[ICLR 2020] Strategies for Pre-training Graph Neural Networks. [paper] [code]

Node/Graph Tasks: Graph classification

Training Type: pre-training and fine tuning

Pretext task data: this paper includes four self-supversied learning strategies, two focusing on nodes and two focusing on graphs, all of which utilize node features and graph topology

Context prediction

The pretext task here is to train a GNN encoder such that nodes appearing in similar structure contexts are mapped to nearby embeddings. For every node v, K-hop neighborhood of v contains all nodes and edges that are at most K-hops away from v in the graph. The context graph of v is a subgraph that is between r_1-hops and r_2-hops away from node v. It is required $r_1 < K$ so that some nodes are shared between the neighborhood and the context graph, which is referred as context anchor nodes. Two GNN encoders are set up: the main GNN encoder is to get the node embedding $h_v^{(K)}$ based on their K-hop neighborhood nodes features and the auxiliary GNN is to get the node embeddings of every other node in the context graph, which are averaged to get the node context embedding c_v^G. The negative sampling is then used where the positive samples are the same node in the same graph while the negative samples are different nodes. The learning objective is a binary classification of whether a particular neighborhood and a particular context graph belong to the same node, which is realized by the cross entropy loss:

$$\mathcal{L} = \frac{1}{|\mathcal{T}|} \sum_{v \in \mathcal{T}} (y_v \log(\sigma(\mathbf{h}_v^{(K),\mathrm{T}}, \mathbf{c}_{v'}^G)) + (1 - y_v) \log(1 - \sigma(\mathbf{h}_v^{(K),\mathrm{T}}, \mathbf{c}_{v'}^G))) \tag{53}$$

where $y_v = 1$ for the positive sample while $y_v = 0$ for the negative sample.

Attribute masking

The pretext task here is to mask node/edge attributes and let GNNs predict those attributes based on neighboring structure. Specifically we randomly mask input node/edge attributes by replacing them with special masked indicators. We than apply GNNs to obtain the corresponding node/edge embeddings (edge embeddings can be obtained as a sum of node embeddings of the edge's end nodes). Finally, a linear model is applied on top of embeddings to predict a masked node/edge attributes.

Supervised graph-level property prediction

The pretext task here is to apply a linear regression to the generated graph-level representation $\mathbf{h}_G$ to predict whether a given graph has a given functionality. Note that naively performing the extensive multi-task graph-level pre-training alone can fail to give transferable graph-level representations. One solution would be to select truly-relevant supervised pre-training tasks, which is extremely costly. Another solution is to first regularize GNNs at the level of individual nodes via node-level pre-training methods.

Structural similarity prediction

The pretext task here is to model the structural similarity of two graphs. Due to finding the ground truth graph distance values is difficult and large datasets usually have quadratic number of graph pairs, this paper leaves it for future work.

Initial short summary here

The main content is attached in the previous section. Results demonstrate that using an expressive model is crucial to fully utilize pre-training, and that pre-training can even hurt performance when used on models with limited expressive power.

Bibtex:

@articlehu2019strategies, title=Strategies for pre-training graph neural networks, author=Hu, Weihua and Liu, Bowen and Gomes, Joseph and Zitnik, Marinka and Liang, Percy and Pande, Vijay and Leskovec, Jure, journal=arXiv preprint arXiv:1905.12265, year=2019