Meta-Aggregator: Learning to Aggregate for 1-bit Graph Neural Networks – *Supplementary Material* –

Yongcheng Jing[1], Yiding Yang[2], Xinchao Wang[3], Mingli Song[4], Dacheng Tao[5,1]
[1]The University of Sydney, Australia, [2]Stevens Institute of Technology,
[3]National University of Singapore, [4]Zhejiang University, [5]JD Explore Academy, China
yjin9495@sydney.edu.au, yyang99@stevens.edu,
xinchao@nus.edu.sg, brooksong@zju.edu.cn, dacheng.tao@gmail.com

We provide in this documents supporting materials that cannot fit into the manuscript due to page limit, including theoretical analysis and more results of the 1-bit graph neural networks (GNNs) with the proposed meta aggregators.

Specifically, we begin by giving the propositions on how the proposed *Adaptable Hybrid Neighborhood Aggregator (ANA)* can approximate various standard aggregators, followed by detailed theoretical proofs for each proposition. We then demonstrate more results on the tasks of graph regression and multi-label node classification, and also provide additional results of point cloud classification models.

1. Theoretical Proof

In this section, we provide the propositions and the corresponding theoretical proofs on how and why the proposed *Adaptable Hybrid Neighborhood Aggregator (ANA)* can approximate various existing aggregation methods, such as max, mean, and variance.

We start by showing the mathematical form of ANA again, based on the *Log-Sum-Exp* function in convex optimization:

$$f(\mathcal{G},\mathcal{X}) = \frac{1}{\mathcal{A}^{\ell}(\mathcal{G})}\log\left[\frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^{\ell}(\mathcal{G})\,\mathcal{X}_j^{\ell}}\right], \tag{1}$$

where $\mathcal{A}^{\ell}$ denotes the 1-bit graph auto-encoder at layer ℓ. $\deg_i$ is the in-degree of the node v_i, and $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$ is the graph sample with edges $(v_i,v_j)\in\mathcal{E}$. $\mathcal{X}_j^{\ell}$ represents the feature vector of the neighboring node v_j at layer ℓ, whereas $f(\mathcal{G},\mathcal{X})$ denotes the obtained diffused aggregator.

Based on Eq. 1, we provide the following propositions and the corresponding proofs:

Proposition 1 (Mean)**.** *ANA, as defined in Eq. 1 as $f(\mathcal{G},\mathcal{X})$, can approximate the mean aggregator when $\mathcal{A}^{\ell}(\mathcal{G})\to 0$.*

Proof. We prove Proposition 1 primarily based on the inequality of arithmetic and geometric, defined as:

$$\frac{1}{n}\sum_{i=1}^{n} x_i \geq \Big(\prod_{i=1}^{n} x_i\Big)^{\frac{1}{n}}, \tag{2}$$

where the equality holds when $x_1=x_2=\cdots=x_n$.
By combining Eq. 1 and Eq. 2, we can derive the following inequation:

$$f(\mathcal{G},\mathcal{X}) = \log\left[\frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^{\ell}(\mathcal{G})\,\mathcal{X}_j^{\ell}}\right]^{\frac{1}{\mathcal{A}^{\ell}(\mathcal{G})}} \geq \log\left[\prod_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^{\ell}(\mathcal{G})\,\mathcal{X}_j^{\ell}}\right]^{\frac{1}{\deg_i \mathcal{A}^{\ell}(\mathcal{G})}} = \log\left[\prod_{(j,i)\in\mathcal{E}} e^{\mathcal{X}_j^{\ell}}\right]^{\frac{1}{\deg_i}}. \tag{3}$$

The equality in Eq. 3 holds when $\mathcal{A}^{\ell}(\mathcal{G})\to 0$, *i.e.*,

$$f(\mathcal{G},\mathcal{X}) = \log\left[\prod_{(j,i)\in\mathcal{E}} e^{\mathcal{X}_j^{\ell}}\right]^{\frac{1}{\deg_i}} = \frac{1}{\deg_i}\log\left[\prod_{(j,i)\in\mathcal{E}} e^{\mathcal{X}_j^{\ell}}\right] = \frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{X}_j^{\ell}}. \tag{4}$$

Eq. 4 is, in fact, the formulation of the mean aggregation. Thus, the proposed AN can approximate the mean aggregator. □

Proposition 2 (Max). *ANA defined in Eq. 1 can approximate the max aggregator when* $\mathcal{A}^\ell(\mathcal{G}) \to \infty$.

Proof. To prove Proposition 2, we begin by reformulating Eq. 1 into:

$$f(\mathcal{G},\mathcal{X}) = \frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log\left[\frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell}\right] = \log\left[\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell}\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}} - \frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log(\deg_i). \tag{5}$$

Meanwhile, in our implementation, we keep $\mathcal{A}^\ell(\mathcal{G}) > 0$ by using an absolute operation. As such, we can also obtain the following inequation:

$$\left[\max_{(j,i)\in\mathcal{E}}(e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell})\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}} \le \left[\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell}\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}} \le \left[\deg_i\cdot\max_{(j,i)\in\mathcal{E}}(e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell})\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}}. \tag{6}$$

By combining Eq. 5 and Eq. 6, we can obtain:

$$\log\left[\max_{(j,i)\in\mathcal{E}}(e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell})\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}} \le \log\left[\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell}\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}} \le \frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log(\deg_i) + \log\left[\max_{(j,i)\in\mathcal{E}}(e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell})\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}}. \tag{7}$$

When $\mathcal{A}^\ell(\mathcal{G}) \to \infty$, we have: $\frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log(\deg_i) \to 0$. As such, by replacing $\frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log(\deg_i)$ with 0 in Eq. 7, we can obtain the following equation:

$$\log\left[\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell}\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}} = \log\left[\max_{(j,i)\in\mathcal{E}}(e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell})\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}}. \tag{8}$$

By combing Eq. 8 and Eq. 5, and meanwhile replacing $\frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log(\deg_i)$ in Eq. 5 with 0, we can derive the following equation:

$$f(\mathcal{G},\mathcal{X}) = \log\left[\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell}\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}} = \log\left[\max_{(j,i)\in\mathcal{E}}(e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell})\right]^{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}}. \tag{9}$$

The right part of Eq. 9 is, in fact, the mathematical form of the max aggregation method, which indicates that the proposed ANA can approximate the max aggregator when $\mathcal{A}^\ell(\mathcal{G}) \to \infty$. □

Proposition 3 (Variance). *The variant of ANA, defined as* $h(\mathcal{G},\mathcal{X})$ *in Eq. 10, can approximate the variance aggregator when* $\mathcal{A}^\ell(\mathcal{G}) \to 0$.

$$h(\mathcal{G},\mathcal{X}) = \frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log\left[\frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,(\mathcal{X}_j^\ell)^2}\right] - \left\{\frac{1}{\mathcal{A}^\ell(\mathcal{G})}\log\left[\frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{A}^\ell(\mathcal{G})\,\mathcal{X}_j^\ell}\right]\right\}^2. \tag{10}$$

Proof. By combining Eq. 1 and Eq. 10, we can obtain: $h(\mathcal{G},\mathcal{X}) = f(\mathcal{G},\mathcal{X}^2) - [f(\mathcal{G},\mathcal{X})]^2$. When $\mathcal{A}^\ell(\mathcal{G}) \to 0$, we have:

$$h(\mathcal{G},\mathcal{X}) = \frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{(\mathcal{X}_j^\ell)^2} - \left[\frac{1}{\deg_i}\sum_{(j,i)\in\mathcal{E}} e^{\mathcal{X}_j^\ell}\right]^2, \tag{11}$$

which is, in fact, the formulation of the variance aggregator. □

The final form of the proposed ANA consists of one term in the form of Eq. 1 and also one term in the form of Eq. 10, with the two corresponding graph auto-encoders $\mathcal{A}^\ell$ as well as two learnable weighting factors that determine the portion of each approximated aggregator. As such, the proposed ANA can potentially approximate various aggregators at the same time, and generate a hybrid behavior of different aggregators by controlling the weighting factors.

2. Additional Results on Graph Regression Task

In this section, we provide additional results on the ZINC dataset for the task of graph regression. Specifically, we conduct extensive ablation studies on ZINC to validate the effectiveness of the proposed method.

Implementation Details. We use the ZINC dataset for the task of graph regression. ZINC is a large-scale molecular dataset, which aims to regress a specific molecular property. The node features in each molecular graph are the types of heavy atoms. The corresponding edge features are the types of bonds between them. For the dataset splittings, we follow the standard splitting protocol in [3]. Specifically, 10,000 molecular graphs in ZINC are used for training, 1,000 graphs are for validation, and the remaining 1,000 ones are used for testing. In training, we use the Adam optimizer [6]. The batch size is set to 128. For the learning rate, we set the initial value as 10^{-3}, which is reduced by half if there is no improvement in the validation loss after 10 epochs. The training process is stopped when the learning rate reaches 10^{-3}. We measure the performance using the mean absolute error (MAE) between the predicted property and the ground-truth one. Detailed network architectures that are used in both the main manuscript and also this supplementary material are summarized in Tab. S1.

Table S1. Detailed network architectures for the task of graph regression on the ZINC dataset, where Architecture-ZINC-Main-GAT and Architecture-ZINC-Main-GCN represent the architectures of the two models shown in Tab. 1 and Tab. 2 of the main manuscript. Architecture-ZINC-Supp denotes the architectures that will be used for ablation studies in this supplement material.

Models	**Layers**	**Attention Heads**	**Hidden**	**Output**
Architecture-ZINC-Main-GAT	6	{8, 8, 8, 8}	18	144
Architecture-ZINC-Main-GCN	6	–	145	145
Architecture-ZINC-Supp-V1	5	{8, 8, 8}	18	144
Architecture-ZINC-Supp-V2	5	{8, 8, 8}	22	176
Architecture-ZINC-Supp-V3	8	{8, 8, 8, 8, 8, 8}	22	176

Ablation Studies. We show in Tab. S2 the regression results of the 1-bit GNNs with different network architectures. Specifically, from left to right, Tab. S2 shows the results of the full-precision GNNs (Full Prec.), those of the 1-bit GNNs without the proposed meta aggregators (Vanilla), and the results of the 1-bit GNNs with GNA and ANA. In the last line of Tab. S2, we also provide the p-value of the paired t-test between the proposed meta aggregator and the vanilla one, so as to demonstrate the statistically meaningful improvements by the proposed GNA and ANA. It is noticeable that both GNA and ANA achieve performance superior to that of the vanilla one that depends on a single fixed and pre-defined aggregator.

Table S2. Results on the ZINC dataset for the task of graph-property regression, in terms of the mean absolute error (MAE). The detailed network architectures of Architecture-ZINC-Supp-V1 and Architecture-ZINC-Supp-V2 are shown in Tab. S1.

Architecture	**Architecture-ZINC-Supp-V1**				**Architecture-ZINC-Supp-V2**			
Methods	Full Prec. [9]	Vanilla [5]	**GNA (Ours)**	**ANA (Ours)**	Full Prec. [9]	Vanilla [5]	**GNA (Ours)**	**ANA (Ours)**
Bit-width	32/32	1/1	**1/1**	**1/1**	32/32	1/1	**1/1**	**1/1**
Param Size	316.691KB	78.0156KB	**78.2811KB**	**78.1226KB**	466.816KB	111.164KB	**111.488KB**	**111.294KB**
Test MAE±SD	0.495±0.008	0.647±0.064	**0.598±0.022**	**0.576±0.031**	0.496±0.006	0.687±0.081	**0.590±0.020**	**0.566±0.015**
Train MAE±SD	0.372±0.017	0.588±0.065	**0.536±0.024**	**0.471±0.035**	0.362±0.013	0.629±0.083	**0.523±0.022**	**0.444±0.024**
***p*-value**	GNA vs. Vanilla: 6.316×10^{-4}/ANA vs. Vanilla: 7.101×10^{-6}				GNA vs. Vanilla: 3.869×10^{-7}/ANA vs. Vanilla: 1.768×10^{-9}			

Furthermore, we provide in Tab. S3 the results of the 32-bit models with the proposed GNA and ANA, corresponding to Tab. 2 of the main manuscript but with a different additional network architecture. The proposed meta aggregators, as shown in Tab. S3, also achieve results superior to the state-of-the-art on the full-precision models.

Table S3. Results of the proposed GNA and ANA as well as other methods for 32-bit full-precision models on the ZINC dataset, in terms of MAE. The detailed network architectures of the proposed methods are shown as Architecture-ZINC-Supp-V3 in Tab. S1. For the architectures of the comparison methods [1, 4, 12, 8, 7, 9], we follow the network architecture designs in [3].

Methods	**Param Size**	**Test MAE±SD**	**Train MAE±SD**	**Methods**	**Param Size**	**Test MAE±SD**	**Train MAE±SD**
GraphSage [4]	1973.99KB	0.398±0.002	0.081±0.009	RingGNN [2]	2059.70KB	0.353±0.019	0.236±0.019
GIN [12]	1990.43KB	0.526±0.051	0.444±0.039	MoNet [8]	1968.80KB	0.292±0.006	0.093±0.014
GCN [7]	1972.96KB	0.367±0.011	0.128±0.019	GAT [9]	2075.57KB	0.384±0.007	0.067±0.004
GNA (Ours)	**858.809KB**	**0.295±0.013**	**0.088±0.016**	**ANA (Ours)**	**846.410KB**	**0.294±0.010**	**0.079±0.018**

3. Additional Results on Multi-label Node Classification Task

In this section, we show more results on the PPI dataset for the task of multi-label node classification. Specifically, except for the architecture mentioned in the main manuscript, we provide here additional results with three newly designed network architectures.

Implementation Details. We use the protein-protein interaction (PPI) dataset for the task of multi-label node classification, which contains biological graphs with the nodes labeled with various protein functions. In particular, each node can concurrently have several labels. In training, the batch size is set to 1. The learning rate is 0.005 for each model. In total, we optimize all the models for 500 epochs and report the corresponding results with the best validation accuracies. The detailed network architectures are demonstrated in Tab. S4. Specifically, the 2nd row of Tab. S4 shows the architecture used in Tab. 3 of the main manuscript. The 3rd, 4th, and 5th rows, on the other hand, correspond to three newly-designed architectures that are used in the following extensive ablation studies.

Table S4. Summary of the detailed network architectures for the task of multi-label node classification on the PPI dataset.

Models	**Layers**	**Attention Heads**	**Hidden**
Architecture-PPI-Main	3	{4, 4, 6}	512
Architecture-PPI-Supp-V1	3	{4, 4, 6}	256
Architecture-PPI-Supp-V2	5	{2, 2, 2, 2, 2}	128
Architecture-PPI-Supp-V3	5	{2, 2, 2, 2, 2}	64

Ablation Studies. We perform here ablation studies on various network architectures for the task of multi-label node classification. The corresponding results are shown in Tab. S5, where the results in the main manuscript are shown again, such that we can observe how the performance changes with varying model sizes. The proposed GNA and ANA, as can be seen from Tab. S5, delivers competitive results as compared with those of the full-precision-based approach across all the four distinct architectures, yet maintaining a compact model size. Also, with a similar lightweight architecture, the 1-bit GNNs with the proposed meta aggregators achieve superior performance to that of the model with a pre-defined aggregator, demonstrating the effectiveness of the proposed learnable aggregation schemes.

Table S5. Results on the PPI dataset for the task of node classification, in terms of micro-averaged F_1 score. Detailed network architectures of Architecture-PPI-Main as well as Architecture-PPI-Supp-V1, V2, and V3 can be found in Tab. S4.

Architecture	**Architecture-PPI-Main**			**Architecture-PPI-Supp-V1**		
Methods	Bit-width	Param Size	F_1 Score	Bit-width	Param Size	F_1 Score
Full Prec. [7]	32/32	43.7712MB	98.70	32/32	13.8884MB	98.67
Vanilla [5]	1/1	28.2560MB	92.68	1/1	10.0058MB	93.27
GNA (Ours)	**1/1**	**28.2572MB**	**97.52**	**1/1**	**10.0064MB**	**96.79**
ANA (Ours)	**1/1**	**28.2565MB**	**97.71**	**1/1**	**10.0060MB**	**97.02**
Architecture	**Architecture-PPI-Supp-V2**			**Architecture-PPI-Supp-V3**		
Methods	Bit-width	Param Size	F_1 Score	Bit-width	Param Size	F_1 Score
Full Prec. [7]	32/32	2.0311MB	98.21	32/32	0.64150MB	94.80
Vanilla [5]	1/1	1.2989MB	48.54	1/1	0.45702MB	45.88
GNA (Ours)	**1/1**	**1.2994MB**	**53.52**	**1/1**	**0.45725MB**	**53.94**
ANA (Ours)	**1/1**	**1.2991MB**	**69.24**	**1/1**	**0.45711MB**	**72.29**

4. Additional Results on 3D Object Recognition Task

We provide in this section more results on the ModelNet40 dataset for the task of 3D object classification. Specifically, we devise two additional architectures and conduct extensive ablation studies accordingly.

Implementation Details. We follow the official dataset splitting protocol in [11, 10]. We set the learning rate as 0.001 and use a batch size of 16. We adopt the Adam optimizer [6] and all the models are optimized for 1000 epochs for full

convergence. The detailed architecture designs are summarized in Tab. S6, where the 2nd row corresponds to the architecture used in Tab. 4 of the main manuscript. The 3rd and 4th rows, on the other hand, demonstrate the architectures that are used in this supplement for extensive ablation studies.

Table S6. Summary of the detailed network architectures for the task of 3D object recognition on ModelNet40.

Models	**Layers**	**Feature Map Channels**	**MLPs**
Architecture-ModelNet40-Main	6	[64, 64, 128, 512]	[256, 40]
Architecture-ModelNet40-Supp-V1	6	[32, 32, 64, 128]	[256, 40]
Architecture-ModelNet40-Supp-V2	8	[64, 64, 128, 256, 1024]	[512, 256, 40]

Ablation Studies. We provide in Tab. S7 the results of different approaches with the two newly-designed architectures. The quantitative results in Tab. S7 indicate that both of the proposed GNA and ANA can boost the performance of 1-bit GNNs by a large margin across various network architectures, as compared with the vanilla fixed aggregation method [5].

Table S7. Results on the ModelNet40 dataset for the task of 3D object recognition, in terms of the overall accuracy (Acc) and the mean class accuracy (mAcc). The details of Architecture-ModelNet40-Supp-V1 and Architecture-ModelNet40-Supp-V2 are shown in Tab. S6.

Architecture	**Architecture-ModelNet40-Supp-V1**				**Architecture-ModelNet40-Supp-V2**			
Methods	Bit-width	Param Size	Acc	mAcc	Bit-width	Param Size	Acc	mAcc
Full Prec. [10]	32/32	388.906KB	92.30%	89.43%	32/32	7068.66KB	93.03%	89.70%
Vanilla [5]	1/1	302.930KB	55.79%	46.28%	1/1	4742.20KB	81.12%	73.88%
GNA (Ours)	**1/1**	**303.023KB**	**60.66%**	**49.74%**	**1/1**	**4742.39KB**	**81.65%**	**75.23%**
ANA (Ours)	**1/1**	**302.962KB**	**74.27%**	**65.96%**	**1/1**	**4742.33KB**	**84.81%**	**78.97%**

More Qualitative Results. We also show in Fig. S1 more qualitative results of different approaches, by visualizing the structures of the learned feature spaces. It can be observed that the proposed GNA and ANA can facilitate the 1-bit GNNs to learn a more similar feature structure to that of the cumbersome full-precision ones.

References

[1] Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017. 3

[2] Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. On the equivalence between graph isomorphism testing and function approximation with gnns. 2019. 3

[3] Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020. 3

[4] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017. 3

[5] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NeurIPS*, 2016. 3, 4, 5

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 3, 4

[7] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017. 3, 4

[8] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *CVPR*, 2017. 3

[9] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018. 3

[10] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 2019. 4, 5

[11] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 4

[12] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019. 3

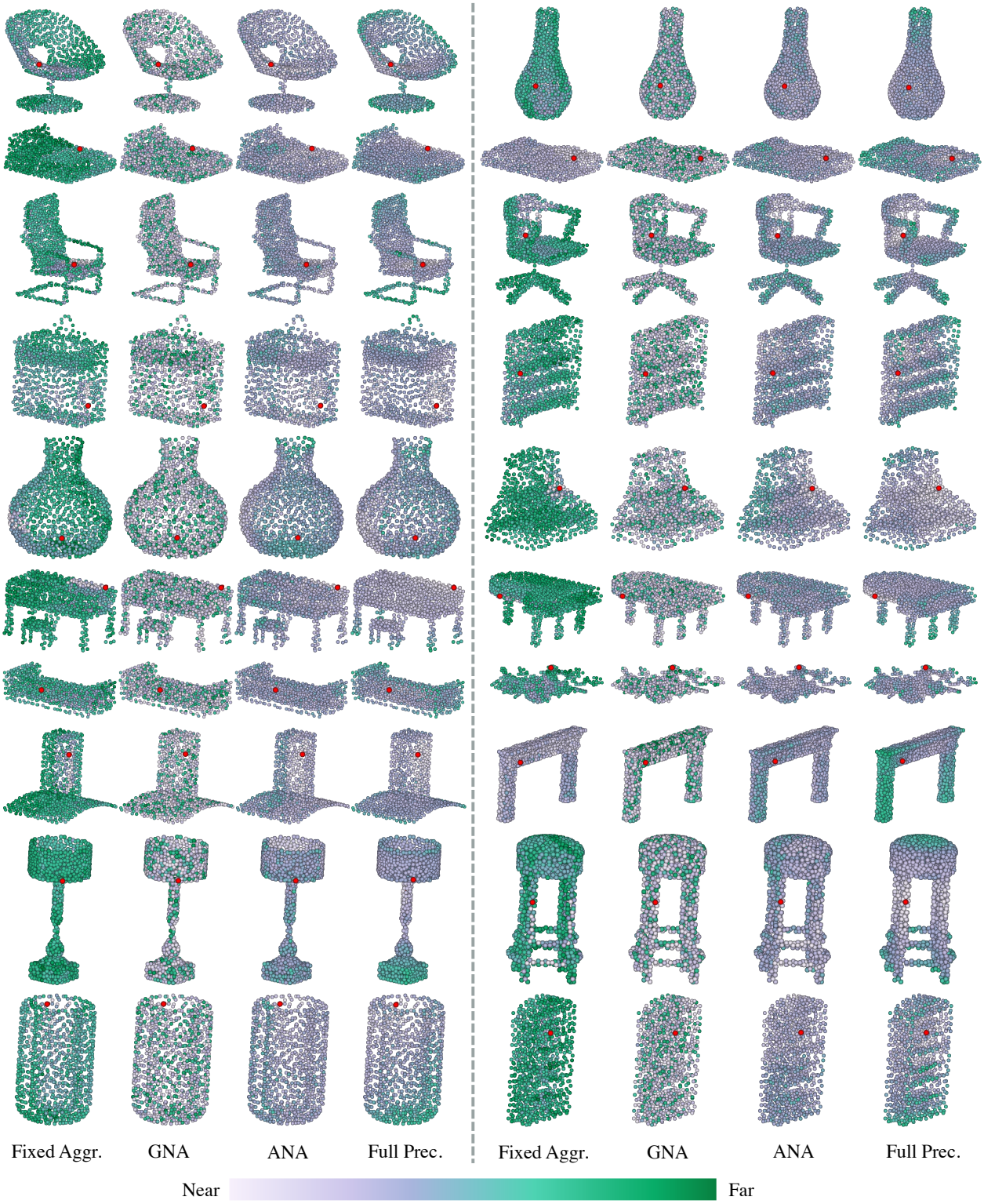

Figure S1. Comparative visualization results. Node color encodes the distance between the red dot and node of interest. All the visualized features are extracted from the intermediate layer of the models.