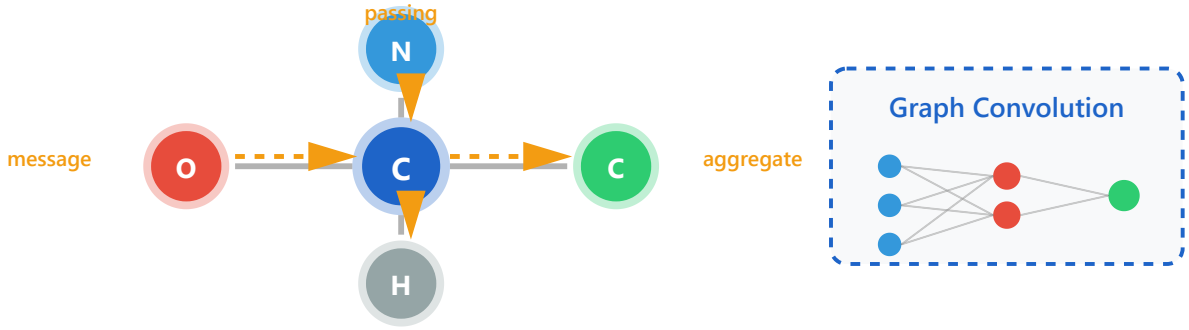
Graph Neural Networks



Atoms (Nodes) Bonds (Edges) Message Flow

Molecular graphs
Atoms as nodes, bonds as edges

Message passing
Information flow between atoms

Graph convolutions
Feature aggregation operations

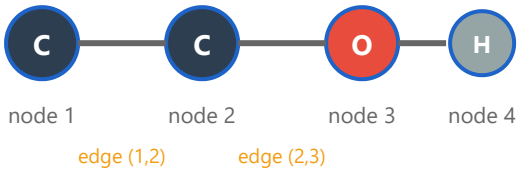
Attention mechanisms
Weighted information aggregation

Pooling strategies
Graph-level representations

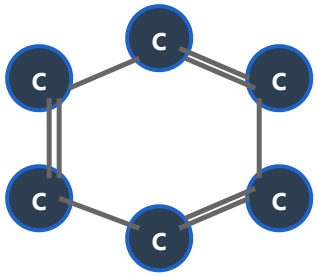
1. Molecular Graphs

Molecular graphs provide a natural representation of chemical compounds where **atoms are represented as nodes** and **chemical bonds as edges**. This graph-based representation captures both the structural and relational properties of molecules, making it ideal for machine learning applications in chemistry and drug discovery.

Ethanol (C_2H_5OH)



Benzene Ring (C_6H_6)



Node Features (Atom Properties)

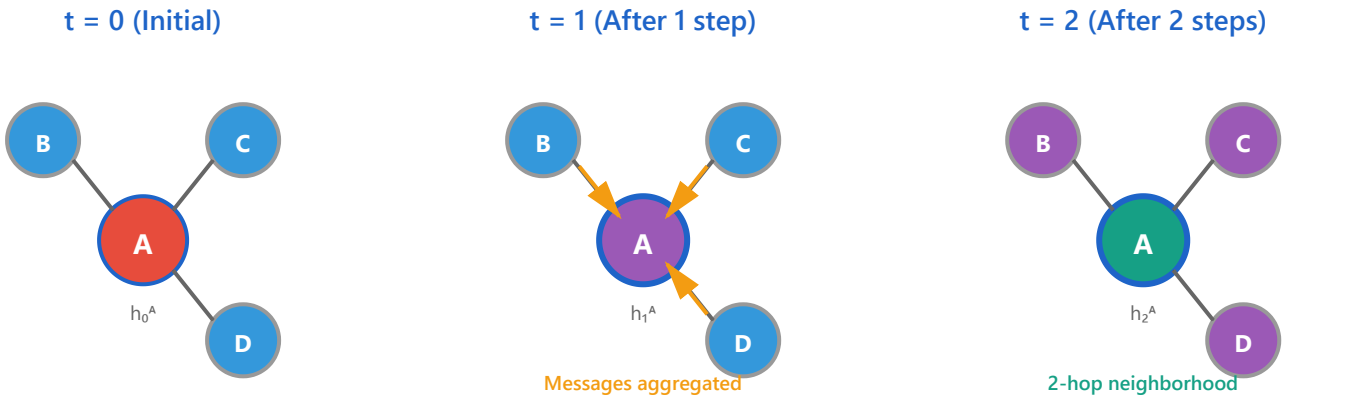
- Atomic number
- Formal charge
- Hybridization
- Aromaticity
- Degree
- Valence
- Chirality
- Mass

Key Characteristics:

- Each node stores features like atomic number, charge, hybridization state
- Edges encode bond types (single, double, triple) and stereochemistry
- Graph structure naturally captures molecular topology and connectivity
- Invariant to atom ordering, unlike SMILES string representations

2. Message Passing

Message passing is the fundamental operation in GNNs where **nodes exchange information with their neighbors**. Each node aggregates messages from connected nodes, allowing information to propagate through the graph structure. This iterative process enables nodes to develop representations that incorporate both local and global graph context.



Message Passing Formula

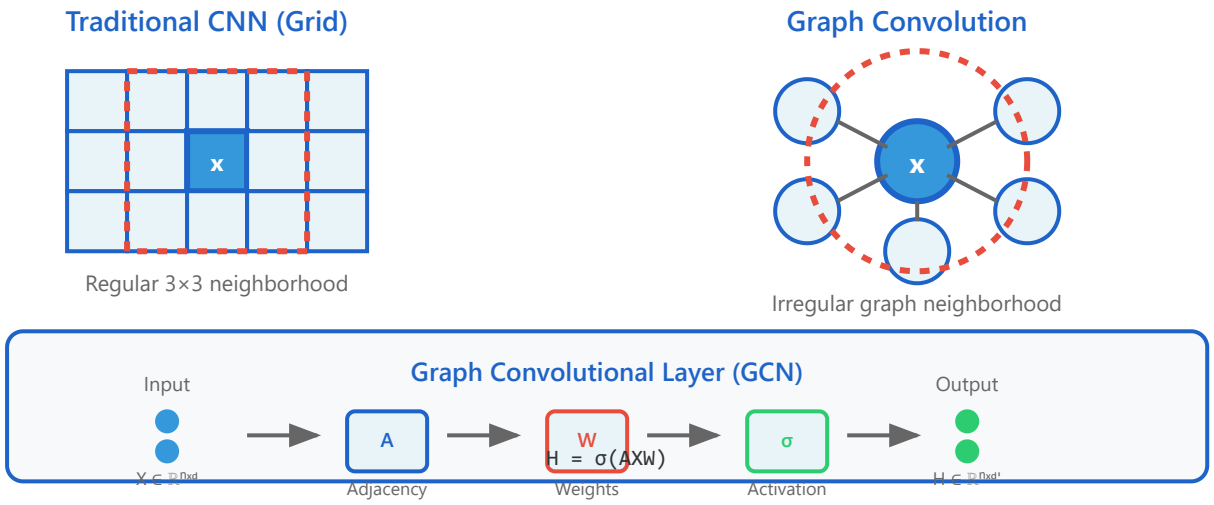
$$h_v^{(k+1)} = \text{UPDATE}_{(k)}(h_v^{(k)}, \text{AGGREGATE}_{(k)}(\{h_u^{(k)} : u \in N(v)\}))$$

Key Properties:

- Information propagates through edges: after k steps, nodes have information from k-hop neighbors
- AGGREGATE function combines neighbor messages (sum, mean, max, or learnable)
- UPDATE function combines aggregated messages with current node state
- Enables learning of both local patterns and global graph structure

3. Graph Convolutions

Graph convolutions extend the concept of convolutional neural networks to graph-structured data. Unlike regular convolutions that operate on grid-like structures, **graph convolutions aggregate and transform features from irregular neighborhoods** defined by graph connectivity.



Common Graph Convolution Variants:

GCN: $H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)})$
GraphSAGE: $h_v^{(l+1)} = \sigma(W \cdot \text{CONCAT}(h_v^{(l)}, \text{AGGREGATE}(\{h_u^{(l)} : u \in N(v)\})))$
GIN: $h_v^{(l+1)} = \text{MLP}((1 + \varepsilon) \cdot h_v^{(l)} + \sum h_u^{(l)})$

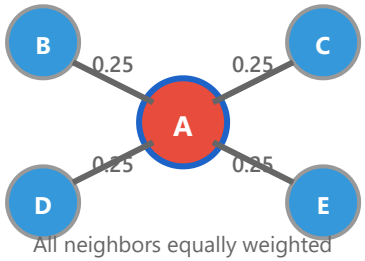
Key Advantages:

- Permutation invariant: output doesn't depend on node ordering
- Parameter sharing across different neighborhood sizes
- Can process graphs of varying sizes and structures
- Learns hierarchical representations through stacking layers

4. Attention Mechanisms

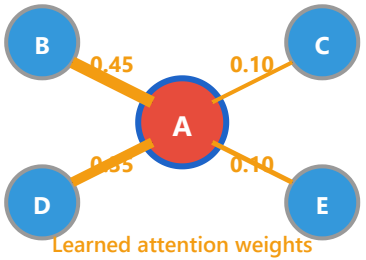
Attention mechanisms in GNNs allow nodes to **selectively weight the importance of different neighbors** during message aggregation. This enables the network to focus on the most relevant connections and improves model interpretability by revealing which graph relationships are most important for predictions.

Uniform Aggregation



All neighbors equally weighted

Attention-based Aggregation



Learned attention weights

Graph Attention Network (GAT) Mechanism

Step 1: Compute attention scores: $e_{ij} = a(Wh_i, Wh_j)$

Step 2: Normalize with softmax: $\alpha_{ij} = \text{softmax}_j(e_{ij})$

Step 3: Aggregate: $h'_i = \sigma(\Sigma_{j \in N(i)} \alpha_{ij} Wh_j)$

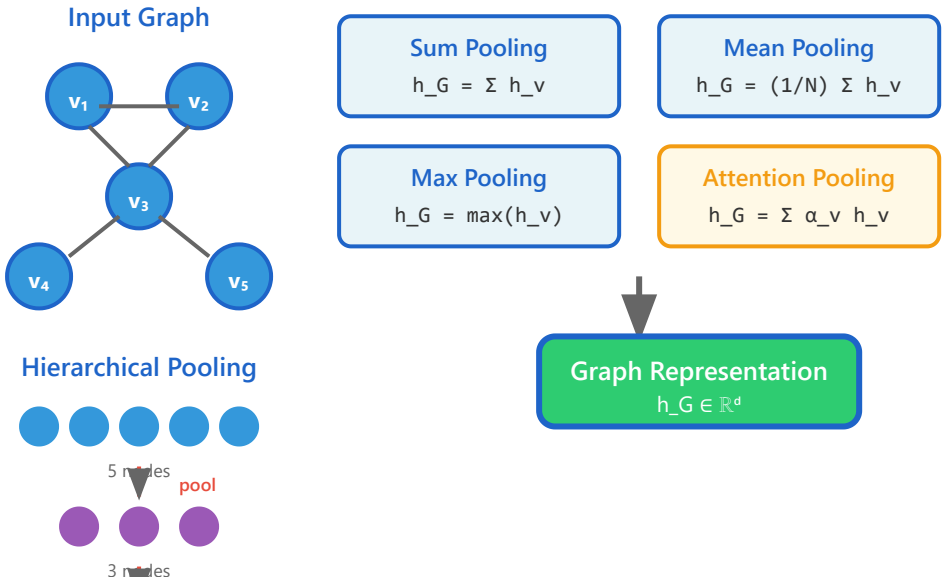
Multi-head: Concatenate K attention heads

Benefits of Attention:

- Automatically learns which neighbors are most relevant for each task
- Provides interpretability by visualizing attention weights
- Multi-head attention captures different types of relationships
- Handles varying neighborhood sizes and heterogeneous graphs naturally
- Improves performance on complex molecular property prediction tasks

5. Pooling Strategies

Pooling operations in GNNs **aggregate node-level representations into graph-level representations**, enabling predictions at the graph level. Different pooling strategies capture different aspects of graph structure and are crucial for tasks like molecular property prediction and graph classification.



Common Pooling Methods:

Global Add Pool: $h_G = \Sigma_{v \in G} h_v$
Global Mean Pool: $h_G = (1/|V|) \Sigma_{v \in G} h_v$
Global Max Pool: $h_G = \max_{v \in G}(h_v)$
Set2Set: Uses LSTM to aggregate node features iteratively
DiffPool: Learns soft cluster assignments for hierarchical pooling

Pooling Considerations:

- Global pooling methods are permutation invariant and simple to implement
- Hierarchical pooling captures multi-scale graph structures

- Attention-based pooling focuses on important nodes
- Choice of pooling affects model capacity and inductive bias
- Critical for graph-level prediction tasks like molecule property prediction