Bi-GCN: Binary Graph Convolutional Network Supplementary Material

1. Vector Binarization

Here, we introduce the vector binarization approach [2], which is the basis of our binarization process. Considering that there exists a vector $V = (V_1, V_2, ..., V_t)$, this approach aims to obtain its binarized approximation with a binary vector $V_B = \{-1, 1\}^t$ and a real-valued scalar α, such that $V \approx \alpha V_B$. The approximation can be formulated as

$$J_v(V_B, \alpha) = ||V - \alpha V_B||_2^2. \tag{1}$$

By minimizing the above optimization problem, the optimal solution can be computed via

$$V_B^* = sign(V), \tag{2}$$

$$\alpha^* = \frac{1}{t}||V||_1, \tag{3}$$

where $sign(\cdot)$ is the signum function which extracts the sign of a real number.

Then, if there exists another vector $I = (I_1, I_2, ..., I_t)$, the inner product of I and V can be approximated via

$$I \cdot V \approx \alpha^* I \cdot V_B^*, \tag{4}$$

where $\cdot$ denotes the vector inner product. If a further binarization to the vector I is desired to compress this inner product, it can be achieved via $I \cdot V \approx \alpha\beta I_B \cdot V_B$, where I_B is a binary vector and β is a scalar. If we want to minimize the straightforward approximation error $|I \cdot V - \alpha\beta I_B \cdot V_B|$ and compute the optimal solution to this optimization problem, an optimal solution, $|I_B \cdot V_B| = 1$ and $\alpha\beta = sign(I_B \cdot V_B) I \cdot V$, can easily be calculated. Unfortunately, this solution possesses a strong dependency on the value of $I \cdot V$ and it tends to lose large amount information of the original vectors. To alleviate this issue, the approximation problem of the inner product $I \cdot V$ is defined as

$$J_{ip}(\alpha, \beta, I_B, V_B) = ||I \odot V - \alpha\beta I_B \odot V_B||_2^2, \tag{5}$$

where $\odot$ denotes the element-wise product. Similar to Eq. 1, the optimal solution can be calculated via

$$\alpha^* = \frac{1}{t}||V||_1, \tag{6}$$

$$\beta^* = \frac{1}{t}||I||_1, \tag{7}$$

$$V_B^* = sign(V), \tag{8}$$

$$I_B^* = sign(I). \tag{9}$$

Then, Eq. 4 can be reformed to

$$I \cdot V \approx \alpha^*\beta^* I_B^* \cdot V_B^*. \tag{10}$$

Eq. 10 is essentially the result of binarizing both I and V according to the vector binarizing algorithm.

2. More Results

Here, we also present the results on CiteSeer [3] for the transductive learning task, OBGN-Products [1] for the inductive learning task and ModelNet40 [5] for point cloud classification task to further verify the effectiveness of our binarization method. The same data division strategy as Planetoid [6] for CiteSeer, OGB benchmarks [1] for OGBN-Products and DGCNN [4] for ModelNet40 are adopted.

Table 1. Transductive learning results on CiteSeer. (M.S., D.S, and C.O. are the abbreviations of Model Size, Data Size and Cycle Operations, respectively.)

Networks	CiteSeer			
	Accuracy	M.S.	D.S.	C.O.
FastGCN	68.8 ± 0.6	927.25K	40.0M	7.90e8
SGC	71.9 ± 0.1	86.79K	40.0M	7.32e7
GAT	72.5 ± 0.7	927.8K	40.0M	7.91e8
GCN	**70.9 ± 0.5**	927.25K	40.0M	7.90e8
Bi-GCN	68.8 ± 0.9	**29.25K**	**1.48M**	**1.31e7**

As is shown in Table 1, our Bi-GCN can achieve ∼60.1x faster inference speed and ∼31.7x lower memory consumption than the uncompressed GNNs, with a comparable performance, similar to the results on the other two citation networks.

Table 2 shows the results of OGBN-Products, which is a medium-scale dataset in the OGB benchmarks with 2,449,029 nodes and 61,859,140 edges. The benchmark

Table 2. Accuracy on OGBN-Products (averaged over 10 runs). (D.S. and C.O. are the abbreviations of Data Size and Cycle Operations. GraphSAINT* corresponds to GraphSAINT with the SAGE aggregation function.)

Networks	Reddit		
	F1-micro	D.S.	C.O.
GraphSAGE	**78.7 ± 0.4**	934.23M	2.03e11
Bi-GraphSAGE	76.8 ± 0.3	**38.54M**	**2.45e10**
GraphSAINT	**79.1 ± 0.3**	934.23M	2.03e11
Bi-GraphSAINT*	77.5 ± 0.4	**38.54M**	**2.45e10**

results of GraphSAGE and GraphSAINT (with SAGE aggregation) are employed as our baselines (reported by OGB Team). The proposed binarization methods is applied to these two GNNs to generalize their binarized version. The results indicate that, our binarized GNNs can still achieve ∼24x data compression and ∼8x inference speedup on this challenging dataset, with comparable results.

Table 3. Results on ModelNet40 (1024 points).

Methods	Accuracy
DGCNN	92.89
Bi-DGCNN	88.29

Table 3 shows the results on ModelNet40. Note that DGCNN[4] is selected as the baseline method because it is a popular GNN on point cloud classification task. Bi-DGCNN is constructed by our binarization methods. The results demonstrate that our binarized GNN is still effective for this graph-classification tasks.

References

[1] 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 *NIPS*, 2020.

[2] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, pages 525–542, 2016.

[3] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[4] 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.

[5] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *IEEE,CVPR*, pages 1912–1920, 2015.

[6] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *ICML*, pages 40–48, 2016.