Investigating Transfer Learning for Link Prediction in Graph Neural Networks

Mandelz Thomas*
Institute for Data Science
FHNW
Windisch, Switzerland
thomas.mandelz@fhnw.ch

Zwicky Jan*
Institute for Data Science
FHNW
Windisch, Switzerland
jan.zwicky@fhnw.ch

Perruchoud Daniel
Institute for Data Science
FHNW
Windisch, Switzerland
daniel.perruchoud@fhnw.ch

Heule Stephan
Institute for Data Science
FHNW
Windisch, Switzerland
stephan.heule@fhnw.ch

***Abstract*—Graph Neural Networks (GNNs) emerged as powerful tools for learning representations of graph-structured data, increasingly applied to various domains. Despite their growing popularity, the transferability of GNNs remains underexplored. Transfer learning showed remarkable success in traditional deep learning tasks, enabling faster training and enhanced performance. Although GNNs are gaining popularity and are being applied in many areas, their transferability in link prediction is not well-studied.**

This research investigates the applications of transfer learning in link prediction using GNNs, focusing on enhancing model performance as well as training efficiency through pre-training GNN models, followed by fine-tuning. Specifically, we train Graph Convolutional Network (GCN), GraphSAGE and Graph Isomorphism Network (GIN) architectures and investigate the benefits of transfer learning by pre-training and fine-tuning models on public data (i.e. on ogbn-papers100M and ogbn-arxiv datasets). Reference models, constructed with identical capacity and trained on the same datasets, ensure a fair comparison to the fine-tuned models. Jumpstart and asymptotic performance are used to determine the transferability between models, while training time ratios measure training efficiency.

Our findings show that transfer learning improves fine-tuned model performance, boosting jumpstart scores in relation to the reference models range from 0.63 (jumpstart) for GCN, 0.47 for GraphSAGE, 0.48 for GIN, while also reducing training time up to 15 times for GraphSAGE.

***Index Terms*—Graph Neural Networks, Graph Theory, Link Prediction, Transfer Learning, Deep Learning**

I. Introduction

Graph data is characterised by entities (nodes or vertices) that are interconnected by relationships (edges or links), forming a complex topological structure. A key task in the analysis of graph data is link prediction, which involves predicting the existence of a link between two nodes [1], [2]. This task is critical in numerous real-world applications, including social network analysis, recommendation systems, and biological network inference. Link prediction on such data requires models that can effectively capture the intricate patterns inherent within the graph. Traditional methods often fall short in capturing these complexities [3], leading to the development of more sophisticated approaches including Graph Neural Networks (GNNs).

*These authors contributed equally to this work, thanks to FHNW, University of Applied Sciences Northwestern Switzerland, I4DS

GNNs emerged as powerful tools in the field of machine learning for graph-structured data [4], [5], [6]. Their ability to learn representations that encapsulate both node features and the topology of the graph led to significant advances in various tasks, including link prediction. However, training GNNs often requires large amounts of labelled data, which may not always be available. Additionally, GNNs typically have long training times. In domains like computer vision or natural language processing such challenges were addressed by leveraging transfer learning, a technique that involves transferring knowledge from one domain to another to enhance performance on tasks where data is scarce.

This research investigates the application of transfer learning to improve link prediction using GNNs, when initialising the model with the parameters from the pre-trained model [7]. While GNNs and transfer learning were extensively studied separately [8], [9], their intersection, particularly in the context of link prediction, remains underexplored. By examining how pre-trained models can be fine-tuned for specific link prediction tasks, this research aims to bridge this gap and provide insights into the potential benefits of combining these approaches.

The scope of this research focuses on the feasibility and effectiveness of transfer learning for link prediction using GNNs, rather than pursuing a broader foundation model approach. The research is grounded in practical applications, specifically utilising citation networks as a case study for link prediction. We investigated transfer learning by comparing a reference model, trained from scratch on the target task, with a fine-tuned model, which adapted a pre-trained model by adjusting its parameters to improve performance. To ensure a fair comparison, reference models were constructed with identical capacity and trained on the same datasets as the fine-tuned models. The pre-trained models were initially trained on a large citation network to capture broad features for downstream tasks. Prior research focuses on node/graph-level transfer, leaving open questions about how pre-trained GNNs adapt to link prediction in real-world networks [10]. Our research examines this gap, evaluating whether pre-training on large graphs enhances initialisation efficiency (jumpstart) and asymptotic performance. To explore the critical aspects of transfer learning in the context of link prediction using GNNs,

we focus on two main research questions:

- Can transfer learning with a pre-trained model enhance jumpstart performance or asymptotic performance (see Figure 3), when compared to a reference GNN model?
- Does increasing the model capacity lead to improved model performance in transfer learning?

Our code repository[1] and experiments[2] are publicly available. The subsequent sections of this paper are structured as follows: A succinct overview of the literature can be found in **Related Work**. In **Data & Methods**, we detail our datasets, pre-processing techniques, methodology, model architectures, and evaluation metrics. In **Experiments & Discussion**, we demonstrate the effectiveness of our approach and quantitatively compare the different (pre-)training strategies. Conclusions and outlooks are summarised in **Conclusion**.

II. Related Work

To the best of our knowledge, this is the first study to evaluate the feasibility of transfer learning specifically in the context of link prediction using GNNs.

While transfer learning emerged as a state-of-the-art technique in various data modalities and tasks, its potential in link prediction remains largely unexplored. The following section aims to provide an overview of existing transfer learning research and methodologies in related fields.

In contrast to the extensive research on transfer learning with pre-trained model in Computer Vision and Natural Language Processing, the field of transfer learning in graph prediction tasks saw comparatively little exploration [11]. Graph prediction tasks encompass node-level, graph-level, and edge-level predictions. Within these categories, a multitude of tasks can be addressed using GNNs [12]. This variety underscores the potential for diverse transfer learning approaches.

Several studies assessed transferability of GNNs in node- and graph-level prediction tasks [10], [13], [14], [15], [16]. However, edge-level tasks are underexplored, and to the best of our knowledge, there was no research conducted on transfer learning with GNNs specifically for link prediction.

A framework for transferring knowledge in GNNs, particularly focusing on link-classification, is presented in [13]. It introduces Ego-Graph Information Maximisation to capture essential graph information for transferable GNN training, supported by theoretical analysis and synthetic experiments that validates its effectiveness. In contrast, [14] addresses the challenge of effectively pre-training GNNs for tasks characterised by distributional differences and limited labels. They propose a novel strategy along with self-supervised methods for pre-training GNNs at both the node and the graph levels, enabling the simultaneous learning of local and global representations.

A pre-training framework for GNNs, emphasising classification tasks for nodes, graphs, and links, is presented in [15]. This framework employs techniques such as denoising link reconstruction, centrality score ranking, and cluster preserving tasks on synthetic graphs to effectively capture transferable structural information. By requiring less labeled data and domain-specific features, the pre-trained GNN achieves high performance across various downstream classification tasks. Building on this work, [16] introduces a GPT-GNN framework, which leverages generative pre-training to initialise GNNs with self-supervised learning. GPT-GNN presents a novel self-supervised attributed graph generation task to pre-train GNNs, enabling them to capture both structural and node feature properties of graphs.

Applications of transfer learning in the context of graph and node classification are investigated in [10]. The research demonstrates the effectiveness of transfer learning with GNNs for their tasks and highlight the importance of selecting the appropriate GNN architecture to enhance transferability. Both real-world and synthetic data are used to evaluate node and graph classification tasks, with a proposed methodology for transfer learning experimentation and a novel algorithm for synthetic task generation. Their findings indicate that GNNs significantly improve transfer performance, particularly when source and target tasks share community structures.

III. Data & Methods

In this section, we provide a detailed overview of the datasets, the employed methodology, the architectures of our GNNs, and evaluation metrics.

Our datasets are based on Open Graph Benchmark (OGB) datasets [17], we renamed to arXiv CS (*ogbn-arxiv*) and arXiv w/o CS (*ogbn-papers100M*) after pre-processing . In Table I the final datasets are summarised with their respective usage, node and edge counts.

A. Data

This research utilises two citation datasets: for pre-training, we employ the *ogbn-papers100M* dataset, which encompasses the entire arXiv archive, while for fine-tuning, we use the smaller *ogbn-arxiv* dataset which contains arXiv computer science papers (cf. Table I).

ogbn-papers100M: The ogbn-papers100M dataset is a vast directed citation graph containing 111 million papers, indexed by the Microsoft Academic Graph (MAG). Each node represents a paper, with edges indicating citations. The papers have 128 dimensional Word2vec feature vectors (cf. [18]) encapsulating title and abstract information. Approximately 1.5 million of its nodes represent arXiv papers, many of them

TABLE I
Dataset Overview

Dataset	Nodes	Edges
OGB Datasets		
ogbn-arxiv	169 343	1 166 243
ogbn-papers100M	111 059 956	1 615 685 872
Preprocessed Datasets		
arXiv CS (reference & fine-tuning)	169 343	1 166 243
arXiv w/o CS (pre-training)	1 324 684	12 823 767

[1]https://github.com/tmandelz/link-prediction-in-graphs
[2]https://www.comet.com/transfer-learning-link-prediction

labelled with a research field from arXiv. This dataset serves as our source domain and is used for pre-training.

ogbn-arxiv: The ogbn-arxiv dataset is a directed graph of computer science arXiv paper citations from MAG. It shares its graph structure and node feature construction method with ogbn-papers100M. This dataset serves as our target domain. The fine-tuning and reference models will be trained on this dataset.

B. Pre-processing

The pre-processing includes two steps, i.e., cleaning (cf. *Reduction*) and partitioning of data (cf. *Data split*) described in the following paragraphs.

Reduction: To fully ensure the prevention of data leakage from our pre-trained model to the fine-tuned model, we took steps to refine the ogbn-papers100M dataset by removing nodes that could potentially cause information leakage.

Firstly, we eliminated all nodes categorised as 'arXiv Computer Science'[3] (arXiv CS), as these nodes are highly likely to overlap with those in the *ogbn-arxiv* dataset. This reduction removed about 0.2% of nodes and 0.05% of edges. Secondly, we removed all papers that were not categorised under any arXiv label. This additional measure was implemented to minimise the risk of data leakage, given that we cannot fully categorise these instances and, therefore, cannot guarantee they are not already present in the arXiv CS category. This second reduction removed about 98.81% of nodes and 99.21% of edges, ensuring that the pre-training dataset consists exclusively of arXiv papers, with the arXiv CS category fully excluded.

Data split: We followed the splitting methodology of the *ogbl-citation2* dataset [17], i.e., we simulate a practical scenario in citation recommendation and split the edges based on publication time. Imagine a user writing a new paper who already cited various existing papers but seeks recommendations for additional references. The most recent papers (published in 2019 for *ogbn-arxiv* and 2018 respectively for *ogbn-papers100M*) are designated as the source papers from which we aim to suggest references. For each source paper, we omit two papers from its list of references. These omitted papers form two edges, one for validation and one for testing, pointing from the source paper to the omitted ones (cf. Figure 1). The remaining edges constitute the training data. If a paper has less than two edges, all of them are put into the trainset. Additionally, 1000 negative samples [17] are randomly drawn for validation and test splits (cf. Figure 2). This approach was applied to both datasets, the *ogbn-arxiv* and *ogbn-papers100M* (after the reduction).

C. Model Architectures

We compared three prominent GNN architectures: Graph Convolutional Networks (GCN), Graph Isomorphism Networks (GIN), and GraphSAGE for link prediction tasks. These models were selected based on prior research in transfer learning by [10], which highlighted their effectiveness in similar tasks. Below, we outline the key characteristics, advantages, and limitations of each architecture.

GCN [4], employ convolution operations to aggregate features, including neighbourhood information. This architecture excels in capturing local graph structure, however faces challenges with respect to scalability and efficient handling of large-scale graphs [19].

GIN [6] address some limitations of GCNs by offering enhanced discriminative power. GINs are designed to be as powerful as the Weisfeiler-Lehman graph isomorphism test. GINs demonstrate superior performance in distinguishing different graph structures compared to GCNs and GraphSAGE; however, [19] and [20] report higher computational resource utilisation for GINs compared to these models.

GraphSAGE [5] introduces an inductive learning framework to the GNN domain. It employs a neighbourhood sampling and aggregation technique, which allows for efficient processing of large-scale graphs [19], [5]. However, the sampling process can lead to a loss of information, as the sampled neighbourhoods may not fully capture the structure of the graph, potentially resulting in reduced accuracy [21].

D. Evaluation

Negative Sampling: For the evaluation, each of the positive links is evaluated against 1000 randomly selected links. To conserve computing resources, no constraints are imposed on the randomly selected links, except that actual existing links and self-loops cannot be selected.

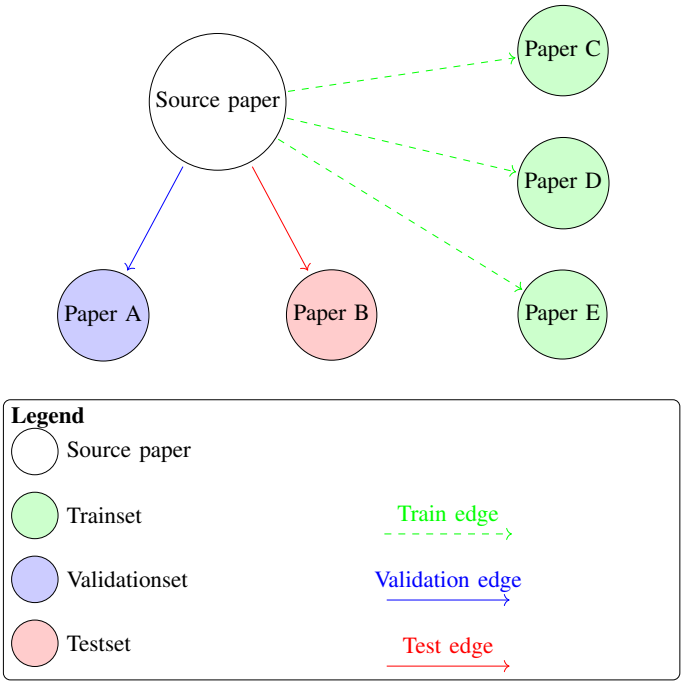

Fig. 1. Schematic of data split, for all papers in the evaluation years (e.g. 2019 for ogbn-arxiv). Per paper, one edge is put into the validation set (blue) and a second edge is put into the test set (red). All remaining edges are put into the trainset (green).

[3]arXiv computer science category

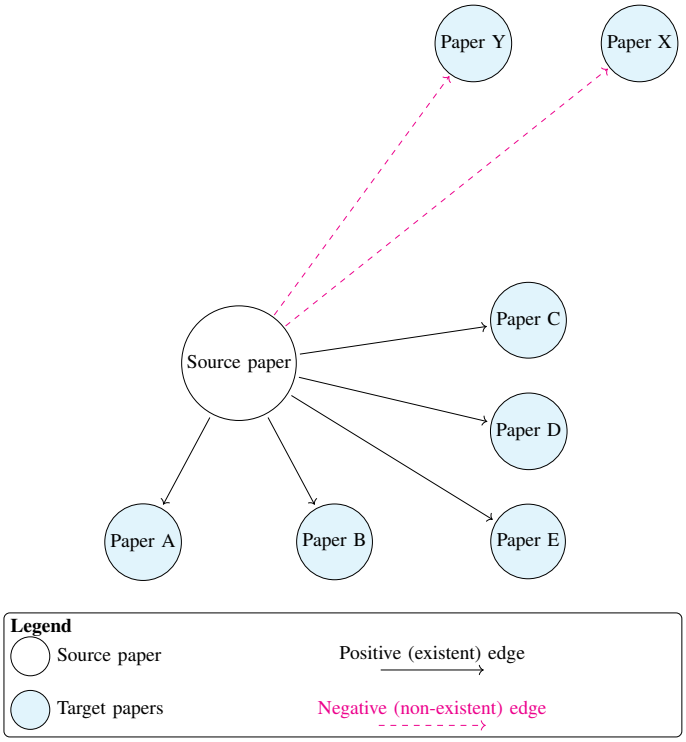

Fig. 2. Schematic of negative edge sampling, 1000 non-existent edges per paper are sampled as negative samples. In the schematic these are magenta-dashed edges, the existing edges are in black.

Metrics: We measure model performance based on mean reciprocal rank (MRR) and assess the transfer learning abilities of the model as in [10] and [22], i.e. we use jumpstart and asymptotic performance (cf. Figure 3).

Training Time Improvement: We evaluate training time improvements by first measuring the test MRR of the reference model after n hours of training (e.g. 2 hours). Then, we record the time it takes for the fine-tuned model to reach the same test MRR. The ratio of these times serves as an indicator of the training time improvement.

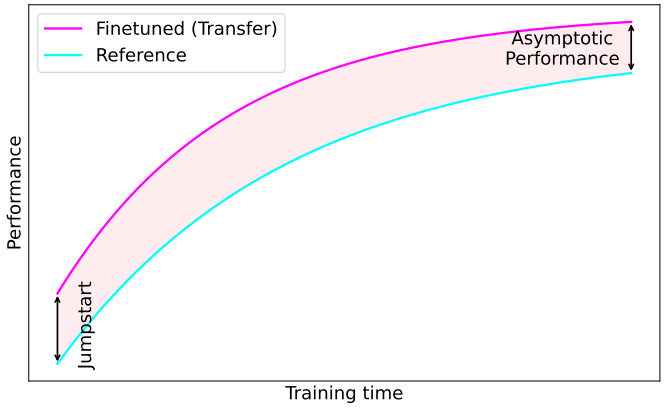

Fig. 3. Illustration of initial and asymptotic performance gains from transfer learning displayed performance (MRR) vs training time (cf. [10]).

IV. Experiments & Discussion

We begin by presenting the performance results of reference, pre-trained, and fine-tuned models. Additionally, we examine the transfer learning metrics between reference model and fine-tuned models.

A. Model Performance

For each model architecture we trained a reference, pre-train and fine-tuned model. Their performance is shown in Table II. GCN achieves an MRR of 0.848 after fine-tuning. GIN performs better than GCN with an MRR of 0.874 after fine-tuning. GraphSAGE performs similarly, with a fine-tuned MRR of 0.885. The higher-capacity GraphSAGE variant achieves the highest fine-tuned MRR of 0.929.

The pre-trained model achieves a lower MRR because its model capacity is insufficient to fully learn the *arXiv w/o CS* dataset. This dataset is more complex as it covers a broader range of fields. Similarly, the fine-tuned model does not show improved performance, as its capacity is insufficient to fully leverage the additional information from the extended dataset. A more detailed analysis of this limitation follows in the subsequent paragraph.

B. Transferability of Models

Our results demonstrate that transfer learning enhances the performance of a fine-tuned model compared to its reference model (cf. Table III). All models benefit from a significant jumpstart: GCN (0.633), GIN (0.479), and GraphSAGE (0.468).

The substantial improvement highlights the similarity between the data distributions, allowing the pre-trained model to begin training with a relatively high MRR metric.

Additionally, the fine-tuned model demonstrates a faster training time compared to the reference model. As shown in Figure 4, the fine-tuned model reaches the same test MRR about 15 times faster than the reference model, requiring approximately 0.13 hours.

The asymptotic performance, does not differ substantially indicating a performance ceiling. Notably, all architectures struggled to overfit on the training data. The training MRR curve is lower than the validation curve, for the GraphSAGE architecture it is illustrated in Figure 5. This suggests a possible constraint in the capacity of the model. In machine learning, the performance metric, like MRR, is typically higher on the training set than on the validation or test sets. However, our results, illustrated in Figure 5, deviate from this norm.

Our explanation for the observed performance discrepancy is threefold. Firstly, as illustrated in Figure 6, the distributions

TABLE II
Model Performance Results (MRR)

Model Architecture	Reference	Pre-trained	Fine-tuned
GCN	0.852	0.808	0.848
GIN	0.897	0.807	0.874
GraphSAGE	0.891	0.803	0.885
GraphSAGE more capacity	0.931	0.836	0.929

TABLE III
TRANSFER LEARNING RESULTS

Model Architecture	Jumpstart	Asymptotic Performance
GCN	0.633	-0.004
GIN	0.479	-0.022
GraphSAGE	0.468	-0.006
GraphSAGE more capacity	0.556	-0.002

of pairwise cosine similarity of embeddings within the training and validation set respectively are shown. The validation set shows a distribution shifted to the right, indicating that the validation nodes are more similar to each other compared to those in the training set. This higher similarity among validation nodes might provide an advantage for the models, making it easier to predict validation edges more accurately than those in the training set.

A second factor contributing to the performance discrepancy is the random sampling of negative links during training. For the validation edges, a fixed set of negative samples is used to evaluate performance. In contrast, for the training set, negative links are resampled at each training step. This resampling introduces additional noise into the training process, making the training set more challenging to learn. However, this noise acts as a regularisation factor, preventing the model from overfitting and ensuring that the evaluation task remains non-trivial. A detailed analysis of the advantages of random sampling can be found in [23].

Finally, in the arXiv CS dataset, all links that do not fall within the computer science domain were removed by OGB. This could have a stronger impact on the training data, as there are fewer links available due to the earlier publication years. Further analysis is needed to comprehensively understand and address this deviation of the norm.

C. Increased Model Capacity

To further investigate the performance limitations observed in the GraphSAGE architecture, we increased model capacity by expanding the hidden channel size from 384 to 1024. This modification aimed to enhance the ability of the model

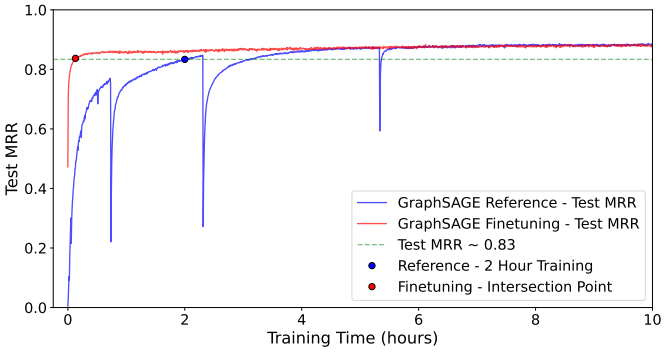

Fig. 4. MRR test curves for the GraphSAGE reference and GraphSAGE fine-tuning models plotted against training time in hours. The blue point marks the test MRR of the reference model at 2 hours of training.

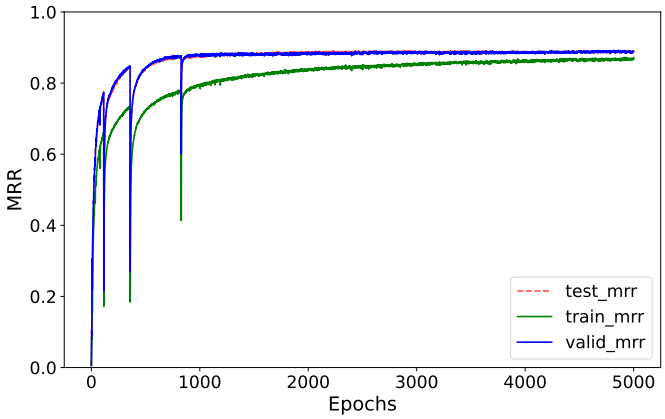

Fig. 5. MRR curves for the GraphSAGE reference model across the train, validation, and test sets. Notably, the train set exhibits a lower MRR compared to the validation and test sets.

to capture more complex patterns in the data. As shown in Table II, the MRR for the GraphSAGE model with more capacity demonstrate a noticeable improvement in performance compared to the original model. The fine-tuned model with more capacity achieves a higher MRR throughout the training process, indicating that the additional parameters allow the model to better fit the training data.

However, as seen in Table III, while the increased model capacity leads to a higher MRR, the asymptotic performance of the model does not show an improvement. This observation aligns with the earlier hypothesis that the capacity of the model may still be a limiting factor, even with the increased number of parameters.

V. CONCLUSION

This research demonstrates the significant benefits of transfer learning for link prediction using GNNs. By pre-training models on the arXiv w/o CS dataset and fine-tuning on the arXiv CS target domain, we achieved notable improvements in training efficiency and initial performance. Specifically, fine-tuned models exhibited a jumpstart improvement of up to 0.63 MRR (GCN), 0.47 (GraphSAGE), and 0.48 (GIN) compared

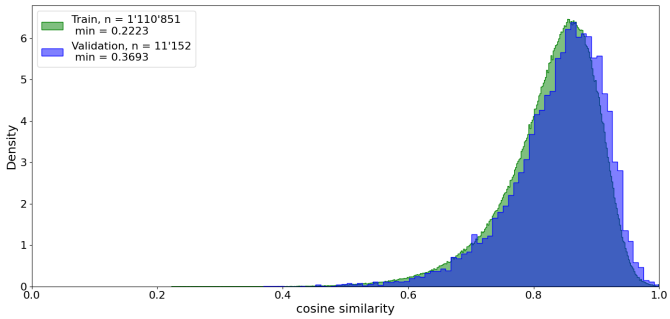

Fig. 6. Distribution of cosine similarities in the arXiv CS dataset differentiating between validation and train nodes, some validation nodes may intersect with the trainset.

to reference models trained from scratch. Notably, the fine-tuned models reached competitive MRR values 15 times faster than reference models, highlighting the practical advantage of transfer learning in resource-constrained scenarios.

Our experiments revealed that transfer learning is effective across diverse GNN architectures (GCN, GIN, GraphSAGE), underscoring its robustness. While asymptotic performance did not improve, the accelerated convergence and strong initial gains emphasise the value of pre-training in scenarios with limited data. Furthermore, increasing model capacity for GraphSAGE enhanced fine-tuning performance, though it confirmed the existence of a performance ceiling, suggesting that architectural choices and pre-training strategies play critical roles in optimising transferability.

Future work could explore architectures such as Graph Attention Networks [24] and methods like Cluster-GCN, which enable scaling to larger models without a proportional increase in GPU memory requirements [25]. Additionally, following [10], researchers could investigate the impact of synthetic graphs in transfer learning for link prediction. Model performance could also benefit from strategic negative sampling, where carefully selecting negative examples may lead to improved optimisation [26]. Finally, the observed discrepancy between lower training MRR and higher validation/test MRR warrants further investigation.

REFERENCES

[1] B. Wittmann, J. C. Paetzold, C. Prabhakar, D. Rueckert, and B. Menze, "Link Prediction for Flow-Driven Spatial Networks," Jan. 2024, issue: arXiv:2303.14501 arXiv:2303.14501 [cs]. [Online]. Available: http://arxiv.org/abs/2303.14501

[2] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in *Proceedings of the twelfth international conference on Information and knowledge management*. New Orleans LA USA: ACM, Nov. 2003, pp. 556–559. [Online]. Available: https://dl.acm.org/doi/10.1145/956863.956972

[3] M. Zhang and Y. Chen, "Link Prediction Based on Graph Neural Networks," 2018, publisher: arXiv Version Number: 3. [Online]. Available: https://arxiv.org/abs/1802.09691

[4] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," Feb. 2017, issue: arXiv:1609.02907 arXiv:1609.02907 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1609.02907

[5] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," Sep. 2018, issue: arXiv:1706.02216 arXiv:1706.02216 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1706.02216

[6] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful are Graph Neural Networks?" Feb. 2019, issue: arXiv:1810.00826 arXiv:1810.00826 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1810.00826

[7] Q. Yang, Y. Zhang, W. Dai, and S. J. Pan, *Transfer Learning*, 1st ed. Cambridge University Press, Jan. 2020. [Online]. Available: https://www.cambridge.org/core/product/identifier/9781139061773/type/book

[8] M. Iman, H. R. Arabnia, and K. Rasheed, "A Review of Deep Transfer Learning and Recent Advancements," *Technologies*, vol. 11, no. 2, p. 40, Mar. 2023, number: 2. [Online]. Available: https://www.mdpi.com/2227-7080/11/2/40

[9] B. Neyshabur, H. Sedghi, and C. Zhang, "What is being transferred in transfer learning?" in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 512–523. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/0607f4c705595b911a4f3e7a127b44e0-Abstract.html

[10] N. Kooverjee, S. James, and T. Van Zyl, "Investigating Transfer Learning in Graph Neural Networks," *Electronics*, vol. 11, no. 8, p. 1202, Apr. 2022, number: 8. [Online]. Available: https://www.mdpi.com/2079-9292/11/8/1202

[11] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A Comprehensive Survey on Transfer Learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, number: 1. [Online]. Available: https://ieeexplore.ieee.org/document/9134370/

[12] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, Jan. 2021, number: 1 arXiv:1901.00596 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1901.00596

[13] Q. Zhu, C. Yang, Y. Xu, H. Wang, C. Zhang, and J. Han, "Transfer Learning of Graph Neural Networks with Ego-graph Information Maximization," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 1766–1779. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/0dd6049f5fa537d41753be6d37859430-Paper.pdf

[14] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec, "Strategies for Pre-training Graph Neural Networks," Feb. 2020, issue: arXiv:1905.12265 arXiv:1905.12265 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1905.12265

[15] Z. Hu, C. Fan, T. Chen, K.-W. Chang, and Y. Sun, "Pre-Training Graph Neural Networks for Generic Structural Feature Extraction," May 2019, issue: arXiv:1905.13728 arXiv:1905.13728 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1905.13728

[16] Z. Hu, Y. Dong, K. Wang, K.-W. Chang, and Y. Sun, "GPT-GNN: Generative Pre-Training of Graph Neural Networks," Jun. 2020, issue: arXiv:2006.15437 arXiv:2006.15437 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2006.15437

[17] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open Graph Benchmark: Datasets for Machine Learning on Graphs," Feb. 2021, issue: arXiv:2005.00687 arXiv:2005.00687 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2005.00687

[18] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," Oct. 2013, issue: arXiv:1310.4546 arXiv:1310.4546 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1310.4546

[19] S. Job, X. Tao, T. Cai, L. Li, H. Xie, and J. Yong, "Towards Causal Classification: A Comprehensive Study on Graph Neural Networks," Jan. 2024, arXiv:2401.15444 [cs] version: 1. [Online]. Available: http://arxiv.org/abs/2401.15444

[20] J. You, Z. Ying, and J. Leskovec, "Design Space for Graph Neural Networks," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 17 009–17 021. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/c5c3d4fe6b2cc463c7d7ecba17cc9de7-Abstract.html

[21] J. Oh, K. Cho, and J. Bruna, "Advancing GraphSAGE with A Data-Driven Node Sampling," Apr. 2019, arXiv:1904.12935 [cs]. [Online]. Available: http://arxiv.org/abs/1904.12935

[22] M. E. Taylor and P. Stone, "Transfer Learning for Reinforcement Learning Domains: A Survey," vol. Journal of Machine Learning Research, no. 10, pp. 1633–1685, Jul. 2009, number: 10.

[23] J. Li, H. Shomer, H. Mao, S. Zeng, Y. Ma, N. Shah, J. Tang, and D. Yin, "Evaluating Graph Neural Networks for Link Prediction: Current Pitfalls and New Benchmarking," Nov. 2023, issue: arXiv:2306.10453 arXiv:2306.10453 [cs]. [Online]. Available: http://arxiv.org/abs/2306.10453

[24] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," Feb. 2018, issue: arXiv:1710.10903 arXiv:1710.10903 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1710.10903

[25] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Jul. 2019, pp. 257–266, arXiv:1905.07953 [cs]. [Online]. Available: http://arxiv.org/abs/1905.07953

[26] Z. Yang, M. Ding, T. Huang, Y. Cen, J. Song, B. Xu, Y. Dong, and J. Tang, "Does Negative Sampling Matter? A Review with Insights into its Theory and Applications," Feb. 2024, arXiv:2402.17238 [cs] version: 1. [Online]. Available: http://arxiv.org/abs/2402.17238