Introduction to

Graph Machine Learning

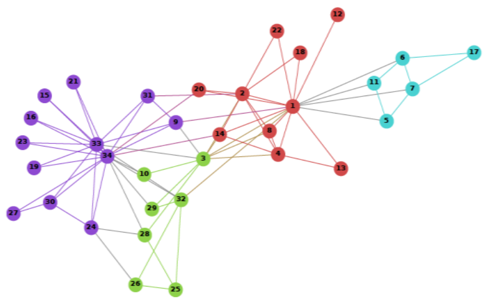
Rizal Fathony
rizal@fathony.com

*Some of the materials are modified from Juree Leskovec's graph presentations

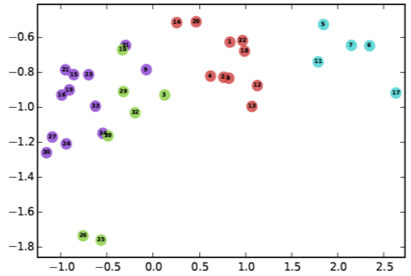
Graph Neural Networks

End-to-end learning for graph data

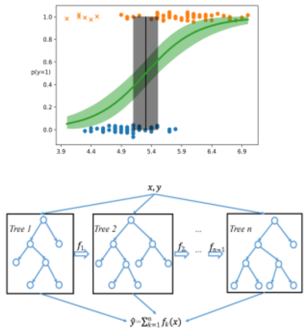
Node Embedding Limitations



Graph



Embedding



Classifier
(ML models)

Solving problems in two steps

Learn embedding first, then learn predictive model

Do not consider node features

Embeddings are generated solely based on the graph structure

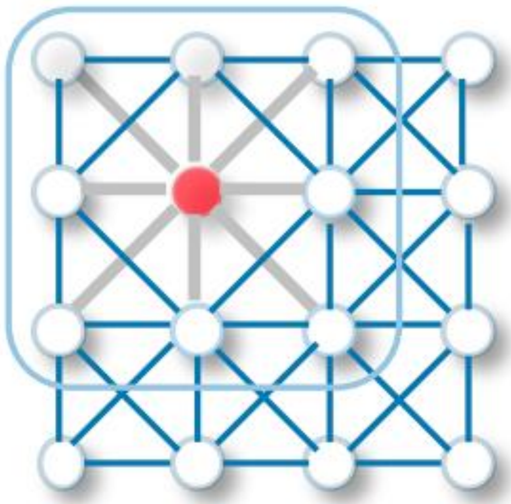
Transductive learning (instead of inductive)

Impossible to generate new embedding for new nodes not seen in the training

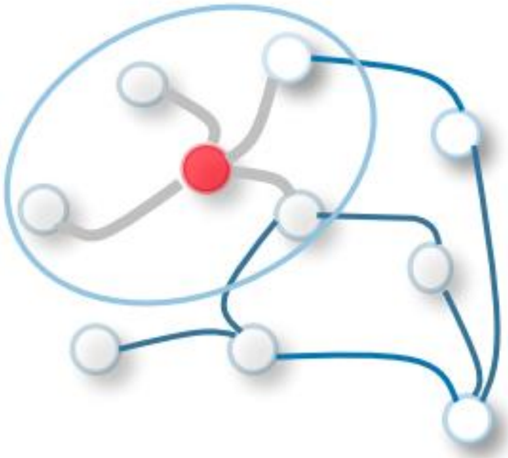
“Shallow” learning

Unable to take advantage of the representation power of deep neural networks

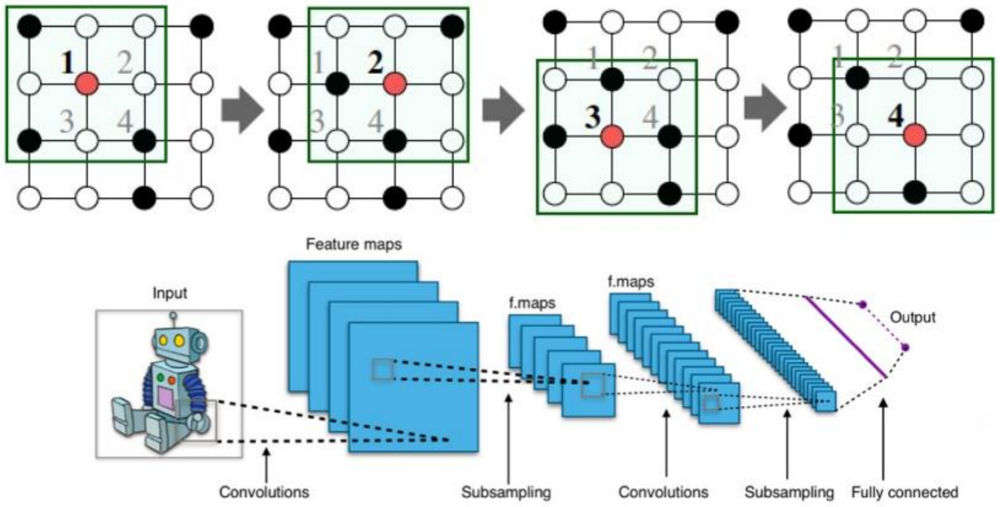
GNN Inspiration: CNN



Grid Computation Flow



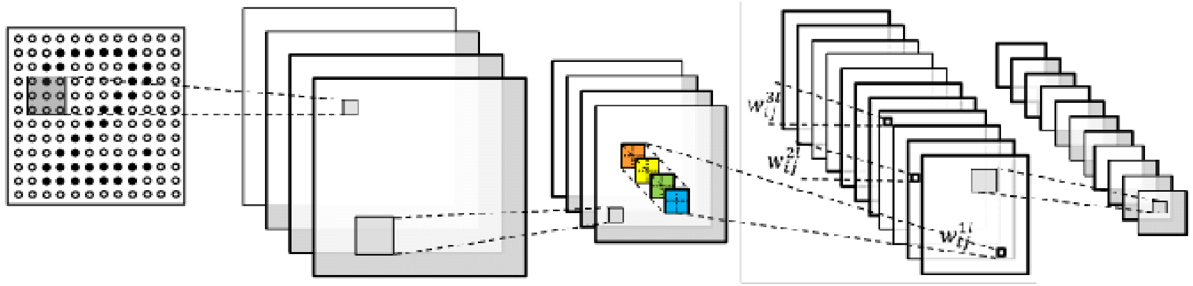
Graph Computation Flow



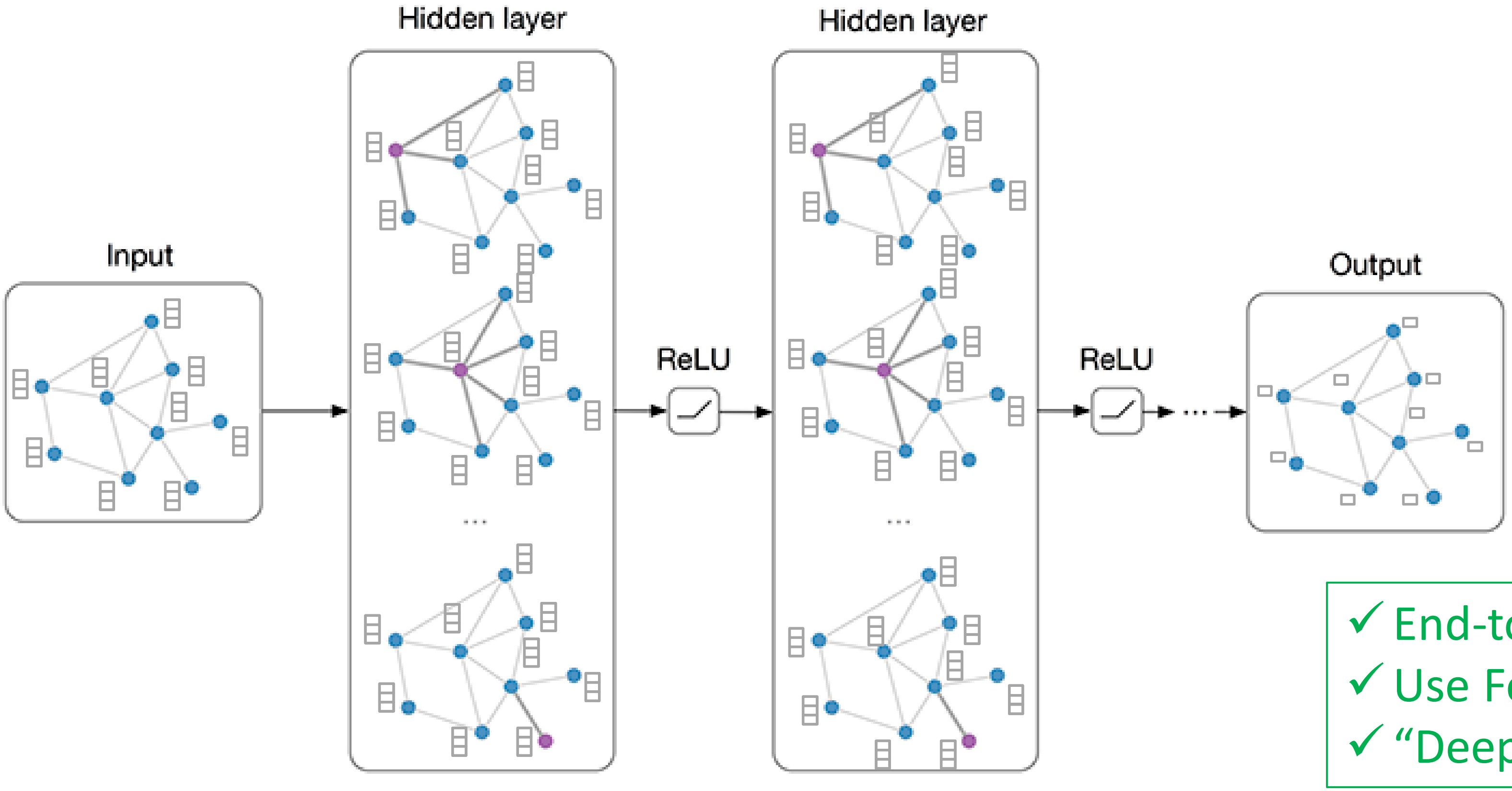
Leverage node attributes

https://arxiv.org/pdf/1901.00596.pdf

Graph Convolutional Networks



Convolutional Neural Networks (CNN)



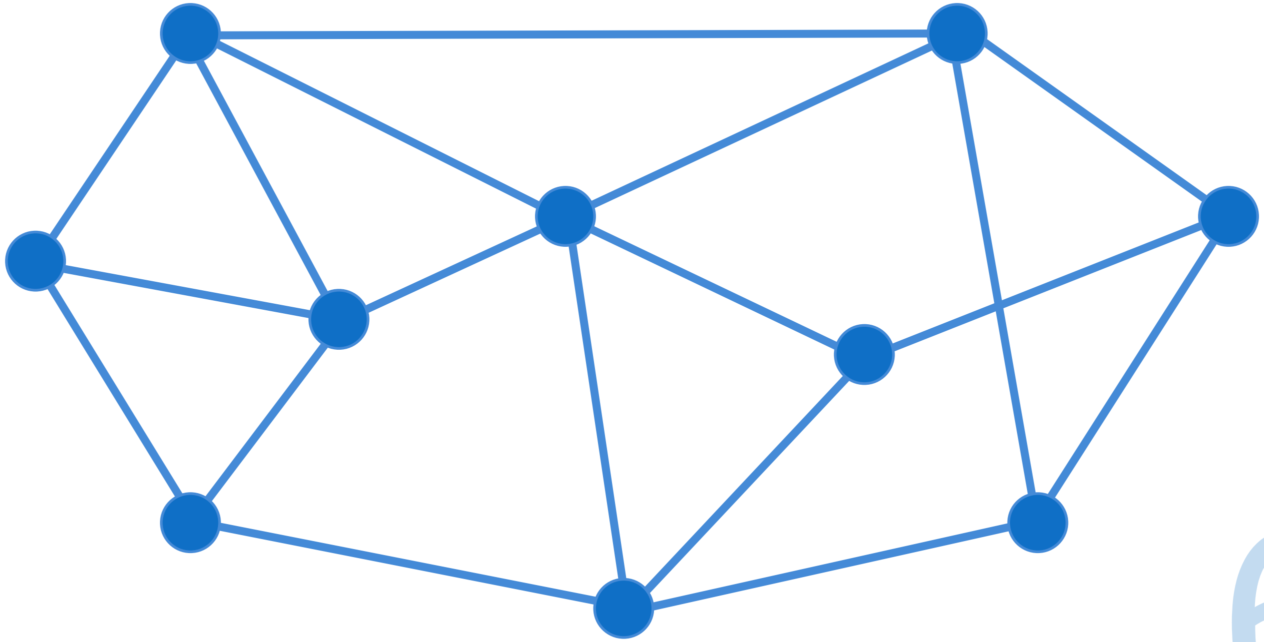
Graph Convolutional Networks (GCN)

- ✓ End-to-end Learning
- ✓ Use Features
- ✓ "Deep" Learning

https://tkipf.github.io/graph-convolutional-networks/

Setup

- **Assume we have a graph G:**
  - V is the **vertex set**
  - $\boldsymbol{A}$ is the **adjacency matrix** (assume binary)
  - $\boldsymbol{X} \in \mathbb{R}^{m \times |V|}$ is a matrix of **node features**
  - v: a node in V; $N(v)$: the set of neighbors of v.



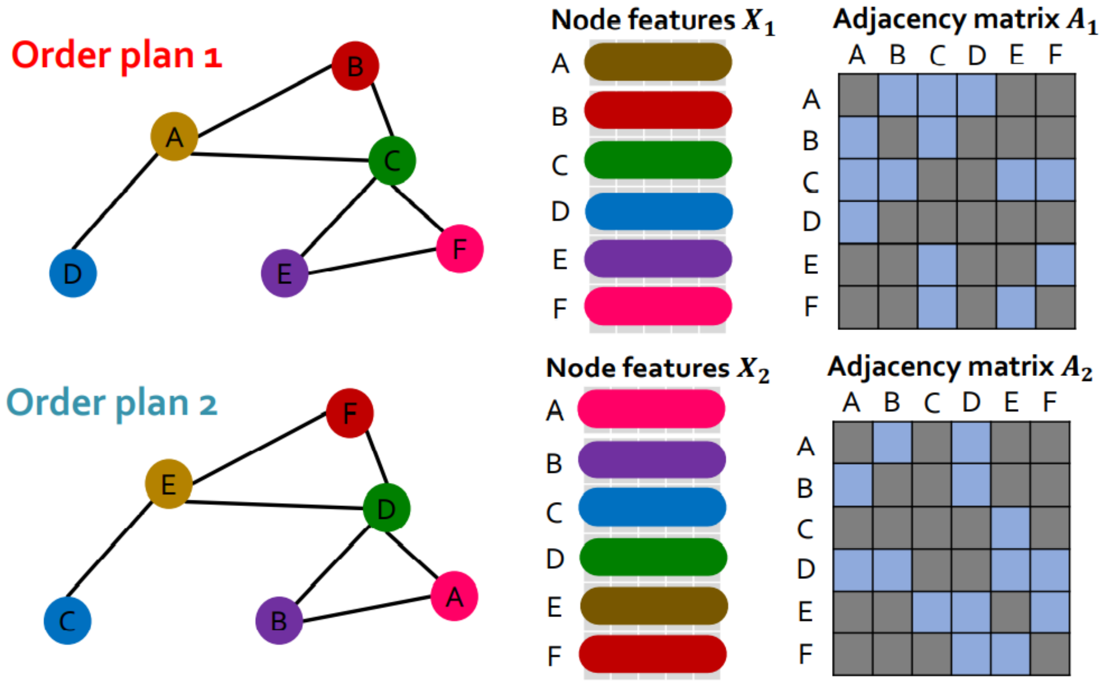
Desirable properties

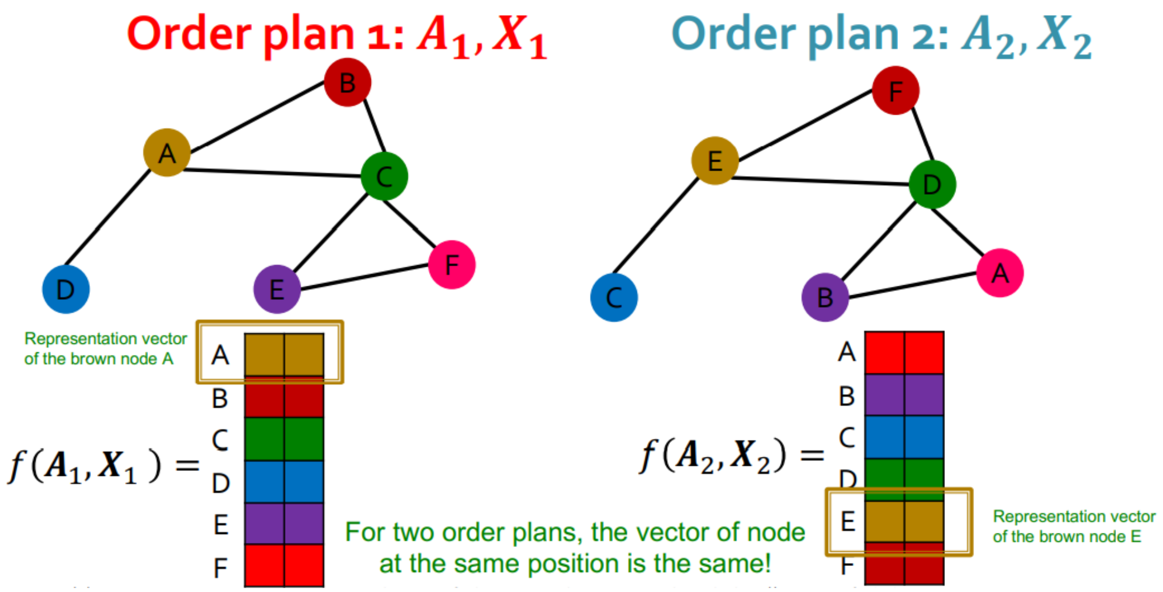
Permutation invariance

Graph does not have a canonical order of the nodes.

Permutation equivariance

For two order plans, the vector of node at the same position is the same

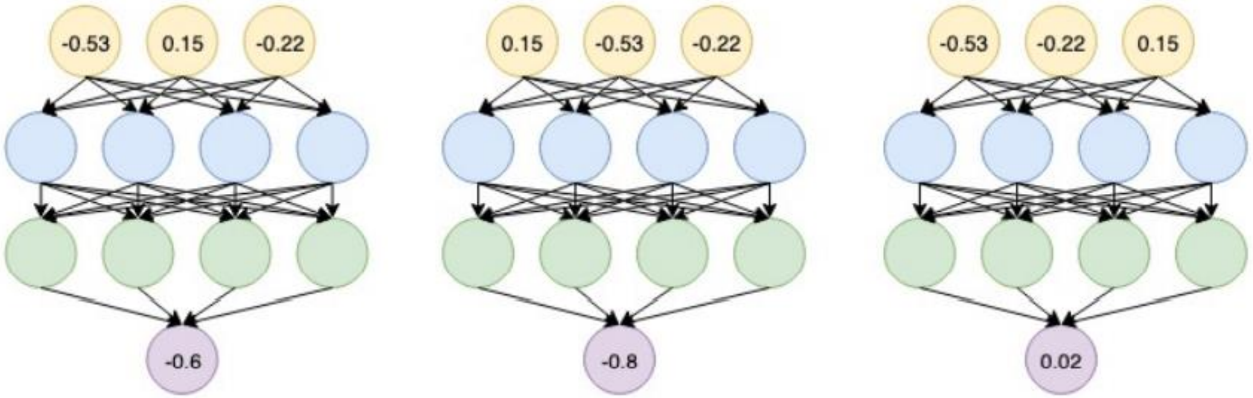


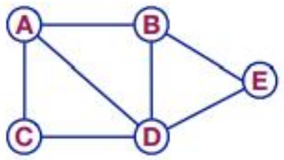


Are MLPs permutation invariance/equivariance?

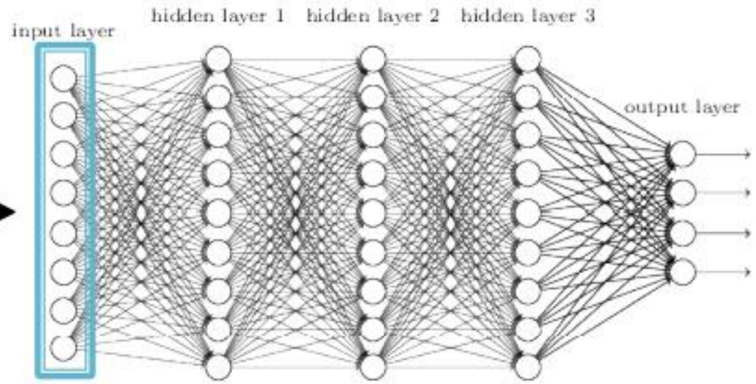
No!

Switching the order of the input leads to different outputs!



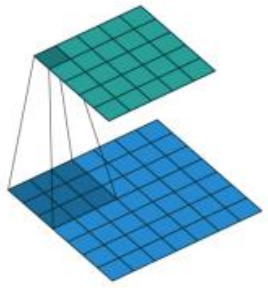


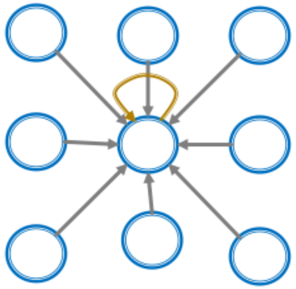
	A	B	C	D	E	Feat	
A	0	1	1	1	0	1	0
B	1	0	0	1	1	0	0
C	1	0	0	1	0	0	1
D	1	1	1	0	1	1	1
E	0	1	0	1	0	1	0



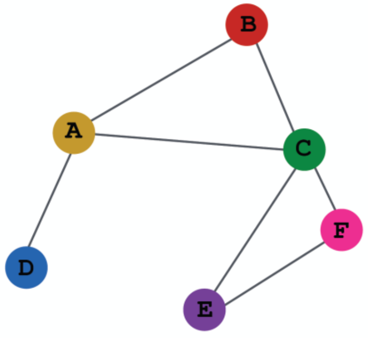
?

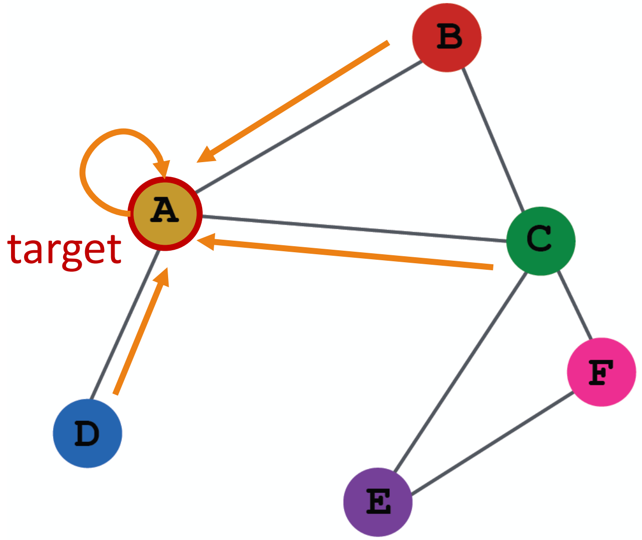
GCN Convolution Operator





CNN layer with 3x3 filter computation flow

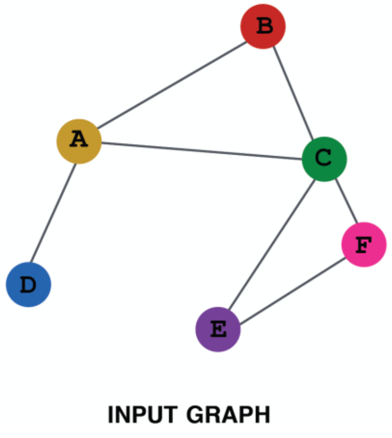


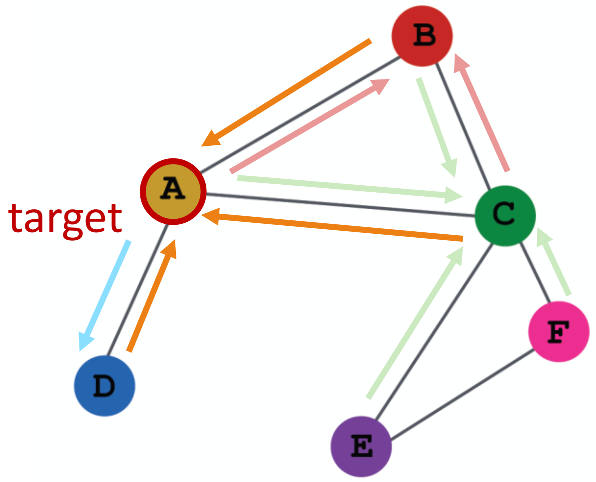


GCN computation flow

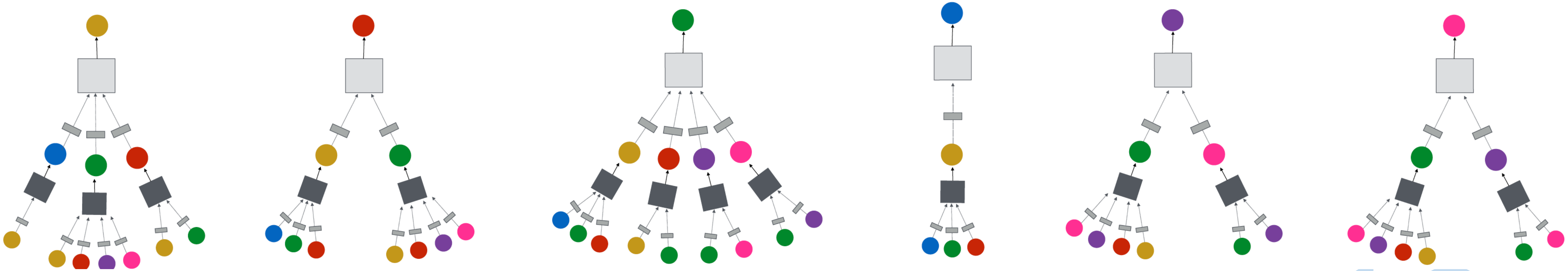
http://web.stanford.edu/class/cs224w/

GCN Computation Flow

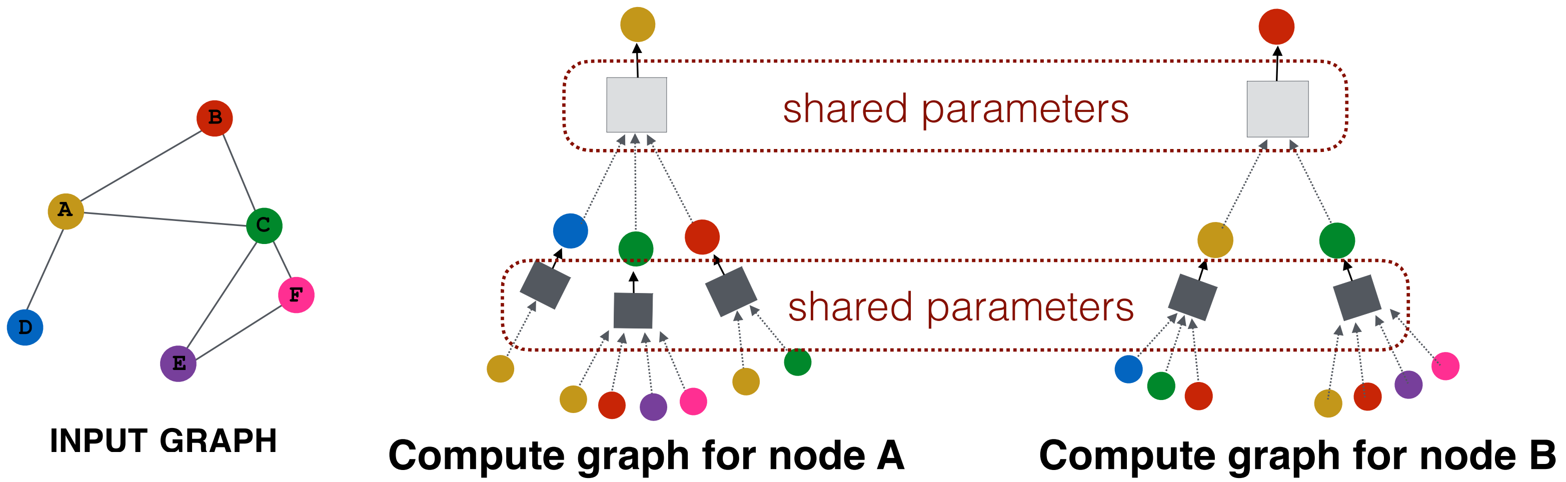




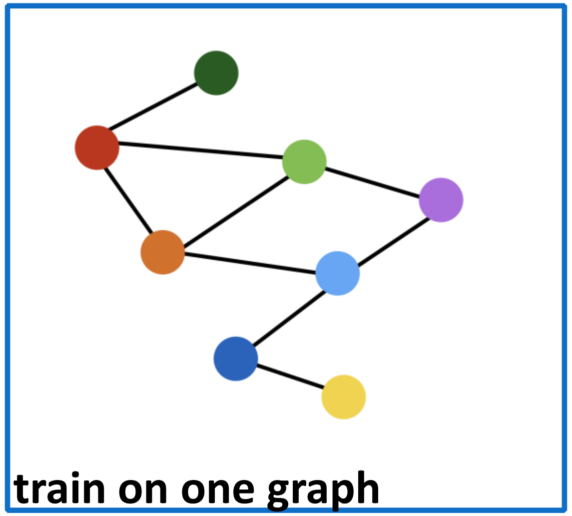
2 layers GCN computation flow

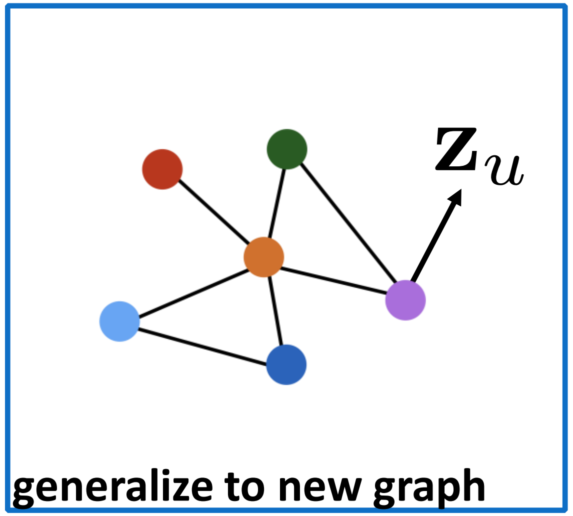


Inductive Capability



Weight Sharing



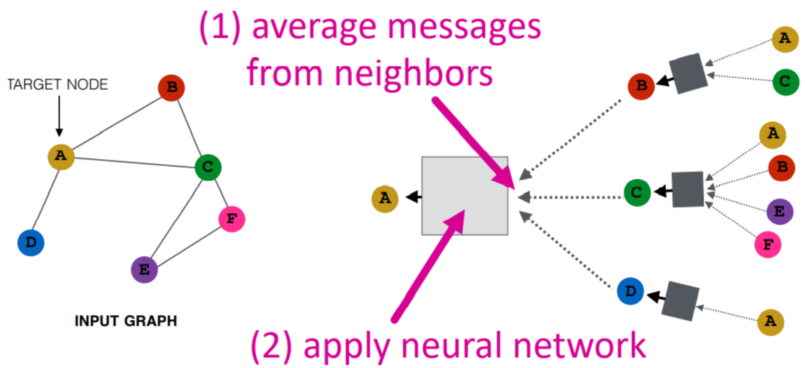


✓ Inductive

Applicable to unseen nodes

A bit of Math

Average information from neighbors and apply a neural network



$$\boxed{h_v^0 = x_v}$$

Initial 0-th layer embeddings are equal to node features

$$h_v^{(k+1)} = \boxed{\sigma}\left(W_k \boxed{\sum_{u \in N(v)} \frac{h_u^{(k)}}{|N(v)|}} + B_k \boxed{h_v^{(k)}}\right), \forall k \in \{0, \ldots, \boxed{K} - 1\}$$

embedding of v at layer k

Total number of layers

Average of neighbor's previous layer embeddings

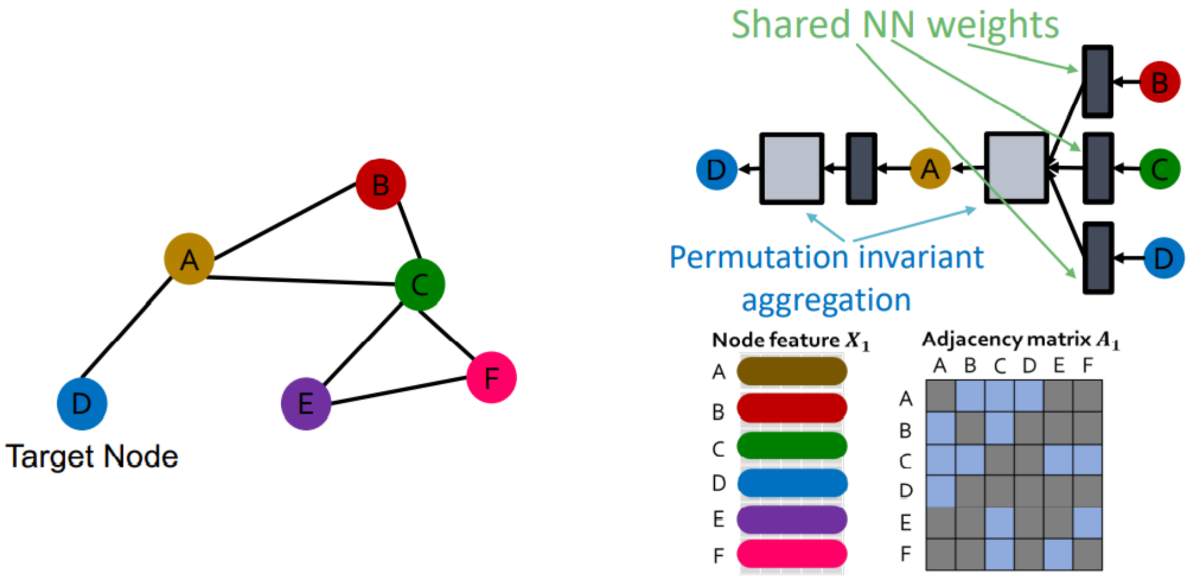
Non-linearity (e.g., ReLU)

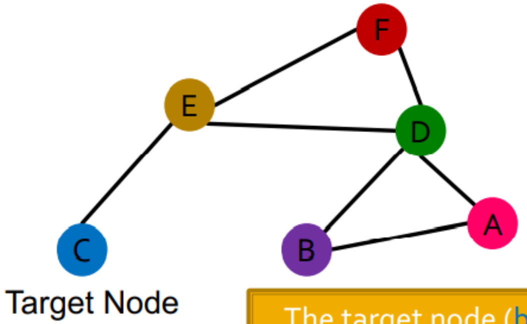
$$\boxed{z_v} = h_v^{(K)}$$

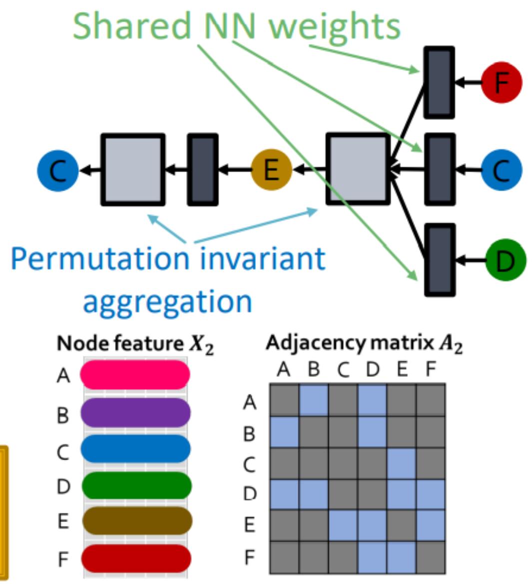
Embedding after L layers of neighborhood aggregation

Notice summation is a permutation invariant pooling/aggregation.

Equivariance property

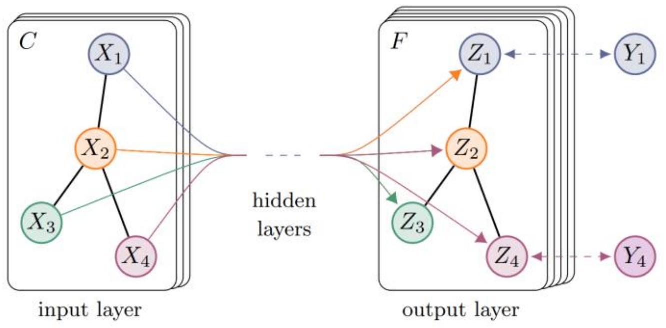




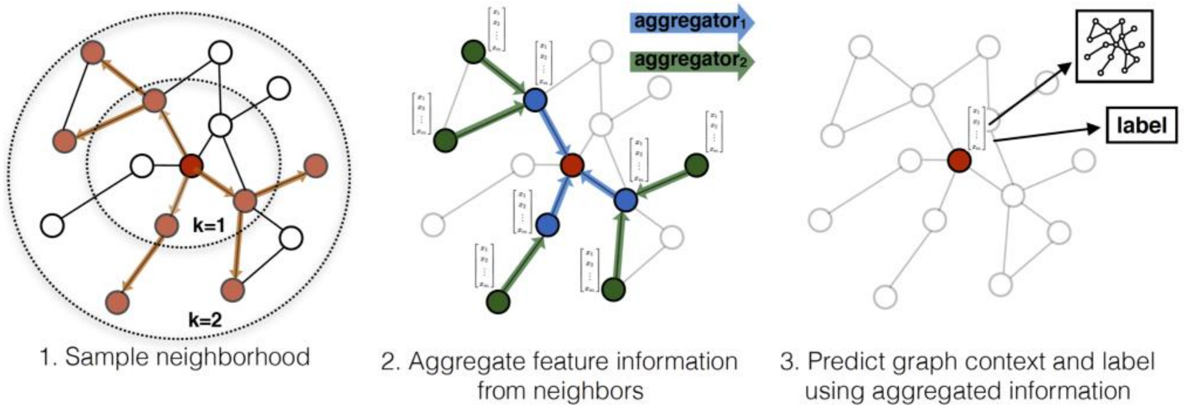


The target node (blue) has the same computation graph for different order plans

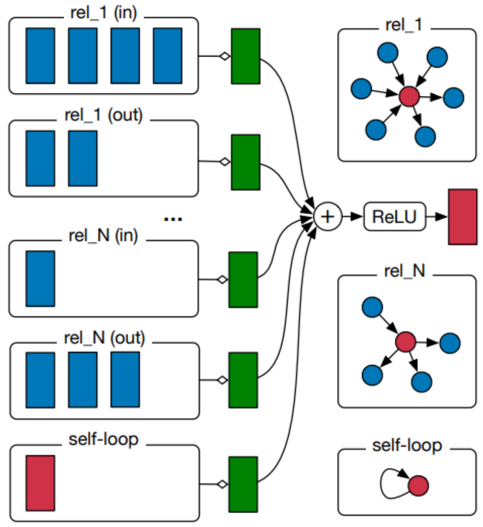
Some Flavors of Graph Neural Networks



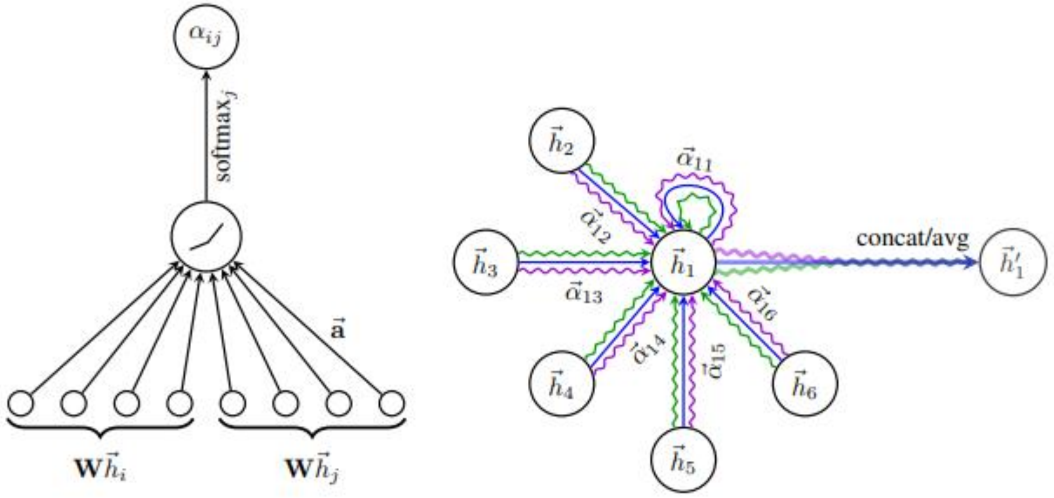
Graph Convolutional Networks (GCN)



GraphSAGE [Sample and Aggregate]

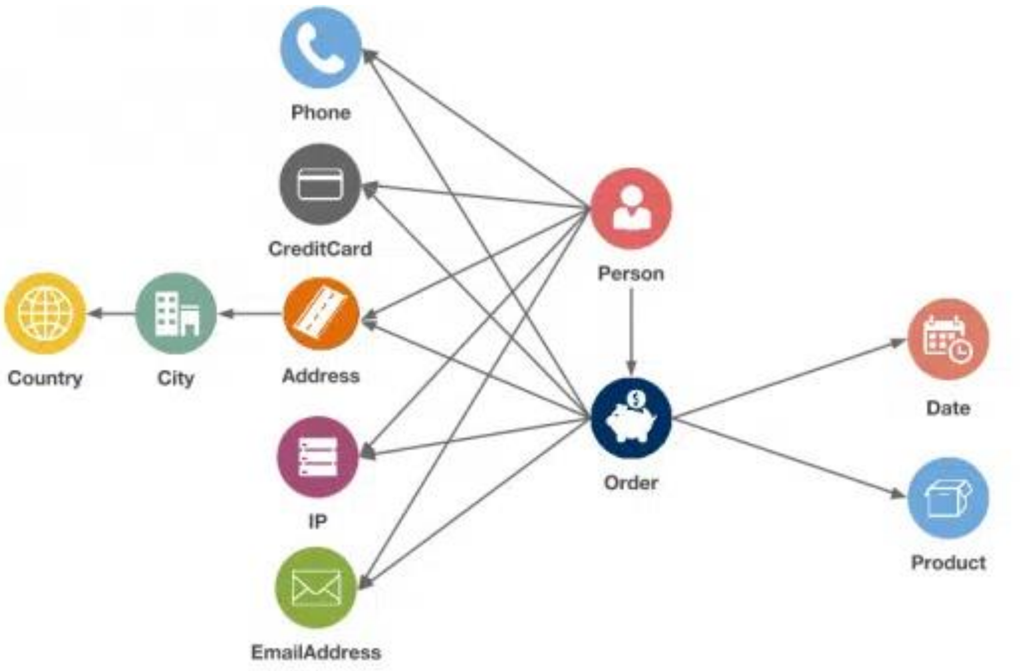


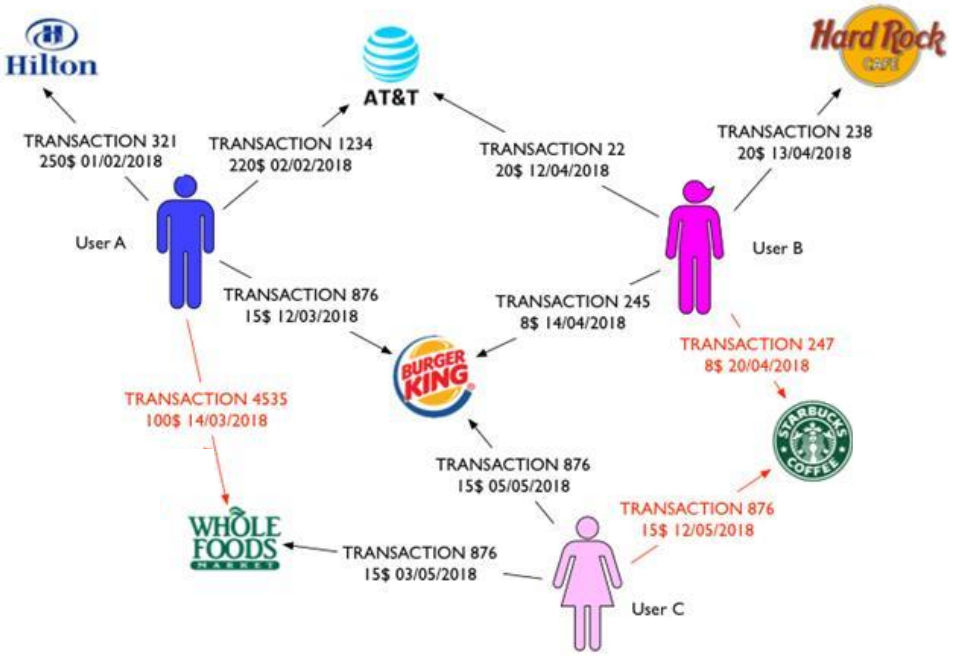
Relational GCN (R-GCN)



Graph Attention Networks (GAT)

Heterogeneous Graph





E-Commerce Data

Nodes: Person, Product, Credit Cards, …

Edges: Has Phone, Has Address, Orders, …

Financial Transactions

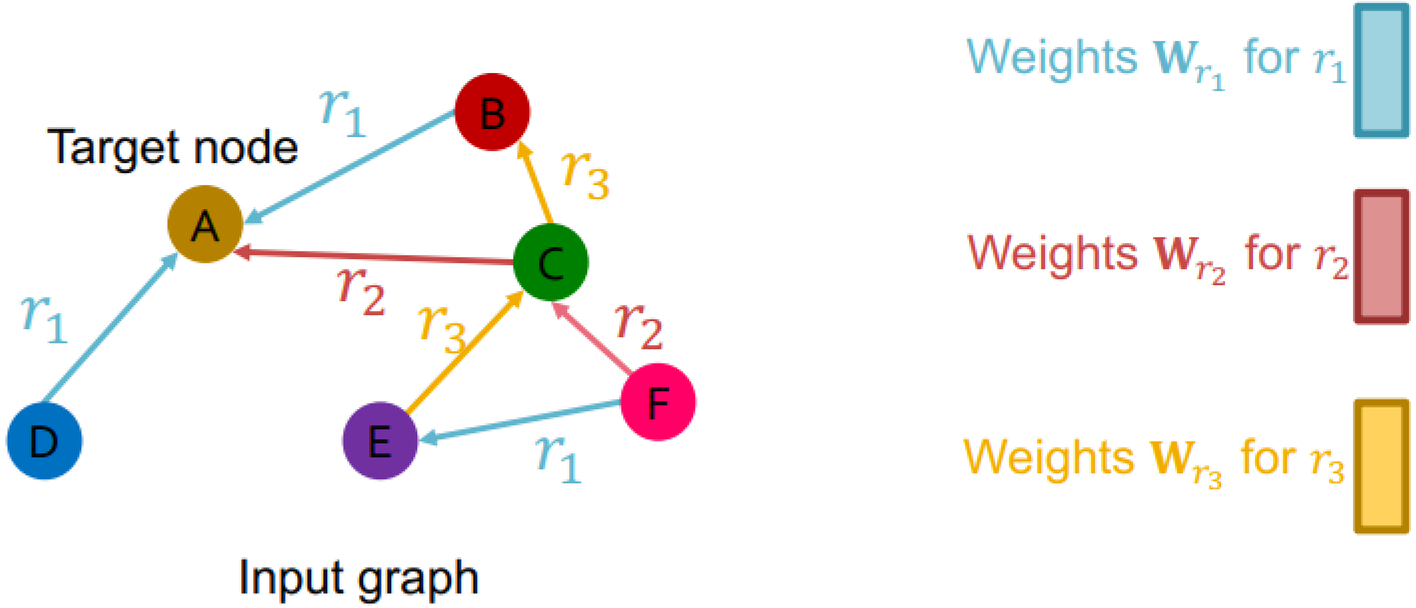
Nodes: Customer, Merchant

Edges: Transaction/Payment

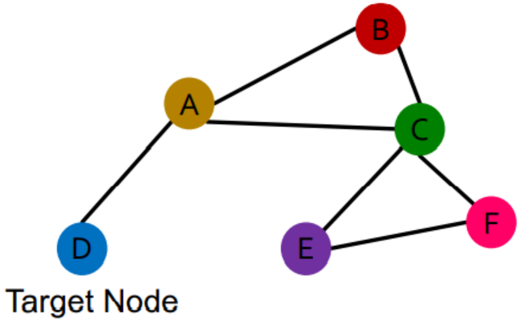
Relational GCN for Heterogeneous Graph

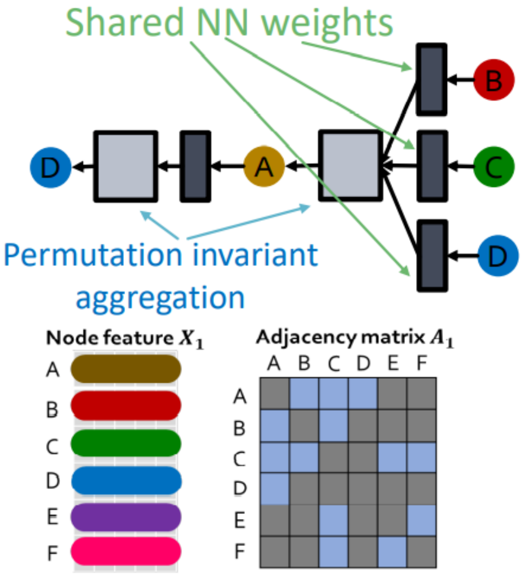
What if the graph has multiple relation types?

Use different neural network weights for different relation types.

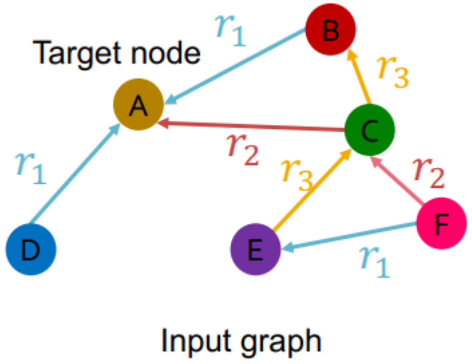


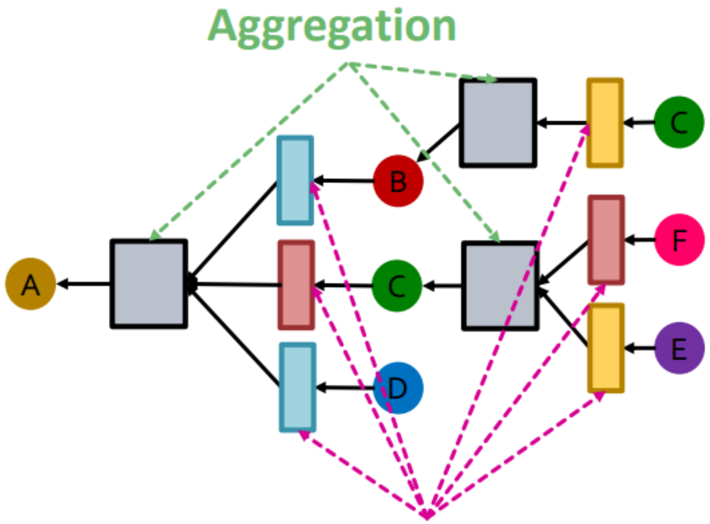
Relational GCN Computation





GCN





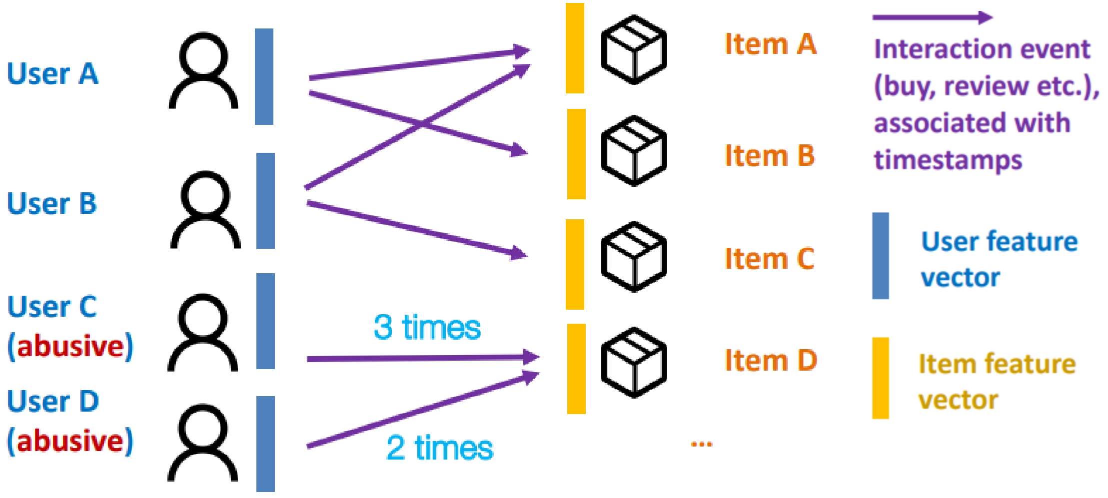
Relational GCN

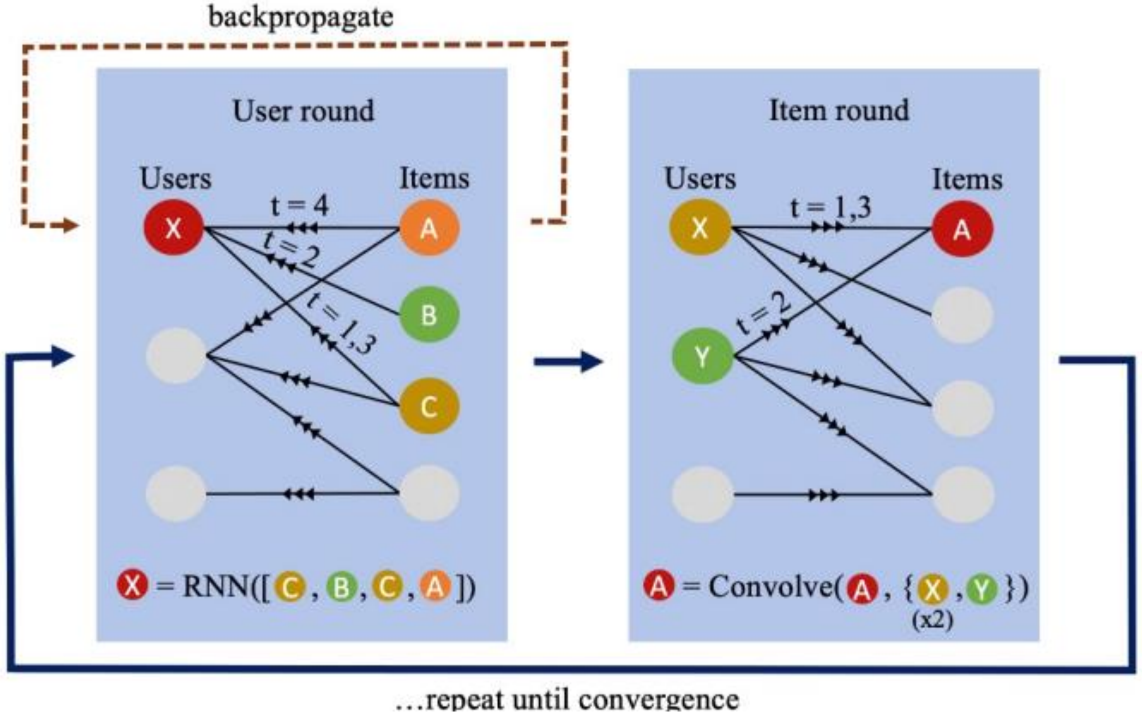
Application: Abuse Detection in Web

Bipartite Dynamic Representations for Abuse Detection (Andrew Wang, et.al, 2021) [KDD | Stanford U, Purdue U, Amazon]

Trolling, propagating misinformation, offensive language

Fake reviews or purchases to inflate product rankings





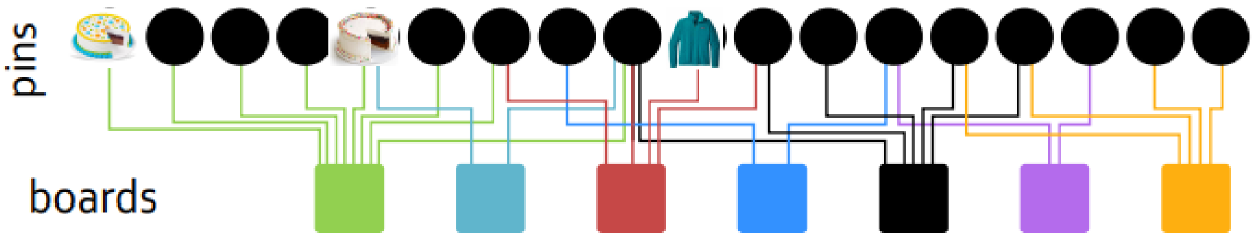
Architecture GCN + RNN

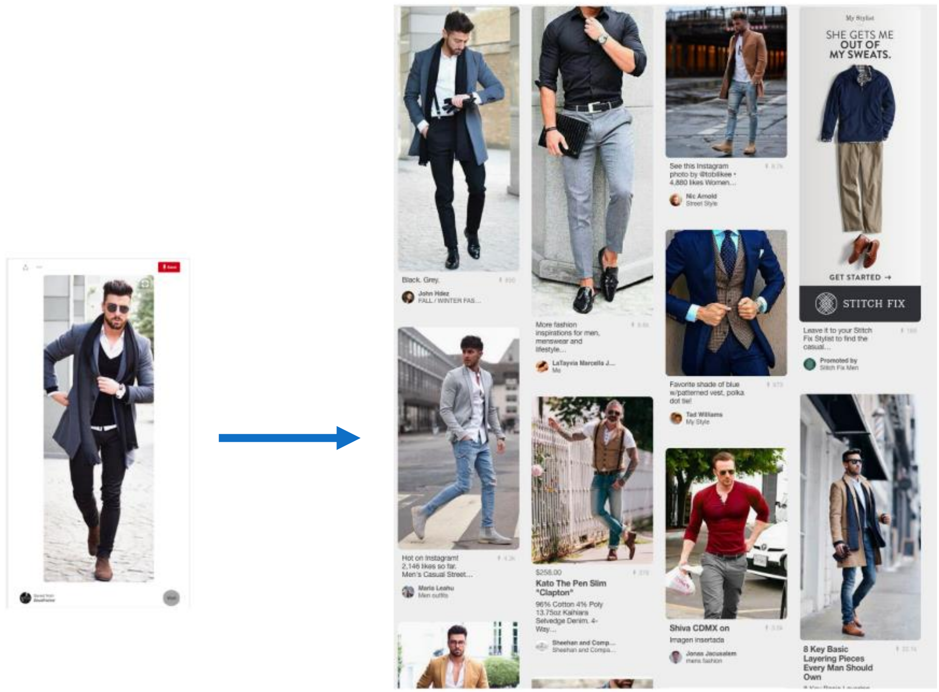
http://snap.stanford.edu/graphlearning-workshop/slides/stanford_graph_learning_Abuse_detection.pdf
https://cs.stanford.edu/people/jure/pubs/bydin-kdd21.pdf

Application: PinSAGE, Pinterest's Recommendation System

Large-scale GCN/GraphSAGE implementation
Contextual Image Recommendation

Graph Convolutional Neural Networks for Web-Scale Recommender Systems (Rex Ying et.al, 2018) [KDD | Pinterest, Stanford]





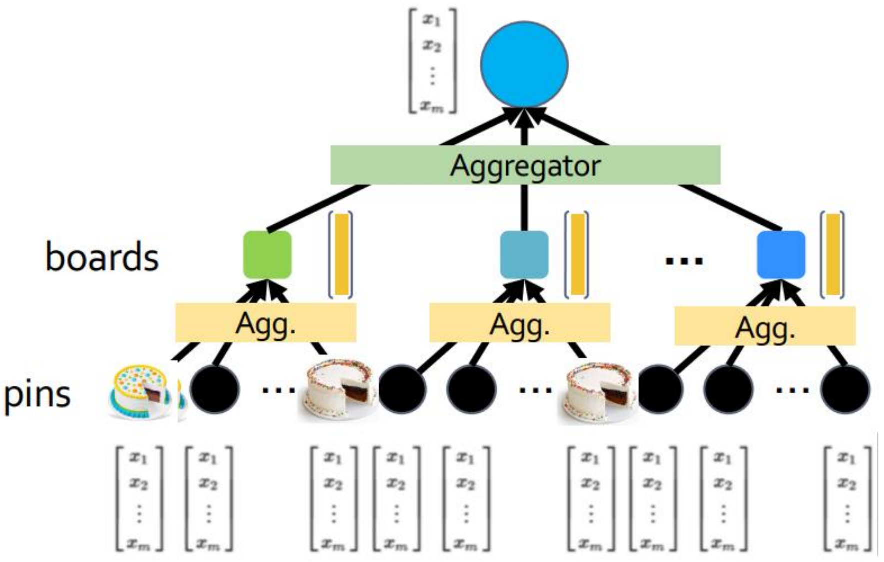
Pin
(image + desc.)

Recommend
related pins

7.5 billion
training data

1.2 billion
positive pairs

6.5 billion
negative pairs



Features: image embedding + text embedding

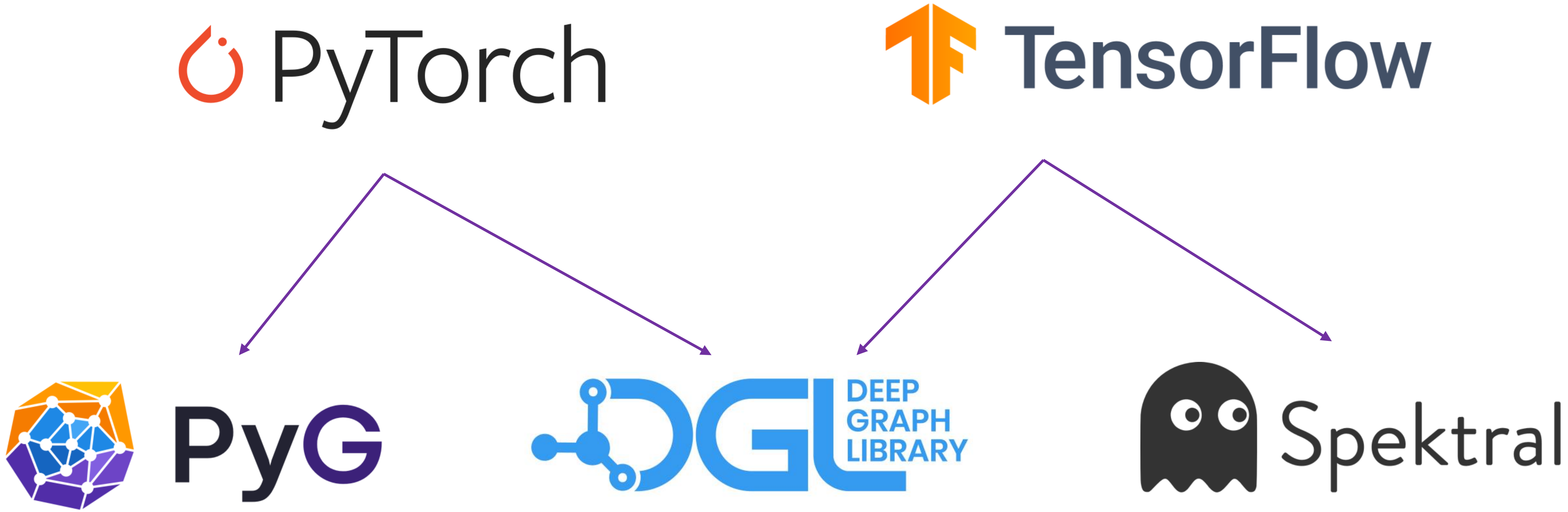
https://medium.com/pinterest-engineering/pinsage-a-new-graph-convolutional-neural-network-for-web-scale-recommender-systems-88795a107f48
https://arxiv.org/pdf/1806.01973.pdf

Implementation

Tools and Frameworks

Tools for Graph Neural Networks

Deep Learning Frameworks



Graph Neural Network Frameworks

Code Example

PyTorch PyG

PyG is PyTorch-on-the-rocks:

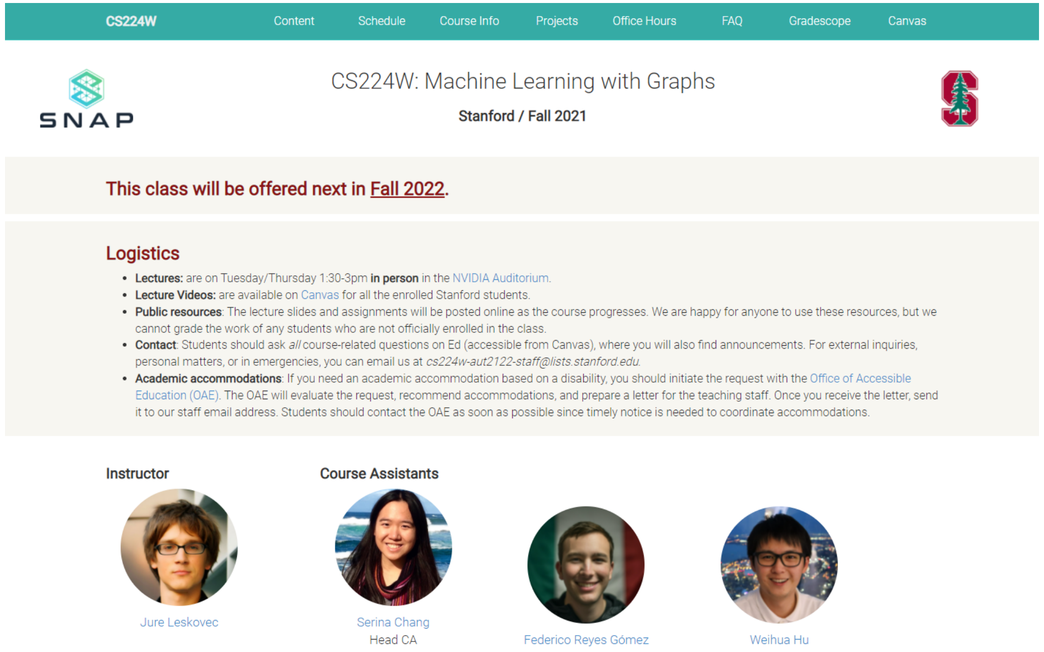
```python
from torch.nn import Conv2d

class CNN(torch.nn.Module):
    def __init__(self):
        self.conv1 = Conv2d(3, 64)
        self.conv2 = Conv2d(64, 64)

    def forward(self, input):
        h = self.conv1(input)
        h = h.relu()
        h = self.conv2(h)
        return h
```

```python
from torch_geometric.nn import GCNConv

class GNN(torch.nn.Module):
    def __init__(self):
        self.conv1 = GCNConv(3, 64)
        self.conv2 = GCNConv(64, 64)

    def forward(self, input, edge_index):
        h = self.conv1(input, edge_index)
        h = h.relu()
        h = self.conv2(h, edge_index)
        return h
```

http://snap.stanford.edu/graphlearning-workshop/slides/stanford_graph_learning_PyG_2_0.pdf

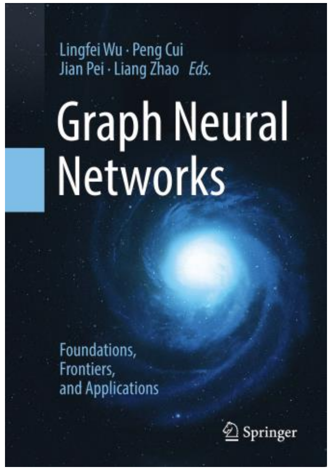
Learning & Resources

Stanford's Machine Learning with Graphs class



Course Slides, Video Lectures

Comprehensive resources for Graph ML from ***Jure Leskovec***, one of the authorities on Graph ML

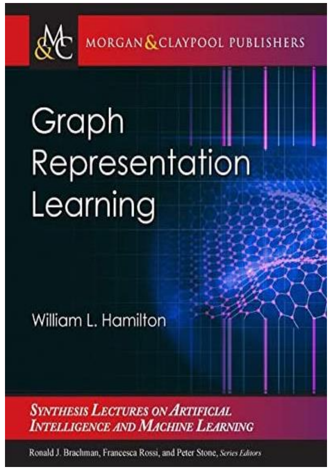


Graph Neural Networks Foundation, Frontier, and Applications

Lingfei Wu et. al.

Comprehensive, focus on applications and use cases

Free pre-print version is available



Graph Representation Learning

William L. Hamilton

Foundational, focus on building conceptual understanding

Free pre-print version is available

Thank You