Advanced Topics in Deep Learning - Assignment 2

Class 1, Group 8
Sachin Srivathsa Satish Kumar, Alexandros Kyriakopoulos, Gabriela Kirejczyk

22 October, 2024

1 Introduction

The objective of this paper is to replicate the key experiments from the paper *"Graph Neural Networks are Inherently Good Generalizers: Insights by Bridging GNNs and MLPs"* (ICLR 2023) by Yang et al., 2023. This study focuses on verifying whether Propagational Multi-Layer Perceptrons (PMLPs) have a better ability to generalize compared to traditional Multi-Layer Perceptrons (MLPs), and whether PMLPs exhibit similar performance to Graph Neural Networks (GNNs). By replicating this study, we aim to assess if the model introduced by Yang et al., 2023 is reliable and successfully addresses the issue of GNNs being computationally intensive.

This paper aims to address the following research questions:

1. Can we replicate the efficiency gains of PMLPs over MLPs and compare their performance to GNNs?
2. How do PMLPs generalize across different types of datasets?
3. What are the resource costs involved in reproducing these experiments?

The accuracy and computational efficiency of MLP, PMLP, and GNN models will be compared. For all models, hyperparameters will be tuned to find the most optimal configurations. The original authors' code and hyperparameter search can be found at https://github.com/AlexandrosKyr/PMLP-Replication-Study-ADL.

2 Datasets and Model Descriptions

2.1 Datasets

The datasets we use represent a diverse range of sizes, node types and graph structures, including both homophilic and heterophilic graphs, which allows us to evaluate the generalization capabilities of the models across different settings.

By selecting these datasets, we aim to replicate the original paper's results in both small and large-scale environments and test the generalization ability of the models in both homophilic and heterophilic settings. The combination of Cora, Coauthor-CS and OGBN-Arxiv offers insights into homophilic performance, while Wisconsin challenges the models' robustness in a heterophilic graph structure. This subset ensures that we cover a wide range of graph structures and sizes, crucial for testing the validity of the claims in the original paper.

Dataset	Type	Nodes	Edges	Classes
Cora	Citation Network	2,708	5,429	7
Wisconsin	Webpage Network	251	515	5
OGBN-Arxiv	Citation Network	169,343	1,166,243	40
Coauthor-CS	Co-authorship Network	18,333	81,894	15

Table 1: Details of the datasets used in the replication study

2.2 Model Descriptions

The models compared in Yang et al., 2023 and in the following study are MLP, PMLP and GNN. MLP is the most basic form of a feed-forward neural network composed of fully connected layers. It processes each node independently without taking into consideration the connections between them, therefore completely ignoring the graph structure of the data. According to Yang et al., 2023, eq. 2 is the equation of MLP with a series of FF layers. GNN, on the other hand, is designed to capture the relationships between the data points. On top of the fully connected layers, GNN introduces message-passing layers, thanks to which each node can aggregate information from the neighbouring nodes. This enables the model to capture both the individual node features and the graph structure, allowing for incorporating information from different data points when updating each node. According to Yang et al., 2023, eq. 2 is a general formulation of a GNN:

$$(\text{MP}): \quad \tilde{\mathbf{h}}_u^{(l-1)} = \sum_{v \in \mathcal{N}_u \cup \{u\}} a_G(u,v) \cdot \mathbf{h}_v^{(l-1)}, \quad (\text{FF}): \quad \mathbf{h}_u^{(l)} = \psi^{(l)}\left(\tilde{\mathbf{h}}_u^{(l-1)}\right), \tag{1}$$

$$\hat{y}_u = \psi^{(L)}\left(\cdots \psi^{(1)}(\mathbf{x}_u)\right) = \psi(\mathbf{x}_u). \tag{2}$$

PMLP's structure is a combination of both of the above. During training and validation, it follows the structure of an MLP ignoring the graph context of the nodes. However, during the testing phase, it introduces the message-passing layers as in the GNN structure. These layers allow the model to aggregate node information from the graph during inference, enabling it to benefit from the graph structure without the computational cost of message passing during training. This structure allows PMLP to balance the efficiency of MLPs with the ability to capture relationships between nodes of GNNs.

As Yang et al., 2023 argue, the placement of the message passing layers can change the performance of the model. Therefore by combining the research of Gasteiger et al., 2022 and Wu et al., 2019 on GNNs, Yang et al., 2023 propose three different versions of PMLP, namely PMLP_{GCN}, PMLP_{SGC}, and $\text{PMLP}_{\text{APPNP}}$. PMLP_{GCN} integrates a message-passing layer in each layer's feedforward process, PMLP_{SGC} includes multiple message-passing layers in the first layer and finally $\text{PMLP}_{\text{APPNP}}$ adds them in the last layer.

3 Implementation Details

3.1 Codebase

The replication experiments were conducted using the codebase developed by the authors (Yang et al., 2023), which we accessed from their public GitHub repository. We chose to replicate Version A of the PMLP implementation because it offers a more flexible and general approach by encompassing three models (MLP, PMLP, and GNN) in a single class. The `use_conv` flag allows us to easily switch between architectures, enabling or disabling message passing during inference. This versatility makes Version A more adaptable compared to the other versions, which are more specialized and less flexible for our replication study.

The Python files that we made some slight alterations to are: `main.py` and `dataset.py`; this was done so that we would be able to load and run the hyperparameter search using our implementation of `hyperparametersearch.py`. More specifically, we made minor modifications to the original code, primarily adjusting file paths and dataset loading routines to match the Google Cloud Platform environment. No significant changes were made to the model's logic or the training process. The execution of the Python scripts was performed through the terminal where we called the `hyperparametersearch.py` file which was programmed to conduct a grid search through a subset of hyperparameters that was searched by Yang et al., 2023.

The codebase was executed on the Google Cloud Platform (GCP), which provided the necessary computational resources for training the models on larger datasets like OGBN-Arxiv and Coauthor-cs. We ran the experiments using GCP's virtual machines (VMs) with the following specifications: 4 vCPUs, 16 GB RAM and TensorFlow:2.11 environment.

3.2 Challenges

One of the significant challenges we encountered during our experiments was the limited capacity in which we could access high-performance GPUs on the Google Cloud Platform (GCP) due to high GPU demand. This limited our ability to conduct extensive hyperparameter searches and run multiple models for more datasets. Despite the resource challenges, our initial hyperparameter searches, guided by the PMLP authors' comments on GitHub and insights from the paper, helped us identify some more promising ranges.

We also faced compatibility issues while setting up the environment, requiring dependency adjustments to avoid conflicts, especially in logging and package versions.

4 Results & Discussion

4.1 Result Analysis

	Wisconsin	Cora	Coauthor-CS	OGBN-arxiv
Heterophily	High	Low	Low	Low
Authors results				
GNN_{GCN}	63.92 ± 4.51	74.82 ± 1.09	91.79 ± 0.35	69.04 ± 0.18
PMLP_{GCN}	66.67 ± 2.77	75.86 ± 0.93	91.76 ± 0.27	63.74 ± 2.28
MLP	83.53 ± 3.28	55.30 ± 0.58	87.51 ± 0.51	53.86 ± 0.28
Our results				
GNN_{GCN}	60.00 ± 5.65	76.06 ± 0.90	90.36 ± 1.10	64.70 ± 0.42
PMLP_{GCN}	65.10 ± 4.47	76.08 ± 1.26	90.43 ± 0.56	60.26 ± 1.96
MLP	85.49 ± 3.28	54.06 ± 0.65	85.57 ± 1.16	50.60 ± 0.26

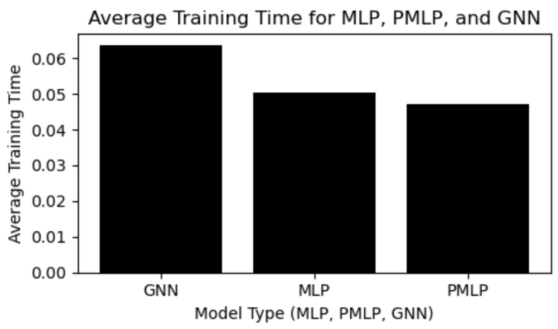
Table 2: Comparison of results across different datasets

For the Wisconsin and Cora datasets, our results show slight deviations from the authors' original findings. On Wisconsin, our models performed slightly worse: GNN achieved 60.00 compared to 63.92, and PMLP reached 65.10 versus 66.67. In contrast, our Cora results closely match the authors', with GNN at 76.06 (authors' 74.82) and PMLP at 76.08, slightly above their 75.86. For the Coauthor-CS and OGBN-Arxiv datasets we observed similar trends with minor deviations. On Coauthor-CS, GNN achieved 90.36 compared to 91.79, and PMLP 90.43 versus 91.76. On OGBN-Arxiv, both models showed a performance drop: GNN at 64.70 (original 69.04) and PMLP at 60.26 (compared to 63.74). These larger datasets might pose challenges for model scalability and training consistency, potentially explaining the discrepancies. Differences in hyperparameters or initialization could also account for the variations. However, overall trends in performance between MLP, PMLP, and GNN remain consistent, largely influenced by the heterophily level.

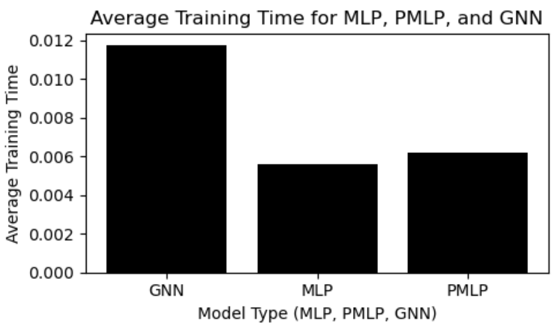
For the remainder of the analysis, we will split the analysis of the results of our replication study into the results for smaller and for larger datasets.

4.2 Small datasets

For our analysis of small datasets, we selected Cora and Wisconsin, which offer a balanced mix of homophilic and heterophilic graph structures. Together, these two datasets allow us to assess how well the models generalize across different types of graph structures.



(a) Wisconsin training time



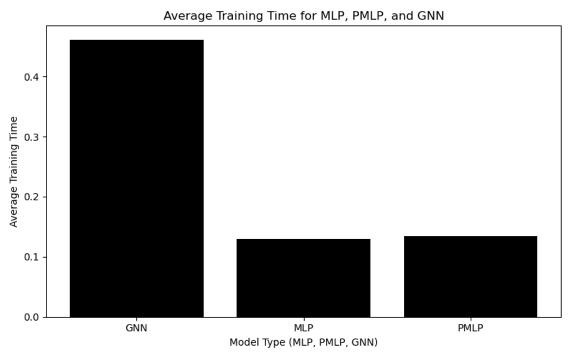
(b) Cora training time

Figure 1: Training times for Cora and Wisconsin for different models

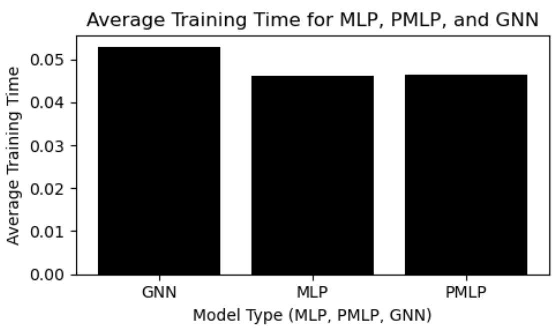
On the heterophilic Wisconsin dataset, where neighboring nodes usually belong to different classes, MLP performed better than both GNN and PMLP. Since MLP treats each node independently, it isn't affected by potentially misleading information from neighboring nodes, which can negatively affect models like GNN and PMLP that rely on message passing. This explains why MLP had the best performance and the shortest training time on Wisconsin, making it more efficient for heterophilic graphs. On the other hand, on the homophilic Cora dataset, both GNN and PMLP outperformed MLP. In homophilic settings models like GNN and PMLP benefit from aggregating information from neighbors making them perform better. In terms of training time, GNN was the slowest due to its message-passing layers, while MLP was the fastest, and PMLP fell somewhere in between.

4.3 Larger datasets

For our analysis of larger datasets, we chose Coauthor-CS and OGBN-Arxiv because they represent large, real-world homophilic graphs making them suitable for evaluating model generalization and scalability on large graphs.



(a) Arxiv training time



(b) Coauthor-cs training time

Figure 2: Training times for Arxiv and Coauthor-cs for different models

On the larger datasets, OGBN-Arxiv and Coauthor-CS, the results show clear differences in both training time and performance. On OGBN-Arxiv, GNN had the longest training time but performed similarly to PMLP, while MLP underperformed due to its inability to leverage the homophilic structure of the dataset. In contrast, on Coauthor-CS, the training times were more balanced across models, and all models—GNN, PMLP, and MLP—achieved similar performance, suggesting that both MLP and GNN-based methods can effectively handle this dataset's structure.

5 Conclusion

Our replication study largely confirms the findings of the original paper, proving that their research is reproducible. It supports the claims of Yang et al., 2023, confirming that PMLPs generalize well and perform comparably to GNNs. Moreover, regarding different datasets, on homophilic datasets like Cora, PMLPs and GNNs outperformed MLPs, while on heterophilic datasets like Wisconsin, MLPs performed slightly better, showing that it may be more suitable for homophilic types. Lastly, PMLPs proved to be computationally efficient compared to GNNs, particularly on larger datasets, as they reduce the need for costly message passing during training. Despite resource constraints affecting our ability to fully explore hyperparameters, the results were successfully reproduced and confirm the reliability and scalability of the model introduced by Yang et al., 2023.

References

Gasteiger, J., Bojchevski, A., & Günnemann, S. (2022). Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*. https://doi.org/10.48550/arXiv.1810.05997

Wu, F., Zhang, T., de Souza Jr, A. H., Fifty, C., Yu, T., & Weinberger, K. Q. (2019). Simplifying graph convolutional networks. *arXiv preprint arXiv:1902.07153*. https://doi.org/10.48550/arXiv.1902.07153

Yang, C., Wu, Q., Wang, J., & Yan, J. (2023). Graph neural networks are inherently good generalizers: Insights by bridging GNNs and MLPs. *arXiv preprint arXiv:2212.09034*. https://doi.org/10.48550/arXiv.2212.09034