# Survey of GNN

范梅浩

2022 年 5 月 1 日

# What is graph

- Picture.

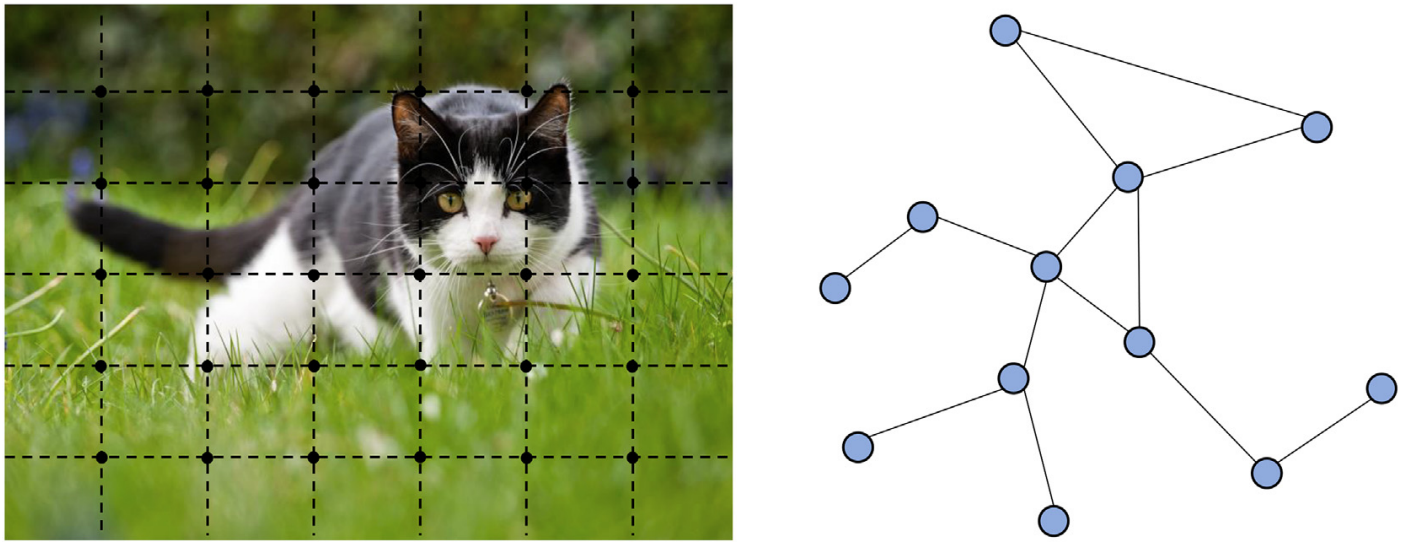
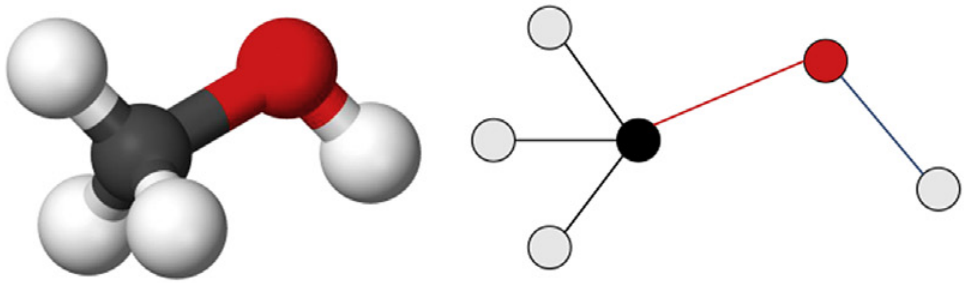
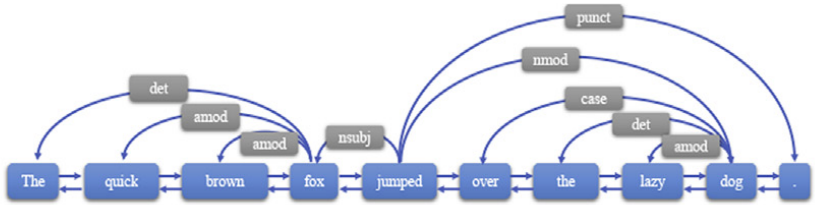


**Fig. 1.** Left: image in Euclidean space. Right: graph in non-Euclidean space.

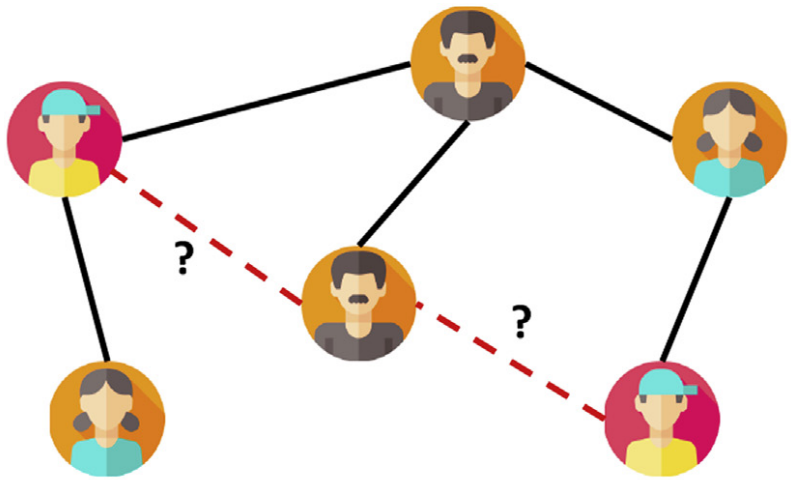
- Molecule.



- Text.


punct
nmod
case
det
amod
det
amod
amod
nsubj
The
quick
brown
fox
jumped
over
the
lazy
dog
.

- Social Network.



# Different definitions of GNN

- A GNN is an optimizable transformation on all attributes of the graph that preserves graph symmetries ——A gentle introduction to GNN

# Different definitions of GNN

- A GNN is an optimizable transformation on all attributes of the graph that preserves graph symmetries ——A gentle introduction to GNN
- GNNs are neural models that capture the dependence of graphs via message passing between the nodes of graphs. ——Graph Neural Networks: A Review of Methods and Applications

# Different definitions of GNN

- A GNN is an optimizable transformation on all attributes of the graph that preserves graph symmetries ——A gentle introduction to GNN
- GNNs are neural models that capture the dependence of graphs via message passing between the nodes of graphs. ——Graph Neural Networks: A Review of Methods and Applications
- Models that aim to learn a state embedding which contains the information of the neightborhood and itself for each nodes ——Graph Neural Networks: A Review of Methods and Applications

# Long standing history of NN for graph

- Recursive NN on directed acyclic graphs. 1997, 1998
- Recurrent NN and Feedforward NN. 2009
- Convolutional NN. 2017 Michael M et al.

# Graph representation learning-Deep Walk

- Random Walk for randomly sampling sequences of nodes.
- Skip Gram for updating node representations.

**Algorithm 1** DeepWalk($G$, $w$, $d$, $\gamma$, $t$)

**Input:** graph $G(V, E)$
window size $w$
embedding size $d$
walks per vertex $\gamma$
walk length $t$
**Output:** matrix of vertex representations $\Phi \in \mathbb{R}^{|V| \times d}$

Initialization: Sample $\Phi$ from $\mathcal{U}^{|V| \times d}$
Build a binary Tree $T$ from $V$
**for** $i = 0$ to $\gamma$ **do**
  $\mathcal{O} = \text{Shuffle}(V)$
  **for each** $v_i \in \mathcal{O}$ **do**
    $\mathcal{W}_{v_i} = RandomWalk(G, v_i, \text{t})$
    SkipGram($\Phi$, $\mathcal{W}_{v_i}$, $w$)
  **end for**
**end for**

# Graph representation learning-node2vec

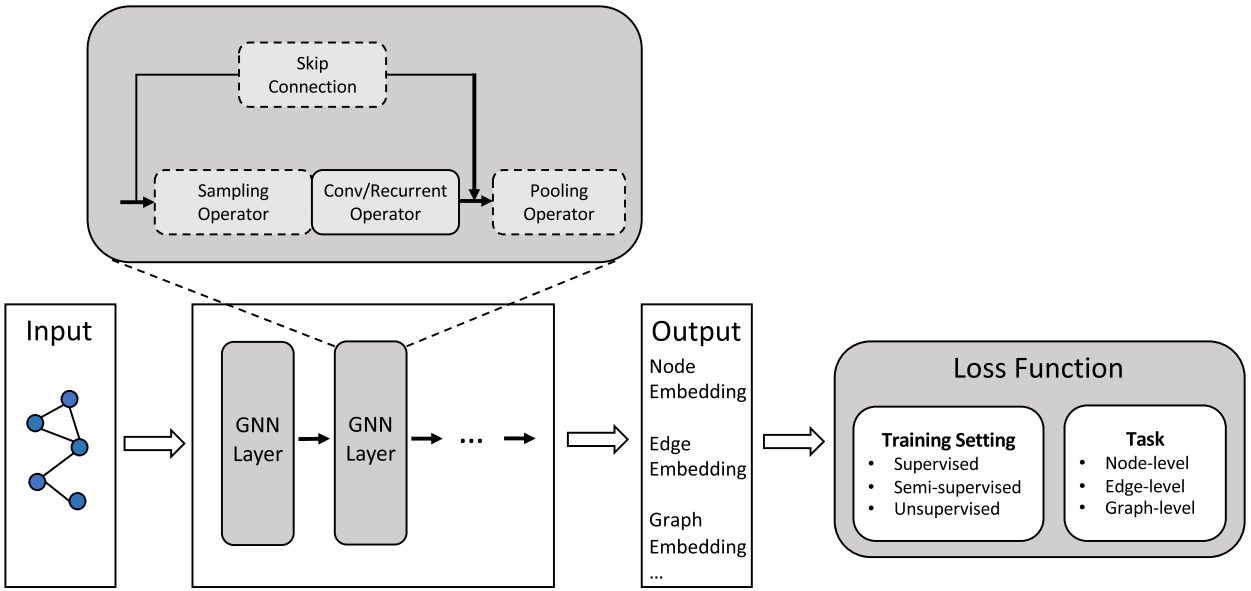
- Random Walk for randomly sampling sequences of nodes.
- Regard node sequences as text sequences and use word2vec to generate node representations.
- word2vec: train a classifier to classify whether word A is surrounding with word B. Then use the regression weights as the embeddings.

- Find graph structure

- Find graph structure
- Specify graph type and scale

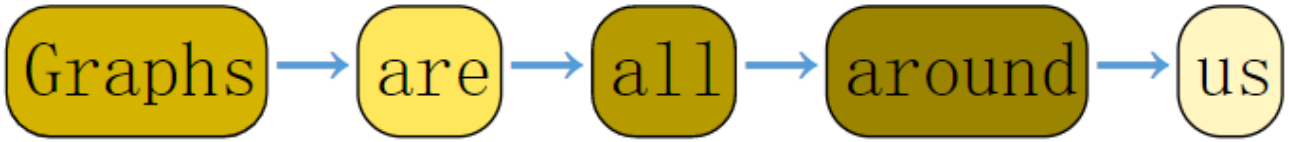
- Find graph structure
- Specify graph type and scale
- Build model using computational modules

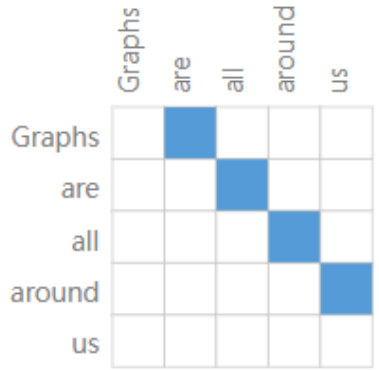
- Find graph structure
- Specify graph type and scale
- Build model using computational modules
- Design loss function



# How to build graph-strucured data

- Text as graph



- Image as graph

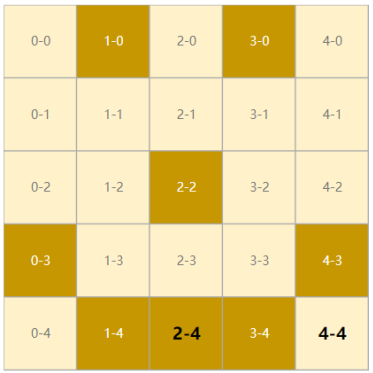
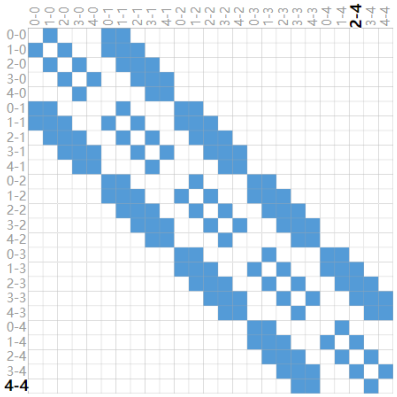
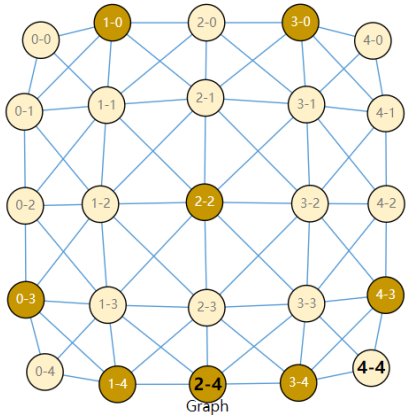


Image Pixels

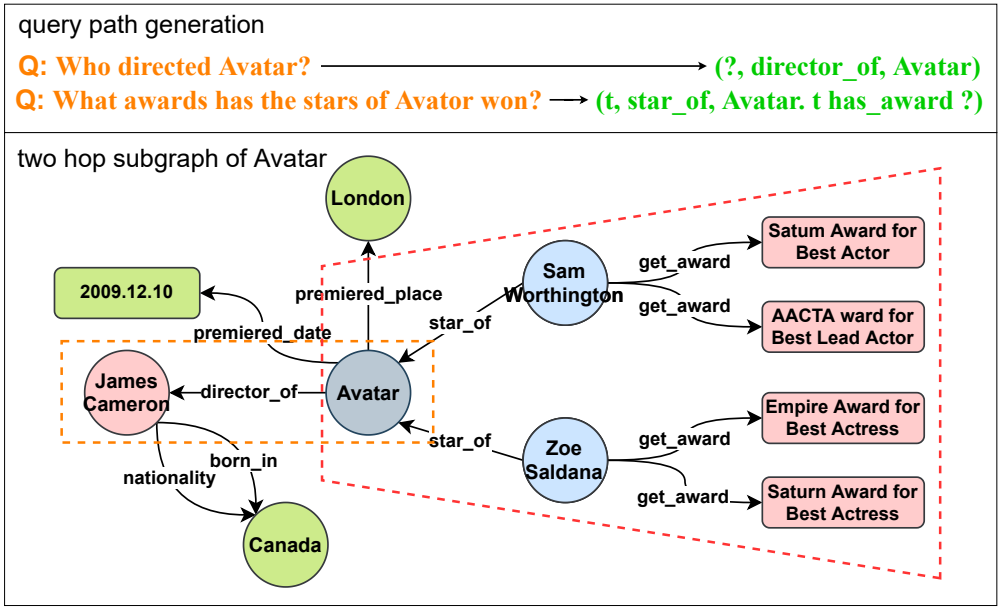

Adjacency Matrix



Graph

# Graph type

- directed/undirected
- homogeneous/heterogeneous
- static/dynamic



# Different tasks

- Node-level: node classification, node regression, node clustering
- Edge-level: edge classfication, link prediction
- Graph-level: graph classification, graph regression, graph matching

- Supervised setting
- Semi-supervised setting: small amount of labeled nodes and a large amount of unlabeled nodes for training
- Unsupervised setting: Node clustering

# Propagation module

This module is used to propagate information between nodes so that the aggregated information could capture both feature and topological information.

- Convolution operator
- Recurrent operator
- Skip connection: gather information from historical representations of nodes and mitigate the over-smoothing problem.

# Convolution operator

The main idea of convolution operators is to generalize from other domain to the graph domain.

- Spectral approaches
- Spatial approaches
  - Basic spatial approaches
  - Attention-based spatial approaches
  - General frameworks for spatial approaches

# Spectral approaches

- Graph signal: A graph signal defined on the nodes is represented as a vector $x \in R^n$, where $x(i)$ is the signal value on the node i. Denote $X \in R^{n \times d}$ as the node attribute matrix of an attributed graph, and then, the columns of X are the d signals of the graph.
- A graph signal $x$ is first trainsformed to the spectral domain, conduct the convolution operation and transformed back.

$$\begin{aligned} g \star x &= F^{-1}(F(g) \odot F(x)) \\ &= U(U^T g \odot U^T x) \end{aligned}$$

where $U$ is the matrix of eigenvectors of the normalized graph Laplacian $L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$(D is the degree matrix, A is the adjacency matrix)

# Spectral approaches

- Since $U^T g$ is the filter in the specral domain, if we simplify the filter by using a learnable diagonal matrix $g_w$, basic function can be simplified

$$g_w \star x = U g_w U^T x$$

- Typical methods which design different filters $g_w$
  - Spectral Network: Learn $g_w$.
  - ChebNet suggest that $g_w$ can be approximated in terms of Chebyshev polynomials $T_k(x)$ up to $K^{th}$ order.

$$g_w \star x \approx \sum_{k=0}^{K} w_k T_k(\hat{L}) x$$

where $\hat{L} = \frac{2}{\lambda_{max}} L - I_N$, $\lambda_{max}$ denotes the largest eigenvalue of $L$ whose range is [-1, 1].

# Spectral approaches

- Typical methods which design different filters $g_w$
  - GCN simplify ChebNet by set $K = 1$ to alleviate the problem of overfitting and further assume $\lambda_{max} \approx 2$.

$$g_w \star x \approx w_0 x + w_1 (L - I_N) x = w_0 x - w_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x$$

With parameter constraint $w = w_0 = -w_1$, following expression can be obtained.

$$g_w \star x \approx w(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) x$$

Finally, the compact form of GCN is defined as:

$$H = \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} X W$$

where GCN use $I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}}$ to solve the exploding/vanishing gradient problem, with $\hat{A} = A + I_N$ and $\hat{D}_{ii} = \sum_j \hat{A}_{ij}$.

# Spectral approaches

- Typical methods which design different filters $g_w$
  - Based on GCN, AGCN learns a "residual" graph Laplacian and add it to the original Laplacian matrix.
  - DGCN uses two convolutional networks to capture the local and global consistency and adopts an unsupervised loss to ensemble(合成) them.

$$H' = \rho(D_P^{-\frac{1}{2}} A_P D_P^{-\frac{1}{2}} HW)$$

where $A_P$ is the PPMI matrix and $D_P$ is the diagonal degree matrix of $A_P$.

  - Advantage and Drawback.

# Basic spatial approaches

Define Convolutions directly on the graph based on the graph topology.

- Neural FPs.

$$t = h_v^t + \sum_{u \in N_v} h_u^t$$

$$h_v^{t+1} = \sigma(tW_{|N_v|}^{t+1})$$

where $W_{|N_v|}^{t+1}$ is the weight matrix for nodes with degree $\mid N_v \mid$.

- DCNN use transition matrices to define the neighborhood for nodes.

$$H = f(W_c \odot P^* X) \in R^{N \times K \times F}$$

where $P^*$ is an $N \times K \times N$ tensor which contains the power series $\{P, P^2, ..., P^K\}$ of matrix P. $P_{ij}^t$ denotes the possibility of drumping from node i to node j in hop t.

# Basic spatial approaches

- PATCHY-SAN extracts and normalizes a neighborhood of exactly k nodes for each node.
- GraphSAGE generates embeddings by sampling and aggregating features from a node's local neighborhood.

$$h_{N_v}^{t+1} = AGG_{t+1}(\{h_u^t, u \in N_v\})$$

$$h_v^{t+1} = \sigma(W^{t+1} \cdot [h_v^t, h_{N_v}^{t+1}])$$

$AGG_{t+1}$ is the aggregation function and can be instantiated by Mean, Pooling and LSTM aggregator.

# Attention-based spatial approaches

Assign different weights for neighbors.

- GAT incorporates the attention mechanism into the propagation step.

$$h_v^{t+1} = \rho(\sum_{u \in N_v} a_{vu} W h_u^t)$$

$$a_{uv} = \frac{exp(LeakyReLU(a^T[Wh_v, Wh_u]))}{\sum_{k \in N_v} exp(LeakyReLU(a^T[Wh_v, Wh_u]))}$$

Moreover, GAT utilizes the multi-head attention to stabilize the learning process.

$$h_v^{t+1} = ||\sigma(\sum_{u \in N_v} a_{vu}^k W_k h_u^t)$$

$$h_v^{t+1} = \sigma(\frac{1}{K} \sum_{k=1}^{K} \sum_{u \in N_v} a_{vu}^k W_k h_u^t)$$

# General Frameworks for spatial approaches

Aims to integrate different models into one single framework.

- MoNet try to unifies models for non-euclidean domains.

$$f \star g = \sum_{j=1}^{J} g_j D_j(v) f$$

$$D_j(v) f = \sum_{u \in N_v} w_j(\mathbf{u}(v,u)) f(u)$$

where $w_j$ is the function assigning weights for neighbors according to their pseudo-coordinates. $\mathbf{u}(v,u)$ is pseudo-coordinates.

# Recurrent operator

The major difference between recurrent operators and convolution operators is that layers in convolution operators use different weights while layers in recurrent operators share same weights.

$$H = F(H, X)$$

$$O = G(H, X_N)$$

where $H$, $O$, $X$, and $X_N$ are the matrices constructed by stacking all the states, all the outputs, all the features, and all the node features, respectively. **Limitations:**

- The model requires f to be a contraction map which limits the model's ability.
- The distribution of representation in the fixed point will be much smoother in value and less informative for distinguishing each node.

# Skip connection

More layers(i.e. k layers) make each node aggregate more information from neighbors k hops away. However, more layers can also propagate noisy information.

- JKN propose the jump knowledge network which could learn adaptive and structure-aware representations. JKN selects from all of the intermediate representations from each node at the last layer and use approaches of concatenation such as max-pooling, LSTM-attention to aggregate information. JKN can be combined with models like GraphSAGE and GAT to improve performance.
- DeepGCNs borrow ideas from ResNet which incorporates residual connections and dense connections to solve the problems of vanishing gradient and oversmoothing.

# Sampling module

To alleviate this "neighbor explosion" issue, sampling is needed to conduct the propagation.

- Node sampling: select a subset from each node's neighborhood
- Layer sampling: retain a small set of nodes for aggregation in each layer
- Subgraph sampling: restrict neighborhood search within subgraphs

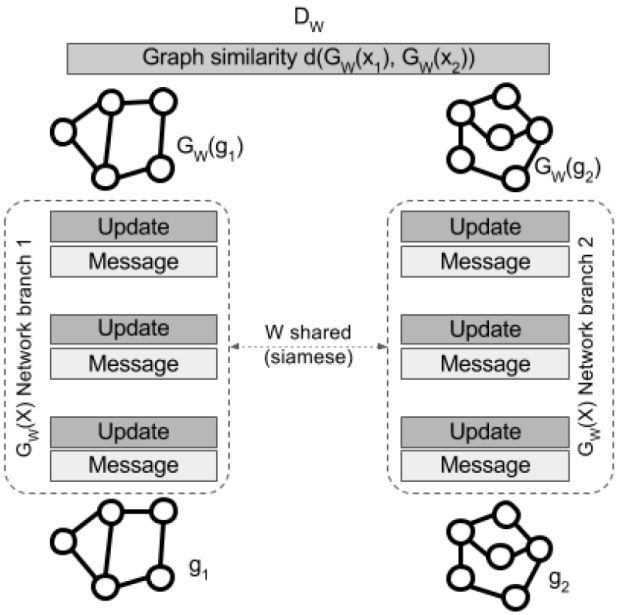
# Pooling module

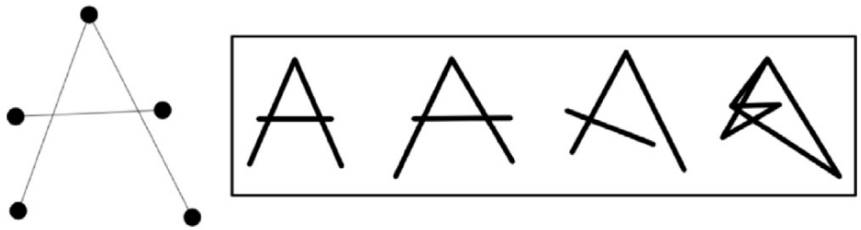
A convolutional layer is usually followed by a pooling layer to get more general features.

- Simple Pooling: sum/max/mean/attention
- Hierarchical pooling modules: investigate the hierarchical property of graph structure

# Graph Minning

- Graph Matching.



- Graph Clustering: group the vertices of a graph into clusters based on the graph structure and/or node attributes

# Chemistry and Biology

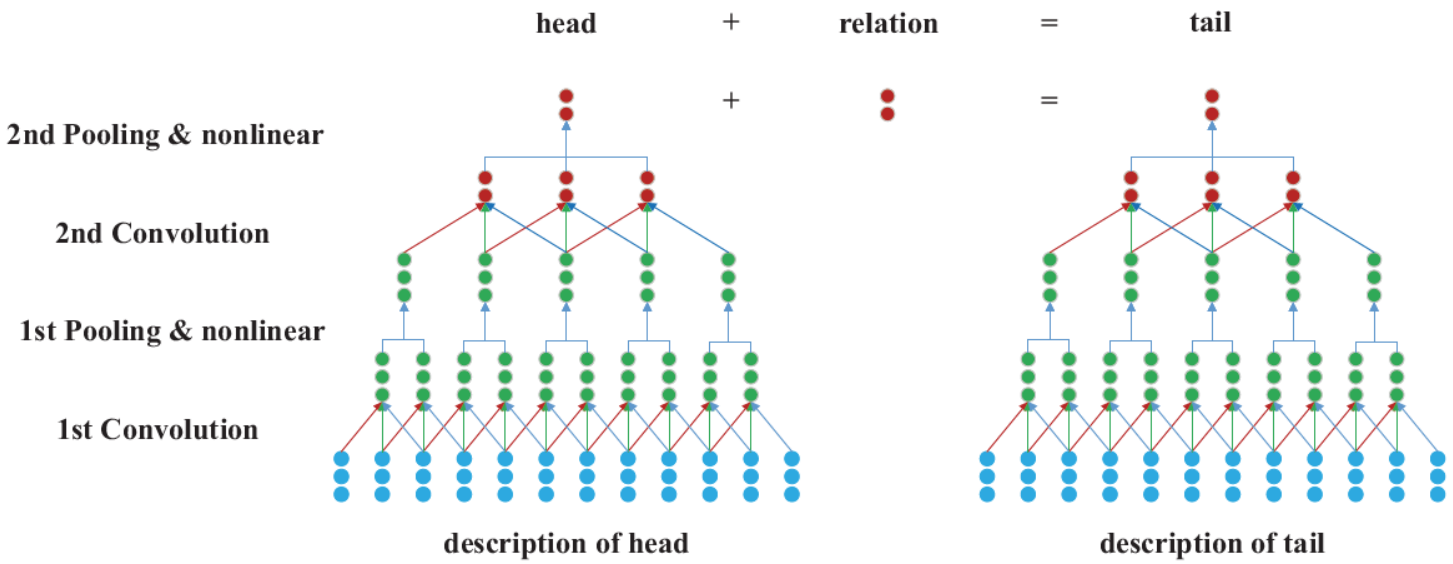
- Chemical Reaction Prediction: encode the input molecules and generates an intermediate graph with a node pair prediction network and a policy network

# Chemistry and Biology

- Chemical Reaction Prediction: encode the input molecules and generates an intermediate graph with a node pair prediction network and a policy network
- Protein Interface Prediction: determine whether particular residues constitute part of a protein

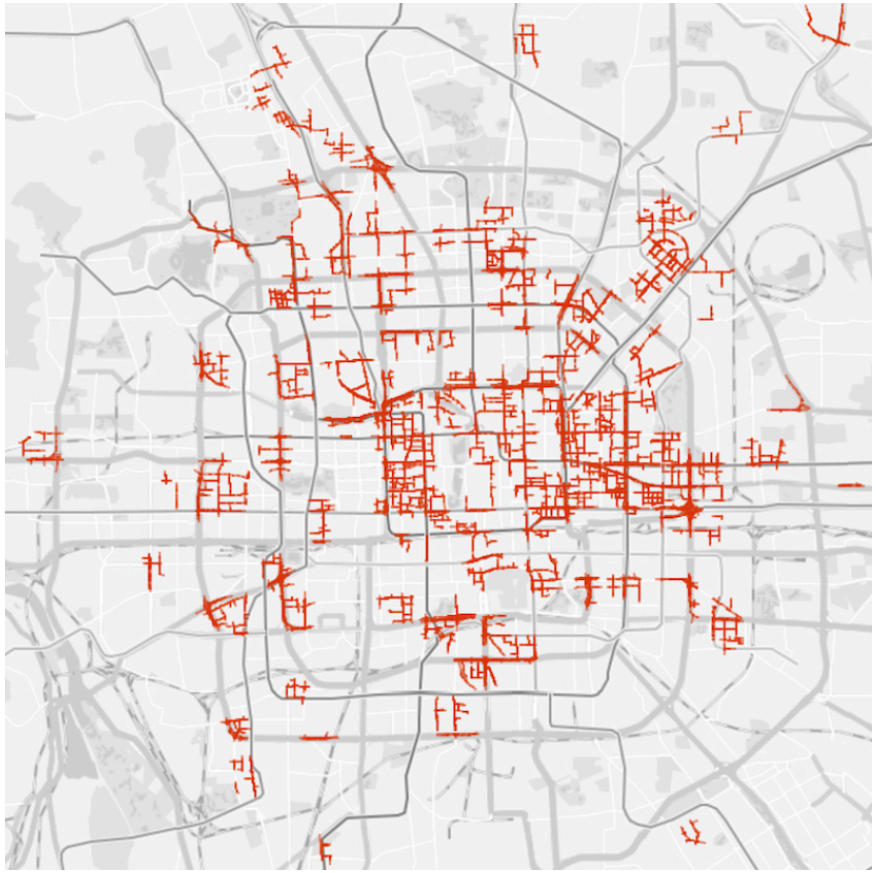
# Knowledge Graph

- Knowledge Graph Representation Learning



# Traffic networks

Use GNNs to better capture the structural information among modes.



- Robustness

- Robustness
- Interpretability

- Robustness
- Interpretability
- Graph Pretraining

- Robustness
- Interpretability
- Graph Pretraining
- Complex Graph Structures

*Thanks!*