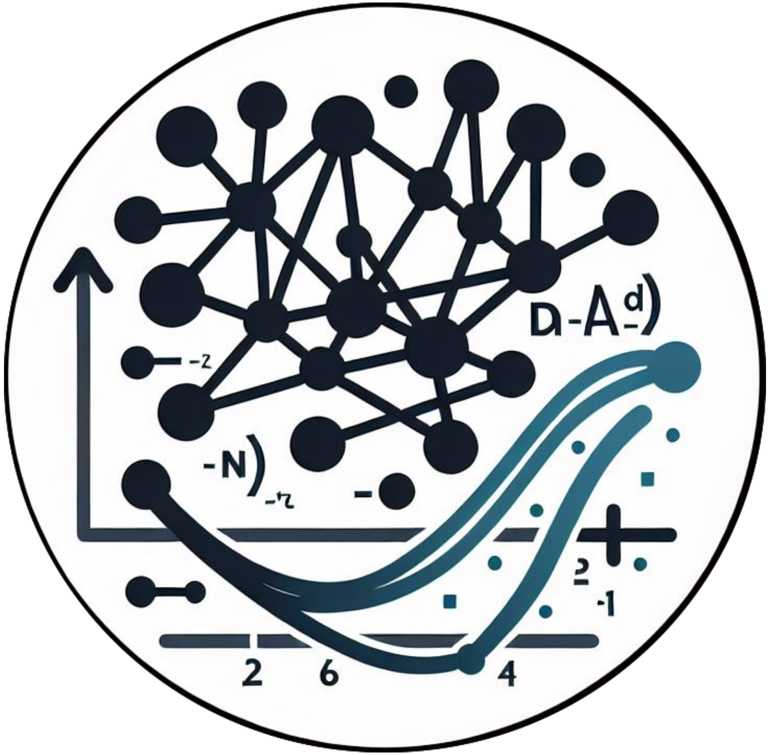
Graph Neural Nets

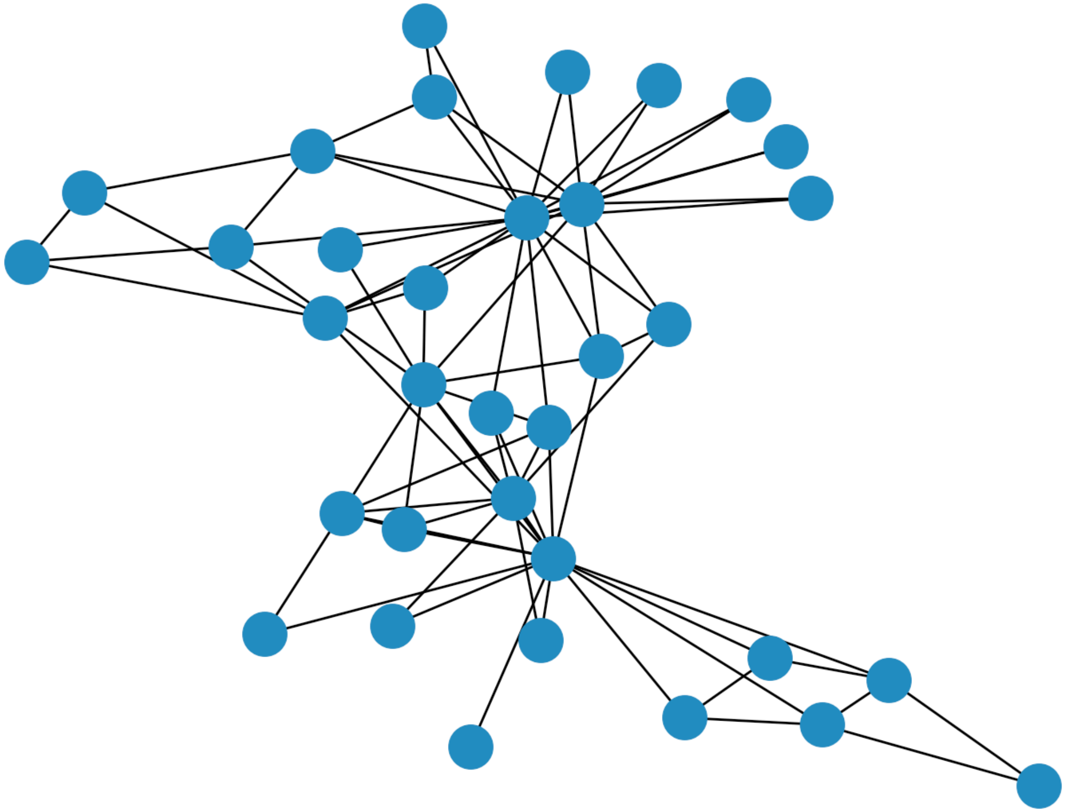


Deep Learning for Engineers

Andrew Ning

aning@byu.edu

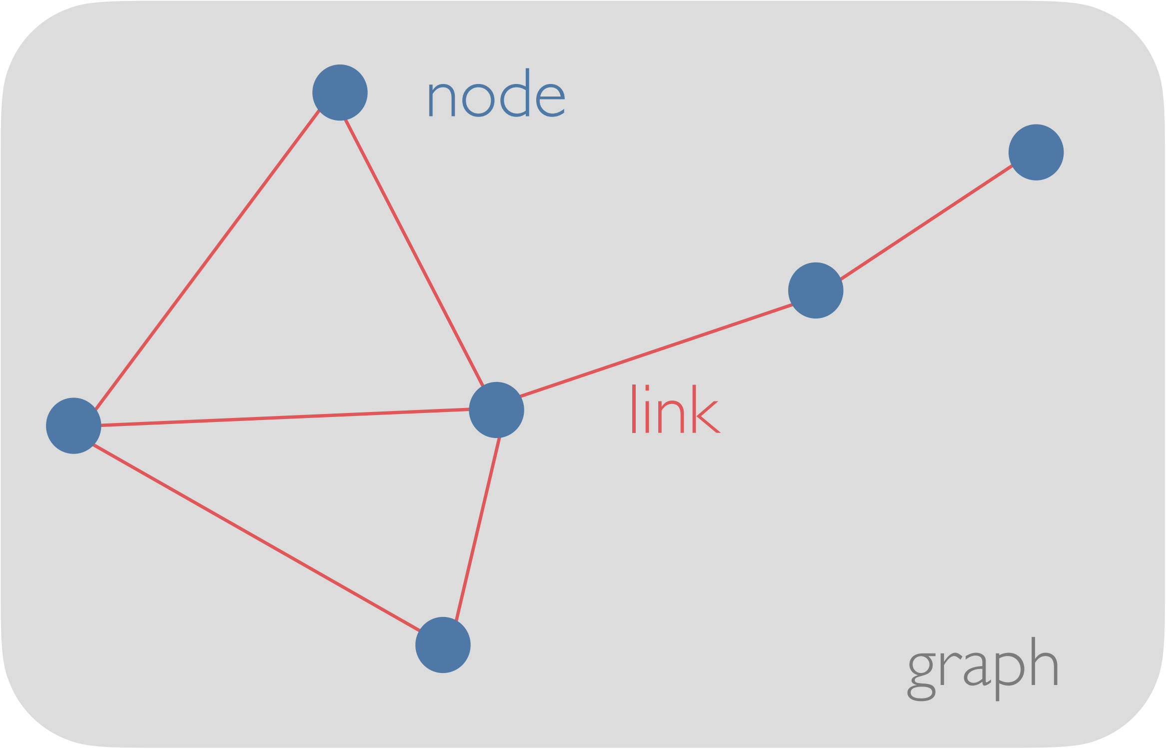
Motivation



Challenges

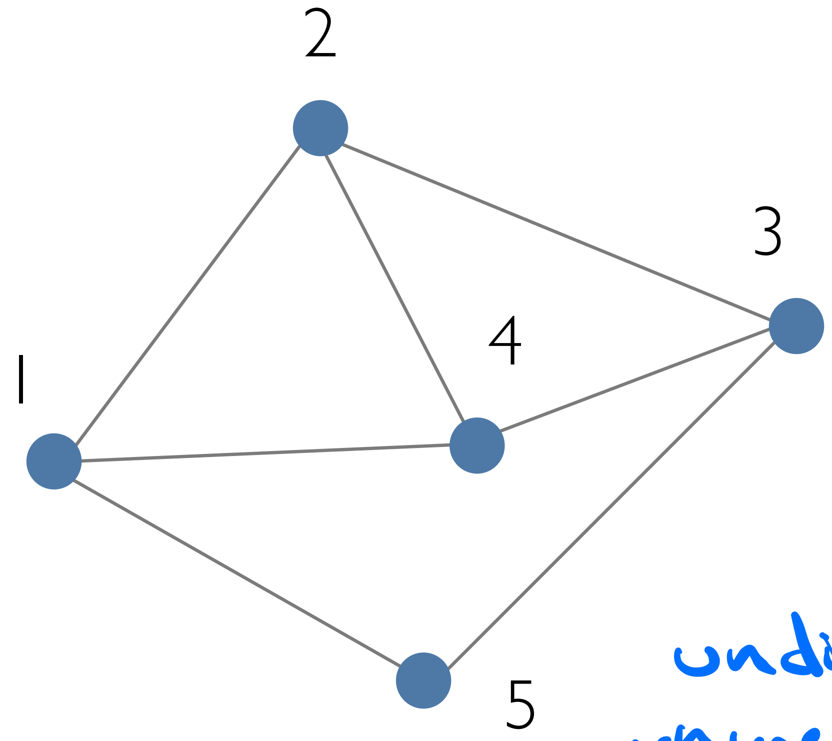
- variety of sizes
- structure is complex
- ordering

Node, link, and graph classification/regression



Adjacency matrix

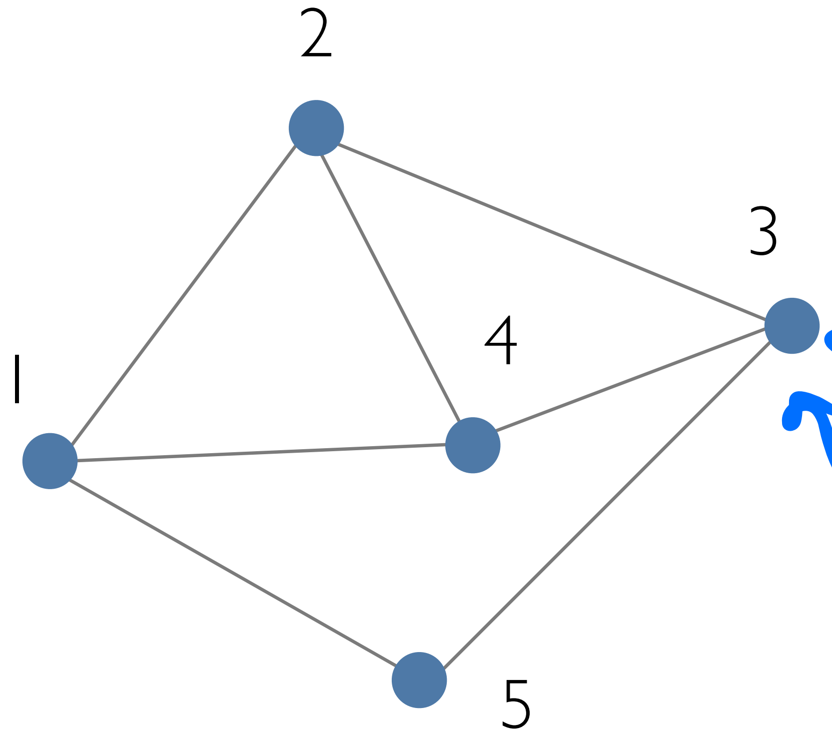
$$A_{ij} = \begin{cases} 1 & \text{connected} \\ 0 & \text{otherwise} \end{cases}$$



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

undirected
unweighted

Features and Targets/Labels



$$x_3 = \begin{bmatrix} 2.1 \\ 1.6 \\ -1.2 \\ 4.1 \end{bmatrix}$$

$$y_3 = \begin{bmatrix} 0.3 \\ -1.1 \end{bmatrix}$$

X

size $n \times d$

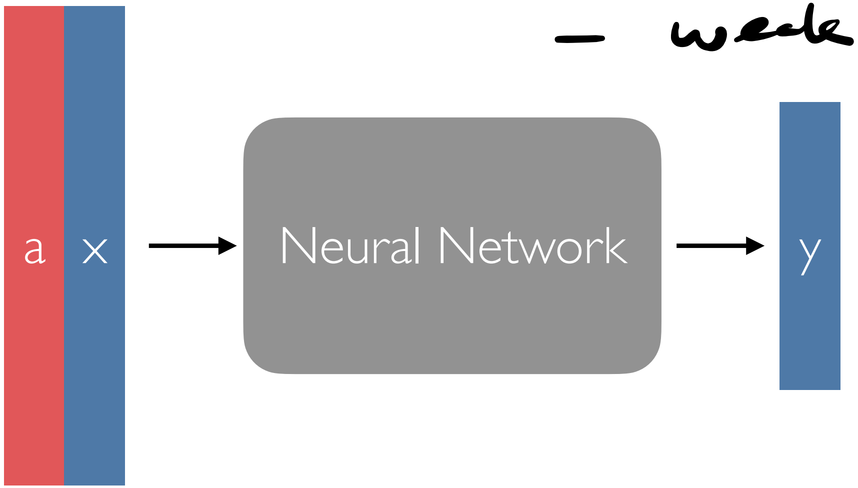
# of nodes

dimension of features

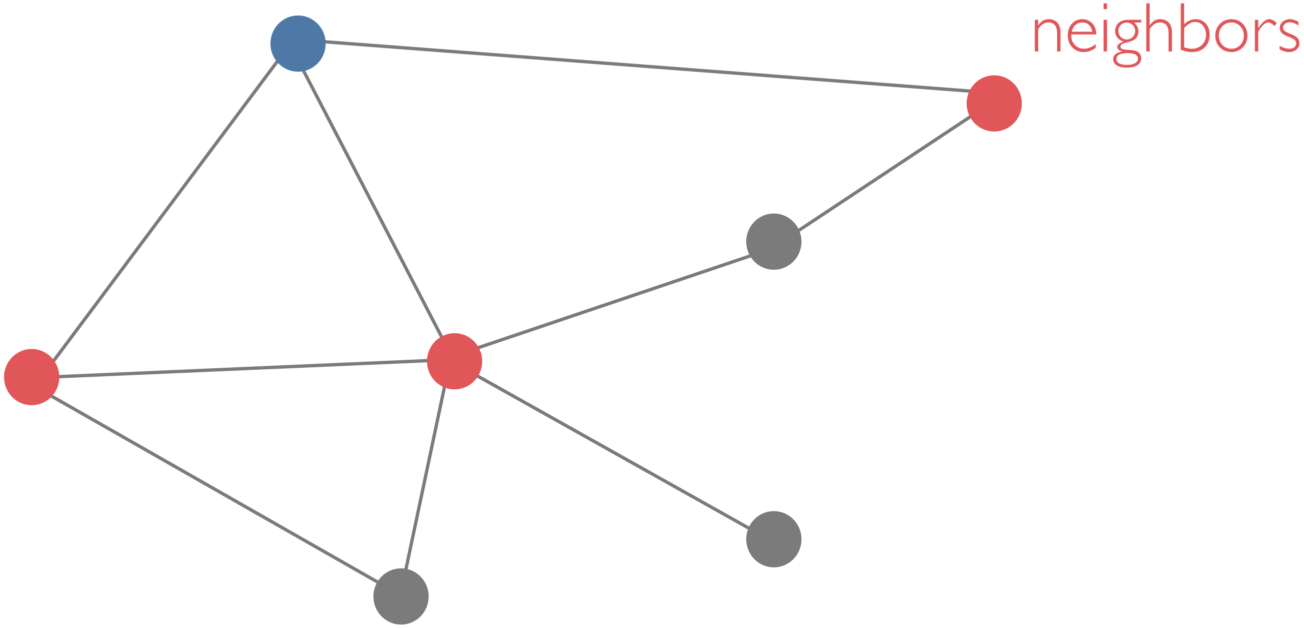
First Approach

First Approach

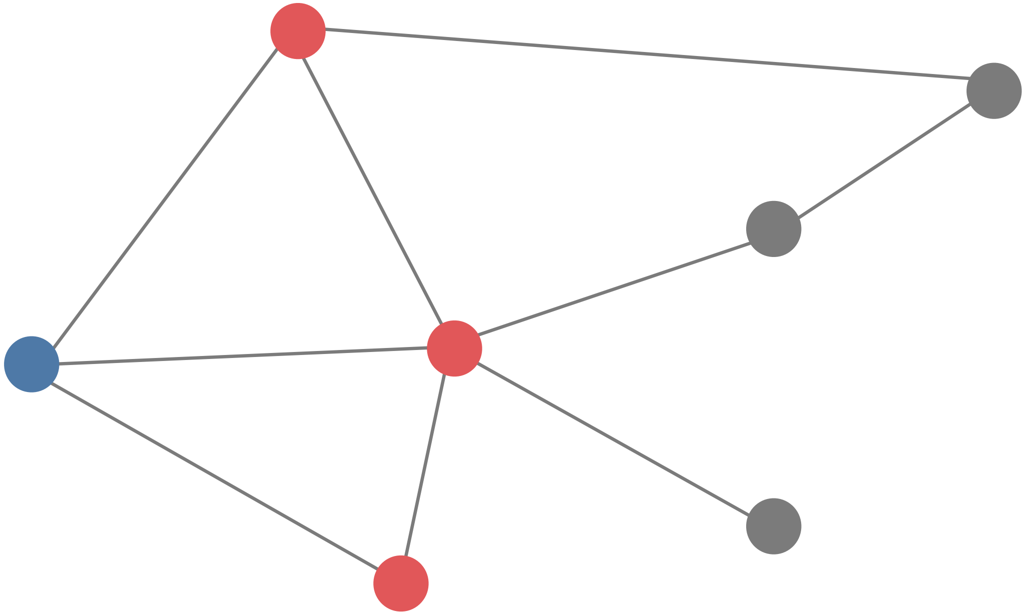
- doesn't accommodate different graph sizes/connectivity
- node ordering?
- weak passage of info
- overfitting



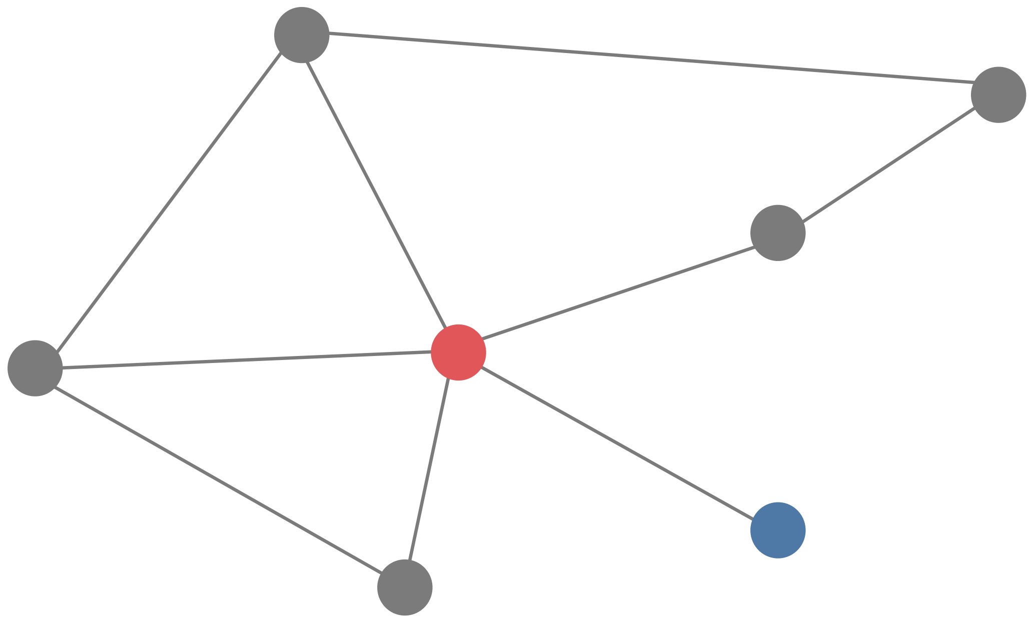
Neighbors



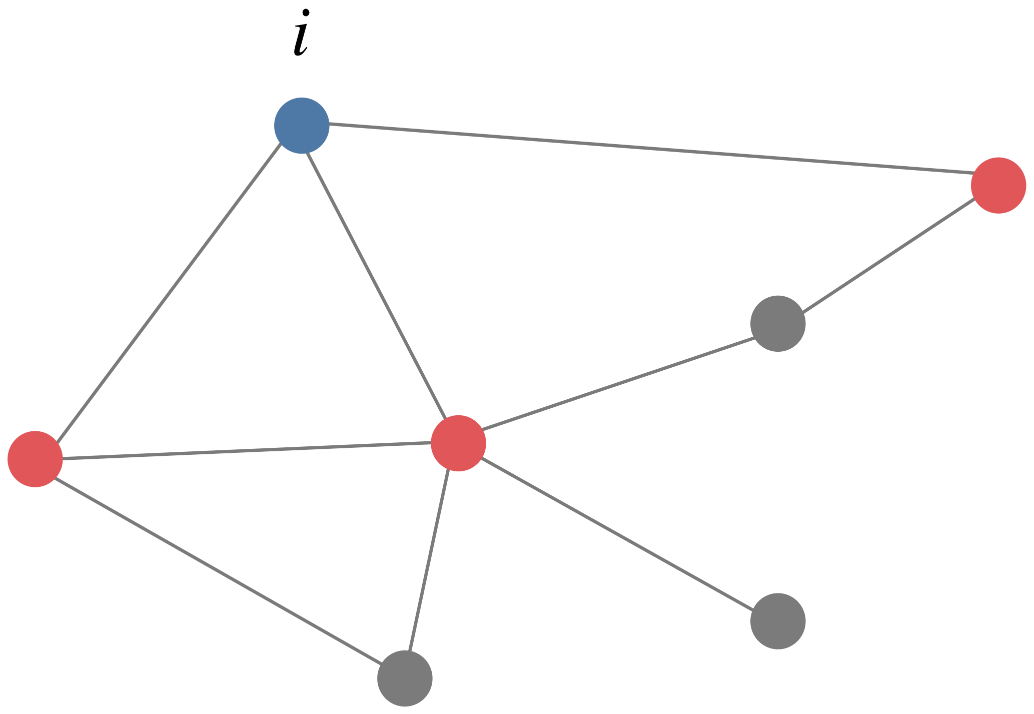
Neighbors



Neighbors



Neighbors



$$\mathcal{N}(i) = \{j : a_{ij} \in A\}$$

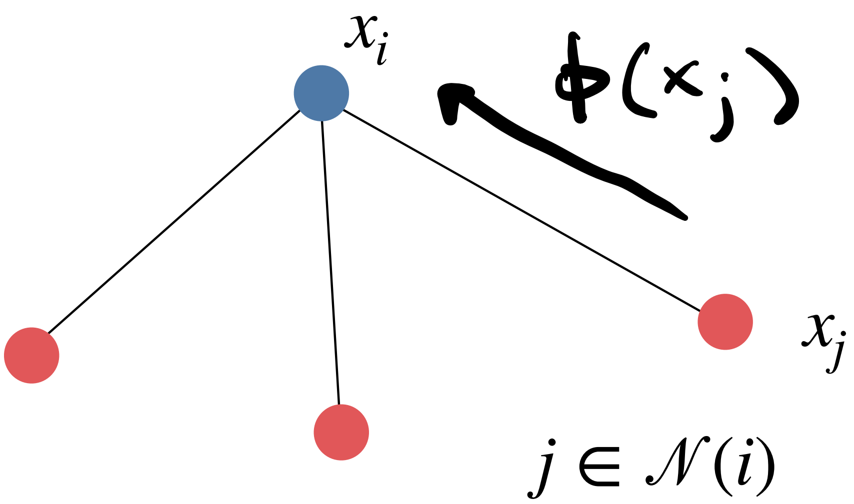
Message Passing GNN

1. each node construct a <u>message</u> for its neighbors

2. <u>aggregate</u> messages from neighbors

3. each node <u>update</u> its attributes.

Create a message

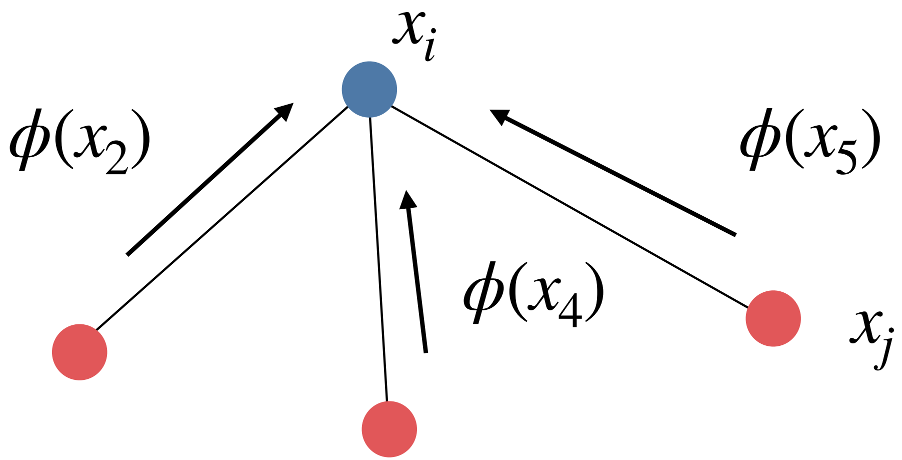


Simple: $\mathbf{W} x_j$

more general version: $\phi(x_j, x_i)$

Aggregate

order invariant



Aggregate

$$x_{agg} = \sum_{j \in \mathcal{N}(i)} \phi(x_j, x_i) \qquad \text{(sum)}$$

Aggregate

$$x_{agg} = \sum_{j \in \mathcal{N}(i)} \phi(x_j, x_i) \qquad \text{(sum)}$$

$$x_{agg} = \frac{\sum_{j \in \mathcal{N}(i)} \phi(x_j, x_i)}{|\mathcal{N}(i)|} \qquad \text{(mean)}$$

Aggregate

$$x_{agg} = \sum_{j \in \mathcal{N}(i)} \phi(x_j, x_i) \qquad \text{(sum)}$$

$$x_{agg} = \frac{\sum_{j \in \mathcal{N}(i)} \phi(x_j, x_i)}{|\mathcal{N}(i)|} \qquad \text{(mean)}$$

$$x_{agg} = \max_{j \in \mathcal{N}(i)} \phi(x_j, x_i) \qquad \text{(max or min)}$$

Aggregate

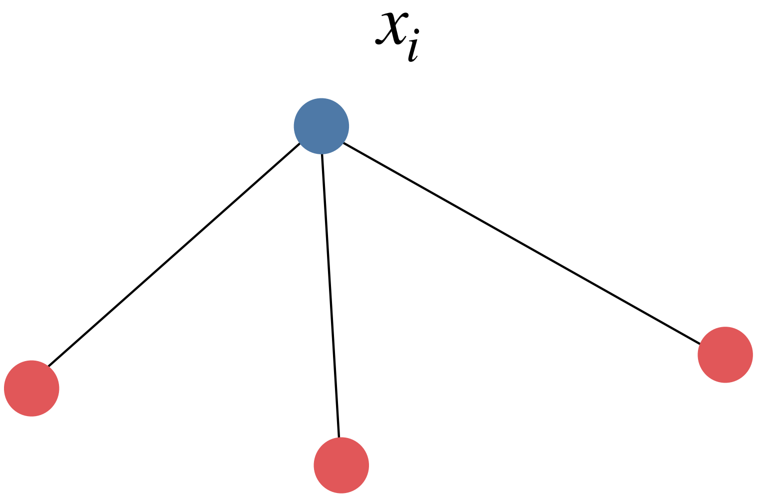
$$x_{agg} = \sum_{j \in \mathcal{N}(i)} \phi(x_j, x_i) \qquad \text{(sum)}$$

$$x_{agg} = \frac{\sum_{j \in \mathcal{N}(i)} \phi(x_j, x_i)}{|\mathcal{N}(i)|} \qquad \text{(mean)}$$

$$x_{agg} = \max_{j \in \mathcal{N}(i)} \phi(x_j, x_i) \qquad \text{(max or min)}$$

$$x_{agg} = \bigoplus_{j \in \mathcal{N}(i)} \phi(x_j, x_i)$$

Update



$$x_i^{(k+1)} = \gamma\left(x_{agg}, x_i^{(k)}\right)$$

All Together

$$x_i^{(k+1)} = \gamma\left(x_i^{(k)}, \bigoplus_{j \in N(i)} \phi\left(x_j^{(k)}, x_i^{(k)}\right)\right)$$

Simple version

activation function

$$x_i^{(k+1)} = \sigma\left(W_{self}^{(k)} x_i^{(k)} + W_{neigh}^{(k)} \sum_{j \in N(i)} x_j^{(k)} + b^{(k)}\right)$$

Graph Convolutional Network (GCN)

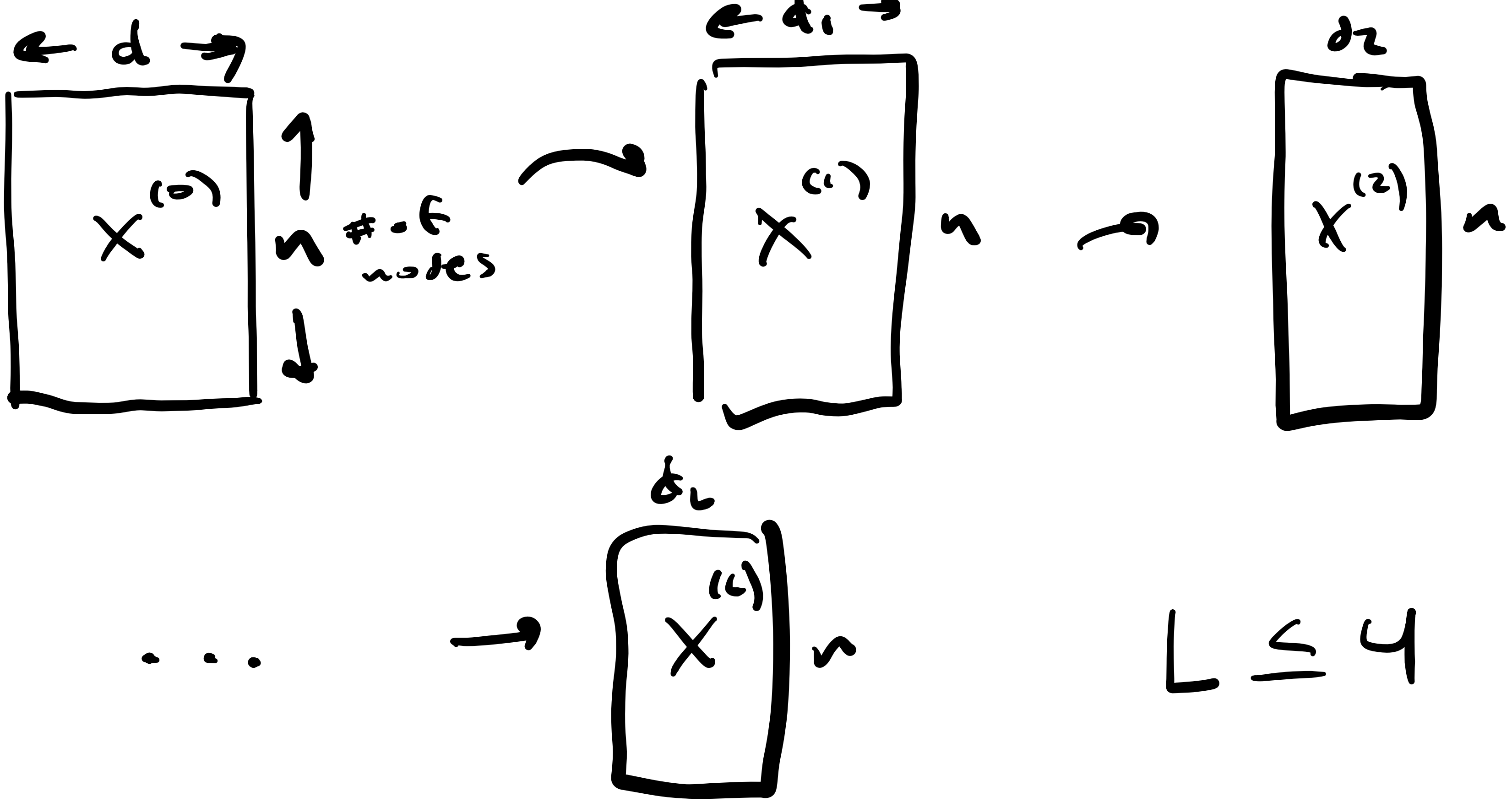
$$x_i^{(k+1)} = \sigma\left(W^{(k)} \sum_{j \in N(i) \cup \{i\}} \frac{x_j^{(k)}}{\sqrt{|N(i)|\,|N(j)|}} + b^{(k)} \right)$$

$$X = \begin{array}{cc} d^{(k)} & \\ \left[\begin{array}{c} \quad \\ \quad \\ \quad \end{array} \right] & n \end{array}$$

Matrix form (for some GNNs)

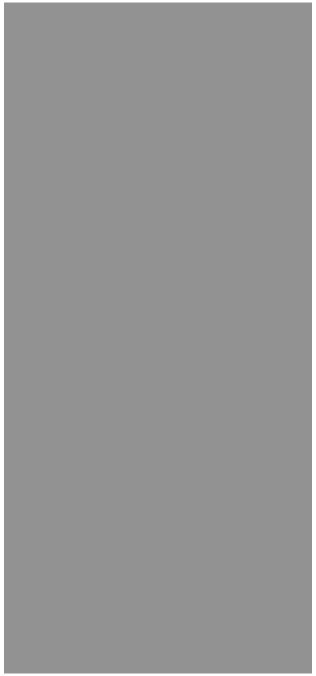
$$Y = \tilde{A} X \mathbf{W}$$

Network of Layers



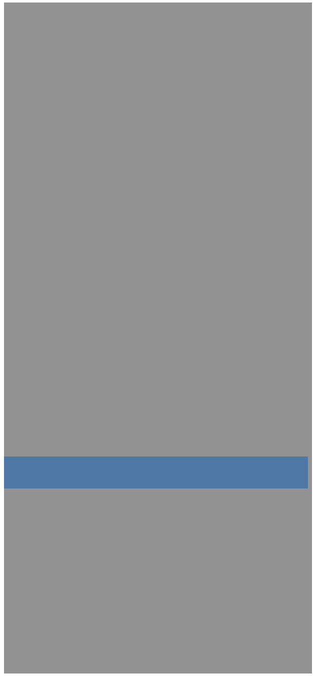
Node, link, and graph classification/regression

$$X^{(L)} =$$



Node, link, and graph classification/regression

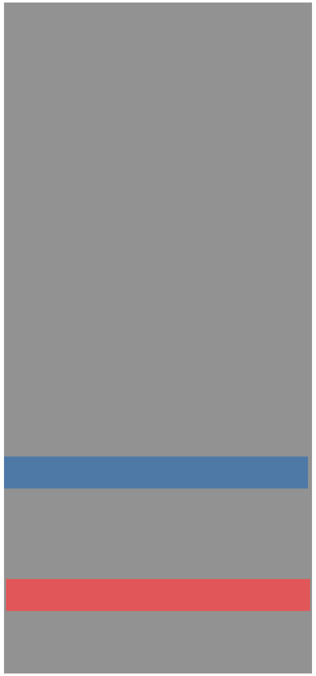
$$X^{(L)} =$$



node

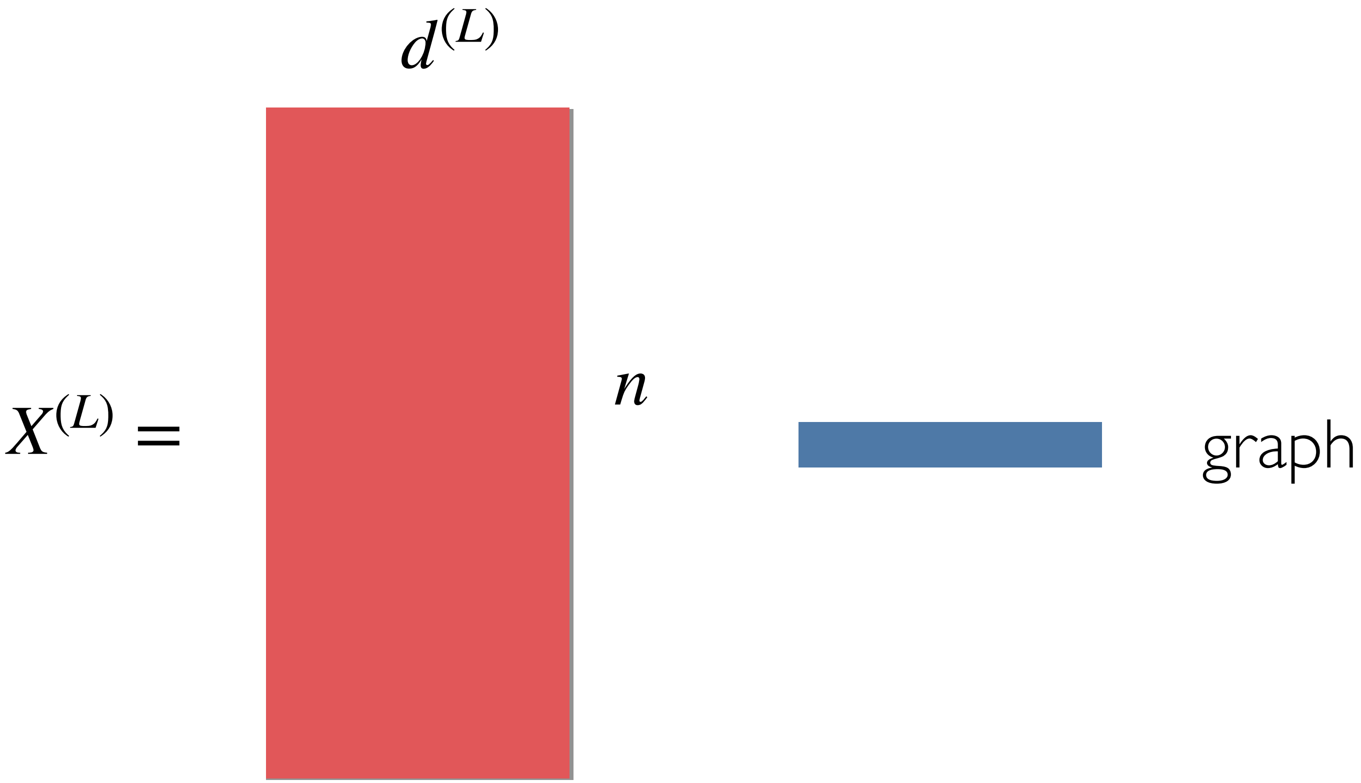
Node, link, and graph classification/regression

$$X^{(L)} =$$

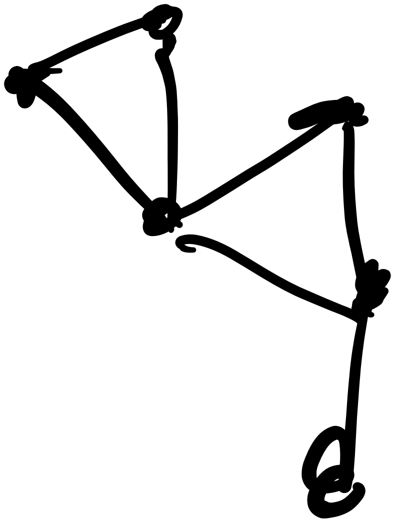


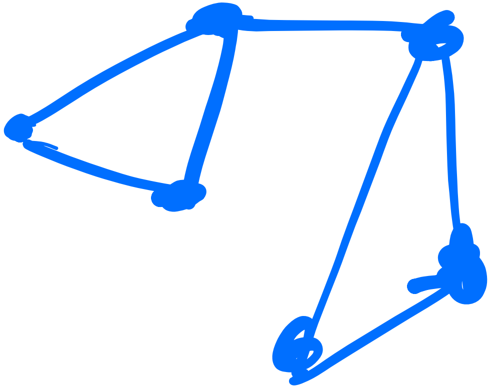
edge

Node, link, and graph classification/regression



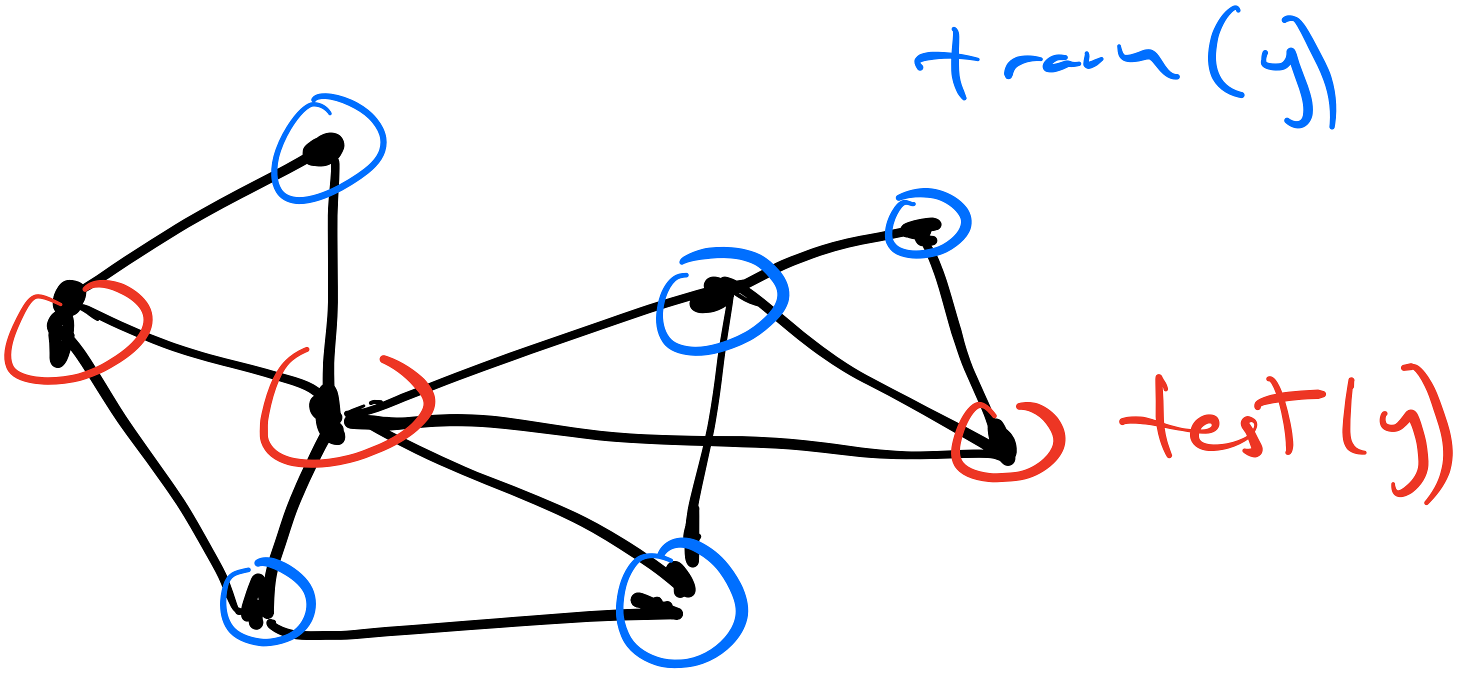
Testing (inductive)





test graphs never seen in training.

Testing (transductive)



Graph Attention Networks

$$x_{agg} = \sum_{j \in N(i)} \alpha_{ij} W x_j$$