Stanford CS224W: Course Logistics

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

CS224W Course Logistics

- **The class meets Tue and Thu 1:30-3:00pm Pacific Time *in person***
  - Videos of the lectures will be recorded and posted on Canvas
- **Structure of lectures:**
  - 60-70 minutes of a lecture
    - During this time you can ask questions
  - 10-20 minutes of a live Q&A/discussion session at the end of the lecture

Logistics: Teaching Staff

Instructor

Jure Leskovec

Course Assistants

Serina Chang
Head CA

Federico Reyes Gómez

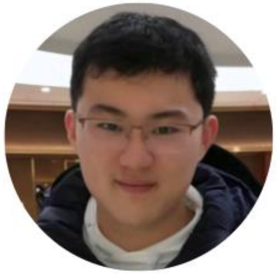
Weihua Hu

Yige Liu

Mehmet Giray Öğüt

Xiyuan (Tracey) Chen

Alexandra Porter



Hongyu Ren

Xuan Su

Course Outline

Date	Topic	Date	Topic
Tue, Sep 21	1. Introduction; Machine Learning for Graphs	**Tue, Oct 26**	11. Reasoning over Knowledge Graphs
Thu, Sep 23	2. Traditional Methods for ML on Graphs	**Thu, Oct 28**	12. Frequent Subgraph Mining with GNNs
Tue, Sep 28	3. Node Embeddings	**Thu, Nov 4**	13. Community Structure in Networks
Thu, Sep 30	4. Link Analysis: PageRank	**Tue, Nov 9**	14. Traditional Generative Models for Graphs
Tue, Oct 5	5. Label Propagation for Node Classification	**Thu, Nov 11**	15. Deep Generative Models for Graphs
Thu, Oct 7	6. Graph Neural Networks 1: GNN Model	**Tue, Nov 16**	16. Advanced Topics on GNNs
Tue, Oct 12	7. Graph Neural Networks 2: Design Space	**Thu, Nov 18**	17. Scaling Up GNNs
Thu, Oct 14	8. Applications of Graph Neural Networks	**Fri, Nov 19**	**EXAM**
Tue, Oct 19	9. Theory of Graph Neural Networks	**Tue, Nov 30**	18. Guest lecture: TBD
Thu, Oct 21	10. Knowledge Graph Embeddings	**Thu, Dec 2**	19. GNNs for Science

Logistics: Website

- **http://cs224w.stanford.edu**
  - Slides posted before the class
- **Readings:**
  - Graph Representation Learning Book by Will Hamilton
  - Research papers
- **Optional readings:**
  - Papers and pointers to additional literature
  - **This will be very useful for course projects**

Logistics: Communication

- **Ed Discussion:**
  - Access via link on Canvas
  - **Please participate and help each other!**
    - Don’t post code, annotate your questions, search for answers before you ask
  - We will post course announcements to Ed (make sure you check it regularly)
- **Please don’t communicate with prof/TAs via personal emails, but always use:**
  - cs224w-aut2122-staff@lists.stanford.edu

Logistics: Office Hours

- **OHs will be virtual**
  - We will have OHs every day, starting from 2nd week of the course
  - See http://web.stanford.edu/class/cs224w/oh.html for Zoom links and link to QueueStatus

Mon	Tue	Wed	Thu	Fri	Sat	Sun
Yige 10:00am-12:00pm	Alex 3:30pm-5:30pm	Giray 1:00pm-3:00pm Tracey 7:00pm-9:00pm	Weihua 10:00am-12:00pm	Federico 3:00pm-5:00pm	Hongyu 9:30am-11:30am Xuan 1:00pm-3:00pm	Serina 5:00pm-7:00pm

Work for Course: Grading

- **Final grade will be composed of:**
  - **Homework: 25%**
    - 3 written homeworks, each worth 8.3%
  - **Coding assignments: 20%**
    - 5 coding assignments using Google Colab, each worth 4%
  - **Exam: 35%**
  - **Course project: 20%**
    - Proposal: 20%; Final report: 70%; Poster: 10%
  - **Extra credit: Ed participation, PyG/GraphGym code contribution**
    - Used if you are on the boundary between grades

Work for Course: Submitting

- **How to submit?**
  - **Upload via Gradescope**
    - You will be automatically registered to Gradescope once you officially enroll in CS224W
  - Homeworks, Colabs (numerical answers), and project deliverables are submitted on Gradescope
- **Total of 2 Late Periods (LP) per student**
  - Max 1 LP per assignment (no LP for the final report)
    - LP gives **4 extra days**: assignments usually due on Thursday (11:59pm) → with LP, it is due the following Monday (11:59pm)

Work for Course: HWs, Colabs

- **Homeworks (25%, n=3)**
  - **Written assignments take longer and take time (~10-20h) – start early!**
    - A combination of data analysis, algorithm design, and math
- **Colabs (20%, n=5)**
  - **We have more Colabs but they are shorter (~3-5h); Colab 0 is not graded.**
    - Get hands-on experience coding and training GNNs; good preparation for final projects and industry

Work for Course: Exam

- **Single exam: Friday, Nov 19**
  - **Take-home, open-book, timed**
    - Administered via Gradescope
    - Released at 10am PT on Friday, available until 10am PT the following day
    - Once you open it, you will have 100 minutes to complete the exam
  - **Content**
    - Will have written questions (similar to Homework), will possibly have a coding section (similar to Colabs)
    - More details to come!

Work for Course: Project

- **Two options**
  - (1) Default project (predefined task)
  - (2) Custom project (open-ended)
- **Logistics**
  - **Groups of up to 3 students**
    - Groups of 1 or 2 are allowed; the team size will be taken under consideration when evaluating the scope of the project. But 3 person teams can be more efficient.
  - **Google Cloud credits**
    - We will provide $50 in Google Cloud credits to each student
    - You can also get $300 with Google Free Trial (https://cloud.google.com/free/docs/gcp-free-tier)
- **Read: http://cs224w.stanford.edu/info.html**

Course Schedule

Assignment	Due on (11:59pm PT)
Colab 0	Not graded
Colab 1	Thu, Oct 7 (week 3)
Homework 1	Thu, Oct 14 (week 4)
Project Proposal	Tue, Oct 19 (week 5)
Colab 2	Thu, Oct 21 (week 5)
Homework 2	Thu, Oct 28 (week 6)
Colab 3	Thu, Nov 4 (week 7)
Homework 3	Thu, Nov 11 (week 8)
Colab 4	Thu, Nov 18 (week 9)
EXAM	Fri, Nov 19 (week 9)
Colab 5	Thu, Dec 2 (week 11)
Project Report	Thu, Dec 9 **(No Late Periods!)**

Honor Code

Make sure you read and understand it!

- **We strictly enforce the [Stanford Honor Code]**
  - Violations of the Honor Code include:
    - Copying or allowing another to copy from one's own paper
    - Unpermitted collaboration
    - Plagiarism
    - Giving or receiving unpermitted aid on a take-home examination
    - Representing as one's own work the work of another
    - Giving or receiving aid on an assignment under circumstances in which a reasonable person should have known that such aid was not permitted
  - The standard sanction for a first offense includes a one-quarter suspension and 40 hours of community service.

Prerequisites

- **The course is self-contained.**
- **No single topic is too hard by itself.**
- **But we will cover and touch upon many topics and this is what makes the course hard.**
  - **Good background in:**
    - Machine Learning
    - Algorithms and graph theory
    - Probability and statistics
  - **Programming:**
    - You should be able to write non-trivial programs (in Python)
    - Familiarity with PyTorch is a plus

Graph Machine Learning Tools

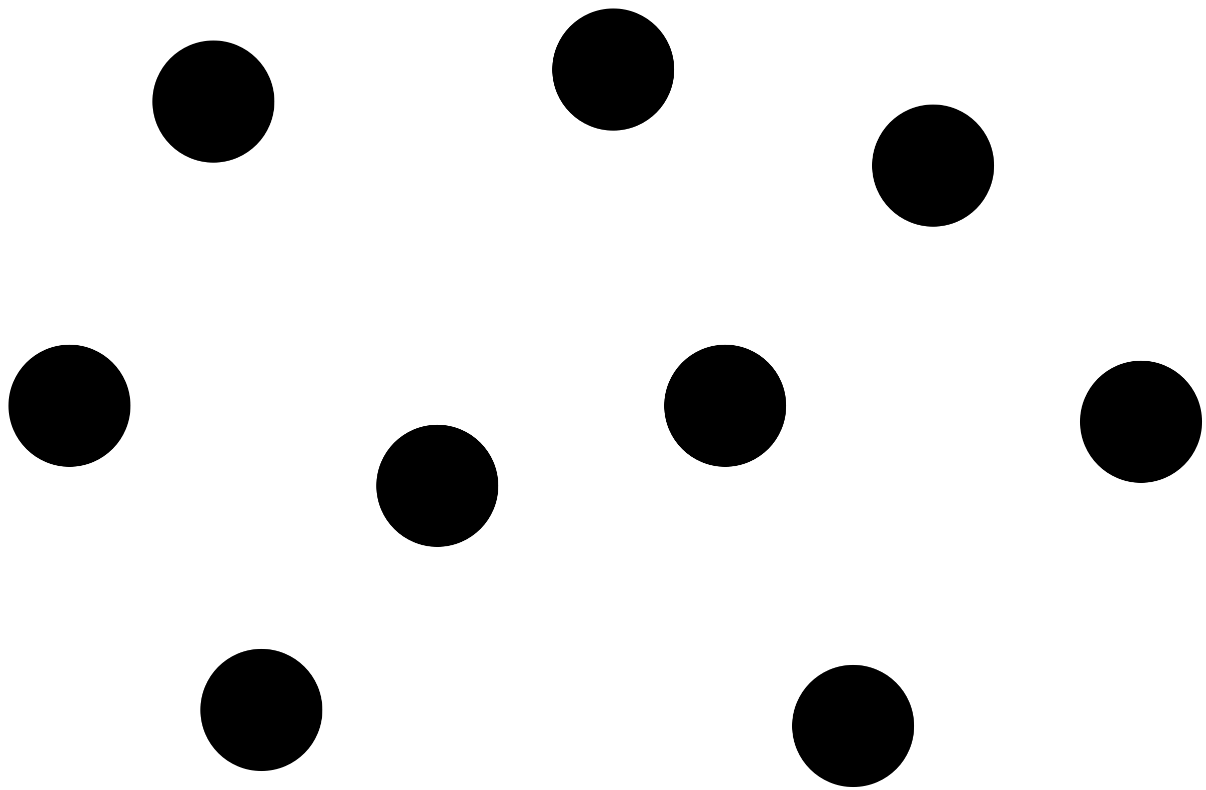
- **We use PyG (PyTorch Geometric):**
  - The ultimate library for Graph Neural Networks
- **We further recommend:**
  - GraphGym: Platform for designing Graph Neural Networks.
    - Modularized GNN implementation, simple hyperparameter tuning, flexible user customization
  - Both platforms are very helpful for the course project (save your time & provide advanced GNN functionalities)
- **Other network analytics tools**: SNAP.PY, NetworkX

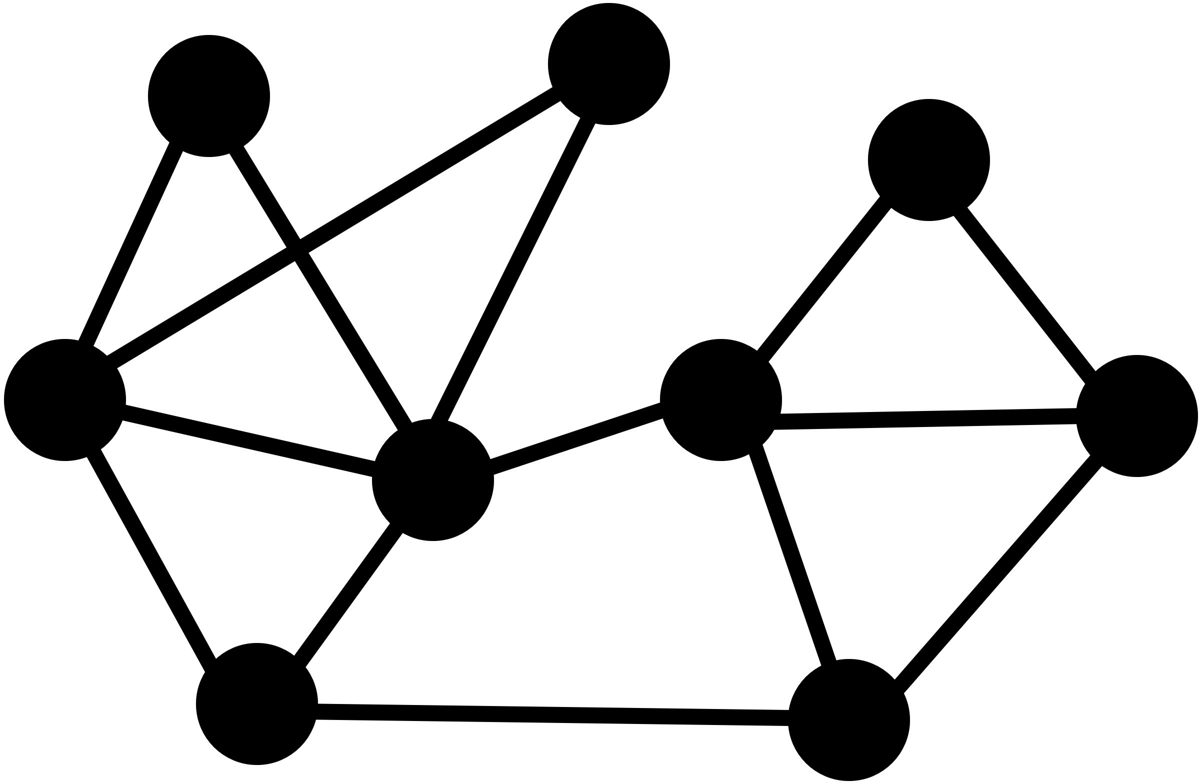
Stanford CS224W: Machine Learning with Graphs

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

Why Graphs?

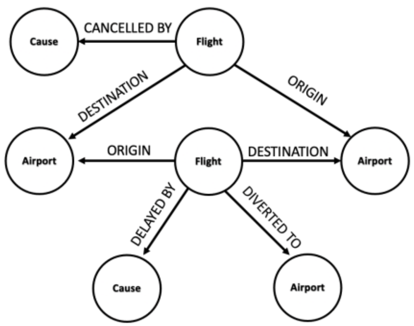
Graphs are a general language for describing and analyzing entities with relations/interactions





Graph

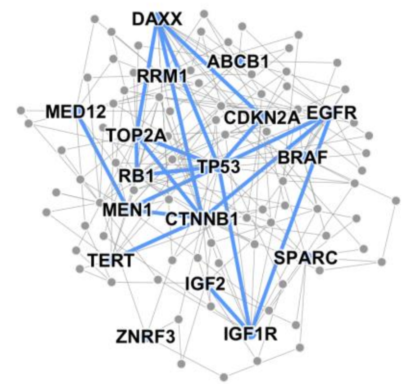
Many Types of Data are Graphs (1)



Event Graphs

Image credit: SalientNetworks

Computer Networks



Disease Pathways

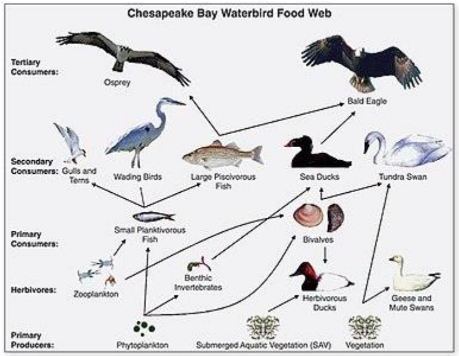


Image credit: Wikipedia

Image credit: Pinterest

Particle Networks

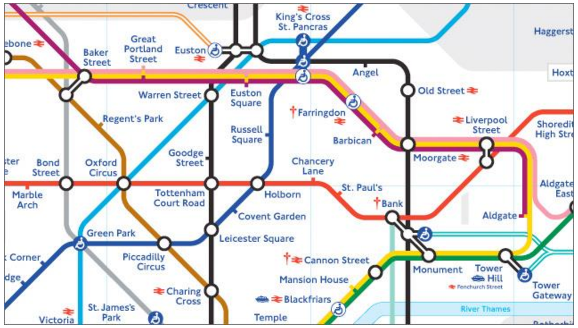


Image credit: visitlondon.com

Food Webs

Underground Networks

Many Types of Data are Graphs (2)

Image credit: Medium

Social Networks

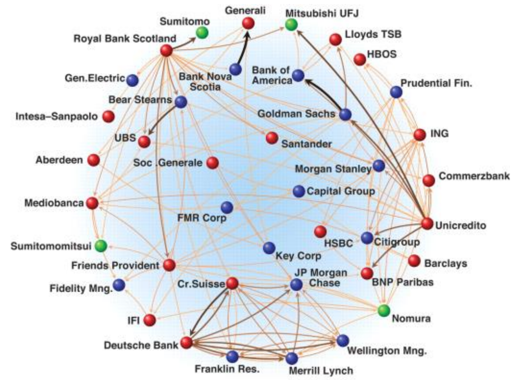


Image credit: Science

Economic Networks

Image credit: Lumen Learning

Communication Networks

Citation Networks

Image credit: Missoula Current News

Internet

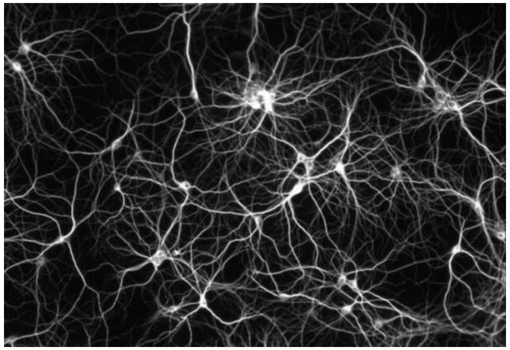


Image credit: The Conversation

Networks of Neurons

Many Types of Data are Graphs (3)

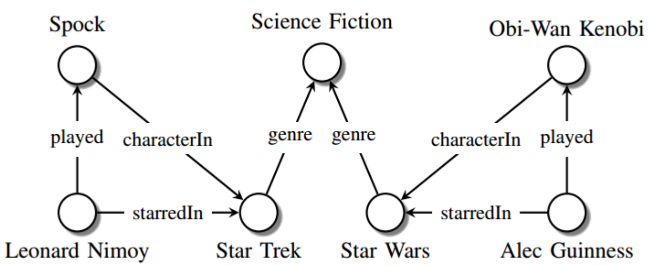


Image credit: Maximilian Nickel et al

Knowledge Graphs

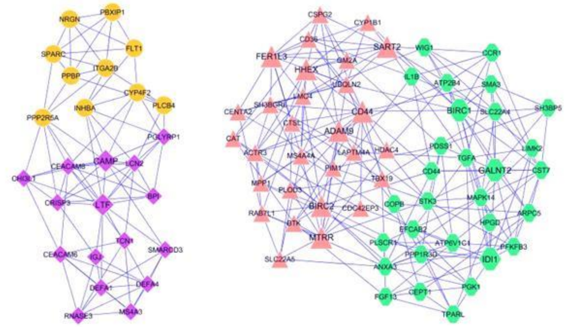


Image credit: ese.wustl.edu

Regulatory Networks

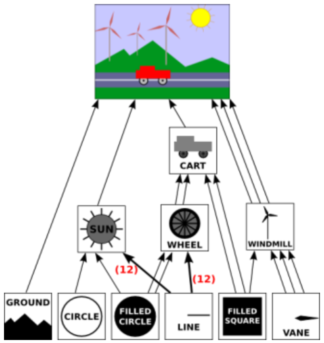


Image credit: math.hws.edu

Scene Graphs

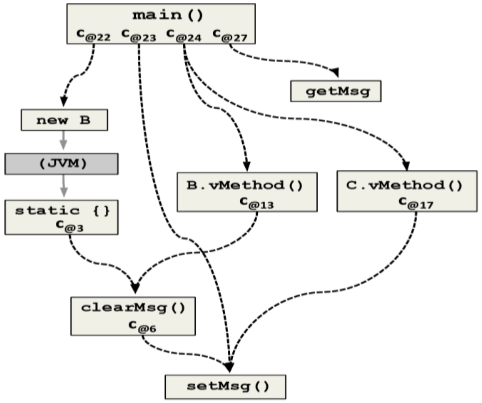


Image credit: ResearchGate

Code Graphs

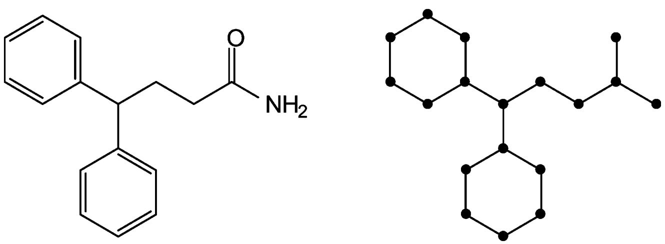


Image credit: MDPI

Molecules

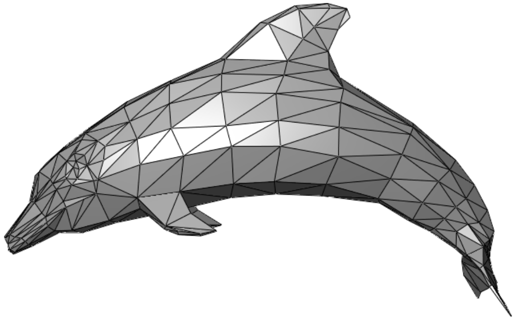


Image credit: Wikipedia

3D Shapes

Graphs and Relational Data

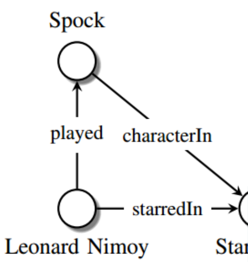
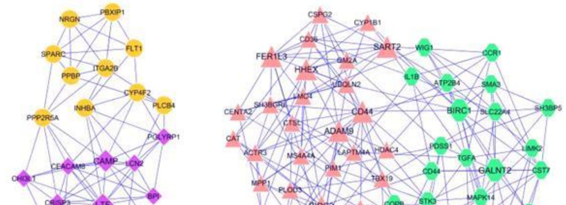
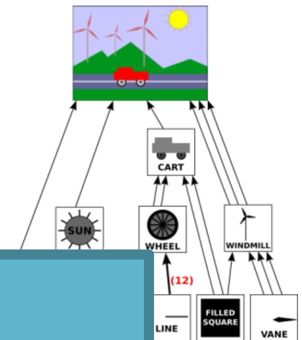


Image cred

Know





ath.hws.edu

Graphs

Main question:

How do we take advantage of relational structure for better prediction?

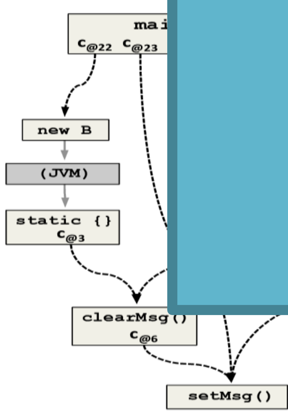


Image credit: ResearchGate

Code Graphs

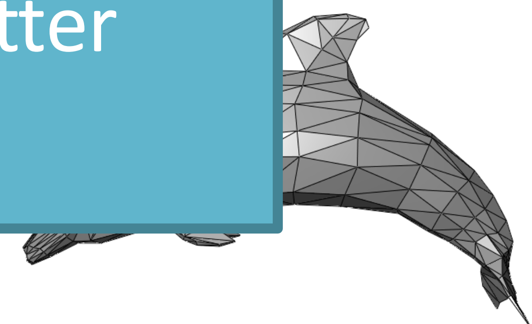


Image credit: MDPI

Molecules

Image credit: Wikipedia

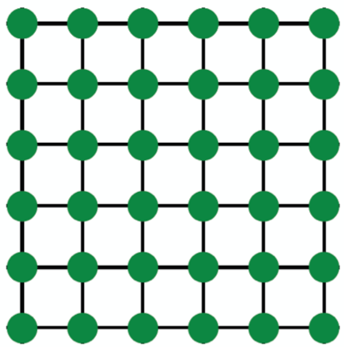
3D Shapes

Graphs: Machine Learning

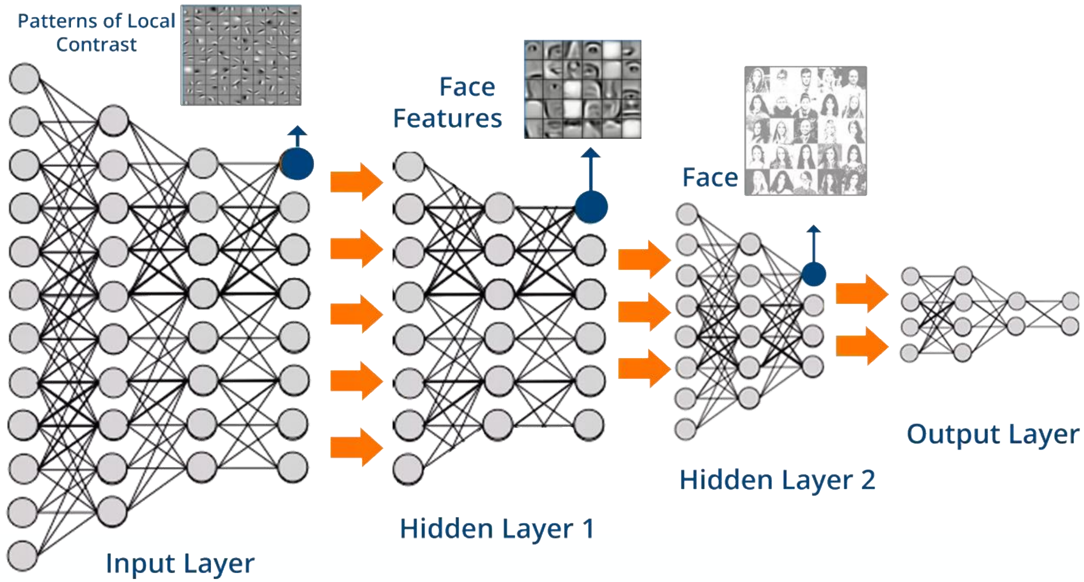
Complex domains have a rich relational structure, which can be represented as a **relational graph**

By explicitly modeling relationships we achieve better performance!

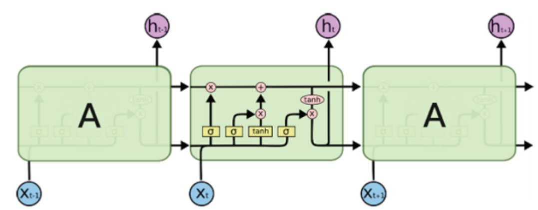
Today: Modern ML Toolbox



Images



Text/Speech



Modern deep learning toolbox is designed for simple sequences & grids

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

Modern
deep learning toolbox
is designed for
sequences & grids

Not everything can be represented as a sequence or a grid

How can we develop neural networks that are much more broadly applicable?

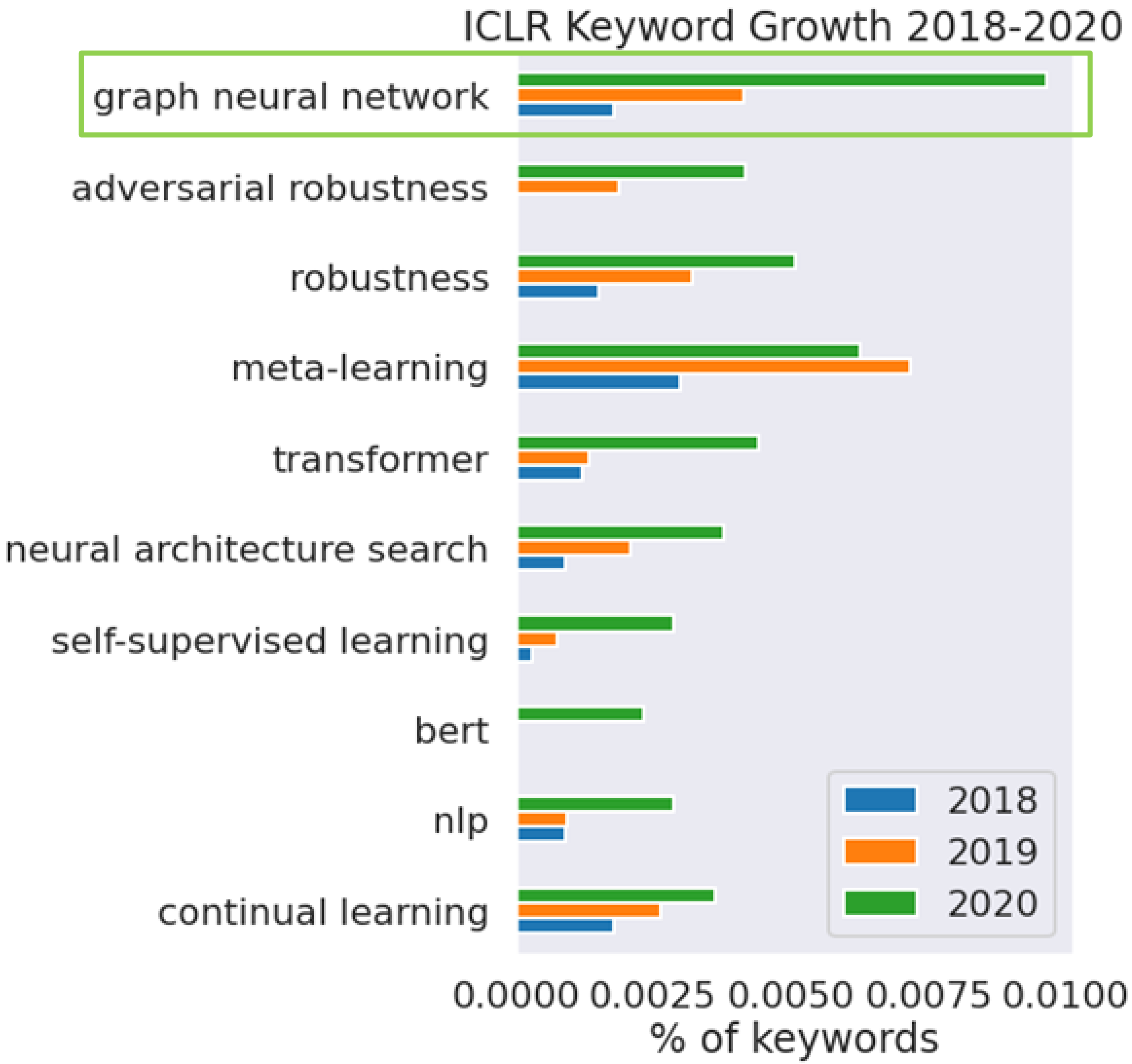
New frontiers beyond classic neural networks that only learn on images and sequences

This Class

Graphs are the new frontier of deep learning

Graphs connect things.

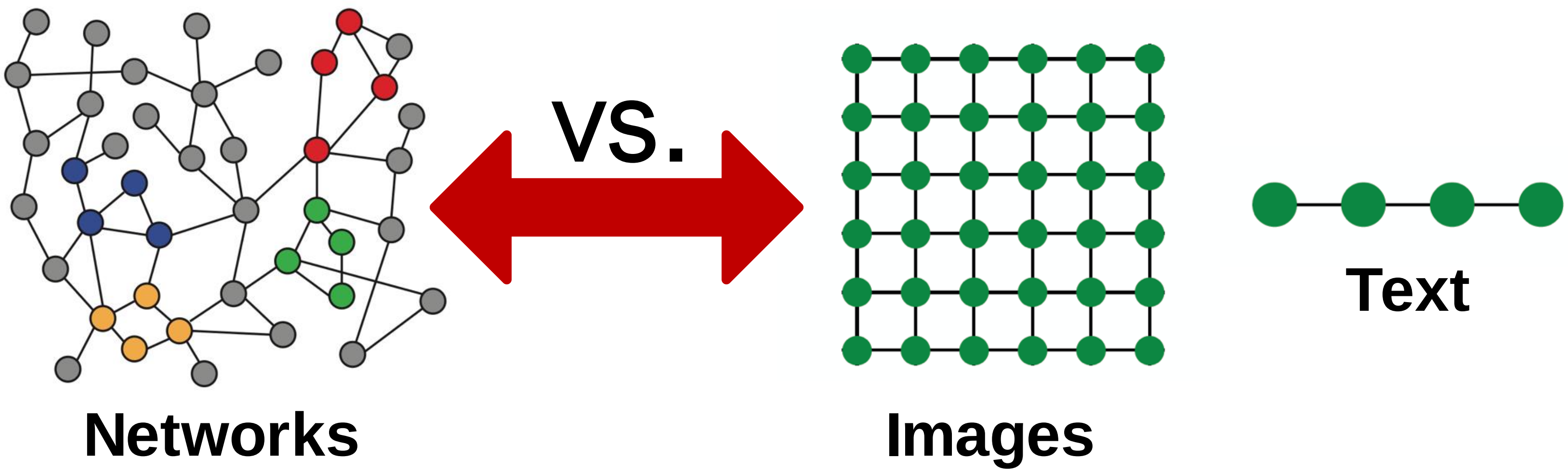
The hottest subfield in ML



Why is Graph Deep Learning Hard?

Networks are complex.

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



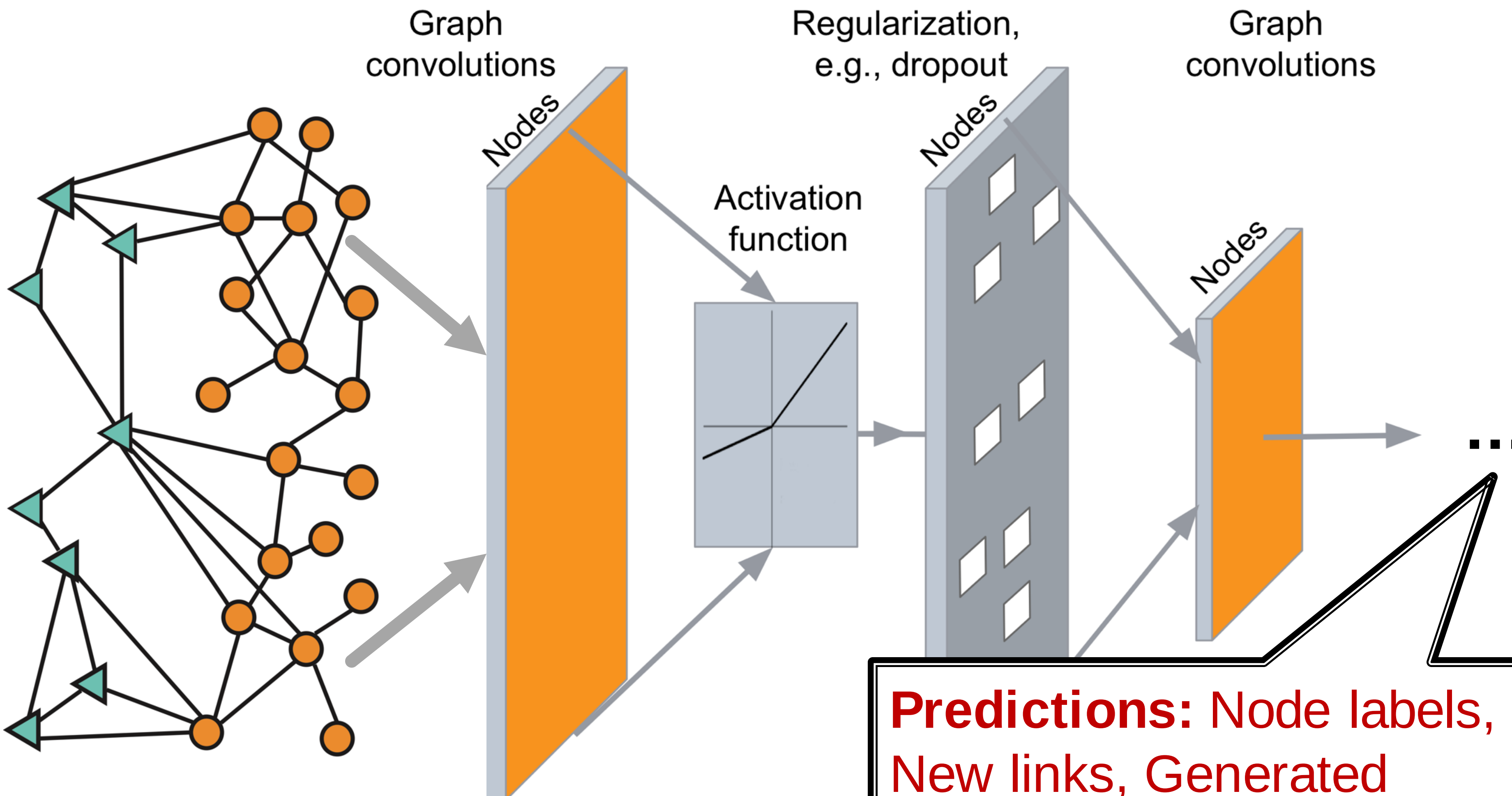
- No fixed node ordering or reference point
- Often dynamic and have multimodal features

This Course

How can we develop neural networks that are much more broadly applicable?

<u>Graphs</u> are the new frontier of deep learning

CS224W: Deep Learning in Graphs

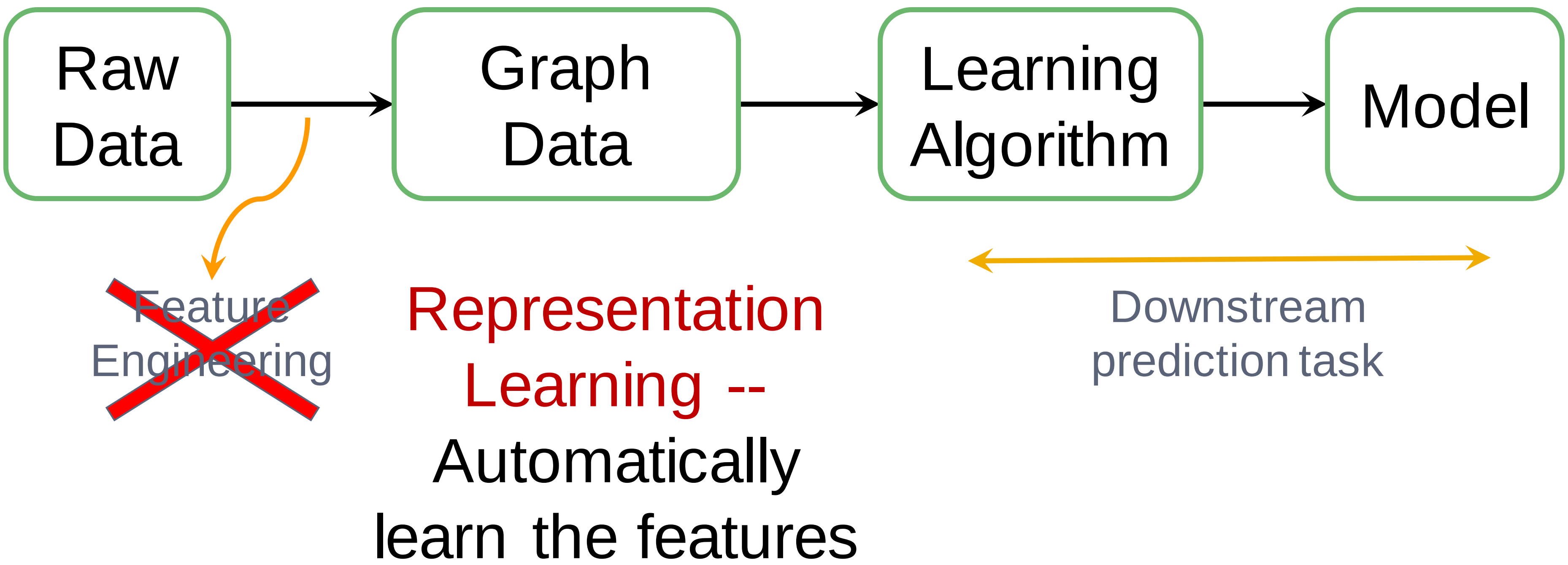


Input: Network

Predictions: Node labels, New links, Generated graphs and subgraphs

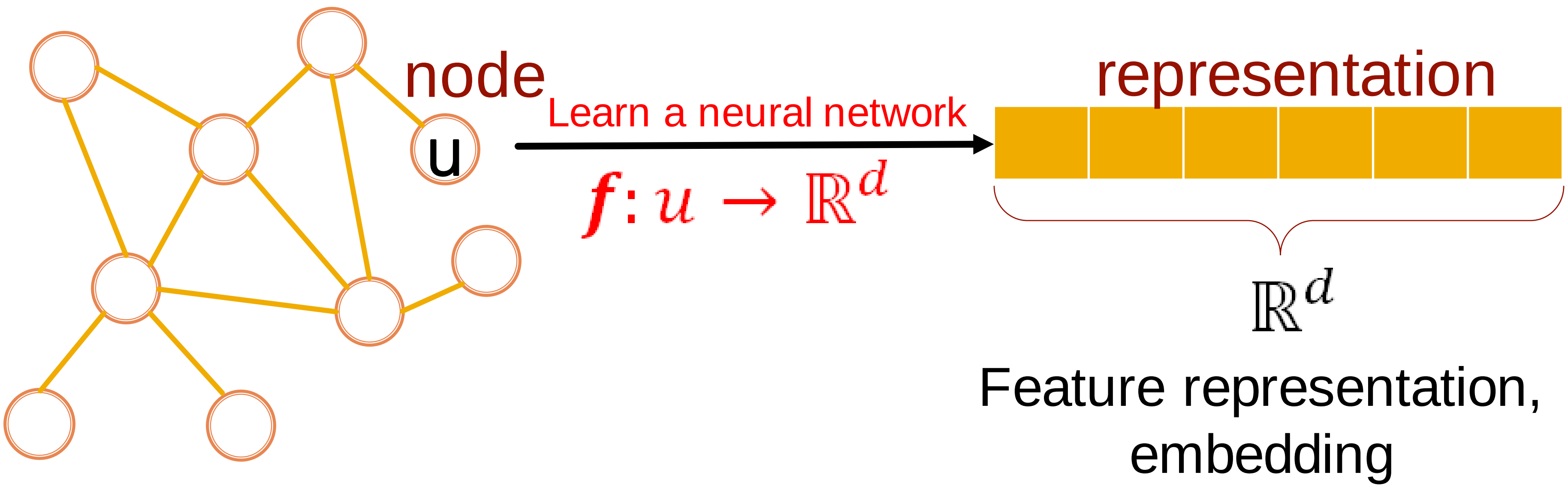
CS224W & Representation Learning

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



CS224W & Representation Learning

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



CS224W Course Outline

We are going to cover various topics in Machine Learning and Representation Learning for graph structured data:

- Traditional methods: Graphlets, Graph Kernels
- Methods for node embeddings: DeepWalk, Node2Vec
- Graph Neural Networks: GCN, GraphSAGE, GAT, Theory of GNNs
- Knowledge graphs and reasoning: TransE, BetaE
- Deep generative models for graphs: GraphRNN
- Applications to Biomedicine, Science, Industry

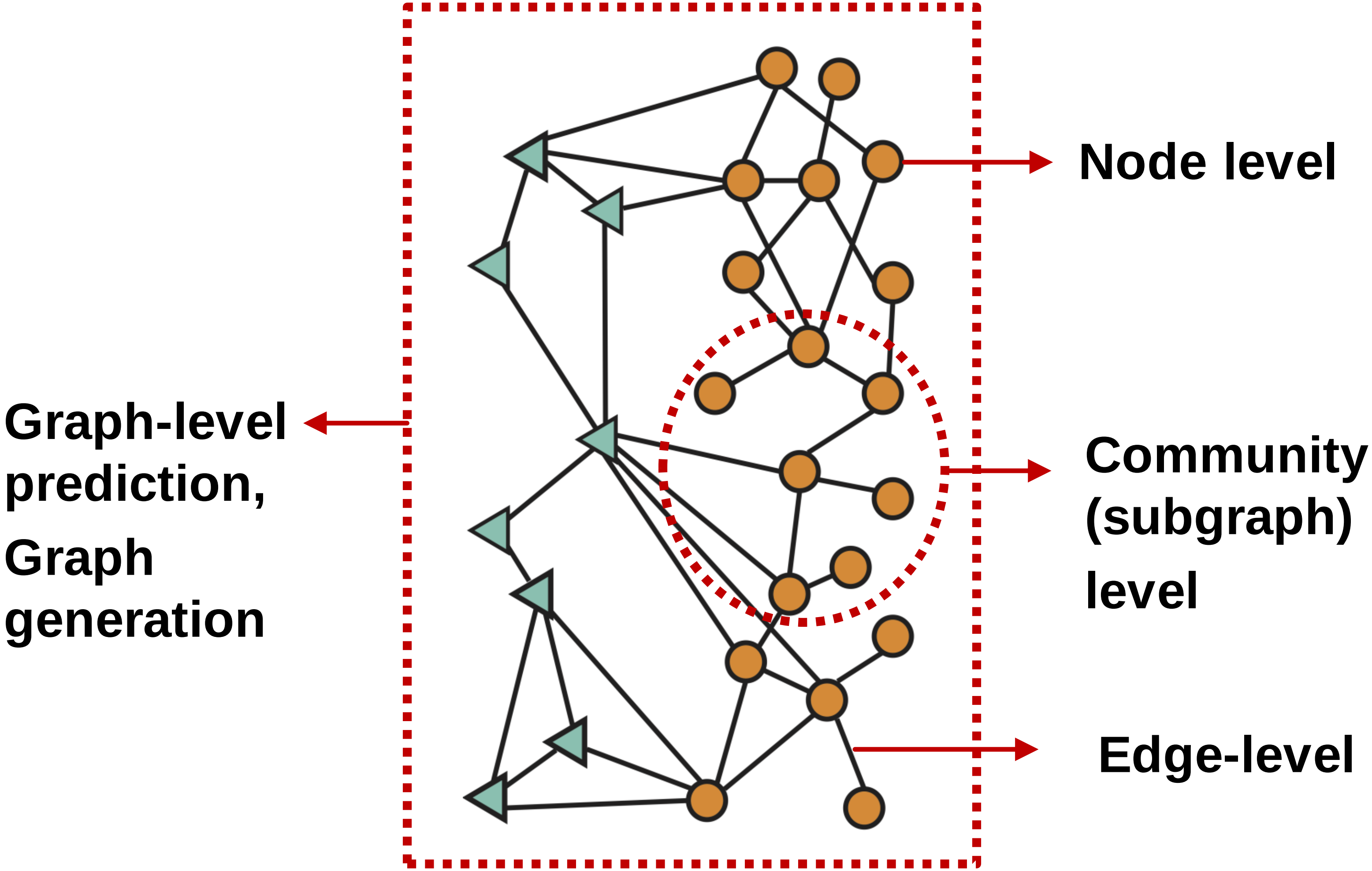
Topics Covered in CS224W

Date	Topic	Date	Topic
Tue, Sep 21	1. Introduction; Machine Learning for Graphs	**Tue, Oct 26**	11. Reasoning over Knowledge Graphs
Thu, Sep 23	2. Traditional Methods for ML on Graphs	**Thu, Oct 28**	12. Frequent Subgraph Mining with GNNs
Tue, Sep 28	3. Node Embeddings	**Thu, Nov 4**	13. Community Structure in Networks
Thu, Sep 30	4. Link Analysis: PageRank	**Tue, Nov 9**	14. Traditional Generative Models for Graphs
Tue, Oct 5	5. Label Propagation for Node Classification	**Thu, Nov 11**	15. Deep Generative Models for Graphs
Thu, Oct 7	6. Graph Neural Networks 1: GNN Model	**Tue, Nov 16**	16. Advanced Topics on GNNs
Tue, Oct 12	7. Graph Neural Networks 2: Design Space	**Thu, Nov 18**	17. Scaling Up GNNs
Thu, Oct 14	8. Applications of Graph Neural Networks	**Fri, Nov 19**	EXAM
Tue, Oct 19	9. Theory of Graph Neural Networks	**Tue, Nov 30**	18. Guest lecture: TBD
Thu, Oct 21	10. Knowledge Graph Embeddings	**Thu, Dec 2**	19. GNNs for Science

Stanford CS224W: Applications of Graph ML

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

Different Types of Tasks



Classic Graph ML Tasks

- **Node classification**: Predict a property of a node
  - **Example:** Categorize online users / items
- **Link prediction**: Predict whether there are missing links between two nodes
  - **Example:** Knowledge graph completion
- **Graph classification**: Categorize different graphs
  - **Example:** Molecule property prediction
- **Clustering**: Detect if nodes form a community
  - **Example:** Social circle detection
- **Other tasks**:
  - **Graph generation**: Drug discovery
  - **Graph evolution**: Physical simulation

Classic Graph ML Tasks

- **Node classification**: Predict a property of a node
  - **Example:** Categorize online users / items
- **Link prediction**: Predict whether there are missing links
  - **Exa**
- **Grap** phs
  - **Exa**
- **Clust** y
  - **Exa**
- **Others**:
  - **Graph generation**: Drug discovery
  - **Graph evolution**: Physical simulation

These Graph ML tasks lead to high-impact applications!

Example of Node-level ML Tasks

Example (1): Protein Folding

A protein chain acquires its native 3D structure

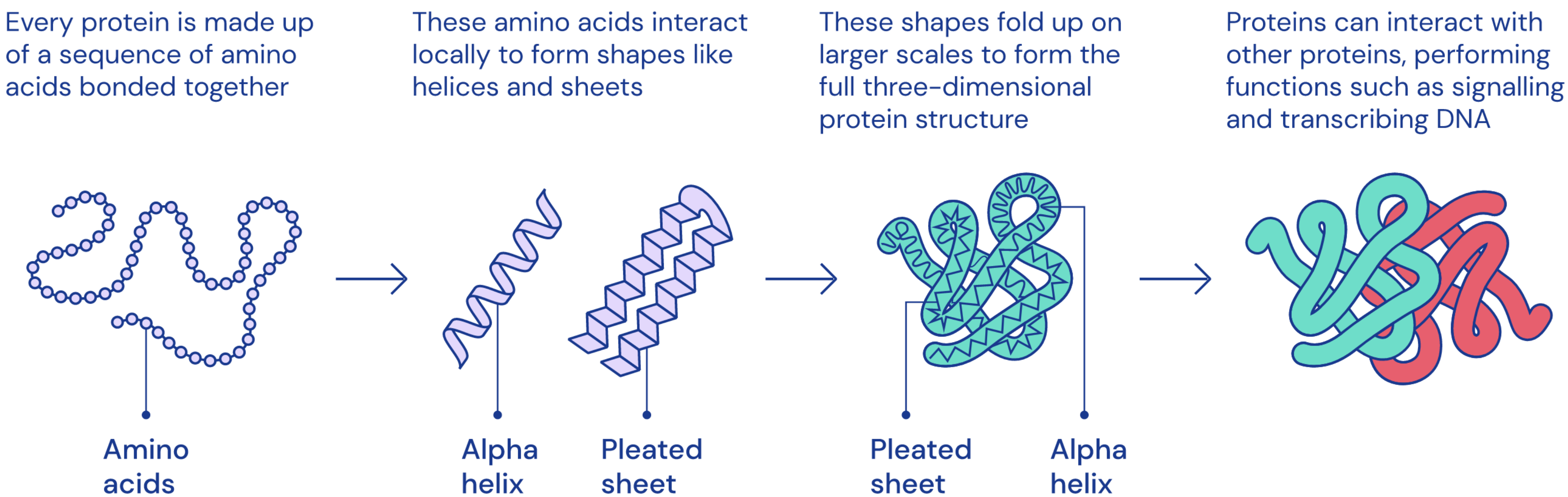


Image credit: [DeepMind]

The Protein Folding Problem

Computationally predict a protein's 3D structure based solely on its amino acid sequence

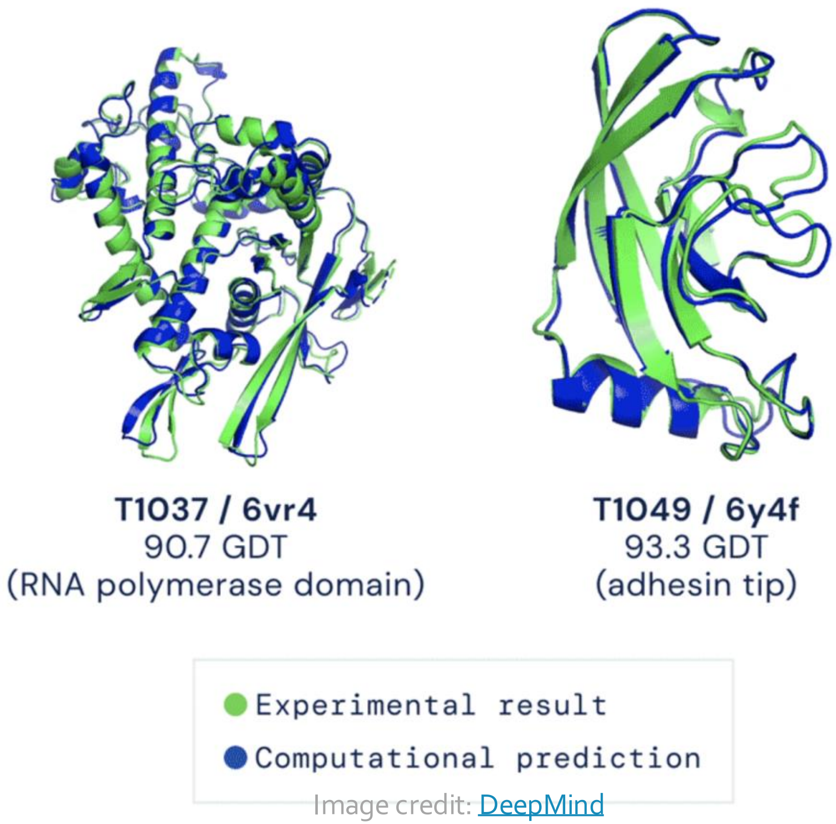


Image credit: [DeepMind]

AlphaFold: Impact

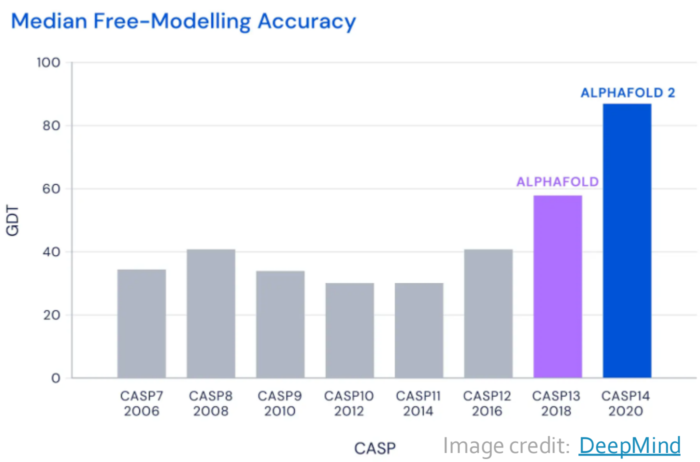


Image credit: DeepMind

Image credit: SingularityHub

AlphaFold's AI could change the world of biological science as we know it

DeepMind's latest AI breakthrough can accurately predict the way proteins fold

Has Artificial Intelligence 'Solved' Biology's Protein-Folding Problem?

12-14-20

DeepMind's latest AI breakthrough could turbocharge drug discovery

AlphaFold: Solving Protein Folding

- **Key idea:** “Spatial graph”
  - **Nodes:** Amino acids in a protein sequence
  - **Edges:** Proximity between amino acids (residues)

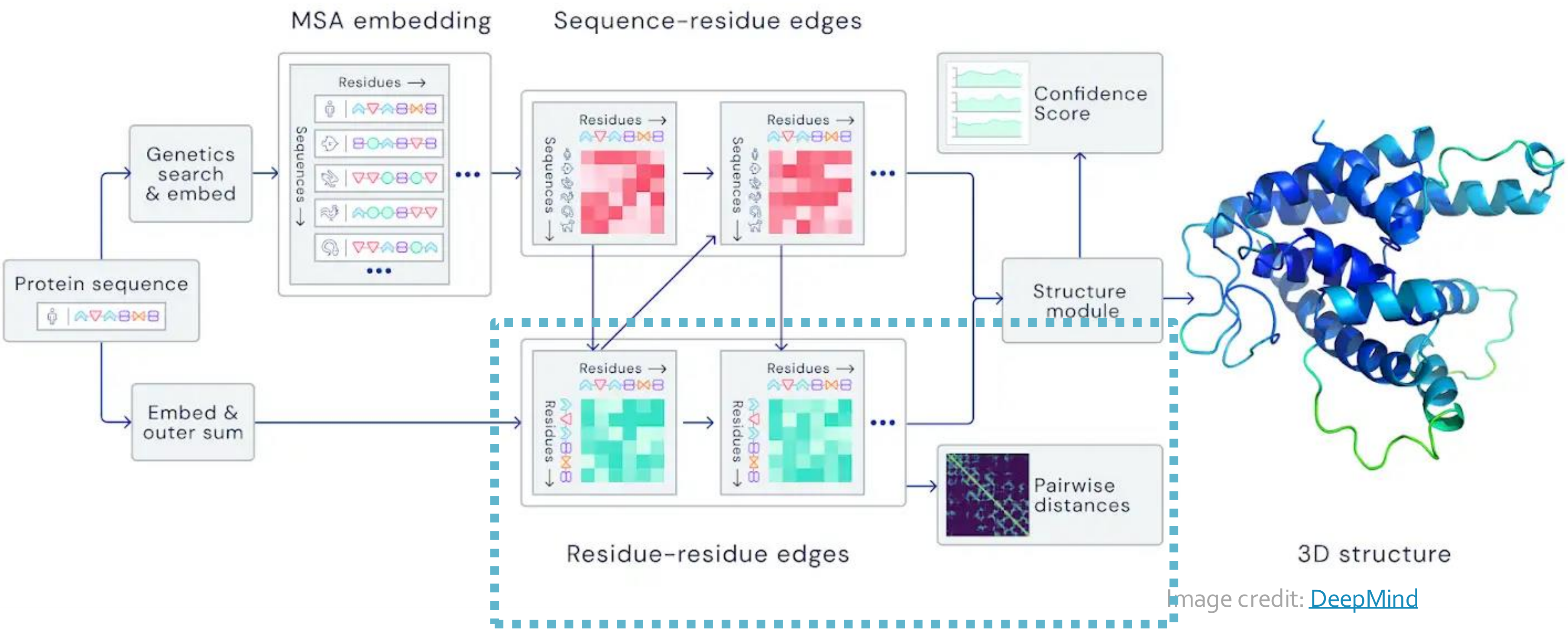


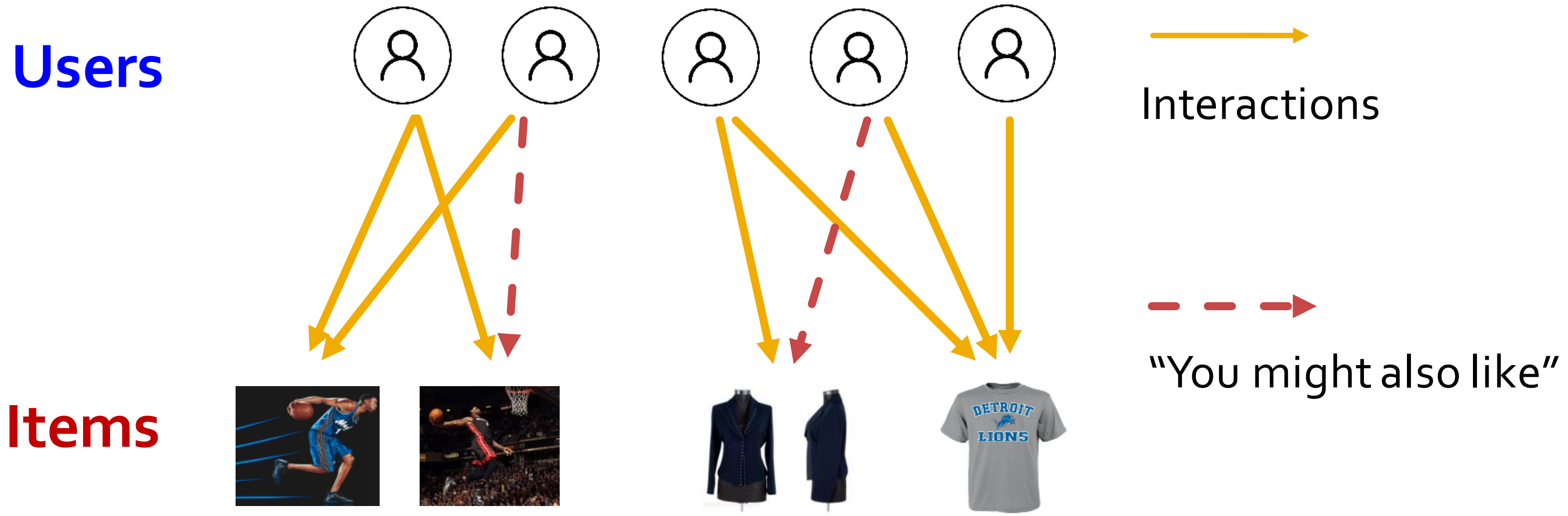
Image credit: DeepMind

Spatial graph

Examples of Edge-level ML Tasks

Example (2): Recommender Systems

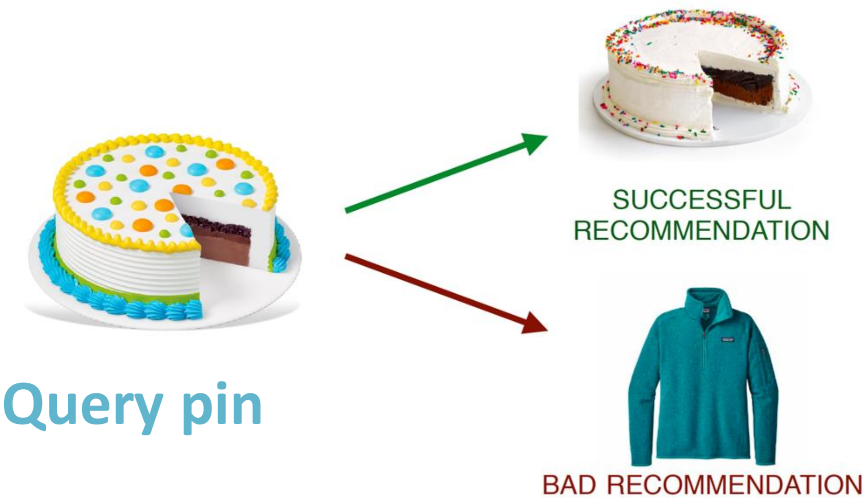
- **Users interacts with items**
  - Watch movies, buy merchandise, listen to music
  - **Nodes:** Users and items
  - **Edges:** User-item interactions
- **Goal: Recommend items users might like**



Ying et al., Graph Convolutional Neural Networks for Web-Scale Recommender Systems, KDD 2018

PinSage: Graph-based Recommender

Task: Recommend related pins to users

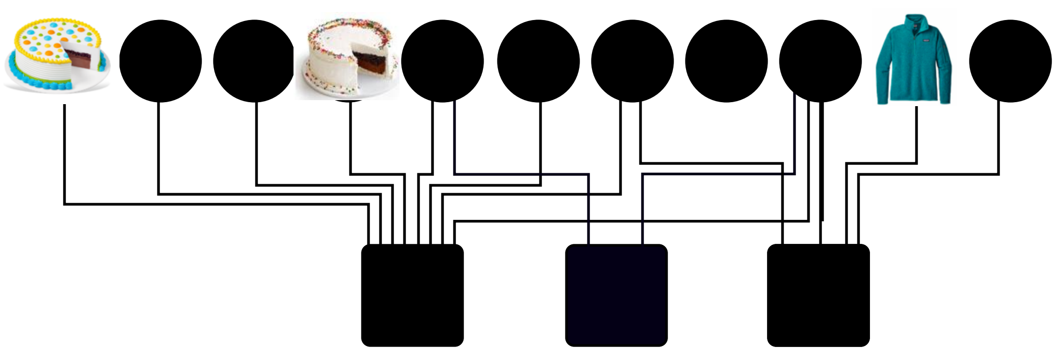


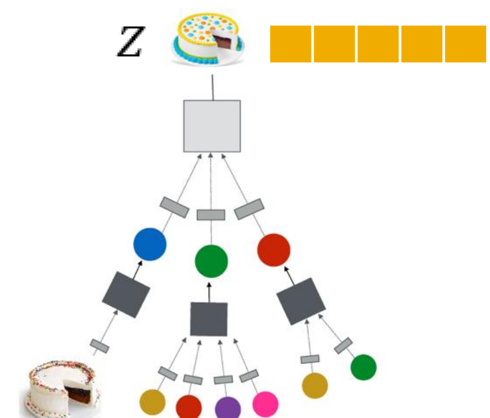
Query pin

Task: Learn node embeddings z_i such that

$$d(z_{cake1}, z_{cake2}) < d(z_{cake1}, z_{sweater})$$

Predict whether two nodes in a graph are related



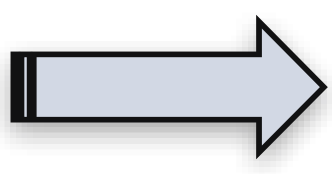


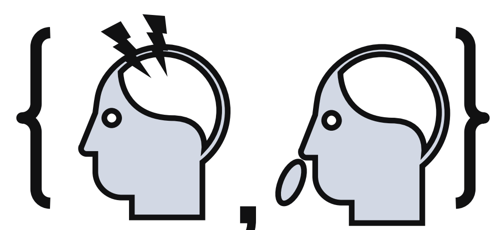
Example (3): Drug Side Effects

Many patients take multiple drugs to treat complex or co-existing diseases:

- 46% of people ages 70-79 take more than 5 drugs
- Many patients take more than 20 drugs to treat heart disease, depression, insomnia, etc.

Task: Given a pair of drugs predict adverse side effects





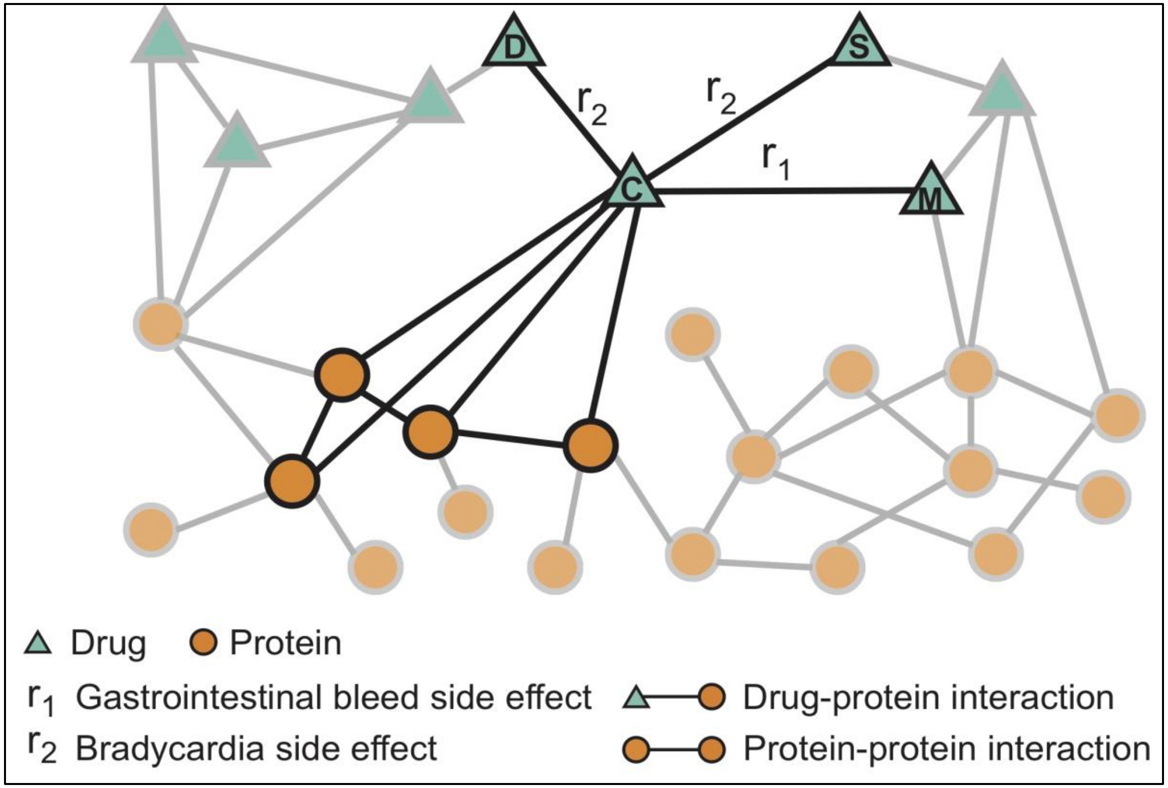
30% prob.

65% prob.

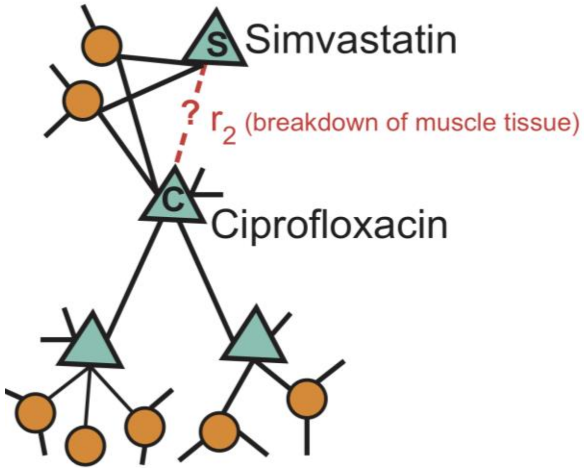
Zitnik et al., Modeling Polypharmacy Side Effects with Graph Convolutional Networks, Bioinformatics 2018

Biomedical Graph Link Prediction

- **Nodes**: Drugs & Proteins
- **Edges**: Interactions



Query: How likely will Simvastatin and Ciprofloxacin, when taken together, break down muscle tissue?



Zitnik et al., Modeling Polypharmacy Side Effects with Graph Convolutional Networks, Bioinformatics 2018

Results: *De novo* Predictions

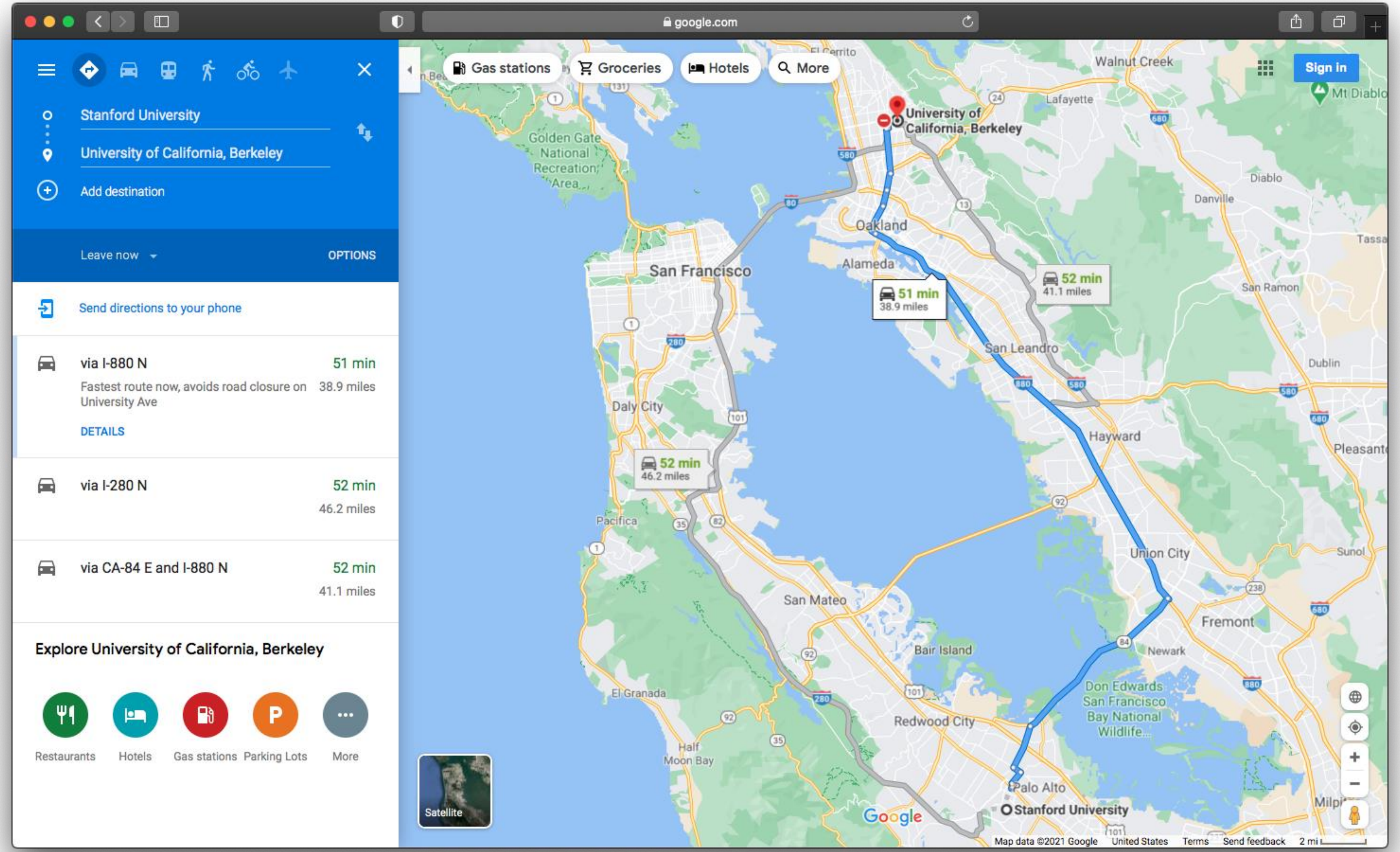
Rank	Drug *c*	Drug *d*	Side effect *r*	Evidence found
1	Pyrimethamine	Aliskiren	Sarcoma	Stage *et al.* 2015
2	Tigecycline	Bimatoprost	Autonomic neuropathy	
3	Omeprazole	Dacarbazine	Telangiectases	
4	Tolcapone	Pyrimethamine	Breast disorder	Bicker *et al.* 2017
5	Minoxidil	Paricalcitol	Cluster headache	
6	Omeprazole	Amoxicillin	Renal tubular acidosis	Russo *et al.* 2016
7	Anagrelide	Azelaic acid	Cerebral thrombosis	
8	Atorvastatin	Amlodipine	Muscle inflammation	Banakh *et al.* 2017
9	Aliskiren	Tioconazole	Breast inflammation	Parving *et al.* 2012
10	Estradiol	Nadolol	Endometriosis	

Case Report

Severe Rhabdomyolysis due to Presumed Drug Interactions between Atorvastatin with Amlodipine and Ticagrelor

Examples of Subgraph-level ML Tasks

Example (4): Traffic Prediction



Road Network as a Graph

- **Nodes:** Road segments
- **Edges:** Connectivity between road segments
- **Prediction:** Time of Arrival (ETA)

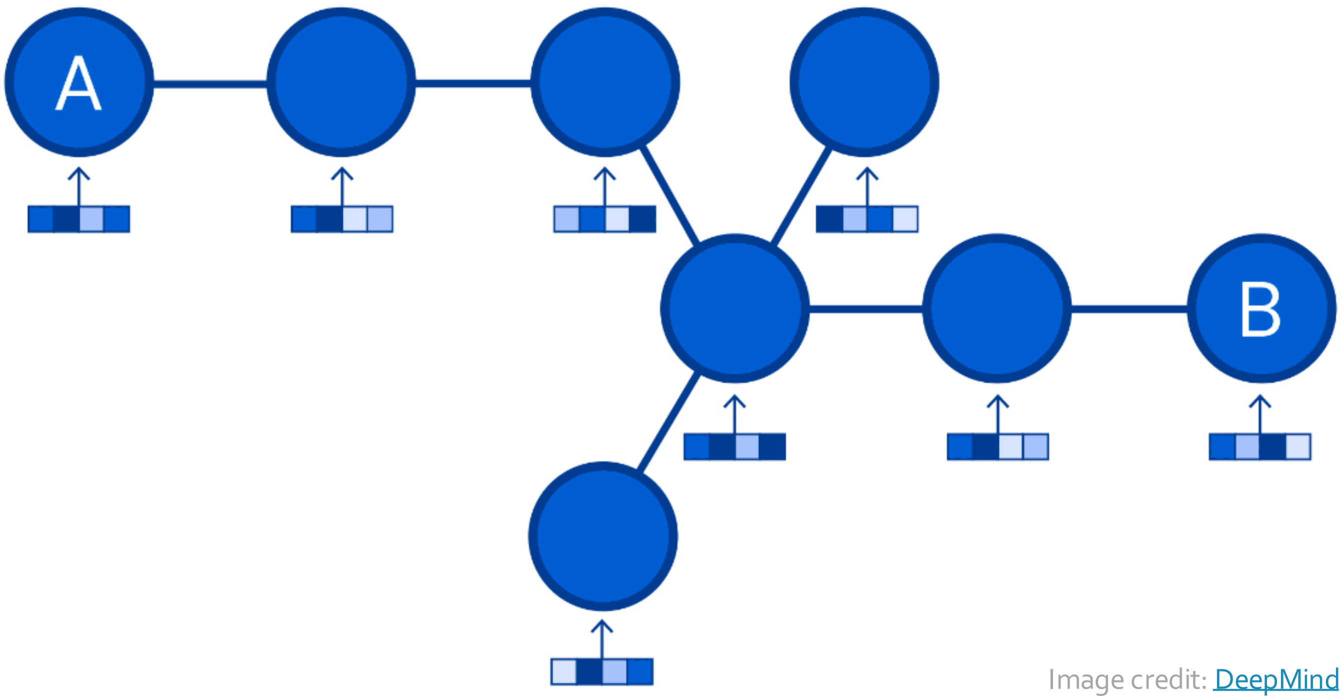
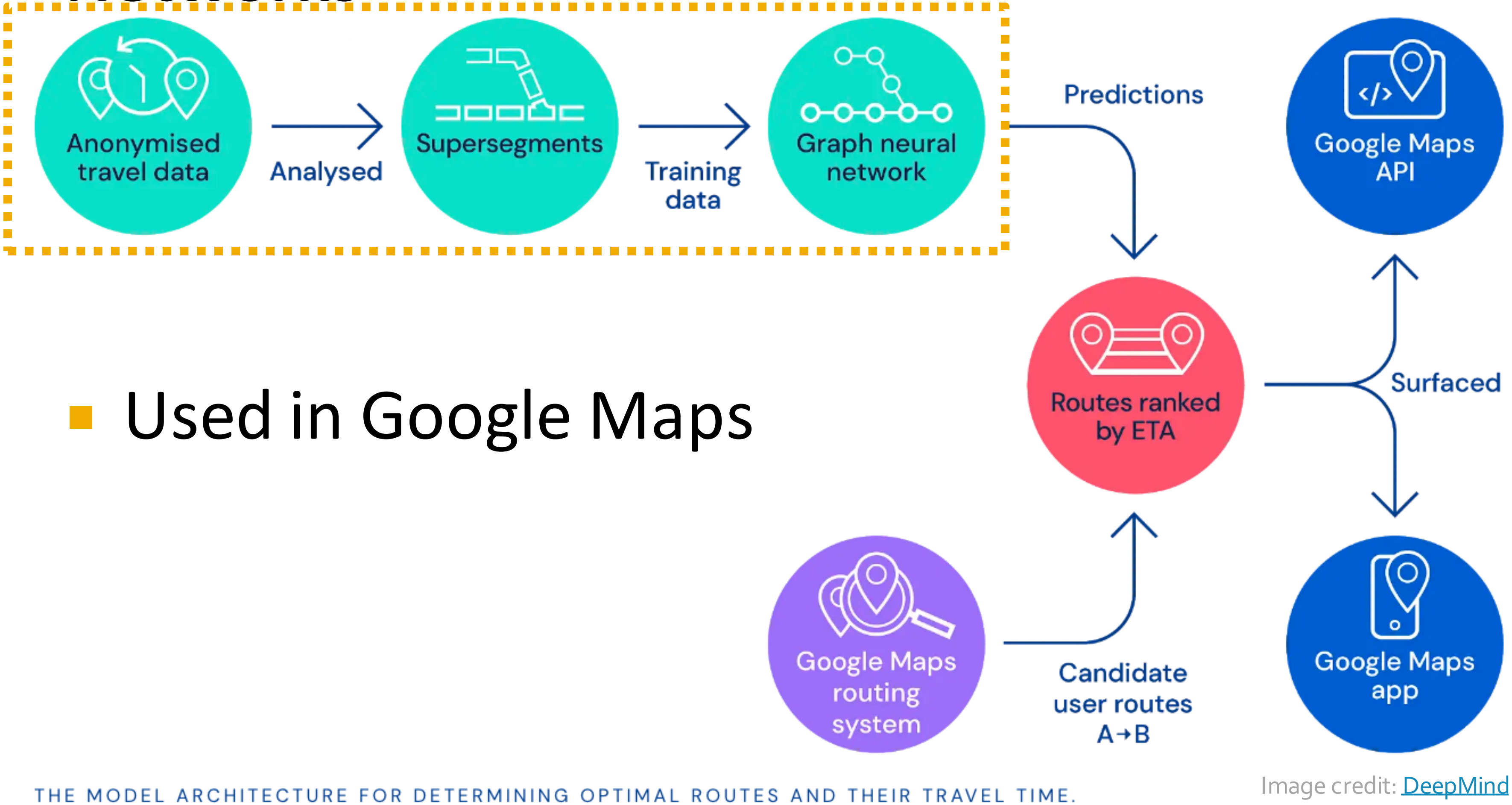


Image credit: [DeepMind]

Traffic Prediction via GNN

Predicting Time of Arrival with Graph Neural Networks



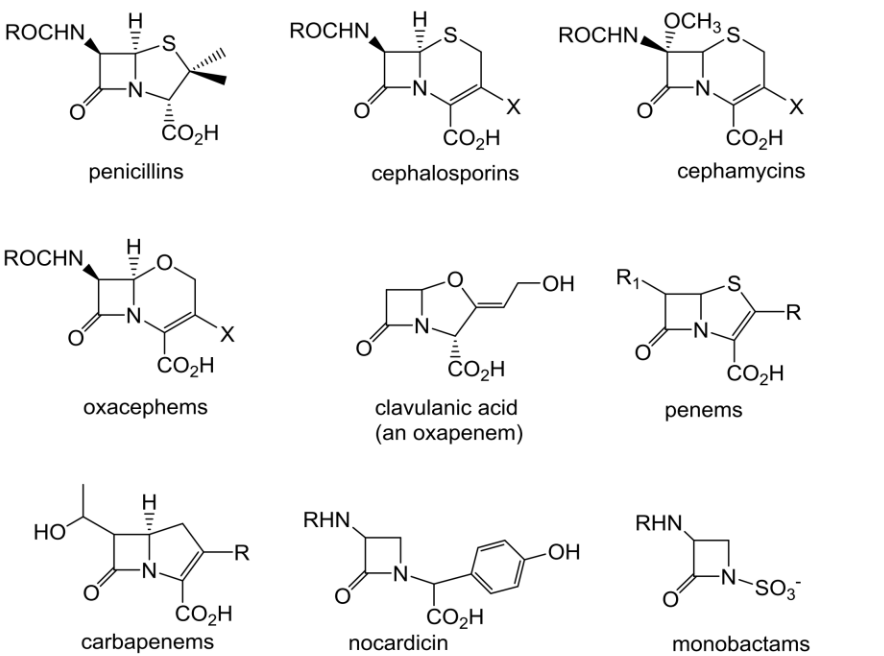
- Used in Google Maps

Image credit: DeepMind

Examples of Graph-level ML Tasks

Example (5): Drug Discovery

- **Antibiotics are small molecular graphs**
  - **Nodes:** Atoms
  - **Edges:** Chemical bonds



Konaklieva, Monika I. "Molecular targets of β-lactam-based antimicrobials: beyond the usual suspects." Antibiotics 3.2 (2014): 128-142.

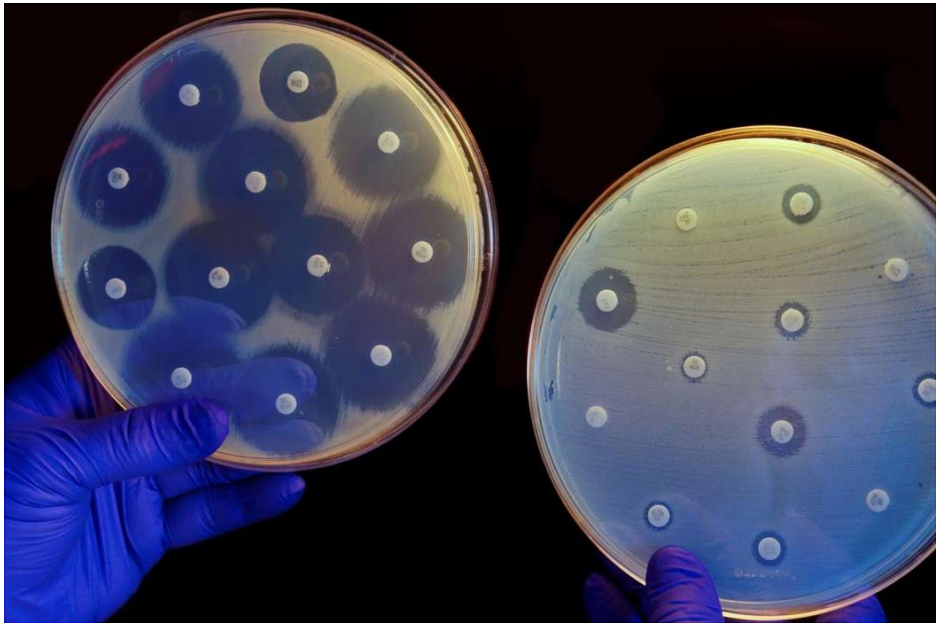
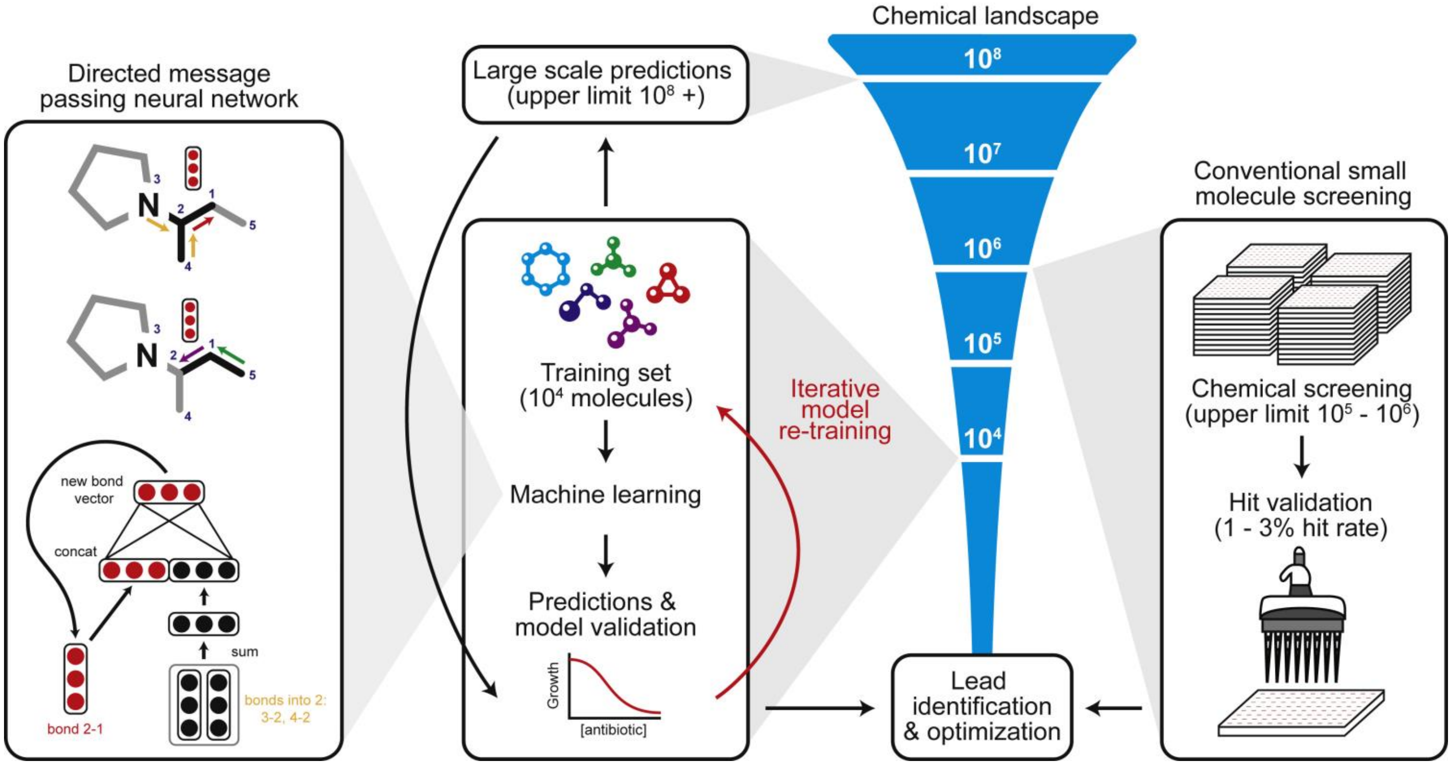


Image credit: [CNN]

Deep Learning for Antibiotic Discovery

- A Graph Neural Network **graph classification model**
- Predict promising molecules from a pool of candidates

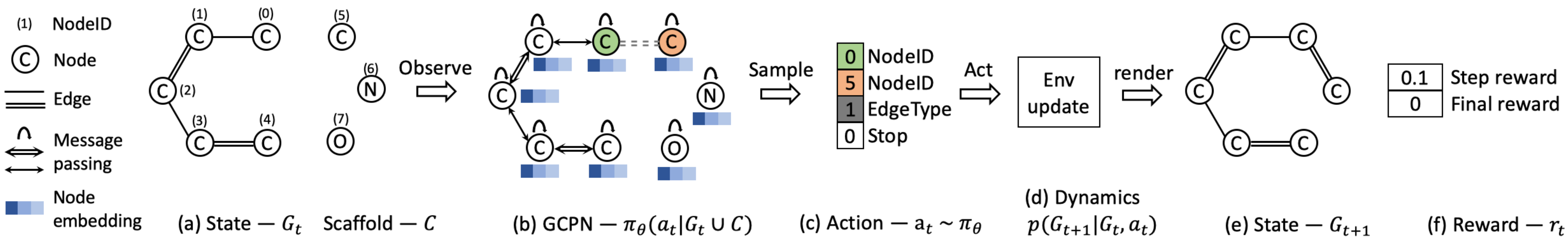


Stokes, Jonathan M., et al. "A deep learning approach to antibiotic discovery." Cell 180.4 (2020): 688-702.

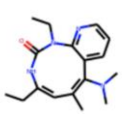
You et al., [Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation](), NeurIPS 2018

Molecule Generation / Optimization

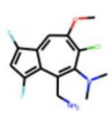
Graph generation: Generating novel molecules



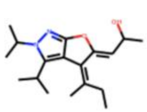
Use case 1: Generate novel molecules with high Drug likeness value



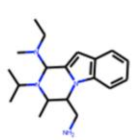
0.948



0.945

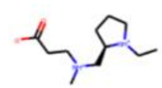


0.944

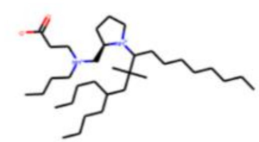


0.941

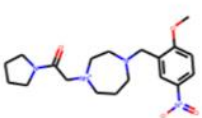
Use case 2: Optimize existing molecules to have desirable properties



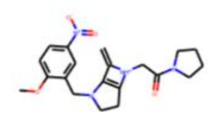
-8.32



-0.71



-5.55

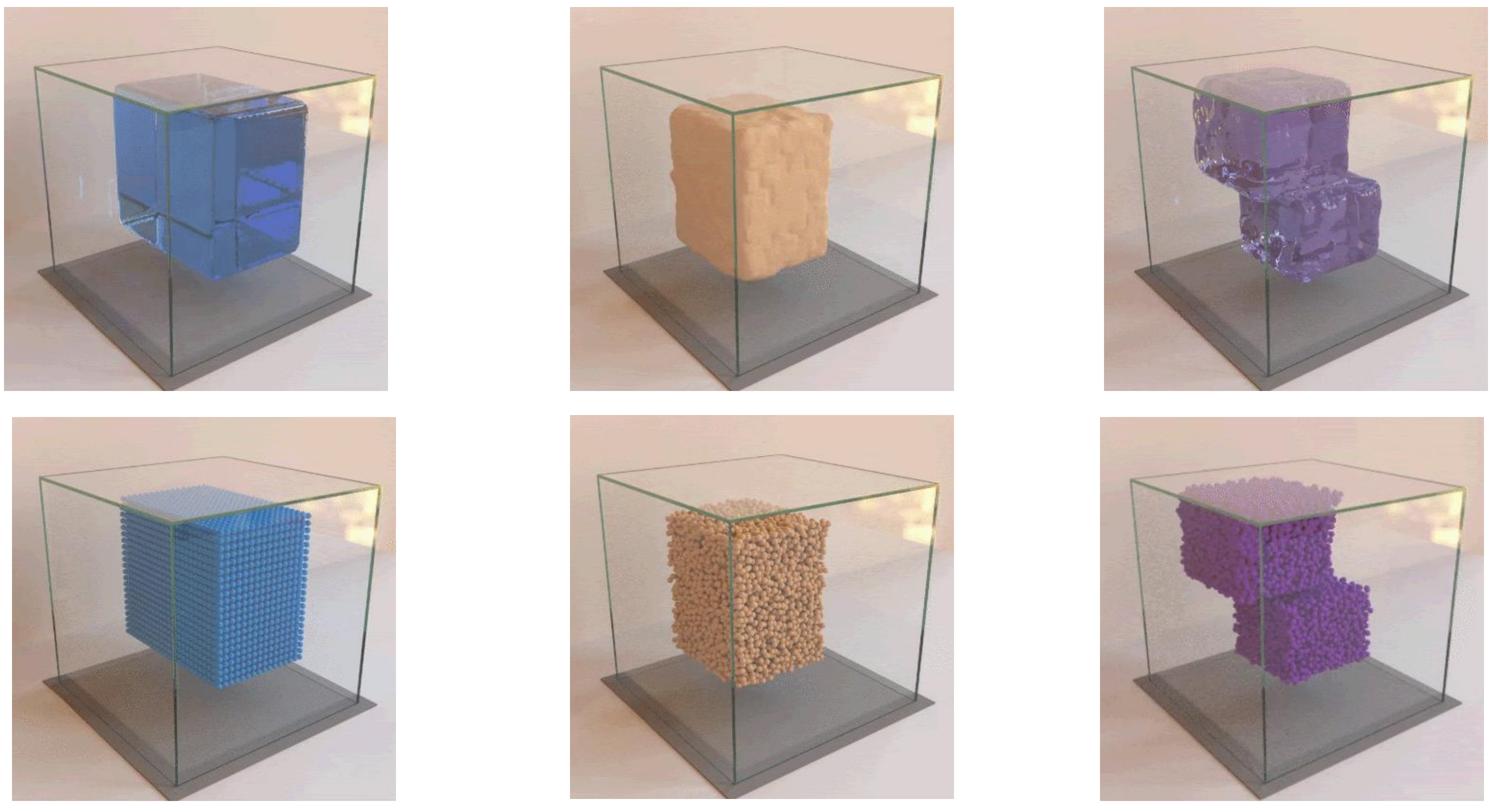


-1.78

Drug likeness

Example (6): Physics Simulation

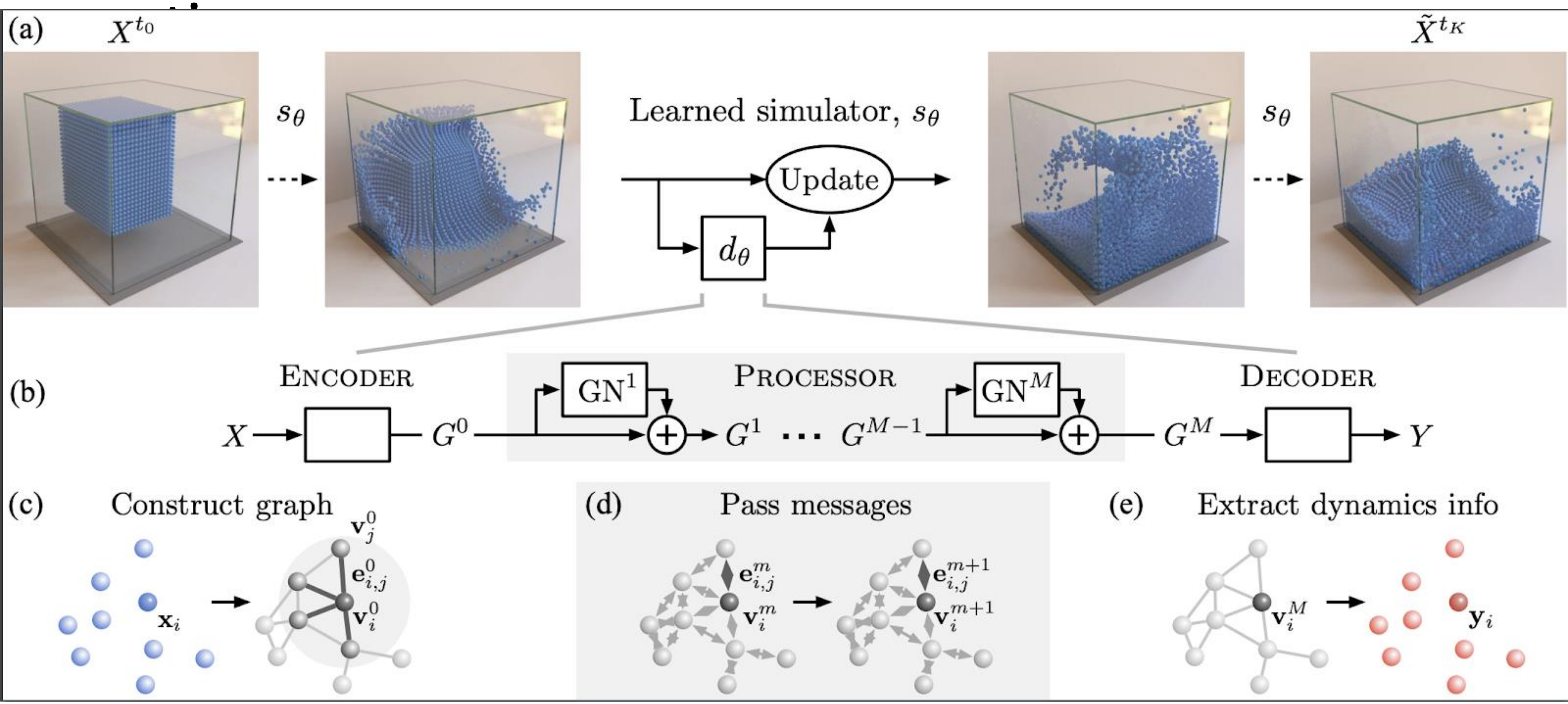
Physical simulation as a graph:
- **Nodes**: Particles
- **Edges**: Interaction between particles



Simulation Learning Framework

A graph evolution task:

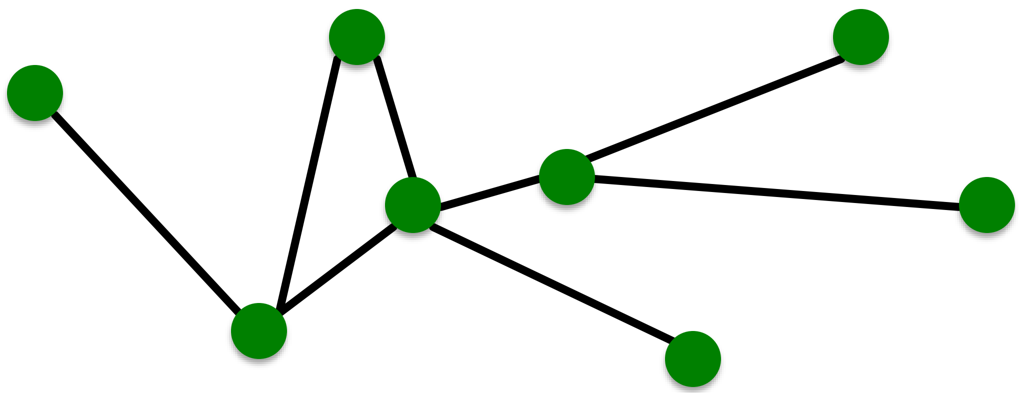
- **Goal**: Predict how a graph will evolve over time



Stanford CS224W: Choice of Graph Representation

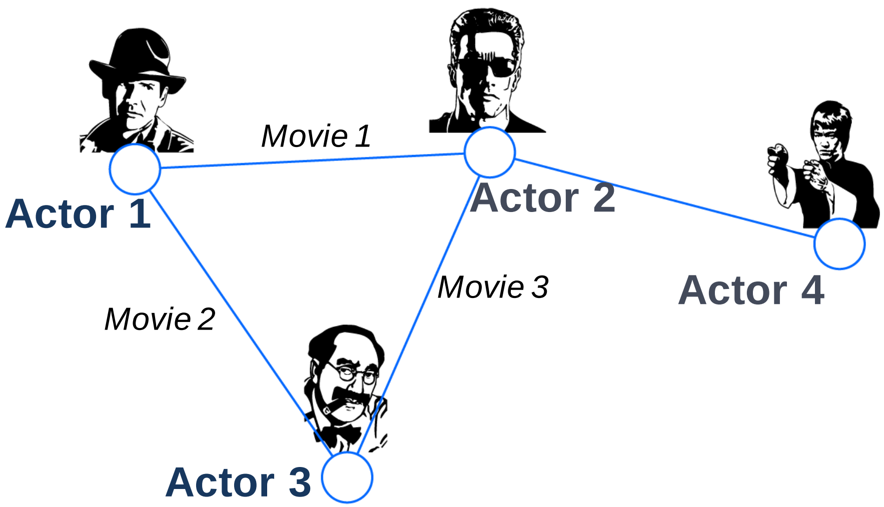
CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

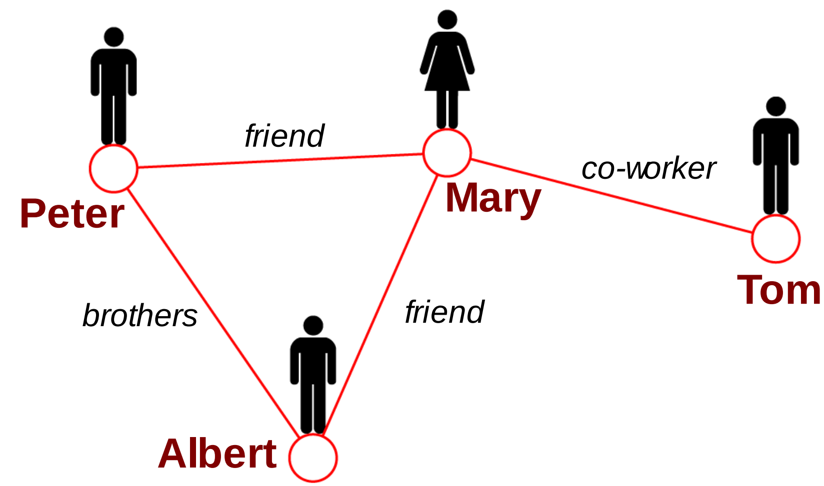
Components of a Network

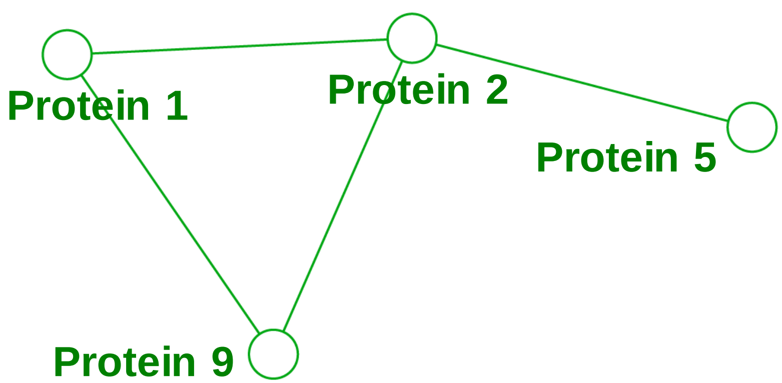


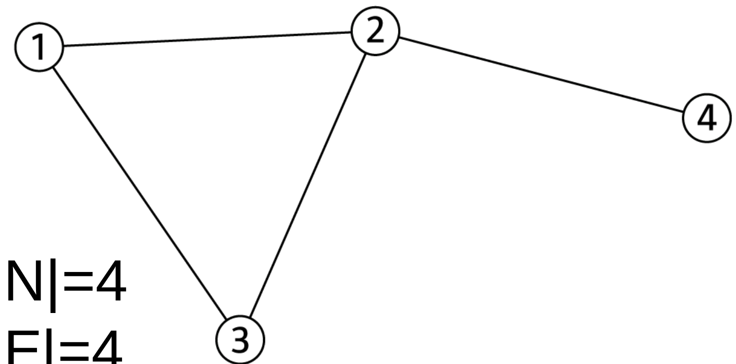
- **Objects:** nodes, vertices N
- **Interactions:** links, edges E
- **System:** network, graph $G(N,E)$

Graphs: A Common Language









|N|=4
|E|=4

Choosing a Proper Representation

- If you connect individuals that work with each other, you will explore a **professional network**

Image credit: Euro Scientists

- If you connect those that have a sexual relationship, you will be exploring **sexual networks**

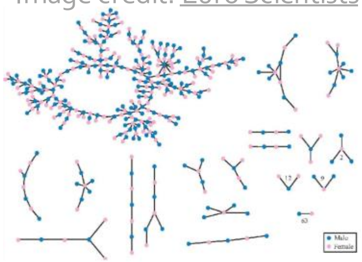


Image credit: ResearchGate

- If you connect scientific papers that cite each other, you will be studying the **citation network**

- **If you connect all papers with the same word in the title, what will you be exploring?** It is a network, nevertheless

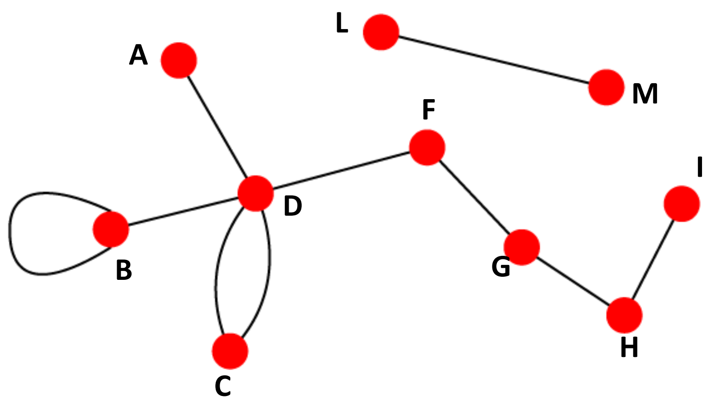
How do you define a graph?

- **How to build a graph:**
  - **What are nodes?**
  - **What are edges?**
- **Choice of the proper network representation of a given domain/problem determines our ability to use networks successfully:**
  - In some cases, there is a unique, unambiguous representation
  - In other cases, the representation is by no means unique
  - The way you assign links will determine the nature of the question you can study

Directed vs. Undirected Graphs

Undirected

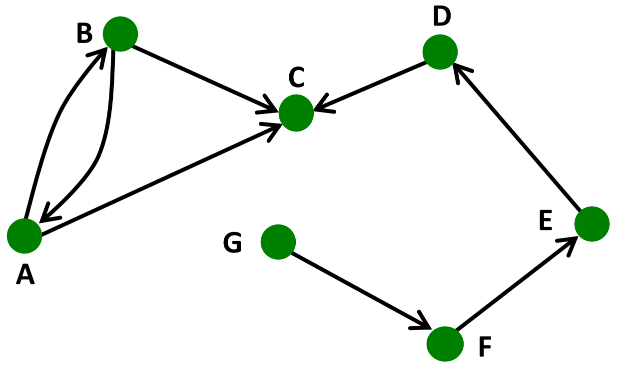
- **Links:** undirected (symmetrical, reciprocal)



- **Examples:**
  - Collaborations
  - Friendship on Facebook

Directed

- **Links:** directed (arcs)



- **Examples:**
  - Phone calls
  - Following on Twitter

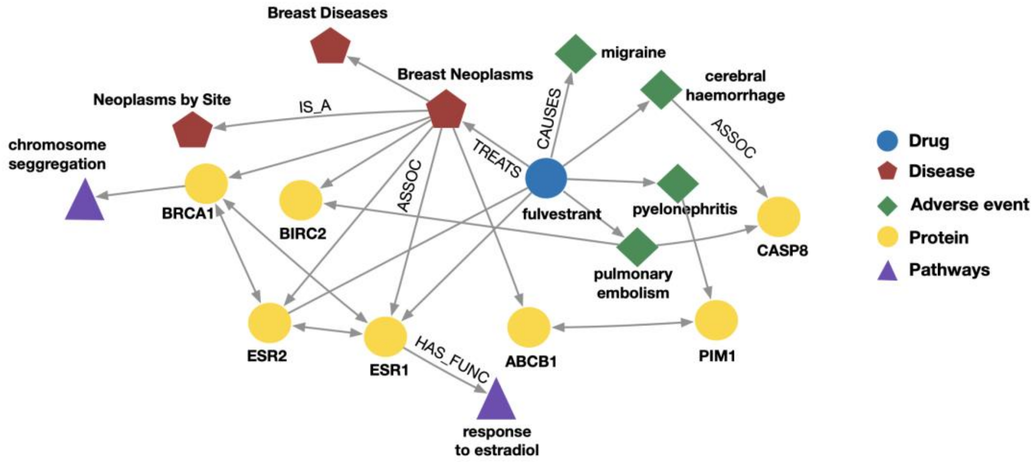
Heterogeneous Graphs

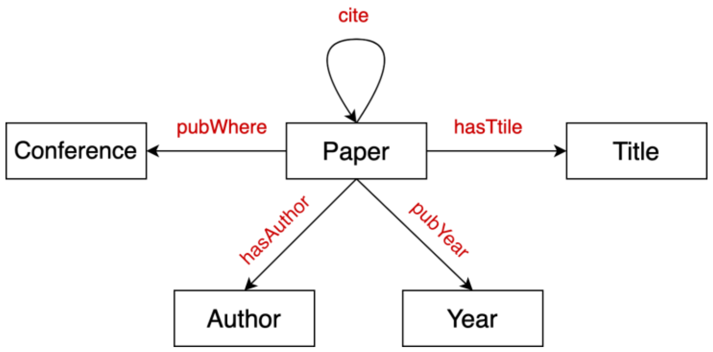
- **A heterogeneous graph is defined as**

$$\boldsymbol{G = (V, E, R, T)}$$

  - Nodes with node types $v_i \in V$
  - Edges with relation types $(v_i, r, v_j) \in E$
  - Node type $T(v_i)$
  - Relation type $r \in R$

Many Graphs are Heterogeneous Graphs





Biomedical Knowledge Graphs

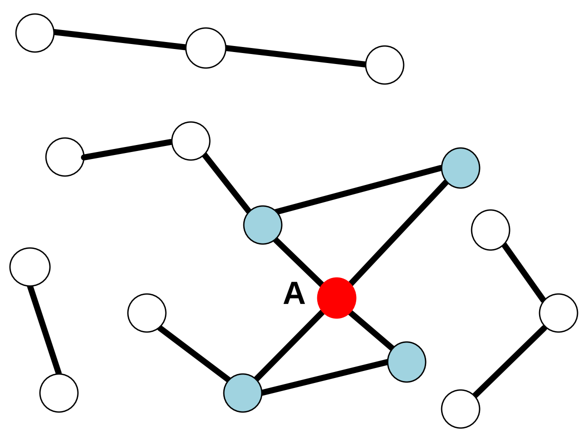
Example node: Migraine
Example edge: (fulvestrant, Treats, Breast Neoplasms)
Example node type: Protein
Example edge type (relation): Causes

Academic Graphs

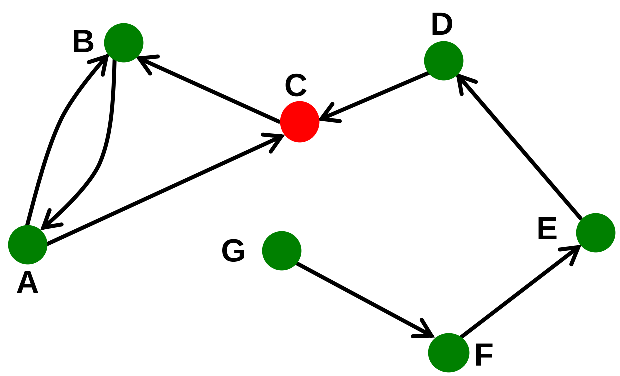
Example node: ICML
Example edge: (GraphSAGE, NeurIPS)
Example node type: Author
Example edge type (relation): pubYear

Node Degrees

Undirected



Directed



Source: Node with $k^{in} = 0$

Sink: Node with $k^{out} = 0$

Node degree, k_i: the number of edges adjacent to node ***i***

$$k_A = 4$$

Avg. degree: $\bar{k} = \langle k \rangle = \frac{1}{N}\sum_{i=1}^{N} k_i = \frac{2E}{N}$

In directed networks we define an **in-degree** and **out-degree.**

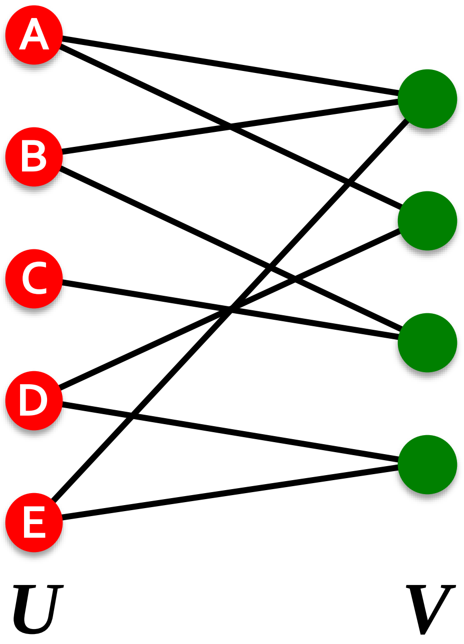
The (total) degree of a node is the sum of in- and out-degrees.

$$k_C^{in} = 2 \qquad k_C^{out} = 1 \qquad k_C = 3$$

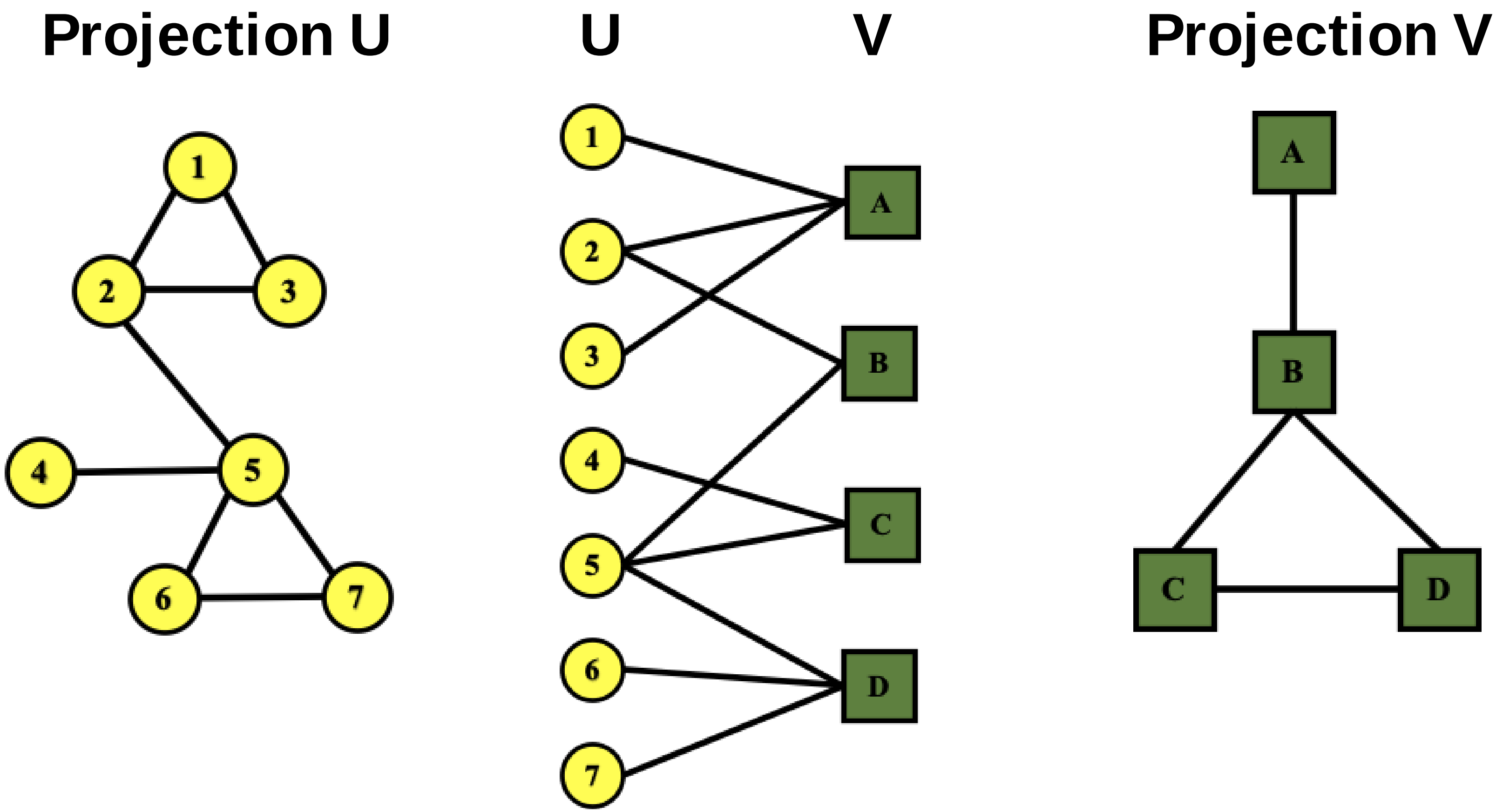
$$\bar{k} = \frac{E}{N} \qquad \overline{k^{in}} = \overline{k^{out}}$$

Bipartite Graph

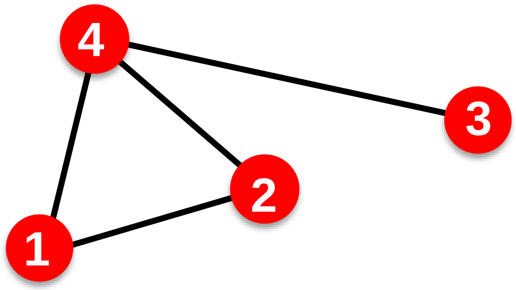
- **Bipartite graph** is a graph whose nodes can be divided into two disjoint sets ***U*** and ***V*** such that every link connects a node in ***U*** to one in ***V***; that is, ***U*** and ***V*** are **independent sets**
- **Examples:**
  - Authors-to-Papers (they authored)
  - Actors-to-Movies (they appeared in)
  - Users-to-Movies (they rated)
  - Recipes-to-Ingredients (they contain)
- **"Folded" networks:**
  - Author collaboration networks
  - Movie co-rating networks

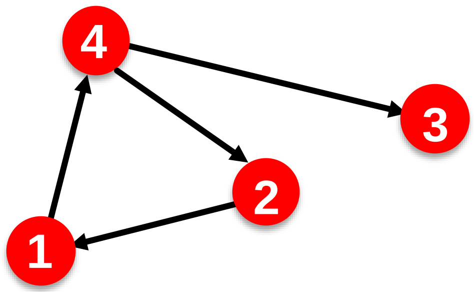


Folded/Projected Bipartite Graphs



Representing Graphs: Adjacency Matrix





$A_{ij} = 1$ if there is a link from node ***i*** to node ***j***

$A_{ij} = 0$ otherwise

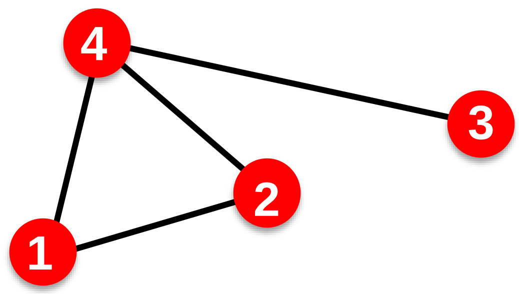
$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

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

Note that for a directed graph (right) the matrix is not symmetric.

Adjacency Matrix

Undirected



$$A_{ij} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

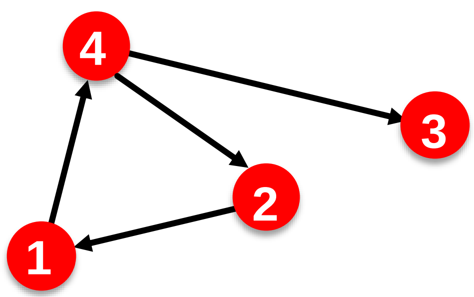
$$A_{ij} = A_{ji}$$
$$A_{ii} = 0$$

$$k_i = \sum_{j=1}^{N} A_{ij}$$

$$k_j = \sum_{i=1}^{N} A_{ij}$$

$$L = \frac{1}{2}\sum_{i=1}^{N} k_i = \frac{1}{2}\sum_{ij}^{N} A_{ij}$$

Directed



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

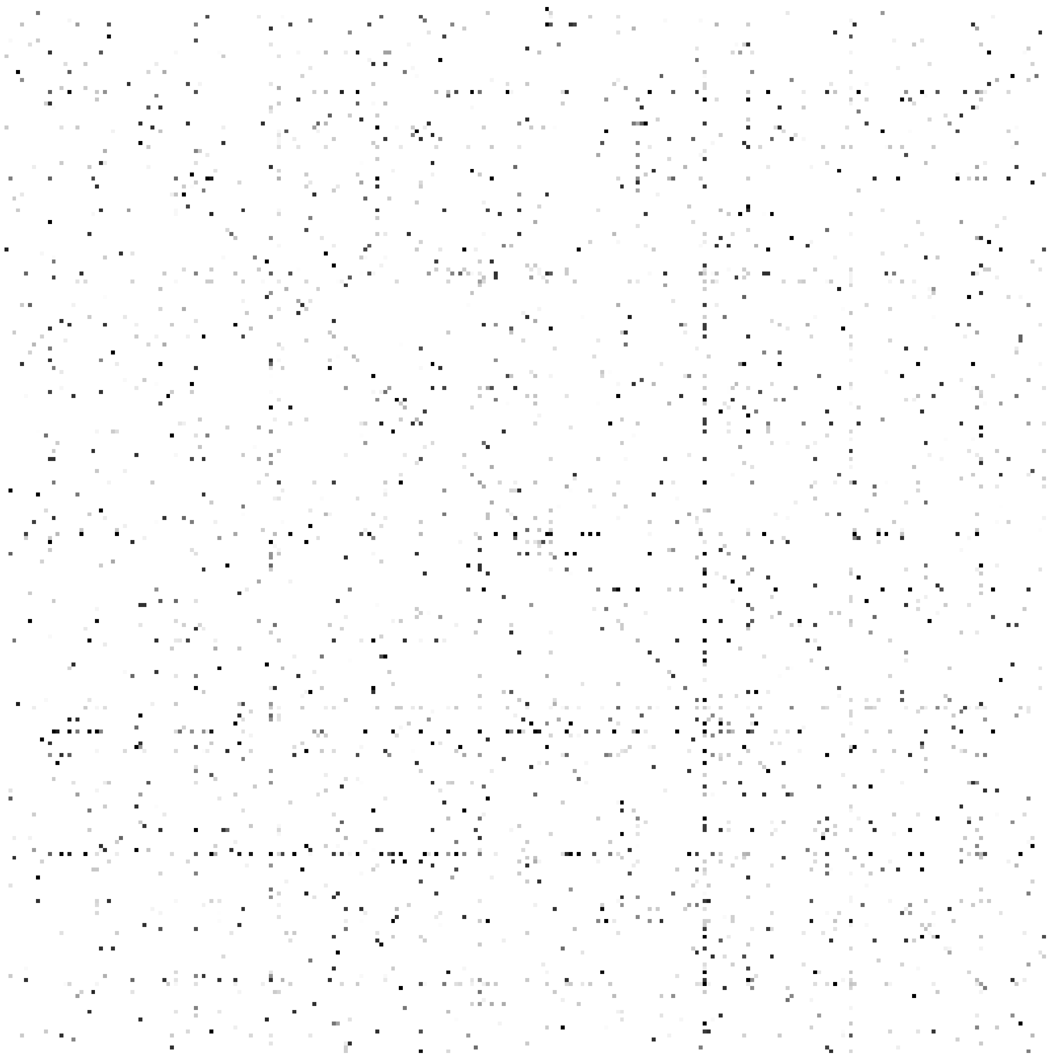
$$A_{ij} \neq A_{ji}$$
$$A_{ii} = 0$$

$$k_i^{out} = \sum_{j=1}^{N} A_{ij}$$

$$k_j^{in} = \sum_{i=1}^{N} A_{ij}$$

$$L = \sum_{i=1}^{N} k_i^{in} = \sum_{j=1}^{N} k_j^{out} = \sum_{i,j}^{N} A_{ij}$$

Adjacency Matrices are Sparse



Networks are Sparse Graphs

Most real-world networks are sparse

$E \ll E_{max}$ (or $\underline{k} \ll N-1$)

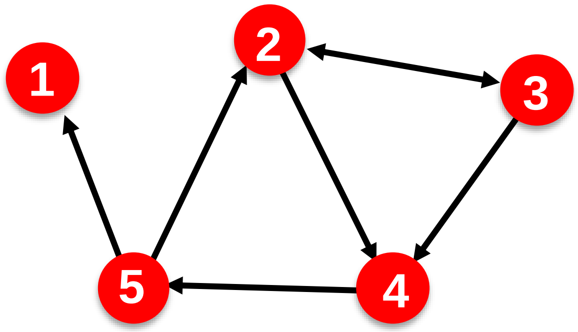
NETWORK	NODES	LINKS	DIRECTED/ UNDIRECTED	N	L	<k>
Internet	Routers	Internet connections	Undirected	192,244	609,066	6.33
WWW	Webpages	Links	Directed	325,729	1,497,134	4.60
Power Grid	Power plants, transformers	Cables	Undirected	4,941	6,594	2.67
Phone Calls	Subscribers	Calls	Directed	36,595	91,826	2.51
Email	Email Addresses	Emails	Directed	57,194	103,731	1.81
Science Collaboration	Scientists	Co-authorship	Undirected	23,133	93,439	8.08
Actor Network	Actors	Co-acting	Undirected	702,388	29,397,908	83.71
Citation Network	Paper	Citations	Directed	449,673	4,689,479	10.43
E. Coli Metabolism	Metabolites	Chemical reactions	Directed	1,039	5,802	5.58
Protein Interactions	Proteins	Binding interactions	Undirected	2,018	2,930	2.90

Consequence: Adjacency matrix is filled with zeros!

(**Density of the matrix (E/N^2):** WWW=1.51×10^{-5}, MSN IM = 2.27×10^{-8})

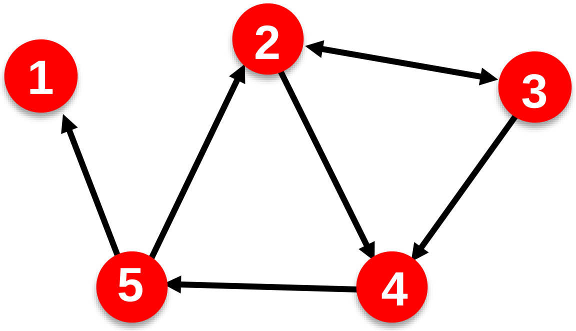
Representing Graphs: Edge list

- **Represent graph as a list of edges:**
  - (2, 3)
  - (2, 4)
  - (3, 2)
  - (3, 4)
  - (4, 5)
  - (5, 2)
  - (5, 1)



Representing Graphs: Adjacency list

- **Adjacency list:**
  - Easier to work with if network is
    - Large
    - Sparse
  - Allows us to quickly retrieve all neighbors of a given node
    - 1:
    - 2: 3, 4
    - 3: 2, 4
    - 4: 5
    - 5: 1, 2



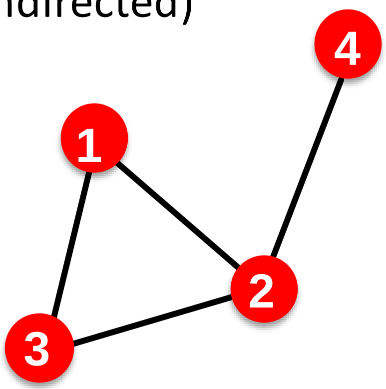
Node and Edge Attributes

Possible options:

- Weight (*e.g.*, frequency of communication)
- Ranking (best friend, second best friend…)
- Type (friend, relative, co-worker)
- Sign: Friend vs. Foe, Trust vs. Distrust
- Properties depending on the structure of the rest of the graph: Number of common friends

More Types of Graphs

- **Unweighted**
  (undirected)



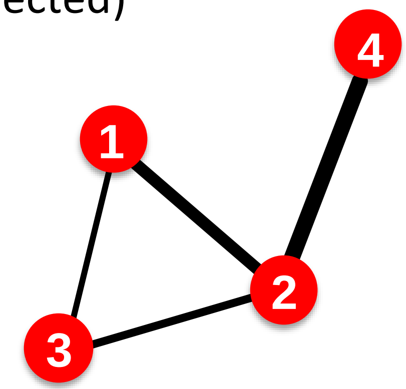
$$A_{ij} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

$$E = \frac{1}{2} \sum_{i,j=1}^{N} A_{ij} \qquad \bar{k} = \frac{2E}{N}$$

Examples: Friendship, Hyperlink

- **Weighted**
  (undirected)



$$A_{ij} = \begin{pmatrix} 0 & 2 & 0.5 & 0 \\ 2 & 0 & 1 & 4 \\ 0.5 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 \end{pmatrix}$$

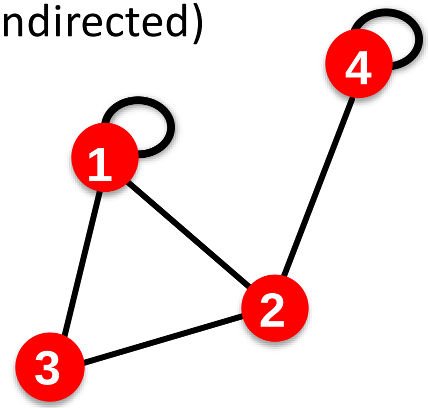
$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

$$E = \frac{1}{2} \sum_{i,j=1}^{N} nonzero(A_{ij}) \qquad \bar{k} = \frac{2E}{N}$$

Examples: Collaboration, Internet, Roads

More Types of Graphs

- **Self-edges (self-loops)**
  (undirected)



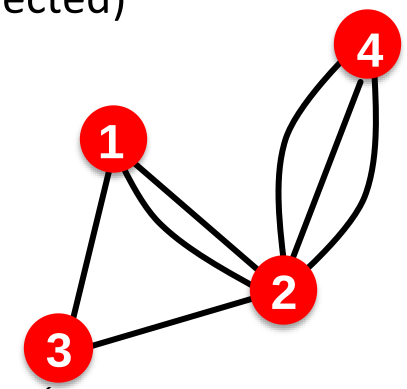
$$A_{ij} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

$$A_{ii} \neq 0 \qquad A_{ij} = A_{ji}$$

$$E = \frac{1}{2} \sum_{i,j=1, i \neq j}^{N} A_{ij} + \sum_{i=1}^{N} A_{ii}$$

Examples: Proteins, Hyperlinks

- **Multigraph**
  (undirected)



$$A_{ij} = \begin{pmatrix} 0 & 2 & 1 & 0 \\ 2 & 0 & 1 & 3 \\ 1 & 1 & 0 & 0 \\ 0 & 3 & 0 & 0 \end{pmatrix}$$

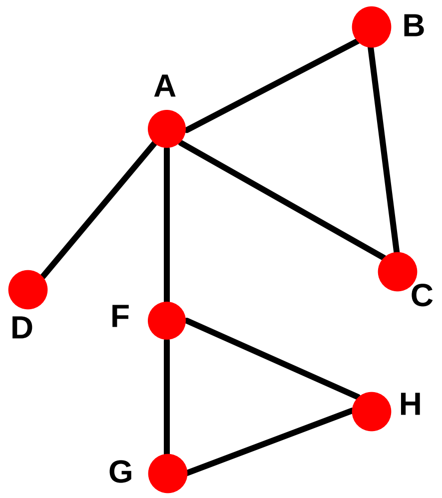
$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

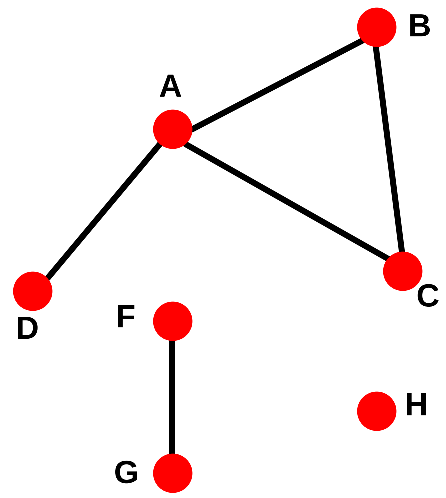
$$E = \frac{1}{2} \sum_{i,j=1}^{N} nonzero(A_{ij}) \qquad \bar{k} = \frac{2E}{N}$$

Examples: Communication, Collaboration

Connectivity of Undirected Graphs

- **Connected (undirected) graph:**
  - Any two vertices can be joined by a path
- A disconnected graph is made up by two or more connected components





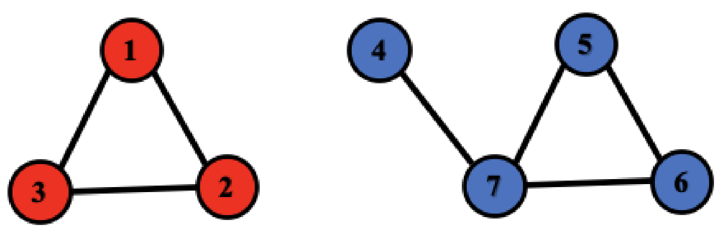
Largest Component:
Giant Component

Isolated node (node H)

Connectivity: Example

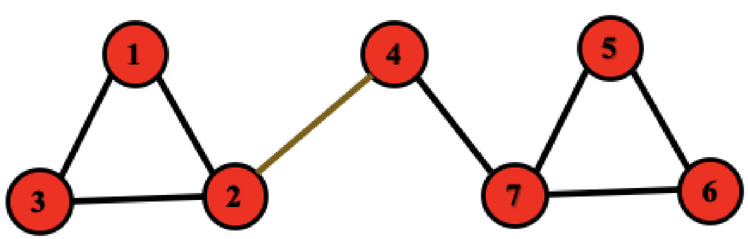
The adjacency matrix of a network with several components can be written in a block- diagonal form, so that nonzero elements are confined to squares, with all other elements being zero:

Disconnected



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

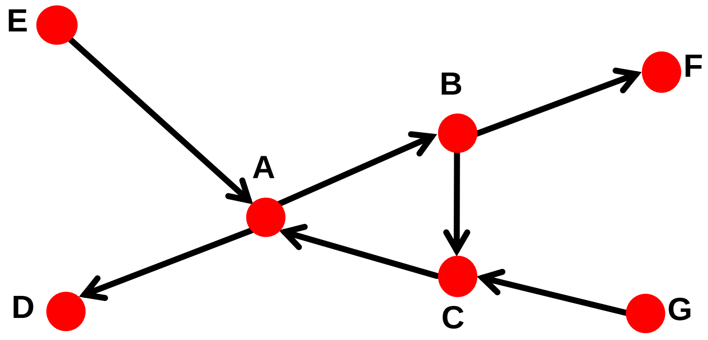
Connected



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

Connectivity of Directed Graphs

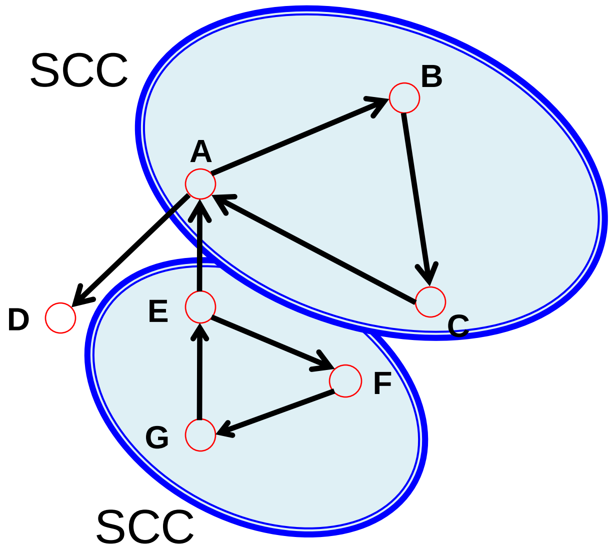
- **Strongly connected directed graph**
  - has a path from each node to every other node and vice versa (e.g., A-B path and B-A path)
- **Weakly connected directed graph**
  - is connected if we disregard the edge directions

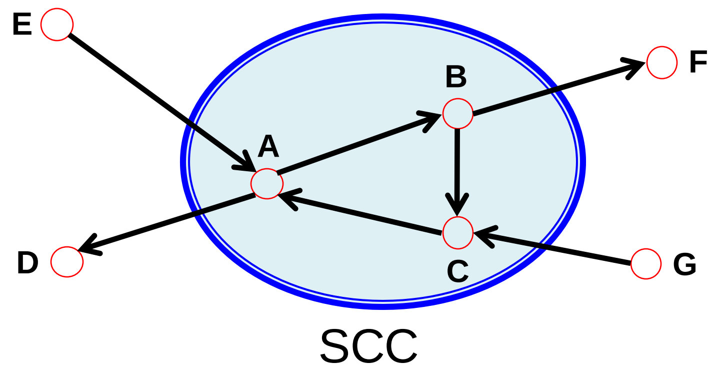


Graph on the left is connected but not strongly connected (e.g., there is no way to get from F to G by following the edge directions).

Connectivity of Directed Graphs

- **Strongly connected components (SCCs)** can be identified, but not every node is part of a nontrivial strongly connected component.





In-component: nodes that can reach the SCC,

Out-component: nodes that can be reached from the SCC.

Summary

- **Machine learning with Graphs**
  - Applications and use cases
- **Different types of tasks:**
  - Node level
  - Edge level
  - Graph level
- **Choice of a graph representation:**
  - Directed, undirected, bipartite, weighted, adjacency matrix