From Node Interaction to Hop Interaction: New Effective and Scalable Graph Learning Paradigm
Supplementary Material

A. Appendix

A.1. Details of Datasets

We provide the details of datasets in the following:

- Homophily Datasets
  - *Citeseer, Pubmed, Cora* [9]: For the benchmark citation datasets, nodes correspond to papers, edges correspond to citation links, the sparse bag-of-words are the features, and each node's label represents the paper's topic.
- Heterophily Datasets
  - *Texas, Wisconsin, Cornell* [11]: Nodes and edges represent the web pages and hyperlinks captured from the computer science departments of these universities in the WebKB dataset. Nodes' features are bag-of-word representations of contents on these web pages. Each node is labeled into five categories: student, project, course, staff, and faculty.
  - *Squirrel, Chameleon* [11]: Chameleon and Squirrel are web pages extracted from different topics in Wikipedia. Nodes and edges denote the web pages and hyperlinks among them, respectively, and informative nouns in the web pages are employed to construct the node features in the bag-of-word form. Webpages are labeled in terms of the average monthly traffic level.
  - *Actor* [11]: The actor network contains the co-occurrences of actors in films, which describes the complex relationships among films, directors, actors and writers. In this network, nodes and edges represent actors and their co-occurrences in films, respectively. The actor's Wikipedia page is used to extract features and node labels.
- Large-scale Datasets
  - *Flickr, Reddit* [6, 15]: Flickr and Reddit are inductive datasets for multiclass categorization. The goal of Reddit is to predict online post communities based on user comments. The task of Flickr is to classify the categories of images based on their description and common characteristics of online photographs.
  - *Products* [7]: The Products dataset is a large-scale Amazon product co-purchasing network in a transductive setting. Nodes represent products sold in Amazon, edges indicate the products purchased together, and features are 100-dimensional product descriptions after Principal Component Analysis. Labels are the category of products.

A.2. Implementation Details

Hyper-parameters. For HopGNN, we search the hyper-parameters, including learning rate from [0.01, 0.001, 0.005], weight decay from [0, 5e-4, 5e-5, 5e-6], dropout rate from [0.2, 0.4, 0.5, 0.6], α from [0.01, 0.1, 0.5, 0.8] and λ from [1e-4, 5e-4] via validation sets of each dataset. For other baselines, we search the layers/hops from [2, 8, 16, 32] and fix the hidden dimensions as 128. The search space of other hyper-parameters, such as learning rate, weight decay, and dropout, is the same as that of HopGNN.

Hardware and Environment. We run our experiments on a single machine with Intel Xeon CPUs (Gold 5120 @ 2.20GHz), one NVIDIA Tesla V100 GPU (32GB of memory) and 512GB DDR4 memory. We use PyTorch 1.11.0 with CUDA 10.2 to train the model on GPUs.

Throughput and Memory. For fairness, we set the hidden dimension to 256 and control the batchsize as 3000 across different models on the Products dataset. We report the hardware throughput and activation usage-based on [3, 5]. The throughput measures how many times each model can complete training steps within a second. We measure the activation memory using *torch.cuda.memory_allocated*.

Table 1. Mean test accuracy ± stdev. The best performance is highlighted. ‡ denotes the results obtained from previous works [14, 17].

	Texas	Wisconsin	Actor	Squirrel	Chameleon	Cornell	Citeseer	Pubmed	Cora	Avg
GCN+JK‡	66.49±6.64	74.31±6.43	34.18±0.85	40.45±1.61	63.42±2.00	64.59±8.68	74.51±1.75	88.41±0.45	86.79±0.92	65.79
GCN-Cheby‡	77.30±4.07	79.41±4.46	34.11±1.09	43.86±1.64	55.24±2.76	74.32±7.46	75.82±1.53	88.72±0.55	86.76±0.95	68.39
MixHop‡	77.84±7.73	75.88±4.90	32.22±2.34	43.80±1.48	60.50±2.53	73.51±6.34	76.26±1.33	85.31±0.61	87.61±0.85	68.21
GEOM ‡	67.57	64.12	31.63	38.14	60.90	60.81	77.99	90.05	85.27	64.05
FAGCN	78.11±5.01	81.56±4.64	35.41±1.18	42.43±2.11	56.31±3.22	76.12±7.65	74.86±2.42	85.74±0.36	83.21±2.04	68.18
DAGNN	70.27±4.93	71.76±5.25	35.51±1.10	30.29±2.23	45.92±2.30	73.51±7.18	76.44±1.97	89.37±0.52	86.82±1.67	64.43
HopGNN	81.35±4.31	84.96±4.11	36.66±1.39	60.95±1.65	70.13±1.39	83.70±6.52	76.16±1.53	89.98±0.39	87.12±1.35	74.56
HopGNN+	82.97±5.12	85.69±5.43	37.09±0.97	64.23±1.33	71.21±1.45	84.05±4.48	76.69±1.56	90.28±0.42	87.57±1.33	75.53
HopGNN+SCL	81.65±7.47	84.37±4.91	36.72±1.05	61.42±1.98	70.45±1.03	84.32±6.71	76.59±1.51	90.01±0.29	87.28±1.71	74.76

A.3. More Experiment Results

More Baselines. We provide more baseline results in Table 1, including 1) the classical advanced node interaction with high-order neighbor information: GCN+JK [13], GCN-ChebyNet [4], and MixHop [1]. 2) the heterophilic GNNs: Geom-GCN [11] and FAGCN [2] and 3) decoupled GNN: DAGNN [10]. However, their average performance on nine datasets is less than 70, indicating that they have been shown to be outperformed by the SOTA methods.

Supervised Contrastive Learning Objective. As discussed in [16], the supervised contrastive loss [8] (SCL) may help to regularize the feature space and make it more discriminative, *i.e.*, minimizing the SCL is equivalent to minimizing the class-conditional entropy $\mathcal{H}(\mathbf{H}|\mathbf{Y})$ and maximizing the feature entropy $\mathcal{H}(\mathbf{H})$. Therefore, it can also be used to enhance the discriminative ability of hop interaction, and we provide the results of HopGNN with SCL in Table 1. Formaly, the SCL objective can be defined as:

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{SCL}}, \tag{1}$$

$$\mathcal{L}_{\text{SCL}} = \sum_{i \in I_{\mathcal{L}}} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp\left(\mathbf{h}_i \cdot \mathbf{h}_p / \tau\right)}{\sum_{a \in A(i)} \exp\left(\mathbf{h}_i \cdot \mathbf{h}_a / \tau\right)}, \tag{2}$$

where $i \in I_{\mathcal{L}} \equiv \{1 \dots 2b\}$ is the index of an arbitrary augmented sample within a training batch (size=b), and $A(i) \equiv I_{\mathcal{L}} \backslash \{i\}$. $P(i) \equiv \{p \in A(i) : \mathbf{y}_p = \mathbf{y}_i\}$ is the set of indices of all positives that share same label y in a multi-view batch distinct from i, and $|P(i)|$ is its cardinality.

As shown in the last row of Table 1, the result demonstrates that the SCL can also improve the performance of HopGNN and is comparable with HopGNN+, which validates the compatibility of the HopGNN framework to different SSL objectives. However, the complexity of SCL is $O(N^2)$ in a batch due to the instance discrimination task. Therefore, the SCL is not as scalable as the feature-level SSL used in HopGNN+.

The Effects of Interaction Layers. We test the effects of different interaction layers in HopGNN under various datasets, including Cora, Citeseer, and Pubmed as homophily examples; Chameleon and Squirrel as heterophily examples; Flickr and Products as large-scale examples. In Figure 1, we observe that HopGNN achieves competitive results using two interaction layers in most cases. Thus, we adopt the default layers of hop interaction as two. Compared with homophily datasets, increasing the layers of hop interaction would remarkably boost the performance of heterophily datasets. The reason may be that such high-order hop interactions may capture the co-occurrence pattern of multi-hop neighbors, which contain the discriminative clues for heterophily.

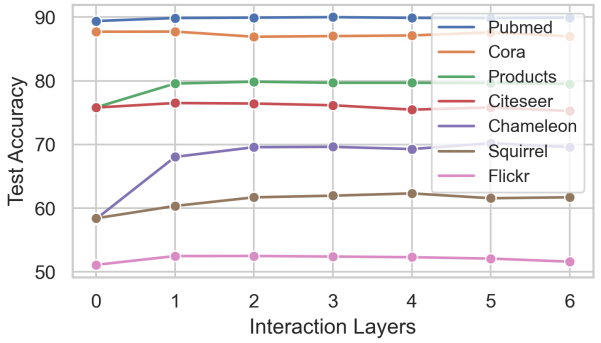

Figure 1. Classification accuracy with different interaction layers.

Training Efficiency. In Figure 2, we have added experiments with representative baselines. HopGNN converges most quickly and performs best empirically, which also validates that our hop interaction framework can achieve a better trade-off between effectiveness and efficiency.

Visualization of Representation. To qualitatively investigate the effectiveness of the learned feature representation of nodes, we provide a visualization of the t-SNE [12] for the last layer features of GCN (node interaction), SIGN (decoupled), and HopGNN (hop interaction) on the Chameleon (Figure 3) and Cora (Figure 4). Compared with GCN (first row), the SIGN (second row) and HopGNN (last row) are both robust when increasing layers under the Chameleon and Cora. Moreover, the representation of HopGNN ex-

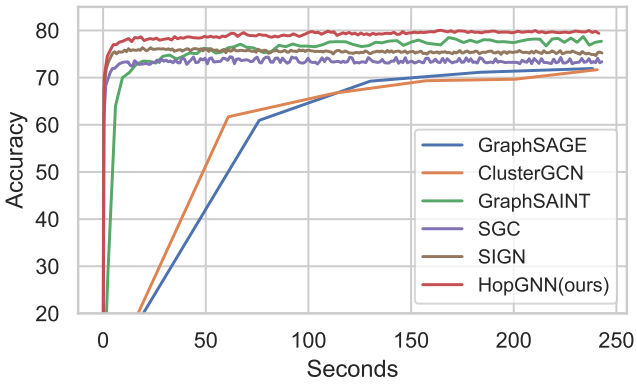

Figure 2. Empirical efficiency study on the largest Product dataset: x-axis shows the total training time (previous 250 seconds), y-axis is the accuracy on the test set.

hibits more discernible clustering in Chameleon. Note that these clusters correspond to the five classes of the dataset, verifying the better discriminative power of HopGNN under different layers in Heterophily.

References

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International Conference on Machine Learning*, pages 21–29. PMLR, 2019. 2

[2] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3950–3957, 2021. 2

[3] Jianfei Chen, Lianmin Zheng, Zhewei Yao, Dequan Wang, Ion Stoica, Michael W Mahoney, and Joseph E Gonzalez. Actnn: Reducing training memory footprint via 2-bit activation compressed training. In *International Conference on Machine Learning*, 2021. 1

[4] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016. 2

[5] Keyu Duan, Zirui Liu, Peihao Wang, Wenqing Zheng, Kaixiong Zhou, Tianlong Chen, Xia Hu, and Zhangyang Wang. A comprehensive study on large-scale graph training: Benchmarking and rethinking. In *Advances in Neural Information Processing Systems*, 2022. 1

[6] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1025–1035, 2017. 1

[7] 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 *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 1

[8] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020. 2

[9] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. 1

[10] Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 338–348, 2020. 2

[11] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 2020. 1, 2

[12] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 2

[13] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pages 5453–5462. PMLR, 2018. 2

[14] Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 1287–1292. IEEE, 2022. 2

[15] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor K. Prasanna. Graphsaint: Graph sampling based inductive learning method. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 1

[16] Yifan Zhang, Bryan Hooi, Dapeng Hu, Jian Liang, and Jiashi Feng. Unleashing the power of contrastive self-supervised visual models via contrast-regularized fine-tuning. *Advances in Neural Information Processing Systems*, 34:29848–29860, 2021. 2

[17] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Advances in Neural Information Processing Systems*, volume 33, 2020. 2

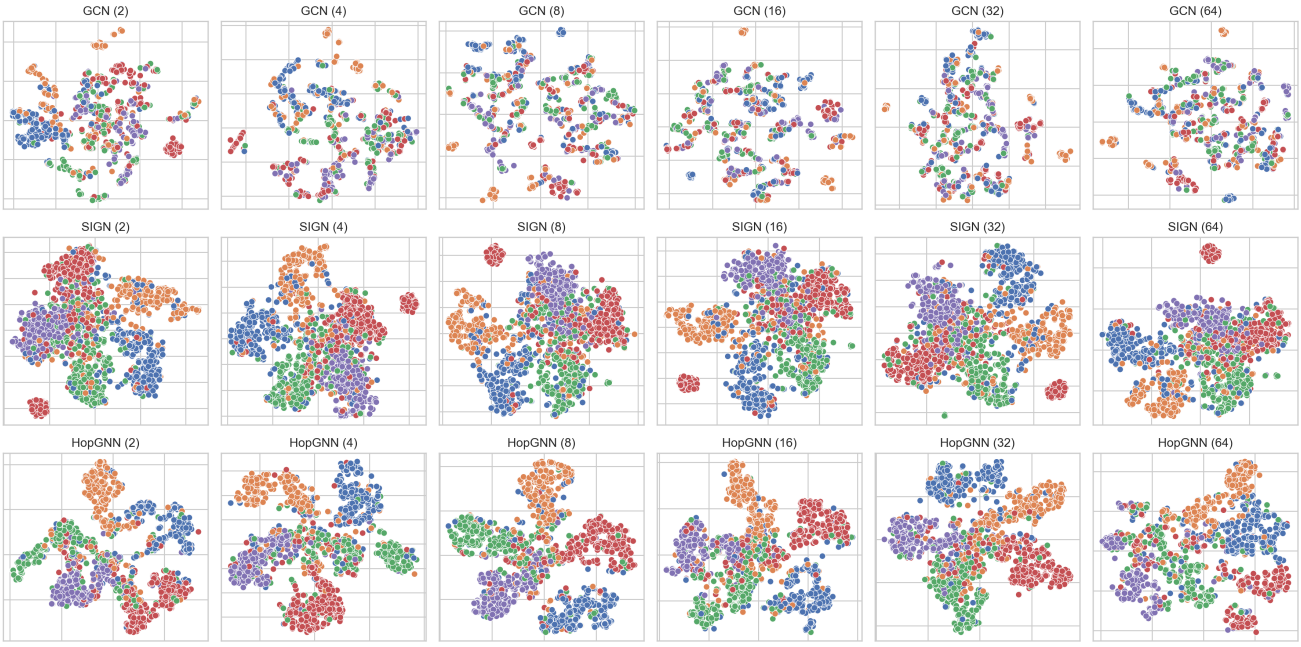

Figure 3. t-SNE visualization of node representations derived by different numbers of layers/hops of models on **Chameleon**. Colors represent node classes, and the number in the bracket indicates the layers/hops. Note that these clusters correspond to the five classes, indicating that HopGNN has better discriminative power under different layers in heterophily.

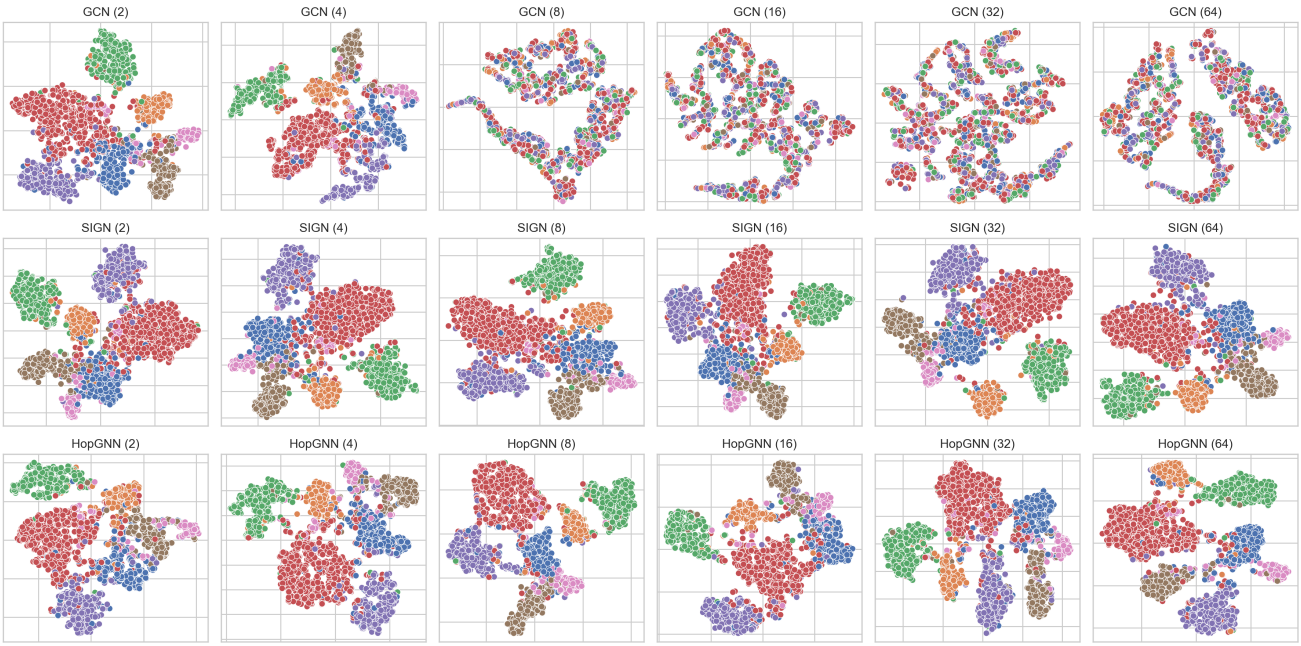

Figure 4. t-SNE visualization of node representations derived by different numbers of layers/hops of models on **Cora**. Compared with GCN, the representation of both SIGN and HopGNN are discriminative when increasing layers in a homophily scenario. However, due to the simplicity of homophily datasets, the SIGN and HopGNN achieve similar results.