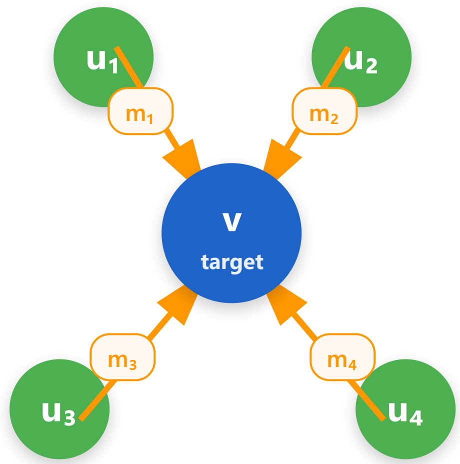
Graph Neural Networks (GNN) Fundamentals

Deep Learning on Graph-Structured Data

Message Passing Mechanism



1 **Aggregation**
Collect neighbor messages

2 **Transformation**
Apply neural network

3 **Update**
Compute new representation

GNN Pipeline

Node Features

↓

Hidden Representations

↓

Predictions

✓ Key Properties

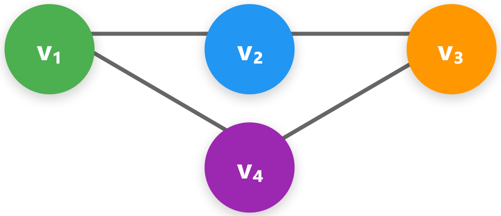
- 🔄 Permutation invariant
- 🧮 Learn node embeddings
- 🗺️ Preserve graph structure

🚀 Foundation

Basis for modern graph machine learning methods

How to Convert Graph to Convolution Input

📊 Example: Simple Graph



Adjacency Matrix (A)

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

Feature Matrix (X)

$$\begin{bmatrix} x_{1\,1} & x_{1\,2} & \cdots & x_{1\,a} \\ x_{2\,1} & x_{2\,2} & \cdots & x_{2\,a} \\ x_{3\,1} & x_{3\,2} & \cdots & x_{3\,a} \\ x_{4\,1} & x_{4\,2} & \cdots & x_{4\,a} \end{bmatrix}$$

⚙️ Graph Convolution Operation

Step 1: Normalize Adjacency

$\tilde{A} = D^{-½} A D^{-½}$
(Add self-loops & normalize)

↓

Step 2: Aggregate Features

$\tilde{H} = \tilde{A} X$
(Weighted sum of neighbors)

↓

Step 3: Apply Weights

$H' = \tilde{H} W$
(Linear transformation)

↓

Step 4: Non-linearity

$H^{(l+1)} = \sigma(H')$
(Apply activation function)

Complete Formula

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)})$$

💡 Key Points

- **Adjacency Matrix (A):** Encodes graph structure (who connects to whom)
- **Feature Matrix (X):** Node features (d-dimensional vectors for each node)
- **Degree Matrix (D):** Diagonal matrix with node degrees for normalization
- **Weight Matrix (W):** Learnable parameters (like CNN filters)
- **Output (H):** New node representations that incorporate neighborhood information