

A Graph Reinforcement Learning Approach for Volt-Var Control in Power Distribution Systems

Supplementary Materials

Xian Yeow Lee^{1,2}, Soumik Sarkar¹, Yubo Wang²
Iowa State University¹, Siemens Technology²
xylee@iastate.edu, soumiks@iastate.edu, yubo.wang@siemens.com

November 18, 2021

1 Training detail and hyper-parameters

In this section, we provide additional description of the hyper-parameters used to train both Dense-PPO and Graph-PPO. We based the implementation of the PPO agent of an open-source library PFRL [1]. For Graph-PPO, we implement the graph convolution layers using another open source library, DGL [2]. Both Dense-PPO and Graph-PPO utilizes the same architecture consisting of separate networks for the actor and value function. For Dense-PPO, both actor and value networks consists of three dense layers (with the exception of 8500Node system, where four dense layers were used) with ReLU activation functions and 64 hidden units. The Graph-PPO agent has three graph convolutional layers instead of the dense layers (with the exception of 8500Node system, where four graph convolution layers were used) with ReLU activation functions and graph nodal embeddings of 64 units. After the final layer of graph convolution, we use a mean-pooling layer to get the average nodal embeddings of all the nodes in the networks before passing the logits to a dense layer to output the control actions. Table 1 tabulates the additional parameters used to trained both policies.

Variable	Value
Optimizer	Adam
Learning rate	3E-4
Discount factor	0.95
ϵ -clip value	0.2
Batch size	64
Model update interval	512
Entropy coefficient	0.01

Table 1: Parameters used for training Dense-PPO and Graph-PPO

2 Additional results

2.1 Additional results for missing/noisy observations

The figures below serves to complement the results shown in Tables 1 and 2 in the main article by providing a more detailed illustration of the rewards achieved by Dense-PPO and Graph-PPO under the conditions of missing node voltages (communication failures), noisy node voltages (measurement misalignment) and missing edges in the graph representation. Specifically, Figures 1, 2, 3 and 4 visualized the experiments for missing node observations, Figures 5, 6, 7 and 8 illustrates the experiments for noisy node voltages. Note that while Graph-PPO tend to perform slightly worse than Dense-PPO specifically for the 8500Node system with full observations, our conclusion that Graph-PPO is more robust remains valid when comparing the sensitivity of the rewards in the setting of missing or noisy observations. Based on the trends observed for 13Bus, 34Bus and 123Bus systems, we believe that additional training iterations and/or using augmentation and local readout functions as discussed in the case study will result in Graph-PPO performing on-par with Dense-PPO for the 8500Node system.

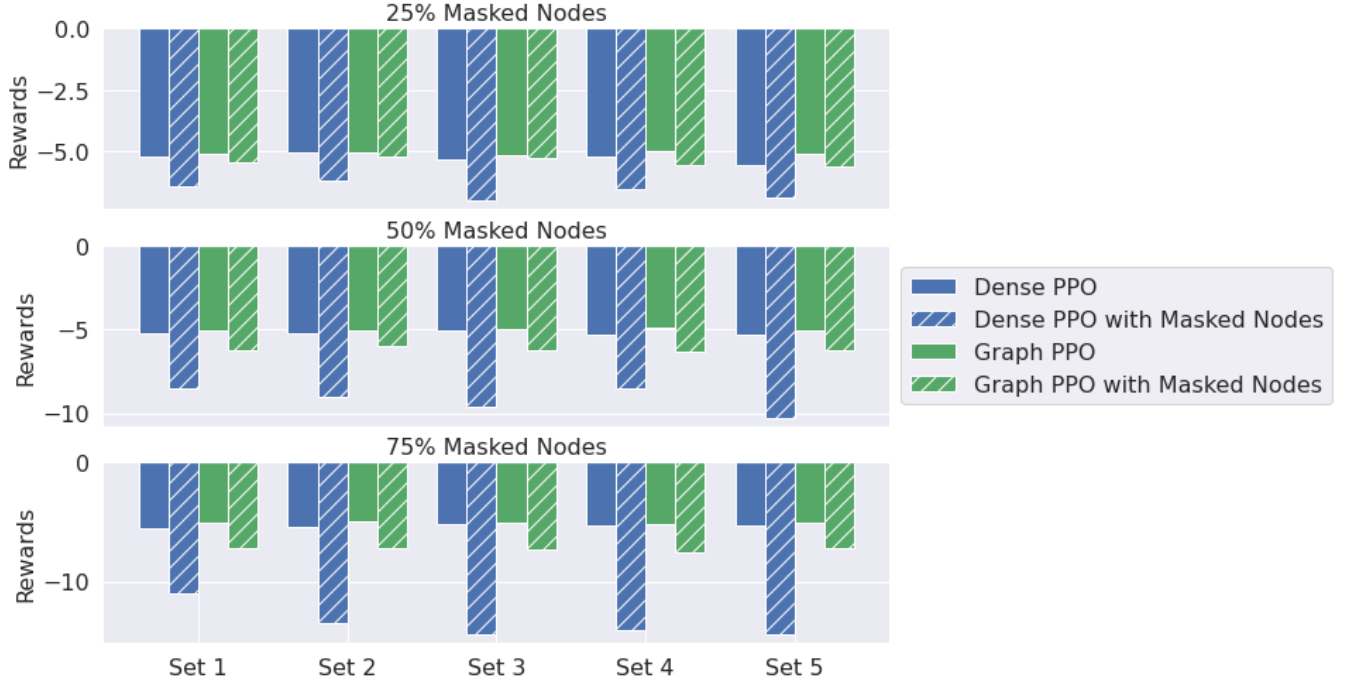


Figure 1: Effects of missing observations on the rewards of Dense-PPO and Graph-PPO for 13Bus system. Solid bars represents the performance under nominal conditions with full observations and hatched bars represents the performance with missing observations. Each group of bar-plot represents the masking of a different subset of nodes.

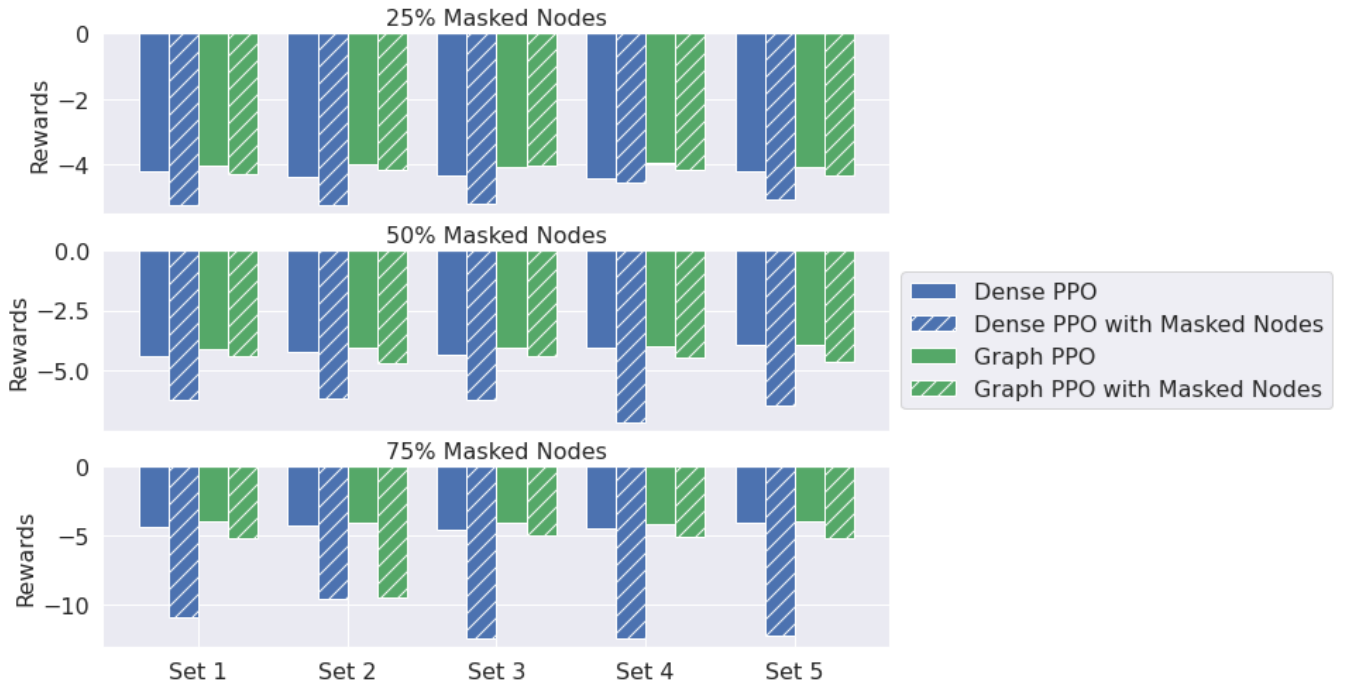


Figure 2: Effects of missing observations on the rewards of Dense-PPO and Graph-PPO for 34Bus system. Solid bars represents the performance under nominal conditions with full observations and hatched bars represents the performance with missing observations. Each group of bar-plot represents the masking of a different subset of nodes.

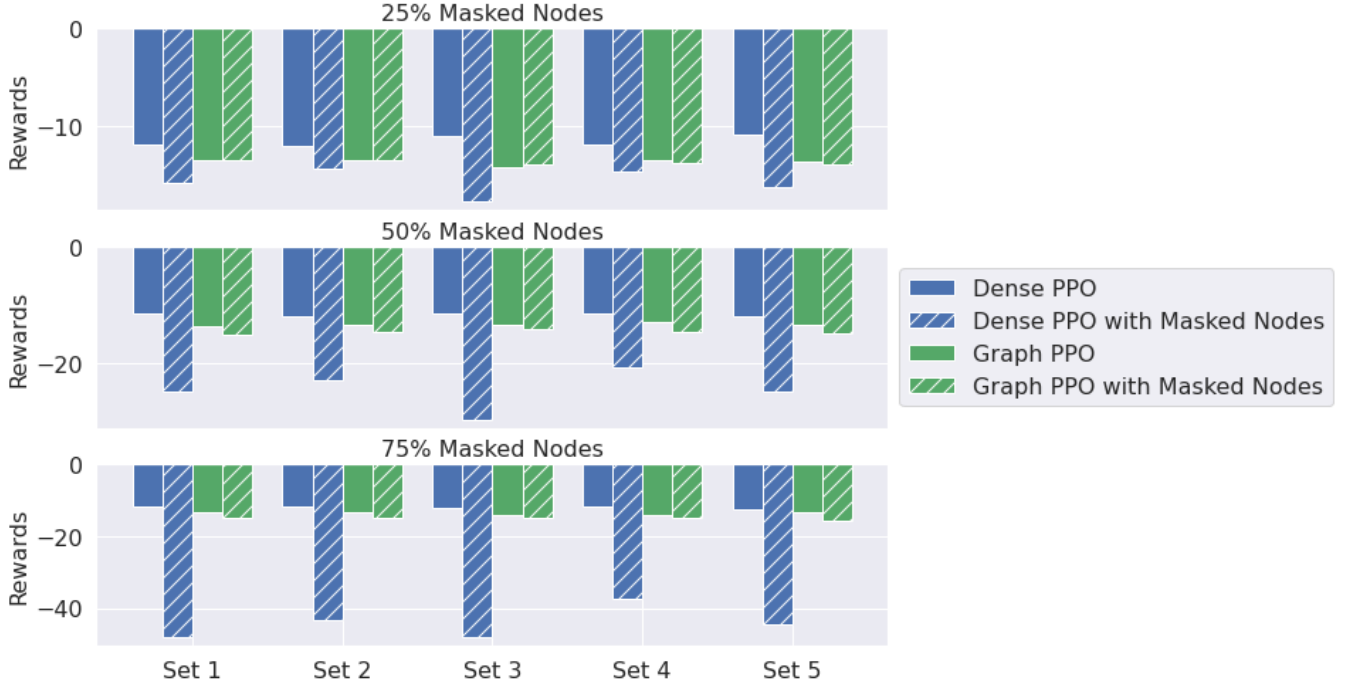


Figure 3: Effects of missing observations on the rewards of Dense-PPO and Graph-PPO for 123Bus system. Solid bars represents the performance under nominal conditions with full observations and hatched bars represents the performance with missing observations. Each group of bar-plot represents the masking of a different subset of nodes.

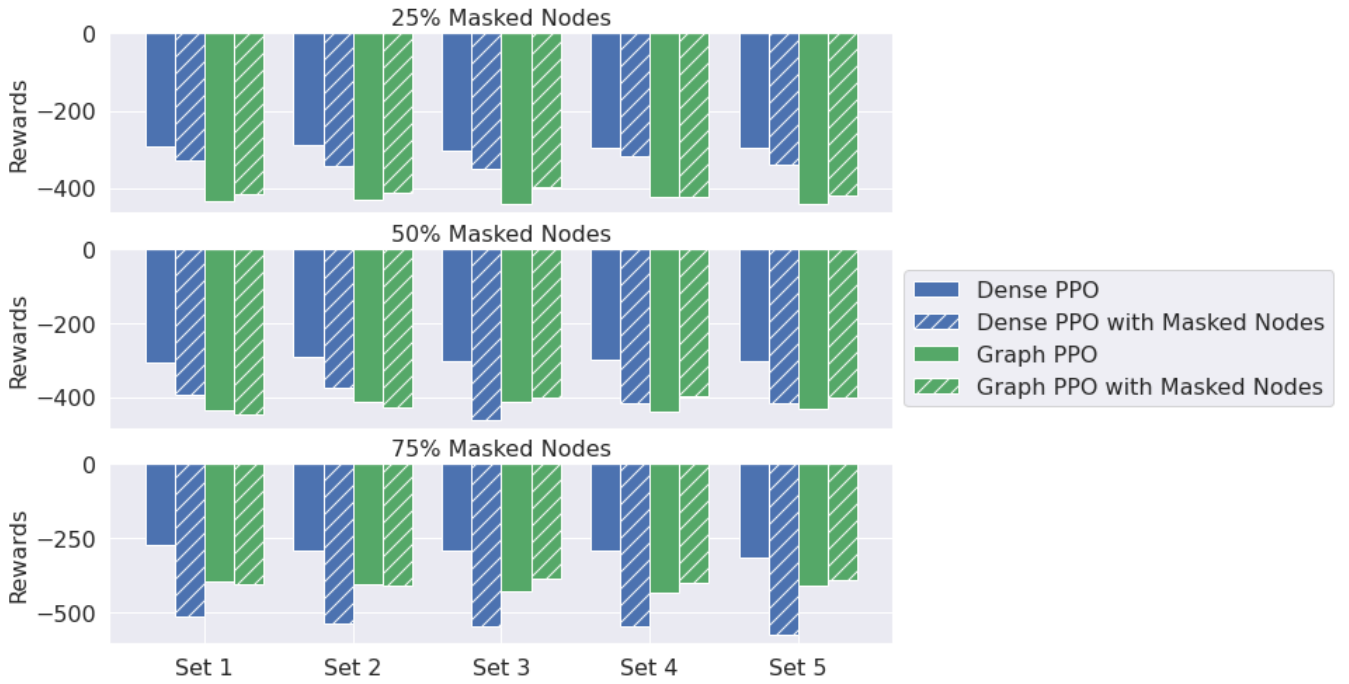


Figure 4: Effects of missing observations on the rewards of Dense-PPO and Graph-PPO for 8500Node system. Solid bars represents the performance under nominal conditions with full observations and hatched bars represents the performance with missing observations. Each group of bar-plot represents the masking of a different subset of nodes.

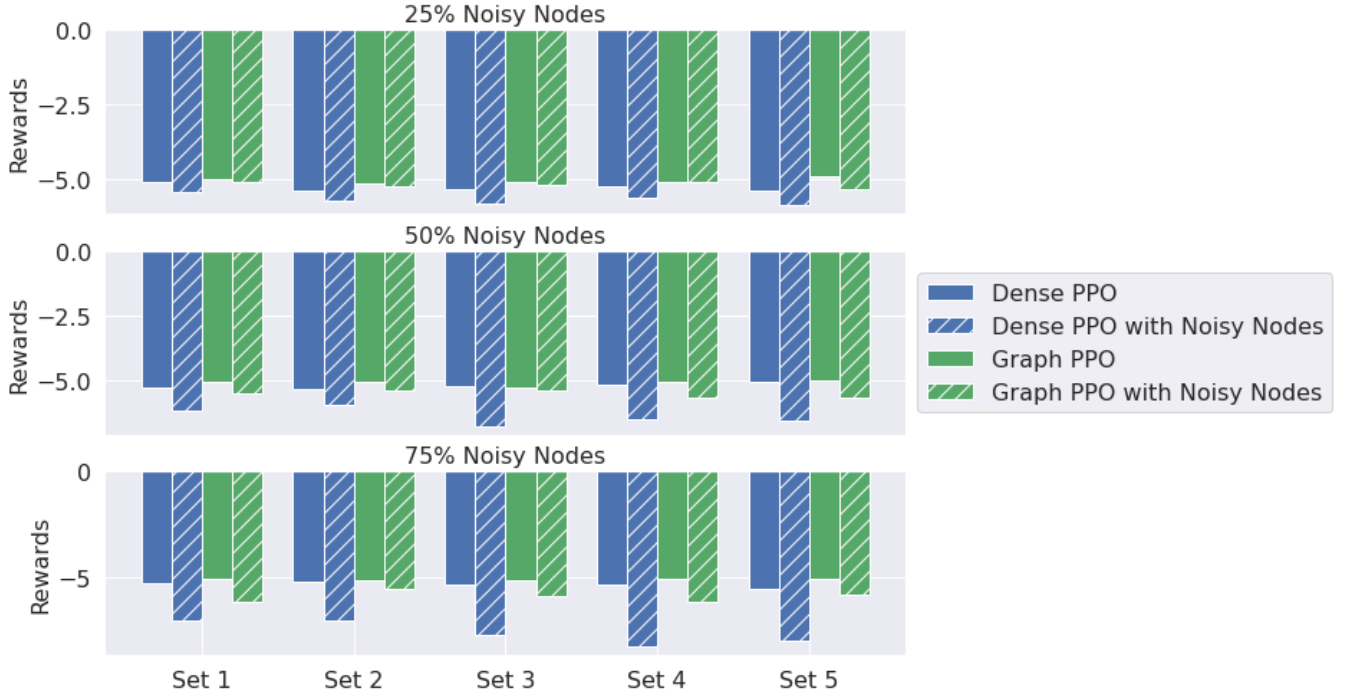


Figure 5: Effects of noisy observations on the rewards of Dense-PPO and Graph-PPO for 13Bus system. Solid bars represents the performance under nominal conditions with clean observations and hatched bars represents the performance with noisy observations. Each group of bar-plot represents noise added to different subsets of nodes.

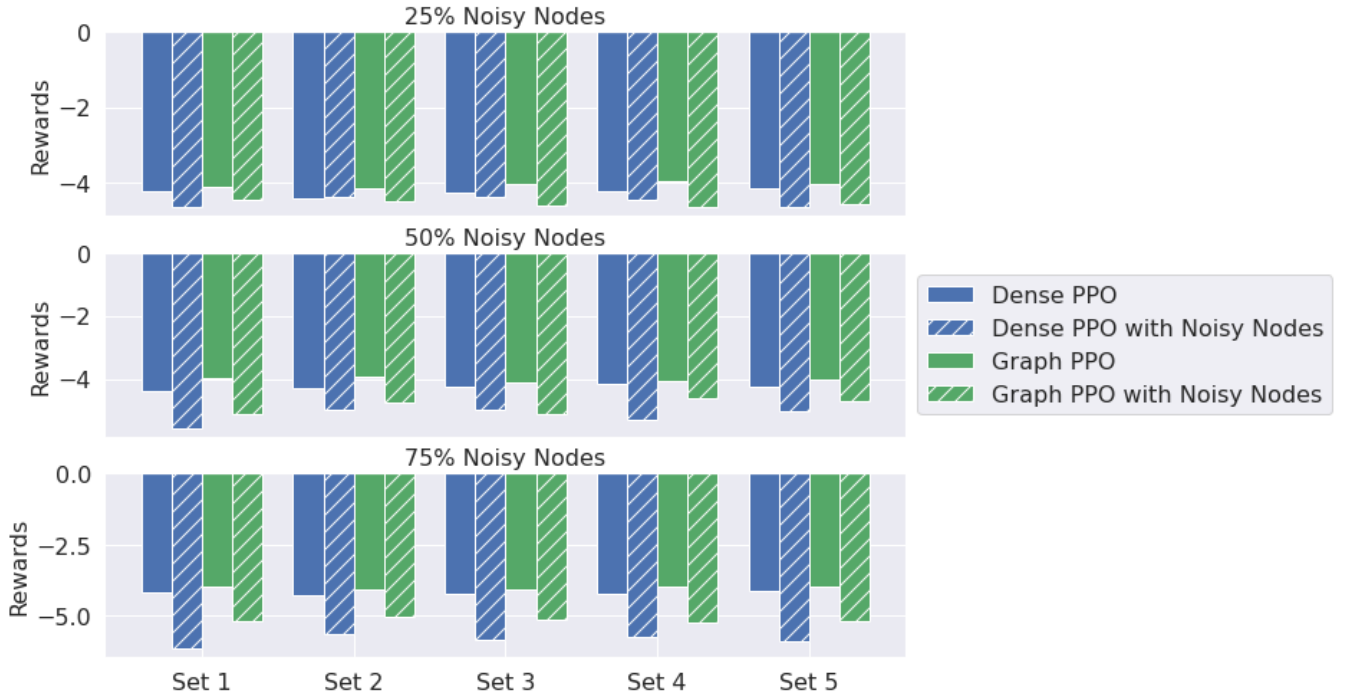


Figure 6: Effects of noisy observations on the rewards of Dense-PPO and Graph-PPO for 34Bus system. Solid bars represents the performance under nominal conditions with clean observations and hatched bars represents the performance with noisy observations. Each group of bar-plot represents noise added to different subsets of nodes.

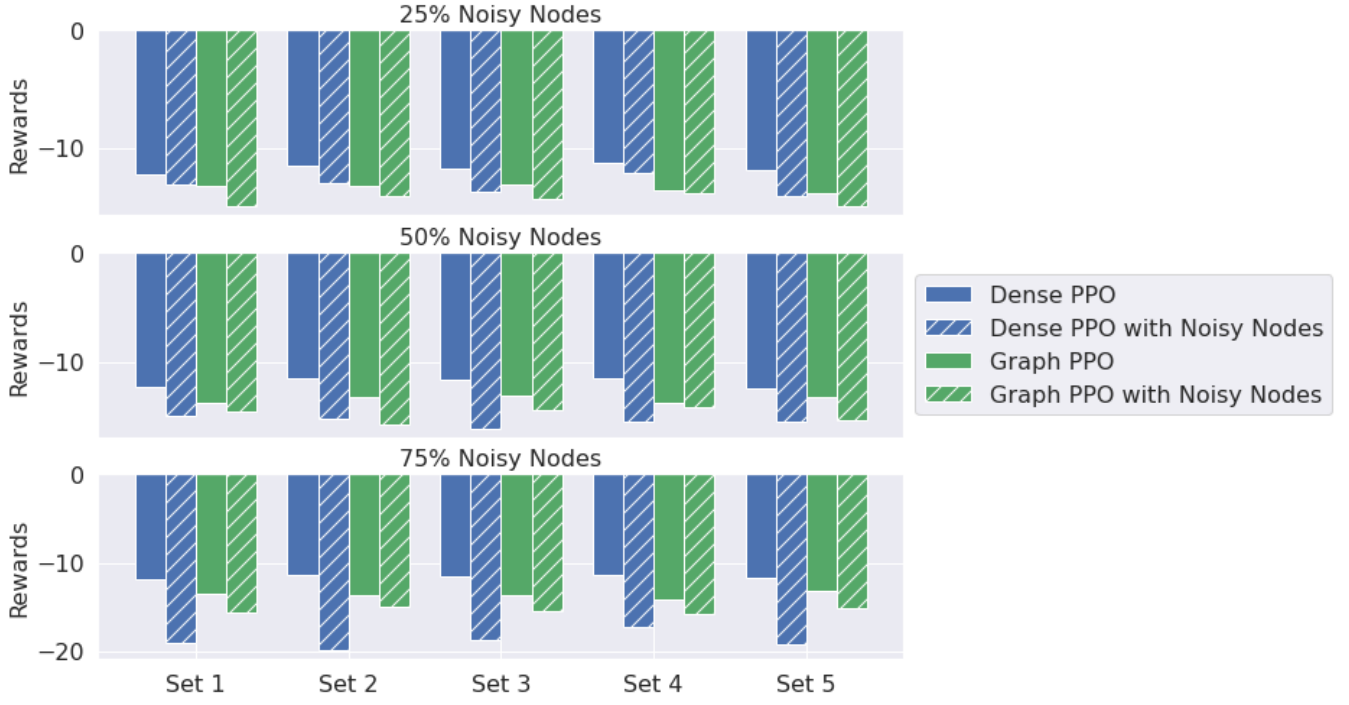


Figure 7: Effects of noisy observations on the rewards of Dense-PPO and Graph-PPO for 123Bus system. Solid bars represents the performance under nominal conditions with clean observations and hatched bars represents the performance with noisy observations. Each group of bar-plot represents noise added to different subsets of nodes.

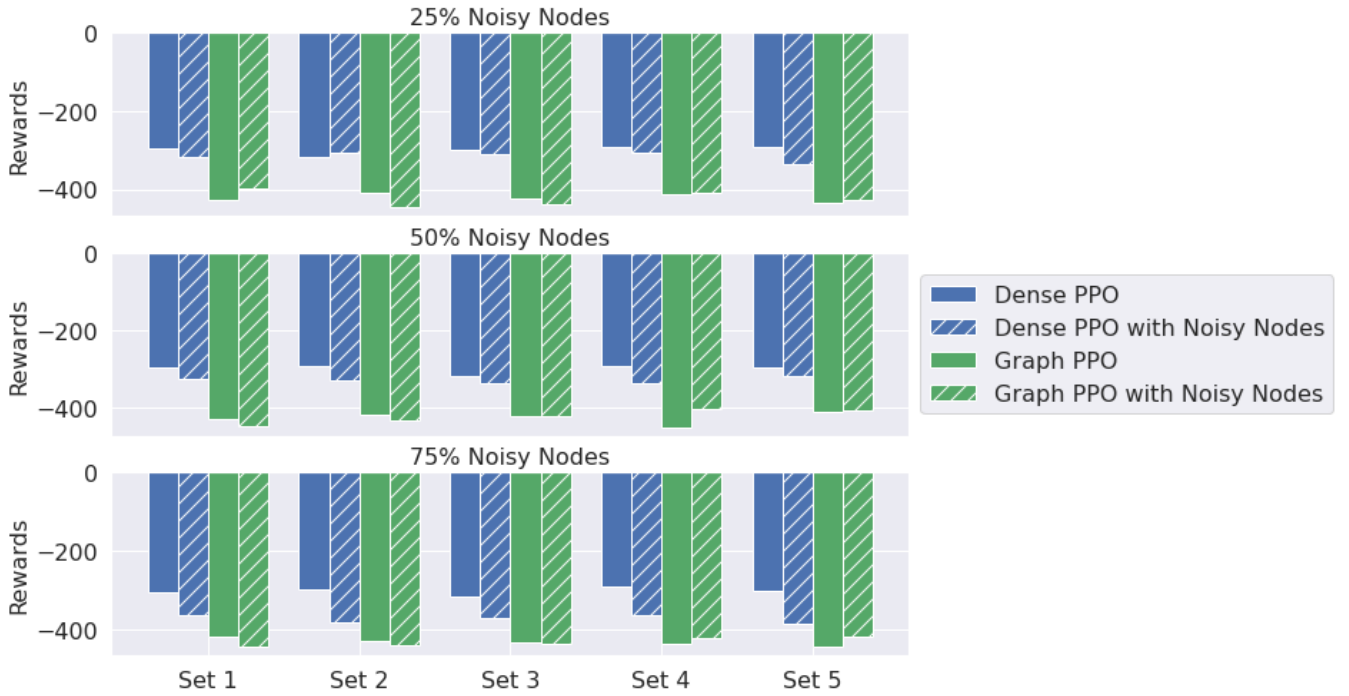


Figure 8: Effects of noisy observations on the rewards of Dense-PPO and Graph-PPO for 8500Node system. Solid bars represents the performance under nominal conditions with clean observations and hatched bars represents the performance with noisy observations. Each group of bar-plot represents noise added to different subsets of nodes.

2.2 Robustness of Graph-PPO with Augmentation and Local Readout

Figure 9 shows a more detailed breakdown of the statistics presented in Table 3. in the main article for each subset of random nodes. To obtain the reward values for each subset of random nodes being masked or perturbed with noise, we ran ten episodes using five agents trained with different random seeds and averaged the rewards.

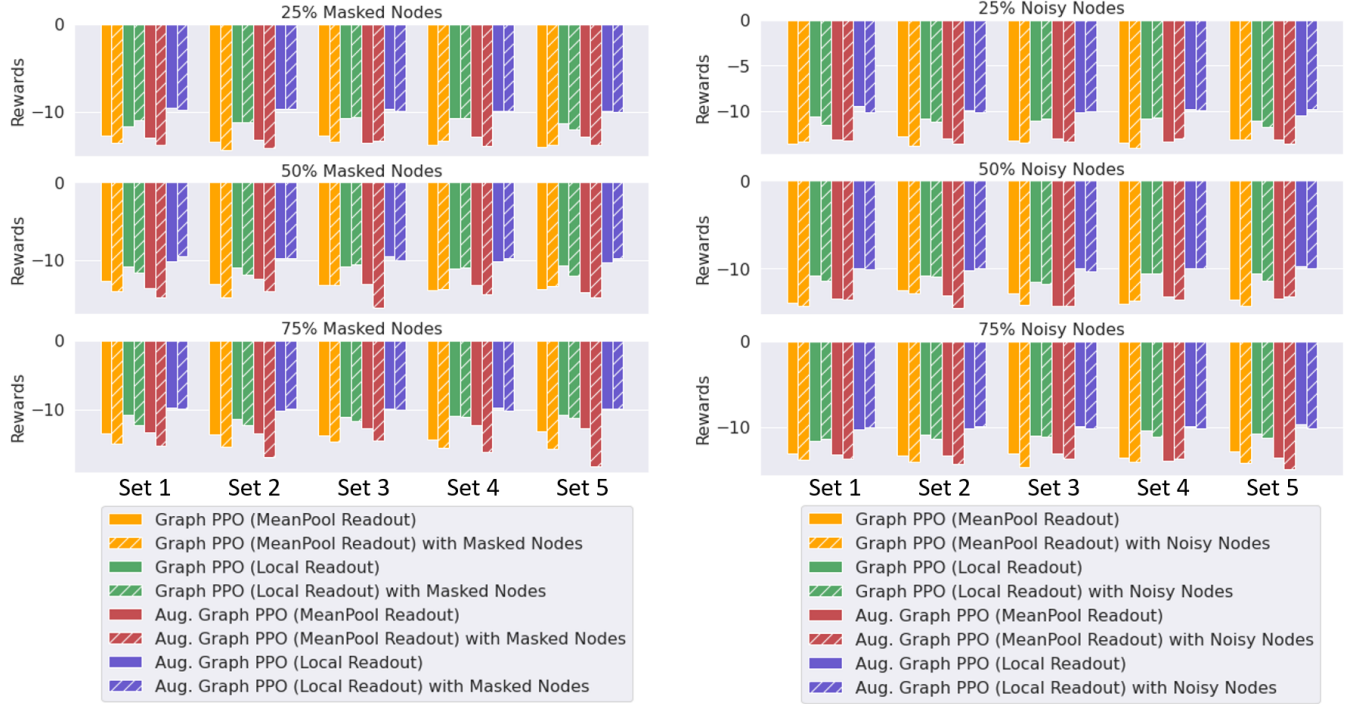


Figure 9: Effects of missing and noisy observations on the rewards of variants of Graph-PPO for 123Bus using augmentation versus no augmentation and mean-pooling versus local readout. Solid bars represents the performance under nominal conditions with clean observations and hatched bars represents the performance with missing or noisy observations. Each group of bar-plot represents noise added to different subsets of nodes. Among the different variants, Graph-PPO with augmentation and local readout remains the most robust.

2.3 Robustness of Graph-PPO to missing edges

As an additional study, we investigate what happens when some connectivity between the nodes is removed in the graph representation, thus preventing some nodes from propagating information to their neighbors. To test this, we randomly remove the edges between different subsets of nodes during testing and observe the performance of Graph-PPO as compared to its performance when observing the complete graph representation.

Figures 10, 11, 12 and 13 shows the experiments for missing edges specific to vanilla Graph-PPO with mean-pooling readout. As observed, the rewards achieved by Graph-PPO remain largely similar even with a significant amount of edges removed. Table 2, which summarizes the effect of edge removal on Graph-PPO also reveals a similar observation, with the largest effect coming from the 34Bus system with 75% of the edges removed. This tells us that the policy learned by Graph-PPO is not entirely dependent on the connectivity but also on the nodal embeddings. Even with the connectivity information corrupted, the graph-based policy is still able to produce a decent control strategy based on the nodal embeddings. This implies that the graph representation does not harm the performance when edges are corrupted but affords a much more robust control strategy when the nodal embeddings are corrupted, thus providing a net benefit.

System	25%	50%	75%
13 Bus	-3 ± 1	-2 ± 1	-3 ± 2
34 Bus	-4 ± 1	-11 ± 3	-21 ± 4
123 Bus	-5 ± 4	-5 ± 3	-5 ± 4
8500 Bus	-7 ± 3	-7 ± 4	-5 ± 3

Table 2: Values in the table denote the percent difference of rewards for Graph-PPO in the nominal setting with true graph representation. Each column represents a scenario with certain percentage of edges removed from a randomly sampled subset of nodes.



Figure 10: Effects of missing edges in the graph representation on the rewards of Graph-PPO for 13Bus system. Solid bars represents the performance under nominal conditions with the true graph and hatched bars represents the performance on a graph with corrupted edges. Each group of bar-plot represents the same graph but with different sets of randomly corrupted edges.

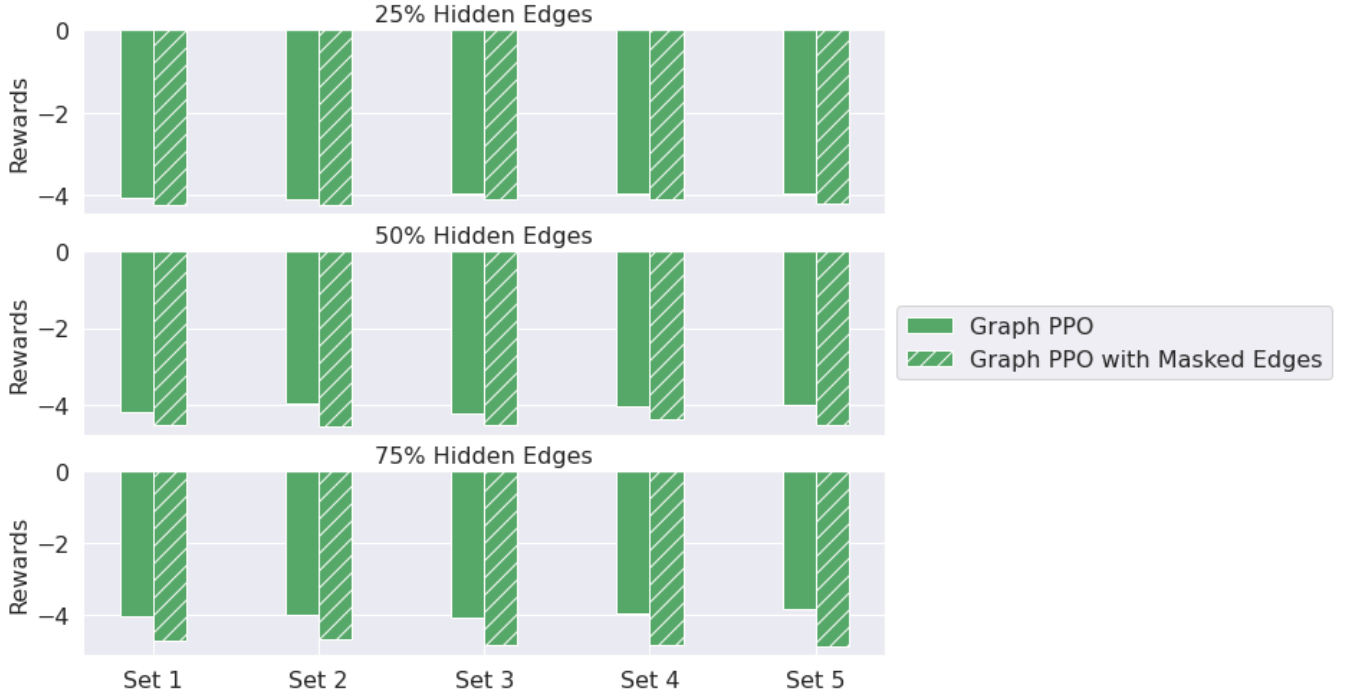


Figure 11: Effects of missing edges in the graph representation on the rewards of Graph-PPO for 34Bus system. Solid bars represents the performance under nominal conditions with the true graph and hatched bars represents the performance on a graph with corrupted edges. Each group of bar-plot represents the same graph but with different sets of randomly corrupted edges.

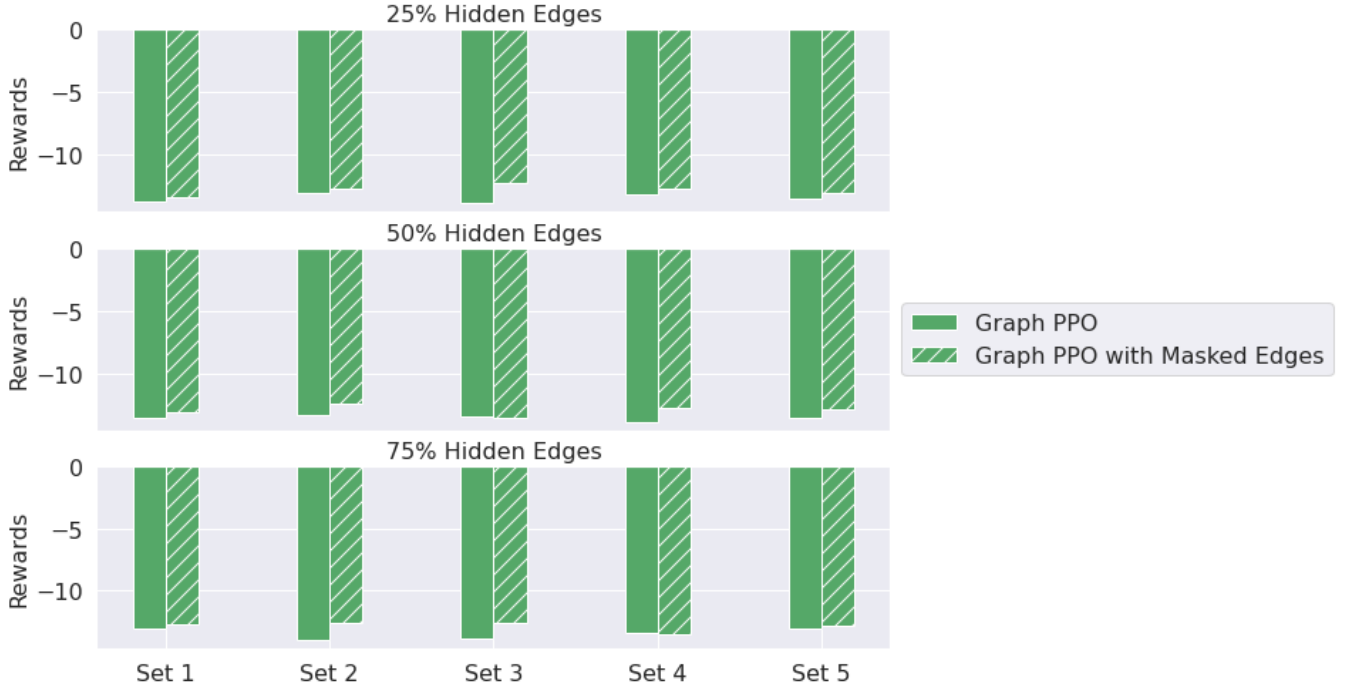


Figure 12: Effects of missing edges in the graph representation on the rewards of Graph-PPO for 123Bus system. Solid bars represents the performance under nominal conditions with the true graph and hatched bars represents the performance on a graph with corrupted edges. Each group of bar-plot represents the same graph but with different sets of randomly corrupted edges.

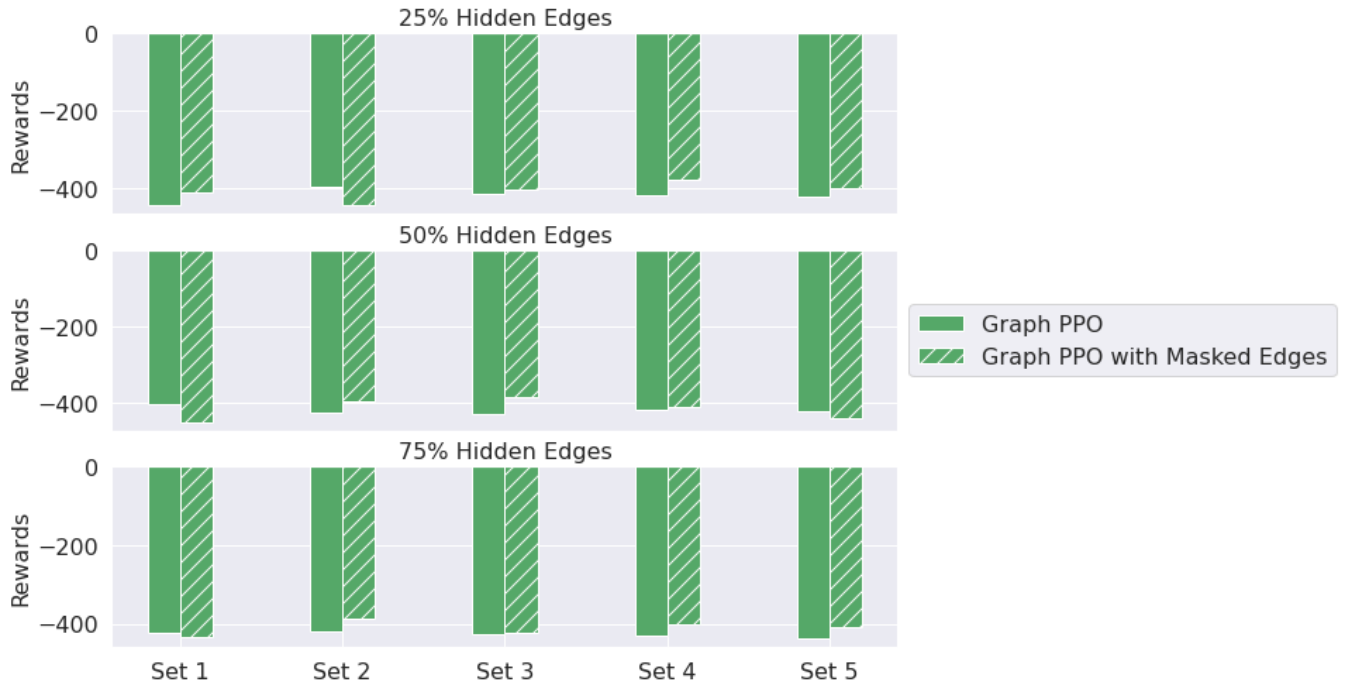


Figure 13: Effects of missing edges in the graph representation on the rewards of Graph-PPO for 8500Node system. Solid bars represents the performance under nominal conditions with the true graph and hatched bars represents the performance on a graph with corrupted edges. Each group of bar-plot represents the same graph but with different sets of randomly corrupted edges.

References

- [1] Yasuhiro Fujita, Prabhat Nagarajan, Toshiki Kataoka, and Takahiro Ishikawa. Chainerrl: A deep reinforcement learning library. *Journal of Machine Learning Research*, 22(77):1–14, 2021.
- [2] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.