Fotios Kapotos
fotiskapotos@gmail.com

Lab session # 7
ALTEGRAD 2023

12/07/25

1 Question 1

Single-head attention. For each head k, attention coefficients between node i and its neighbor j are computed as

$$e_{ij}^{(k)} = \mathrm{LeakyReLU}\Big((a^{(k)})^\top [\, W^{(k)} z_i \,\|\, W^{(k)} z_j \,]\Big), \qquad \alpha_{ij}^{(k)} = \frac{\exp(e_{ij}^{(k)})}{\sum_{\ell \in \mathcal{N}(i)} \exp(e_{i\ell}^{(k)})}.$$

The output representation of node i for head k is then

$$z_i^{(k)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(k)}\, W^{(k)} z_j.$$

Multi-head aggregation. The final node embedding is obtained by concatenating the outputs of all K heads:

$$z_i = \|_{k=1}^{K} z_i^{(k)} \;\in\; \mathbb{R}^{K F'_{\text{out}}}.$$

Number of parameters. Each head uses a projection matrix $W^{(k)} \in \mathbb{R}^{F_{\text{in}} \times F'_{\text{out}}}$ and an attention vector $a^{(k)} \in \mathbb{R}^{2F'_{\text{out}}}$. Thus, a single head has $F_{\text{in}} F'_{\text{out}} + 2F'_{\text{out}}$ parameters. For K heads, the total number of learnable parameters is

$$\#\text{params} = K F'_{\text{out}}(F_{\text{in}} + 2).$$

2 Question 2

1. Attention scores with identical features. Since all node features are equal,

$$x_i = c, \qquad \forall v_i \in V,$$

then after linear projection

$$W z_i = W c \;=\; h, \qquad \forall i.$$

Hence for any edge (i, j),

$$e_{ij} = \mathrm{LeakyReLU}\Big(a^\top [\, h \,\|\, h \,]\Big) = \text{const.}$$

Because all unnormalized scores are identical for node i's neighbors, the softmax gives

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})} = \frac{1}{|\mathcal{N}(i)|}.$$

2. Consequence on expressiveness. Attention becomes uniform and no longer differentiates between neighbors. The layer degenerates into simple neighborhood averaging:

$$z_i^{(t+1)} = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} W z_j^{(t)},$$

which is equivalent to a **mean aggregator GNN**, i.e. the propagation rule of the standard **Graph Convolutional Network (GCN)** or **GraphSAGE (mean)** without attention.
Thus the GAT no longer behaves as an attention mechanism; all neighbors contribute equally.

3. Why classification can still work. Even without feature information, the model exploits **graph structure alone**. Repeated neighborhood aggregation encodes each node's **structural position** (degree, community membership, connectivity patterns). On the Karate network, these structural signals correlate strongly with the two underlying classes, allowing the GNN to achieve accuracy above random guessing despite identical node features. "

3 Question 3

Conditional Variational Graph Autoencoder. Conditions can be introduced by extending the standard VGAE into a *Conditional Variational Graph Autoencoder (CVGAE)*. Let c denote some side information about the graph (e.g., number of nodes, number of blocks, graph type, class label, etc.). Both the encoder and decoder are conditioned on c:

$$q_\phi(z \mid G, c) = \mathcal{N}\big(z \mid \mu(X, A, c), \mathrm{diag}(\sigma^2(X, A, c))\big),$$

$$p_\theta(G \mid z, c).$$

In practice, the condition c is concatenated to the node features or to the latent representation and provided as additional input to the GNN encoder and to the MLP decoder:

$$\mu = \mathrm{GNN}_\mu(X, A, c), \qquad \log \sigma = \mathrm{GNN}_\sigma(X, A, c),$$

$$\hat{A} = \mathrm{MLP}(z, c).$$

This enforces the latent variable z to capture only graph variability *given* the condition, enabling controlled graph generation by sampling new graphs conditioned on desired properties.

4 Question 4

Reparameterization trick. The latent variable is generated as

$$z = \mu + \sigma \odot \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I).$$

This formulation decouples the randomness from the learnable parameters μ and σ, making z a deterministic, differentiable function of (μ, σ) and of a fixed noise source ε. As a result, gradients can be propagated through z, enabling standard back-propagation to optimize the encoder parameters.
If one instead sampled directly

$$z \sim \mathcal{N}(\mu, \sigma^2 I),$$

the sampling operation would be non-differentiable with respect to μ and σ. Consequently, gradients could not flow through the stochastic node, and the encoder could not be trained via back-propagation. The learning process would therefore break or require high-variance gradient estimators, leading to unstable or ineffective training.

References