A Framework for Assessing the Generalizability of GNN-Based AC Power Flow Models

Anonymous Author(s)

ABSTRACT

AC power flow analysis is essential for grid planning, operation, and optimization, but traditional methods like the Newton-Raphson algorithm can be computationally expensive for complicated systems. Simplified approaches, such as DC power flow, perform well in transmission grids but struggle in distribution grids due to high resistance-to-reactance (R/X) ratios that affect voltage stability and power losses. Recent advances in machine learning, particularly Graph Neural Networks (GNNs), offer a flexible, topology-aware alternative for power flow solvers, but existing models lack generalizability and fail to incorporate critical topological features. This work presents the first framework for assessing the generalization performance of these graph-based solvers, including a generalization score to evaluate this model robustness. By analyzing graph similarity statistics and their correlation with model performance, we explore the generalization impact of embedding simple graph characteristics into GNN models for distribution grids, and ultimately demonstrate that neighbourhood-aware GNNs produce more generalizable solutions. These findings address key challenges in power flow analysis, advancing the applicability of GNNs to complex distribution networks.

CCS CONCEPTS

• **Hardware → Power networks**; • **Theory of computation → Sample complexity and generalization bounds**; • **General and reference → Reliability**.

KEYWORDS

Graph learning, Power Grid Analysis, Model Generalizability

ACM Reference Format:

Anonymous Author(s). 2025. A Framework for Assessing the Generalizability of GNN-Based AC Power Flow Models. In . ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In power systems, AC power flow is particularly useful for planning, operation, and optimization of grid performance under various network conditions. It determines the operating conditions of the electrical grid based on some known inputs. One common method to solve this is via the Newton-Raphson (NR) algorithm [7]. This very accurate method uses linear approximations to iteratively solve a set of non-linear equations, but can be very computationally expensive for large systems, and is not guaranteed to converge. This limitation has led to the development of simplified solving methods such as the DC power flow approximation that linearizes the problem such that a non-iterative, convergent solution can be found. While this method works well for the analysis of transmission grids, distribution grids are more complicated due to the higher ratio of resistance to reactance (R/X) in their lines. Power flow equations and control strategies vary significantly between high and low R/X ratio systems. In high R/X networks, resistance plays a more significant role in terms of voltage drops, which can lead to greater active power losses and impact voltage stability, particularly when dealing with larger, more complex grids with distributed generation.

To effectively tackle this increasing complexity, recent research has explored machine learning based approaches for power flow. Among these techniques is the use of graph representation learning, particularly in the form of Graph Neural Networks (GNNs), due to their flexibility and high parallelizability. By taking into account nodes, edges, and connectivity, GNN models capture the importance of grid topology, which has been highlighted as a promising advantage over Artificial Neural Network (ANN) approaches in creating performant and flexible solutions to power flow. While ML methods offer computational advantages over traditional numerical solvers, their development cost in terms of data collection and training only becomes justified if they can function safely and reliably across diverse grids. However, GNN-based PF solvers are still in their primary stages of exploration, development, and verification. Current state of the art GNN solvers are mostly in the context of transmission grids [5, 11, 12] and do not consider neighbourhood characteristics that may help nodes learn more meaningful information. They primarily train and test on very similar grid structures and their results show performance degradation when evaluated in new contexts. Therefore, a robust generalization framework for GNN-based power flow solvers is essential, as current methods risk producing physically incorrect results when applied beyond their training scenarios, potentially leading to poor operational decisions, incorrect security assessments, and unreliable contingency analyses. Generalizable power flow solvers would enable system planners and operators to confidently manage current and future grid scenarios and instill trust in these data-driven methods.

In this work, we introduce a framework for assessing the generalization performance of graph-based distribution grid power flow solvers. Generalization refers to the solver's ability to maintain stable and accurate performance when applied in new contexts, i.e. unseen distribution grids. Generalizable PF solvers should perform reliably across a variety of grid configurations without requiring retraining or extensive modifications. We explore the relationship between grid similarity and GNN model performance in order to create a generalization score for these PF models, thus establishing a measure of trust. Furthermore, we investigate how to improve

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

this score by embedding simple topology characteristics such as positional encoding. Through analysis of similarity and performance, we ultimately show that these features lead to more stable, generalizable power flow solvers that perform comparably to DC power flow for unseen grids. Our contributions are as follows:

(1) The first empirical study and open source framework for evaluating the generalization potential of power flow models in a distribution grid context.
(2) Introduction of a scoring function to inform model choice, affirm robustness, and establish trust. The score is a function of the model's mean performance and its resilience (i.e the model's uncertainty with respect to a unit change in context).
(3) We illustrate that improving generalizibility of GNN solvers can be achieved by making it easier to learn grid structures, not only via more complex models.

We first present the relevant related works in Section 2, focusing on graph-learning methods and their applications for power flow. Next, we present our experiment methodology in Section 3, followed by the experiment results in Section 4. Lastly, we evaluate the framework and discuss possible future work in Section 5 before concluding in Section 6.

2 RELATED WORKS

2.1 Graph Structures

Topological characteristics can help distinguish an electrical networks' grid level, region, or neighbourhood. For example, it is well known that rural areas tend to have different grid topologies than urban ones. In the former, one would find many long, simple radial structures, whereas ring or meshed networks can be found in the latter to support grid reliability and resilience. These topologies are important in grid planning and operation as electricity flow is adjusted and re-routed to deliver power optimally and deal with contingencies. [21] highlights the importance of these graph structures for data-driven power grid analysis, where topological properties can be used as inputs, outputs, and also as a means of validating real-world viability of synthetic grids. As such, graph structures are inherently significant to the operation of power grids.

Graph theory can be used to understand the topological characteristics of a power grid, where each grid is represented as a graph, G, with a set of vertices, V, and edges, E. A descriptor function can be used to map each graph into a vector in $\mathbb{R}^d$ that describes a particular characteristic. Common descriptors include node degree, clustering coefficient, or graph spectrum (i.e. a set of eigenvalues and eigenvectors of the graph's adjacency matrix). Using these representations, one can compare different graphs and ultimately determine how similar one structure is to another. One method is to use the maximum mean discrepancy (MMD), which is commonly applied in graph generative modeling analysis to asses model performance [16, 30].

MMD uses the descriptors and representations of one set of graphs to compute the distance to another reference set by use of kernels, i.e. similarity measures for structured objects. Using the distances between graphical distributions, one can evaluate the similarity of networks such as power grids. An example of the grid similarity evaluation process is depicted in Figure 1. Ultimately, MMD allows us to quantify the amount of change in our graphical datasets. Generalizable models should be able to perform consistently well in the presence of changing topology, and thus MMD is significant in measuring this.

Table 1: Bus types and their information in power flow.

Bus type	P	Q	V	θ
Slack (ref)	Unknown	Unknown	Given	Given
PV (gen)	Given	Unknown	Given	Unknown
PQ (load)	Given	Given	Unknown	Unknown

2.2 Positional Encoding in GNNs

Graph Neural Networks learn via message passing, i.e. aggregating messages from neighbouring nodes and combining this information using neural networks to create node representations. However, non-expressive GNNs have trouble distinguishing regular and isomorphic graphs (i.e. graphs that look very similar from a message passing perspective). Embedding extra information into nodes as a means of positional encoding is an effective technique for increasing the representation power of these models. [10] use degree to improve GNNs without any other node features while [29] shows that including cycle counts can greatly improve accuracy on challenging node, edge, and graph property prediction tasks. Furthermore, [8] presents a framework that is able to learn positional encodings and combine these with structural GNNs to generate more expressive node embeddings. These positional representations ultimately improved the accuracy of various GNN models between 1.79% and 64.14% when benchmarked on molecular, social network, and image datasets [8]. Our hypothesis is that we can find simple positional encodings that improve distribution grid power flow.

2.3 GNN-based Power Flow Methods

$$P_i = \sum_{k=1}^{N} |V_i||V_k|(G_{ik}cos(\theta_i - \theta_k) + B_{ik}sin(\theta_i - \theta_k)) \quad (1)$$

$$Q_i = \sum_{k=1}^{N} |V_i||V_k|(G_{ik}sin(\theta_i - \theta_k) - B_{ik}cos(\theta_i - \theta_k)) \quad (2)$$

The objective of AC power flow analysis is to determine the active power (P), reactive power (Q), voltage magnitude (V), and voltage angle (θ) of every bus in a network. The given information differs based on bus type (Figure 1), but every bus i must satisfy Equations 1 and 2, where G_{ik} and B_{ik} are the real and imaginary part of the admittance between buses i and k, respectively. As an alternative to traditional numerical methods, ML-based power flow approximation is attractive for power flow calculation in complex grids. These solvers are able to calculate solutions that are several orders of magnitude faster than numerical methods such as Newton-Raphson [5, 12]. However, neural-based methods such as MLPs and CNNs are made for tabular data and have a harder time exploiting the network topology in their solutions, often resulting in lower accuracy when compared against GNN-based methods, particularly when evaluating performance on perturbed or unseen grids [11]. Furthermore, GNN models are more flexible and do not require a fixed-sized grid, making them attractive candidates for

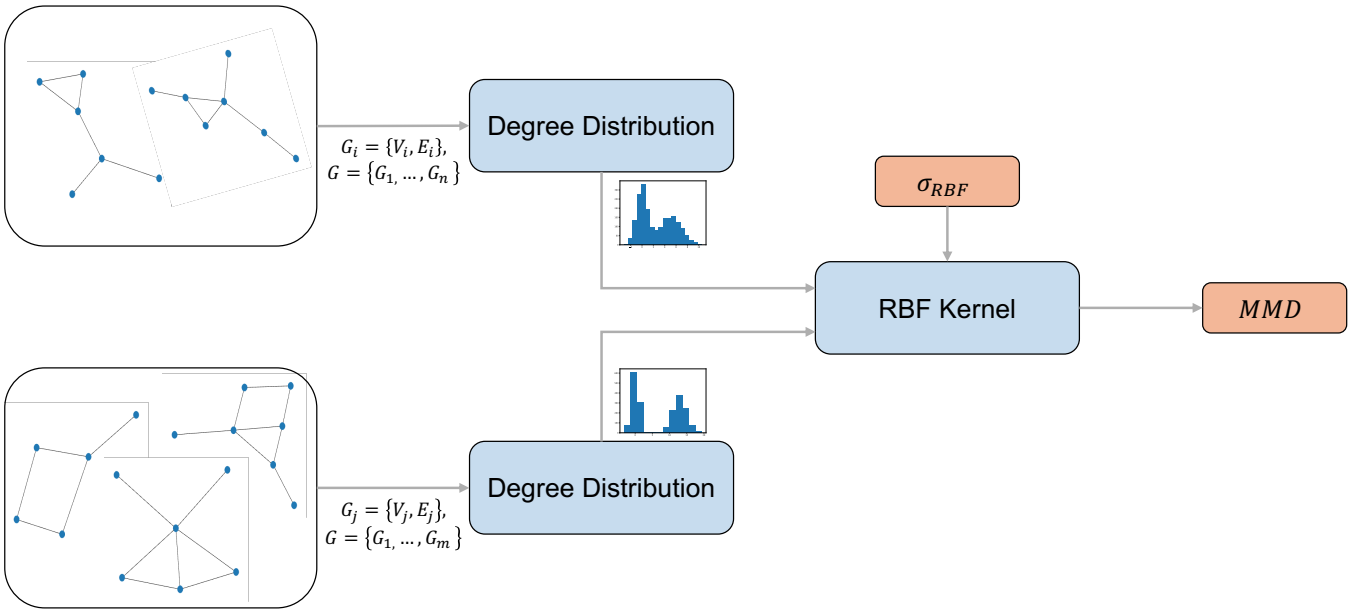


Figure 1: An overview of the MMD graph similarity calculation process using Degree Distribution as the descriptor function and RBF as the kernel.

generalization. A review of graph-based methods and how they apply to the power flow problem can be found in [17] and [28]. [11] uses an ARMA GNN model to learn a power flow solver that is less sensitive to oversmoothing (i.e. when increasing network depth leads to homogeneous node representations). They evaluate their model using Newton-Raphson solutions as the ground truth, and use the DC power flow approximation and custom local and global MLP models as benchmarks. Contrarily, both [5] and [12] instead use an unsupervised approach in which violations of physical laws are penalized during the training process, and as such do not merely imitate the output of a Newton-Raphson solver. In [12], this is combined with a Graph Attention Network (GAT) used to learn the importance of neighbouring nodes during power flow calculation. Realizing that the majority of previous works focus solely on transmission grids, [4] develop a similar physics-based unsupervised method for distribution grids and evaluate on a custom synthetic grid, a SimBench [20] grid, and a real medium voltage grid. Their synthetic and SimBench grids similarly use NR as the target, and results are additionally compared to Graph Neural Solver (GNS) [5].

2.3.1 Research Gaps. The research gaps of these comparative works are summarized in Table 2. No pair of these methods use the same dataset for evaluating their methodology, thus hindering the ability to reproduce results and directly compare with one another. While [12] and [4] merely describe their models, [5] releases the model code and a synthetic data generator based on the IEEE grids. [11] goes one step further and makes the work completely reproducible by releasing both the code and the data used for the analysis. Three of the works are based on transmission grid data, while [4] performs analysis of graph-based power flow in a distribution grid context. In their experiments, most papers do perform basic tests on grids with differing topology and node characteristics; however, none provide in-depth empirical analysis on how grid types affect model performance. [11] notes that their model is performant when operating on familiar grids, but does not generalize very well to unseen grids. [12] alludes to this finding as well and presents a figure illustrating that combination of train and testing grid influences model performance. [4] makes the best effort in understanding generalizability by training their GNN model on their own synthetic dataset, testing with two other datasets, and comparing this to a fine-tuned version of the model. Similar to all other works, they conclude that there is a large discrepancy between the zero-shot and fine-tuned performance, with the model not being able to generalize beyond the training data's topologies. In general, no similar work attempts to understand and improve their models' lack of generalizability, but rather look to find complex model architectures that work for their trained grids.

This work aims to fill these gaps by performing more complete analysis of model robustness for power flow in distribution grids via graph similarities, performance correlation, and positional encoding. Ultimately, these insights help us understand why these power flow solvers fail to generalize and allow us to create more robust and reliable power flow solvers.

3 METHOD

3.1 Data Generation

In order to train our solvers and perform statistical analyses, a representative distribution grid dataset is needed. As outlined in [21], there are various published datasets of synthetic power grids that can be used for ML-based power flow training. These datasets tend to be location-specific and the authors do not give any indication of which dataset is the most comprehensive overall. For

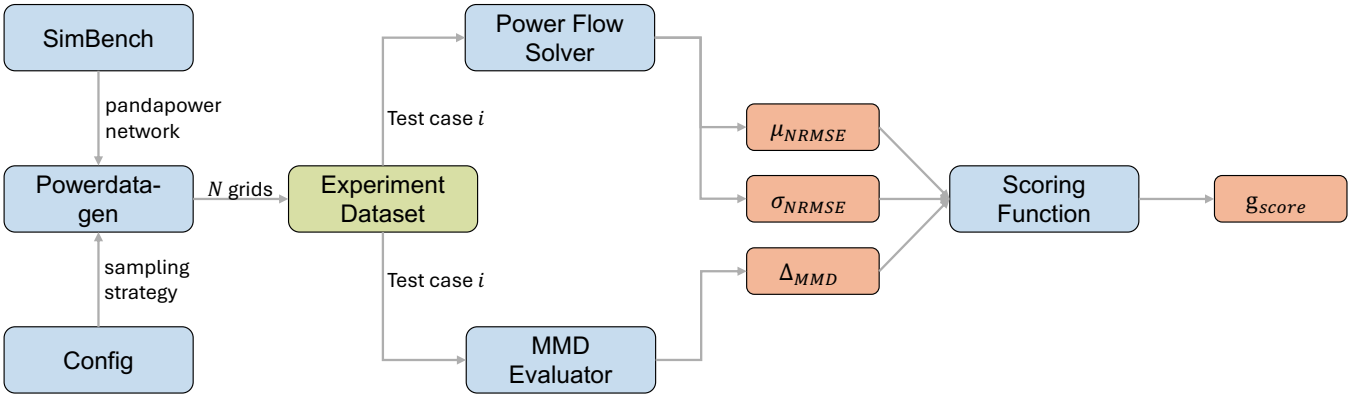


Figure 2: Overview of the presented generalizability evaluation framework.

Table 2: Evaluation of existing graph-based power flow approximation methods.

Model	DG Focus	Reproducibility	Generalizability
[5]	X	✓	X
[11]	X	✓✓	-
[12]	X	-	-
[4]	✓	-	-

example, the paper mentions that SimBench seems to be more common and representative of European (German) distribution grids, while Smart-DS [19] would be more applicable for North America (USA). Other notable reference grids are the IEEE Distribution Test Cases [23], which contain 5 test feeders designed specifically for distribution grid analysis, or the DINGO dataset [2], which contains 3608 synthetic MV grids in high resolution, generated using GIS data and local search metaheuristics [1].

Since these datasets do not all contain many grid configurations, a synthetic graph generator can be used to develop a more comprehensive dataset, if necessary. While graph generation historically relied on expert design, current research seems to favor generation tools [21]. Two options for high quality synthetic generators that are also publicly accessible are Powerdata-gen [6], which uses statistical sampling, or a deep graph generation tool such as GraphRNN [31]. Other relevant tool considerations are outlined in Appendix B.

For this work, we use SimBench and Powerdata-gen to generate our experiment datasets since both tools work out of the box and are readily compatible with one another. Powerdata-gen uses a suite of statistical sampling techniques to generate datasets from a reference pandapower network. After sampling, the grids are solved for AC Power Flow and go through a set of validation checks, and only the grids that meet the requirements are added to the final set. Particularly, from a base grid we generate 300 new synthetic network test cases with varying loads, generations, and topologies. As node features for each bus in a grid, we use: a one-hot encoding of node type (slack, generation, or load), active power, reactive power, voltage angle, and voltage magnitude. As edge features for the branches we use: resistance (per km), reactance (per km), a transformer boolean, and relative short-circuit voltage.

To simultaneously introduce new graph structures and further widen the data distributions represented in the dataset, we perform this generation for multiple base grids. Specifically, we take the 10 distribution grid cases of SimBench's scenario 1 (future grid with normal increase of distributed energy resources). The test cases span the low and middle voltage level, and represent rural, semiurban, urban, and commercial networks. Thus, we create new power networks with differing topologies, buses, rated voltages, supply points, transformer types, etc. The combination of both structural and data variations allows us to develop a rich dataset of 3000 distribution grids that can be used to evaluate generalization. To ensure reproducibility of results, the dataset and the generation configurations are released along with this work.

Ultimately, a model's response to both graph structure and data distribution contribute to its generalization performance. Our framework captures the former via grid similarity analysis (Section 3.2), and assesses the latter using classic machine learning performance metrics (Section 3.3).

3.2 Grid Similarity Analysis

We first analyze our dataset to determine the similarity between reference grids. We focus on structural similarity, i.e. network topology, and not on the individual attributes of entities in the graph.

As indicated in [22], the ability to use MMD to capture differences between distributions is kernel and parameter dependent, which are often domain-specific. We follow the authors' recommendations for choosing the correct parameters. Particularly, we use an RBF kernel, with degree distribution and normalized Laplacian spectrum as descriptor functions. [24] illustrates that the clustering coefficient converges to zero for radial distribution feeders, so we leave this out, since it will not be very informative. Additionally, we do prior analysis to select hyperparameters that align with the area of maximum discrimination in MMD for our given function-kernel combination, such that we can effectively observe differences. Without this analysis, we would be unable to properly measure distances between graph distributions, since MMD is known to be sensitive to appropriate parameter choice. Fortunately, since we have already

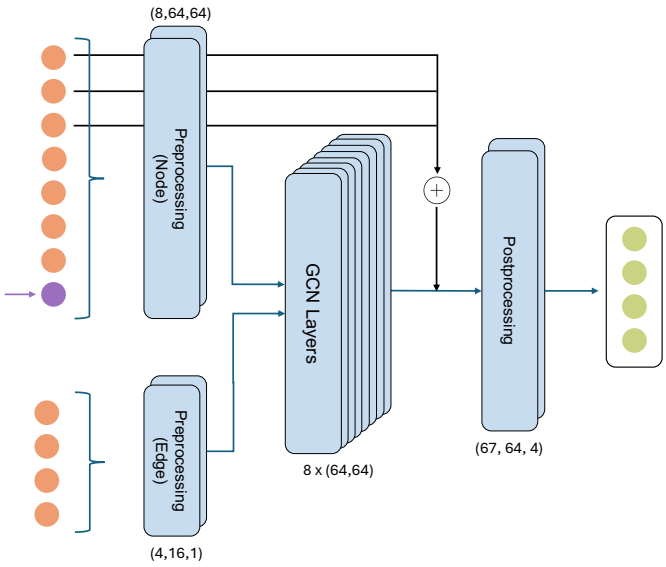


Figure 3: Per node representation of the Reference GCN model for generalization experiments, including the augmented node feature.

conducted this analysis, users of this framework do not have to repeat this, but can instead simply bring their own models for assessment. More information on MMD including its formulation and parameter selection can be found in Appendix A.

To evaluate the distance between distributions, we convert the input graphs to NetworkX and use the open source code from [22] to calculate MMD. We perform this step for every test case in our experiment sets. Since every experiment test has a corresponding graph similarity comparison, we can directly correlate the model evaluation to the MMD between the training and test grids. For all experiments we use both the degree and laplacian MMD descriptors.

3.3 Performance Evaluation

3.3.1 Model Setup. To evaluate our proposed framework, we select Graph Convolutional Networks (GCNs) [14] as the baseline model due to their foundational role in graph representation learning. GCNs offer a balance between simplicity and effectiveness, making them a widely adopted benchmark in graph learning, and an ideal candidate for demonstrating our framework's capabilities. Notably, both GAT and ARMA GNN solvers of contemporary power flow works (Section 2.3) build upon GCN, with GAT introducing attention mechanisms [27] and ARMA GNN employing more sophisticated filtering techniques inspired by signal processing [3]. By using a standard GCN and its variants enhanced with positional encoding, we establish a strong baseline while maintaining focus on our framework's generalization assessment capabilities rather than architectural innovations. This choice ensures that our framework is not tied to a specific model architecture but remains broadly applicable to evaluating contemporary and future GNN-based power flow solvers (Section 5.2.3). Our goal is not to determine the best-performing architecture but to provide a systematic method for quantifying and improving generalization in power flow learning, demonstrating that even simple modifications can lead to meaningful performance gains.

For our analysis, we use a GCN model that is configured similarly to the model in [11], with pre- and post-processing layers, as well as an eight layer GCN (as determined by hyperparameter tuning), which can be seen as a balance between oversmoothing and graph coverage. More specifically, to process the d input node features, we first map them into a 64-dimensional embedding space via an MLP preprocessing layer with one hidden layer of 64 neurons. Similarly, we incorporate the edge features by adding a small MLP layer with 16 hidden neurons that uses edge information to provide a $1d$ edge weight for every message passed between two nodes. The node embeddings then enter the eight layer convolutions, where they receive and aggregate messages from neighbouring nodes. After the final GCN layer, we concatenate the one hot encoding of node type and pass this into the final postprocessing layer, structured similarly to the preprocessing layer. A visualization of our GCN model is presented in Figure 3. We train the model freely for 500 epochs, backpropagating with an MSE loss; however, we incorporate the known values at inference time. We evaluate model performance using Normalized Root Mean Square Error (NRMSE) as defined in Equation 3, in order to balance the difference of scale across feature dimension.

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\frac{1}{D}\sum_{i=1}^{D}(Y_{ij}-\hat{Y}_{ij})^2}}{\frac{1}{D}\sum_{i=1}^{D}(y_{\max j}-y_{\min j})} \tag{3}$$

Where:

N : the number of samples
D : the dimensionality of the vectors
Y : the true values
$\hat{Y}$: the predicted values
$y_{\max j}$: the maximum of the true values for the j-th dimension
$y_{\min j}$: the minimum of the true values for the j-th dimension

3.3.2 Experiment Setup. We conduct two experiments to assess model performance for an unseen grid. The first experiment tests **Cross-Context (CC) Generalization**. The model is trained on one grid (300 samples) and must predict the power flow of a fundamentally different grid. This represents a situation in which we have an existing, highly functioning model for one distribution grid, but we want to apply it in a new context. The second experiment tests **Out-of-Distribution (OOD) Generalization**. The model is trained on data from all but one reference grid ($300 * 9 = 2700$ samples), and is then tasked with predicting the power flow of that left over grid. This test mirrors the case in which there already exists a large, global model, but we need to apply it in a fundamentally new grid structure. This is precisely the case in which all models from [11] failed to generalize as well as DC PF.

To compensate for the data imbalance and ensure a fair comparison between experiment groups, we scale the batch sizes used during training, while keeping epochs and learning rate fixed. Smaller batches result in more parameter updates (iterations) per epoch, meaning more total optimization steps for the same amount of data, and research has shown that smaller batches often leads to

better generalization [13]. This hyperparameter is adjusted such that similar validation error ranges are achieved. Particularly, we conduct the Cross-Context experiment using batches of size 64, and the Out-of-Distribution experiment using batches of size 256.

3.4 Feature Augmentations

We embed additional distribution grid structural information into every node in order to create model variations with different types of positional encoding. Since rural areas are often categorized by radial (tree-like) networks, while ring or mesh structures can be observed in densely populated urban regions, we use cycle length to encode this information. This means each node understands if it is part of a possible cycle and how long the cycle is.

Additionally, it is known that load buses behave differently based on their distance from generation buses and the slack bus. Buses farther from generators have a higher sensitivity to power imbalances, especially in high R/X ratio systems like distribution networks. Buses farther from the slack bus may also see greater phase angle difference, since the slack bus helps compensate for imbalances in power flow. We select path length to the slack bus as an additional, simple feature.

Lastly, bus neighbourhood can give more contextual information for power flow, such as in identifying local bottlenecks. There are many ways to add neighbourhood information, from clustering coefficient to curvature [25], but we opt for simplicity and embed node degree as a new structural feature.

We hypothesize that these very simple additions of local information can extend the expressive power and thus generalizability of the GNN model. We embed the augmented node features into the reference model, creating 4 different experiment variations: reference, reference with cycle length, reference with path length, and reference with degree. We repeat all experiments from Section 3.3 and also compare these to the results of DC PF. It is important to note that DC PF is an analytical model, not a learned model, so its test performance for a particular grid will be consistent regardless of training data. Furthermore, for both generalization experiments we consider all reference grid combinations. For the GCN models, we evaluate 10 reference grids and 4 model variants, resulting in $90 * 4 = 360$ and $10 * 4 = 40$ test cases for the Cross-Context and Out-of-Distribution experiments, respectively.

3.5 Generalization Score

We use the grid similarity of Section 3.2 and the performance evaluation of Section 3.3 to draw conclusions about the effectiveness of simple positional encoding in creating more robust GNN models. To do so, we require a generalization score for a model's performance across a graphical dataset.

3.5.1 Intuition. A model's ability to generalize should be a function of its performance under normal conditions, adjusted by its ability to remain stable in new contexts. A model's mean NRMSE, μ_{NRMSE}, describes its average performance, its standard deviation NRMSE, σ_{NRMSE}, describes its dispersion or instability, while the MMD range of the dataset, Δ_{MMD}, describes the extent to which the context changes. Intuitively, when μ_{NRMSE} gets better or worse, we would want our score to follow the same trend. As σ_{NRMSE} grows we want the generalization score to get worse, while a completely stable

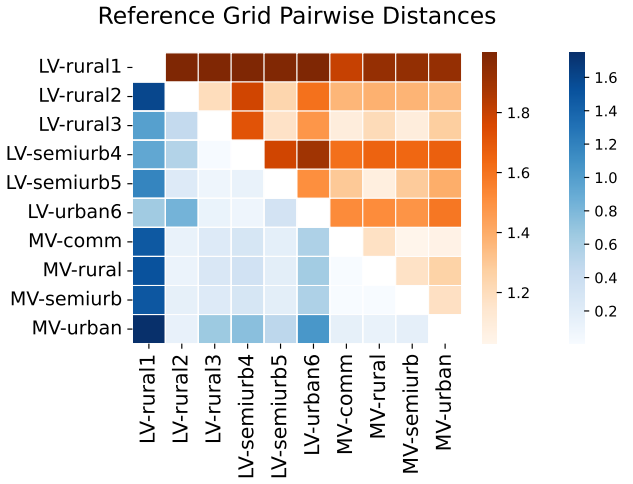


Figure 4: Pairwise comparison of reference distribution grids from the dataset, using degree distribution (blue) and normalized Laplacian spectrum (orange) as descriptor functions.

model ($\sigma_{\text{NRMSE}} = 0$) should be judged according to its mean. Additionally, as the context becomes increasingly different ($\uparrow \Delta_{\text{MMD}}$), the score should depend on how well the model stability scales with this context change. If a model's instability grows faster than the context, the score should additionally grow, while models that remain stable as the context grows should be rewarded.

In this framework, we introduce such a score to assess the generalization performance of several GNN power flow solvers.

3.5.2 Mathematical Formulation. For our proposed framework, we require a dataset of distribution grids, $\mathcal{D}$, a set of test cases $\mathcal{T}$, and a model $\mathcal{M} = (A, \theta)$ with an architecture A and parameter set θ. In order to evaluate the model's ability to resiliently solve tasks across $\mathcal{T}$, we also require an *evaluation metric*, $E_{\mathcal{M}} : \mathcal{T} \to \mathbb{R}^+$, for model performance, and a *comparison metric*, $C : \mathcal{T} \to \mathbb{R}^+$, for assessing distance between distributions: namely of the model's training grid(s), $\mathcal{X}_i \in \mathcal{D}$, and the evaluation grid, $\mathcal{X}_j \in \mathcal{D}$.

Finally, let $\mathcal{G} := \{(E_{\mathcal{M}}(T), C(T)) : E_{\mathcal{M}}(T) >= P_2, E_{\mathcal{M}}(T) <= P_{98}, T \in \mathcal{T}\}$ represent the set of (evaluation, comparison) pairs of all test cases, where P_i is the i-th percentile of the evaluation scores of $\mathcal{M}$ across test cases $\mathcal{T}$. Using NRMSE as E and MMD as C, we define the $\mathfrak{g}_{\text{score}}$ across the set $\mathcal{G}$ as follows:

$$\mathfrak{g}_{\text{score}} = \mu_{\text{NRMSE}} + \sigma_{\text{NRMSE}} \cdot \frac{\log(\Delta_{\text{MMD}}+1)}{\Delta_{\text{MMD}}+\epsilon} \tag{4}$$

Where:

$\mu_{\text{NRMSE}} := \mu(\text{NMRSE}_{\mathcal{M}}, \mathcal{T})$, the mean performance.
$\sigma_{\text{NRMSE}} := \sigma(\text{NMRSE}_{\mathcal{M}}, \mathcal{T})$, the std. of performance.
$\Delta_{\text{MMD}} := \max(\text{MMD}, \mathcal{T}) - \min(\text{MMD}, \mathcal{T})$, the similarity range.
ϵ : a stability parameter (e.g. 10^{-8})

The lower the generalization score, the greater the generalization potential of a model. This has some valuable generalization properties. Firstly, it removes extreme outliers from the dataset,

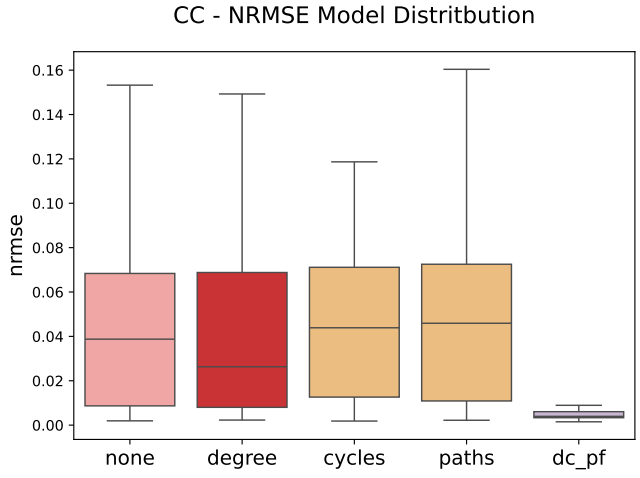


Figure 5: Performance distribution of the four GNN power flow solvers for the Cross-Context experiment, compared with the analytical DC PF solver.

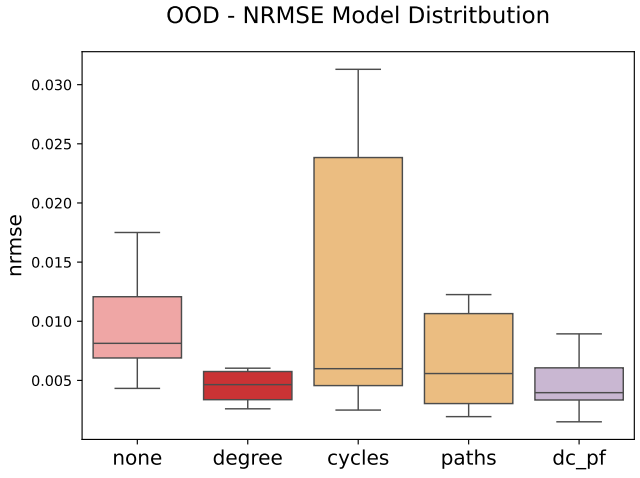


Figure 6: Performance distribution of the four GNN power flow solvers for the Out-of-Distribution experiment, compared with the analytical DC PF solver.

keeping only those values roughly two standard deviations away from the mean. The score values model accuracy by penalizing larger errors, but also rewards models with low volatility, promoting model stability. Additionally, the metric considers the range of graphical difference of the test cases and rewards models whose performance degrades slower with increasing statistical distance. This balance of model performance and model stability is vital to ensure a reliable metric and promote robust GNN solvers for safe use in power grids. Further mathematical properties of this score are discussed in Appendix C.

4 RESULTS

The final overview of our generalizability evaluation framework is presented in Figure 2. Using this structure we are able to understand the changing operating context of each of our experiment test cases and how each model variant responds to this change.

4.1 Grid Similarity Analysis

We first evaluate our dataset using the kernel functions and descriptors we selected earlier. We select parameters such that our dataset test cases have a spread of evenly distributed graph similarity metrics, such that our test cases capture an appropriate range. This allows a robust set of test cases over which we can evaluate generalization performance. Figure 4 presents the distances between pairs of the SimBench reference grids. We note that grid similarity spans both grid type and grid level, and is not always intuitive. For example, when using degree distribution as a descriptor, LV-rural2 seems to be more closely related to the MV grids than the other LV grids, while MV-comm and MV-rural also seem to be quite closely related to one another. We also note that grids that are deemed similar according to one descriptor may actually be quite different according to another (for example, LV-semiurb4 and LV-urban6). This analysis both validates our dataset's coverage of distance, as well as reveals key insights in how to compare different distribution grid structures.

4.2 Model Performance

4.2.1 Cross-Context. The Cross-Context experiment evaluates how well learning on just one single grid helps us perform on another never-before-seen grid. Each model is limited in its ability to capture relevant information for a grid that is unfamiliar to it, which is reflected in the overall poor model performance (Figure 5). Here, all GNN models perform similarly, with added features only providing a marginal benefit. It is clear from both the mean and the distribution that the DC PF model is far superior in this context. With their wide performance distributions, the GNN models are still able to occasionally achieve good results. For example, in 12/90 of the test cases, the degree model actually outperforms the DC PF solver. Ultimately, even after giving the models 'easy access' to simple topology information, it seems there is just not enough generalizable information to learn from one single grid.

4.2.2 Out-of-Distribution. Looking at the results of the Out-of-Distribution experiment, we asses how much learning on several grids can aid in solving power flow for a never-before-seen grid. Here, the results appear more promising. Figure 6 illustrates that the models are much more capable of estimating power flow on a new grid if they have had enough learning exposure. Since the model has more variations in graphs to learn from, it can also learn how to properly take advantage of this extra information. In this experiment we very clearly see the effect of additional graph features on model performance, with all the new models outperforming the reference model on average. Particularly looking at distribution, it seems that the degree GNN model has similar stability characteristics as the DC PF solver, making it a viable candidate for generalizability. Although the DC PF's mean NRMSE is lower than that of the degree model, the degree model still manages to

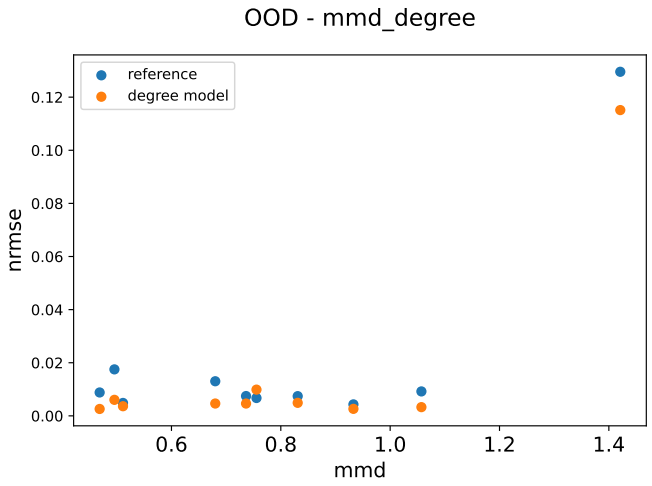


Figure 7: An observed performance shift after embedding degree as a learned feature in a GNN power flow solver.

outperform the DC solution in 6/10 cases. This suggests that the model's generalization capabilities are very comparable to that of DC power flow.

4.3 Generalization Score

This generalization framework allows for a structured way to explore power flow solvers in new contexts, visualize generalizability figures, and quantify what used to be only observed behaviour and intuition about a GNN model's ability to adapt. We combine the results of the previous analyses and present the main contribution of our generalization framework: the generalization score. Without loss of generality, we present results using the degree distribution as a descriptor. The interested reader can verify that the outcomes generally hold for normalized laplacian spectrum as well, using our provided framework and published code. We present the summary table of generalization scores for the Cross-Context and Out-of-Distribution Experiments in Table 3. The following are particular use cases that highlight the type of analysis one can perform using our framework.

4.3.1 Plot Shifting. Recall from Section 4.2 that the degree model proved to be a stable power flow solver, with a low mean and tight distribution when compared to the reference case. Our intuition says that this would make this model better at generalization. Using this framework, we can can visualize this generalizability and simply look at the trends. In Figure 7, we observe a consistent downward shift of the NRMSE datapoints after embedding degree as a node feature and thus can conclude that the degree model is better equipped for generalization. This model improvement can be directly attributed to the additional degree feature, as we see in Table 3 that other feature additions cannot achieve this and may even result in worse performance and more instability.

4.3.2 Inter-model Evaluation. The proposed framework also assists in comparing two models where conclusions are not as clear. In such a situation, the generalization score can provide clarity and a metric grounded in the test statistics. Consider the two generalization visualizations in Figure 8. Each model seems to have its advantages and disadvantages. Both models have a moderate linear and monotonic correlations according to the pearson and spearman correlations coefficients (≈ 0.4) and they follow similar trends. While the path model has a slightly lower slope, it also displays more variance. On the other hand, the cycle model's 98-percentile marker is higher (indicated by the gray area), but its distribution is skewed towards lower NRMSE values. It is evident that these generalizability scores will be quite similar to each other, and indeed they are.

4.3.3 Intra-model Evaluation. In some cases we would like to analyze how our model performs on some particular data segment or test case. For distribution grid power flow this may provide valuable information about the conditional strengths and weaknesses of a model to a DSO. We look to Figure 9 as an example of such a case. We compare the generalization performance of the reference model with respect to the grid it was trained on. Particularly, we compare the effect of training on a commercial grid vs a semi-urban grid on model generalization. In both test cases, the model is observed to have a high pearson correlation (≥ 0.68), but the variant trained on a commercial grid has a slightly tighter fit and smaller slope of increase, as seen by the figure's axes. Contrarily, when training on a semi-urban grid, the model has a number of excellent predictions that ultimately bring its overall mean down and improve its generalizability.

5 DISCUSSION

5.1 Framework Evaluation

By visually comparing the model distributions, analyzing the generalization plots, and computing the generalization score, we conclude that embedding simple structural features improves GNN power flow generalizability for the Out-of-Distribution case. All but one of the models with positional encoding outperform the reference GCN, simply as a result of this addition. Most notably, using degree as an additional feature achieves the best results of all GNN solvers and even outperforms DC PF in some cases. We observe that an increase in dissimilarity does not always lead to a decrease in performance for the OOD models, thus emphasizing our models' ability to adapt. On the other hand, we observe from both the distributions and the generalization scores that this effect is not applicable for the Cross-Context case. Here, it is likely that each grid has some unique structures that do not apply to other grids and thus are difficult for a model to learn. Nonetheless, these statistics are informative in assessing our models' ability to effectively adapt under varying structural assumptions.

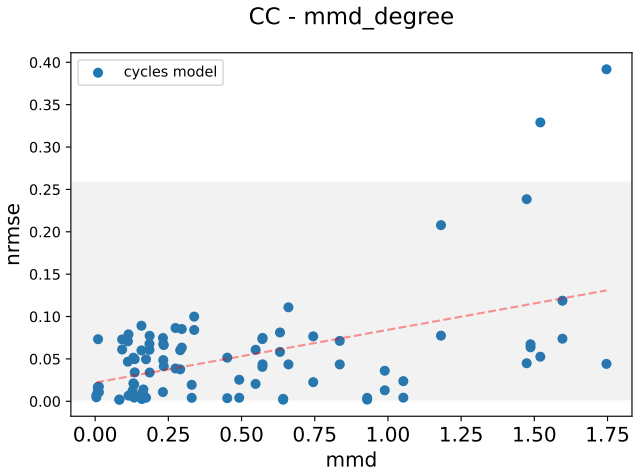
We make our framework including its dataset and experimental analysis publicly available[1]. The generalization score fills the research gap by combining key aspects of performance and distance metrics to create a means of analyzing model robustness across contexts. We recommend using this framework and generalization score for a quantifiable and comparable metric for distribution grid power flow solvers. For researchers, this should encourage more

[1]The framework source code is available at: Anonymized for double-blind reviewing.

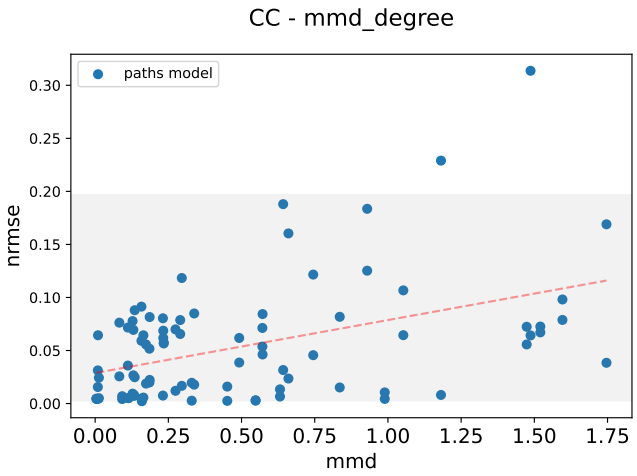
Table 3: Generalization statistics for models in the Cross-Context (left) and Out-of-Distribution (right) experiments.

Model	μ_{NRMSE}	σ_{NRMSE}	Δ_{MMD}	$\mathfrak{g}_{\text{score}}$
Reference	0.0464	0.0432	1.7421	0.0714
Degree	0.0410	0.0385	1.7421	0.0633
Cycles	0.0470	0.0404	1.7421	0.0704
Paths	0.0497	0.0429	1.7421	0.0745
DC PF	0.0044	0.0017	0.0000	**0.0044**

Model	μ_{NRMSE}	σ_{NRMSE}	Δ_{MMD}	$\mathfrak{g}_{\text{score}}$
Reference	0.0094	0.0038	0.5885	0.0124
Degree	0.0050	0.0021	0.5615	0.0066
Cycles	0.0117	0.0096	0.5615	0.0193
Paths	0.0066	0.0036	0.4642	0.0096
DC PF	0.0044	0.0017	0.0000	**0.0044**

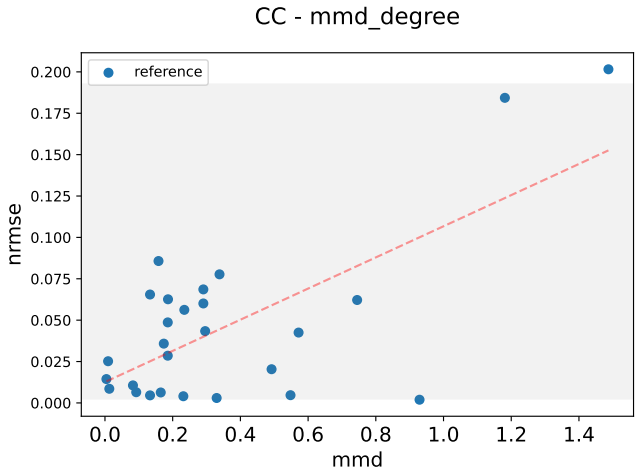


(a) Cycles Model

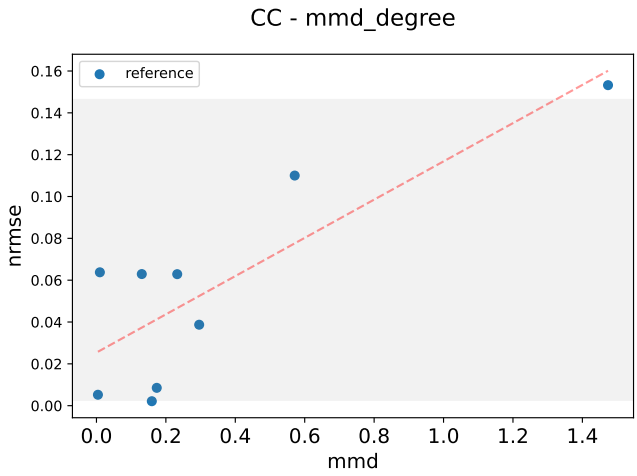


(b) Paths Model

Figure 8: Visualization of Cross-Context Generalization results of two GNN power flow solvers.



(a) Semi-urban Training Grid



(b) Commercial Training Grid

Figure 9: Visualization of Cross-Context Generalization of the Reference model, based on training grid.

transparent generalization benchmarks for power flow, and ultimately promote stability in GNN applications in power systems. When encountering a new grid, one can calculate the graphical distance to the previously trained set, and thus gain insights on how familiar this new topology really is. Conversely, when provided a new model, or a choice of models, one can use the generalization score as an a priori indicator of how the model will scale in context. We demonstrate this with Table 4. When approached with these two models, one can evaluate the generalization score for the particular area of application. It seems that both models perform similarly on

Table 4: Generalization statistics for various testing grid segments using the Reference and Cycles models.

Test Grid	$\mathfrak{g}_{\text{score, REF}}$	$\mathfrak{g}_{\text{score, CYCLES}}$
LV	0.0811	**0.0802**
MV	0.0555	**0.0537**
rural	**0.0962**	0.1090
semiurb	**0.0611**	0.0616
urban	0.0533	**0.0438**
comm	**0.0359**	0.0470

average across low and medium voltage grid level, with the cycles model being the best overall performer. However, the influential factor could be the region in which the model will be deployed, since the model performance varies more significantly by grid type. If a distribution grid operator is primarily operating in rural contexts, the more reliable choice would clearly be the reference model. With the help of this generalization score, grid operators are able to clearly identify how trustworthy a model is and are thus more informed and confident when choosing the appropriate solver.

5.2 Limitations & Future Work

5.2.1 Dataset Generation. One limitation of this study is the evaluation dataset. This dataset is developed via statistical sampling and is consistent with the methods of the current AC power flow literature, but more sophisticated data generation techniques would provide more comprehensive datasets. For example, although Powerdatagen provides topology variations, it does so by semi-randomly disconnecting lines, generators, and loads in the network. As a result, the generator is limited in its ability to create completely new and valid grid structures. Future work could address this by improving the synthetic dataset, preferably using deep graph generation techniques, and re-running our experiments. Particularly, deep graph generation is preferred since machine learning can be used to learn hidden representations, avoiding the need to hard-code specific properties or make assumptions on the importance of certain characteristics.

Furthermore, we are limited by our number of SimBench reference grid cases, and thus were only able to conduct 10 experiments for the Out-of-Distribution case. In the future, we would ideally consider a larger suite of base grids. These reference networks must be principally different than one another to allow separation into distinct groups, which is a limitation of the current public synthetic networks. Future research could focus on combining several subsets of multiple datasets into one larger generalization benchmark set, or investigating clustering methods to automatically create these subsets from a larger set using machine learning.

5.2.2 MMD Parametrization. It is well known that MMD is sensitive to the kernel, descriptors, and parameters used for evaluation. For this framework, we perform prior analysis to ensure appropriate parameters and descriptors, and thus users of our framework do not need to repeat this work. Future work could focus on improving this framework with parameter-free kernels such as a linear kernel. Additionally, more expressive descriptor functions such as curvature filtrations [25] could help better understand differences between grid sets.

5.2.3 Model Evaluation. A natural progression of this work would be to establish a benchmark for AC Power Flow generalization. Using the proposed generalization score, this study would quantify the generalization potential of the state of the art GNN power flow solvers, and would serve as a standard for future solvers. Since the majority of the more recent and advanced power flow solvers did not release their models as open source code, one would need to re-implement the methods to perform this large-scale evaluation. Strictly speaking, this type of generalization analysis is not limited to GNN-based models, so other neural methods such as MLPs and CNNs could also be evaluated with this framework. However, these models cannot evaluate grids of arbitrary size and thus may not be compared fairly in such a framework, unless they create a new model variation for every context (which is not very generalizable).

5.2.4 Generalization of DC Power Flow. In this work we evaluate a series of Power Flow solvers against a DC power flow implementation in an attempt to match or surpass the performance of DC PF, which requires no training and thus is impervious to structural grid changes. The experiment results reveal that DC Power Flow is overall more generalizable than our introduced models, but in 6/10 of the Out-of-Distribution cases, the enhanced degree model outperforms DC PF. Future work could explore the situations in which DC PF performs poorly and propose guidelines for when to use an alternative neural-based method instead.

6 CONCLUSION

This study introduces a framework for evaluating the generalization performance of graph-based power flow solvers in distribution grids, addressing a critical gap in understanding their applicability across diverse grid topologies. By embedding topological features, such as node degree, into Graph Neural Network (GNN) models, we demonstrate improved generalizability and stability, particularly in Out-of-Distribution scenarios. The proposed generalization score offers a quantifiable and practical metric for assessing model robustness and aiding in model selection. While the enhanced GNN solvers approach the generalization performance of DC power flow, surpassing it in several individual test cases, they still exhibit limited resilience to changing grid structure. This suggests avenues for future research, including the development of richer synthetic datasets, parameter-free similarity metrics, and benchmarks for assessing the generalization performance of state-of-the-art power flow solvers. This work underscores the potential of topology-aware GNNs in power systems and provides a foundation for developing reliable, scalable, and adaptable power flow solvers for increasingly complex distribution networks.

REFERENCES

[1] J. Amme, G. Pleßmann, J. Bühler, L. Hülk, E. Kötter, and P. Schwaegerl. 2018. The eGo grid model: An open-source and open-data based synthetic medium-voltage grid model for distribution power supply systems. *Journal of Physics: Conference Series* 977, 1 (Feb. 2018), 012007. https://doi.org/10.1088/1742-6596/977/1/012007 Publisher: IOP Publishing.

[2] Jonathan Amme, Guido Pleßmann, Jochen Bühler, Ludwig Hülk, Editha Kötter, and Peter Schwägerl. 2017. *Distribution grid data generated by DINGO.* https://doi.org/10.5281/zenodo.890479

[3] Filippo Maria Bianchi, Daniele Grattarola, Lorenzo Livi, and Cesare Alippi. 2021. Graph neural networks with convolutional arma filters. *IEEE transactions on pattern analysis and machine intelligence* 44, 7 (2021), 3496–3507.

[4] Luis Böttcher, Hinrikus Wolf, Bastian Jung, Philipp Lutat, Marc Trageser, Oliver Pohl, Andreas Ulbig, and Martin Grohe. 2023. Solving AC Power Flow with Graph Neural Networks under Realistic Constraints. In *2023 IEEE Belgrade PowerTech*. 1–7. https://doi.org/10.1109/PowerTech55446.2023.10202246 arXiv:2204.07000 [cs].

[5] Balthazar Donon, Rémy Clément, Benjamin Donnot, Antoine Marot, Isabelle Guyon, and Marc Schoenauer. 2020. Neural networks for power flow: Graph neural solver. *Electric Power Systems Research* 189 (Dec. 2020), 106547. https://doi.org/10.1016/j.epsr.2020.106547

[6] Donon, Balthazar. 2022. Powerdata-gen. https://github.com/bdonon/powerdata-gen. GitHub repository, last accessed: 2024-12-16.

[7] Simone Dutto, Giulio Masetti, Silvano Chiaradonna, and Felicita Di Giandomenico. 2019. On Extending and Comparing Newton–Raphson Variants for Solving Power-Flow Equations. *IEEE Transactions on Power Systems* 34, 4 (2019), 2577–2587. https://doi.org/10.1109/TPWRS.2019.2897640

[8] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2021. Graph Neural Networks with Learnable Structural and Positional Representations. In *International Conference on Learning Representations*. https://openreview.net/forum?id=wTTjnvGphYj

[9] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *J. Mach. Learn. Res.* 13, null (March 2012), 723–773.

[10] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf

[11] Jonas Berg Hansen, Stian Normann Anfinsen, and Filippo Maria Bianchi. 2023. Power Flow Balancing With Decentralized Graph Neural Networks. *IEEE Transactions on Power Systems* 38, 3 (May 2023), 2423–2433. https://doi.org/10.1109/TPWRS.2022.3195301 Conference Name: IEEE Transactions on Power Systems.

[12] Ashkan B. Jeddi and Abdollah Shafieezadeh. 2021. A Physics-Informed Graph Attention-based Approach for Power Flow Analysis. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 1634–1640. https://doi.org/10.1109/ICMLA52953.2021.00261

[13] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. In *International Conference on Learning Representations*. https://openreview.net/forum?id=H1oyRlYgg

[14] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SJU4ayYgl

[15] Ming Liang, Yao Meng, Jiyu Wang, David L. Lubkeman, and Ning Lu. 2021. FeederGAN: Synthetic Feeder Generation via Deep Graph Adversarial Nets. *IEEE Transactions on Smart Grid* 12, 2 (2021), 1163–1173. https://doi.org/10.1109/TSG.2020.3025259

[16] Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Will Hamilton, David K Duvenaud, Raquel Urtasun, and Richard Zemel. 2019. Efficient graph generation with graph recurrent attention networks. *Advances in neural information processing systems* 32 (2019).

[17] Wenlong Liao, Birgitte Bak-Jensen, Jayakrishnan Radhakrishna Pillai, Yuelong Wang, and Yusen Wang. 2022. A Review of Graph Neural Networks and Their Applications in Power Systems. *Journal of Modern Power Systems and Clean Energy* 10, 2 (March 2022), 345–360. https://doi.org/10.35833/MPCE.2021.000058 Conference Name: Journal of Modern Power Systems and Clean Energy.

[18] Sean Lovett, Miha Zgubic, Sofia Liguori, Sephora Madjiheurem, Hamish Tomlinson, Sophie Elster, Chris Apps, Sims Witherspoon, and Luis Piloto. 2024. OPFData: Large-scale datasets for AC optimal power flow with topological perturbations. https://doi.org/10.48550/arXiv.2406.07234 arXiv:2406.07234 [cs].

[19] Carlos Mateo, Fernando Postigo, Fernando de Cuadra, Tomás Gómez San Roman, Tarek Elgindy, Pablo Duenas, Bri-Mathias Hodge, Venkat Krishnan, and Bryan Palmintier. 2020. Building large-scale US synthetic electric distribution system models. *IEEE Transactions on Smart Grid* 11, 6 (2020), 5301–5313.

[20] Steffen Meinecke, Džanan Sarajlić, Simon Ruben Drauz, Annika Klettke, Lars-Peter Lauven, Christian Rehtanz, Albert Moser, and Martin Braun. 2020. SimBench—A Benchmark Dataset of Electric Power Systems to Compare Innovative Solutions Based on Power Flow Analysis. *Energies* 13, 12 (Jan. 2020), 3290. https://doi.org/10.3390/en13123290 Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[21] M. Hossain Mohammadi and Khaled Saleh. 2021. Synthetic Benchmarks for Power Systems. *IEEE Access* 9 (2021), 162706–162730. https://doi.org/10.1109/ACCESS.2021.3124477 Conference Name: IEEE Access.

[22] Leslie O'Bray, Max Horn, Bastian Rieck, and Karsten Borgwardt. 2022. Evaluation Metrics for Graph Generative Models: Problems, Pitfalls, and Practical Solutions. In *International Conference on Learning Representations*. https://openreview.net/forum?id=tBtoZYKd9n

[23] Kevin P Schneider, BA Mather, Bikash Chandra Pal, C-W Ten, Greg J Shirek, Hao Zhu, Jason C Fuller, Jose Luiz R Pereira, Luis F Ochoa, Leandro Ramos de Araujo, et al. 2017. Analytic considerations and design basis for the IEEE distribution test feeders. *IEEE Transactions on power systems* 33, 3 (2017), 3181–3188.

[24] Eran Schweitzer. 2019. *Creating, Validating, and Using Synthetic Power Flow Cases: A Statistical Approach to Power System Analysis.* Ph. D. Dissertation. Arizona State University.

[25] Joshua Southern, Jeremy Wayland, Michael M. Bronstein, and Bastian Rieck. 2023. Curvature Filtrations for Graph Generative Model Evaluation. https://openreview.net/forum?id=Dt71xKyabn

[26] Anna Varbella, Kenza Amara, Blazhe Gjorgiev, Mennatallah El-Assady, and Giovanni Sansavini. 2024. PowerGraph: A power grid benchmark dataset for graph neural networks. https://doi.org/10.48550/arXiv.2402.02827 arXiv:2402.02827.

[27] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJXMpikCZ

[28] Arbel Yaniv, Parteek Kumar, and Yuval Beck. 2023. Towards adoption of GNNs for power flow applications in distribution systems. *Electric Power Systems Research* 216 (March 2023), 109005. https://doi.org/10.1016/j.epsr.2022.109005

[29] Jiaxuan You, Jonathan M. Gomes-Selman, Rex Ying, and Jure Leskovec. 2021. Identity-aware Graph Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 12 (May 2021), 10737–10745. https://doi.org/10.1609/aaai.v35i12.17283 Number: 12.

[30] Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. https://doi.org/10.48550/arXiv.1802.08773 arXiv:1802.08773 [cs].

[31] Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. https://github.com/snap-stanford/GraphRNN. Accessed: 2024-12-16.

A MAXIMUM MEAN DISCREPANCY

MMD [9] provides a structured, principled, and expressive means of comparing distributions of structured objects. In principle, it is a distance measure between feature means. Given a kernel function k, n samples in $X = \{x_1, \ldots, x_n\}$, and m samples in $Y = \{y_1, \ldots, y_m\}$, MMD is formulated as follows:

$$\mathrm{MMD}^2(X, Y) = \frac{1}{n^2} \sum_{i,j=1}^{n} k(x_i, x_j) + \frac{1}{m^2} \sum_{i,j=1}^{m} k(y_i, y_j) - \frac{2}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} k(x_i, y_j) \tag{5}$$

There are multiple possible kernel options that can be used to evaluate similarity in MMD, with perhaps the most well-known being the Wasserstein distance (EMD), total variation distance (TV), and the radial basis function kernel (RBF). However, as reported in [22], not every kernel is appropriate for this model comparison task. For example, EMD is inefficient to compute and as such does not scale well with an increasing size and number of graphs. Additionally, the TV-based kernel is not positive semi-definite and thus leads to an indefinite kernel, meaning that the behaviour of the function is not well-defined. As recommended by [22], alternatives to these could be the use of an RBF, Laplacian, or linear kernel. In this work, we use the RBF kernel, defined as:

$$k_{\mathrm{RBF}}(x, y) = \exp\left(-\frac{||x - y||^2}{2\sigma^2}\right) \tag{6}$$

Since this kernel requires a sigma parameter, we additionally analyze the dataset in order to find the correct parameter. More specifically, for each descriptor we systematically compare graph distributions across the experiment test cases using different values of sigma, and select the value that allows us to best capture similarity. The selected sigma values are as follows:

$$\sigma_{\text{CC, degree}} = 1e^{1}$$

$$\sigma_{\text{CC, laplacian}} = 1e^{-2}$$

$$\sigma_{\text{OOD, degree}} = 1e^{2}$$

$$\sigma_{\text{OOD, laplacian}} = 1e^{-2}$$

These sigma values allow us to capture the similarity range of the tests in our dataset. We observe in Figure 10 that our dataset shows a fairly consistent increase in dissimilarity, so our test cases show good coverage.

Additionally, [22] reports that the specific choice of descriptor function is crucial to ensure that we can use MMD to capture differences in distribution. Particularly, it is not guaranteed that MMD will monotonically increase as two distributions become increasingly dissimilar, since different descriptors measure different graph aspects. We use degree and Laplacian spectrum histograms because they are a recommended standard, but other functions are possible.

B EXISTING ML POWER GRID FRAMEWORKS

There exists a deep graph generation mechanism specific to distribution grids named FeederGAN [15] but the authors did not release the tool publicly, so this method is not as readily usable. Furthermore, there also exist ML-ready frameworks for power grid analysis, namely PowerGraph [26] and OPFData [18]. OPFData is a large dataset for investigating the optimal power flow problem in the context of transmission grids so does not fit the distribution grid use case and would need to be altered significantly to work for power flow (potentially by preprocessing the grids to mask certain values). PowerGraph is a much smaller dataset that can directly be applied to the power flow problem but is similarly based on transmission grids and has a very limited amount of grid topologies and loading conditions. Lastly, the authors of [11] make their work publicly available, including a dataset from their analysis. However, the input graphs are similarly based on transmission grid data and moreover use an atypical graph structure (line graph model) that would require conversion for all other models.

C GENERALIZATION SCORE EXTENDED ANALYSIS

The generalization score is a function of the NRMSE mean (μ_{NRMSE}), NRMSE standard deviation (σ_{NRMSE}), and MMD range (Δ_{MMD}) of the experiment cases, as introduced in Equation 4. Since the score is real and continuous for all values of μ_{NRMSE}, σ_{NRMSE} and Δ_{MMD}, we analyze the limits of this scoring function to support its effectiveness as a generalization metric. We consider the following limits:

$$\lim_{\mu_{\mathrm{NRMSE}} \to \infty} \mathfrak{g}_{\mathrm{score}} = \infty$$

$$\lim_{\mu_{\mathrm{NRMSE}} \to 0} \mathfrak{g}_{\mathrm{score}} = 0 \quad \text{(because } \sigma = 0\text{)}$$

$$\lim_{\sigma_{\mathrm{NRMSE}} \to \infty} \mathfrak{g}_{\mathrm{score}} = \infty$$

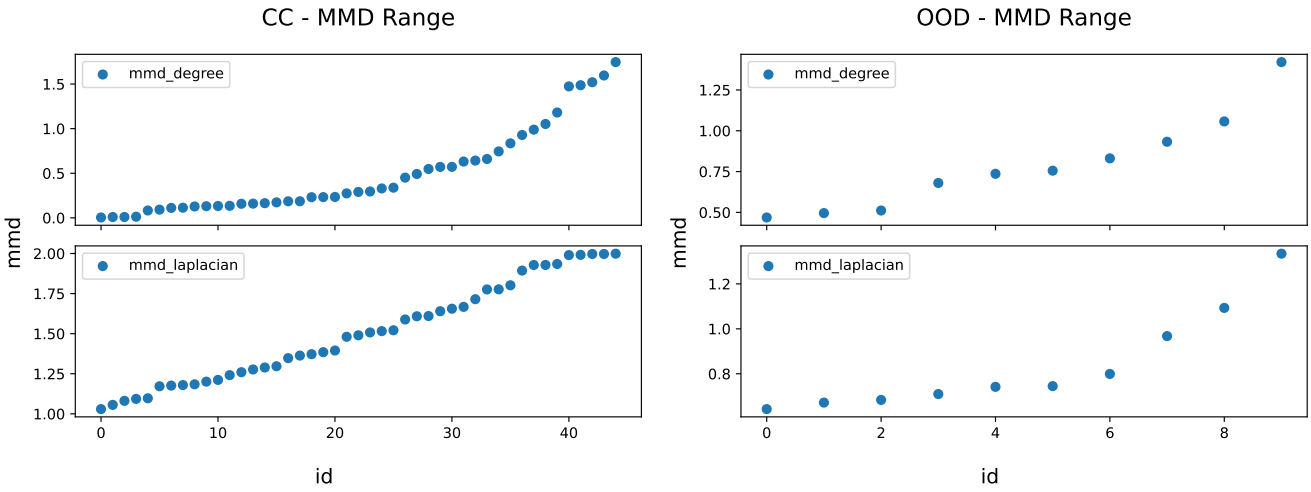
$$\lim_{\sigma_{\mathrm{NRMSE}} \to 0} \mathfrak{g}_{\mathrm{score}} = \mu_{\mathrm{NRMSE}}$$

$$\lim_{\Delta_{\mathrm{MMD}} \to \infty} \mathfrak{g}_{\mathrm{score}} = \mu_{\mathrm{NRMSE}}$$

$$\lim_{\Delta_{\mathrm{MMD}} \to 0} \mathfrak{g}_{\mathrm{score}} = \mu_{\mathrm{NRMSE}}$$

The behaviour of each variable is consistent with how we would want a generalizability metric to perform. With increasing model error, we would like this metric to increase, and a perfect model ($\mu_{\mathrm{NRMSE}} = 0$) can only be achieved if all predictions are also perfect, thus earning the generalization score of 0. As the model grows increasingly unstable, the metric grows as well, but a perfectly consistent model is as good as its mean prediction. Additionally, $\lim_{\Delta_{\mathrm{MMD}} \to \infty}$ is synonymous to a perfectly consistent model, if Δ_{MMD} scales faster than sigma, and synonymous to $\lim_{\sigma_{\mathrm{NRMSE}} \to \infty}$ if it does not. Lastly, a model that is not effected by graphical distance reduces to its mean. This occurs when always training and testing on the same graph or when a model requires no training at all.

It is worthy to note that this deviation term can optionally be scaled by some variable α in order to penalize or relax the constraints on prediction stability. The equation as it is presented



(a) Cross-Context Experiment

(b) Out-of-Distribution Experiment

Figure 10: MMDs of the training and test sets using degree and normalized laplacian spectrum as descriptor functions. Each id represents a unique test case within the experiment group.

assumes an α of 1.0. This is a hyperparameter that can be tuned depending on the needs of the application.