[Openreview 2020] How to Find Your Friendly Neighborhood: Graph Attention Design with Self-Supervision [paper]

Node/Graph Tasks: Node classification

Training Type: joint training

Pretext task data: graph topology, node features
The pretext task here is to use weights learned in GAT-based model to predict the edges in the graph. The edge prediction problem is formulated as a binary classification problem.

Initial short summary here
Although graph attention network model (GAT) achieve the performance improvements over the original GCN, there is little understanding of what graph attention learns. To this end, this paper proposes a specific pretext task to leverage the edge information to supervise what graph attention learns. Two basic graph attention mechanisms, GAT's original single-layer neural network (GO) and dot-product (DP), are used to construct the more advanced attention networks, scaled dot product (SD) and mixed GO and DP (MX).

In original GAT model, for each pair of nodes with embeddings $\mathbf{h}_i^l, \mathbf{h}_j^l$ at layer l, the coefficient used for calculating the edge attention is $e_{ij}^{l+1} = a_e(W^{l+1}\mathbf{h}_i^l, W^{l+1}\mathbf{h}_j^l)$. In the dot-product (DP) attention, the coefficient is calculated as $e_{ij,\mathrm{DP}}^{l+1} = (W^{l+1}\mathbf{h}_i^l)^{\mathrm{T}} W^{l+1}\mathbf{h}_j^l$. Then two advanced edge attention, scaled dot-product and mixed GO and DP, are proposed. In scaled dot-product, we have:

$$e_{ij,\mathrm{SD}} = e_{ij,\mathrm{DP}}/\sqrt{F}, \tag{54}$$

$$e_{ij,\mathrm{MX}} = e_{ij,\mathrm{GO}}\sigma(e_{ij,\mathrm{DP}}), \tag{55}$$

where SD divides the dot-product of nodes by a square root of dimension as Transformer, which prevents some large values to dominate the entire attention after softmax. MX multiplies GO and DP attention with sigmoid, which can softly drop neighbors that are not likely linked. The binary classification self-supervised learning is to predict edges and here the training samples are a set of edges E and the complementary set $E^c = (V \times V)\backslash E$. To reduce the number of the possible negative cases in E^c, we arbitrarily choose a total of $p_n|E|$ negative samples E^- from E^c. Then the objective of layer l is:

$$\mathcal{L}^l = \frac{1}{|E \cup E^-|} \sum_{(j,i)\in E\cup E^-} \mathbf{1}_{(j,i)=1} \log \phi_{ij}^l + \mathbf{1}_{(j,i)=0} \log(1-\phi_{ij}^l) \tag{56}$$

By comparing the label-agreement and graph attention based on K-L divergence of the normalized attention $\alpha_{\mathbf{k}} = \{\alpha_{kk}, \alpha_{k1}, ..., \alpha_{kJ}\}$ with label agreement distribution for the center node k and its neighbors 1 to J. The label agreement distribution $\mathbf{l}_k = [l_{kk}, l_{k1}, ..., l_{kJ}]$ is defined by:

$$l_{kj} = \tilde{l}_{kj} / \sum_{s} \tilde{l}_{ks}, \tilde{l}_{kj} = 1(\text{if } k \text{ and } j \text{ have the same label or 0 otherwise}) \tag{57}$$

$$\mathrm{KLD}(\alpha_k, \mathbf{l}_k) = \sum_{j \in \mathcal{N}_k \cup \{k\}} \alpha_{kj} \log(\alpha_{kj} / l_{kj}) \tag{58}$$

The paper found that GO learns label agreement better than DP while DP predicts edge presence better than GO. Also, there is a trade-off between node classification and link prediction, which implies that it is hard to learn the relational importance from edges by simply optimizing graph attention for link prediction. Besides, the paper also performs a large amount of experiments to determine which graph attention, SD or MX works best for given graphs and found that it really depends on the graph homophily and average degree of the graph.

Bibtex:
@inproceedings kim2021how, title=How to Find Your Friendly Neighborhood: Graph Attention Design with Self-Supervision, author=Dongkwan Kim and Alice Oh, booktitle=International Conference on Learning Representations, year=2021, url=https://openreview.net/forum?id=Wi5KUNlqWty