Technical Report for OGB Graph Property Prediction

Yan Wang, Hao Zhang, Jing Yang, Ruixin Zhang, Shouhong Ding

Tencent Youtu Lab, Shanghai, China

Abstract

This technical report introduces our solution for two OGB Graph Property Prediction challenges ogbg-molhiv and ogbg-molpcba.

1. Introduction of Open Graph Benchmark Challenge

The ogbg-molhiv and ogbg-molpcba datasets are two molecular property prediction datasets of different sizes: ogbg-molhiv (small) and ogbg-molpcba (medium). They are adopted from the MoleculeNet [1], and are among the largest of the MoleculeNet datasets. All the molecules are pre-processed using RDKit. The task is to predict the target molecular properties as accurately as possible, where the molecular properties are cast as binary labels, e.g, whether a molecule inhibits HIV virus replication or not. Specifically, for ogbg-molhiv, we use ROC-AUC for evaluation.

2. Method

2.1. Heterogeneous interpolation on graph

We propose Heterogeneous interpolation method, which can be applied as a node augmentation method to solve graph classification task. We mark the featuremap of a graph consisted of node features and edge connections as $G = \{N, E\}$. Firstly, we randomly select several nodes and clear their corresponding feature value as 0. Then, we update the feature of these nodes as the normalized feature mixture of their neighbour nodes. For selected Node i, its feature is updated as :$N_i = \sum k_j * N_j$, where $E_i, j = 1$ and k_j is the mixing ratio of Node j. This process is showed in Figure1.

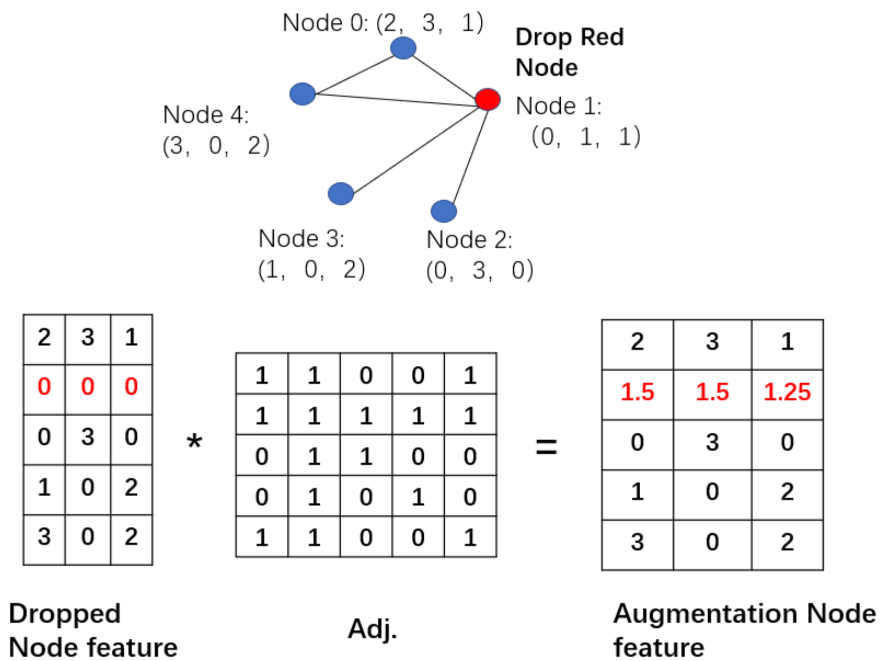

Figure 1: Here we explain the Heterogeneous interpolation process using a graph consisted of 5 nodes as an example. For simplicity, we show the result when $k_j = 1$.

2.2. KL Divergence Constraint

We add KL Divergence constraint loss to restrain the distributions of graph feature augmented from same graphs to be similar[1]. For two distribution p and q, the more common way to see KL divergence written is as follows: $D(p,q) = \sum_{i=1}^{N} p(x_i) \cdot log\frac{p(x_i)}{q(x_i)}$.

3. Result

For ogbg-molhiv, as the dataset is relatively small, we use DeeperGCN[2] as our backbone. The result is showed in Table1.

For ogbg-molpcba, we use Graphormer[3] as our backbone. The result is showed in Table2.

Table 1: **Verification ROC-AUC on ogbg-molhiv dataset.**

Datasets	**Method**	**Test AUROC**	**Validation AUROC**	**Parameters**	**Hardware**
ogbg-molhiv	DeepGCN+HIG	0.8403 ± 0.0021	0.8176 ± 0.0034	1019408	Tesla V100 (32GB)

Table 2: **Verification AP on ogbg-molpcba dataset.**

Datasets	**Method**	**Test AP**	**Validation AP**	**Parameters**	**Hardware**
ogbg-molpcba	Graphormer+HIG	0.3167 ± 0.0034	0.3252 ± 0.0043	119529665	Tesla V100 (32GB)

References

[1] Z. Yuan, Y. Yan, M. Sonka, T. Yang, Robust deep auc maximization: A new surrogate loss and empirical studies on medical image classification, arXiv preprint arXiv:2012.03173.

[2] G. Li, C. Xiong, A. Thabet, B. Ghanem, Deepergcn: All you need to train deeper gcns, arXiv preprint arXiv:2006.07739.

[3] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, T.-Y. Liu, Do transformers really perform bad for graph representation?, arXiv preprint arXiv:2106.05234.