[KDD 2020] GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. [pdf] [code]

Node/Graph Tasks: Node classification, graph classification and similarity search

Training Type: pretraining and feed the extracted representations to logistic regression or SVM for downstream tasks. Or directly full fine-tuning.

Pretext task data: structure The pretext task here is to train our GNN model to learn structure embeddings capable of discriminating subgraph instances.

Initial short summary here
In view that most representation learning work on graphs are only suitable one single graph or a fixed set of graphs and very limited work can be transferred to out-of-domain data and tasks, this paper is to universally learn transferable representative graph embeddings from networks. The novice of this work is that it focues on structural representation learning without node attributes and node labels.

First subgraph instances are designed and sampled. For a certain vertex, an instance is defined as its r-ego network which is the subgraph induced by nodes that have shortest path with length shorter than r. The GCC model treats each r-ego network as a distinct class of its own and encourages the model to distinguish similar instances from dissimilar instances. Next, the similar instance pair is defined as augmenting the same r-ego network by two random augmentation ways. Specifically we begin a random walk with restart at ego vertex v and define the subgraph induced by nodes that are visited during random walk as the augmented version of the r-ego network. By performing random walk twice, we create two data augmentation for each ego vertex v, which forms a similar instance pair (x^q, x^{k_+}). And negative instance pair corresponds to two subgraphs augmented from different r-ego networks. Then the node features are defined by top eigenvectors of the normalized graph laplacian with the addition of the one-hot encoding of vertex degrees and the binary indicator of the ego vertex. After feeding such node features to graph neural networks to get the node embeddings, the InfoNCE is adopted to optimize the GNN encoder:

$$\mathcal{L} = -\log \frac{\exp(\mathbf{q}^{\mathrm{T}}\mathbf{k}_+/\tau)}{\sum_{i=0}^{K}\exp(\mathbf{q}^{\mathrm{T}}\mathbf{k}_i/\tau)}, \tag{66}$$

where $\mathbf{k}_+, \mathbf{k}_i, \mathbf{q}$ are instances embeddings from GNN encoder.

Bibtex:
inproceedingsqiu2020gcc, title=Gcc: Graph contrastive coding for graph neural network pre-training, author=Qiu, Jiezhong and Chen, Qibin and Dong, Yuxiao and Zhang, Jing and Yang, Hongxia and Ding, Ming and Wang, Kuansan and Tang, Jie, booktitle=Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages=1150–1160, year=2020