Inductive Graph Neural Networks for Power Systems

Àlex De Los Santos Subirats , **Francesco Piccoli** , **Marco van Veen** and **Marcus Plesner**
TU Delft
{a.delossantossubirats, f.piccoli, r.j.vanveen, m.k.plesner}@student.tudelft.nl

Abstract

Continuous changes in the topology of power grids due to factors such as faults, the integration of renewable energy sources, and system expansions make it increasingly challenging to promptly monitor the state, evaluate conditions, and effectively address the issues that can arise within a power system. In this dynamic environment, methods capable of rapidly inferring the grid's state with accuracy and adaptability are of crucial importance. In our project, we leverage Graph Neural Networks to devise an inductive framework that is applicable to common power systems operational calcualations, such as power flow. The framework aims to provide fast and accurate predictions despite evolving topology of the power grid.

1 Introduction

In the last decade, the field of power grid analysis has seen significant improvements in several tasks through the use of artificial intelligence and Deep Neural Networks [2; 14]. More recently, Graph Neural Networks (GNNs) have emerged as a highly suitable architecture for handling non-Euclidean data, including power systems data. Graph-structured data plays an important role in various applications within the field of power systems research, and is suitable in the context of machine learning methods. Applications of graph structured data include, for instance, optimal power flow (OPF) and power flow analysis. GNNs have the capability to effectively leverage the grid's physical topology and model the complex interdependence between nodes [2; 14]. Additionally, GNNs are less sensitive to noise, and inference is significantly faster compared to traditional numerical solvers [18]. While electrical power grids are still a relatively unexplored domain in Graph Neural Network research [18], recent studies have begun to shed light on this field [14]. These studies generally operate within a transductive setting, where the GNNs are trained and tested on the same graph topology. However, the topology of modern power grids undergoes frequent changes due to various factors, including faults, integration of new renewable energy sources and expansions of the grids. These factors necessitate the model's ability to adapt seamlessly to these changes. In a transductive setting, this adaptation requires the retraining of the model on the updated topology, incurring substantial computational and time costs.

The goal of our project[1] is to devise an inductive GNN framework for power grids, capable of accurate predictions on unseen or modified graphs, without retraining. The framework aims at providing a common inductive method for different power system operational calculations, for instance, power flow analysis and optimal power flow. To achieve this, we employ a three-fold approach. First, we create a dataset of graphs, where each graph is a subgraph (i.e. a valid subgrid) of a power grid. This allows the model to better learn local patterns that generalize across grids, instead of learning single grid-specific patterns [10; 26]. Secondly, we implement a physics-based loss function which minimizes Kirchoff's law violations, in an unsupervised learning setting. A physics-based loss is free from the need to imitate the output of a classical numerical solver, as the Newton-Raphson solver, and shows promising inductive capabilites [3; 6; 15]. Finally, we experiment on a variety of convolution operations of architectures that have proven to have good inductive performance as GraphSAGE [10] and Graph Attention Network (GAT) [22] and on architectures that have a high expressive power such as the Graph Isomorphism Network (GIN) [24].

2 Related Work

2.1 GNNs for power grids

Graph Neural Networks have emerged as a valuable asset for power systems analysis and for computational operations on the topology of the grid [14].

In [2] the authors apply GNNs to approximate Alternating Current (AC) power flow to provide accurate load flow approximations, with fast convergence and great performance specifically in the case of large grids.

Another study [16] tackles the OPF problem using GNNs in an imitation learning framework. The proposed method efficiently approximates AC-OPF solutions, and provides a scalable and effective method for the OPF problem on large grids.

GNNs have also been applied to tackle the challenge of power flow balance within energy grids. In [1] the authors

[1] Project repository: https://github.com/AlexDeLos/GNN_OPF

train a GNN to predict current and power injections at grid branches, maintaining power flow equilibrium.

While often achieving precision, adaptability, and fast inference time, it's worth noting that all the aforementioned approaches operate in a transductive setting, and they face limitations in generalizing to unseen or modified real-life power grids.

A notable study [6] focuses on a GNN-based architecture for AC power flow computations. This architecture directly minimizes Kirchhoff's law violations, avoiding the need to imitate Newton-Raphson solvers. The authors show that their physics-based approach has good inductive performance, being robust to topology perturbations. While promising, the inductive performance are tested only on small graphs and requires further empirical validation on real-life grids to assess scalability.

In [3] the authors claim to achieve higher generalization performance than [6] in approximating AC power flow. They also utilize a physics-informed loss minimization and they achieve greater accuracy by fine-tuning the model on a particular topology. However their inductive results are still far from the performance of a numerical solver, such as Newton-Rhapson.

Another approach [15] uses typed GNNs to solve the AC power flow problem. As in [3] and [6], the training is done in an unsupervised manner by minimizing violation of the physical laws that govern the system. While the authors claim good inductive performance, their tests are limited to a small number of different grids, with most of them being relatively small. Furthermore, they assume that the reactive powers of the generators are locally compensated, which means that they seem to be ignoring the reactive power imbalances for these busses.

In [18] the authors also investigate inductive approaches for the power grid state estimation problem. They propose a very deep model, claiming that that deep architectures are crucial for real-world power grids to handle long range dependencies introduced by physical power flows. To overcome the over-smoothing problem in deep GNN architectures [5], they propose a Jumping Knowledge mechanism [25], based on a Transformer [21]. Despite their method demonstrates robust performance when handling noisy data, it does not show good inductive performance. Additionally, the paper shows how power grids contain unique topological features making them significantly different from standard benchmark datasets on which most (inductive) architectures are tested.

These limitations and insights underscore the need for further research and development in the field, emphasizing the importance of our ongoing project. Moreover, to the best of our knowledge, the vast majority of the above works do not provide code supporting their results. We provide our code, in the hope that future research will build upon our work. Finally, for a comprehensive exploration of GNNs in power grids, we recommend referring to [14].

2.2 Inductive GNNs

Inductive Graph Neural Networks (GNNs) have gained importance in various domains, due to their ability to generalize to unseen data and adapt to changing graph topologies, making them a valuable tool in scenarios where the graph structure are constantly evolving.

One crucial development in the field was the GraphSAGE architecture [10]. The approach leverages node feature information to efficiently generate embeddings for previously unseen nodes. Unlike traditional transductive approaches, it learns local graph structures by training on subgraphs by sampling from each node's neighbourhood at each convolution, for each node. This allows the model to capture local patterns and generalize to new graph topologies.

Another study proposes an inductive GNN architecture (IGNNK) for spatiotemporal kriging [23]. IGNNK tackles the challenge of recovering data for unsampled sensors on a network or graph structure. It leverages inductive learning by generating random subgraphs as samples and reconstructing adjacency matrices for each sample. This approach effectively learns the spatial message passing mechanism, enabling it to generalize to new locations without requiring full retraining.

In GraphSAINT [26], the authors propose a scalable model with good inductive performance. The method deviates from traditional approaches by constructing minibatches through graph sampling.

Despite all the above efforts, inductivity in GNNs is still an open problem and an active area of research, no current approach is able to generalize well in all GNNs domains.

3 Methodology

In the following sections, we provide a comprehensive breakdown of our methodology, covering data collection, augmentation, and preprocessing, graph representation, the GNN convolutational operations we used, our training strategies, loss functions and our overall experimental design. This methodology not only addresses the unique characteristics of power grids, but also aims to contribute to the broader field of inductive GNN research.

3.1 Dataset

We utilize the PandaPower[2] library in Python to obtain the data for our experiments. PandaPower provides numerous built-in Power System Test Cases[3], ranging from small networks with few busses to large power systems, including the Iceland and Great Britain power grids. We focused our tests on the standard IEEE power grid case118, as it is a medium-size grid, frequently tested in many papers. Since we are using the power flow problem as a case study for our inductive framework, we also harnessed the library's built-in numerical solver to generate the expected power flow outputs (i.e. labels) for each network.

To achieve inductive capabilities, we train our models on subgrids ranging from 50 to 70 nodes (out of the total 118 nodes in case118), and not on the full power grids. The goal is to let the model learn local patterns that repeat across different graphs [10; 23; 26]. We describe the subgraphing methods

[2]PandaPower is an open-source Python tool: http://www.pandapower.org/

[3]PandaPower Power System Test Cases: https://pandapower.readthedocs.io/en/v2.13.1/networks/power_system_test_cases.html

we utilized in subsection 3.2. The models are validated and tested on the full case118 power grid topology, to evaluate their inductive performances.

Furthermore, we perform data augmentation on the graphs by applying a uniformly distributed scaling factor to the active and reactive powers of all load busses, such that the resulting values fall within 80% to 120% of the original values [15; 16; 17]. Additional Gaussian noise of the form $X \sim \mathcal{N}(0,\, 0.05)$ is applied to each load to allow for some individual variation besides the overall uniform variation. The active powers of the generators are then multiplied by the average change in power over all the load busses to ensure that the total power supplied is approximately in line with the total power demand after the random perturbations. In total, we create 2000 subgrids for training, which are then expanded to 10.000 through data augmentation. The validation and test sets contain 1000 instances each, where each instance is obtained via data augmentation on the original full grid.

Since we use the PyTorch Geometric library to create our heterogenous GNN models, we converted the PandaPower network objects into PyTorch Geometric *HeteroData* objects. We also experimented with modeling the graph as homogeneous data, but we found that it had worse performance compared to its heterogeneous counterpart. The network busses are then represented as nodes of different types in the graph, depending on which elements are connected to each bus. The different node types are loads, generators, load-generators, and external grids. The input values for loads are their fixed active (*p_mw*) and reactive (*q_mvar*) powers, and the labels are the missing voltage magnitudes (*vm_pu*) and angles (*va_degree*). Generators have fixed active power and voltage magnitudes as inputs, while the labels are the reactive power and voltage angle. Load-generator nodes are busses which have both loads and generators connected to them, so these nodes have active and reactive power, and voltage magnitudes as input. The labels are the (total) reactive power at the bus and the voltage angle. Finally, the external grid nodes have their fixed voltage magnitude and angles as inputs, which are used by the other nodes as a reference point for their voltage angle predictions.

The power lines of the grid form the (undirected) edges between the corresponding bus nodes. The features we consider for the lines are the conductances (*G*) and susceptances (*B*) of the line admittances $Y = G + jB$, since these features are explicitly used in the power flow equations [7, p. 351]. To obtain the admittance values, the given line impedances $Z = R + jX$ are used, where *R* is the line resistance in ohm per km (*r_ohm_per_km*) and *X* is the line reactance in ohm per km (*x_ohm_per_km*). These values are multiplied by the length of the line (*length_km*) before being used. We then calculate G and B using the definition of admittance: $Y = \frac{1}{Z}$.

Transformers connect high and low voltage busses in power grids. We represented them as edges in the graph. However, transformers have different features compared to the normal power lines, which makes converting them to edges, a nontrivial task. We represent them as lines by using only an approximation of the reactance:

$$x = SCV(\%)\frac{V_{base}^2}{S_{base}} \tag{1}$$

where $SCV(\%)$ is the short circuit voltage in percentages at the transformer, and V_{base} and S_{base} are the rated voltage and power, respectively. The reactance (and zero-valued resistance) are then converted to admittances using the same conversion as for lines.

Finally, many of the values provided in the PandaPower networks are given in different units and depend on which side of the transformer they occur. Therefore, all the values are converted to the per-unit system before they are used by the models. The voltage magnitudes are already in per-units and voltage angles do not need to be converted. The active and reactive power values are divided by the rated power of the network (*sn_mva*). The line impedances are converted to per-unit before calculating the admittances. This is done by dividing the impedances by the base impedance Z_{base}, where $Z_{base} = \frac{V_{base}^2}{S_{base}}$. V_{base} is the voltage level (*vn_kv*) at the low-voltage bus and S_{base} is the rated power of the network.

3.2 Subgraph Generation

In this section, we discuss the methodology for generating subgraphs used in the creation of training datasets for our study on power grids. We explored several subgraphing techniques to generate valid subgrid topologies. We could not find any reference discussing subgraphing on power grids, so all the statements and comments about each subgraphing technique are based on our previous knowledge, intuition, and empirical tests. All subgraphing methods start from a bus node connected to a generator, ensuring the presence of a power source in the subgrid. However, each method has its own distinctive characteristics:

- Random neighbor sampling is a method that selects a random node from the current subgraph and subsequently includes a randomly selected neighbor in the subgraph. As nodes are added randomly based on their connectivity, these subgraphs tend to exhibit a blend of both local and distant nodes. This feature provides a wide variety of topologies, which could aid the model in learning patterns in the widest possible variety of configurations.
- Random walks with restart, starts from a node (connected to a generator) and selects randomly one of its neighbors. At each iteration, the previously selected node becomes the focal point for the next neighborhood sampling. An essential parameter in this method is the restart probability: the higher, the more likely the subgraph will be centered around the initial node. We set the probability to a low value of 0.1. This allows for increased variation between the subgraphs generated, making it valuable for training machine learning models to handle real-world uncertainties encountered by power systems.
- Breadth first search (BFS) is a more systematic approach to constructing subgrids. Starting from a chosen generator, BFS traverses the network by exploring the immediate neighborhood node by node. It's worth mentioning that this method may result in subgraphs being centered around generators, potentially neglecting nodes distant from the generator.

3.3 Model Architecture

In our project, we explore various graph convolution operations which are used in existing GNN architectures, each with its own characteristics and advantages. To model different node types in the power grids (loads, generators, load-generators, and external grids), we utilize heterogeneous GNNs which support the use of different node types. Here we briefly present the different graph convolutions we experimented with:

- **GraphSAGE (Graph Sample and Aggregated)** architecture (and its convolution operation) is renowned for its inductive and scalable performance [10]. At its core, GraphSAGE employs a mechanism that involves randomly sampling neighboring nodes (instead of using all neighbors) and effectively aggregating their features. This sampling mechanism allows it to efficiently explore local graph structures while maintaining scalability. The default GraphSAGE model utilizes only node features for embedding generation and does not inherently incorporate edge features. In power grid analysis, edge features, such as line properties, are often essential for accurately modeling electrical flow dynamics and network behavior. We modified the built-in Pytorch geometric SAGE implementation, to include edge features.
- **Graph Isomorphism Network (GIN)** architecture was specifically crafted to tackle the challenging graph isomorphism problem [24]. The convolution operation utilized, learns node embeddings by employing a recursive approach, repeatedly applying a neural network layer followed by a set operation. GIN's strength lies in its high level of expressiveness, granting it the capability to effectively capture intricate and complex properties within various types of graphs. In the context of power grid analysis, we often deal with diverse and intricate grid structures, making GIN a good candidate to model such intricacies. Since line features are crucial in power grid analysis, we use a modified version of GIN which also takes into account edge features [11].

Other architectures such as Graph Attention Networks [22] and a general Message Passing Graph Neural Network [9] were considered and tested, but for the final experiments only the two convolution types mentioned above were used.

Our heterogeneous GNN architecture contains 2 convolutional layers consisting of the GIN convolution. For certain experiments, the GraphSAGE convolution was used instead of GIN. The Jumping Knowledge mechanism [25] with an LSTM aggregation scheme was used together with the convolutional layers. The final layer of the architecture consisted of a linear layer to map the convolution outputs to the expected output dimensions of each node type.

For training the model, the Adam optimiser [13] was used with a batch size of 32 and learning rate of 0.001 over 200 epochs. The training dataset consisted of subgraphs generated by the random walk method and the loss function used during training was the physics-informed loss (section 3.4). These settings were used by default, unless specified otherwise during the discussion of the experiments.

3.4 Loss Functions

In addition to experimenting with different architectures and subgraph-generating processes, we also test two different types of loss functions during training, namely a supervised and an unsupervised loss function. The models are given the same node features as input, as described in subsection 3.1, when using either of the two losses.

Voltage Phasor Supervised Loss

For the supervised loss setting, only the missing voltage magnitudes $|V|$ and angles θ are predicted, since the power values can be calculated from the voltages using the power flow equations. Using the predicted voltage values, we create the complex voltage phasors $V = |V|e^{j\theta} = |V|cos(\theta) + j|V|sin(\theta)$. We then calculate the Euclidean distances between the predicted and ground truth voltage phasors for each node and minimise the average distance. The benefit of using voltage phasor differences, compared to a more traditional loss such as the mean squared error, is that this loss explicitly combines the voltage magnitude and angles instead of treating them as two separate, potentially unrelated features. Highlighting the relationship between the two voltage features, thus, hopefully benefits the model's learning process for these two features.

Physics-based Unsupervised Loss

The second loss function is inspired by [3] and takes a different approach compared the supervised loss. It does not evaluate the model outputs against some given expected output from a numerical solver. Instead, this loss function directly calculates the power flow imbalance in the network using the fixed and predicted node features. The network is then trained to minimise this imbalance, with any loss larger than zero indicating some power flow imbalance within the network. The goal of this loss function is to help the model learn the underlying fundamental physical properties of the network using some additional knowledge about power systems to guide the model's training. The main benefit of using this approach is that the model can be trained in an unsupervised manner without the need for labelled data. Additionally, this can also increase the interpretability of the model, as it is clearer what the model is trying to learn compared to when the model is trying to copy a numerical solver [12].

The power flow equations used are taken from [7, p. 351]. They enforce that the power going into each node equals the power going out of the node in order to ensure a power equilibrium exists within the network. The equations are split into pairs of real and imaginary equations for the active and reactive powers, respectively, and consist of the following for each node k:

$$P_k = V_k \sum_{n=1}^{N} V_n \left[G_{kn}\cos(\delta_k - \delta_n) + B_{kn}\sin(\delta_k - \delta_n)\right]$$
$$Q_k = V_k \sum_{n=1}^{N} V_n \left[G_{kn}\sin(\delta_k - \delta_n) - B_{kn}\cos(\delta_k - \delta_n)\right] \tag{2}$$

where P_k and Q_k are the active and reactive powers of node k, V_k and δ_k are the voltage magnitude and angle, and G_{kn}

and B_{kn} are the conductance and susceptance of the line connecting nodes k and n. The models are then trained by minimising the differences between the left and right hand sides of the above equations by learning the values of the missing node features.

4 Experiments

In this section, we discuss the experiments that were performed to evaluate the performance of the models. The experiments cover subgraph generation techniques, loss functions, the utilization of edge features, and convolution types to assess the inductive performance of our framework.

4.1 Subgraphs Generation Methods

The purpose of this experiment is to determine which subgraph generation method creates topologies which lead to the best inductive performance in power system problems. The crucial element is the proclivity of the method to generate a wide variety of realistic topologies, which include a large variety of both common and uncommon topologies. The three methods utilized to generate subgraphs are: random neighbor, random walk, and breadth first search.

The experiment results are shown in Figure 1. The x-axis shows the thresholds for the prediction errors, which are calculated as the relative errors of the predictions with respect to the ground truth values in percentages. The y-axis then indicates the percentage of nodes that has errors less than or equal to each threshold value. We chose to use such figures for representing the results, because for power flow predictions to be useful, they have to be within a certain percentage of the true values. These types of graphs can quickly show how many nodes fall within acceptable error thresholds in order to assess the prediction qualities.

The figure shows that, overall, the voltage magnitude predictions are significantly better than voltage angle predictions. This is potentially due to the tight range of the targets. Voltage magnitude typically varies between 0.9 and 1.1, whereas voltage angle has a range of about thirty degrees. Another possible reason for the better voltage magnitude predictions is that the voltage angle is based around the difference between a node's angle and that of its neighbours, and these values are all relative to the fixed slack node angle from which everything is derived. This increases the difficulty of learning the correct voltage angles, as nodes that are further away from the slack bus will have to receive the information through more convolutions.

The results in the figure additionally show that breadth first search and random neighbor methods perform better for predicting voltage magnitude than random walks. However, the random walk method performs best, and breadth first search performs the worst for predicting angles. The result indicates that different subgraphing methods influences the inductive performance differently across features.

A possible explanation for the better angle predictions when using random walk subgraphs is the tendency of the random walk to generate networks which are less centered around the starting node, but instead are more likely to contain isolated paths, which lowers the average amount of

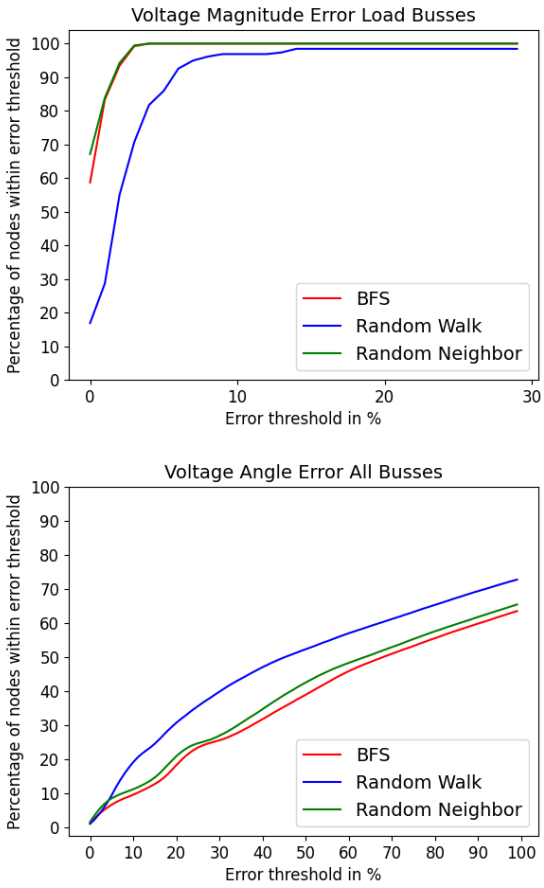

Figure 1: The voltage magnitude and angle prediction errors when using different subgraphing methods.

neighbours each node has compared to using random neighbours or BFS. Using networks which are more clustered around the starting node, such as in random neighbour and BFS, can lead to increased noise when learning the correct voltage angles, since the voltage angles are learnt through the power flow equations by taking the differences between a node's and all of its neighbour's angles. Because all angles (except the single slack bus angle) have to be predicted, and are thus not very accurate during the initial learning stages, having a larger amount of neighbours can make it more difficult to learn the correct voltage angles.

4.2 Effect of Different Loss Functions

A common approach to predicting the power flow values has been to use imitation learning to essentially mimic a numerical solver. Our expectation was that learning in such a way would limit the model's effectiveness to only producing relatively good results in transductive settings, as the model is not explicitly learning the fundamental underlying physical properties of power grids. Some previous works have attempted to use such unsupervised settings and claimed that it improved the inductive performances of the models [3; 6; 15]. So, we expected that an unsupervised, physics-informed loss based on the power flow equations that govern the power

grid dynamics would allow the model to explicitly learn the fundamental properties of a system, thus increasing its generalisability to networks with different topologies.

Figure 2 shows the results of experimenting with both a supervised and unsupervised loss in an inductive setting by training on the random walk subgraphs and testing both models on the full graphs. Besides the different loss functions, the two models are identical and make use of the GINE graph convolutions. The figure shows that both loss functions seem to achieve similar performance when predicting the voltage magnitudes. However, there is a noticeable performance difference in the prediction of the voltage angles. The voltage angles should be predicted relative to the angle of the slack bus, which is fixed to the default case118 value of 30 degrees in each sample. The supervised loss significantly outperforms the physics-informed loss, which contradicts our initial expectations. We believe that the lack of performance of the physics-informed loss is partially due to the way the voltage angles must be learned in this unsupervised setting. In the power flow equations (Equation 2) it can be seen that it is not the individual voltage angles that matter, but rather the differences between the voltage angles of each bus and its neighbours. Furthermore, all these angles must be learnt with respect to the fixed voltage angle of the single slack bus in the network. So, the correct angle differences must be learnt starting from the slack node and this information then has to be propagated throughout the entire network. We believe that this is one of the potential reasons for the poor voltage angle predictions.

Additionally, in the unsupervised setting, the voltage values are partially learnt using the predicted reactive powers of the generator nodes, which form a significant part of the networks. So, high quality voltage angle predictions require good reactive power predictions, which themselves again depend on the predicted angles. This combined with the fact that the power flow problem is non-convex, seems to make obtaining high quality predictions relatively difficult.

Most previous work use significantly larger datasets for training the models compared to the 10.000 training samples we used, so this may be a limiting factor for our model's performances. Besides using larger datasets, another potential future direction for improving the predictions could be to include some additional features to the input features of the nodes, which currently consist of their fixed power flow values. [3] includes additional engineered features to the inputs of each node to improve their unsupervised training of the model, but they do not discuss what these features are or how they were created. Creating additional informative features ourselves from the current inputs would require a larger amount of expertise in power systems.

4.3 Edge Features Utilization

One of the key assumptions we had from the initial phase of the project is the vital role played by edge features to effectively learn the dependencies between nodes and hence to learn to approximate power flow accurately. In fact, the features (i.e. conductance and susceptance) of the lines of a power grid are crucial to solve tasks such as power flow analysis. We decided to test our assumption by comparing

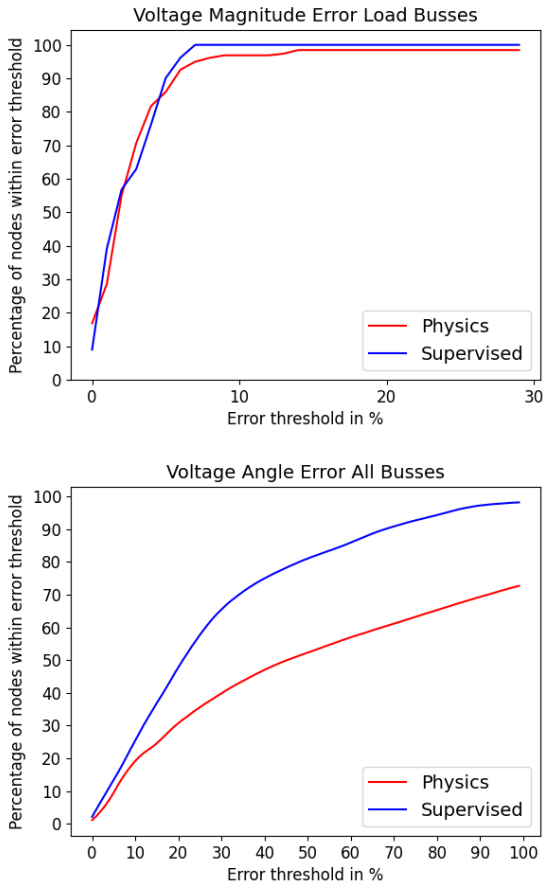

Figure 2: The voltage magnitude and angle prediction errors when using a supervised and unsupervised (physics-informed) loss. The slack bus angle was fixed to the default case118 slack bus angle for all testing samples.

the performance of our model when utilizing and not utilizing the edge features during training. We tested for both the GINE and SAGE convolutions.

Contrarily to our belief, we discovered that edge features do not improve performance significantly. As visible in Figure 3, the models for both GINE and SAGE perform similarly, irrespective of edge features. We justify such results by the lack of sufficient training data. The modelling of edges in heterogeneous graphs in Pytorch Geometric, required the creation of a new edge type between any pair of node types, with a new edge type also for each reverse links (necessary to make the graph undirected). New edge types were also required to model transformer connections. In total we have 32 edge types, each with their own adjacency matrix and learnable parameters. Such a big number of parameters requires far more than our 10.000 instances training dataset. Hence, our assumption is that the model is not able to learn and exploit the edge features properly. The great increase in learnable parameters is a common issue in multi-relational heterogenous graph neural networks [19].

One possible direction for future work is to share parameters for all matrices, as proposed in [19] and in [4]. The idea

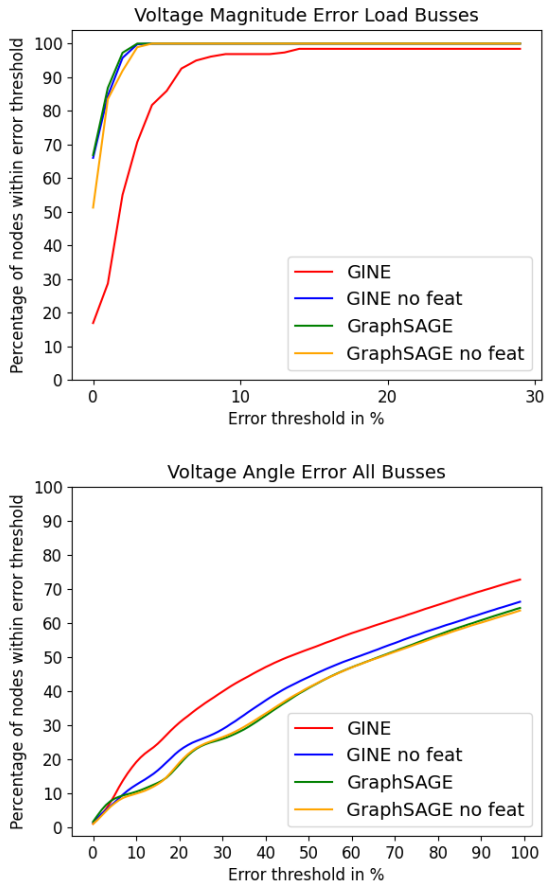

Figure 3: Graphs comparing the performance in the predictions of magnitudes and angles of our model when utilizing and not utilizing edge features.

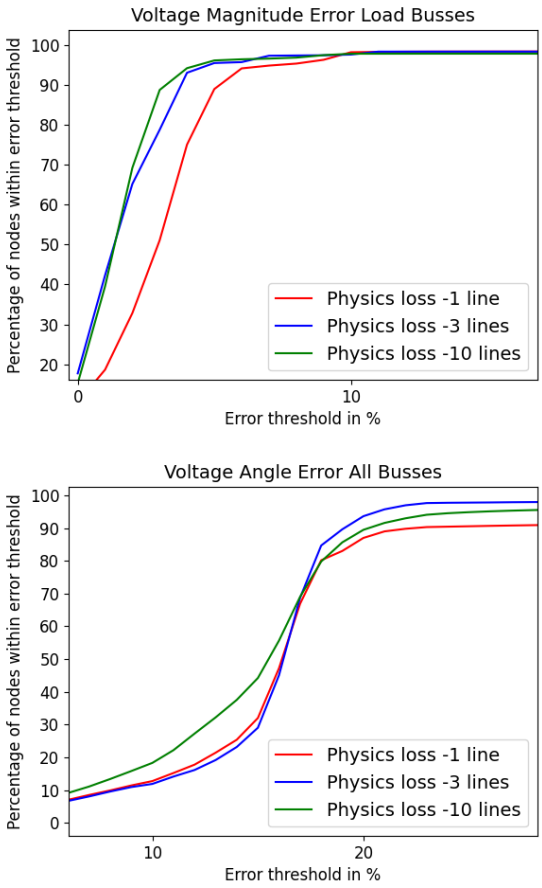

Figure 4: Graphs showing the performance in the predictions of magnitudes and angles of our model when lines are removed from the grid with respect to the topology utilized in the training set.

is interesting in the context of power grids because the lines between different node types share the same features.

4.4 Line Removal Experiments

One of the most typical and regular changes in topology that can occur is the removal of a line due to some malfunction [8], usually caused by birds, rodents, or vandalism. The failure of a line causes a change in topology of the power grid which would require another run of a numerical solver. An inductive GNN model should be able to perform well on the modified topology, without being retrained. Hence, we tested our model in the following scenarios to assess such inductive performance.

For this experiment we trained on 10.000 case 118 full network samples. The test consisted of using 3 different batches of testing data where the graphs were missing one, three and ten randomly selected lines respectively. We will refer to these sets as set one, set three and set ten respectively. We then tested our model on each of these three testing sets to examine whether a decrease in predictive performance can be observed.

The results of these experiments are shown in Figure 4. There are two main points to note from these experiments.

Firstly, and most importantly for the goals of this experiment, we see that the performance is similar for every one of the test cases. This indicates that the model seems to have some inductive properties, as it seems to be able to generalize its performance to modified topologies it has not seen before during training.

Secondly, we observe that in some cases removing lines and changing the topology of the graph actually gives an improvement in the performance. This goes against our intuition and would require more investigation. Our hypothesis is that our line removal could be biased in the way that it does not allow for the removal of lines that would isolate a node or group of nodes from the graph. We believe that the nodes connected to these edges might be the easiest to predict, as they are located in areas with less node density and, therefore, would have an easier time predicting the voltage angle (where we indeed see the biggest increase in performance). So, when the easier edges are unable to be removed (or are less likely to), the edges with the higher error nodes will be removed more often, hence increasing accuracy. We saw similar behaviours in the line removal experiments with a larger number of lines removed, as they had a “coin-flip” behaviour that made them

very unstable (with poorer increased accuracy).

5 Conclusion

The results we obtained from our experiments, show that our model is not able to provide exact approximations of power flow solutions in an inductive setting. Despite the fact that the framework implemented cannot compete with modern numerical solvers, it shows potential for inductive learning with GNNs on power grids. Nevertheless, we believe that our analysis and experiments can provide valuable insights for future research in the field of inductive GNNs for power grids.

Our experiments demonstrate how different subgraphing methods have an impact on inductive performance. Different methods perform differently for different attributes, but none of the implemented method seems to hold a clear edge over the others.

Moreover, our tests showed that training in a supervised setting leads to better inductive performance, contrarily to our initial beliefs. Our conclusion is that to learn the physical laws regulating the power grid via an unsupervised loss, we require a dataset substantially bigger than the one we utilized. While to learn to imitate the output of a numerical solver, as we do in the supervised loss, less data is required.

Furthermore, we observed that the utilization of edge features does not improve performance. As for the unsupervised loss, we deem that a larger dataset is needed to exploit such features effectively.

Additionally, our experiments did successfully show that our model, when trained on the full grid topology, is resilient to small topology changes, such as line removals, and it demonstrates some degree of inductive capabilities.

Finally, throughout the project we experimented with both homogeneous and heterogeneous graphs, to model the power grids. We concluded that the latter are more apt, yielding better performance. Heterogeneous graphs seem also a more natural fit for power grids, considering the different entities (i.e. generators and loads) present in them.

6 Limitations and Future Work

One limitation we encountered during our research is the dataset size and computational resources available. We worked on a relatively small dataset when compared to other studies in the reviewed literature, which usually have dataset ranging from more than 10.000 to 200.000 data instances. Increasing dataset size greatly increases the training time and we could not utilize a larger dataset due to our limited resources. Future work could try to train our model on a larger dataset, as we saw that increasing the dataset size greatly increased performance. Given the large numbers of learnable parameters needed for all the possible edge types in heterogeneous graphs, we also suggest experimenting with typed graph neural networks [15], or with methodologies that allow for parameter sharing among the different edge types adjacency matrices [4; 19], this could partially reduce the necessity for a very big dataset.

Another limitation we faced, is related to the optimization of the model's parameters. From the noticeable improvements that hyperparameters tweaking has produced during our work, we believe that the results of our model could still be substantially improved. The main challenge with hyperparameters optimization is the large number of tunable parameters and the fact that each training setting (i.e. loss function, subgraphing technique) has a different optimal configuration.

As none of our subgraphing methods showed particularly better performance, compared to the others, we encourage future research on other ways to create subgrids, in particular we believe that clustering algorithms could lead to more meaningful and realistic subgrids.

In addition to these limiting factors, we believe that the main task to address in our framework is the accuracy of the voltage angle predictions. This could be done by changing the loss function to better provide an incentive to learn the proper angles, or finding a convolutional operation that allows for better transferal of angle information. As the angle at each bus depends highly on the slack bus angle, methodologies that implement a better propagation mechanism of the angle information (for instance by adding directed links from the slack bus to every other node) could be explored.

Ethical Statement

The renewable energy transition has become a crucial topic when dealing with power grids due to the environmental challenges we face nowadays. This has lead to the significant rise of renewable energy sources, such as wind and solar energy, and many households increasing their electricity consumption due to, for example, electric cars or heating. However, one of the consequences of these changes is that the power grid models have become much more complex [20; 27], increasing the computational complexities and time required to solve power system problems.

While deep learning methods can help with significantly speeding up computations, they are also hard to interpret due to the large number of parameters within the (black-box) models [12]. Furthermore, these models are also quite sensitive to biases within the training data, which may be hard to identify, partially due to the lack of interpretability of the models, and can have large impacts on the results.

For these reasons, it is important to not simply take the results of these models at face value when applying or building upon the methods and models presented in our work. The results must be carefully analysed to check whether they make sense and where the model seems to perform well, or perhaps more importantly, seems to fail. For example, if the model seems to generally have high errors when predicting outputs related to wind farms, then this should be taken into account when using the model. Otherwise, if such incorrect results are applied to real networks or used in future research related to renewable energy, they could potentially lead to inefficient integration of renewable energy sources in the power grids, reducing their positive environmental impact.

References

[1] Jonas Berg Hansen, Stian Normann Anfinsen, and Filippo Maria Bianchi. Power flow balancing with decentralized graph neural networks. *arXiv e-prints*, pages arXiv–2111, 2021.

[2] V. Bolz, J. Rueß, and A. Zell. Power flow approximation based on graph convolutional networks. In *2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1679–1686. IEEE, December 2019.

[3] Luis Böttcher, Hinrikus Wolf, Bastian Jung, Philipp Lutat, Marc Trageser, Oliver Pohl, Xiaohu Tao, Andreas Ulbig, and Martin Grohe. Solving ac power flow with graph neural networks under realistic constraints. In *2023 IEEE Belgrade PowerTech*, pages 1–7. IEEE, 2023.

[4] Daniel Busbridge, Daniel Sherburn, Philippe Cavallo, and Nils Y. Hammerla. Relational graph attention networks. *arXiv preprint arXiv:1904.05811*, 2019.

[5] Chenxu Cai and Yifei Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

[6] Balthazar Donon, Rémy Clément, Benjamin Donnot, Antoine Marot, Isabelle Guyon, and Marc Schoenauer. Neural networks for power flow: Graph neural solver. *Electric Power Systems Research*, 189:106547, 2020.

[7] J.D. Glover, T. Overbye, and M.S. Sarma. *Power System Analysis and Design, SI Edition*. Cengage Learning, 2015.

[8] 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):682, 2019.

[9] William L. Hamilton. *Graph Representation Learning*. Morgan & Claypool Publishers, 2020.

[10] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

[11] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[12] Mahdi Khodayar and Jacob Regan. Deep neural networks in power systems: A review. *Energies*, 16(12):4773, 2023.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] 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. *Journal of Modern Power Systems and Clean Energy*, 10(2):345–360, 2021.

[15] T. B. Lopez-Garcia and J. A. Domínguez-Navarro. Power flow analysis via typed graph neural networks. *Engineering Applications of Artificial Intelligence*, 117:105567, 2023.

[16] Damian Owerko, Fernando Gama, and Alejandro Ribeiro. Optimal power flow using graph neural networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5930–5934. IEEE, 2020.

[17] Xiang Pan, Minghua Chen, Tianyu Zhao, and Steven H Low. Deepopf: A feasibility-optimized deep neural network approach for ac optimal power flow problems. *IEEE Systems Journal*, 17(1):673–683, 2022.

[18] M. Ringsquandl, H. Sellami, M. Hildebrandt, D. Beyer, S. Henselmeyer, S. Weber, and M. Joblin. Power to the relational inductive bias: Graph neural networks in electrical power grids. In *Proceedings of the 30th ACM International Conference on Information Knowledge Management*, pages 1538–1547, October 2021.

[19] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rob Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings*, volume 15, pages 593–607. Springer International Publishing, 2018.

[20] Oliver Smith, Oliver Cattell, Etienne Farcot, Reuben D O'Dea, and Keith I Hopcraft. The effect of renewable energy incorporation on power grid stability and resilience. *Science advances*, 8(9):eabj6734, 2022.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[22] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[23] Y. Wu, D. Zhuang, A. Labbe, and L. Sun. Inductive graph neural networks for spatiotemporal kriging. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4478–4485, May 2021.

[24] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[25] Keyulu Xu, Chengtao Li, Yizhe Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, July 2018.

[26] Hanqing Zeng, Hongxu Zhou, Amit Srivastava, Ramakrishnan Kannan, and Viktor K. Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019.

[27] Qianyu Zhao, Wenlong Liao, Shouxiang Wang, and Jayakrishnan Radhakrishna Pillai. Robust voltage control considering uncertainties of renewable energies and loads via improved generative adversarial network. *Journal of Modern Power Systems and Clean Energy*, 8(6):1104–1114, 2020.