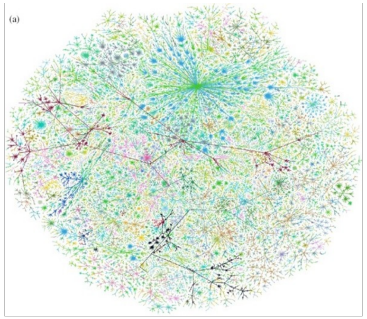
Graph Neural Networks

Yogesh Kulkarni

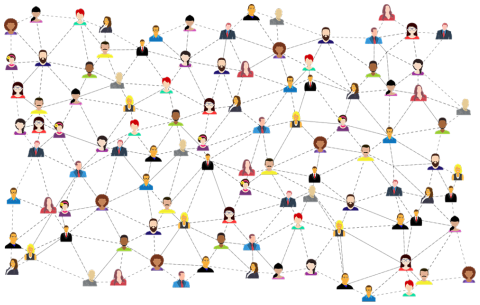
August 5, 2022

Introduction

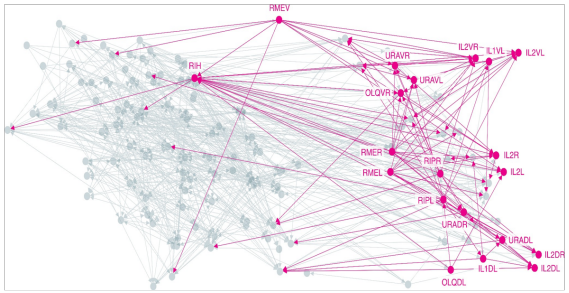
Graph-structured Data Are Ubiquitous



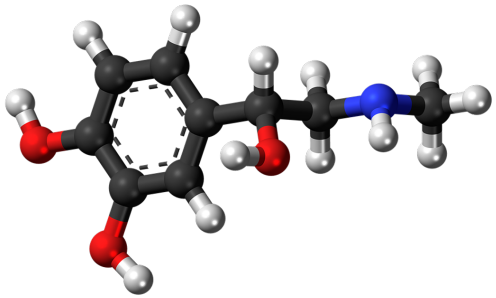
Internet



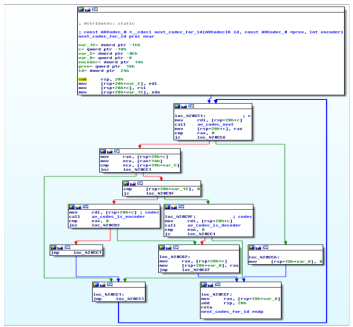
Social networks



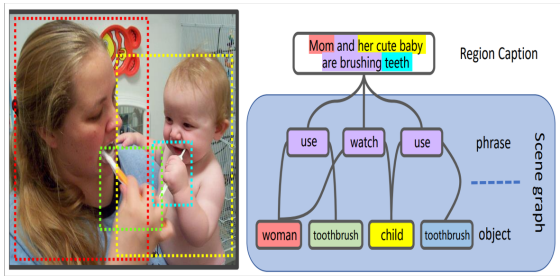
Networks of neurons



Biomedical graphs

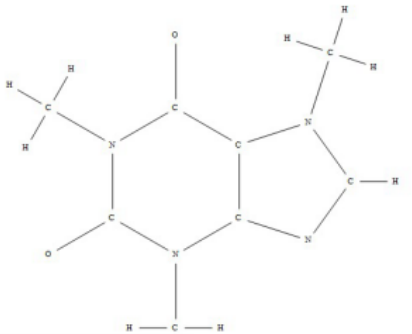


Program graphs

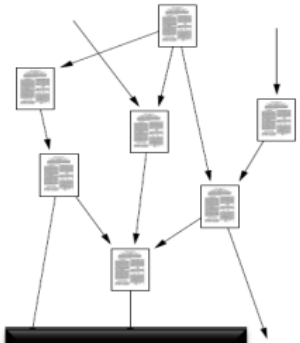


Scene graphs

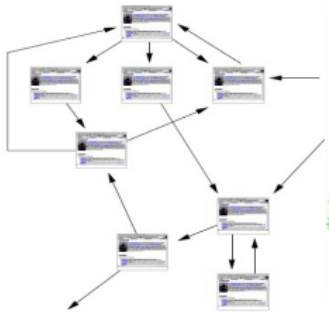
Networks around us!



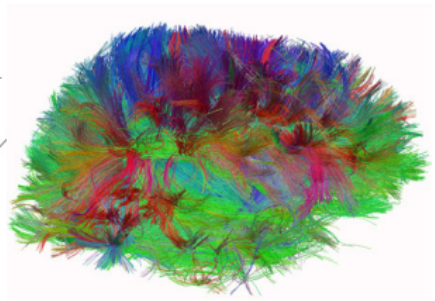
Molecules



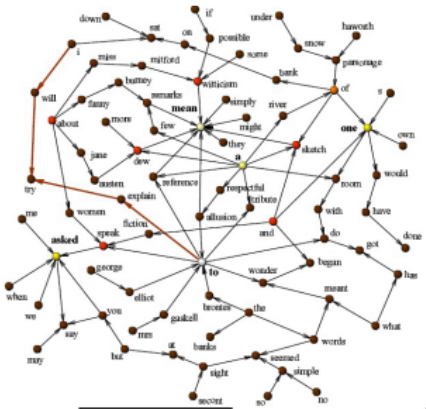
Knowledge



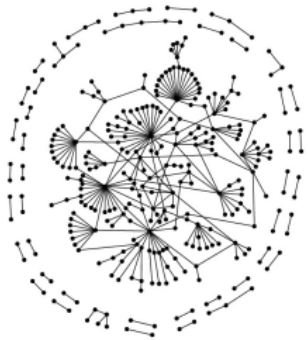
Information



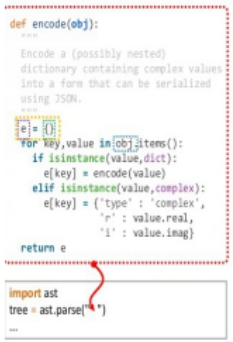
Brain/neurons



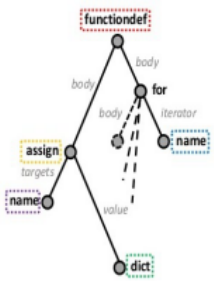
Genes



Communication



Software

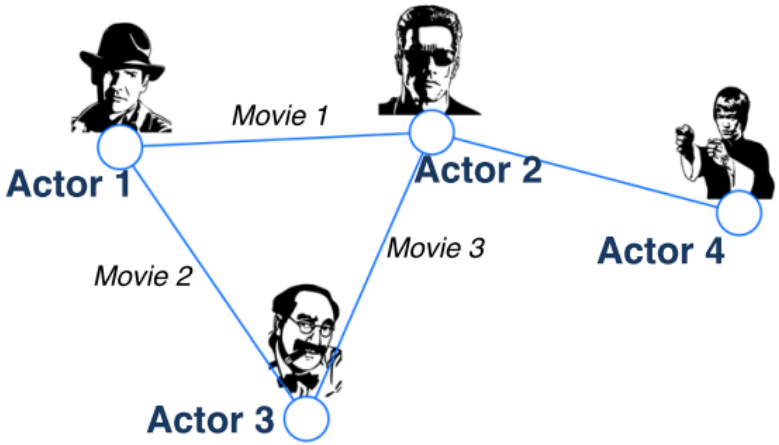


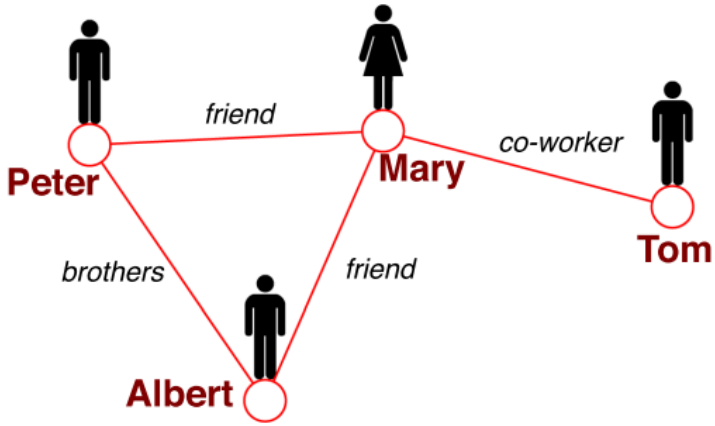
Social

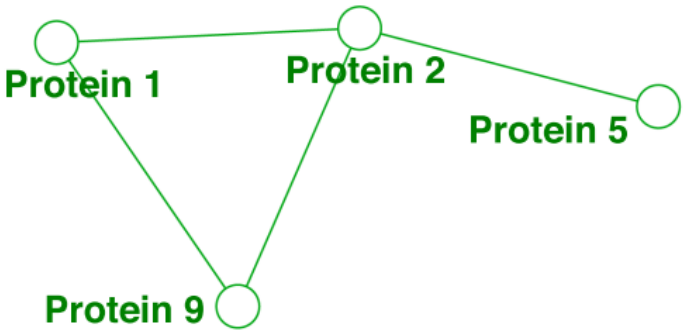
Jure Leskovec (@jure), Stanford University

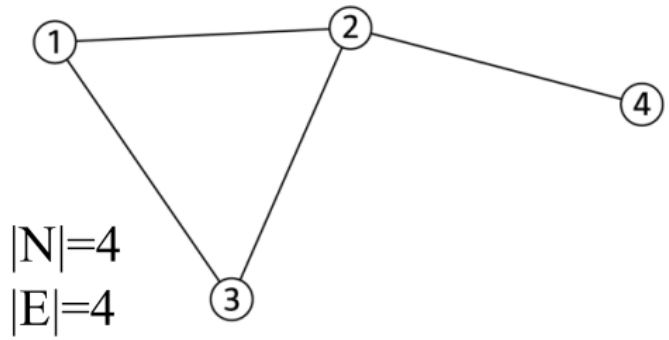
2

Graphs: Common Language



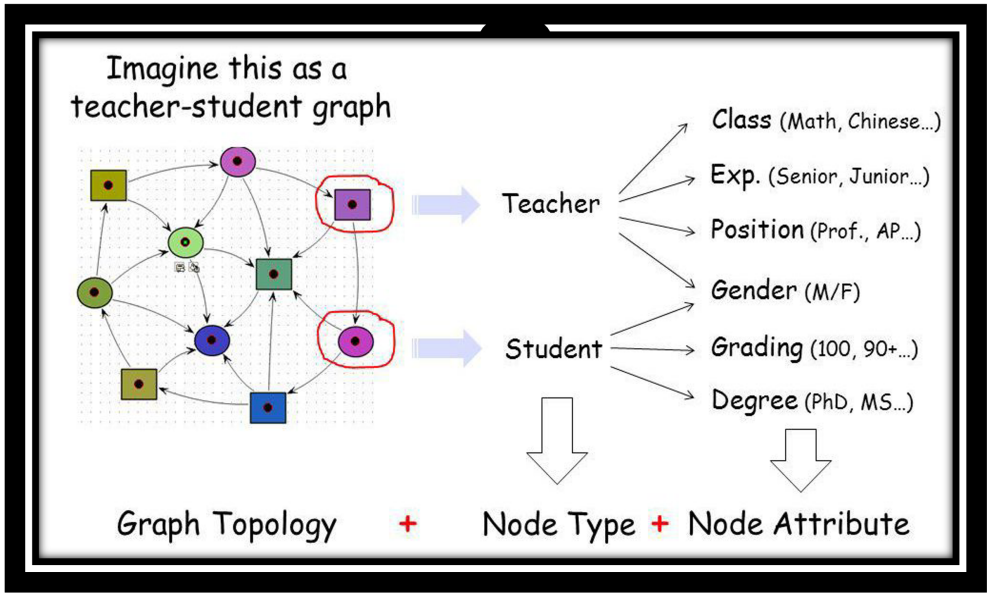






Graphs: A Universal Language

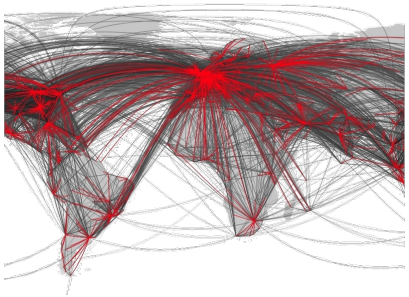
Graphs are a general language for describing and modeling complex systems



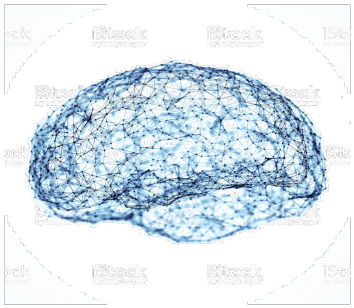
Graph!

Data as Graphs - Explicit

Social Graphs

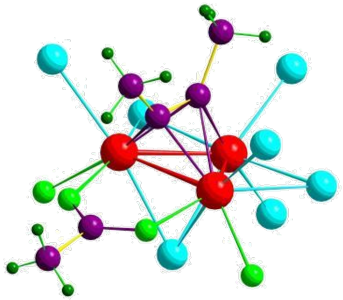


Transportation Graphs

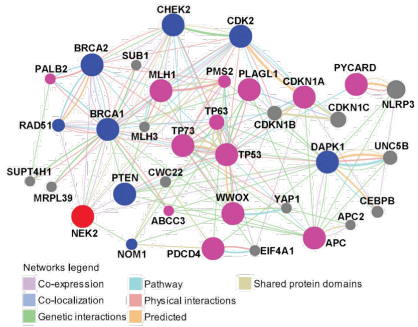


Brain Graphs

Web Graphs

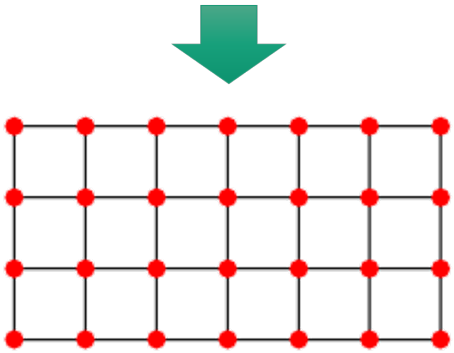


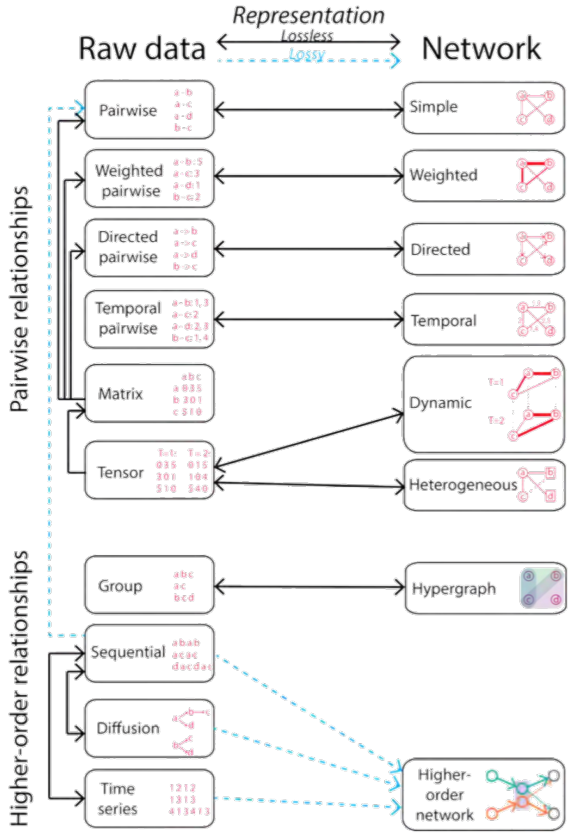
Molecular Graphs



Gene Graphs

Data as Graphs - Implicit





Jian Xu. Representing Big Data as Networks. PhD Dissertation, University of Notre Dame

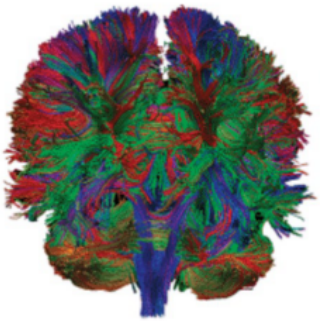
Applications of DL on Graphs

Computer graphics

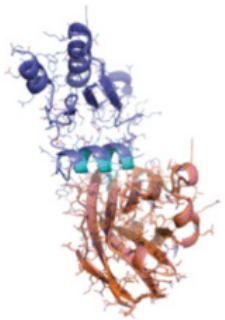
Virtual/augmented reality

Robotics

Autonomous driving



Medicine



Drug design

Jure Leskovec (@jure), Stanford University

3

Why Graphs? Why Now?

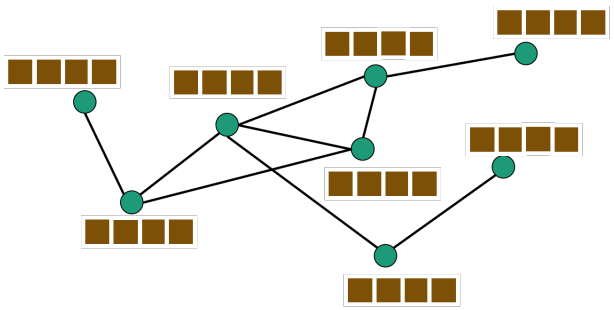
- Universal language for describing complex data: Networks/graphs from science, nature, and technology are more similar than one would expect
- Shared vocabulary between fields: Computer Science, Social science, Physics, Biology, Economics
- Data availability (+ computational challenges): Social/Internet, text, logic, program, bio, health, and medical
- Impact: Social networking, Social media, Drug design, Event detection, Natural language processing, Computer vision, and Logic reasoning

Homogeneous vs Multi-relational vs Heterogeneous Graphs

Graph types	Homogeneous	Multi-relational	Heterogeneous
# of node types	1	1	> 1
# of edge types	1	> 1	>= 1

64

Graphs and features



$$\mathcal{V} = \{v_1, \ldots, v_N\}$$

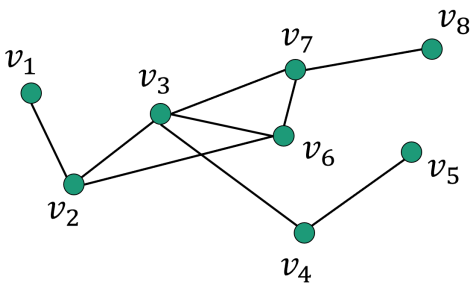
$$\mathcal{E} = \{e_1, \ldots, e_M\}$$

$$\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$$

Graph Signal: $f : \mathcal{V} \rightarrow \mathbb{R}^{N \times d}$

$$v \longrightarrow \begin{bmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{bmatrix}$$

Matrix Representations of Graphs



Adjacency Matrix: $A[i,j] = 1$ if v_i is adjacent to v_j

$A[i,j] = 0$, otherwise

Degree Matrix: $\mathbf{D} = \text{diag}(degree(v_1), \ldots, degree(v_N))$

Degree Matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

D

$-$

Adjacency Matrix

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

A

$=$

Laplacian Matrix

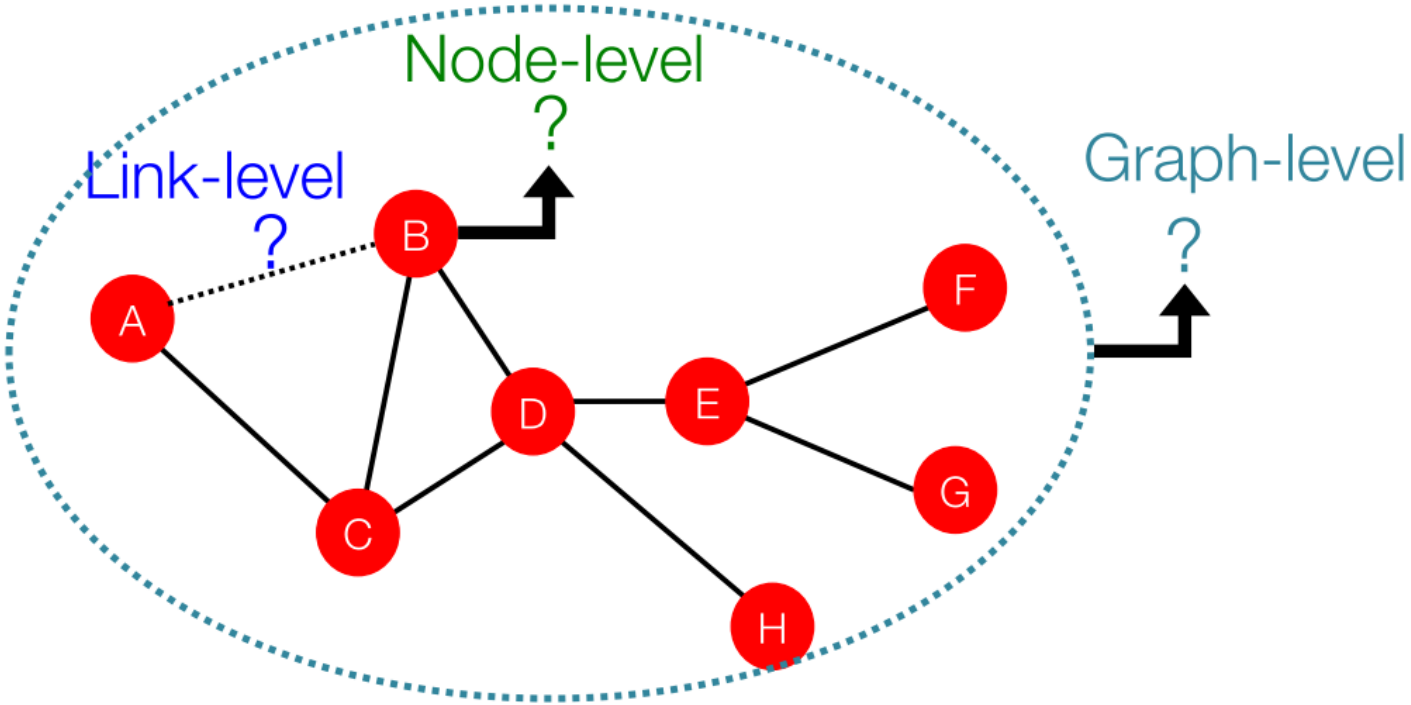
$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

L

12

Spectral graph theory. American Mathematical Soc.; 1997.

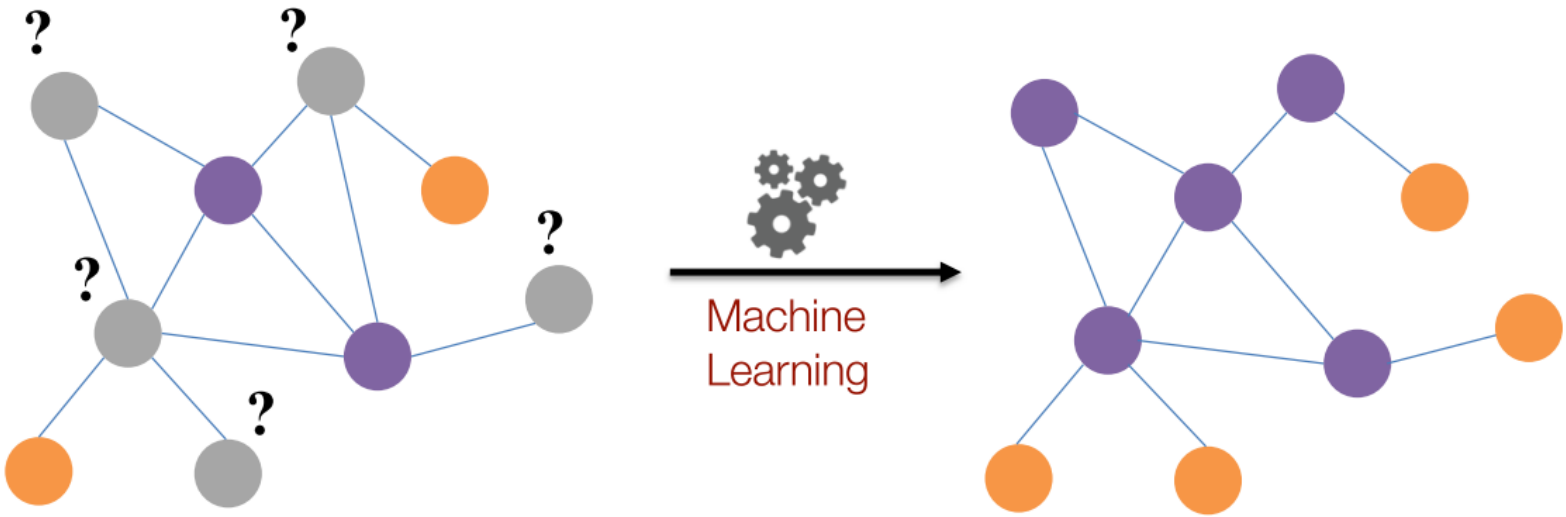
Machine Learning Tasks



Jure Leskovec (@jure), Stanford University

8

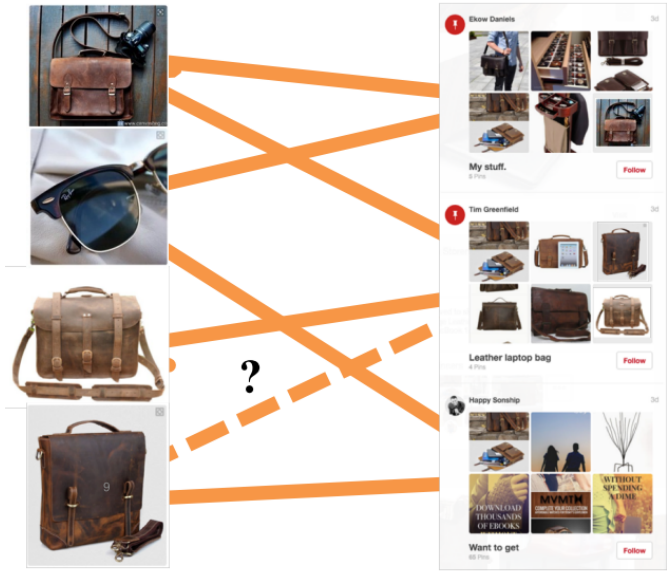
Example: Node Classification



- What users are going to churn?
- What is the disease of a patient?
- What are functions of proteins?

Link Prediction: An Example

Content recommendation is link prediction!



9

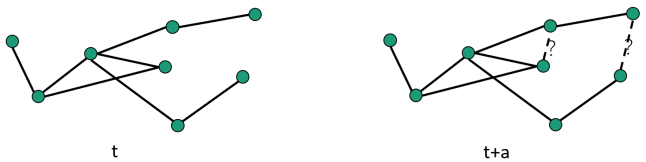
Tasks in graph learning

- Node classification
  - Detect malicious accounts
  - Target right customers
- Link prediction
  - Recommendations
  - Predict missing relations in a knowledge graph
- Graph classification
  - Predict the property of a chemical compound

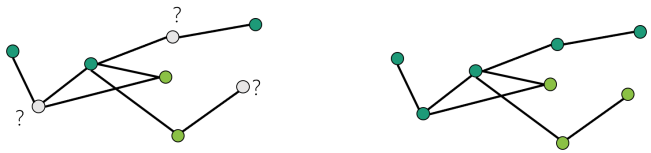
Tasks on Graph-Structured Data

Node-level

Link Prediction

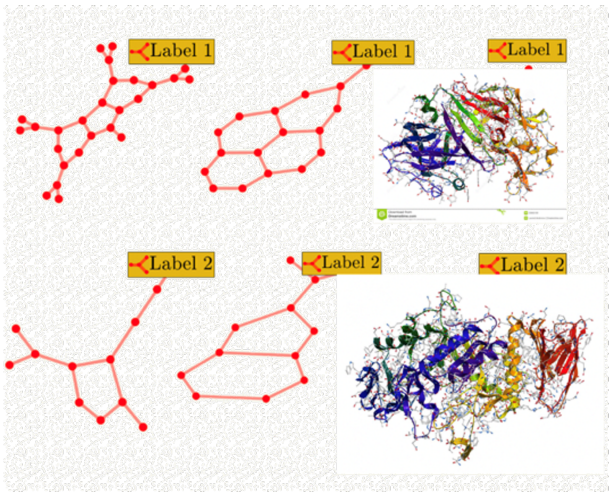


Node Classification



Graph-level

Graph Classification

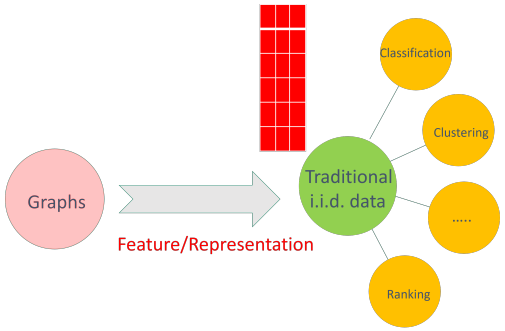


ML on Graphs

Numerous real-world problems can be summarized as a set of tasks on graphs

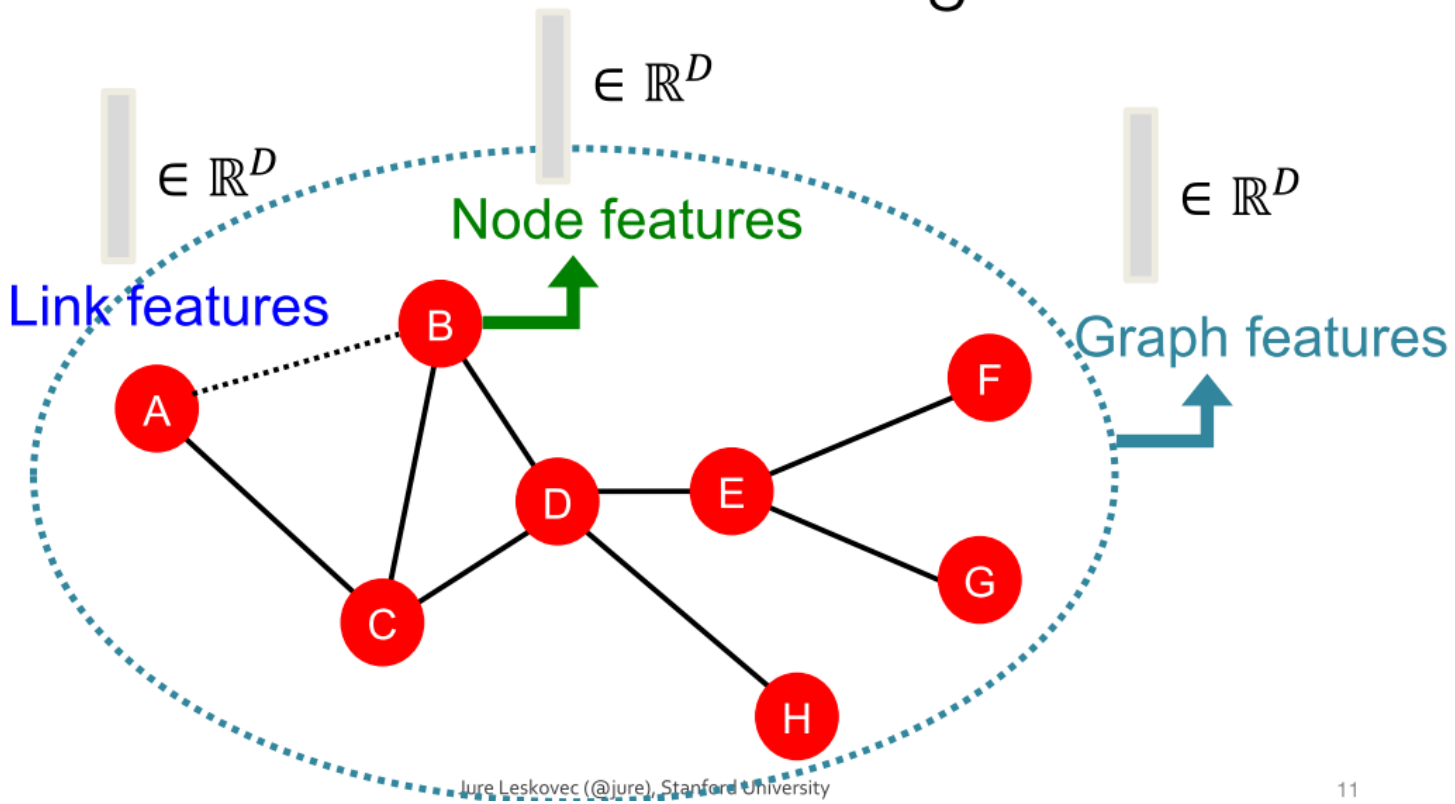
- Link prediction
- Node Classification
- Community Detection
- Ranking . . .

ML solutions



Graph Feature Engineering

- Design features for nodes/links/graphs
- Obtain features for all training data



Two Pain Points: One

Data Scientist's pain point #1:

- Data scientists have to hand encode features to solve prediction problems.
- Hand encoding graph features is…

… complex and involves expensive queries

… error prone

… suboptimal

… labor intensive

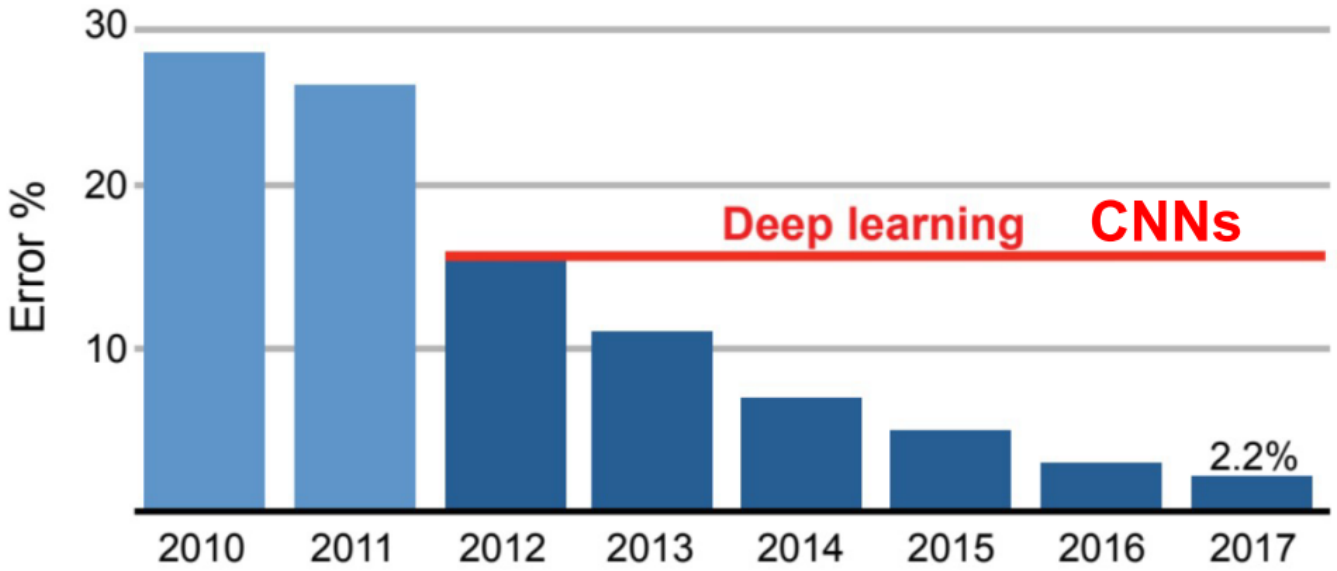
Two Pain Points: Two

Data Scientist's pain point #2:

- Data is often incomplete.
  - Address Books, Follows, Interests, Protein Protein Interaction, Ancestry
- Entity information is incomplete.
- Predictions often entail completing the "missing information".
  - Relational structure is often not leveraged due to scalability issues.

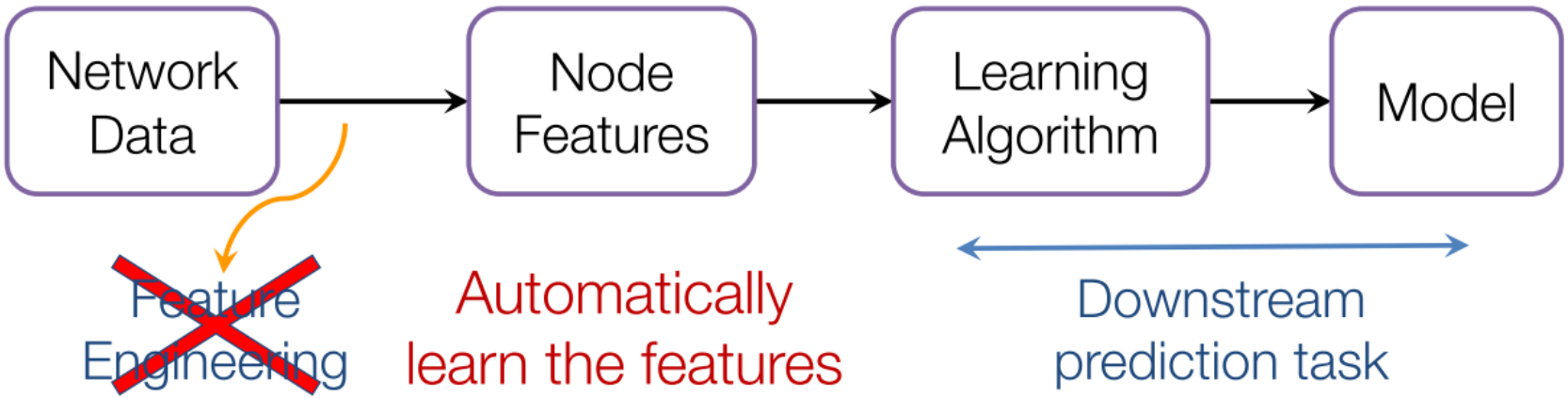
The Deep Learning Revolution

Breakthroughs in image recognition fueled by Convolutional Neural Networks.



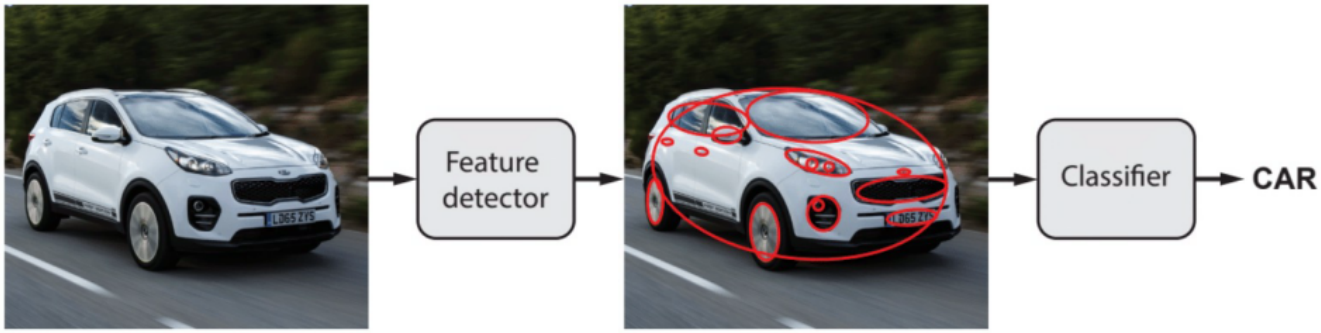
Jure Leskovec (@jure), Stanford University 16

Machine Learning Lifecycle

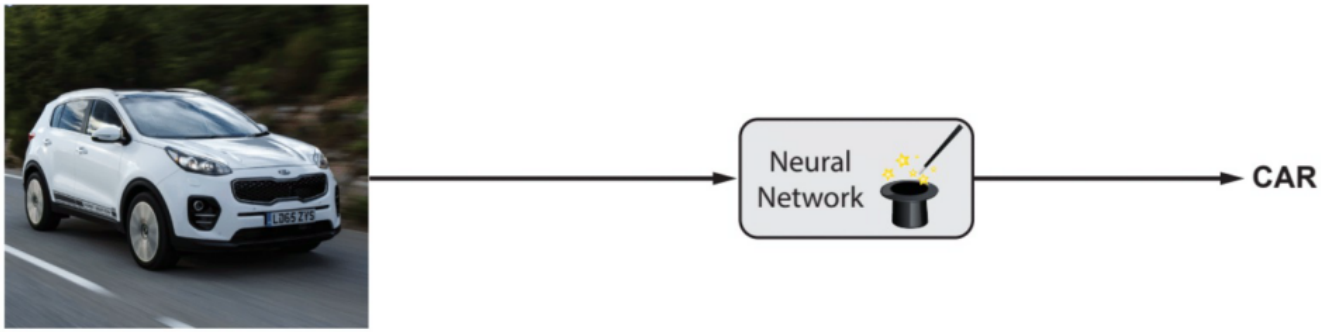


(Supervised) Machine Learning Lifecycle:
This feature, that feature.
Every single time!

Representation Learning



Classical computer vision: hand-crafted features (e.g. SIFT)
+ simple classifier (e.g. SVM)



Modern computer vision: data-driven end-to-end systems

Doubt thou the stars are fire;
Doubt that the sun doth move;
Doubt truth to be a liar;
But never doubt I love...

Text

Audio signals

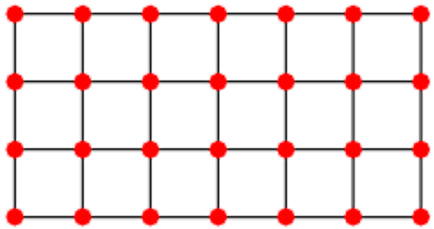
Images

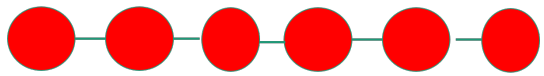
But, modern
deep learning toolbox
is designed for
sequences & grids

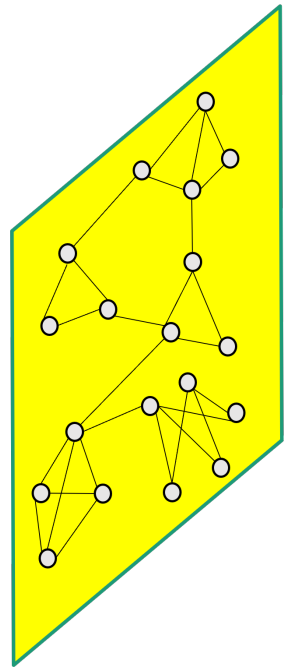
Deep Learning Meets Graphs: Challenges

Traditional DL is designed for simple grids or sequences

- CNNs for fixed-size images/grids
- RNNs for text/sequences





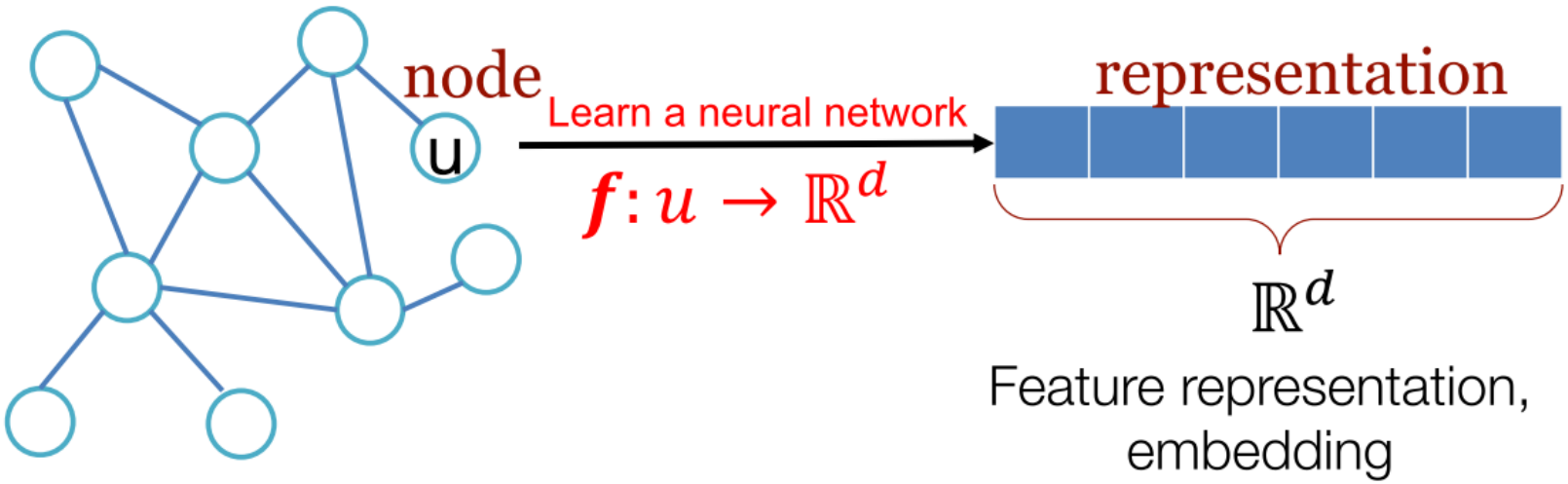


But nodes on graphs have different connections

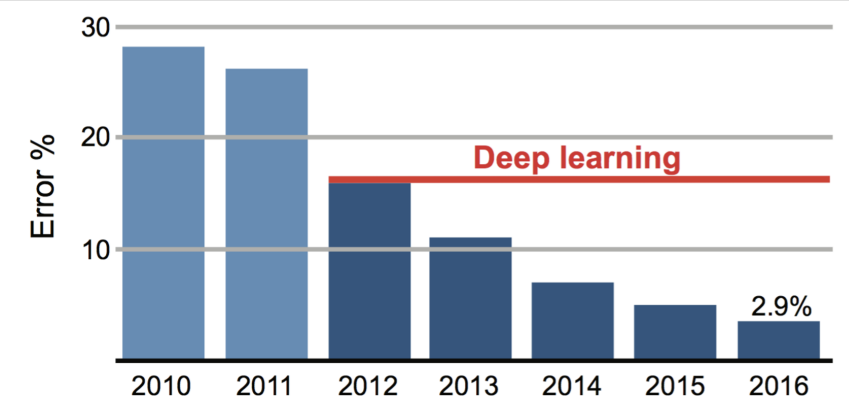
- Arbitrary neighbor size
- Complex topological structure
- No fixed node ordering

Goal: Representation Learning

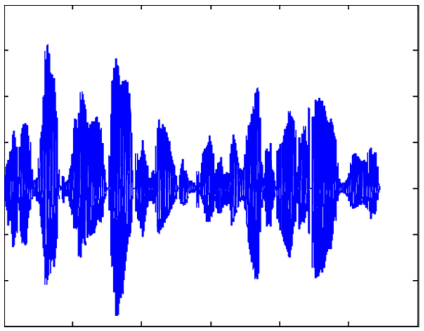
Map nodes to d-dimensional **embeddings** such that similar nodes in the network are embedded close together



The Power of Deep Learning

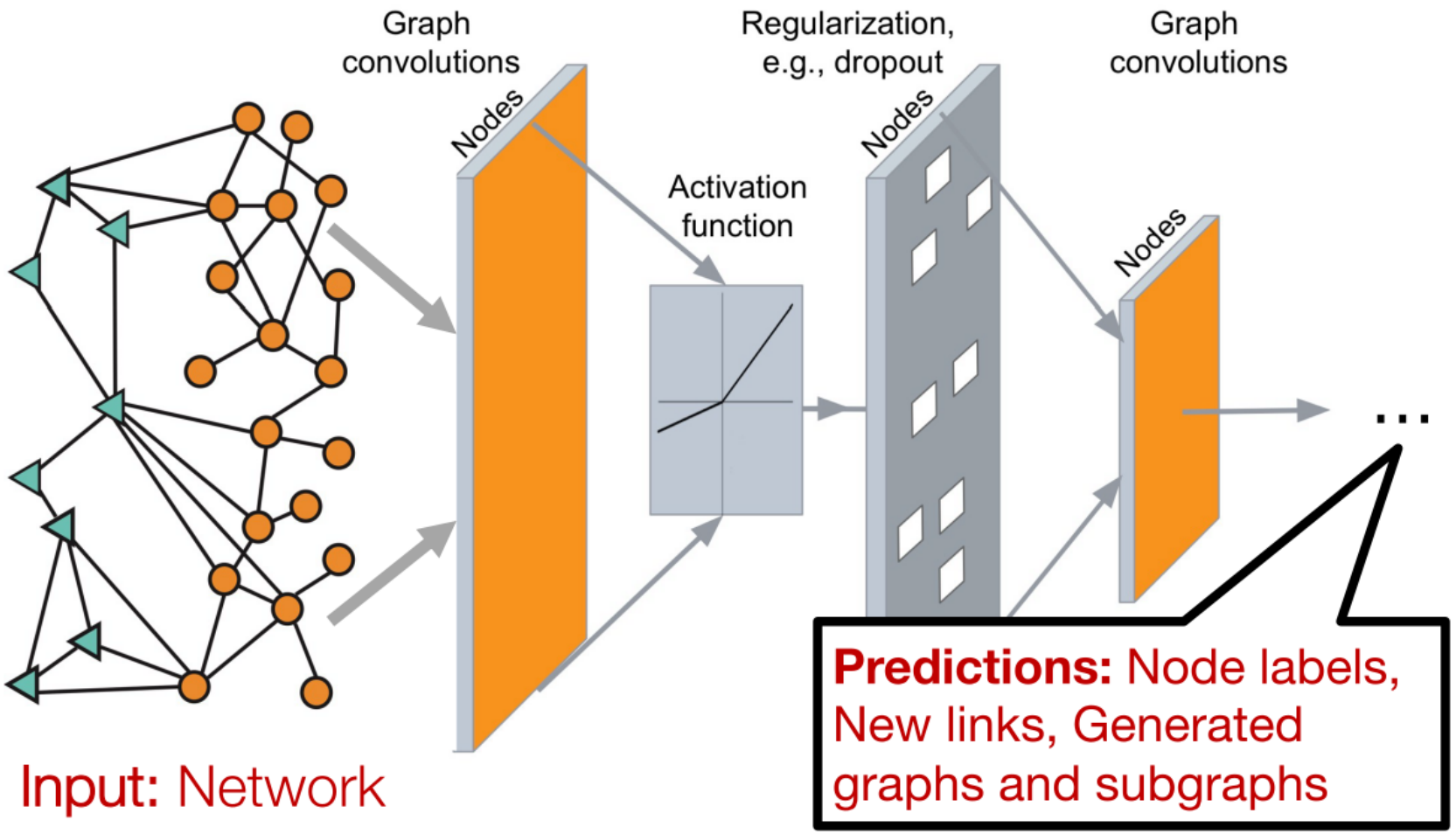


IMAGENET



Acoustic model	Recog \ WER	RT03S FSH	Hub5 SWB
Traditional features	1-pass –adapt	27.4	23.6
Deep Learning	1-pass –adapt	18.5	16.1

Deep Learning in Graphs

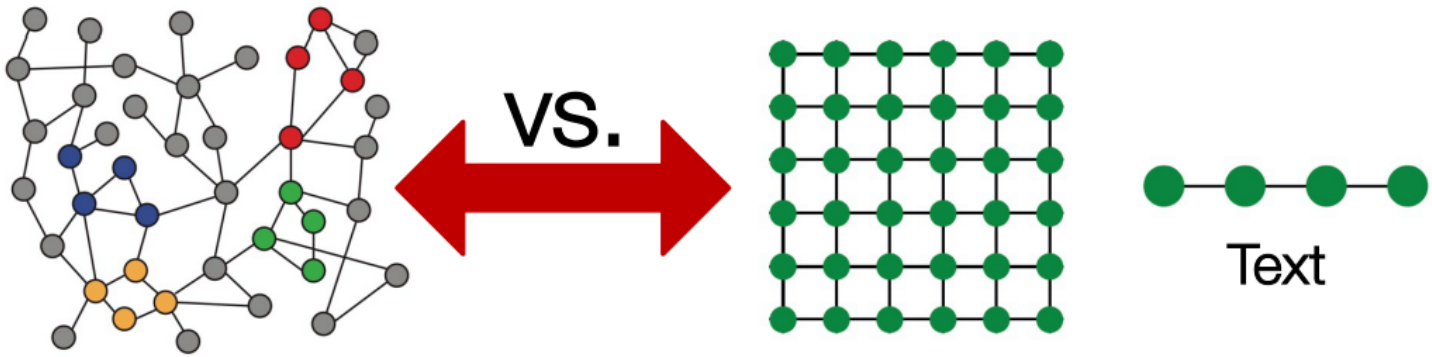


Jure Leskovec (@jure), Stanford University 21

Why is it Hard?

Networks are complex!

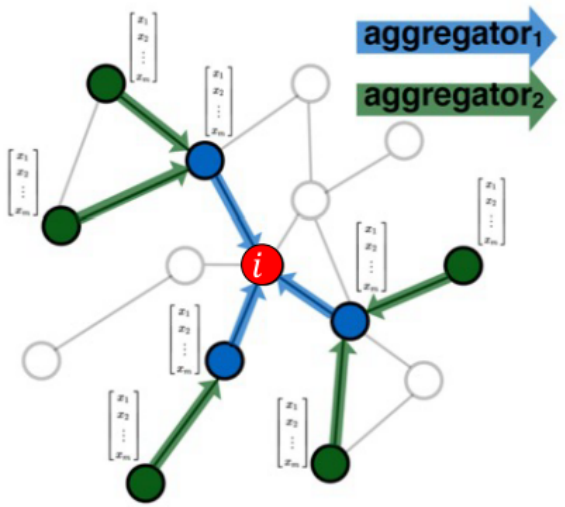
- Arbitrary size and complex topological structure (i.e., no spatial locality like grids)



- No fixed node ordering or reference point
- Often dynamic and have multimodal features

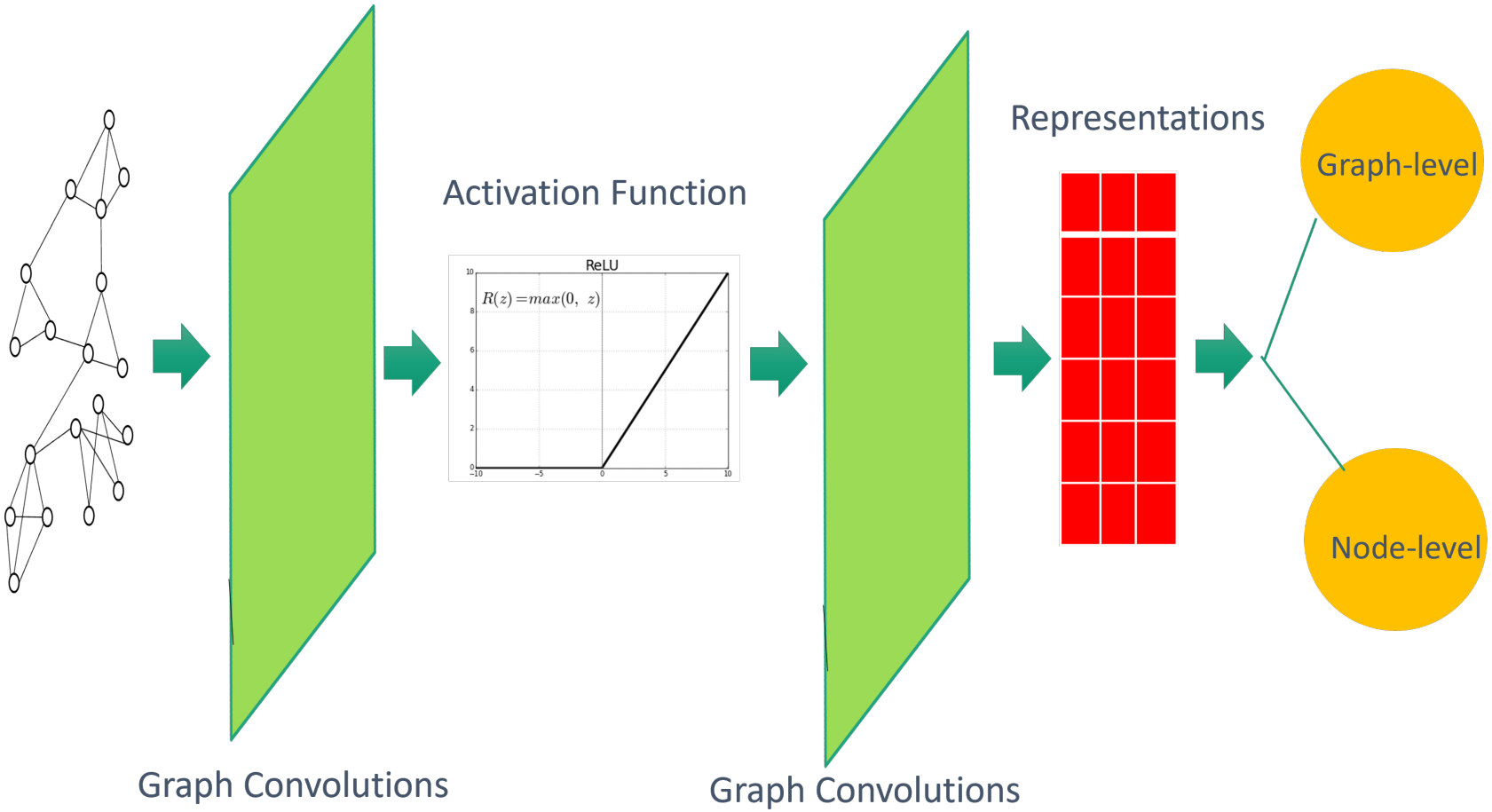
Networks as computation graphs

Key idea: Network is a computation graph



Learn how to propagate information across the network

Graph Neural Networks



Machine(Deep) Learning with Graphs

Classical ML tasks in graphs:

- Node classification: Predict a type of a given node
- Link prediction: Predict whether two nodes are linked
- Community detection: Identify densely linked clusters of nodes
- Graph similarity: How similar are two (sub)graphs

Recent ML tasks in graphs:

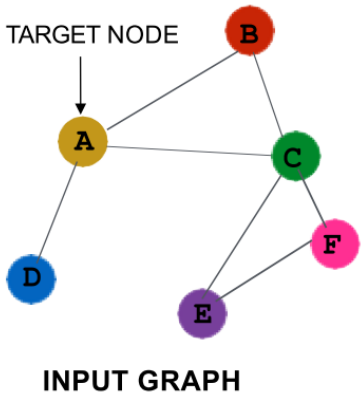
- Graph classification: Predict a type of a given graph
- Graph generation: Generate graphs from learned distribution
- Graph structure learning : Identify densely linked clusters of nodes
- Graph-to-XXX learning: Graph Inputs – XXX outputs

Graph Representation Learning (GNNs)

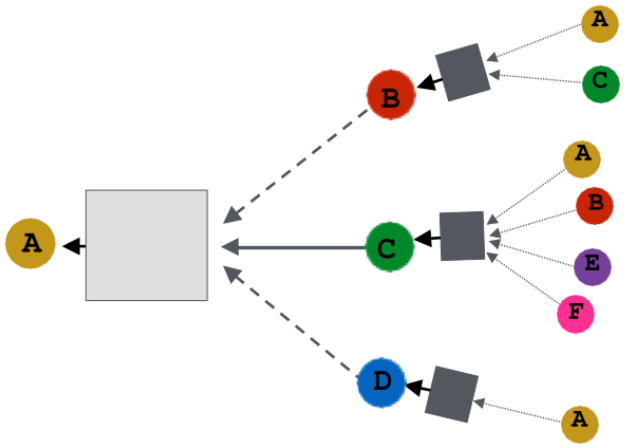
- Graph Neural Networks (GNNs) extends the well known CNN and RNN on graphs, from Euclidean data to Graphs and Manifolds
- RNN-based GNNs:
  - Graph neural networks (Scarselli et al., 2009)
  - Gated graph sequence neural networks (GGS-NNs) (Li et al., ICLR 2016)
- CNN-based GNNs:
  - Graph Convolutional Networks (GCN) (Kipf & Welling, ICLR 2017)
- Message Passing-based GNNs:
  - GraphSAGE (Hamilton & Ying & Leskovec, NIPS 2017)
  - Graph Attention Networks (GAT) (Velickovic et al., ICLR 2018)
  - MPNN (Gilmer et al., ICML 2017)

Graph Neural Networks

- Key idea: Generate node embeddings based on local neighborhoods.



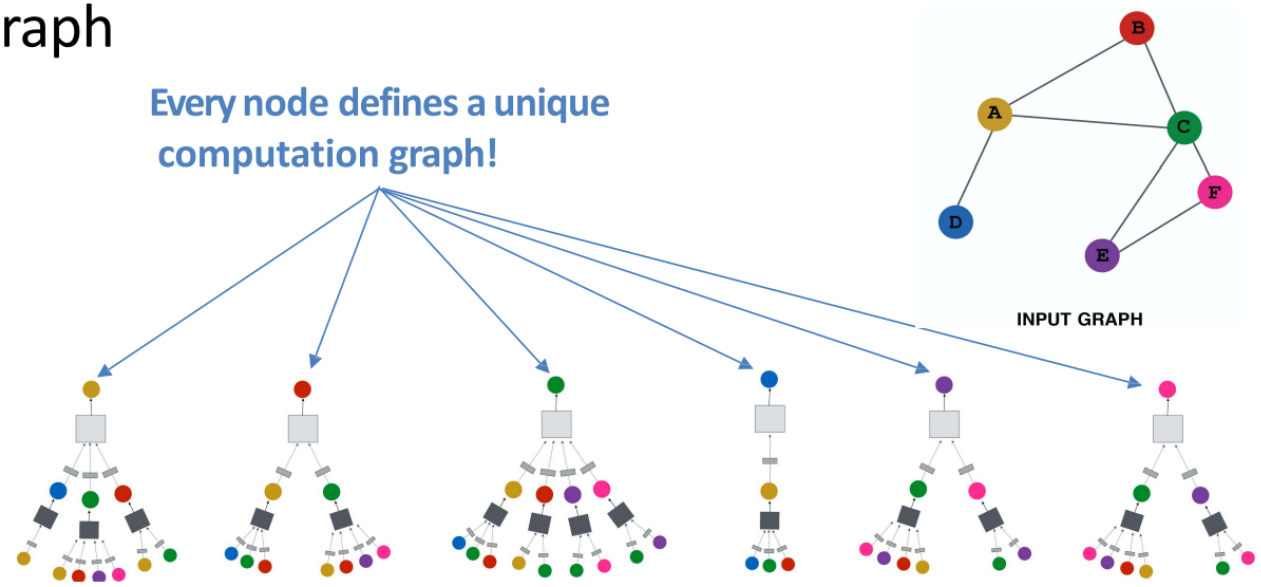
INPUT GRAPH



12

Neighborhood Aggregation

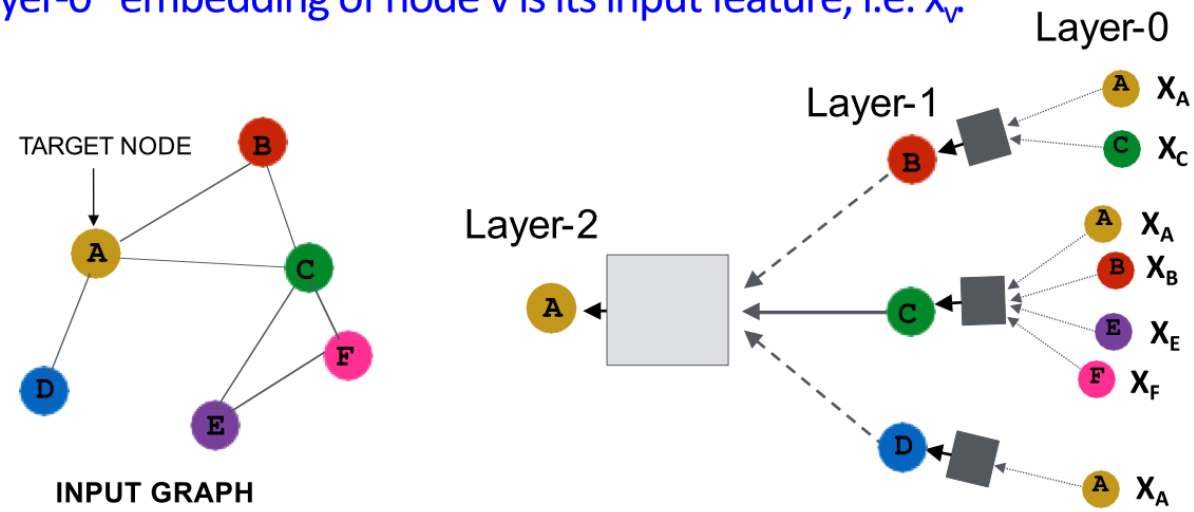
- Intuition: Network neighborhood defines a computation graph



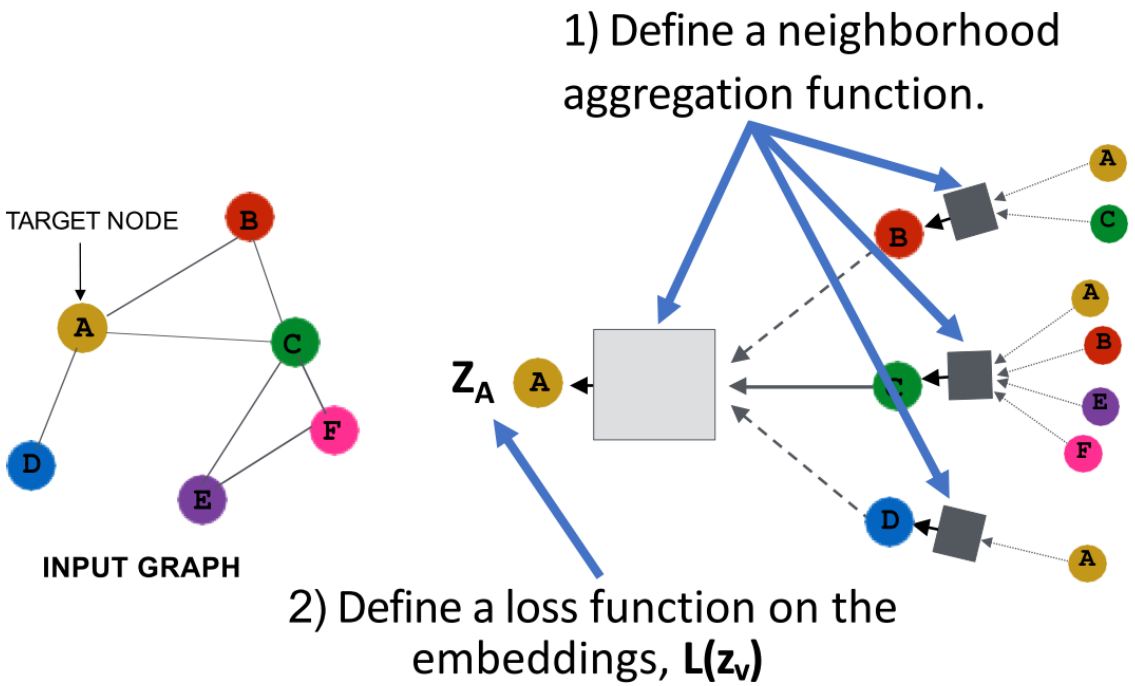
13

Neighborhood Aggregation

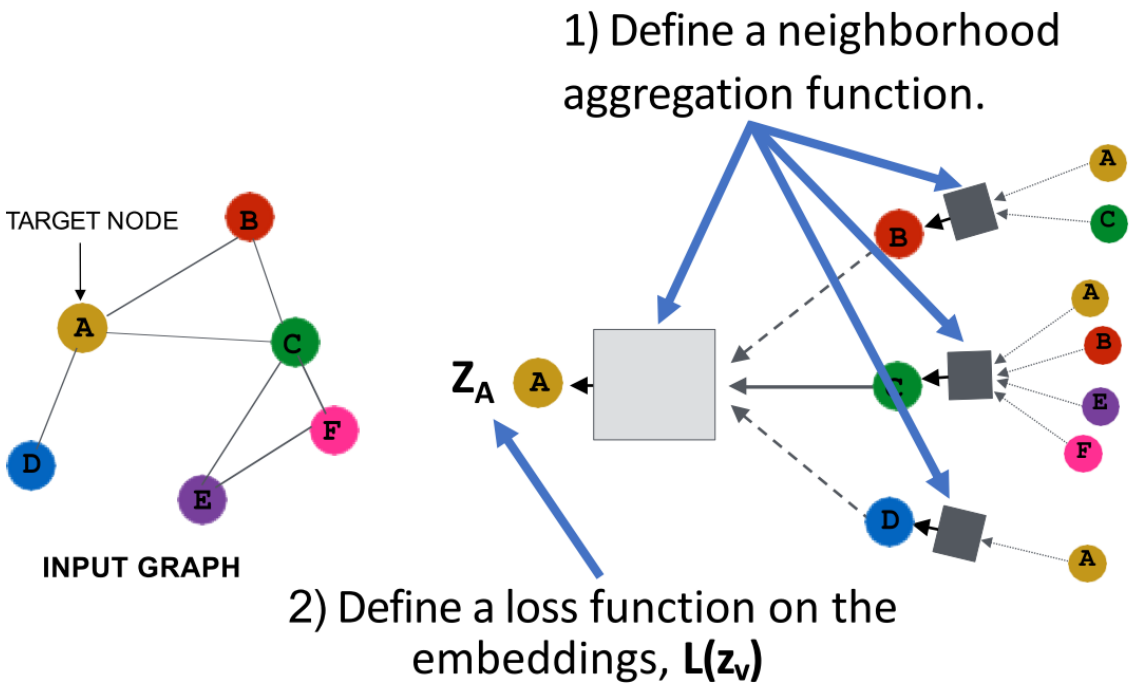
- Nodes have embeddings at each layer.
- Model can be arbitrary depth.
- "layer-0" embedding of node v is its input feature, i.e. x_v.



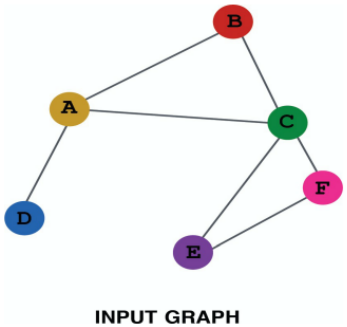
Overview of GNN Model



Overview of GNN Model

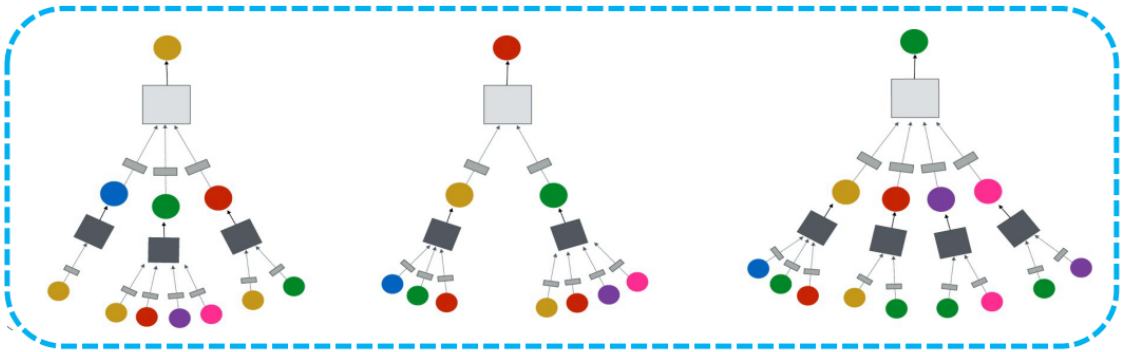


Overview of GNN Model

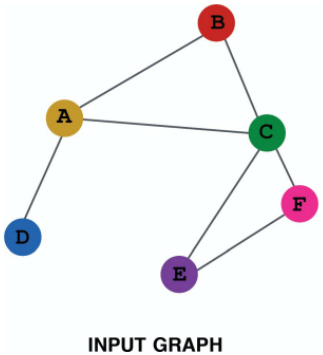


INPUT GRAPH

3) Train on a set of nodes, i.e., a batch of compute graphs

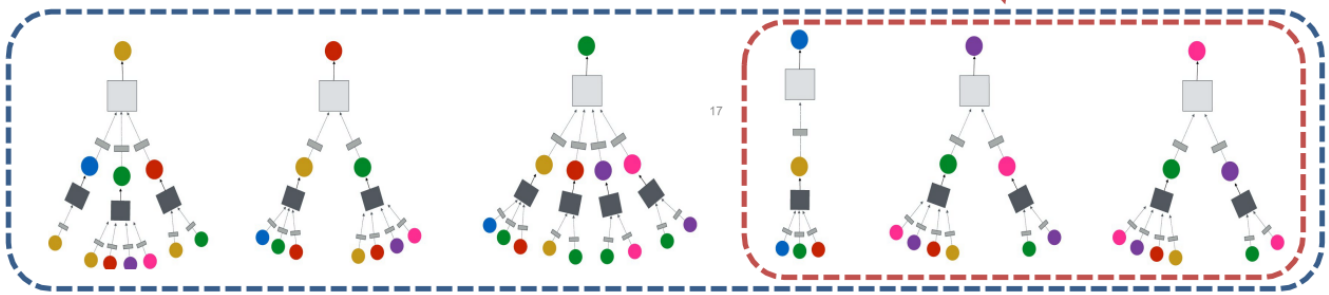


Overview of GNN Model



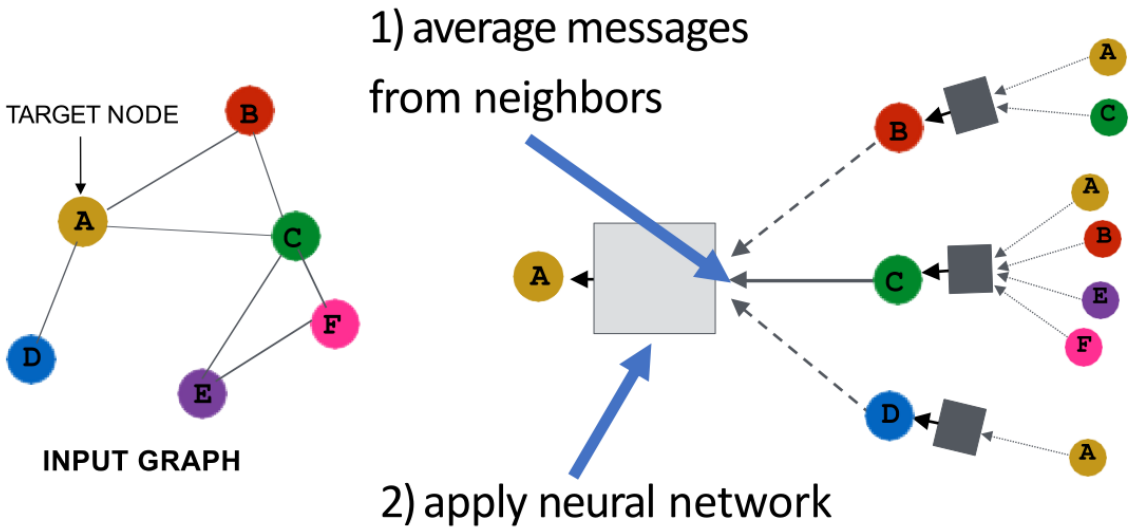
4) Generate embeddings for nodes as needed

Even for nodes we never trained on!



GNN Model: A Case Study

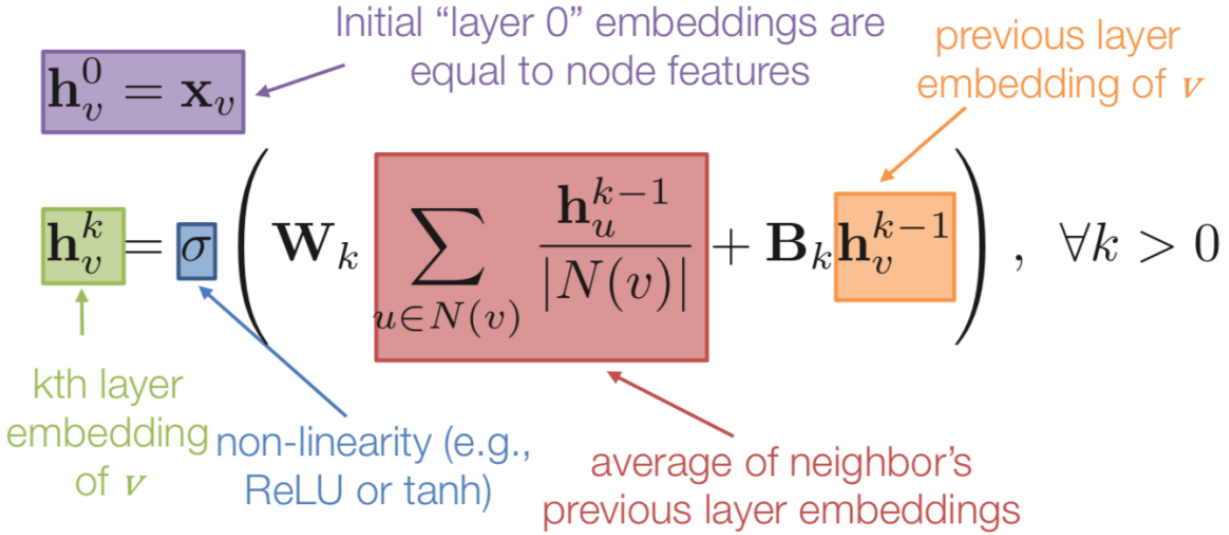
- Basic approach: Average neighbor information and apply a neural network.



18

GNN Model: A Case Study

- Basic approach: Average neighbor information and apply a neural network.



Graph Neural Networks: Popular Models

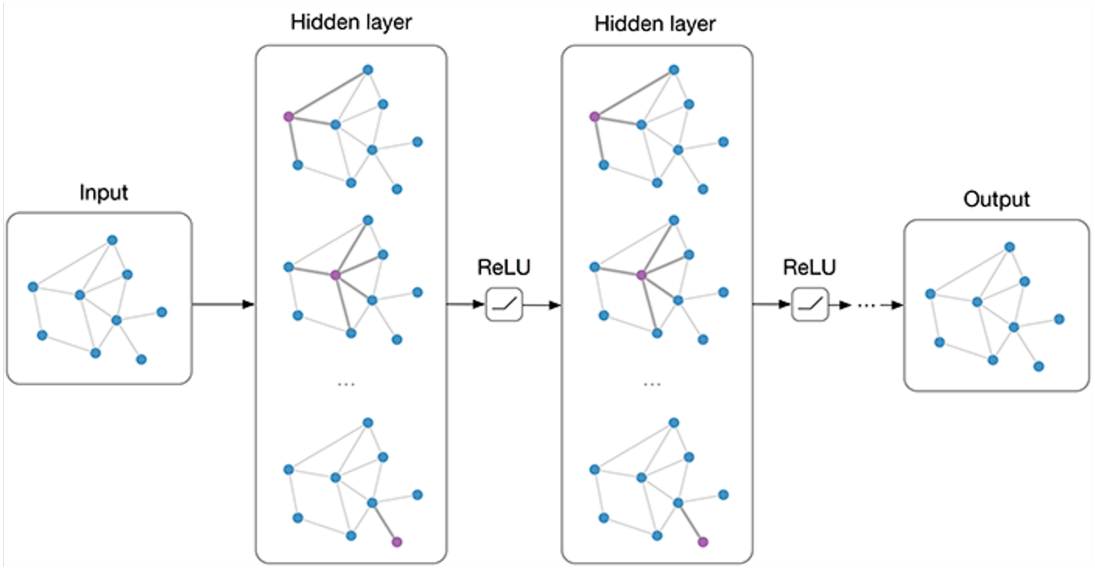
- Spectral-based Graph Filters
  - GCN (Kipf & Welling, ICLR 2017), Chebyshev-GNN (Defferrard et al. NIPS 2016)
- Spatial-based Graph Filters
  - MPNN (Gilmer et al. ICML 2017), GraphSage (Hamilton et al. NIPS 2017)
  - GIN (Xu et al. ICLR 2019)
- Attention-based Graph Filters
  - GAT (Velickovic et al. ICLR 2018)
- Recurrent-based Graph Filters
  - GGNN (Li et al. ICLR 2016)

Example 1: Graph convolution network (GCN)

$$M_{vw}^{(l)} = \frac{h_w^{(l-1)}}{d_v+1}$$

$$m_v^{(l)} = \sum_{w \in N(v) \cup \{v\}} M_{vw}^{(l)}$$

$$h_v^{(l)} = \phi(m_v^{(l)} W^{(l)})$$



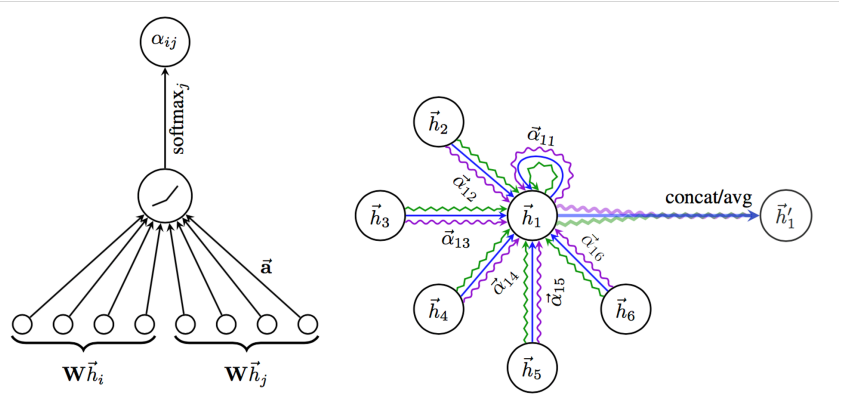
Example 2: Graph attention networks (GAT)

GAT provides weighted sum over the neighborhood—Enables to selectively integrate information.

$$M_{vw}^{(l)} = \alpha_{vw} h_w^{(l-1)}$$

$$m_v^{(l)} = \sum_{w \in N(v) \cup \{v\}} M_{vw}^{(l)}$$

$$h_v^{(l)} = \phi(m_v^{(l)} W^{(l)})$$

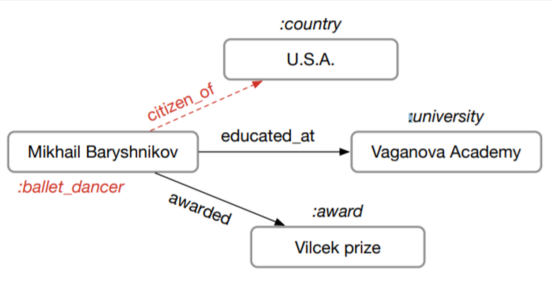


$$\alpha_{vw} = \frac{\exp(\text{LeakyReLU}(\vec{a}^T [W\vec{h}_v || W\vec{h}_w]))}{\sum_{k \in N_v} \exp(\text{LeakyReLU}(\vec{a}^T [W\vec{h}_v || W\vec{h}_k]))}$$

45

Example 3: Relational graph convolution networks (RGCN)

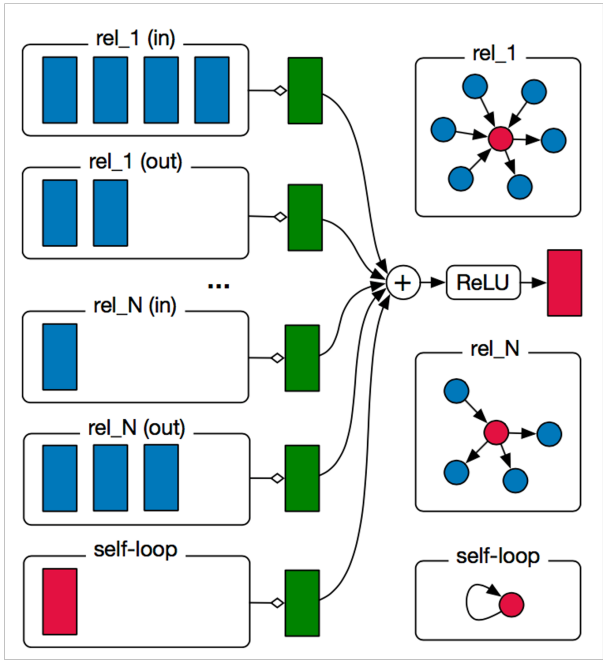
Handles graphs whose nodes are connected with different relations.



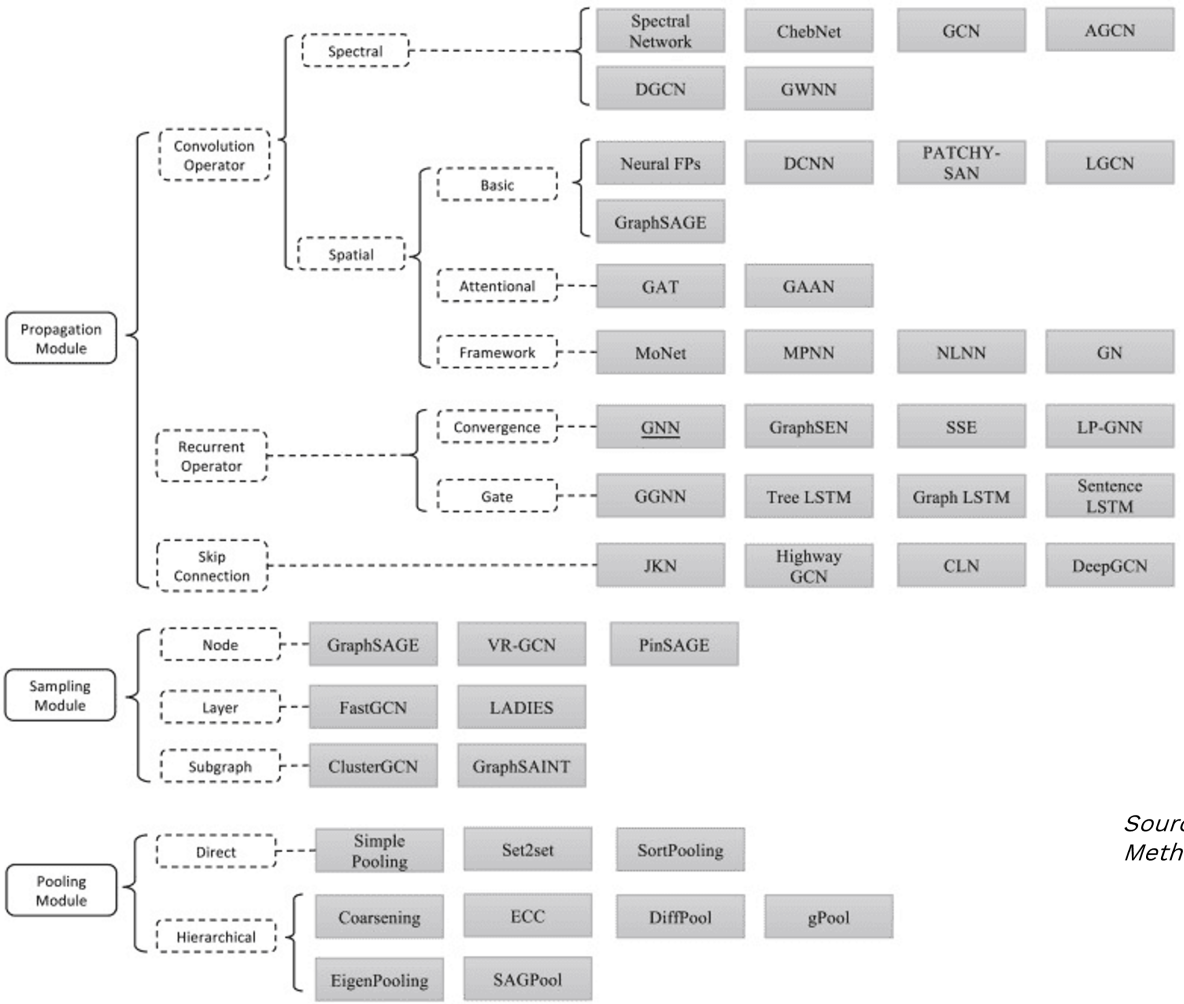
$M_{vw}^{(l)} = \frac{1}{c_{v,r}} W_r^{(l)} h_w^{(l-1)}$, r is the relation of e_{vw}.

$$m_v^{(l)} = \sum_{w \in N(v) \cup \{v\}} M_{vw}^{(l)}$$

$$h_v^{(l)} = \sigma(m_v^{(l)} W^{(l)})$$



46

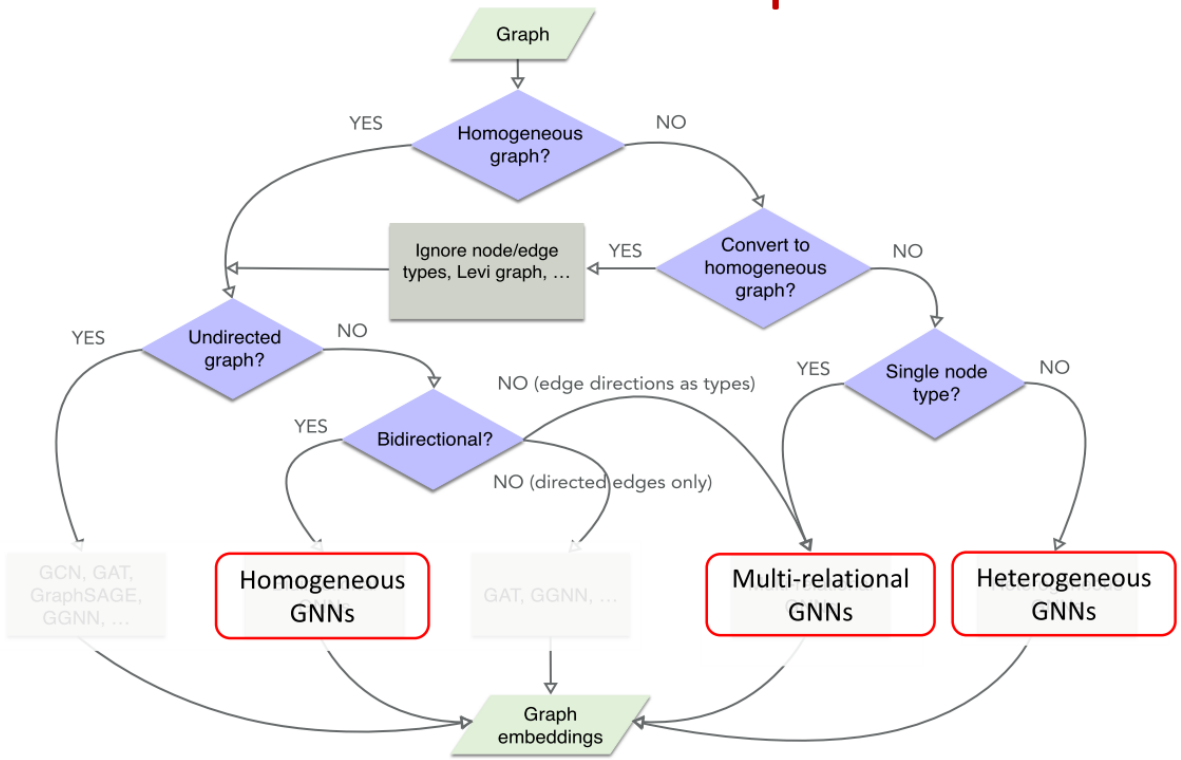


Source: Graph Neural Networks: A Review of Methods and Applications

GNN Model: Quick Summary

- Key idea: generate node embeddings by aggregating neighborhood information.
  - Allows for parameter sharing in the encoder
  - Allows for inductive learning
- Other state of the art GNNs variants:
  - Graph convolutional networks
  - Graph attention networks
  - Graph isomorphism networks

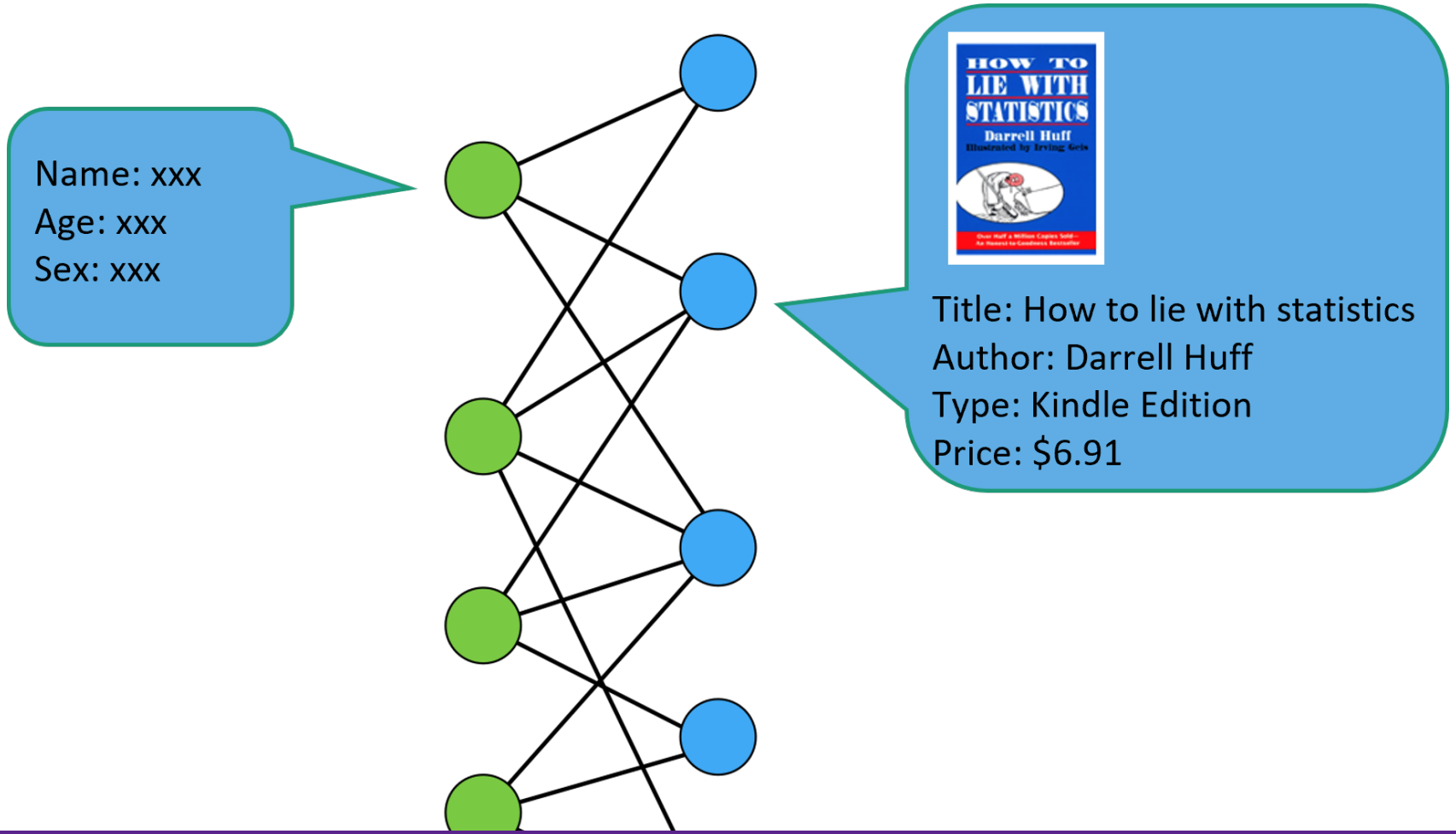
Which GNNs to Use Given a Graph?



65

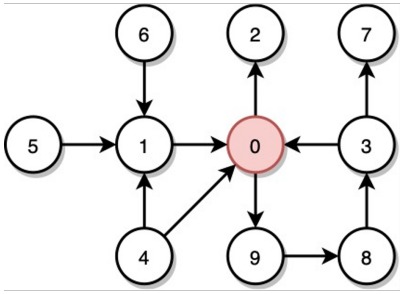
Why are graph neural networks better?

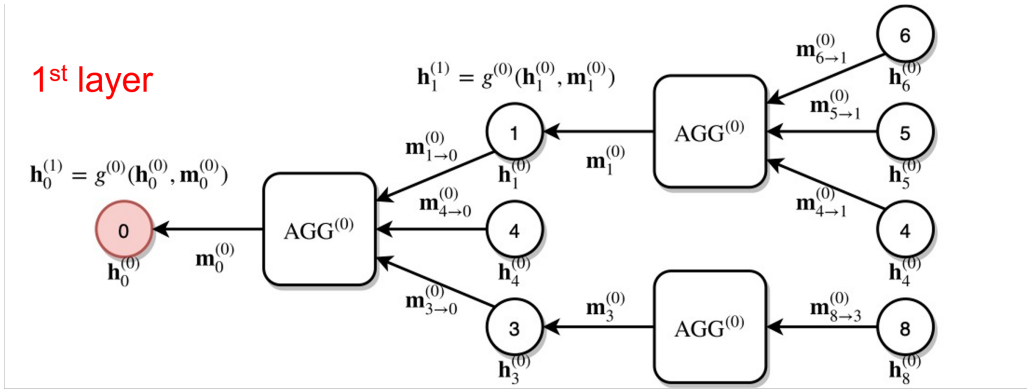
GNNs compute node embeddings using both the structure of the graph and the features of the nodes and edges.

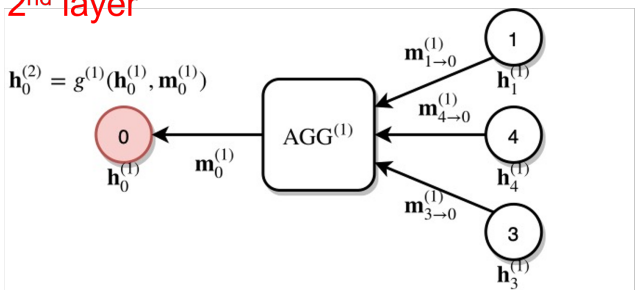


Why are graph neural networks better?

GNNs can integrate topologically distant information in a non-linear fashion.

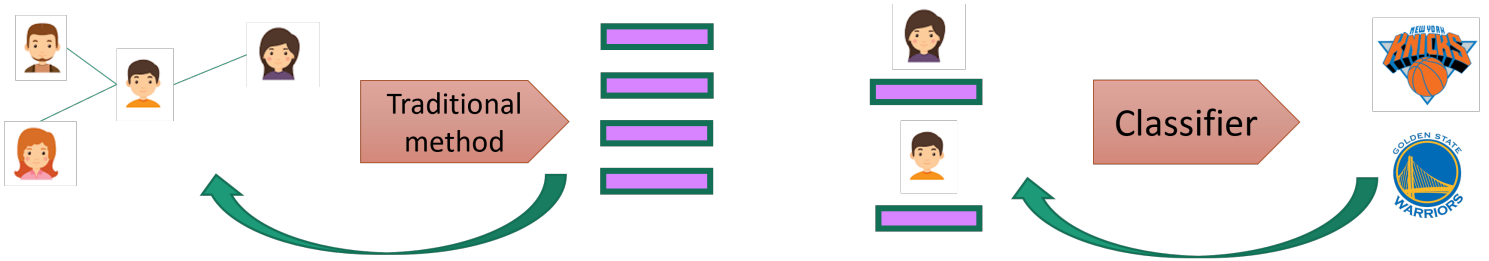


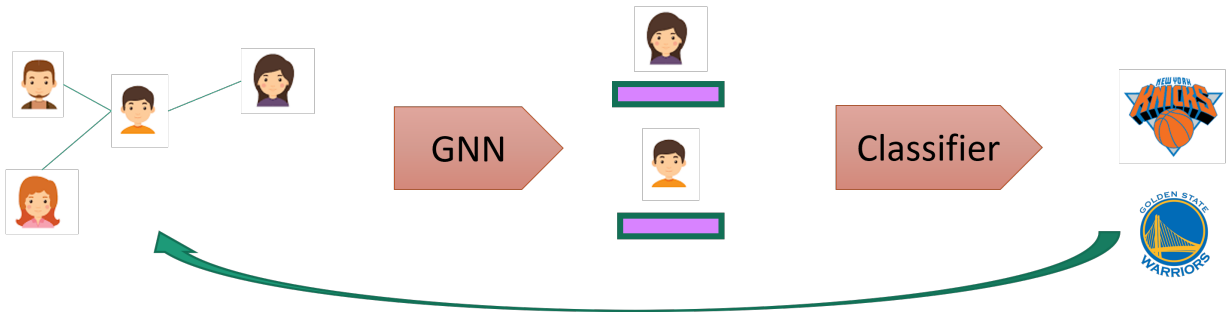




Why are graph neural networks better?

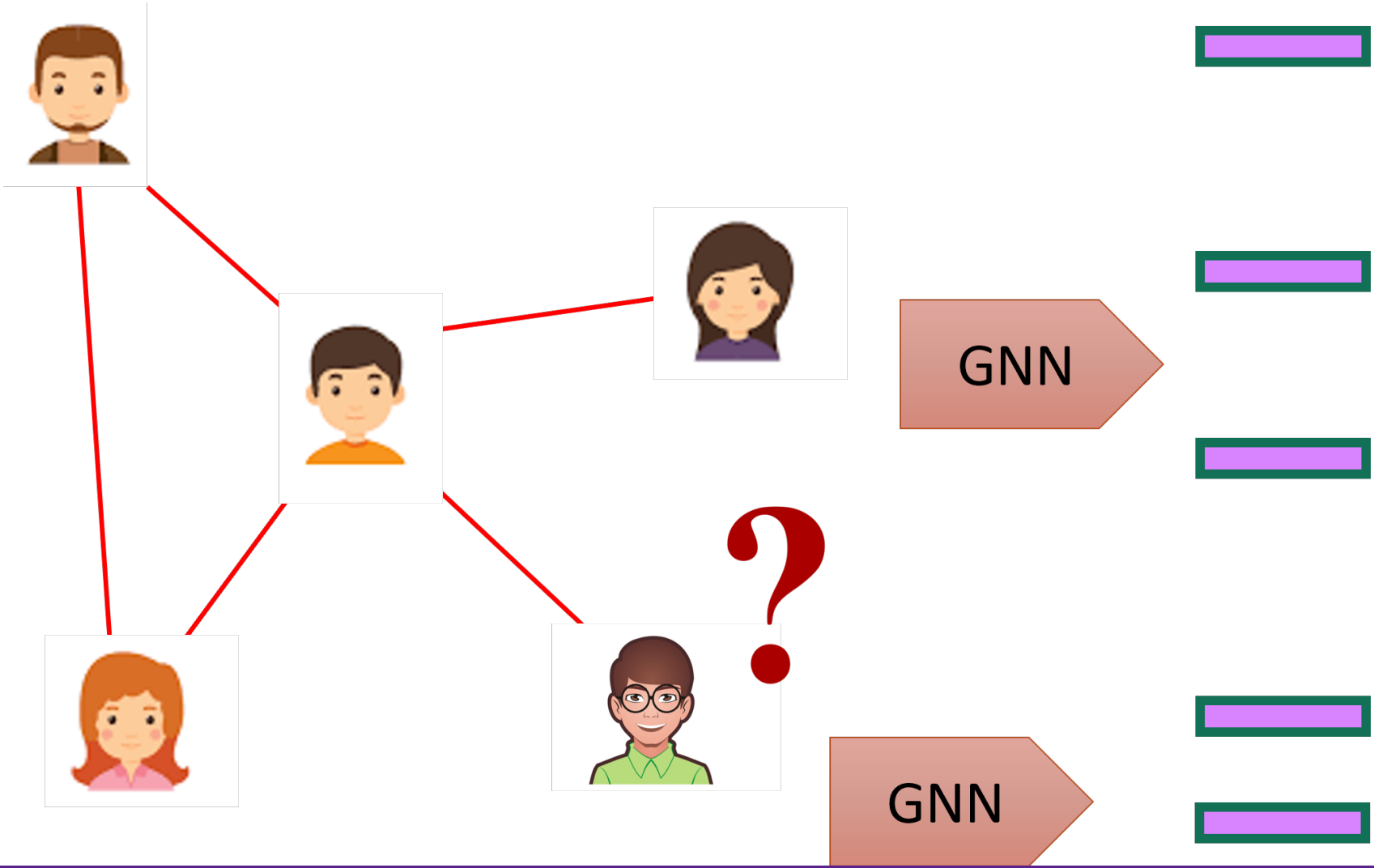
GNNs and the downstream classification/regression models can be trained in an end-to-end fashion.





Why are graph neural networks better?

GNNs are naturally inductive because they learn the same neural networks on all the nodes and edges.



Summary

- The solution to many applications can be formulated as graph learning problems.
- Graph neural networks are a new technique for graph learning. They have multiple advantages over traditional methods.
- GNNs are used in multiple graph tasks and can be trained end-to-end.

References

Slides primarily borrowed from . . .

- Overview of Graph Neural Networks - George Karypis
- Graph Neural Networks: Models and Applications - Yao Ma, et al
- GraphSAGE: Deep Learning for Relational Data - Jure Leskovec
- Deep Learning on Graphs in Natural Language Processing and Computer Vision - Lingfei Wu, IBM Research AI
- Deep Learning on Graphs for Natural Language Processing - Lingfei Wu, Yu Chen, Heng Ji, and Yunyao Li, NAACL-2021 Tutorial

Thanks ... yogeshkulkarni@yahoo.com