Proximal Policy Optimization with Graph Neural Networks for Optimal Power Flow

Ángela López-Cardona[1] [a], Guillermo Bernárdez[3] [b], Pere Barlet-Rose[1,2] [c] and Albert Cabellos-Aparicio[1,2] [d]

[1] *Universitat Politècnica de Catalunya, Barcelona, Spain*

[2] *Barcelona Neural Networking Center, Barcelona, Spain*

[3] *UC Santa Barbara, California, U.S.A.*

-

Keywords: Optimal Power Flow (OPF), Graph Neural Networks (GNN), Deep Reinforcement Learning (DRL), Proximal Policy Optimization (PPO).

Abstract: Optimal Power Flow (OPF) is a key research area within the power systems field that seeks the optimal operation point of electric power plants, and which needs to be solved every few minutes in real-world scenarios. However, due to the non-convex nature of power generation systems, there is not yet a fast, robust solution for the full Alternating Current Optimal Power Flow (ACOPF). In the last decades, power grids have evolved into a typical dynamic, non-linear and large-scale control system —known as the power system—, so searching for better and faster ACOPF solutions is becoming crucial. The appearance of Graph Neural Networks (GNN) has allowed the use of Machine Learning (ML) algorithms on graph data, such as power networks. On the other hand, Deep Reinforcement Learning (DRL) is known for its proven ability to solve complex decision-making problems. Although solutions that use these two methods separately are beginning to appear in the literature, none has yet combined the advantages of both. We propose a novel architecture based on the Proximal Policy Optimization (PPO) algorithm with Graph Neural Networks to solve the Optimal Power Flow. The objective is to design an architecture that learns how to solve the optimization problem and, at the same time, is able to generalize to unseen scenarios. We compare our solution with the Direct Current Optimal Power Flow approximation (DCOPF) in terms of cost. We first trained our DRL agent on the IEEE 30 bus system and with it, we computed the OPF on that base network with topology changes.

1 INTRODUCTION

After several decades of development, power grids have transformed into a dynamic, non-linear, and large-scale control system, commonly referred to as the power system (Zhou et al., 2020a). Today, this power system is undergoing changes for various reasons. Firstly, the high penetration of Renewable Energy Sources (RES), such as photovoltaic plants and wind farms, introduces fluctuations and intermittence to power systems. This generation is inherently unstable, influenced by several external factors like solar irradiation and wind velocity for solar and wind power, respectively (Li et al., 2021). Concurrently, the integration of flexible sources (e.g., electric vehicles) brings about modifications to networks, including relay protection, bidirectional power flow, and voltage regulation (Zhou et al., 2020a). Lastly, emerging concepts like Demand Response—defined as the alterations in electricity usage by end-use customers from their typical consumption patterns in response to variations in electricity prices over time—affect the operational point within the electrical grid (Wood et al., 2013). All these transformations render the optimization of production in power networks increasingly complex. In this context, Optimal Power Flow comprises a set of techniques aimed at identifying the optimal operating point by optimizing the power output of generators in power grids (Wood et al., 2013).

The traditional approach to solving the OPF involves numerical methods (Li et al., 2021), with Interior Point Optimizer (IPOPT) (Thurner et al., 2018) being the most commonly employed. However,

[a] https://orcid.org/0009-0008-9785-6740
[b] https://orcid.org/0000-0002-6790-4878
[c] https://orcid.org/0000-0001-7837-0886
[d] https://orcid.org/0000-0001-9329-7584

López-Cardona, Á., Bernárdez, G., Barlet-Rose, P., Cabellos-Aparicio and A.
Proximal Policy Optimization with Graph Neural Networks for Optimal Power Flow.
DOI: 10.5220/0013462700003967
In *Proceedings of the 14th International Conference on Data Science, Technology and Applications (DATA 2025)*, pages 347-354
ISBN: 978-989-758-758-0; ISSN: 2184-285X

as networks grow increasingly complex, traditional methods struggle to converge due to their non-linear and non-convex characteristics (Li et al., 2021). Non-linear ACOPF problems are often approximated using linearized DCOPF solutions to derive real power outcomes, where voltage angles and reactive power flows are eliminated through substitution (thus removing Alternating Current (AC) electrical behavior). This approximation, however, becomes invalid under heavy loading conditions in power grids (Owerko et al., 2020). Additionally, the OPF problem is inherently non-convex because of the sinusoidal nature of electrical generation (Wood et al., 2013). Alternative techniques seek to approximate the OPF solution by relaxing this non-convex constraint, employing methods such as Second Order Cone Programming (SOCP) (Wood et al., 2013). In daily operations that necessitate solving OPF within a minute every five minutes, TSO is compelled to depend on linear approximations. The solutions derived from these approximations tend to be inefficient, resulting in power wastage and the overproduction of hundreds of megatons of CO2-equivalent annually. Today, fifty years after the problem was first formulated, we still lack a fast, robust solution technique for the complete Alternating Current Optimal Power Flow (Mary et al., 2012). For large and intricate power system networks with numerous variables and constraints, achieving the optimal solution for real-time OPF in a timely manner demands substantial computing power (Pan et al., 2022), which continues to pose a significant challenge.

In power systems, as in many other fields, algorithms of ML have recently begun to be utilized. The latest proposals employ Graph Neural Networks, a neural network that naturally facilitates the processing of graph data (Liao et al., 2022). An increasing number of tasks in power systems are being addressed with GNN, including time series prediction of loads and RES, fault diagnosis, scenario generation, operational control, and more (Diehl, 2019). The primary advantage is that by treating power grids as graphs, GNN can be trained on specific grid topologies and subsequently applied to different ones, thereby generalizing results (Liao et al., 2022). Conversely, Deep Reinforcement Learning is recognized for its ability to tackle complex decision-making problems in a computationally efficient, scalable, and flexible manner—problems that would otherwise be numerically intractable (Li et al., 2021). It is regarded as one of the state-of-the-art frameworks in Artificial Intelligence (AI) for addressing sequential decision-making challenges (Munikoti et al., 2024). The DRL based approach seeks to progressively learn how to optimize power flow in electrical networks and dynamically identify the optimal operating point. While some approaches utilize various DRL algorithms, none have integrated it with GNN, which limits their ability to generalize and fully leverage the information regarding connections between buses and the properties of the electrical lines that connect them. Given this context, and considering that the combination of DRL and GNN has demonstrated improvements in generalizability and reductions in computational complexity in other domains (Munikoti et al., 2024), we explore their implementation in this work.

Contribution: This paper presents a significant advancement through the proposal of a novel architecture that integrates the Proximal Policy Optimization algorithm with Graph Neural Networks to address the Optimal Power Flow problem. To the best of our knowledge, this unique architecture has not been previously applied to this challenge. Our objective is to rigorously test the design of our architecture, demonstrating its capability to solve the optimization problem by effectively learning the internal dynamics of the power network. Additionally, we aim to evaluate its ability to generalize to new scenarios that were not encountered during the training process. We compare our solution against the DCOPF in terms of cost, following the training of our DRL agent on the IEEE 30 bus system. Through various modifications to the base network, including changes in the number of edges and loads, our approach yields superior cost outcomes compared to the DCOPF, achieving a reduction in generation costs of up to 30%.

2 RELATED WORK

Until this paper, there had been no solution for the OPF problem that utilized GNN to handle graph-type data and DRL, enabling generalization and understanding the internal dynamics of the power grid. Nevertheless, methods can be found in the literature that employ each of the approaches independently. Data-driven methods based on deep learning have been introduced to solve OPF in approaches such as (Owerko et al., 2020), (Donon et al., 2019), (Donon et al., 2020), (Pan et al., 2022), and (Donnot et al., 2017), among others. However, these approaches require a substantial amount of historical data for training and necessitate the collection of extensive data whenever there is a change in the grid. Conversely, the DRL based approach aims to gradually learn how to optimize power flow in electrical networks and dynamically identify the optimal operating point. Approaches like (Zhen et al., 2022), (Li et al., 2021), (Cao et al., 2021), and (Zhou et al., 2020b) utilize different DRL algorithms to solve the OPF, but none incorporate GNN, resulting in a loss of generalization capability.

3 PROBLEM STATEMENT

As illustrated in Figure 1, an agent is trained using DRL. Over multiple timesteps, the agent iteratively modifies the generation values of a power grid, aiming to maximize the reward. This reward reflects the reduction in generation cost compared to the previous timestep. The training process begins with a base case where the agent minimizes the cost while optimizing the search for feasible solutions. Once trained, this agent can be deployed to compute OPF in power grids with altered topologies, such as the loss of an electrical line due to maintenance or the disconnection of a load. The results obtained, in terms of cost, are often better or comparable to those achieved using DCOPF.

4 BACKGROUND

In this section, we provide the necessary background for GNN (subsection 4.1), the DRL algorithm used (subsection 4.2) and we expand the definition of OPF. Commonly, OPF minimizes the generation cost, so the objective is to minimize the cost of power generation while satisfying operating constraints and demands. Some of these constraints are restrictions of both maximum and minimum voltage in the nodes or that the net power in each bus is equal to the power consumed minus generated (Mary et al., 2012). At the same time, the Power Flow (PF) or load flow refers to the generation, load, and transmission network equations. It is a quantitative study to determine the flow of electric power in a network under given load conditions whose objective is to determine the steady-state operating values of an electrical network (Mary et al., 2012).

4.1 Graph Neural Networks

Graph Neural Networks are methods based on deep learning that function within the graph domain. Due to their effectiveness, GNN has recently emerged as a widely utilized approach for graph analysis (Zhou et al., 2020a). The concept of GNN was first introduced by (Scarselli et al., 2009). This architecture can be viewed as a generalization of convolutional neural networks tailored for graph structures, achieved by unfolding a finite number of iterations. We employ Message Passing Neural Networks (MPNN), as introduced in (Gilmer et al., 2017), which represents a specific type of GNN that operates through an iterative message-passing algorithm, facilitating the propagation of information among elements in a graph $G = (N, E)$. Initially, the hidden states of the nodes are set using the graph's node-level features from the data. Subsequently, the message-passing process unfolds (Gilmer et al., 2017): Message (Equation 1), Aggregation (Equation 1), and Update (Equation 2). After a defined number of message-passing steps, a readout function $r(\cdot)$ takes the final node states h_v^K as input to generate the ultimate output of the GNN model. The readout can predict various outcomes at different levels, depending on the specific problem at hand.

$$M_v^k = a(\{m(h_v^k, h_k^i)\}_{i\in\beta(v)}) \tag{1}$$

$$h_v^{k+1} = u(h_v^k, M_k^v) \tag{2}$$

4.2 Deep Reinforcement Learning

The objective in Reinforcement Learning (RL) is to learn a behavior (policy). In RL, an agent acquires a behavior through interaction with an environment to achieve a specific goal (Schulman et al., 2017). This approach is grounded in the reward assumption: all objectives can be framed as the result of maximizing cumulative rewards. DRL is recognized for its robust ability to tackle complex decision-making challenges, making it suitable for capturing the dynamics involved in the power flow reallocation process (Li et al., 2021).

Within the DRL algorithms, we use Proximal Policy Optimization, formulated in 2017 (Schulman et al., 2017) and becoming the default reinforcement learning algorithm at *OpenAI* (Schulman et al., 2017) because of its ease of use and its good performance. As an actor-critic algorithm, the critic evaluates the current policy and the result is used in the policy training. The actor implements the policy and it is trained using Policy Gradient with estimations from the critic (Schulman et al., 2017). PPO strikes a balance between ease of implementation, sample complexity, and ease of tuning, trying to compute an update at each step that minimizes the cost function while ensuring that the deviation from the previous policy is relatively small (Schulman et al., 2017). PPO uses Trust Region and imposes policy ratio to stay within a small interval (policy ratio r_t is clipped), rt will only grow to as much as $1+\varepsilon$ (Equation 4) (Schulman et al., 2017). The total loss function for the PPO comprises L^{CLIP} (Equation 4), the mean-square error

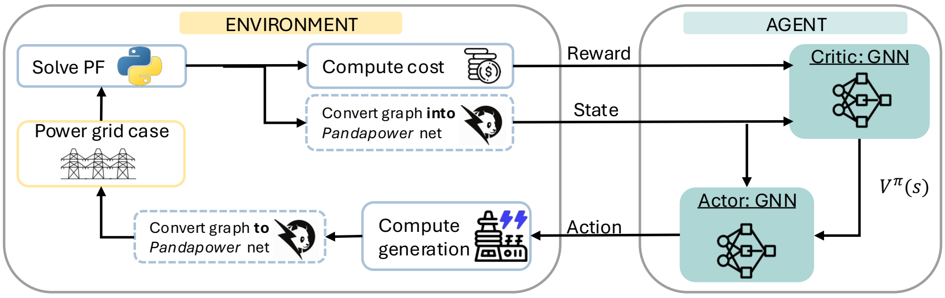

Figure 1: Overview of the PPO-based architecture for power grid optimization. The system consists of an environment and an agent. The environment simulates a power grid case. The agent, implemented using PPO with GNN, consists of an actor-critic structure: the Actor-GNN selects actions, while the Critic-GNN evaluates state values. The agent interacts with the environment by receiving state information, actions (change generation), and rewards based on the computed cost.

loss of the value estimator (critic loss), and an additional term that promotes higher entropy (enhancing exploration) (Equation 3). PPO employs Generalized Advantage Estimate (GAE) to compute the advantage ($\hat{A}_t$), as shown in Equation 5. This advantage method is detailed in (Schulman et al., 2015).

$$L_{TOTAL} = L^{CLIP} + L^{VALUE} * k_1 - L^{ENTROPY} * k2 \quad (3)$$

$$L^{CLIP}(\theta) = \hat{E}_t \left[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)\right] \quad (4)$$

$$A_0^{GAE} = \delta_0 + (\lambda\gamma)A_1^{GAE} \quad (5)$$

5 PROPOSED METHOD

In this section, we outline our approach, which is schematically illustrated in Figure 1. Both the actor and critic of the DRL agent are represented as GNN, while the state of the environment corresponds to the resulting graph of the power grid. Within the DRL environment, the agent executes an action at each time step, adjusting the power of the generator. Subsequently, the power grid graph is updated through a Power Flow.

We treat our power grid as graph-structured data by utilizing information on the power grid topology, where electrical lines serve as edges and buses as nodes, along with the associated loads and generations. For the electrical lines, we define features using resistance R and reactance X ($e_{n,n}^{ACLine} = [R_{n,m}, X_{n,m}]$). For the buses, we incorporate voltage information, including its magnitude V and phase angle θ, as well as the power exchanged at that bus between the connected loads and generators, represented as $X_n^{AC} = [V_n, \theta_n, P_n, Q_n]$.

The overall architecture of the GNN is illustrated in Figure 2, which includes the message passing and readout components. At each message-passing step k, each node v receives the current hidden states of all nodes in its neighborhood and processes them individually by applying a message function $m()$ (NN) along with its own internal state h_v^k and the features of the connecting edge. These messages are then aggregated through a concatenation of min, max, and mean operations. By combining this message aggregation with the node's hidden state and updating the combination using another NN, new hidden state representations are generated. After a specified number of message passing steps, a readout function $r()$ takes the final node states h_v^K as input to produce the final output of the GNN model.

For the **actor**, whose output is the RL policy, the readout consists of a 3-layer MLP NN where the input comprises each of the *node representations*. We independently pass through this readout the representation of all nodes with a generator, resulting in N output values. Each output value signifies the probability of selecting that generator to enhance its power. This approach to managing the readout ensures that the architecture remains generalizable to any number of generators. These values are utilized to form a probability distribution, from which a value is sampled (representing the ID of the generator whose generation is increased at that time horizon t). The **critic** employs a centralized readout that takes all node hidden states as inputs (by concatenating the sum, minimum, and maximum), producing an output that estimates the value function. Consequently, the input dimension is *3*node representation* with a single output for the entire graph. The critic is also structured as a 3-layer MLP.

Regarding the environment with which the agent interacts, at each time instant t, the state is defined by the graph updated by the Power Flow. In each horizon step (t), the action performed by the agent involves increasing the generation of one of the generator nodes. The agent will determine which of the available gen-

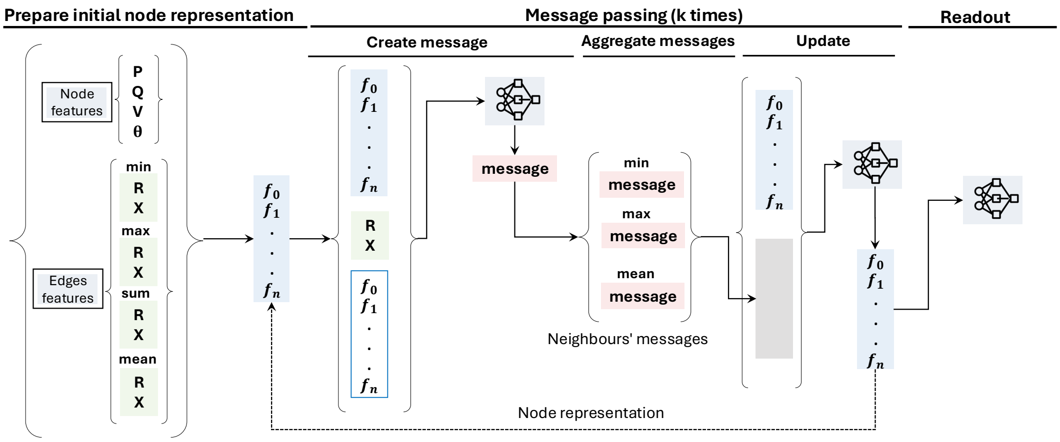

Figure 2: GNN architecture. Both critic and actor employ the same GNN architecture, differing only in their readout layers. The process initiates with the preparation of initial node representations, leveraging both node and edge features. Specifically, the GNN's input comprises the electrical parameters of the grid. During the message-passing phase (repeated k times), each node generates messages based on its features, which are subsequently aggregated from its neighbors. These aggregated messages refine the node representations via an update function. Ultimately, in the readout phase, the actor utilizes these refined representations to compute the action, while the critic uses them to estimate the value function.

erators will have its generation increased by one portion. For each generator, the power range between its maximum and minimum power is divided into N portions. When a generator reaches its maximum power, generation cannot be increased, resulting in the power grid (and thus the state of the environment) remaining unchanged. If the PF does not converge, it indicates that with the given demand and generation, meeting the constraints is not feasible; we refer to this situation as an infeasible solution. When initializing an episode (the initial state of the power grid), we aim for the generation to be as low as possible, allowing the agent to raise it until it reaches the optimum. Additionally, we must consider that if the generation is too low in the initial time steps, the solution may become infeasible. Consequently, we decide to set the minimum generation at 20%. The reward at time t is calculated as the improvement in the solution's cost compared to $t-1$, as shown in Equation 6. The reward is positive during a time step when the agent's action results in a decrease in generation cost. Conversely, if the agent selects a generator that is already at its maximum capacity, leads to an infeasible solution, or increases the cost, the reward will be negative.

$$r(t) = \begin{cases} MinMaxScaler(cost) - Last(MinMaxScaler(cost)) \\ cte_1 \text{ if selected generator already in } P_{max} \\ cte_2 \text{ if solution no feasible} \end{cases} \tag{6}$$

6 PERFORMANCE EVALUATION

This section outlines the experimentation conducted to validate the proposed approach, the data utilized, and discusses the results obtained.

Overview: We train the agent using a base case and subsequently evaluate its performance in modified scenarios. On one hand, we adjust the number of loads and their values, as real power grid operations involve continuous changes in loads. On the other hand, we simulate the unavailability of certain electrical lines due to breakdowns or maintenance. This approach demonstrates that the agent, once trained, can generalize to previously unseen cases. We compare the cost differences between our method and the industry standard method, the DCOPF. Our goal is to demonstrate that our method can produce a solution that is equal to or better than the DCOPF, while avoiding its disadvantages.

6.1 Experimental Setting

We train the agent using the IEEE 30 bus system as our case study (Figure 3). This system consists of thirty nodes, forty links, five generators, and twenty loads, with all generators modeled as thermal generators. We utilize *Pandapower*, a Python-based, BSD-licensed power system analysis tool (Thurner et al., 2018). This tool enables us to perform calculations such as OPF using the IPOPT optimizer and PF analysis, which we employ to evaluate our costs and update our environment. Additionally, this library allows us to verify the physical feasibility of our solutions by ensuring they comply with PF constraints.

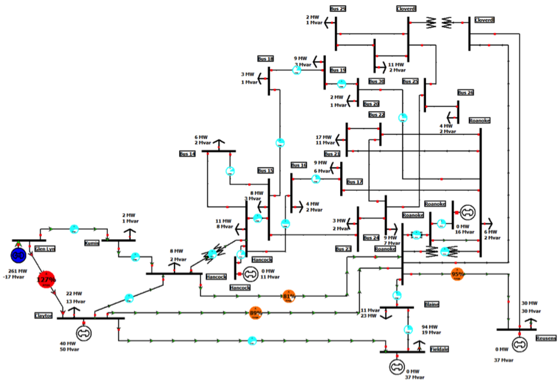

Figure 3: IEEE 30 bus system (Fraunhofer, 2022).

The objective of the training is to optimize the parameters so that the actor becomes a good estimator of the optimal global policy and the critic learns to approximate the state value function of any global state. Many hyperparameters can be modified, and they are divided into different groups. Grid search has been performed on many of them, and the final selected values of the most important ones are shown below.

- Related to learning loop: Minibatch (25), epochs (3) and optimizer (ADAM), with its parameters like learning rate lr (0.003).
- Related to the power grid: Generator portions (50).
- Related to RL: Episodes (500), horizon size T (125), reward cte_1 (-1) and cte_2 (-2).
- Actor and Critic GNN: Message iterations k (4), node representation size (16). The NN to create the messages is a 2-layer MLP and the updated one is 3-layer MLP.

PPO is an online algorithm that, similar to other reinforcement learning algorithms, learns from experience. The training pipeline is organized as follows:

- An episode of length T is generated by following the current policy. While at the same time the critic's value function V evaluates each visited global state; this defines a trajectory $\{s_t, a_t, r_t, p_t, V_t, s_{t+1}\}_{t=0}^{T-1}$.
- This trajectory is used to update the model parameters –through several epochs of minibatch Stochastic Gradient Descent– by maximizing the global PPO objective.

The same process of generating episodes and updating the model is repeated for a fixed number of iterations to guarantee convergence. *MinMaxScaler* has been used for data preprocessing for node features, edge features and generation output. More implementation details can be found in the public repository [1].

6.2 Experimental Results

Once the training has been done and the best combination of hyperparameters, network design and reward modelling has been chosen, the best checkpoint of the model is selected to compute OPF in different networks. To validate our solution, we use the deviation of the cost concerning the minimum cost, the one obtained with the ACOPF (%DRL+OPF perf. in Table 1 - Table 3). We compare it with the cost deviation obtained with DCOPF (%DCOPF perf in Table 1 - Table 3). We compute the ratio between these two deviations. We calculate the improvement ratio by dividing the first value by the second, which reflects the enhancement over the DCOPF.

Once the model is trained, only the actor part is used in the evaluation. During T steps of an episode, the actions sampled from the probability distribution obtained from the actor for each state of the network are executed. Finally, the mean cost of the best ten evaluations is measured, as well as the convergence of the problem. We evaluate 100 times for each test case. In Table 1 - Table 3, it is highlighted between the deviation in % of our solution concerning the OPF's one and the deviation obtained by the DCOPF. We also assess the convergence and physical feasibility of our solution, finding that it was feasible in the majority of cases.

First, all network loads are varied by multiplying their value by a random number between a value less than 1 and a value higher than 1 (Table 1). Each row in the table is a test in which the name specifies the upper and lower percentages by which loads have been varied. In all tests, performance with our method is better with ratios of up to 1.30.

Table 1: Results on case IEEE 30 varying loads from base case.

	% DRL+OPF perf.	%DCOPF perf.	ratio
load_inf0.1_sup0.1	**0,75**	0,77	1,02
load_inf0.2_sup0.1	**0,59**	0,68	1,16
load_inf0.3_sup0.1	**0,53**	0,73	1,38
load_inf0.4_sup0.1	**0,61**	0,67	1,10

After varying the load value, we experiment with removing n loads from the grid. We randomly choose several loads, remove them from the network and evaluate the model (Table 2). Each row in the table is a test in which the name specifies the number of loads that have been removed. Our cost deviation

[1] `https://github.com/anlopez94/opf_gnn_ppo`

is lower or similar than the DCOPF even by eliminating almost 50% of the loads. Finally, in Table 3 we show the results of creating networks from the original one by removing one or more electrical lines (edges). Each row in the table is a test in which the name specifies the number of power lines removed.

Table 2: Results on case IEEE 30 removing loads from base case.

	% DRL+OPF perf.	%DCOPF perf.	ratio
load_1	**0,67**	0,72	1,07
load_2	**0,71**	0,71	1,00
load_3	**0,67**	0,67	1,00
load_4	1,06	**0,68**	0,65
load_5	**0,61**	0,64	1,05
load_8	1,10	**0,63**	0,52

Table 3: Results on case IEEE 30 removing edges from base case.

	% DRL+OPF perf.	%DCOPF perf.	ratio
edge_1	**0,73**	0,77	1,05
edge_2	**0,41**	1,16	2,83
edge_3	0,62	**0,61**	0,99
edge_4	**0,65**	0,88	1,36
edge_5	**0,90**	0,96	1,07
edge_8	**0,59**	0,89	1,51

In experiments removing electrical lines (Table 3) as more power lines are removed (more than 8), sometimes, the agent does not find a good feasible solution (no convergence). When we experimented with changing the load values in the second test (Table 1), we observed that increasing the loads by more than 10% caused the tests to fail to converge. With the other changes in topology, 100% of tests converged, so we can conclude that our model is capable of generalizing to unseen topologies (based on the trained one).

7 DISCUSSION

We have successfully designed a solution to address the OPF, capable of generalization, utilizing DRL and GNN. The network topology has been modified, and we have demonstrated that the agent can identify a strong solution (with performance closely aligned to the current industry standard DCOPF), ensuring that this solution is both feasible and compliant with the constraints. Thanks to the design of GNN, it can be trained on various cases and subsequently applied to different scenarios. In this paper, we validate that this architecture effectively tackles OPF, showcasing the generalization capability of our solution by considering modifications to the network scenario encountered during training (including different loads and a reduction in the number of edges). By integrating these two technologies for the first time, we conclude that their combination is feasible, leveraging the advantages of both. Our findings indicate that the proposed architecture represents a promising initial step toward solving the OPF. Future work could explore the incorporation of additional features in the node representation, such as the maximum and minimum allowable voltage and model other types of electrical generation.

ACKNOWLEDGMENTS

This research is supported by the Industrial Doctorate Plan of the Department of Research and Universities of the Generalitat de Catalunya, under Grant AGAUR 2023 DI060.

REFERENCES

Cao, D., Hu, W., Xu, X., Wu, Q., Huang, Q., Chen, Z., and Blaabjerg, F. (2021). Deep reinforcement learning based approach for optimal power flow of distribution networks embedded with renewable energy and storage devices. *Journal of Modern Power Systems and Clean Energy*, 9(5):1101–1110.

Diehl, F. (2019). Warm-starting ac optimal power flow with graph neural networks. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, pages 1–6.

Donnot, B., Guyon, I. M., Schoenauer, M., Marot, A., and Panciatici, P. (2017). Fast power system security analysis with guided dropout. *ArXiv*, abs/1801.09870.

Donon, B., Clément, R., Donnot, B., Marot, A., Guyon, I., and Schoenauer, M. (2020). Neural networks for power flow: Graph neural solver. In *Electric Power Systems Research*, volume 189, page 106547.

Donon, B., Donnot, B., Guyon, I., and Marot, A. (2019). Graph Neural Solver for Power Systems. In *IJCNN 2019 - International Joint Conference on Neural Networks*, Budapest, Hungary.

Fraunhofer, IEE, U. o. K. (2022). Pandapower documentation.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

Li, J., Zhang, R., Wang, H., Liu, Z., Lai, H., and Zhang, Y. (2021). Deep reinforcement learning for optimal power flow with renewables using graph information. *ArXiv*, abs/2112.11461.

Liao, W., Bak-Jensen, B., Pillai, J. R., Wang, Y., and Wang, Y. (2022). A review of graph neural networks and their applications in power systems. *Journal of Modern Power Systems and Clean Energy*, 10(2):345–360.

Mary, A., Cain, B., and O'Neill, R. (2012). History of optimal power flow and formulations. *Fed. Energy Regul. Comm.*, 1:1–36.

Munikoti, S., Agarwal, D., Das, L., Halappanavar, M., and Natarajan, B. (2024). Challenges and opportunities in deep reinforcement learning with graph neural networks: A comprehensive review of algorithms and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 35(11):15051–15071.

Owerko, D., Gama, F., and Ribeiro, A. (2020). Optimal power flow using graph neural networks. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5930–5934.

Pan, X., Chen, M., Zhao, T., and Low, S. H. (2022). Deepopf: A feasibility-optimized deep neural network approach for ac optimal power flow problems. In *IEEE Systems Journal*, pages 1–11.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20:61–80.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2015). High-dimensional continuous control using generalized advantage estimation. *ArXiv*, abs/1506.02438.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347.

Thurner, L., Scheidler, A., Schäfer, F., Menke, J.-H., Dollichon, J., Meier, F., Meinecke, S., and Braun, M. (2018). Pandapower—an open-source python tool for convenient modeling, analysis, and optimization of electric power systems. *IEEE Transactions on Power Systems*, 33(6):6510–6521.

Wood, A. J., Wollenberg, B. F., and Sheble, G. B. (2013). *Power Generation, Operation, and Control*. John Wiley & Sons, Hoboken, NJ, USA, 3rd edition.

Zhen, H., Zhai, H., Ma, W., Zhao, L., Weng, Y., Xu, Y., Shi, J., and He, X. (2022). Design and tests of reinforcement-learning-based optimal power flow solution generator. *Energy Reports*, 8:43–50. 2021 The 8th International Conference on Power and Energy Systems Engineering.

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

Zhou, Y., Zhang, B., Xu, C., Lan, T., Diao, R., Shi, D., Wang, Z., and Lee, W.-J. (2020b). A data-driven method for fast ac optimal power flow solutions via deep reinforcement learning. *Journal of Modern Power Systems and Clean Energy*, 8(6):1128–1139.