Troubled Waters: When Graph-based Models Perform Worse in Practice

Anonymous Submission

Abstract

Deep neural networks, including graph-based models, often exhibit unexpected failure modes when confronted with slight domain or task variations. We explore the pitfalls of applying graph neural networks to a structured prediction task in an industrial dataset. This challenge highlights how seemingly small design choices and low-level interactions can degrade performance, underscoring the tension between theoretical elegance and practical realities.

1 Introduction

Deep learning provides powerful tools for complex tasks. However, its real-world deployment exposes vulnerabilities and negative or inconclusive results [**??**]. In particular, graph neural networks (GNNs) promise strong representational capabilities [**??**], yet we report here a case study in which GNN-based methods underperform simpler baselines due to subtle hyperparameter choices and unanticipated data nuances.

Our key contributions include: (1) Observing a decrease in performance with increased GNN complexity. (2) Highlighting factors like architectural hyperparameters and data pre-processing that exacerbated the difficulties. (3) Demonstrating how these negative results guide practical improvements for future deployments.

2 Related Work

Many studies praise the representational capacity of GNNs [**??**], but there are fewer reports on their real-world failures. Practical pitfalls involving domain shifts or unexpected data patterns have also been noted in conventional deep learning [**??**]. Our work extends these findings by pinpointing conditions under which GNNs do not outperform simpler methods [**?**], contextualizing such failures within industrial applications.

3 Method / Problem Discussion

We focus on an industrial structured prediction task with heterogeneous feature interactions. Our baseline is a feed-forward network that consumes concatenated features, while the GNN attempts to learn pairwise relationships among these features encoded as graph edges.

We configured a standard GNN pipeline, experimenting with layer depths and embeddings. Despite tuning, the GNN unexpectedly lagged behind, suggesting overfitting on node-level representation and sensitivity to certain hyperparameters.

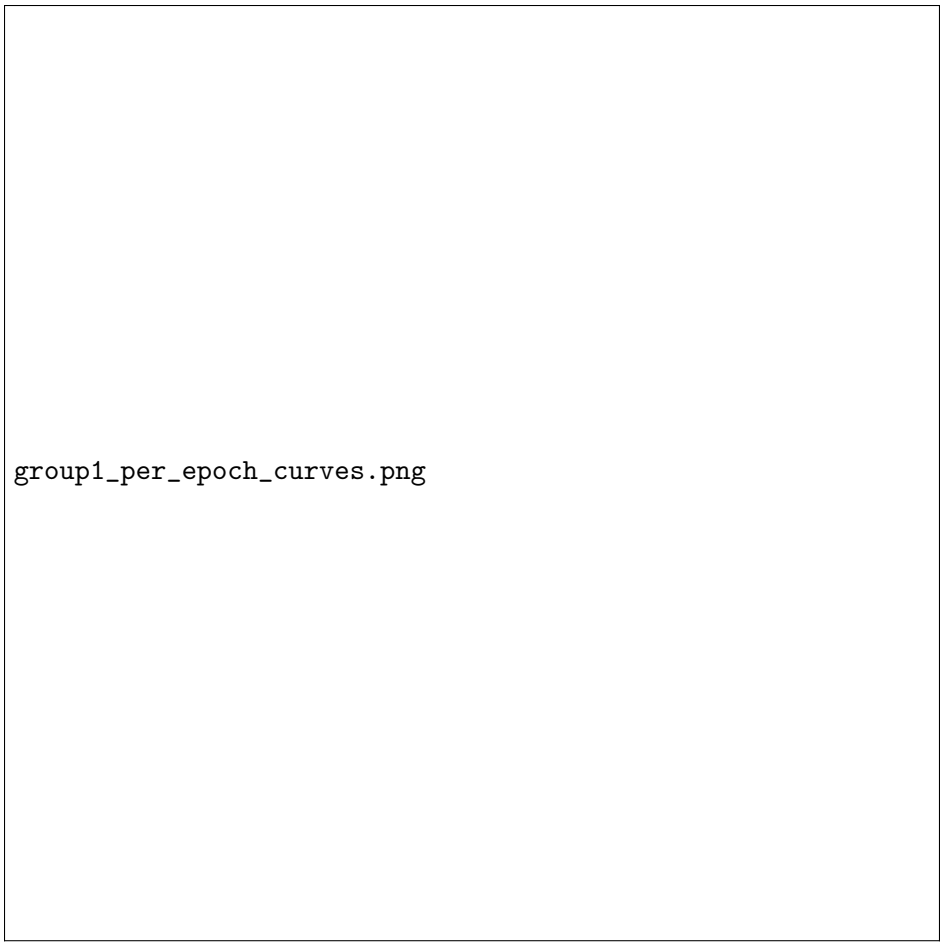

Figure 1: Training (left), validation (center), and cumulative weighted accuracy (right) for baseline vs. GNN. Notice the GNN's validation volatility and inconsistent improvements.

4 Experiments

We train on a private dataset with three distinct sets of features. The baseline consistently showed better or comparable performance, while the GNN performed erratically under small perturbations. Figure 1 captures per-epoch curves for both methods, highlighting oscillations in validation accuracy and increased volatility with the GNN. Further confusion matrix details are in the appendix.

5 Conclusion

Although GNNs show promise, pragmatic application in this setting yielded worse results than simpler baselines. We identified risk factors such as unstable hyperparameters and possibly misaligned edge representations. Future work should test systematic data augmentations and explore ways to mitigate overfitting. We hope that documenting these pitfalls prevents others from encountering similar implementation surprises.

References

Appendix

Here we provide additional experiment details, confusion matrices (Figure 2), and extended hyperparameter specifications.

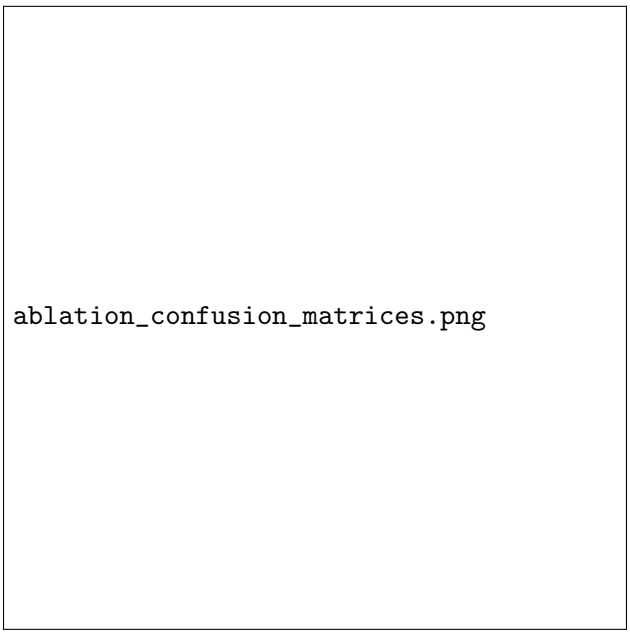

Figure 2: Confusion matrices for ablation studies. Removing batch normalization or positional embeddings can aggravate the performance gap.