Supplementary material:
Learning Adaptive Neighborhoods for Graph Neural Networks

Avishkar Saha[1], Oscar Mendez[1], Chris Russell[2], Richard Bowden[1]
[1]Centre for Vision Speech and Signal Processing, University of Surrey, Guildford, UK
[2] University of Oxford, Oxford, UK
{a.saha, o.mendez, r.bowden}@surrey.ac.uk, christopher.m.russell@gmail.com

A. Node classification Experiments

A.1. Datasets

Transductive datasets. We evaluate on the three citation benchmark datasets Cora, Citeseer and Pubmed [26] introduced by [41]. Each citation graph contains nodes corresponding to documents and edges indicating citations. Node features are bag-of-words features and node labels are categorized by topic. We follow the semi-supervised setting in [41] and [16] along with their train/test splits. **Inductive datasets.** In an inductive setting, we evaluate our approach on three datasets: 1. Flickr [42] — categorizing images based on their descriptions; 2. Reddit [42] — predicting the communities of online posts from user comments. 3. PPI [11] — classifying protein-protein interactions.

A.2. Dataset details

In Table 1 we provide dataset statistics on the node classification datasets used in this paper.

Table 1. Dataset statistics. ‘s’ stands for single class classification and ‘m’ for multi-class.

Dataset	Nodes	Edges	Features	Classes	Train / Val / Test
Cora	2,708	5,429	1,433	7 (s)	140 / 500 / 1,000
Citeseer	3,327	4,732	3,703	6 (s)	120 / 500 / 1,000
PubMed	19,717	44,338	500	3 (s)	100 / 500 / 1,000
PPI	56,944	81,8716	50	121 (m)	44,906 / 6,514 / 5,524
Flickr	89,250	899,756	500	7 (s)	44,760 / 22,312 / 22,312
Reddit	232,965	11,606,919	602	41 (s)	153,756 / 23,295 / 55,911

A.3. Implementation details

We integrate our DGG into the official publicly available code of all baselines, without architectural modification. Models are trained and evaluated as their original implementation.

In our DGG, all MLPs use a single fully-connected layer of dimension 64. In our DGG, all MLPs use a single fully-connected layer of dimension 64. We use a Gumbel-Softmax temperature of 2.5. When adding Gumbel noise to the edge log-probabilities, we do not add any to the self-loops (i.e. the diagonal of the edge probability matrix). During training, we keep the Gaussian and Gumbel noise on, but turn it off during inference. While in practice it can be left on, we found it does not significantly impact the results.

A.4. Training details

We train the entire network end-to-end using the classification loss from the downstream model and an annealed MSE loss on the adjacency matrix generated by the DGG:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{class}} + \frac{\alpha}{M} \sum_{i=1}^{M} (y_i - \hat{y_i})^2, \tag{1}$$

where the first term is the classification loss from the downstream GCN, the second term is the MSE loss applied to every element $\hat{y_i}$ of the adjacency matrix $\mathbf{A} \in \hat{\mathbb{R}}^{N \times N}$ for which we have a node label, and α is loss weight. The model can be trained by annealing α smoothly or in a stepwise manner. In practice we keep α constant for the first 100 epochs and then set it to zero for the rest of the training schedule (where the total number of epochs is determined by the schedule of the downstream GCN).

A.5. Comparison with bespoke architectures experiment settings

In Table 4 of the main paper we compare against bespoke SOTA architectures which learn the graph structure. Here we explain the choice of our backbone and the experiment setting to compare against each architecture:

1. Setting 1: IDGL [3] uses an attention mechanism so we choose the Graph Attention Network (GAT) [35] as our backbone. The experiment setting is the same as that in [16].

2. Setting 2: Both LDS [9] and SLAPS [8] use regular graph convolutions so we choose the GCN [16] as our backbone. In this setting there is no input graph, and the training split includes half of the validation graph.

3. Setting 3: NodeFormer [37] is based on a Transformer [34] so we choose GAT [35] as our backbone. The data splits now include 50 %, 25 % and 25 % of the full graph for training, validation and testing respectively.
4. Setting 4: VGCN [7] uses a variational framework, and as there is no directly applicable backbone, we compare it against our best performing backbone GAT [35]. The training split includes half of the validation set.

A.6. Robustness to noise

We test the effect of the DGG when the input graph has random edges added across it. We do this by adding edges between previously unconnected nodes. Broadly, the results in Fig. 1 demonstrate the detrimental effects of noisy edges on classification accuracy, but the inclusion of the DGG can mitigate this. Interestingly, the state-of-the-art GCNII [2] demonstrates the largest drops in accuracy as the noise increases, which may be attributed to the depth of their graph convolution layers. In such deeper message-passing models, the edge structure is even more significant, highlighting the importance of learning a structure that prevents the propagation of irrelevant information.

B. Trajectory Prediction Experiments

B.1. Dataset details

ETH and UCY. ETH [27] and UCY [18] are two common benchmarks for pedestrian trajectory prediction. These datasets consist of 5 subsets of widely used real-world pedestrian trajectories [**?**, **?**, 22, **?**]. The primary challenge in these datasets are the frequent interactions of agents in very crowded scenes. Furthermore, the number of pedestrians varies considerably. Some frames contain only 2 pedestrians, while many have over 50.

SportVU. The STATS SportVu [32] is a tracking dataset composed of multiple NBA seasons. Each scene consists of two teams of 5 players, with each team categorized as either making an offensive or defensive play in a particular game. Each play contains 50 timesteps sampled at 5Hz, with the player trajectories expressed in (x, y, z) coordinates.

Stanford Drone. The Stanford Drone Dataset (SDD) [28] is a large dataset with 20 different top-down scenes across multiple areas at Stanford University. Each scene consists of agents of different types, from pedestrians and skaters to cars and buses. Trajectories are recorded at 2.5Hz and expressed in (x, y) world coordinates. Despite the heterogeneity of agents, the maximum number of agents in any one scene is 21.

Baselines. We integrate our DGG module into two state-of-the-art trajectory prediction pipelines: **Social-STGCNN** [22] built upon a spatio-temporal convolutional network using graphs to represent pedestrian trajectories and **DAGNet** [23] built upon a VAE backbone [14] with a graph attention network modelling agent interactions across a fully-connected graph. Our DGG is placed within both networks to generate the adjacency matrix on the fly and forms part of its forward and backward pass. To integrate the DGG with DAGNet's attention mechanism, the adjacency generated is multiplied by the attention weights.

Implementation details. We integrate DGG into the publicly available code of each method, without any architectural modification. We use the same DGG hyperparameters as for node classification except the intermediate loss is disabled and the training signal is entirely from the downstream task.

B.2. Qualitative results

In Fig. 2 we plot the node-degree distribution learned by our DGG across multiple datasets. While on average, a pedestrian may only look at their 2 nearest neighbors in crowded scenes such as Zara1 and Univ, this can increase to almost 5 nearest neighbors in some cases. This suggests that both a fully-connected graph, or one with a fixed node-degree like DGM [13] are both suboptimal.

Figure 3 compares our predicted trajectories to DGM [13], on the SportVU dataset. As shown, our trajectories are closer to the ground truth. We illustrate this further in Fig. 4, which shows the graph generated by our DGG for 3 different basketball players in a game. The figure demonstrates how our DGG lets each player look at a different number of neighbors, while DAGNet [23] forces each player to look at all others in the game.

References

[1] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[2] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, pages 1725–1735. PMLR, 2020. 2

[3] Yu Chen, Lingfei Wu, and Mohammed Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in neural information processing systems*, 33:19314–19326, 2020. 1

[4] Nicholas Choma, Federico Monti, Lisa Gerhardt, Tomasz Palczewski, Zahra Ronaghi, Prabhat Prabhat, Wahid Bhimji, Michael M Bronstein, Spencer R Klein, and Joan Bruna. Graph neural networks for icecube signal classification. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 386–391. IEEE, 2018.

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29:3844–3852, 2016.

[6] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and

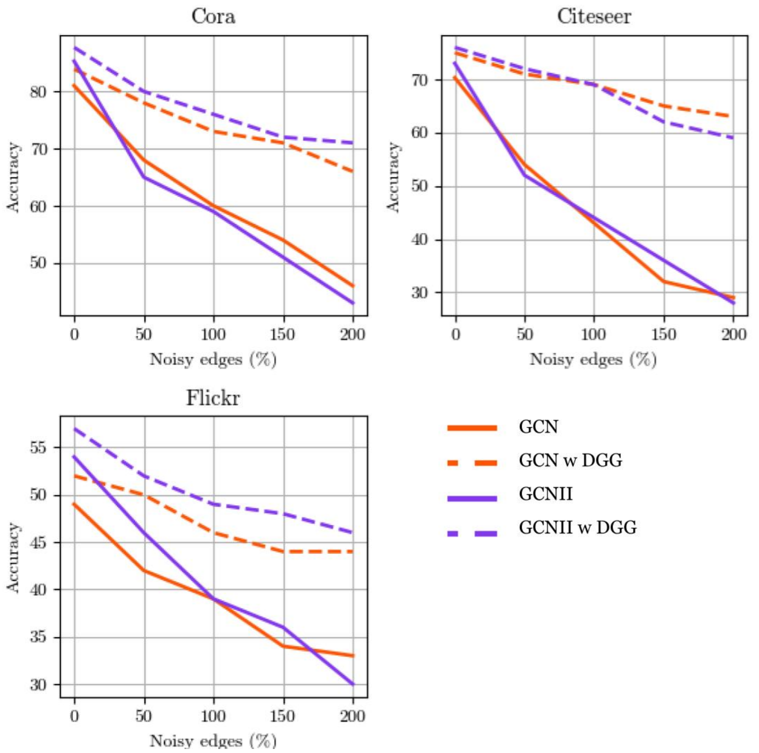


Figure 1. Node classification accuracy with noisy edges added to the input graph of different datasets.

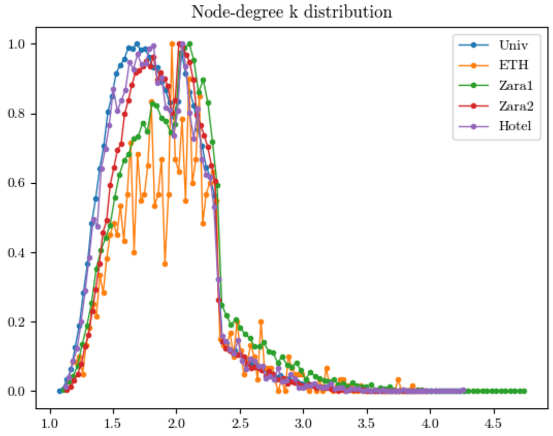


Figure 2. Distribution of the learned node degree k over the test split for different trajectory prediction datasets.

Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

[7] Pantelis Elinas, Edwin V Bonilla, and Louis Tiao. Variational inference for graph convolutional networks in the absence of graph data and adversarial settings. *Advances in Neural Information Processing Systems*, 33:18648–18660, 2020. 2

[8] Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. Slaps: Self-supervision improves structure learning for graph neural networks. *Advances in Neural Information Processing Systems*, 34:22667–22681, 2021. 1

[9] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *International conference on machine learning*, pages 1972–1982. PMLR, 2019. 1

[10] Pablo Gainza, Freyr Sverrisson, Frederico Monti, Emanuele Rodola, D Boscaini, MM Bronstein, and BE Correia. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*, 17(2):184–192, 2020.

[11] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1025–1035, 2017. 1

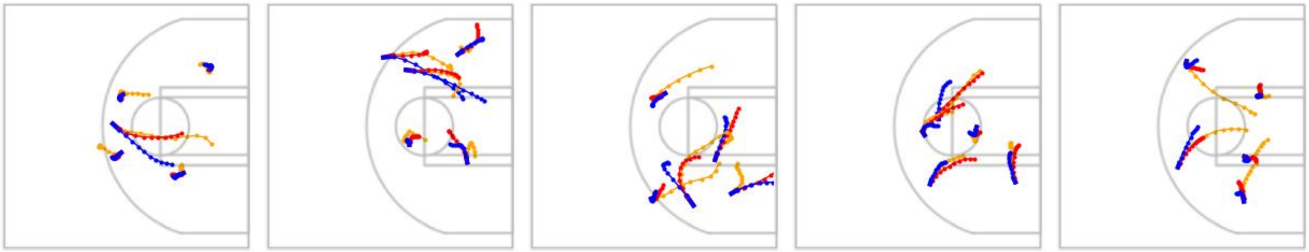
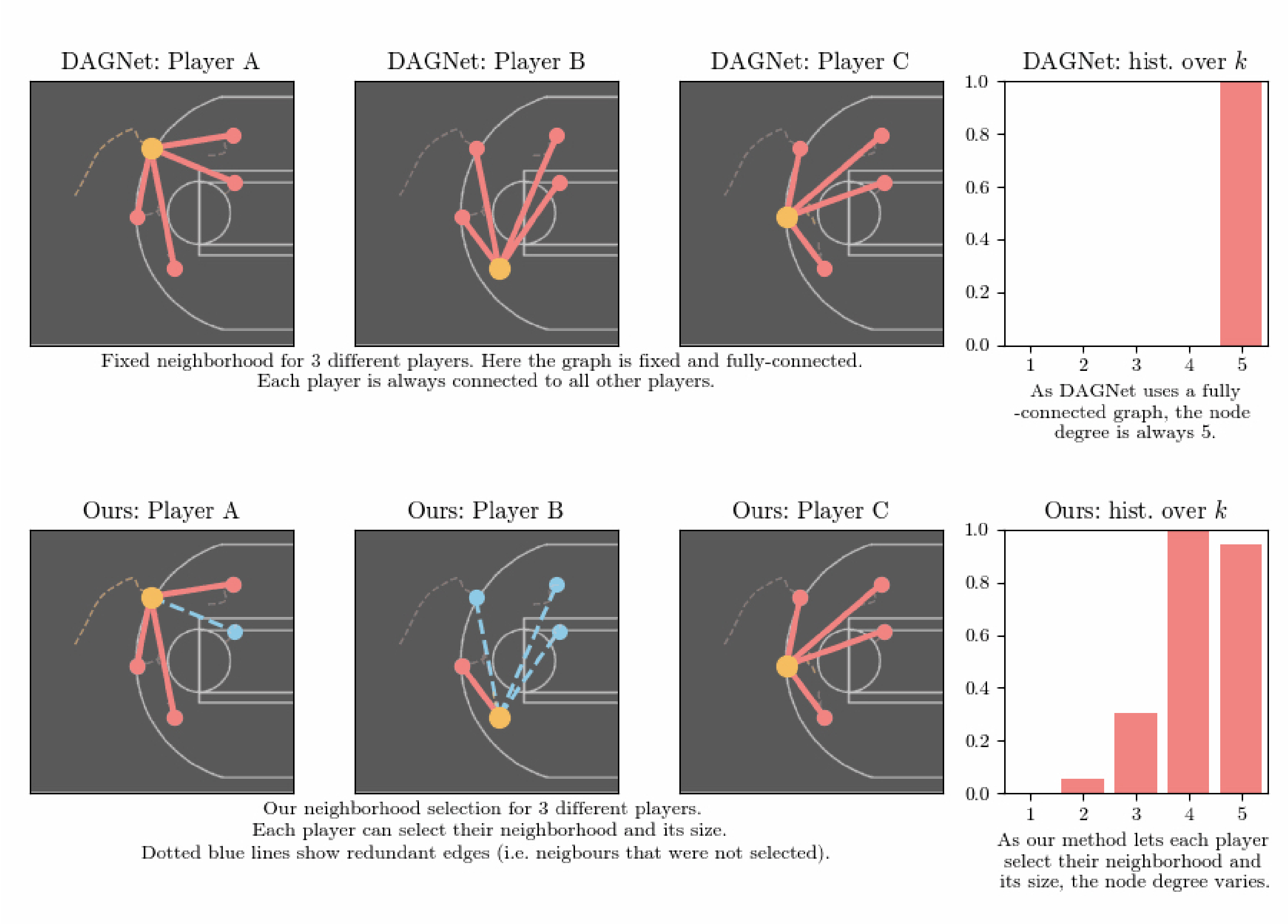


Figure 3. Qualitative results for trajectory prediction on the SportVU dataset. Orange: ground truth; Blue: DGM prediction; Red: our prediction.



Fixed neighborhood for 3 different players. Here the graph is fixed and fully-connected.
Each player is always connected to all other players.

As DAGNet uses a fully -connected graph, the node degree is always 5.

Our neighborhood selection for 3 different players.
Each player can select their neighborhood and its size.
Dotted blue lines show redundant edges (i.e. neigbours that were not selected).

As our method lets each player select their neighborhood and its size, the node degree varies.

Figure 4. Graph topology visualization in basketball players on the SportVU dataset: ours vs. DAGNet [23]. We display the selected neighborhood for 3 different players and a histogram of the node-degree k accumulated over the dataset/scene. First row: DAGNet's fully-connected graph, second row: our DGG.

[12] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[13] Anees Kazi, Luca Cosmo, Nassir Navab, and Michael Bronstein. Differentiable graph module (dgm) for graph convolutional networks. *arXiv preprint arXiv:2002.04999*, 2020. 2

[14] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[15] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, pages 2688–2697. PMLR, 2018.

[16] Thomas N. Kipf and Max Welling. Semi-supervised classifica-

tion with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 1

[17] Danning Lao, Xinyu Yang, Qitian Wu, and Junchi Yan. Variational inference for training graph neural networks in low-data regime through joint structure-label estimation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 824–834, 2022.

[18] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007. 2

[19] Guohao Li, Matthias Müller, Guocheng Qian, Itzel Carolina Delgadillo Perez, Abdulellah Abualshour, Ali Kassem Thabet, and Bernard Ghanem. Deepgcns: Making gcns go as deep as cnns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[20] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 143–152, 2020.

[21] Dongsheng Luo, Wei Cheng, Wenchao Yu, Bo Zong, Jingchao Ni, Haifeng Chen, and Xiang Zhang. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 779–787, 2021.

[22] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14424–14432, 2020. 2

[23] Alessio Monti, Alessia Bertugli, Simone Calderara, and Rita Cucchiara. Dag-net: Double attentive graph neural network for trajectory forecasting. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2551–2558. IEEE, 2021. 2, 4

[24] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5115–5124, 2017.

[25] Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M Bronstein. Fake news detection on social media using geometric deep learning. *arXiv preprint arXiv:1902.06673*, 2019.

[26] Pubmed. courtesy of the us national library of medicine. 1

[27] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009. 2

[28] Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision*, pages 549–565. Springer, 2016. 2

[29] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*, 2020.

[30] Berkant Savas and Inderjit S Dhillon. Clustered low rank approximation of graphs in information science applications. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, pages 164–175. SIAM, 2011.

[31] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035, 2019.

[32] Sportvu basketball, Sep 2019. 2

[33] Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNair, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackermann, et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2

[35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. 1, 2

[36] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[37] Qitian Wu, Wentao Zhao, Zenan Li, David Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 2

[38] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[39] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[40] Yujia Xie, Hanjun Dai, Minshuo Chen, Bo Dai, Tuo Zhao, Hongyuan Zha, Wei Wei, and Tomas Pfister. Differentiable top-k with optimal transport. *Advances in Neural Information Processing Systems*, 33:20520–20531, 2020.

[41] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pages 40–48. PMLR, 2016. 1

[42] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. GraphSAINT: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020. 1

[43] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pages 11458–11468. PMLR, 2020.