Is Expressivity Essential for the Predictive Performance of Graph Neural Networks?

Fabian Jogl
TU Wien

Pascal Welke
TU Wien

Thomas Gärtner
TU Wien

{firstname.lastname}@tuwien.ac.at

Abstract

Motivated by the large amount of research on the expressivity of GNNs, we study the impact of expressivity on the predictive performance of GNNs. By performing knowledge distillation from highly expressive teacher GNNs to less expressive student GNNs, we demonstrate that knowledge distillation reduces the predictive performance gap between teachers and students significantly. As knowledge distillation does not increase the expressivity of the student GNN, it follows that most of this gap in predictive performance cannot be due to expressivity.

1 Introduction

We investigate whether expressivity *causes* more expressive graph neural networks (GNNs) to achieve better predictive performance than less expressive architectures. Expressivity, as a measure of the amount of non-isomorphic graphs that can be distinguished, is necessary to achieve different predictions for different graphs. It is known that common GNN architectures have limited expressivity: Consider the graphs in Figure 1, message passing graph neural networks (MPNNs)—the most common GNN—are known to be unable to distinguish these graphs [Xu et al., 2019, Morris et al., 2019]. It follows that MPNNs are unable to learn any function that has different outputs for these graphs. This negative result is independent from the amount of training data and the actual number of weights. Recently, a lot of effort has been devoted towards building *higher-order* GNNs, i.e., GNNs that are *provably* more expressive than MPNNs. While many papers [Morris et al., 2019, Bevilacqua et al., 2021, Bodnar et al., 2021, Morris et al., 2020, Zhao et al., 2022, Bouritsas et al., 2022, Feng et al., 2022, Barceló et al., 2021, Qian et al., 2022, Morris et al., 2022, Frasca et al., 2022, Vignac et al., 2020, Zhang et al., 2024a] provide higher-order architectures which empirically surpass MPNNs, it is not clear whether this improvement in predictive performance is actually due to expressivity or other inductive biases. Indeed, in a recent position paper, Morris et al. [2024a] have argued that the influence of GNN expressivity on generalization ability of GNNs is still underexplored.

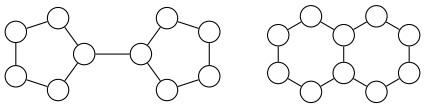

Figure 1: Two graphs that cannot be distinguished by MPNNs.

In this work, we perform knowledge distillation [Buciluǎ et al., 2006] from higher-order *teacher* GNNs to less-expressive *student* MPNNs. We show that knowledge distillation can close the predictive performance gap between teacher GNNs and student MPNNs. This improvement shows that most of the initial gap in predictive performance without distillation is not due to expressivity.

Workshop on Scientific Methods for Understanding Deep Learning, NeurIPS 2024.

Table 1: All GNNs used in this paper.

Model	Type	$>$ MPNN	Citation
GIN	Message Passing (MPNN)	✗	Xu et al. [2019]
GSN	MPNN + Counting Subgraphs	✓	Bouritsas et al. [2022]
CWN	Topological GNN	✓	Bodnar et al. [2021]
DSS	Subgraph GNN	✓	Bevilacqua et al. [2021]
L2GNN	Local 2-GNN	✓	Morris et al. [2020]

2 Related Work

Expressivity. Graphs are defined on unordered sets of nodes and defining a canonical ordering seems computationally hard [Babai, 2016]. This means that isomorphic graphs can have different representations. GNNs are designed to be permutation invariant which means that they compute the same prediction for all isomorphic graphs. It follows, that if a GNN computes different embeddings for two graphs, these two graphs cannot be isomorphic. The ability of a GNN to distinguish pairs of graphs is referred to as *expressivity* and is measured by comparing the sets of graph pairs that can be distinguished by different GNN types. For MPNNs — the most common type of GNN — Xu et al. [2019] and Morris et al. [2019] have shown that their expressivity is limited by the Weisfeiler-Leman graph isomorphism test (WL) [Weisfeiler and Leman, 1968]. Note, that this limitation persists even if the MPNN has an infinite number of layers, weights, and training data. As a result, MPNNs are unable to distinguish simple graphs (see Figure 1) or compute properties that are important to molecular predictions such as cycle counts [Chen et al., 2020]. To improve predictive performance — mainly for molecular tasks — researchers have developed more expressive GNNs and demonstrated that they outperform MPNNs. Of particular relevance in the community are the GNNs we investigate in this paper. We list them in Table 1.

Knowledge Distillation. Knowledge distillation was introduced by Buciluǎ et al. [2006] to compress a larger model by training a smaller model on artificially generated data labeled by the larger model. We refer to the survey of Gou et al. [2021] for general information about knowledge distillation and to Tian et al. [2023] for knowledge distillation on GNNs. We could not find any previous work that uses knowledge distillation to analyze the impact of expressivity on predictive performance of GNNs.

3 Methodology

We perform knowledge distillation from a teacher GNN_T to a student GNN_S. We use two methods for knowledge distillation: (1) we add an additional loss term such that the student learns similar graph embeddings as the teacher and (2) we extend the original training dataset with generated graphs. Note that knowledge distillation only affects the training and leaves the evaluation completely untouched. This ensures that we have no data contamination caused by leaking information about the test set.

Layer alignment. We use the pooled embedding generated by each layer of GNN_T to align the learned layers of the student with the teacher. Let G be a graph from with label y and $\hat{y}$ be the label predicted by the student. We denote the global pooling operation of a GNN as $\bigoplus$ and the pooled graph embedding produced in layer i as $\bigoplus \text{GNN}^{(i)}(G)$. Figure 2 shows the process of knowledge distillation for a single training graph. For a teacher GNN_T with $L \geq 1$ layers we construct a student with $k \cdot L + l$ layers where $k \geq 1$ and $l \geq 0$ are hyperparameters. We perform knowledge distillation by training the student with a special loss function

$$\mathcal{L}(G, y, \hat{y}) = \mathcal{L}_\text{pred}(y, \hat{y}) + \alpha \cdot \mathcal{L}_\text{emb}(\text{GNN}_\text{T}(G), \text{GNN}_\text{S}(G)). \tag{1}$$

Here, $\mathcal{L}_\text{pred}$ could be any loss function; for our experiments we chose the same loss function that was used to train GNN_T. We call $\mathcal{L}_\text{pred}(y, \hat{y})$ the *prediction loss* and $\mathcal{L}_\text{emb}(\text{GNN}_\text{T}(G), \text{GNN}_\text{S}(G))$ the

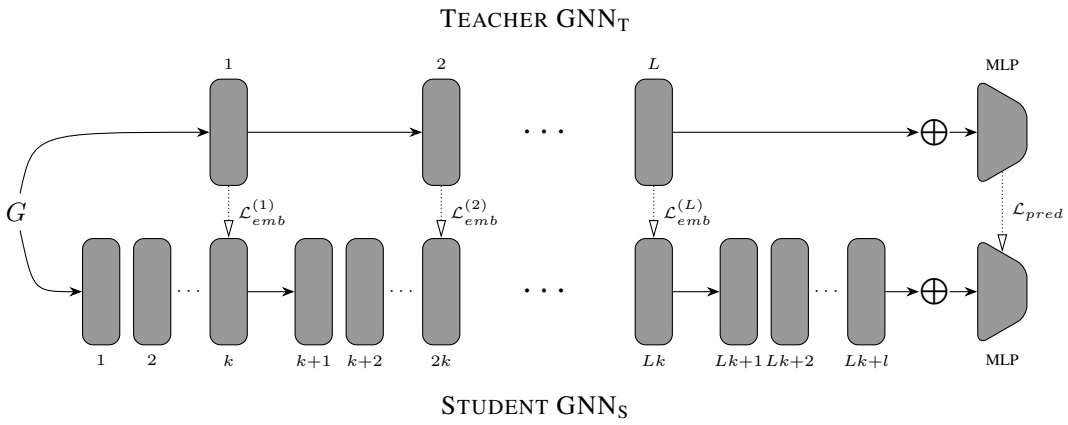

Figure 2: Illustration of knowledge distillation from a teacher GNN to a student GNN.

embedding loss which we define as the squared Euclidean distance between the pooled embeddings

$$\mathcal{L}_{\text{emb}}(\text{GNN}_\text{T}(G), \text{GNN}_\text{S}(G)) = \sum_{\ell=1}^{L} \mathcal{L}_{\text{emb}}^{(i)} = \sum_{\ell=1}^{L} \left\| \bigoplus \text{GNN}_\text{T}^{(\ell)}(G) - \bigoplus \text{GNN}_\text{S}^{(k\ell)}(G) \right\|_2^2 .$$

The constant $\alpha \geq 0$ in Equation (1) makes it possible to tune the importance of the embedding loss relative to the prediction loss. We chose $\alpha = 10$. For classification tasks, we additionally change $\mathcal{L}_{\text{pred}}$ by having the student predict the "soft" logits of the teacher instead of the "hard" training set labels. For this, we generate the labels y with the teacher and use L_1 as a loss function.

Extending Training Set. In some cases, adapting the loss function is not sufficient to significantly improve the predictive performance of the student. In this case, we augment the training set by generating new graphs and labeling them with the teacher. In total, we generate m times as many graphs as in the original training set and say we augment the dataset with a factor of m. To generate graphs, we simply modify graphs in the training set by randomly changing features, adding edges, and dropping edges (more information in Appendix A.2).

Relation to Expressivity. MPNNs are limited in expressivity by WL [Xu et al., 2019, Morris et al., 2019]. This limitation persists even when using knowledge distillation, regardless of the model's size or parameters. As a result, knowledge distillation cannot enhance the expressivity of a student model beyond the WL limit. If a teacher model is more expressive than WL, the student will remain less expressive, even after distillation. Therefore, if knowledge distillation greatly reduces the predictive performance gap between student and teacher it follows that this gap *cannot* be due to expressivity as knowledge distillation does not extend expressivity beyond WL.

4 Experiments

We focus our experiments on the two most commonly used molecular datasets in which an increase in expressivity usually leads to better predictive performance. While we would have preferred to perform experiments on more datasets, it is difficult to find datasets with a significant gap between more elaborate models and baselines where this gap persists even after elaborate tuning of the baselines (see for example Tönshoff et al. [2023]). We perform knowledge distillation from four different teacher GNNs (see Table 1) to one student MPNN GIN [Xu et al., 2019] on three datasets: `ZINC` (12k graphs) [Gómez-Bombarelli et al., 2018, Sterling and Irwin, 2015] and `MOLHIV` (41k graphs) [Hu et al., 2020]. We perform knowledge distillation for all combinations of datasets and teachers as long as the teacher outperforms the student (without knowledge distillation) on the dataset. For all three datasets, we use $\mathcal{L}_{\text{emb}}$ as described in Section 3. For `MOLHIV` we use the soft targets from the teacher (all other datasets are regression datasets) and do not augment the training set as this is not necessary for the student to achieve teacher performance. For the `ZINC` dataset we also experiment with increasing the training set size up to an augmentation factor m of 220 for `ZINC`.

Table 2: Results on ZINC ($m = 220$).

Teacher	Student Performance	Teacher Performance
-	0.187 ± 0.005	-
CWN	$\mathbf{0.143 \pm 0.004}$	0.13
DSS	$\mathbf{0.123 \pm 0.005}$	0.094
GSN	$\mathbf{0.116 \pm 0.001}$	0.1
L2GNN	$\mathbf{0.12 \pm 0.01}$	0.07

Table 3: Results on MOLHIV.

Teacher	Student Performance	Teacher Performance
-	77.9 ± 1	-
CWN	$\mathbf{79.6 \pm 0.4}$	80.1
L2GNN	$\mathbf{79.0 \pm 0.2}$	78.6
GSN	$\mathbf{78.8 \pm 0.6}$	80

Our teachers are trained with standard hyperparameters that are common to each dataset, for more details about the teachers see Appendix A.1. Students on ZINC are designed to have 23 message passing layers, this is significantly more than common but we have found this to work well in preliminary experiments. We validate our design decisions on ZINC in an ablation study where we ablate both the impact of a larger model and of layer alignment via the special loss term (see Section 3). On MOLHIV, our students simply use the same number of layers and embedding dimensions as the teachers. However, preliminary experiments showed that for students to achieve a good performance on MOLHIV, they need a variable learning rate (in contrast to the commonly used constant learning rate) and that the validation set is unreliable for selecting a good model. Thus, for MOLHIV we use a Cosine learning rate scheduler and use the model after the last epoch as the final model.

Results. Overall, knowledge distillation gives a consistent boost in performance to students for all datasets. Interestingly, results vary for different datasets but are mostly uniform across different models. For MOLHIV, Table 3 shows that all students get a significant boost in performance and obtain results that are similar to the teachers. For ZINC, Table 2 shows that students achieve big boosts in performance but also that there is still a small gap in performance between teachers and students. Figure 3 shows the relation between student performance and dataset augmentation (figures for the other models can be found in Appendix A.3). We can see that initially, the performance clearly improves when the augmentation factor is increased. However, this improvement soon levels off which indicates that this is the limit of the distillation procedure. Finally, Table 4 shows an ablation study. First, we can see that data augmentation already leads to an improvement above the baseline (5 layers, no distillation) that is further improved when we use layer alignment. Secondly, we can see that without knowledge distillation an MPNN with few layers outperforms an MPNN with a large amount of layers. Interestingly the opposite is the case with distillation which means that such a large MPNN cannot be trained without distillation. For more information see about the ablations see Appendix A.3.

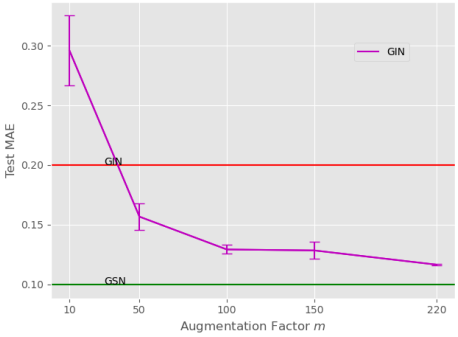

Figure 3: Results on ZINC with GSN as a teacher for different augmentation factors m.

5 Conclusion

We introduced knowledge distillation as a tool to study the impact of expressivity on the predictive performance of GNNs. Our experiments indicate that most of the gap between higher order GNNs and MPNNs is *not* due to expressivity: Knowledge distillation cannot increase the expressivity of the student. However, using distillation, we reduced the gap between less expressive students

Table 4: Ablations on ZINC with DSS as teacher, for models trained with augmentation we use $m = 100$.

Layers	Knowledge Distillation	Test MAE
5	None	0.187 ± 0.005
23	None	0.282 ± 0.009
23	Augmentation only	0.162 ± 0.008
5	Alignment&Augmentation	0.171 ± 0.004
23	Alignment&Augmentation	0.141 ± 0.008

and more expressive teachers. In some cases, we even closed it. This implies that the majority of the gap in predictive performance cannot be due to expressivity. We leave an investigation for the true cause of this gap as open work.

Acknowledgements

FJ is funded by the Center for Artificial Intelligence and Machine Learning (CAIML) at TU Wien. PW is supported by the Vienna Science and Technology Fund (WWTF) through project ICT22-059. TG acknowledges support by the Austrian Science Fund (FWF) through project NanOX-ML (6728) and by TU Wien DK SecInt.

References

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

Christopher Morris, Martin Ritzert, Matthias Fey, William Hamilton, Jan Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *AAAI*, 2019.

Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael Bronstein, and Haggai Maron. Equivariant subgraph aggregation networks. In *ICLR*, 2021.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yu Guang Wang, Pietro Liò, Guido Montúfar, and Michael Bronstein. Weisfeiler and Lehman go cellular: CW networks. In *NeurIPS*, 2021.

Christopher Morris, Gaurav Rattan, and Petra Mutzel. Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings. In *NeurIPS*, 2020.

Lingxiao Zhao, Wei Jin, Leman Akoglu, and Neil Shah. From stars to subgraphs: Uplifting any GNN with local structure awareness. In *ICLR*, 2022.

Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *Transactions on Pattern Analysis and Machine Intelligence*, 45(1):657–668, 2022.

Jiarui Feng, Yixin Chen, Fuhai Li, Anindya Sarkar, and Muhan Zhang. How Powerful are K-hop Message Passing Graph Neural Networks. In *NeurIPS*, 2022.

Pablo Barceló, Floris Geerts, Juan L. Reutter, and Maksimilian Ryschkov. Graph neural networks with local graph parameters. In *NeurIPS*, 2021.

Chendi Qian, Gaurav Rattan, Floris Geerts, Christopher Morris, and Mathias Niepert. Ordered Subgraph Aggregation Networks. In *NeurIPS*, 2022.

Christopher Morris, Gaurav Rattan, Sandra Kiefer, and Siamak Ravanbakhsh. SpeqNets: Sparsity-aware Permutation-equivariant Graph Networks. In *ICML*, 2022.

Fabrizio Frasca, Beatrice Bevilacqua, Michael M. Bronstein, and Haggai Maron. Understanding and Extending Subgraph GNNs by Rethinking their Symmetries. In *NeurIPS*, 2022.

Clément Vignac, Andreas Loukas, and Pascal Frossard. Building powerful and equivariant graph neural networks with structural message-passing. In *NeurIPS*, 2020.

Bohang Zhang, Jingchu Gai, Yiheng Du, Qiwei Ye, Di He, and Liwei Wang. Beyond weisfeiler-lehman: A quantitative framework for gnn expressiveness. In *ICLR*, 2024a.

Christopher Morris, Fabrizio Frasca, Nadav Dym, Haggai Maron, Ismail Ilkan Ceylan, Ron Levie, Derek Lim, Michael M. Bronstein, Martin Grohe, and Stefanie Jegelka. Position: Future directions in the theory of graph machine learning. In *ICML*, 2024a.

Cristian Buciluă, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *KDD*, 2006.

László Babai. Graph isomorphism in quasipolynomial time. In *STOC*, 2016.

Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *Nauchno-Technicheskaya Informatsia*, 9, 1968.

Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. Can graph neural networks count substructures? In *NeurIPS*, 2020.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge articledistillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

Yijun Tian, Shichao Pei, Xiangliang Zhang, Chuxu Zhang, and N. Chawla. Knowledge distillation on graphs: A survey. In *ArXiv abs/2302.00219*, 2023.

Jan Tönshoff, Martin Ritzert, Eran Rosenbluth, and Martin Grohe. Where did the gap go? reassessing the long-range graph benchmark. In *Learning on Graphs Conference*, 2023.

Rafael Gómez-Bombarelli, Jennifer N. Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268–276, 2018.

Teague Sterling and John J. Irwin. Zinc 15 – ligand discovery for everyone. *Journal of Chemical Information and Modeling*, 55(11):2324–2337, 2015.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open Graph Benchmark: Datasets for machine learning on graphs. In *NeurIPS*, 2020.

Fabian Jogl, Maximilian Thiessen, and Thomas Gärtner. Expressivity-preserving GNN simulation. In *NeurIPS*, 2023.

Bohang Zhang, Jingchu Gai, Yiheng Du, Qiwei Ye, Di He, and Liwei Wang. Beyond weisfeiler-lehman: A quantitative framework for gnn expressiveness. In *ICLR*, 2024b.

Franka Bause, Fabian Jogl, Pascal Welke, and Maximilian Thiessen. Maximally expressive GNNs for outerplanar graphs. In *The Second Learning on Graphs Conference*, 2023.

Radoslav Dimitrov, Zeyang Zhao, Ralph Abboud, and İsmail İlkan Ceylan. Plane: Representation learning over planar graphs. In *NeurIPS*, 2023.

Caterina Graziani, Tamara Drucks, Fabian Jogl, Monica Bianchini, Franco Scarselli, and Thomas Gärtner. The expressive power of path-based graph neural networks. In *ICML*, 2024.

Raffaele Paolino, Sohir Maskey, Pascal Welke, and Gitta Kutyniok. Weisfeiler and leman go loopy: A new hierarchy for graph representational learning. In *NeurIPS*, 2024.

Gaspard Michel, Giannis Nikolentzos, Johannes F. Lutzeyer, and Michalis Vazirgiannis. Path neural networks: Expressive and accurate graph neural networks. In *ICML*, 2023.

Luis Müller and Christopher Morris. Aligning transformers with weisfeiler-leman. In *ICML*, 2024.

Markus Zopf. 1-wl expressiveness is (almost) all you need. In *IJCNN*, 2022.

Christopher Morris, Fabrizio Frasca, Nadav Dym, Haggai Maron, İsmail İlkan Ceylan, Ron Levie, Derek Lim, Michael Bronstein, Martin Grohe, and Stefanie Jegelka. Future directions in the theory of graph machine learning. In *ICML*, 2024b.

Table 5: Hyperparameters on ZINC. * marks that the student hyperparameter is dependent on the teacher.

	Student	GSN	CWN	DSS	L2GNN
Num. layers	23	5	5	5	5
Emb. dim.	*	256	256	256	256
Pooling	*	sum	mean	mean	sum
Dropout	0	0	0.5	0.5	0
Batch size	512	128	128	128	128
Epochs	1000	1000	1000	1000	1000
LR	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}
LR scheduling	Reduce On Plateau	Reduce On Plateau	Reduce On Plateau	Reduce On Plateau	Reduce On Plateau
Min. LR	$2.5 \cdot 10^{-4}$	10^{-5}	10^{-5}	10^{-5}	10^{-5}

Table 6: Hyperparameters on MOLHIV. * marks that the student hyperparameter is dependent on the teacher.

	Student	GSN	CWN	L2GNN
Num. layers	*	5	4	5
Emb. dim.	*	64	64	32
Pooling	*	sum	mean	sum
Dropout	0	0.5	0.5	0.5
Batch size	128	32	32	32
Epochs	100	100	100	100
LR	10^{-3}	10^{-3}	10^{-3}	10^{-3}
LR scheduling	Cosine	-	-	-

A Appendix

Our code can be found at https://github.com/ocatias/KnowledgeDistillationGNNs_SciForDL.

A.1 Details about Teachers

We use four different teacher models: GSN [Bouritsas et al., 2022], CWN [Bodnar et al., 2021], DSS [Bevilacqua et al., 2021] and L2GNN [Morris et al., 2020]. Our implementation of GSN, is a simple GIN model together with subgraph counts attached to the node features. For this, we count the number of cycles of length up to 9 in each graph and for each $\ell \in \{3, \dots, 9\}$ we attach to each node the number of ℓ cycles that this node is a part of. For CWN, we use a lifting map the lifts all cycles of length up to ℓ to cells and performs message passing on the resulting cell complex. Our implementation is based on a simulation i.e. a graph transformation together with an MPNN [Jogl et al., 2023]. DSS is a subgraph GNN and uses a policy that extracts all 3-hop subgraphs. Finally, L2GNN is a local 2-GNN [Morris et al., 2020] for which we use an implementation by Zhang et al. [2024b].

Hyperparameters The hyperparameters are shown in Tables 5 and 6. Note that when we perform knowledge distillation we adapt the hyperparameters to speed up the experiments (by increasing the batch size and for "Reduce on Plateau" scheduling increasing the minimum learning rate).

A.2 Data Augmentation

On ZINC we evaluate the effectiveness of data augmentations for knowledge distillation. We chose a simple procedure to generate new graphs that is based on modifying graphs in the dataset. For a given graph with edges E, our procedure begins by iterating through all edges and dropping each edge from E with a probability of 0.05. Then, we randomly add edges to the graph, in total we add $E \cdot 0.05$ edges. For each newly added edge, we generate its edge features by dimension-wise

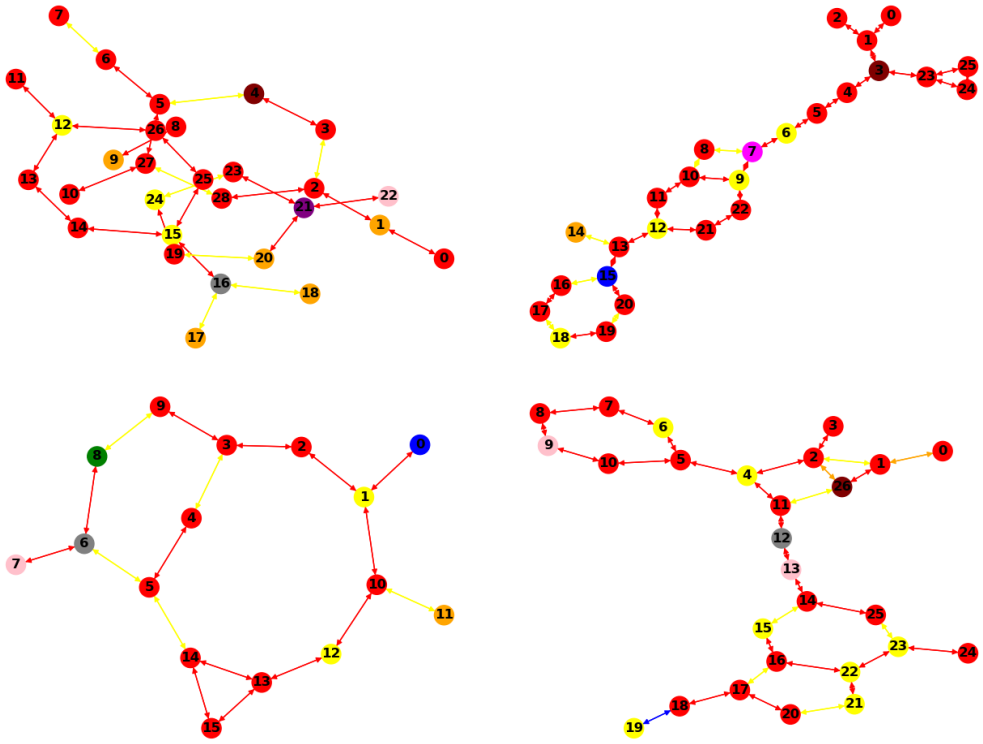

Figure 4: Graphs generated by our method based on the ZINC dataset.

randomly drawing features from the marginal edge feature distribution of the training set. Finally, we change each node-feature dimension-wise with a probability of 0.05[1]. Similarly to edge features, whenever a dimension of a node feature gets changed we randomly sample from the marginal feature distribution on the training set conditioned on the node degree. Figure 4 shows a graph generated in this way.

A.3 Additional Experimental Details and Results

Figure 5 shows how different teachers on ZINC scale with the data augmentation factor. In the ablation study we compare the following models, the results can be found in Table 4.

1. GIN with 5 message passing layers and no knowledge distillation.
2. GIN with 23 message passing layers and no knowledge distillation.
3. GIN with 23 message passing layers and data augmentation only ($m = 100$). That means the model is only trained to predict the output of the teacher and no additional loss term is used.
4. GIN with 5 message passing layers data augmentation ($m = 100$) and layer alignment i.e., the additional loss term from 3..
5. Same as above but with 23 message passing layers.

The idea behind this is that (1) and (2) check whether our improvements on ZINC are only due to the fact that we use a larger model as a student. (3) and (5) checks whether layer alignment and augmentation performs better than augmentation alone. (4) and (5) checks whether a larger model truly performs better for knowledge distillation.

[1]Except for DSS on ZINC where this value is 0.2. This was not intentional but seems to have little impact on the results.

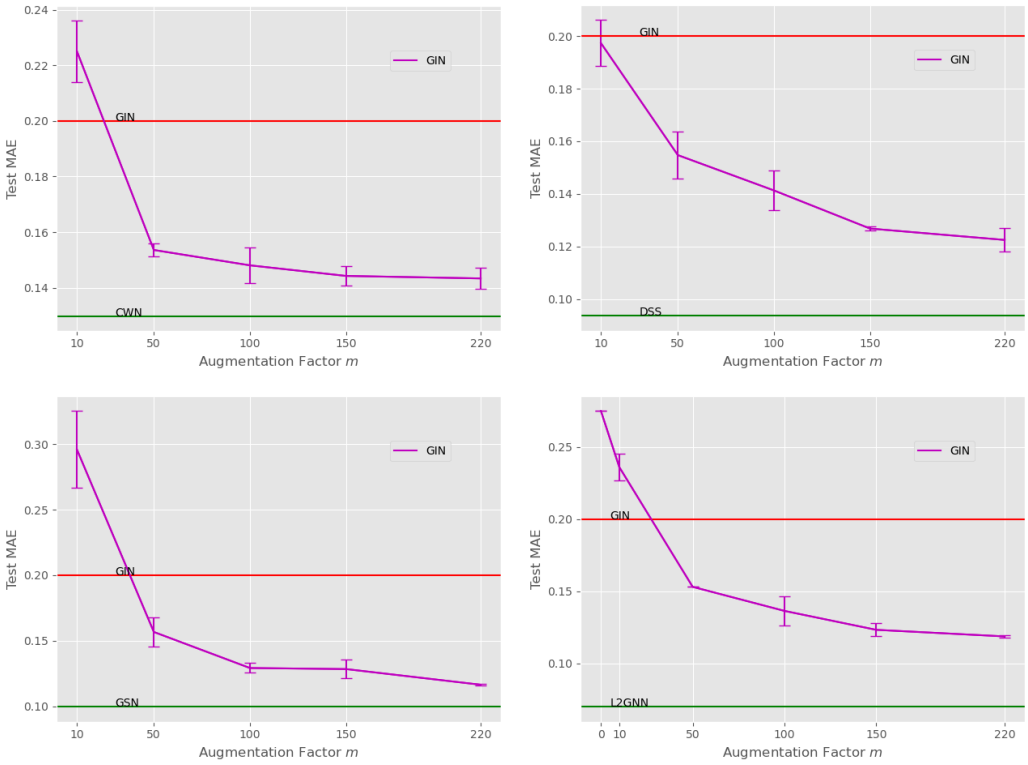

Figure 5: Results on ZINC with for all teachers and different augmentation factors.

B Debunking Challenge Submission

B.1 What commonly-held position or belief are you challenging?

Provide a short summary of the body of work challenged by your results. Good summaries should outline the state of the literature and be reasonable, e.g. the people working in this area will agree with your overview. You can cite sources beside published work (e.g., blogs, talks, etc).

Starting with Xu et al. [2019], Morris et al. [2019] *expressivity*—i.e, the ability to distinguish non-isomorphic graphs—has served as motivation for more and more new GNN architectures. Many GNNs are specifically designed to be sufficiently expressive to solve specific tasks. For instance, cyclic structures are important for molecular tasks leading to GNNs that are designed to perform message passing on cycles [Bodnar et al., 2021], incorporate cycle counts [Bouritsas et al., 2022], or are maximally expressive for graph classes common among molecules [Bause et al., 2023, Dimitrov et al., 2023]. Furthermore, a diverse range of general-purpose GNNs designed primarily to be more expressive has emerged: subgraph-based GNNs [Bevilacqua et al., 2021], path-based GNNs [Graziani et al., 2024, Paolino et al., 2024, Michel et al., 2023], and those utilizing higher-dimensional Weisfeiler-Lehman (WL) variants [Morris et al., 2019, 2020, Müller and Morris, 2024], among others. These highly expressive GNNs often exhibit superior predictive performance compared to their less expressive counterparts. However, at the moment there are no obvious theoretical arguments that show that more expressive GNNs generalize better on real world data, after all most real-world data can be distinguished by less expressive MPNNs [Zopf, 2022]. Despite this, expressivity has served as the primary motivation to develop new GNN architectures and often as the only provided explanation of their improved predictive performance. A recent positional paper by leading GNN researchers [Morris et al., 2024b] identified several open research directions based on developing a more nuanced understanding of expressivity and its link to the generalization capabilities of GNNs. Thus, the community is already trying to shift towards developing an understanding of the impact of expressivity on predictive performance. An important first step in this direction is asking the fundamental question: Is expressivity truly the reason why highly expressive GNNs perform better? We challenge the belief that expressivity is the primary factor behind the strong performance of these models.

B.2 How are your results in tension with this commonly-held position?

Detail how your submission challenges the belief described in (1). You may cite or synthesize results (e.g. figures, derivations, etc) from the main body of your submission and/or the literature.

Our experiments show that knowledge distillation significantly reduces the gap between highly expressive teachers and less expressive students. For `ZINC` this reduction is 57-82% (Table 2) and for `MOLHIV` it is 42-100% (Table 3). Note that these are two of the most prominent datasets used to demonstrate the empirical effectiveness of novel GNNs with high expressivity (see for example Bodnar et al. [2021], Bevilacqua et al. [2021], Morris et al. [2020]). As knowledge distillation reduces the gap in predictive performance between expressive GNN teachers and less expressive GNN students it follows that the majority of this gap is *not* due to expressivity. The reasoning for this is simple: knowledge distillation cannot increase the expressivity of the student but does clearly reduce the gap in performance between such models.

B.3 How do you expect your submission to affect future work?

Perhaps the new understanding you are proposing calls for new experiments or theory in the area, or maybe it casts doubt on a line of research.

We believe that our work will lead to a more nuanced understanding of expressivity and its relevance to practical performance. Our work shows that expressivity is not the key driver behind GNN performance. We hope that knowledge about this fact will shift the community's understanding of expressivity analysis. Ideally, expressivity analysis of novel architectures will in the future be seen as a worst case analysis of the limitations of a given architecture. After all, if a GNN cannot distinguish two graphs, it will not be able to learn any function that requires to handle them separately. We hope that our work motivates researchers to develop novel analysis methods that are well-aligned with predictive performance.