MODEL DESCRIPTION

1. Motivation: Sparsity + Differentiable

1. Most of the current graphical pooling method aims to learn an allocation matrix $[N * K]$. The computation for the embedding of the aggregated nodes requires $O(N^2)$. $([K * N] * [N * D])$, thus non-scalable.

2. As far as I know, none of the current sparse graphical pooling method are differentiable. Typically, an importance score is learned and then sparsity is achieved by top selection.

3. For top selection method, the embedding for coarsened graph only contain information for selected cluster/nodes, some information in the original graph may be lost.

2. Model

2.1. Notations.

N : number of nodes,
K : number of vectors in the codebook,
D : dimension of embedding
A : Adjacency matrix [N*N]
X : Positional matrix [N*D]
F : Feature matrix [N*D]
C : codebook vectors [K*D]
DS : distance matrix [N*K]

2.2. Model for learning the topological structure.

2.2.1. *Algorithm.*

1. Initialize embedding X0, codebook C
2. Use two GCN layers to capture local structure: X = GCN(GCN(X0, A), A)
3. Compute the distance matrix DS, where $DS[i, j] = distances(X_i, C_j)$
4. Quantize the embedding with closest vector in the notebook Z

2.2.2. *Loss function.*

Consider the sum of vq-Loss and ELBO loss as the loss function, where vq Loss is the same as the original paper[1]: $||Z - sg(X)||^2 + \beta||X - sg(Z)||^2$.

ELBO loss is adapted to encode Dirichlet process:

- Original ELBO[2] is to maximize the lower bound

$$E_q[\log \frac{p(X, Z)}{q(Z)}] = \log p(X|Z) + E_q[\log \frac{p(Z)}{q(Z)}],$$

or equivalently to minimize

$$\log p(X|Z) + E_q[\log \frac{q(Z)}{p(Z)}],$$

where Z is the latent variable. The first term is also known as reconstruction term, while the second term is the KL-divergence between $q(Z)$ and $p(Z)$.

- For the reconstruction term, we consider

$$\frac{1}{|pos|} \sum_{(i,j)\in pos} \log p(Z_i^T Z_j) - \frac{1}{|neg|} \sum_{(i,j)\in neg} \log(1 - Z_i^T Z_j)$$

- Formally, the ELBO loss for vq-vae wouldn't contain the second term, since $q(Z)$ is a categorical distribution where the probablity for the closest class equals to 1. $((\hat{\pi}_1, \cdots, \hat{\pi}_i, \cdots, \hat{\pi}_K) = (0, \cdots, 1, \cdots, 0))$ To encode Dirichelet process in the loss function, we consider a continuous relaxation of this distribution as $\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^{N} \hat{\pi}_{nk}$, where $\hat{\pi}_{nk} \propto exp(-DS[n,k])$ and $\sum_{k=1}^{K} \hat{\pi}_{nk} = 1$.
- Consider the prior distribution as stick-breaking prior, where π is generated as follows:

  Sample v_i from $Beta(1, \alpha_0), (i = 1, 2, \cdots, K-1),\ v_K = 1$.

  Transoform $v$to $\pi_k = v_k \prod_{i=1}^{k-1}(1 - v_k), (i = 1, \cdots, K)$.
- Since it's hard to directly compute the margincal distribution $p(Z)$, we use a MC simulation to approximate the second term as

$$E_q[\log \frac{q(Z)}{p(Z)}] = \frac{1}{M} \sum_{m=1}^{M} E_q[\log \frac{q(Z)}{p(Z|v_m)}],\ v_m \sim Beta(1, \alpha_0)$$

- Thus,

$$L = \frac{1}{|pos|} \sum_{(i,j)\in pos} \log p(Z_i^T Z_j) - \frac{1}{|neg|} \sum_{(i,j)\in neg} \log(1 - Z_i^T Z_j) + \frac{1}{M} \sum_{m=1}^{M} E_q[\log \frac{q(Z)}{p(Z|v_m)}]$$

2.3. **Model for learning the feature.**

We assume that the feature matrix for each node is meaningful, and aim to learn a quantized vector which is able to reflect the original feature vector. Similar algorithm and loss function is considered.

2.3.1. *Algorithm.*

1. Initialize codebook C
2. Use two GCN layers to abosorb local structure: F = GCN(GCN(F, A), A)
3. Compute the distance matrix DS, where $DS[i,j] = distances(F_i, C_j)$
4. Quantize the embedding with closest vector in the notebook Z
5. Recover the initial embedding based on quantized latent embedding F_d = MLP(Z)

2.3.2. *Loss function.*

Similarly, the loss function is considered as the sum of reconstruction loss and KL loss, where KL loss is the same as last subsection. Reconstruction loss is computed based on the MSE between recovered feature embedding F_d and initial feature embedding F.

2.4. **Supervised learning.** For supervised learning, we consider the complete quantized embedding as a concatenation of quantized positional embedding and quantized feature embedding, and the label for each node is predicted using this quantized complete embedding.

2.5. **Practical concern.** Several practical concern is observed and recorded as floows:

1. We use dirichlet prior with different parameter α_0 to control the number of clusters, and thus a smaller α_0 correspond with less number of clusters. In pracctice, however, the KL for a smaller α_0 may be larger than that for a larger α_0, making the number of clusters for a smaller α_0 larger than that of a larger α_0.

2. In practice, $\beta-$VAE (use β to balance reconstruction loss and KL loss) is observed to have better performace. Also, paying different attention for positional and feature embedding might have better effect. We need to balanced between too many loss functions, especially when considering supervised learning.

3. In Stochastic Model, the model doesn't perform well when the true number of clusters is large.

4. The model quantize the embedding for each node. Unique embedding might be more helpful for node prediction.

5. Since we consider positional embedding, it is not inductive.

Potentially, change the strutucre of current method:

1. Other ways to update parmeters instead of using loss function 2. Consider mixing the two embedding / distances 3. Consider the codebook as complete graph, update the codebook (similar to MP on coarsened graph) 4. Generate unique node embedding during backpropagation.