix and from their respective edge features. Therefore, each instance of the dataset is a graph with a different topology. The total number of instances is reported in Table 1. For each initial power grid state, we have knowledge of the post-outage evolution of the system, i.e., the demand not served (DNS) and the number of tripped lines. We label it as a cascading failure in each case that results in branches tripping after the initial outage. With these two results, we can assign an output label to each graph for different models:

Binary classification - we assign each instance to two classes:

- DNS=0, initial state results in a stable state, label 0
- DNS>0, initial state results in an unstable state, label 1

Multi-class classification - we assign each instance to four classes:

- DNS>0, cascading failure of components besides the first trigger, Category A
- DNS>0, no cascading failure of components besides the first trigger Category B
- DNS=0, cascading failure of components besides the first trigger, Category C
- DNS=0, no cascading failure of components besides the first trigger, Category D

Regression - we assign each instance the DNS in MW

The choice among binary classification, multi-class classification, or regression depends on the use of the GNN model trained with the PowerGraph dataset. The binary classification model serves as an early warning system, i.e., detects initial states of the power grid that are critical. The multi-class classification model allows us to distinguish different scenarios. Indeed, a transmission system operator could benefit from knowing when a cascading failure does not necessarily cause demand not served and vice-versa. Finally, with the regression model, we can directly access the final demand not served associated with particular pre-outage states of the system. In this case, the GNN model becomes a surrogate of the physics-based model useful both as an early warning system and to perform security evaluation with low computational cost.

Table 2: Multi-class classification of datasets. c.f. stands for *cascading failure* and describes a state resulting in cascading failure of components. DNS denotes demand not served.

Category A	**Category B**	**Category C**	**Category D**
DNS $>$ 0 MW	DNS $>$ 0 MW	DNS = 0 MW	DNS = 0 MW
c.f. ✓	c.f. $\times$	c.f. ✓	c.f. $\times$

Explainability mask We assign ground-truth explanations as follows: when a system state undergoes a cascading failure, the cascading edges are considered to be explanations for the observed demand not served. Therefore, for the Category A instances, we record the branches that fail during the development of the cascading event. We set the explainability mask as a Boolean vector

Table 3: Results of categorization in percentage.

Power grid	Category A	Category B	Category C	Category D
IEEE39	2.18%	3.48%	1.46%	92.88%
IEEE118	0.07%	5.84%	2.01%	92.08%
IEEE24	33.90%	4.88%	0.16%	61.06%
UK	4.06%	0%	8.02%	87.92%

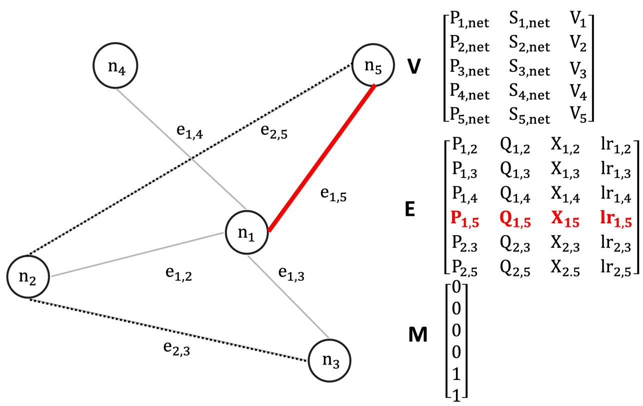

Figure 2: Structure of one instance of the GNN dataset for an exemplary power grid. The same structure is kept for all the power grids in PowerGraph, IEEE24, IEEE39, UK, and IEEE118. We highlight the initial outage in red, the line is removed both from the graph connectivity matrix and from the edge feature matrix. The cascading edges are highlighted with the dotted line and encoded in the **M** boolean vector (0 - the edge has not tripped during cascading development, 1 - otherwise).

$\mathbf{M} \in \mathbb{R}^{|\mathcal{E}|\times 1}$, whose elements are equal to 1 for the edges belonging to the cascading stage and 0, otherwise (see Figure 2).

4 Benchmarking graph classification and regression models

In this section, we outline the method used to benchmark classification and regression models.

Experimental setting and evaluation metrics For each power grid dataset, we utilize baseline GNN architectures as they are common in the graph xAI community. Specifically, we use GCNConv [34], GATConv [55], and GINEConv [31] to demonstrate that the PowerGraph datasets can be used to benchmark GNN and methods used to explain them. Furthermore, we experimented with the state-of-the-art graph transformer convolutional layers [52] since they are the backbones of the most recent Graph Transformer models: GraphGPS [49], Transformer-M [41], TokenGT [33]. Finally, we resort to all of the aforementioned models because they account for the edge features, which are highly relevant in the case of power grids. We tune the number of MPL $\in \{1, 2, 3\}$ and the hidden dimensionality $\in \{8, 16, 32\}$. Adam optimizer is used with the initial learning rate of 10^{-3}. Each model is trained for 200 epochs with learning rate adjusted in the learning process using a scheduler, which automatically reduces the learning rate if a metric has stopped improving. We split train/validation/test with 80/10/10% for all datasets and choose a batch size of 128. We present three graph-level models, namely, binary/ multi-class classification, and regression. For classification models, we consider balanced accuracy [11] as the reference evaluation metric. Indeed, balanced accuracy has been designed as a metric for classification tasks where a strong class imbalance is observed (see Table 3). It allows prioritizing all the classes equally, in contrast to the F1 or F2 score, and it gives interpretable results for multiclass classification, in contrast to ROC-AUC [50]. Indeed, a strong class imbalance is observed. For regression models, we use mean squared error as metric.

Observations We report the best model performance for each power grid and MPL in Tables 4, 5, and 6. For the different MPL, we only show the set of hyper-parameters yielding the best performance, and the best model per power grid is highlighted in bold. The GNN architecture comprises 1) a number of MPLs, each followed by PReLU [27] activation function, 2) a global pooling operator to obtain graph-level embedding from node embeddings, and 3) one fully connected layer. For the classification model, we do not observe relevant differences among the mean, max, and sum global pooling operators. The classification results are obtained with max global pooling. The regression results are obtained by concatenating max and sum global poolings.

Table 4: Binary classification models results on the test set averaged over five random seeds. Balanced accuracy is used as reference metric.

Power grid	**MPL type**	**No MPL**	**Hidden dimension**	**Test Accuracy**	**Test Balanced Accuracy**
IEEE24	GCN	2	32	0.8667 ± 0.0049	0.8769 ± 0.0056
	GINe	3	32	0.9798 ± 0.0046	0.9800 ± 0.0035
	GAT	3	32	0.9008 ± 0.0052	0.9067 ± 0.0034
	Transformer	**3**	**16**	**0.9907 ± 0.0040**	**0.9910 ± 0.0037**
IEEE39	GCN	3	32	0.9733 ± 0.0012	0.8113 ± 0.0011
	GINe	2	32	0.9939 ± 0.0020	0.9550 ± 0.0041
	GAT	3	32	0.9697 ± 0.0023	0.7865 ± 0.0061
	Transformer	**3**	**16**	**0.9952 ± 0.0015**	**0.961 ± 0.016**
UK	GCN	3	32	0.9657 ± 0.0027	0.7176 ± 0.0023
	GINe	**2**	**32**	**0.9975 ± 0.0018**	**0.9820 ± 0.0010**
	GAT	3	8	0.9889 ± 0.0005	0.9175 ± 0.0012
	Transformer	**3**	**16**	**0.9960 ± 0.0016**	**0.9820 ± 0.0045**
IEEE118	GCN	3	32	0.9917 ± 0.0015	0.9364 ± 0.0032
	GINe	3	8	0.9992 ± 0.0046	0.9921 ± 0.0035
	GAT	3	32	0.9880 ± 0.0012	0.9427 ± 0.0005
	Transformer	**3**	**32**	**0.9992 ± 0.0005**	**0.9947 ± 0.0041**

Table 5: Multi-class classification models results on the test set averaged over five random seeds. Balanced accuracy is used as reference metric.

Power grid	**MPL type**	**No MPL**	**Hidden dimension**	**Test Accuracy**	**Test Balanced Accuracy**
IEEE24	GCN	2	32	0.8465 ± 0.0023	0.6846 ± 0.0009
	GINe	2	32	0.9798 ± 0.0019	0.9426 ± 0.0028
	GAT	3	32	0.9054 ± 0.0020	0.8375 ± 0.0009
	Transformer	**3**	**32**	**0.9829 ± 0.0012**	**0.9894 ± 0.0016**
IEEE39	GCN	2	8	0.9242 ± 0.0019	0.4071 ± 0.0012
	GINe	3	16	0.9939 ± 0.0015	0.9693 ± 0.0019
	GAT	2	16	0.9497 ±0.0022	0.5577 ± 0.0027
	Transformer	**3**	**32**	**0.9550 ± 0.0009**	**0.9742 ± 0.0016**
UK	GCN	3	32	0.9068 ± 0.0023	0.4615 ± 0.0038
	GINe	2	32	0.9798 ± 0.0020	0.9347 ± 0.0017
	GAT	3	8	0.9563 ± 0.0009	0.7452 ± 0.0014
	Transformer	**3**	**8**	**0.9912 ± 0.0009**	**0.9798 ± 0.0013**
IEEE118	GCN	3	8	0.9771 ± 0.0010	0.8303 ± 0.0016
	GINe	3	32	0.9968 ± 0.0018	0.9586 ± 0.0010
	GAT	3	16	0.9677 ± 0.0010	0.7392 ± 0.0011
	Transformer	**3**	**8**	**0.9992 ± 0.0013**	**0.9833 ± 0.0006**

Discussion Most GNN models achieve high performance on the power grids of PowerGraph. We compare GCN, GAT, GINe, and Transformer. Of all MPL considered, only GCN does not take edge features into account; as a result its performance is low in most cases. Transformer achieves the state-of-the-art on all power grids for the binary and multi-class models. In the regression model, Transformer and GINe are the best-performing models. Overall, the model for binary and classification models exhibit excellent results. However, the regression model, which is of importance in providing a prediction of the demand not served, does not achieve the desired level of performance. While the classification models showed consistent performance across various

Table 6: Regression models results on the test set averaged over five random seeds. MSE error is used as reference metric.

Power grid	MPL type	No MPL	Hidden dimension	MSE loss
IEEE24	GCN	1	32	2.80E-03 ± 5.69E-04
	GINe	3	16	2.90E-03 ± 2.88E-04
	GAT	2	16	2.90E-01 ± 5.00E-04
	Transformer	**3**	**8**	**2.70E-03 ± 3.16E-04**
IEEE39	GCN	2	32	5.61E-04 ± 5.04E-05
	GINe	**3**	**32**	**5.04E-04 ± 5.04E-05**
	GAT	3	32	5.62E-04 ± 4.66E-05
	Transformer	3	32	5.47E-04 ± 8.50E-05
UK	GCN	3	32	7.07E-03 ± 6.45E-04
	GINe	2	32	7.65E-03 ± 6.17E-04
	GAT	3	32	7.60E-03 ± 6.12E-04
	Transformer	**3**	**16**	**7.00E-03 ± 5.10E-04**
IEEE118	GCN	2	32	4.00E-06 ± 2.94E-07
	GINe	**2**	**32**	**3.00E-06 ± 3.51E-07**
	GAT	2	8	4.00E-06 ± 3.70E-07
	Transformer	2	8	5.00E-06 ± 6.55E-07

power grids, the regression models demonstrate lower MSE values for larger power grids. This observation can be attributed to the fact that larger power grids offer a greater diversity of scenarios, thus making it increasingly more difficult for a GNN model to identify and learn cascading failure patterns. Nevertheless, a regression model offers the most informative and comprehensive results since it predicts the exact magnitude of demand not served given a component failure and operating conditions. However, our results show that the regression models trained on the PowerGraph datasets do not provide the expected performance. Therefore, further advancements and innovations in GNN architectures are needed to achieve more robust and accurate regression results. Finally, we test the capability of GNN model to generalize to the systems not seen in training, i.e. inductive property of GNN [56]. We report the results in Appendix A.6.

Models trained using the above approach, although representing real systems, are built with synthetic data from a cascading failure model. To render these models applicable to real-world systems further work is necessary. First, the cascading failure model that generates the data needs to be validated and calibrated on the system of interest. Second, the GNN model should be further trained using real-world cascading failure events from the system of interest.

5 Benchmarking explanations on the graph-classification models

In this section, we outline the method used to benchmark explainability methods. We focus on explaining the power grids of Category A of the multi-class classification model. This choice is explained in Appendix A.2.

Experimental setting and datasets For each dataset, we take the trained Transformer with 3 layers and 32 hidden units described in section 4. To benchmark explainability methods, we do not necessarily need the best GNN model. An appropriate filtering on the nature of the predictions (correct or mix) and the focus of the explanation (phenomenon or model focus) [5] can circumvent smaller test accuracy. We adopt the same training parameters. We evaluate the posthoc explainability methods: Saliency [8], Integrated Gradient [53], Occlusion [19], GradCAM [51], GNNExplainer [60] with and without node feature mask, PGExplainer [40], PGMExplainer [57], SubgraphX [63], and GraphCFE [42]. In Appendix A.3, we report more experimental details on the GNN performance and the explainability methods. The PowerGraph benchmark with explanations is used to test and compare existing explainability methods. The role of explainers is to identify the edges that are necessary for the graphs to be classified as Category A [5]. Then, the resulting edges are evaluated on how well they match the explanation masks, which represent the cascading edges. We compare the results obtained on the PowerGraph datasets with scores computed for the synthetic dataset BA-2Motifs [40]. This dataset has 800 Barabási base graphs. Half graphs are attached with "house" motifs (label

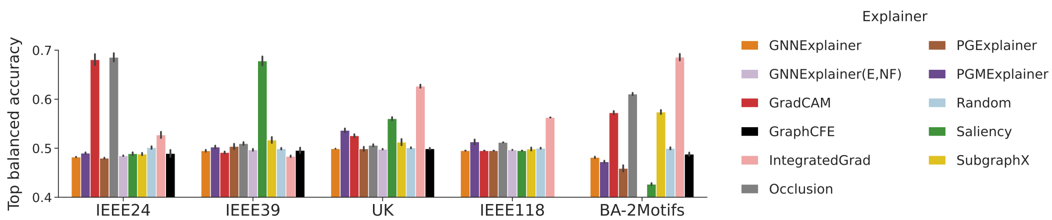

Figure 3: Top balanced accuracy of the PowerGraph datasets and the synthetic dataset BA-2Motifs. The *top* balanced accuracy is computed on explanatory edge masks that contain the *top* k edges that contribute the most to the model predictions, with k being the number of edges in the corresponding ground-truth explanations.

0) and the rest are attached with five-node cycle motifs (label 1). The ground-truth explanations in this graph classification are the type of motifs attached to the base graph (house or five-node cycle). The BA-2Motifs dataset is commonly used to compare the performance of explainability methods [2, 3, 36, 39, 62] because its ground truth explanations enable a simple interpretation for human-based evaluation. The comparison of PowerGraph to the BA-2Motifs dataset allows us to verify if our results align with state-of-the-art research on the explainability of GNN.

Human-based evaluation To evaluate the generated explanations, we use the balanced accuracy metric. It compares the generated edge mask to the ground-truth cascading edges and takes into account the class imbalance, i.e., cascading edges are a small fraction of the total edges. It measures how convincing the explanations are to humans. More details about this metric are given in Appendix A.4. We report the performance of 11 explainability methods on finding ground-truth explanations. All results are averaged on five random seeds. Accuracy scores are computed for the datasets in PowerGraph and the synthetic dataset BA-2Motifs.

Model-centric evaluation Human evaluation is not always practical because it requires ground truth explanations and can be very subjective, and therefore does not necessarily account for the model's reasoning. Model-focus evaluation however measures the consistency of model predictions w.r.t removing or keeping the explanatory graph entities. For more objective evaluation, we therefore evaluate the faithfulness of the explanations using the fidelity+ metric. The fidelity+ measures how necessary are the explanatory edges to the GNN predictions. For PowerGraph, edges with high fidelity+ are the ones necessary for the graph to belong to Category A. We compare the PowerGraph results with BA-2Motifs results, using the fidelity+ metric fid_+^{acc}. The fid_+^{acc} is computed as in the GraphFramEx framework [5] and described in Appendix A.5. We utilize GraphFramEx to compare explainability methods: we choose the *phenomenon* focus and the masks to be *soft* on the edges. Explanations are weighted explanatory subgraphs, where edges are given importance based on their contribution to the true prediction in the multi-class setting. Figure 4 reports the fidelity+ scores for the power grid datasets and for the synthetic dataset BA-2Motifs.

Results Figure 3 shows that the best-balanced accuracies are obtained with the four methods, i.e., Saliency, Integrated Gradient, GradCAM, and Occlusion. Figure 4 also shows that these four methods have on average the highest fidelity+ on all datasets. Therefore, we conclude that they are the most appropriate methods to generate accurate and necessary explanations. Our observations on faithfulness are also consistent with previous results on the GraphFramEx benchmark [5] that has already shown the superiority of gradient-based methods and Occlusion to return necessary explanations, i.e., the model predictions change when those explanatory entities are removed from the graph. However, in Figure 3 and Figure 4, no method globally outperforms the others for all datasets. For balanced accuracy, GradCAM and Occlusion are the best for IEEE24; Saliency for IEEE39; GradCAM for UK; and Integrated Gradient, Occlusion, GradCAM and SubgraphX for BA-2Motifs. On fidelity, GradCAM and Occlusion are the best for IEEE24; Saliency and Integrated Gradient for IEEE39; GradCAM for UK; and Integrated Gradient for BA-2Motifs. The choice of the optimal xAI method depends on the dataset. This is again consistent with the conclusions in [5]. Concerning the IEEE118 dataset, none of the methods is able to generate good explanations. The maximum top balanced accuracy is 0.55 and the maximum fidelity+ score is reached by GNNExplainer on edges and

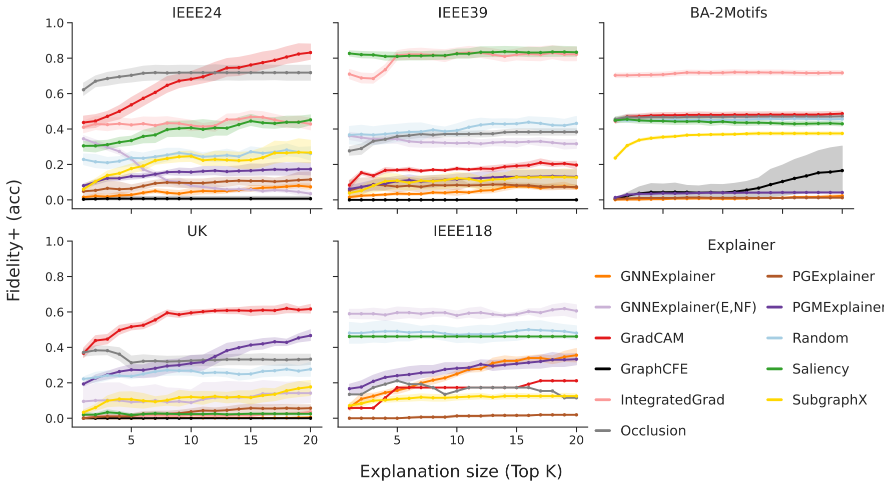

Figure 4: Faithfulness of the PowerGraph datasets and the BA-2Motifs dataset measured with the $fid+^{acc}$ metric as defined in Equation 2 in Appendix A.5. We conducted experiments on five random seeds. In the plot, alongside each data point, we have included confidence intervals calculated based on the standard deviation.

node features and is only 0.6. This performance is likely due to the complexity of the IEEE118. Being the largest power grid with 186 branches (see Table 1), the system contains complex interdependencies between the elements of the power grid during a cascading failure. As a consequence, node and edge-level features play a bigger role in explaining the GNN predictions. Therefore, we believe that an accurate model explanation will be obtained only with methods that provide node and link-level feature masks as well as edge masks. In addition, those methods could play a role in understanding the relevance of the input features to the GNN prediction, allowing to discard noisy features.

6 Conclusions

To strengthen the use of GNN in the field of power systems, we present PowerGraph, a dataset for graph-level tasks and model explainability. The dataset is suited to test graph classification and regression models. The main focus of PowerGraph is the analysis of cascading failures in power grids. Furthermore, experts often require interpretability of the results. Therefore, we benchmark the dataset for a variety of GNN and explainability models. The GNN models show excellent performance, in particular for graph classification, on our new benchmark, while graph regression models should be further developed. Finally, PowerGraph is the first real-world dataset with ground-truth explanations for graph-level tasks in the field of explainable AI. It allows us to evaluate both the accuracy and faithfulness of explainability methods in a real-world scenario. PowerGraph provides consistent outcomes that align with previous research findings and reinforce the concept that there is no universally superior method for explainability. In future work, we aim to extend the PowerGraph with new datasets [9] and include additional power grid analyses, including solutions to the power flow, the optimal power flow, and the unit commitment.

References

[1] Morteza Abedi, Mohammad Reza Aghamohammadi, and Mohammad Taghi Ameli. Svm based intelligent predictor for identifying critical lines with potential for cascading failures using pre-outage operating data. *International Journal of Electrical Power & Energy Systems*, 136:107608, 3 2022.

[2] Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, 2023.

[3] Chirag Agarwal, Marinka Zitnik, and Himabindu Lakkaraju. Probing gnn explainers: A rigorous theoretical and empirical analysis of gnn explanation methods. In *International Conference on Artificial Intelligence and Statistics*, pages 8969–8996. PMLR, 2022.

[4] Ehsan Aliyan, Mohammadreza Aghamohammadi, Mohsen Kia, Alireza Heidari, Miadreza Shafie-khah, and João P.S. Catalão. Decision tree analysis to identify harmful contingencies and estimate blackout indices for predicting system vulnerability. *Electric Power Systems Research*, 178, 1 2020.

[5] Kenza Amara, Rex Ying, Zitao Zhang, Zhihao Han, Yinan Shan, Ulrik Brandes, Sebastian Schemm, and Ce Zhang. Graphframex: Towards systematic evaluation of explainability methods for graph neural networks. *arXiv preprint arXiv:2206.09677*, 2022.

[6] G. Andersson, P. Donalek, R. Farmer, N. Hatziargyriou, I. Kamwa, P. Kundur, N. Martins, J. Paserba, P. Pourbeik, J. Sanchez-Gasca, R. Schulz, A. Stankovic, C. Taylor, and V. Vittal. Causes of the 2003 major grid blackouts in north america europe, and recommended means to improve system dynamic performance. *IEEE Transactions on Power Systems*, 20(4):1922 – 1928, 2005. Cited by: 995.

[7] et al. B. Gjorgiev. Cascades platform. 2019.

[8] Federico Baldassarre and Hossein Azizpour. Explainability techniques for graph convolutional networks. May 2019.

[9] Adam B. Birchfield, Ti Xu, Kathleen M. Gegner, Komal S. Shetye, and Thomas J. Overbye. Grid structural characteristics as validation criteria for synthetic networks. *IEEE Transactions on Power Systems*, 32(4):3258–3265, 2017.

[10] H.R.E.H. Bouchekara. Optimal power flow using black-hole-based optimization approach. *Applied Soft Computing*, 24:879–888, nov 2014.

[11] Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M. Buhmann. The balanced accuracy and its posterior distribution. In *2010 20th International Conference on Pattern Recognition*, pages 3121–3124, 2010.

[12] B. A. Carreras, V. E. Lynch, I. Dobson, and D. E. Newman. Critical points and transitions in an electric power transmission model for cascading failure blackouts. *Chaos*, 12:985–994, 2002.

[13] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch: A scientific computing framework for luajit. *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 24:237–245, 2011.

[14] Swiss National Supercomputing Centre (CSCS). Euler wiki. https://scicomp.ethz.ch/wiki/Euler, 2023. [Accessed: April 26, 2023].

[15] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[16] Texas A&M University Engineering. Ieee 118-bus system. https://electricgrids.engr.tamu.edu/electric-grid-test-cases/ieee-118-bus-system/.

[17] Texas A&M University Engineering. Ieee 24-bus system. https://electricgrids.engr.tamu.edu/electric-grid-test-cases/ieee-24-bus-system/.

[18] Texas A&M University Engineering. New england ieee 39-bus system. https://electricgrids.engr.tamu.edu/electric-grid-test-cases/new-england-ieee-39-bus-system/.

[19] Lukas Faber, Amin K Moghaddam, and Roger Wattenhofer. When comparing to ground truth is wrong: On evaluating GNN explanation methods.

[20] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[21] Scott Freitas, Yuxiao Dong, Joshua Neil, and Duen Horng Chau. A large-scale database for graph representation learning.

[22] Blazhe Gjorgiev, Alexander E. David, and Giovanni Sansavini. Cascade-risk-informed transmission expansion planning of ac electric power systems. *Electric Power Systems Research*, 204:107685, 2022.

[23] Blazhe Gjorgiev and Giovanni Sansavini. Identifying and assessing power system vulnerabilities to transmission asset outages via cascading failure analysis. *Reliability Engineering & System Safety*, 217:108085, 2022.

[24] Blazhe Gjorgiev, Andrej Stankovski, Giovanni Sansavini, Bing Li, and Alexander David. Cascades a platform for power system risk analyses and transmission expansion planning.

[25] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[26] Hassan Haes Alhelou, Mohamad Esmail Hamedani-Golshan, Takawira Cuthbert Njenda, and Pierluigi Siano. A survey on power system blackout and cascading events: Research motivations and challenges. *Energies*, 12(4), 2019.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

[28] Eugen Hruska, Liang Zhao, and Fang Liu. Ground truth explanation dataset for chemical property prediction on molecular graphs. 2022.

[29] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs, 2021.

[30] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs, 2020.

[31] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks, 2020.

[32] José Jiménez-Luna, Miha Skalic, and Nils Weskamp. Benchmarking molecular feature attribution methods with activity cliffs. *Journal of Chemical Information and Modeling*, 62(2):274–283, 2022.

[33] Jinwoo Kim, Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. *Advances in Neural Information Processing Systems*, 35:14582–14595, 2022.

[34] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

[35] Bing Li, Blazhe Gjorgiev, and Giovanni Sansavini. Meta-heuristic approach for validation and calibration of cascading failure analysis. In *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 1–6, 2018.

[36] Peibo Li, Yixing Yang, Maurice Pagnucco, and Yang Song. Explainability in graph neural networks: An experimental survey. *arXiv preprint arXiv:2203.09258*, 2022.

[37] Wenlong Liao, Birgitte Bak-Jensen, Jayakrishnan Radhakrishna Pillai, Yuelong Wang, and Yusen Wang. A review of graph neural networks and their applications in power systems, 2021.

[38] Yuxiao Liu, Ning Zhang, Dan Wu, Audun Botterud, Rui Yao, and Chongqing Kang. Guiding cascading failure search with interpretable graph convolutional network. 1 2020.

[39] Antonio Longa, Steve Azzolin, Gabriele Santin, Giulia Cencetti, Pietro Liò, Bruno Lepri, and Andrea Passerini. Explaining the explainers in graph neural networks: a comparative study. *arXiv preprint arXiv:2210.15304*, 2022.

[40] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. November 2020.

[41] Shengjie Luo, Tianlang Chen, Yixian Xu, Shuxin Zheng, Tie-Yan Liu, Liwei Wang, and Di He. One transformer can understand both 2d & 3d molecular data. *arXiv preprint arXiv:2210.01765*, 2022.

[42] Jing Ma, Ruocheng Guo, Saumitra Mishra, Aidong Zhang, and Jundong Li. Clear: Generative counterfactual explanations on graphs. *arXiv preprint arXiv:2210.08443*, 2022.

[43] Steffen Meinecke, Džanan Sarajlić, Simon Ruben Drauz, Annika Klettke, Lars-Peter Lauven, Christian Rehtanz, Albert Moser, and Martin Braun. Simbench—a benchmark dataset of electric power systems to compare innovative solutions based on power flow analysis. *Energies*, 13(12):3290, June 2020.

[44] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs, 2020.

[45] NationalgridESO. https://data.nationalgrideso.com.

[46] Christian Nauck, Michael Lindner, Konstantin Schürholt, Haoming Zhang, Paul Schultz, Jürgen Kurths, Ingrid Isenhardt, and Frank Hellmann. Predicting basin stability of power grids using graph neural networks. *New Journal of Physics*, 24(4):043041, apr 2022.

[47] Dusko P. Nedic, Ian Dobson, Daniel S. Kirschen, Benjamin A. Carreras, and Vickie E. Lynch. Criticality in a cascading failure blackout model. *International Journal of Electrical Power and Energy Systems*, 28:627–633, 2006.

[48] NVIDIA Corporation. NVIDIA CUDA Toolkit. https://developer.nvidia.com/cuda-toolkit, 2023.

[49] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

[50] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3):1–21, 03 2015.

[51] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. October 2016.

[52] Yunsheng Shi, Zhengjie Huang, Wenjin Wang, Hui Zhong, Shikun Feng, and Yu Sun. Masked label prediction: Unified massage passing model for semi-supervised classification. *CoRR*, abs/2009.03509, 2020.

[53] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. March 2017.

[54] Anna Varbella, Blazhe Gjorgiev, and Giovanni Sansavini. Geometric deep learning for online prediction of cascading failures in power grids. *Reliability Engineering & System Safety*, 237:109341, 2023.

[55] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[56] Clément Vignac, Andreas Loukas, and Pascal Frossard. Building powerful and equivariant graph neural networks with structural message-passing. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14143–14155. Curran Associates, Inc., 2020.

[57] Minh N Vu and My T Thai. PGM-Explainer: Probabilistic graphical model explanations for graph neural networks. October 2020.

[58] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: A benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018.

[59] Arbel Yaniv, Parteek Kumar, and Yuval Beck. Towards adoption of gnns for power flow applications in distribution systems. *Electric Power Systems Research*, 216:109005, 2023.

[60] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. GNNExplainer: Generating explanations for graph neural networks. *Adv. Neural Inf. Process. Syst.*, 32:9240–9251, December 2019.

[61] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. December 2020.

[62] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(5):5782–5799, 2022.

[63] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. February 2021.

[64] Xiangtian Zheng, Nan Xu, Loc Trinh, Dongqi Wu, Tong Huang, S. Sivaranjani, Yan Liu, and Le Xie. Psml: A multi-scale time-series dataset for machine learning in decarbonized energy grids (code), November 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4 paragraph 'Discussion', 5 paragraph 'Results' and the Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments (e.g., for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Section B.2
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4 paragraph 'Experimental setting and evaluation metrics' and 5 paragraph 'Experimental setting and datasets'.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 5 paragraph 'Results'.
   (d) Did you include the total amount of computing and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4 paragraph 'Experimental setting and evaluation metrics' and 5 paragraph 'Experimental setting and datasets'.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 2
   (b) Did you mention the license of the assets? [Yes] See Section B.4
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See Section B.2
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Section B.3
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Supplementary materials

A.1 OGB taxonomy of graph datasets

The Open Graph Benchmark [30] contains a diverse set of real-world datasets of various sizes and operational specifics. It contains medium to large-scale datasets that can be used to feed data-hungry models like GNN. For node and link property prediction tasks, OGB has datasets in all domains, *i.e.*, nature, society, and information. However, Table 7 shows the absence of graph datasets in the society domain. To fill this gap, we propose PowerGraph, the first collection of real datasets in the *society* domain.

Table 7: OGB taxonomy for graph datasets.

	Property prediction task		
Domain	**Node**	**Link**	**Graph**
Nature	`proteins`	`ddi`, `ppa`	`molhiv`, `molpcba/ppa`
Society	`arxiv`, `products`, `papers100M`	`biokg`, `wikikg2`	-
Information	`mag`	`collab`, `citation2`	`code2`

A.2 Class targeted explanations

For benchmarking explanations in section 5, we focus on explaining Category A graphs of the multi-class problem, i.e., the power grids that fail to serve the demand (DNS>0). The objective is to shed light on the lines that are tripped after the first contingency. We use the multi-class problem rather than the binary classification problem that classifies states according to the demand not served (DNS) only, i.e. distinguishes power grids that serve the demand (DNS=0, label 1) from the ones that do not (DNS>0, label 0). In the multi-class problem, the model learns to distinguish cascading failure scenarios, while in the binary setting, Category A and B are considered the same type of grids (class DNS>0). Choosing to explain DNS>0 in the multi-class problem allows us to focus on the case where some lines are tripped when DNS>0 and therefore expect the model to learn the cascading edges for this class of grids.

A.3 Explainability methods

To explain the decisions made by the GNN models, we adopt different classes of explainers including gradient/feature-based methods and perturbation-based methods. In our experiments, we compare the following methods: **Random** gives every edge and node feature a random value between 0 and 1; **Saliency (SA)** measures node importance as the weight on every node after computing the gradient of the output with respect to node features [8]; **Integrated Gradient (IG)** avoids the saturation problem of the gradient-based method Saliency by accumulating gradients over the path from a baseline input (zero-vector) and the input at hand [53]; **Grad-CAM** is a generalization of class activation maps (CAM) [51]; **Occlusion** attributes the importance of an edge as the difference of the model initial prediction on the graph after removing this edge [19]; **GNNExplainer (E,NF)** computes the importance of graph entities (node/edge/node feature) using the mutual information [60]; We also use **GNNExplainer** that considers only edge importance; **PGExplainer** is very similar to GNNExplainer, but generates explanations only for the graph structure (nodes/edges) using the re-parameterization mechanism to overcome computation intractability [40]; **PGM-Explainer** perturbs the input and uses probabilistic graphical models to find the dependencies between the nodes and the output [57]; **SubgraphX** explores possible explanatory sub-graphs with Monte Carlo Tree Search and assigns them a score using the Shapley value [63]; and **GraphCFE** leverages a graph variational autoencoder to generate counterfactual explanations for graphs [42].

Model-aware. Gradient-based methods compute the gradients of target prediction with respect to input features by back-propagation. Features-based methods map the hidden features to the input space via interpolation to measure important scores. Decomposition methods measure the importance of input features by distributing the prediction scores to the input space in a back-propagation manner.

Model-agnostic. Perturbation-based methods use masking strategy in the input space to perturb the input. Surrogate models use node/edge dropping, BFS sampling and node feature perturbation. Counterfactual methods generate counterfactual explanations by searching for a close possible world using adversarial perturbation techniques [25].

Table 8: Explainability methods tested on the PowerGraph benchmark.

Explainer	Model-aware/agnostic	Target	Type	Flow
SA	Model-aware	N/E	Gradient	Backward
IG	Model-aware	N/E	Gradient	Backward
Grad-CAM	Model-aware	N	Gradient	Backward
Occlusion	Model-agnostic	N/E	Perturbation	Forward
GNNExplainer	Model-agnostic	N/E/NF	Perturbation	Forward
PGExplainer	Model-agnostic	N/E	Perturbation	Forward
PGM-Explainer	Model-agnostic	N/E	Perturbation	Forward
SubgraphX	Model-agnostic	N/E	Perturbation	Forward

A.4 Balanced accuracy

Definition The balanced accuracy is the arithmetic mean of the specificity and the sensitivity. The sensitivity or true positive rate or recall measures the proportion of real positives that are correctly predicted out of all positive predictions that could be made by the model. The specificity or true negative rate measures the proportion of correctly identified negatives over the total negative predictions that could be made by the model. The balanced accuracy is then expressed as:

$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2} = \frac{1}{2} \cdot \left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP} \right) \tag{1}$$

The balanced accuracy has the advantage of accounting for imbalance in the explanatory mask. In the context of cascading failure detection, we know that most of the components (links) in the grid will not fail. Therefore, the edge mask has many values that are zeros and only a few that are ones. The balanced accuracy measures if the method was able to recognize both failing and not failing edges, while giving the same importance to both detections.

A.5 Faithfulness metric

To measure the faithfulness of the explanations, we use either the fidelity- or the fidelity+ scores defined in [61, 5]. We evaluate the contribution of the produced explanatory subgraph to the initial prediction, either by giving only the subgraph as input to the model (fidelity-) or by removing it from the entire graph and re-run the model (fidelity+). As explained in section A.2, the generated explanations in the context of PowerGraph are the tripped lines and therefore should be necessary but not sufficient to the grid class. Indeed, the subgraph resulting from isolating the cascading branches does not represent a power grid. Therefore, fidelity- is not relevant in the context of the PowerGraph benchmark and we evaluate the faithfulness of explanations using the fidelity+ metric defined in equations 2 and 3. The fidelity score can be expressed either with probabilities (fid_+^{prob}) or indicator functions (fid_+^{acc}). We adopt the fid_+^{acc}, as it is more suitable for classification models. f is a pre-trained classifier. We denote by $\hat{y}_i$ and $\hat{y}_i^{G_{C\setminus S}}$ the model's predictions when taking as input respectively the input graph G_C and its complement or masked-out graph $G_{C\setminus S}$.

$$fid+^{acc} = \frac{1}{N} \sum_{i=1}^{N} \left| \mathbb{1}(\hat{y}_i = y_i) - \mathbb{1}(\hat{y}_i^{G_{C\setminus S}} = y_i) \right| \tag{2}$$

$$fid+^{prob} = \frac{1}{N} \sum_{i=1}^{N} (f(G_C)_{y_i} - f(G_{C\setminus S})_{y_i}) \tag{3}$$

A.6 Inductive property of GNN models on PowerGraph

We conducted an out-of-distribution test by training GNN models on one power grid dataset and applying the model on a different power grid dataset. GNNs allow to train models that can be tested on grids with different topologies, as long as we feed the same number of features per node and edge. This attribute is often referred to as inductive learning property [56]. We report the results in Tables 9, 10, 11. Table 9 shows that the binary classifier models trained on IEEE39, IEEE118, and UK datasets perform well on most datasets, except when tested on the IEEE24. Indeed, with a test balanced accuracy of 50%, these models are not able to identify patterns in IEEE24 and instead randomly assign instances to a class. Similarly, Table 10 indicates that the multiclass classification model trained on the IEEE39 achieves good performance across other power grid datasets, and in particular with the UK and IEEE118 datasets. However, Table 11 shows that the regression models yield identical MSE errors for all test sets. This behavior stems from the regression model assigning the same DNS value to all instances, indicating an inability to capture any structure in the test dataset. Overall, we conclude that the GNN models obtained from PowerGraph do not show robust results when applied on a different power grid dataset that the model did not observed during training.

Table 9: Out-of-distribution balanced accuracies of binary classification models. The selected model is the best performing model based on the Transformer MPL.

Trained on \Tested on	IEEE24 Binary	IEEE39 Binary	UK Binary	IEEE118 Binary
IEEE24 Binary	0.99	0.35	0.25	0.30
IEEE39 Binary	0.50	0.96	0.75	0.70
UK Binary	0.50	0.65	0.98	0.70
IEEE118 Binary	0.50	0.67	0.77	0.99

Table 10: Out-of-distribution balanced accuracies of multiclass classification models. The selected model is the best performing model based on the Transformer MPL.

Trained on \Tested on	IEEE24 Multiclass	IEEE39 Multiclass	UK Multiclass	IEEE118 Multiclass
IEEE24 Multiclass	0.98	0.071	0.12	0.0018
IEEE39 Multiclass	0.45	0.97	0.66	0.76
UK Multiclass	0.0072	0.048	0.98	0.067
IEEE118 Multiclass	0.0072	0.048	0.22	0.98

Table 11: Out-of-distribution MSE errors of regression models. The selected model is the best performing model based on the Transformer MPL.

Trained on \Tested on	IEEE24 Regression	IEEE39 Regression	UK Regression	IEEE118 Regression
IEEE24 Regression	2.70E-03	3.81E-04	3.81E-04	3.81E-04
IEEE39 Regression	1.73E-04	5.47E-04	1.73E-04	1.73E-04
UK Regression	9.89E-05	9.89E-05	2.34E-03	9.89E-05
IEEE118 Regression	9.44E-08	9.44E-08	9.44E-08	5.00E-06

B Access to PowerGraph dataset

B.1 Dataset documentation and intended uses

PowerGraph is the collection of the following GNN datasets: UK, IEEE24, IEEE39, IEEE118 power grids. We use `InMemoryDataset` [20] class of Pytorch Geometric, which processes the raw data obtained from the Cascades [7] simulation. For each dataset UK, IEEE24, IEEE39, IEEE118, we provide a folder containing the raw data organized in the following files:

- `blist.mat`: branch list also called edge order or edge index
- `of_bi.mat`: binary classification
- `of_reg.mat`: regression labels
- `of_mc.mat`: multi-class labels
- `Bf.mat`: node feature matrix
- `Ef.mat`: edge feature matrix
- `exp.mat`: ground-truth explanation

B.2 Download dataset

The dataset can be viewed and downloaded by the reviewers from https://figshare.com/articles/dataset/PowerGraph/22820534 (~1.8GB, when uncompressed):

```
#!/bin/bash
wget -O data.tar.gz "https://figshare.com/ndownloader/files/40571123"
tar -xf data.tar.gz
```

B.3 Author statement

The authors state here that they bear all responsibility in case of violation of rights, etc., and confirm that this work is licensed under the CC BY 4.0 license.

B.4 Hosting, licensing, and maintenance plan

The code to obtain the PowerGraph dataset in the `InMemoryDataset` [20] format and to benchmark GNN and explainability methods is available as a public GitHub repository at https://anonymous.4open.science/r/PowerGraph/. The authors are responsible for updating the code in case issues are raised and maintaining the datasets. We aim to extend the PowerGraph with new datasets and include additional power grid analyses, including solutions to the power flow, the optimal power flow, and the unit commitment. Over time we plan to release new versions of the datasets and provide updates to the results for both the GNN accuracy and the explainability analysis. In addition, the code will be updated if new pytorch/torch-geometric versions are released or crucial python packages are updated. The data is hosted on figshare at https://figshare.com/articles/dataset/PowerGraph/22820534. The authors give public free access to the PowerGraph dataset. The datasets are identified with the DOI:10.6084/m9.figshare.22820534. The work in this paper (code, data) is licensed under the CC BY 4.0 license.

B.5 Code implementation

We run a hyper-parameters grid search over different GNN models, using torch-geometric 2.3.0 [20] and Torch 2.0.0 with CUDA version 11.8 [13, 48]. The experiments to benchmark graph classification and regression models are performed on a Windows machine with 3 GPUs NVIDIA RTX A6000 with 128 GB RAM memory. For the explainability analysis, experiments are conducted on 8 AMD EPYC 7742 CPUs with a memory of 5GB each on the ETH Euler clusters [14].