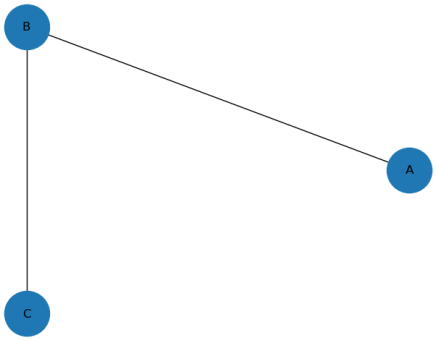
Mid-term Exam Solution (Graph Neural Networks –Fall 2025)

1. (Node2vec) (5p)
   a. (3p) Explain BFS, DFS.
      Ans:
      - BFS encourages walker to stay close to the starting node, focuses on exploring local neighborhood. BFS is good at capturing structural equivalence
      - DFS encourages walker to move farther away from the starting node, explores nodes deeper in the graph. DFS is good at capturing community or global relationships

   b. (2p) Consider an undirected graph G, with p = 2, q = 1. List all possible next nodes from node B and calculate unnormalized transition probabilities $\pi_{BAx} = \alpha_{pq}(A, x)$, where

$$\alpha_{pq}(A,x) = \begin{cases} \frac{1}{p}, if\ d(A,x) = 0 \\ 1, if\ d(A,x) = 1 \\ \frac{1}{q}, if\ d(A,x) = 2 \end{cases}$$



Ans:

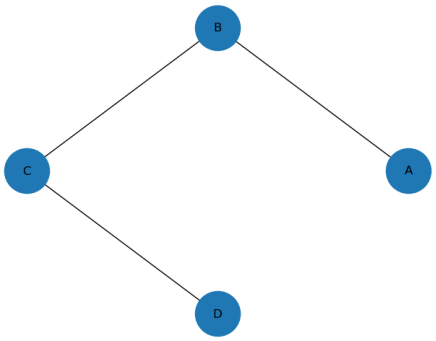
Possible next nodes from node B: {A, C}
Distance from A:
$d(A, A) = 0$
$d(A, C) = 2$
$\pi_{BAA} = \frac{1}{p} = \frac{1}{2}, \pi_{BAC} = \frac{1}{q} = 1$
$P(A|B, A) = \frac{0.5}{1.5} = \frac{1}{3},$
$P(C|B, A) = \frac{1}{1.5} = \frac{2}{3}$

2. (LINE) (5p)
   a. (2p) What is second-order proximity?

It captures the similarity of nodes' connection patterns. Two nodes have high second-order proximity if they share many common neighbors.

b. (3p) Consider an undirected graph G, with edge weights $w_{AB} = 2, w_{BC} = 1, w_{CD} = 3$, and node embedding $u_A = 1, u_B = 0.5, u_C = -0.5, u_D = -1$ and context node $u'_A = 0.5, u'_B = 1, u'_C = -1, u'_D = -0.5$. Calculate $p_2(A|B), p_2(B|B), p_2(C|B), p_2(D|B)$. Given $e^{0.25} = 1.28, e^{0.5} = 1.65, e^{-0.25} = 0.78, e^{-0.5} = 0.61$
Second order:

$$p_2(j|i) = \frac{e^{u'_j u_i}}{\sum_k e^{u'_k u_i}}$$



Ans:

$$e^{u'_A u_B} = e^{0.5*0.5} = e^{0.25} = 1.28$$
$$e^{u'_B u_B} = e^{1*0.5} = e^{0.5} = 1.65$$
$$e^{u'_C u_B} = e^{-1*0.5} = e^{-0.5} = 0.61$$
$$e^{u'_D u_B} = e^{-0.5*0.5} = e^{-0.25} = 0.78$$
$$Sum = 1.28 + 1.65 + 0.61 + 0.78 = 4.32$$
$$p_2(A|B) = \frac{1.28}{4.32} = 0.296$$
$$p_2(B|B) = \frac{1.65}{4.32} = 0.382$$
$$p_2(C|B) = \frac{0.61}{4.32} = 0.141$$
$$p_2(D|B) = \frac{0.78}{4.32} = 0.18$$

3. (GCN) (10p)
   a. (3p) Explain how information is propagated in a GCN layer.
   Each node aggregates and averages features from its neighbors (using the normalized adjacency matrix) and then applies a linear transformation and activation, updating its representation based on local graph structure
   b. (7p) Given a graph with an adjacency matrix A and initial node feature matrix $H^{(0)}$ as follows:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad H^{(0)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 2 \\ 2 & 2 \\ 2 & 0 \end{bmatrix}$$

Assume that the hidden layer of an GCN model of all nodes at layer (k) can be calculated as:

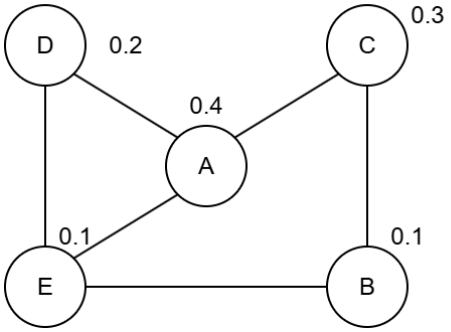
$$H^{(k)} = \sigma\left(A \cdot H^{(k-1)}\right),$$

where $H^{(k)}$ denotes the output at layer k, σ is a ReLU function $\mathrm{ReLU}(x) = \max(0, x)$. Calculate the output of the GCN model at layer $k = 1$.

Ans:

$$H^1 = \sigma(AH^0) = \mathrm{ReLU}\left(\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 2 \\ 2 & 2 \\ 2 & 0 \end{bmatrix}\right) = \mathrm{ReLU}\left(\begin{bmatrix} 2 & 2 \\ 3 & 2 \\ 1 & 0 \\ 3 & 5 \\ 2 & 3 \end{bmatrix}\right) = \begin{bmatrix} 2 & 2 \\ 3 & 2 \\ 1 & 0 \\ 3 & 5 \\ 2 & 3 \end{bmatrix}$$

4. (GraphSAGE) (10p) Consider an undirected graph G of five nodes A, B, C, D, and E given in the following figure. Each node has initial features that are the numbers standing next to it (i.e., the initial feature of node 'A' is $h_A^{(0)} = 0.4$). According to GraphSAGE model with an AGGREGATE is a MEAN function, the feature of a node i at layer k can be updated as:



$$h_{N(i)}^{(k)} = \mathrm{AGGREGATE}\left(\left\{h_u^{(k-1)}, \forall u \in N(i)\right\}\right)$$
$$h_i^{(k)} = \mathrm{ReLU}\left(h_i^{(k-1)} || h_{N(i)}^{(k)}\right)$$

where $||$ is a concatenation, $\mathrm{ReLU}(x) = \max(0, x)$, *N(i)* is the neighbour nodes of node *i.*

a) (4p) What is the main idea behind the GraphSAGE model, and how does it differ from traditional GCNs?
GraphSAGE learns node embeddings by sampling and aggregating features from a node's neighbors, enabling inductive learning for unseen nodes, unlike traditional GCNs that require the entire graph during training.

b) (3p) Calculate the feature of each node at $k = 1$.

c) (3p) Calculate a graph-level embedding h_G by using a 'Mean' global pooling when $k = 1$.

Ans:

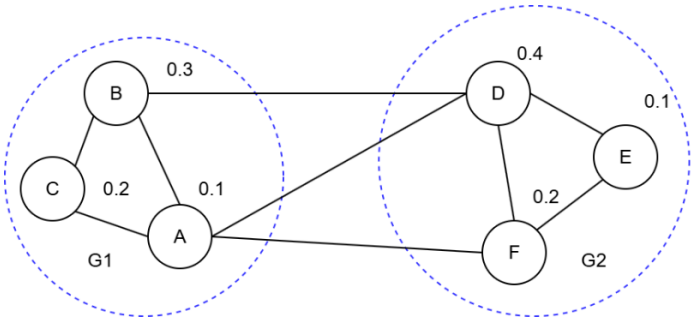
$$h_{N(i)}^{(k)} = \mathrm{AGGREGATE}\left(\left\{h_u^{(k-1)}, \forall u \in N(i)\right\}\right)$$

$$h_{N(A)}^{(1)} = MEAN(h_C, h_D, h_E) = MEAN(0.3, 0.2, 0.1) = 0.2$$
$$h_{N(B)}^{(1)} = MEAN(h_C, h_E) = MEAN(0.3, 0.1) = 0.2$$
$$h_{N(C)}^{(1)} = MEAN(h_A, h_B) = MEAN(0.4, 0.1) = 0.25$$
$$h_{N(D)}^{(1)} = MEAN(h_A, h_E) = MEAN(0.4, 0.1) = 0.25$$
$$h_{N(E)}^{(1)} = MEAN(h_A, h_B, h_D) = MEAN(0.4, 0.2) = \frac{0.7}{3}$$

$$h_i^{(k)} = \text{ReLU}\left(h_i^{(k-1)} || h_{N(i)}^{(k)}\right)$$
$$h_A^{(1)} = \text{ReLU}\left(h_A^{(0)} || h_{N(A)}^{(1)}\right) = Max(0, [0.4,0.2]) = [0.4,0.2]$$
$$h_B^{(1)} = \text{ReLU}\left(h_B^{(0)} || h_{N(B)}^{(1)}\right) = Max(0, [0.1,0.2]) = [0.1,0.2]$$
$$h_C^{(1)} = \text{ReLU}\left(h_C^{(0)} || h_{N(C)}^{(1)}\right) = Max(0, [0.3,0.25]) = [0.3,0.25]$$
$$h_D^{(1)} = \text{ReLU}\left(h_D^{(0)} || h_{N(D)}^{(1)}\right) = Max(0, [0.2,0.25]) = [0.2,0.25]$$
$$h_E^{(1)} = \text{ReLU}\left(h_E^{(0)} || h_{N(E)}^{(1)}\right) = Max(0, [0.1,0.3]) = [0.1,0.7/3]$$

$$h_G = MEAN\left(h_A^{(1)}, h_B^{(1)}, h_C^{(1)}, h_D^{(1)}, h_E^{(1)}\right) = [0.022, 0.227]$$

5. (ClusterGCN) (10p) Consider an undirected graph G of six nodes A, B, C, D, E and F given in the following figure. The graph G contains two cluster G_1 and G_2. Each node has initial features that are the numbers standing next to it. According to ClusterGCN model, the feature of a node i at layer k can be updated as:



$$h_{N(i)}^{(k)} = \text{MEAN}\left(\left\{h_u^{(k-1)}, \forall u \in N(i), G_u = G_i\right\}\right)$$
$$h_i^{(k)} = \text{ReLU}\left(h_i^{(k-1)} || h_{N(i)}^{(k)}\right)$$
where || is a concatenation.

a. (3p) What is mini-batch training?
Mini-batch training means training on graph clusters (subgraphs) instead of the full graph, allowing efficient learning on large graphs while preserving local connectivity within each batch.

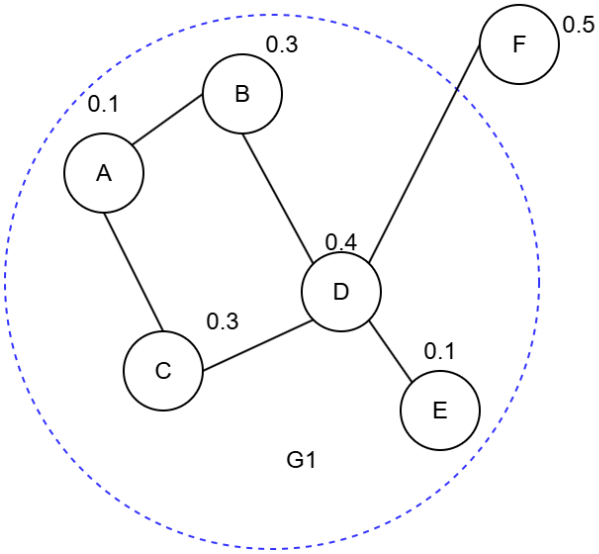
b. (7p) Calculate the output representations of all nodes at layer $k = 1$.

Ans:

$$h_{N(i)}^{(k)} = \text{MEAN}\left(\left\{h_u^{(k-1)}, \forall u \in N(i)\right\}, G_u = G_i\right)$$
$$h_{N(A)}^{(1)} = MEAN(h_B, h_C) = MEAN(0.3, 0.2) = 0.25$$
$$h_{N(B)}^{(1)} = MEAN(h_A, h_C) = MEAN(0.1,0.2) = 0.15$$
$$h_{N(C)}^{(1)} = MEAN(h_A, h_B) = MEAN(0.3,0.1) = 0.2$$
$$h_{N(D)}^{(1)} = MEAN(h_E, h_F) = MEAN(0.1,0.2) = 0.15$$
$$h_{N(E)}^{(1)} = MEAN(h_D, h_F) = MEAN(0.4,0.2) = 0.3$$
$$h_{N(F)}^{(1)} = MEAN(h_D, h_E) = MEAN(0.4,0.1) = 0.25$$

$$h_i^{(k)} = \text{ReLU}\left(h_i^{(k-1)} || h_{N(i)}^{(k)}\right)$$
$$h_A^{(1)} = \text{ReLU}\left(h_A^{(0)} || h_{N(A)}^{(1)}\right) = Max(0, [0.1,0.25]) = [0.1,0.25]$$
$$h_B^{(1)} = \text{ReLU}\left(h_B^{(0)} || h_{N(B)}^{(1)}\right) = Max(0, [0.3,0.15]) = [0.3,0.15]$$
$$h_C^{(1)} = \text{ReLU}\left(h_C^{(0)} || h_{N(C)}^{(1)}\right) = Max(0, [0.2,0.25]) = [0.2,0.2]$$
$$h_D^{(1)} = \text{ReLU}\left(h_D^{(0)} || h_{N(D)}^{(1)}\right) = Max(0, [0.4,0.15]) = [0.4,0.15]$$
$$h_E^{(1)} = \text{ReLU}\left(h_E^{(0)} || h_{N(E)}^{(1)}\right) = Max(0, [0.1,0.3]) = [0.1,0.3]$$
$$h_F^{(1)} = \text{ReLU}\left(h_F^{(0)} || h_{N(F)}^{(1)}\right) = Max(0, [0.2,0.25]) = [0.2,0.25]$$

6. (GraphSAINT) (10p) Consider an undirected graph G of six nodes A, B, C, D, E and F given in the following figure. The graph G has subgraph sampling G_1. Each node has initial features that are the numbers standing next to it. According to GraphSAINT model, the feature of a node i at layer k can be updated as:



$$h_{N(i)}^{(k)} = \text{MEAN}\left(\left\{h_u^{(k-1)}, \forall u \in N(i), G_u = G_i\right\}\right)$$
$$h_i^{(k)} = \text{ReLU}\left(h_i^{(k-1)} || h_{N(i)}^{(k)}\right)$$
where $||$ is a concatenation.

a. (3p) How does GraphSAINT's sampling strategy differ from node-wise or edge-wise sampling methods like GraphSAGE?

GraphSAINT samples entire subgraphs for each mini-batch instead of individual nodes or edges, preserving more structural and neighborhood information while keeping training efficient

b. (7p) Calculate the output representations of nodes A, B, C, D, and E at layer $k = 1$.

Ans:

$$h_{N(i)}^{(k)} = \text{MEAN}\left(\{h_u^{(k-1)}, \forall u \in N(i)\}, G_u = G_i\right)$$
$$h_{N(A)}^{(1)} = MEAN(h_B, h_C) = MEAN(0.3, 0.3) = 0.3$$
$$h_{N(B)}^{(1)} = MEAN(h_A, h_D) = MEAN(0.1, 0.4) = 0.25$$
$$h_{N(C)}^{(1)} = MEAN(h_A, h_D) = MEAN(0.1, 0.4) = 0.25$$
$$h_{N(D)}^{(1)} = MEAN(h_B, h_C, h_E) = MEAN(0.3, 0.3, 0.1) = 0.23$$
$$h_{N(E)}^{(1)} = MEAN(h_D) = MEAN(0.4) = 0.4$$

– –

$$h_i^{(k)} = \text{ReLU}\left(h_i^{(k-1)} || h_{N(i)}^{(k)}\right)$$
$$h_A^{(1)} = \text{ReLU}\left(h_A^{(0)} || h_{N(A)}^{(1)}\right) = Max(0, [0.1, 0.3]) = [0.1, 0.3]$$
$$h_B^{(1)} = \text{ReLU}\left(h_B^{(0)} || h_{N(B)}^{(1)}\right) = Max(0, [0.3, 0.25]) = [0.3, 0.25]$$
$$h_C^{(1)} = \text{ReLU}\left(h_C^{(0)} || h_{N(C)}^{(1)}\right) = Max(0, [0.3, 0.25]) = [0.3, 0.25]$$
$$h_D^{(1)} = \text{ReLU}\left(h_D^{(0)} || h_{N(D)}^{(1)}\right) = Max(0, [0.4, 0.23]) = [0.4, 0.23]$$
$$h_E^{(1)} = \text{ReLU}\left(h_E^{(0)} || h_{N(E)}^{(1)}\right) = Max(0, [0.1, 0.4]) = [0.1, 0.4]$$

7. (JK Network) (10p) Given a graph with an adjacency matrix A and initial node feature matrix $H^{(0)}$ as follows:

$$A = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \qquad H^{(0)} = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 1 & 2 \\ -1 & 1 \\ 2 & 3 \end{bmatrix}$$

Assume that the output of an JK network model of all nodes at layer (k) can be calculated as:

$$H^{(k)} = \max\left(\sigma(\tilde{A} \cdot H^{(0)}), \sigma(\tilde{A} \cdot H^{(1)}), \ldots, \sigma(\tilde{A} \cdot H^{(k-1)})\right)$$

where $H^{(k)}$ denotes the output at layer k, $\tilde{A}$ is the normalized matrix ($\tilde{A} = D^{-1}A$), σ is a ReLU function $\text{ReLU}(x) = \max(0, x)$.

a) (2p) Explain how Jumping Knowledge Networks help mitigate the **over-smoothing problem** in deep GCNs.

JK networks mitigate over-smoothing by combining representations from multiple layers, allowing each node to adaptively use information from both shallow and deep layers, preserving feature diversity

b) (4p) Calculate $\tilde{A}$.

c) (4p) Calculate the output representations at layer $k = 2$.

Ans:

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}, D^{-1} = \begin{bmatrix} 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1/2 \end{bmatrix}$$

$$\tilde{A} = D^{-1}A = \begin{bmatrix} 1/3 & 1/2 & 0 & 1/3 & 0 \\ 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 1/2 & 0 & 1/2 & 0 \end{bmatrix}$$

$$H^{(k)} = \max\left(\sigma\left(\tilde{A} \cdot H^{(0)}\right), \sigma\left(\tilde{A} \cdot H^{(1)}\right), \ldots, \sigma\left(\tilde{A} \cdot H^{(k)}\right)\right)$$

- Layer 1:

$$H^1 = \sigma(AH^0) = \text{ReLU}\left(\begin{bmatrix} 1/3 & 1/2 & 0 & 1/3 & 0 \\ 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 1/2 & 0 & 1/2 & 0 \end{bmatrix} \times \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 1 & 2 \\ -1 & 1 \\ 2 & 3 \end{bmatrix}\right)$$

$$= \text{ReLU}\left(\begin{bmatrix} -1/6 & -1/6 \\ 3/2 & 5/2 \\ 0 & 0 \\ 1/2 & 2 \\ 0 & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & 0 \\ 3/2 & 5/2 \\ 0 & 0 \\ 1/2 & 2 \\ 0 & 0 \end{bmatrix}$$

- Layer 2:

$$H^{(2)} = \max\left(\sigma\left(\tilde{A} \cdot H^{(0)}\right), \sigma\left(\tilde{A} \cdot H^{(1)}\right) \right)$$

$$= \max\left(\sigma\left(\begin{bmatrix} \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 1 & 2 \\ -1 & 1 \\ 2 & 3 \end{bmatrix} \right), \sigma\left(\begin{bmatrix} \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ \frac{3}{2} & \frac{5}{2} \\ 0 & 0 \\ \frac{1}{2} & 2 \\ 0 & 0 \end{bmatrix} \right) \right)$$

$$= \max\left(\sigma\begin{bmatrix} -1/6 & -1/6 \\ 3/2 & 5/2 \\ 0 & 0 \\ 1/2 & 2 \\ 0 & 0 \end{bmatrix}, \sigma\begin{bmatrix} \frac{11}{12} & \frac{23}{12} \\ 0 & 0 \\ 1 & \frac{9}{4} \\ \frac{1}{4} & 1 \\ 1 & \frac{9}{4} \end{bmatrix} \right) = \begin{bmatrix} \frac{11}{12} & \frac{23}{12} \\ 3/2 & 5/2 \\ 1 & \frac{9}{4} \\ \frac{1}{2} & 2 \\ 1 & \frac{9}{4} \end{bmatrix}$$

8. (GCNII) (10p) Given a graph with an adjacency matrix A and initial node feature matrix $H^{(0)}$ as follows:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \qquad H^{(0)} = \begin{bmatrix} 3 \\ -1 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

Assume that the output of an GCNII model of all nodes at layer (k) can be calculated as:

$$H^{(k)} = \sigma\left[\left((1-\beta)I_n\right) \cdot \left((1-\alpha)\tilde{A} \cdot H^{(k-1)} + \alpha H^{(0)}\right)\right]$$

where $H^{(k)}$ denotes the output at layer k, $\tilde{A}$ is the normalized matrix ($\tilde{A} = D^{-1}A$), I_n is the identity matrix, $\alpha = \beta = 0.5$, σ is a ReLU function $\mathrm{ReLU}(x) = \max(0, x)$.

a) (2p) What are the two major components introduced in GCNII to overcome the depth limitation problem in GCNs?

GCNII introduces initial residual connections and identity mapping to preserve original features and enable deeper, more stable GCN training

b) (4p) Calculate $\tilde{A}$.

c) (4p) Calculate the output representations at layer $k = 1$

Ans:

$$D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}, D^{-1} = \begin{bmatrix} 1/2 & 0 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 1/5 & 0 \\ 0 & 0 & 0 & 0 & 1/2 \end{bmatrix}$$

$$\tilde{A} = D^{-1}A = \begin{bmatrix} 1/2 & 0 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 1/5 & 0 \\ 0 & 0 & 0 & 0 & 1/2 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 \\ 0.25 & 0.25 & 0 & 0.25 & 0.25 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0 & 0.5 & 0 & 0 & 0.5 \end{bmatrix}$$

- $H^{(k)} = \sigma\left[\left((1-\beta)I_n\right)\left((1-\alpha)\tilde{A} \cdot H^{(k-1)} + \alpha H^{(0)}\right)\right]$

$$\tilde{A} \cdot H^{(k-1)} = \begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 \\ 0.25 & 0.25 & 0 & 0.25 & 0.25 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0 & 0.5 & 0 & 0 & 0.5 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ -1 \\ 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1.75 \\ 2 \\ 1.6 \\ 1 \end{bmatrix}$$

$$I_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\left[\left((1-\beta)I_n\right)\left((1-\alpha)\tilde{A} \cdot H^{(k-1)} + \alpha H^{(0)}\right)\right]$$

$$= \left(0.5 \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right) \cdot \left(0.5 \begin{bmatrix} 0 \\ 1.75 \\ 2 \\ 1.6 \\ 1 \end{bmatrix} + 0.5 \begin{bmatrix} 3 \\ -1 \\ 1 \\ 2 \\ 3 \end{bmatrix}\right) = \begin{bmatrix} 1.5 \\ 0.375 \\ 1.5 \\ 1.8 \\ 2 \end{bmatrix}$$

$$H^{(k)} = \sigma\left(\begin{bmatrix} 1.5 \\ 0.375 \\ 1.5 \\ 1.8 \\ 2 \end{bmatrix}\right) = \begin{bmatrix} 1.5 \\ 0.375 \\ 1.5 \\ 1.8 \\ 2 \end{bmatrix}$$

9. (DeepGCN) (10p) Given a graph with an adjacency matrix A and initial node feature matrix $H^{(0)}$ as follows:

$$A = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad H^{(0)} = \begin{bmatrix} 0 & -2 \\ -1 & 3 \\ 4 & -2 \\ 0 & -5 \end{bmatrix}$$

Assume that the hidden layer of an DeepGCNs model of all nodes at layer (k) can be calculated as:

$$H^{(k)} = \sigma\left(A \cdot H^{(k-1)}\right) + H^{(k-1)},$$

where $H^{(k)}$ denotes the output at layer k, σ is a ReLU function $\mathrm{ReLU}(x) = \max(0, x)$.

a. (3p) How does DeepGCN mitigate the **over-smoothing problem** as depth increases?
DeepGCN mitigates over-smoothing by using residual connections, normalization layers, and dilated/dynamic aggregation, which help preserve feature diversity and stable information flow in deep layers

b. (7p) Calculate the output of the GCN model at layer $k = 2$.

Ans:

$$H^{(1)} = \sigma\left(AH^{(0)}\right) + H^{(0)} = \mathrm{ReLU}\left(\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & -2 \\ -1 & 3 \\ 4 & -2 \\ 0 & -5 \end{bmatrix}\right) + \begin{bmatrix} 0 & -2 \\ -1 & 3 \\ 4 & -2 \\ 0 & -5 \end{bmatrix}$$

$$= \mathrm{ReLU}\left(\begin{bmatrix} 7 & -7 \\ -1 & 3 \\ 3 & -7 \\ 3 & 1 \end{bmatrix}\right) + \begin{bmatrix} 0 & -2 \\ -1 & 3 \\ 4 & -2 \\ 0 & -5 \end{bmatrix} = \begin{bmatrix} 7 & 0 \\ 0 & 3 \\ 3 & 0 \\ 3 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -2 \\ -1 & 3 \\ 4 & -2 \\ 0 & -5 \end{bmatrix}$$

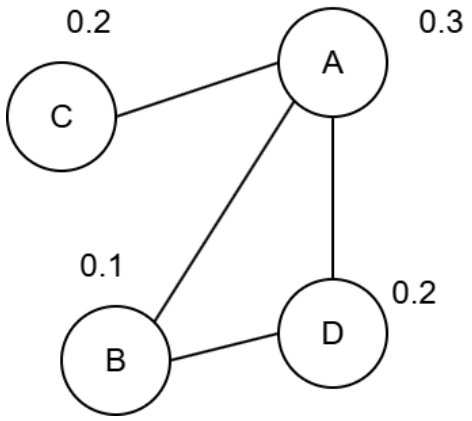
$$= \begin{bmatrix} 7 & -2 \\ -2 & 6 \\ 7 & -2 \\ 3 & -4 \end{bmatrix}$$

$$H^{(2)} = \sigma\left(AH^{(1)}\right) + H^{(1)} = \mathrm{ReLU}\left(\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 7 & -2 \\ -2 & 6 \\ 7 & -2 \\ 3 & -4 \end{bmatrix}\right) + \begin{bmatrix} 7 & -2 \\ -2 & 6 \\ 7 & -2 \\ 3 & -4 \end{bmatrix}$$

$$= \mathrm{ReLU}\left(\begin{bmatrix} 10 & -6 \\ -2 & 6 \\ 10 & -6 \\ 5 & 4 \end{bmatrix}\right) + \begin{bmatrix} 7 & -2 \\ -2 & 6 \\ 7 & -2 \\ 3 & -4 \end{bmatrix} = \begin{bmatrix} 10 & 0 \\ 0 & 6 \\ 10 & 0 \\ 5 & 4 \end{bmatrix} + \begin{bmatrix} 7 & -2 \\ -2 & 6 \\ 7 & -2 \\ 3 & -4 \end{bmatrix}$$

$$= \begin{bmatrix} 17 & -2 \\ -2 & 12 \\ 17 & -2 \\ 8 & 0 \end{bmatrix}$$

10. (GAT) (10p) Consider an undirected graph G of four nodes A, B, C, and D given in the following figure. Each node has initial features that are the numbers standing next to it (i.e., the initial feature of node 'A' is $h_A^{(0)} = 0.3$). According to GAT model, the weight matrix W is randomly initialized as [0.5]. The feature of node 'i' at layer (k) can be updated as:



$$h_i^{(k)} = \sigma\left(\sum_{m \in N(i)} \alpha_{im} W h_m\right)$$

Where $\alpha_{im} = \frac{e_{im}}{\sum_{k \in N(i)} e_{ik}}$ and

$e_{im} = \sigma\big(\text{MEAN}(Wh_i, Wh_m)\big)$

σ is a ReLU function $\text{ReLU}(x) = \max(0, x)$.

a) (2p) What problem in traditional GCNs does GAT aim to solve?
GAT solves the problem of equal weighting in traditional GCNs, it lets the model learn different importance weights for each neighbor instead of treating all neighbors equally

b) (4p) Calculate the attention coefficients e_{AB}, e_{AC}, and e_{AD}

c) (4p) Calculate the feature of node 'A' at $k = 1$.

Ans:

$$h_i^{(k)} = \sigma\left(\sum_{m \in N(i)} \alpha_{im} W h_m\right)$$

where: $\alpha_{im} = \frac{e_{im}}{\sum_{k \in N(i)} e_{ik}}$, and $e_{im} = \sigma\big(\text{MEAN}(Wh_i, Wh_m)\big)$

$$e_{AC} = \sigma\big(\text{MEAN}(Wh_A, Wh_C)\big) = \sigma\big(\text{MEAN}(0.5 * 0.3, 0.5 * 0.2)\big)$$
$$= \sigma\big(\text{MEAN}(0.15, 0.1)\big) = 0.125$$
$$e_{AB} = \sigma\big(\text{MEAN}(Wh_A, Wh_B)\big) = \sigma\big(\text{MEAN}(0.5 * 0.3, 0.5 * 0.1)\big)$$
$$= \sigma\big(\text{MEAN}(0.15, 0.05)\big) = 0.1$$
$$e_{AD} = \sigma\big(\text{MEAN}(Wh_A, Wh_D)\big) = \sigma\big(\text{MEAN}(0.5 * 0.3, 0.5 * 0.2)\big)$$
$$= \sigma\big(\text{MEAN}(0.15, 0.1)\big) = 0.125$$

$$\alpha_{AB} = \frac{e_{AB}}{e_{AB} + e_{AC} + e_{AD}} = \frac{0.1}{0.125 + 0.1 + 0.125} = 0.286$$
$$\alpha_{AC} = \alpha_{AD} = 0.357$$

$$h_i^{(k)} = \sigma\left(\sum_{m \in N(i)} \alpha_{im} W h_m\right) = \sigma(\alpha_{AB} W h_B + \alpha_{AC} W h_C + \alpha_{AD} W h_D)$$
$$= \sigma(0.286 * 0.5 * 0.1 + 0.357 * 0.5 * 0.2 + 0.357 * 0.5 * 0.2) = 0.0857$$

11. (GATv2) (5p)

a. (2p) What is the key limitation of GAT does GATv2 aim to fix?
GATv2 fixes GAT's limitation of limited expressiveness by making the attention mechanism dynamic and nonlinear, allowing it to better capture complex relationships between node pairs.

b. (3p) Calculate attention score e_{ij} to compare different between GAT and GATv2, given

GAT $e_{ij} = LeakyReLU(a(W\overrightarrow{h_i}, W\overrightarrow{h_j}))$

GATv2 $e_{ij} = aLeakyReLU(W[\overrightarrow{h_i}, \overrightarrow{h_j}])$

$$\overrightarrow{h_i} = \begin{bmatrix}1\\2\end{bmatrix}, \overrightarrow{h_j} = \begin{bmatrix}1\\-1\end{bmatrix}, W = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}, a = [1,1,1,1]$$

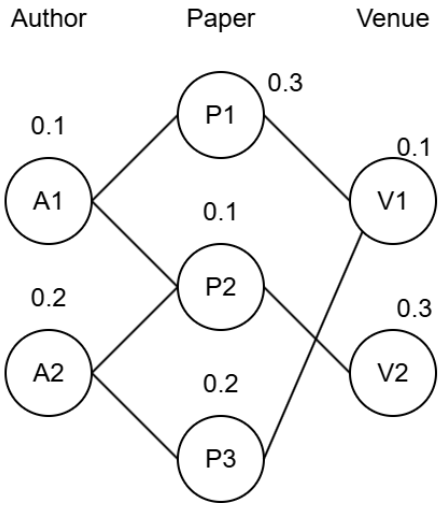
LeakyReLU activation, where $\begin{cases} x = x, if\ x > 0 \\ x = 0.02x, if\ x < 0 \end{cases}$

Ans:

GAT $e_{ij} = LeakyReLU\left(a(W\overrightarrow{h_i}, W\overrightarrow{h_j})\right) = LeakyReLU([1,1,1,1][1,2,1,-1] = 3$

GATv2 $e_{ij} = aLeakyReLU(W[\overrightarrow{h_i}, \overrightarrow{h_j}]) = [1,1,1,1]LeakyReLU([1,2,1,-1]) = [1,1,1,1][1,2,1,-0.02] = 3.98$

12. (HAN) (5p) Consider a heterogeneous graph given in the following figure. There are three types of nodes in the academic network: Author (A), Paper (P), and Venue (V). Each node has initial features that are the numbers standing next to it (i.e., the initial feature of node 'A_1' is $h_{A_1}^{(0)} = 0.2$). According to HAN model, the weight matrix W is randomly initialized as [0.5]. The feature of node 'i' at layer (k) can be updated as:



$$h_i^{(k)} = \sigma\left(\sum_{m \in N(i)} \alpha_{im}^{\Phi} W h_m\right)$$

Where $\alpha_{im}^{\Phi} = \frac{e_{im}^{\Phi}}{\sum_{k \in N^{\Phi}(i)} e_{ik}^{\Phi}}$ and

$$e_{im}^{\Phi} = \sigma\left(\text{MEAN}\left(Wh_i^{\Phi}, Wh_m^{\Phi}\right)\right)$$

σ is a ReLU function $\text{ReLU}(x) = \max(0, x)$.

a) (1p) What is heterogeneous graph?
   Heterogeneous graph is a graph that has different types of node.
b) (2p) List all the meta-path PAP and PVP. Calculate the attention coefficients of each meta-path PAP and PVP.
c) (2p) Calculate the feature of node 'P_1' at $k = 1$.

Ans:

List meta-path:
- PAP: $P_1A_1P_2, P_2A_2P_3$.
- PVP: $P_1V_1P_3$.

$$h_i^{(k)} = \sigma\left(\sum_{m \in N(i)} \alpha_{im}^{\Phi} W h_m\right)$$

Where $\alpha_{im}^{\Phi} = \frac{e_{im}^{\Phi}}{\sum_{k \in N^{\Phi}(i)} e_{ik}^{\Phi}}$ and $e_{im}^{\Phi} = \sigma\left(\text{MEAN}\left(Wh_i^{\Phi}, Wh_m^{\Phi}\right)\right)$

PAP:

$$e_{P_1P_2}^{\Phi} = \sigma\left(\text{MEAN}\left(Wh_{P_1}, Wh_{P_2}\right)\right) = \sigma\left(\text{MEAN}(0.5 * 0.3, 0.5 * 0.1)\right)$$
$$= \sigma\left(\text{MEAN}(0.15, 0.05)\right) = 0.1$$
$$e_{P_2P_3}^{\Phi} = \sigma\left(\text{MEAN}\left(Wh_{P_2}, Wh_{P_3}\right)\right) = \sigma\left(\text{MEAN}(0.5 * 0.1, 0.5 * 0.2)\right)$$
$$= \sigma\left(\text{MEAN}(0.05, 0.1)\right) = 0.075$$

$$e_{P_1P_3}^{\Phi} = \sigma\left(\text{MEAN}\left(Wh_{P_1}, Wh_{P_3}\right)\right) = \sigma\left(\text{MEAN}(0.5 * 0.3, 0.5 * 0.2)\right)$$
$$= \sigma\left(\text{MEAN}(0.15, 0.1)\right) = 0.125$$

$$e_{P_1A_1}^{\Phi} = \sigma\left(\text{MEAN}\left(Wh_{P_1}, Wh_{A_1}\right)\right) = \sigma\left(\text{MEAN}(0.5 * 0.3, 0.5 * 0.1)\right)$$
$$= \sigma\left(\text{MEAN}(0.15, 0.05)\right) = 0.1$$

$$e_{A_1P_2}^{\Phi} = \sigma\left(\text{MEAN}\left(Wh_{A_1}, Wh_{P_2}\right)\right) = \sigma\left(\text{MEAN}(0.5 * 0.1, 0.5 * 0.1)\right)$$
$$= \sigma\left(\text{MEAN}(0.05, 0.05)\right) = 0.05$$

– –

$$\alpha_{P_1P_2}^{\Phi} = \frac{e_{P_1P_2}^{\Phi}}{e_{P_1P_2}^{\Phi} + e_{P_1P_3}^{\Phi}} = \frac{0.1}{0.1 + 0.125} \approx 0.444$$
$$\alpha_{P_1P_3}^{\Phi} = \frac{e_{P_1P_3}^{\Phi}}{e_{P_1P_2}^{\Phi} + e_{P_1P_3}^{\Phi}} = \frac{0.125}{0.1 + 0.125} \approx 0.555$$
$$\alpha_{P_2P_3}^{\Phi} = 0.075$$

$$\alpha^{\Phi}_{P_1A_1} = \frac{e^{\Phi}_{P_1A_1}}{e^{\Phi}_{P_1A_1}} = 1 = \alpha^{\Phi}_{A_1P_2} = \alpha^{\Phi}_{P_1V_1} = \alpha^{\Phi}_{V_1P_3}$$

$-----------------------------------$

PAP: $P_1A_1P_2$

$$h_i^{(k)} = \sigma\left(\sum_{m\in N(i)} \alpha^{\Phi}_{im} W h_m\right) = \sigma\left(\left(\alpha^{\Phi}_{P_1A_1} W h_{A_1}\right) + \left(\alpha^{\Phi}_{A_1P_2} W h_{P_2}\right)\right)$$

$$= \sigma\big((1 * 0.5 * 0.1) + (1 * 0.5 * 0.1\big) = 0.1$$

PVP: $P_1V_1P_3$

$$h_i^{(k)} = \sigma\left(\sum_{m\in N(i)} \alpha^{\Phi}_{im} W h_m\right) = \sigma\left(\left(\alpha^{\Phi}_{P_1V_1} W h_{V_1}\right) + \left(\alpha^{\Phi}_{V_1P_3} W h_{P_3}\right)\right) =$$

$$= \sigma\big((1 * 0.5 * 0.1) + (1 * 0.5 * 0.2\big) = 0.15$$