Deep Learning with Graphs

Jiaxuan You
CS598, Fall 2024
University of Illinois Urbana Champaign

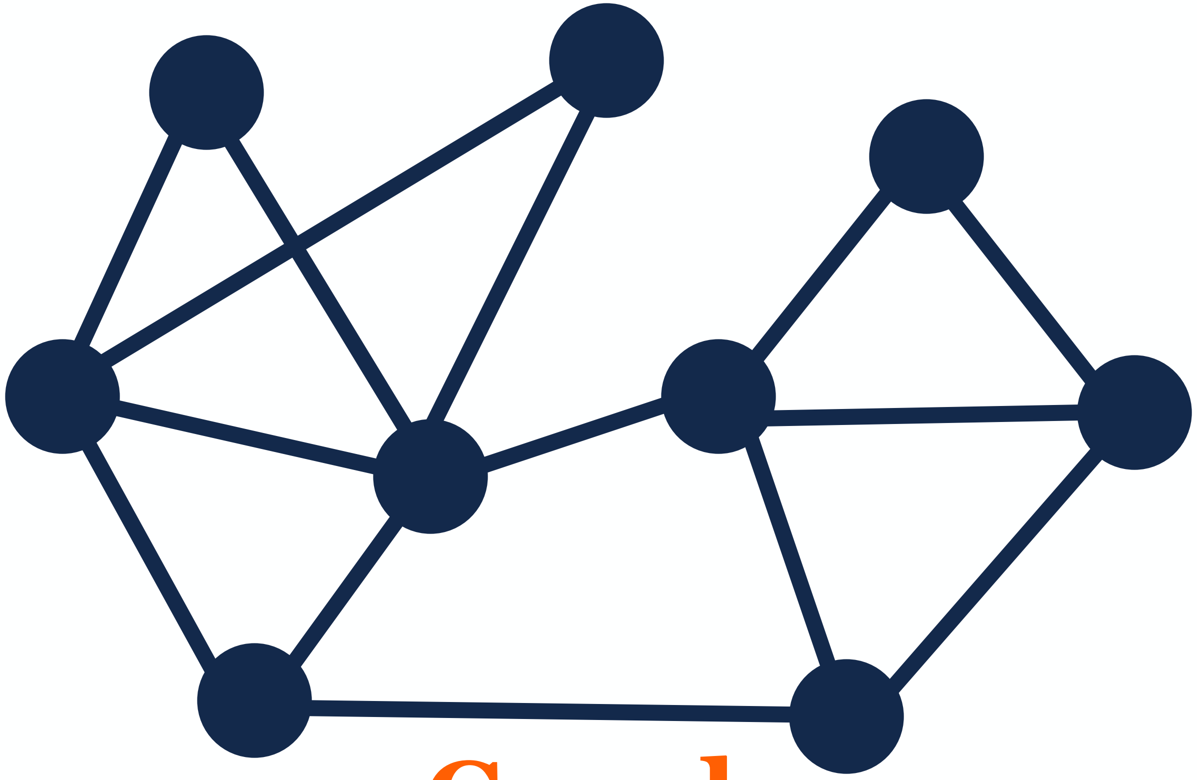
1

Machine Learning with Graphs

Why Graphs?

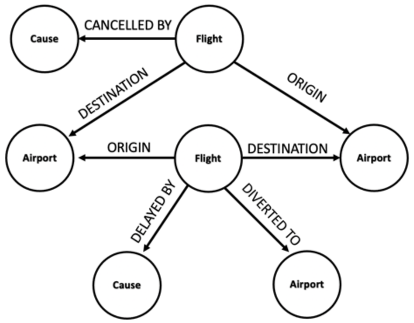
- **Graphs are a general language for describing and analyzing entities with relations or interactions.**

Why Graphs?



Graph

Many Types of Data are Graphs (1)



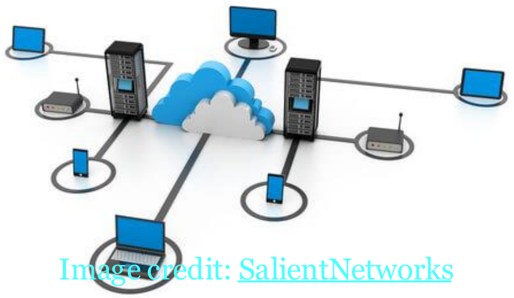
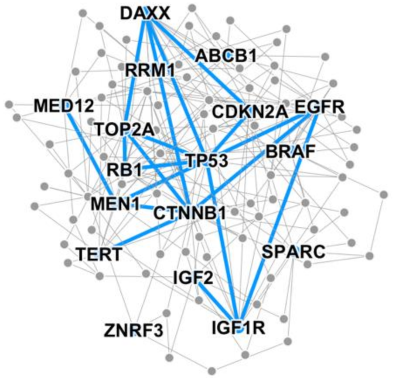


Image credit: [SalientNetworks]



Event Graphs **Computer Networks** **Disease Pathways**

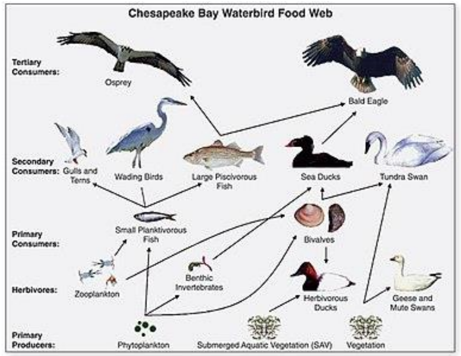


Image credit: [Wikipedia]

Image credit: [Pinterest]

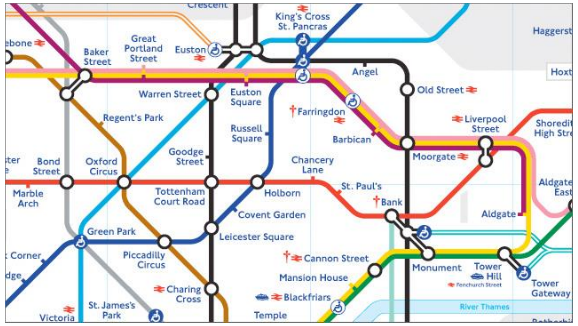


Image credit: [visitlondon.com]

Food Webs **Particle Networks** **Underground Networks**

Many Types of Data are Graphs (2)

Image credit: [Medium]

Social Networks

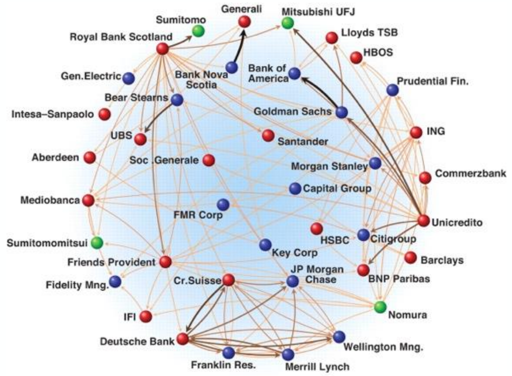
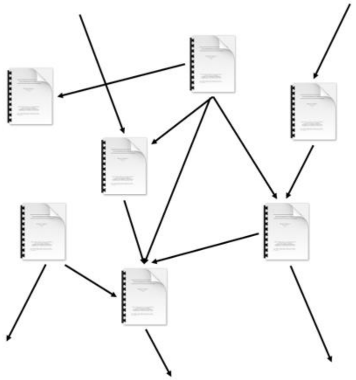


Image credit: [Science]

Economic Networks

Image credit: [Lumen Learning]

Communication Networks



Citation Networks

Image credit: [Missoula Current News]

Internet

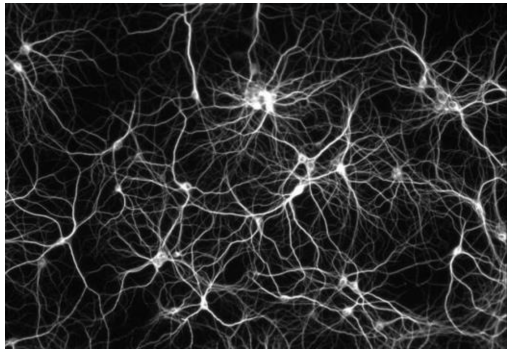


Image credit: [The Conversation]

Networks of Neurons

Many Types of Data are Graphs (3)

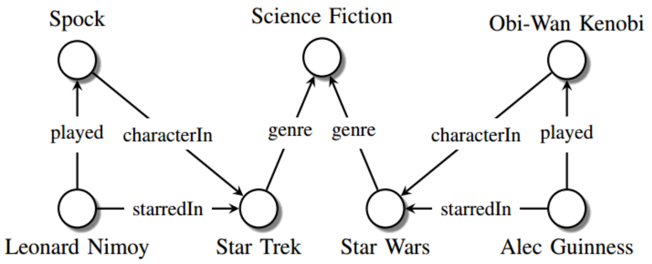


Image credit: Maximilian Nickel et al

Knowledge Graphs

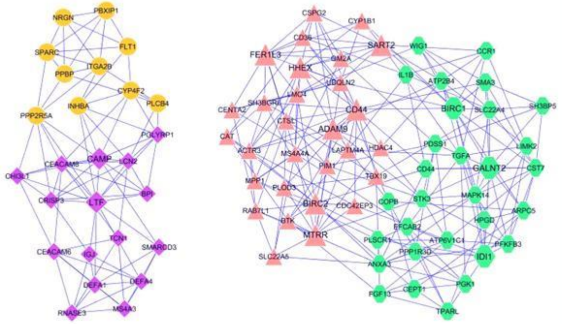


Image credit: ese.wustl.edu

Regulatory Networks

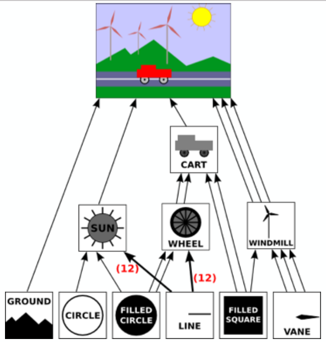


Image credit: math.hws.edu

Scene Graphs

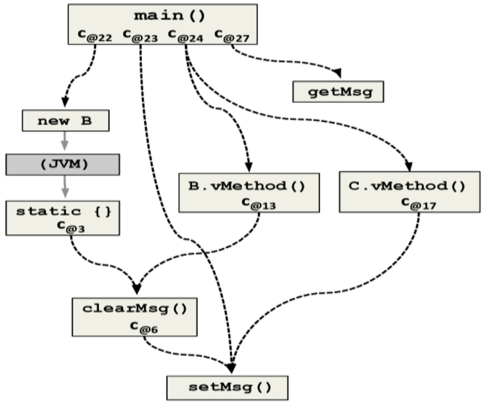


Image credit: ResearchGate

Code Graphs

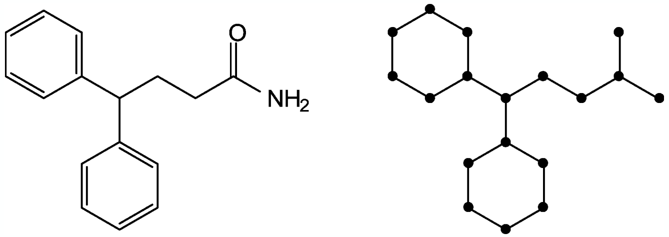


Image credit: MDPI

Molecules

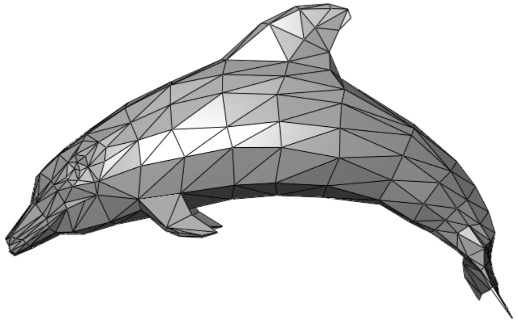


Image credit: Wikipedia

3D Shapes

Graphs and Relational Data

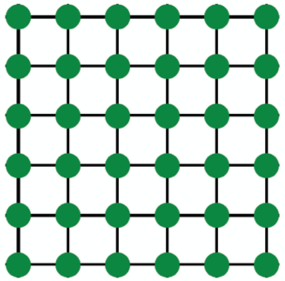
- **Main Question:**

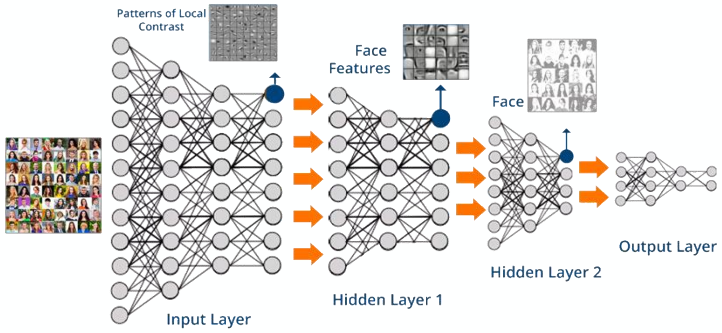
How do we take advantage of relational structure for better prediction?

Graphs: Machine Learning

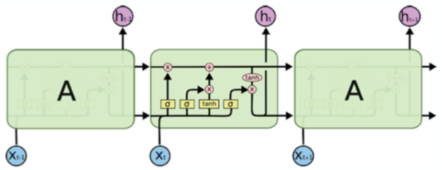
- Complex domains have a rich relational structure, which can be represented as a **relational graph**.
- **By explicitly modeling relationships we achieve better performance!**

Today: Modern ML Toolbox





Images



Text/Speech

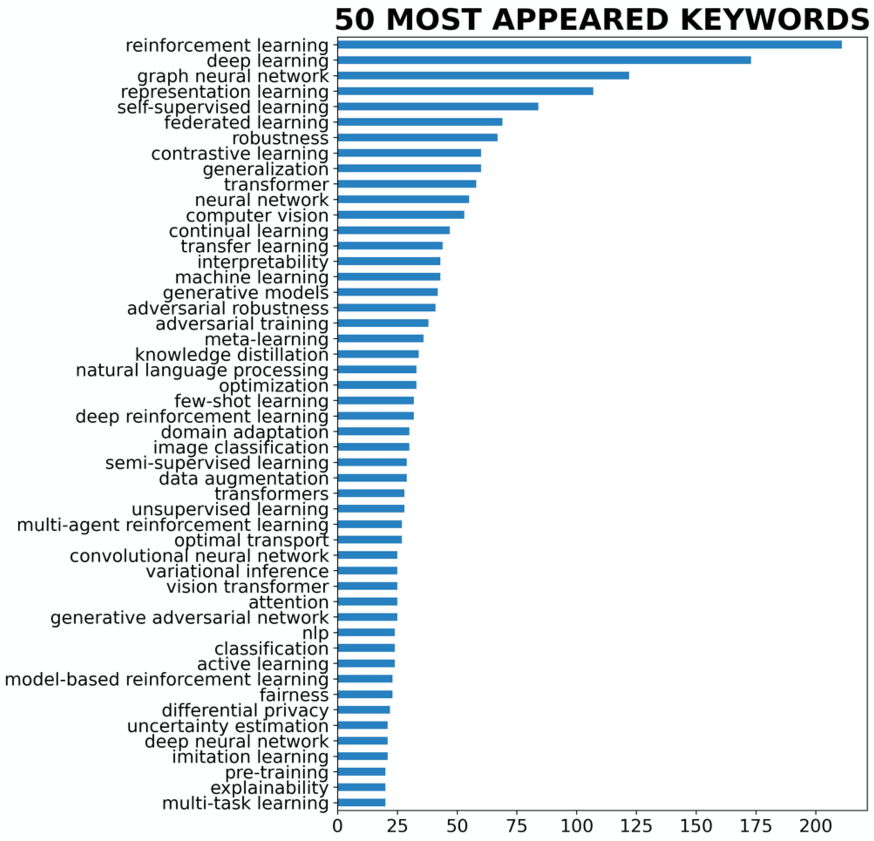
Modern deep learning toolbox is designed for simple sequences & grids.

Today: Modern ML Toolbox

- 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.
- In this course, we consider **graphs** as the new frontier of deep learning. Graphs **connect** things.

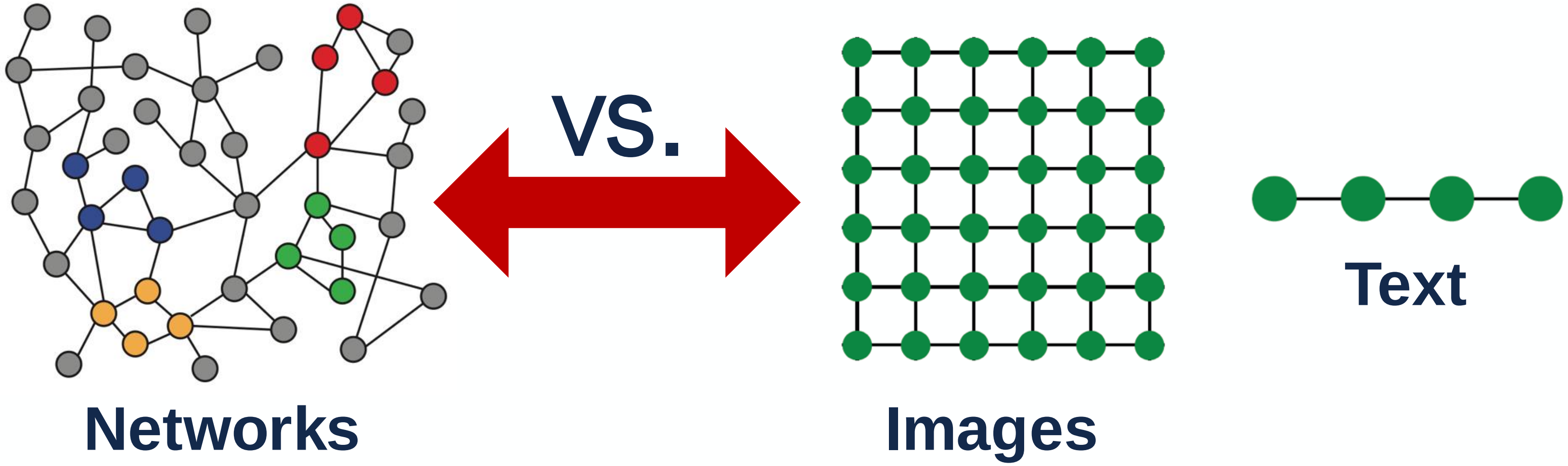
Hot Subfield in ML

ICLR 2022 keywords



Why is Graph Deep Learning Hard?

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



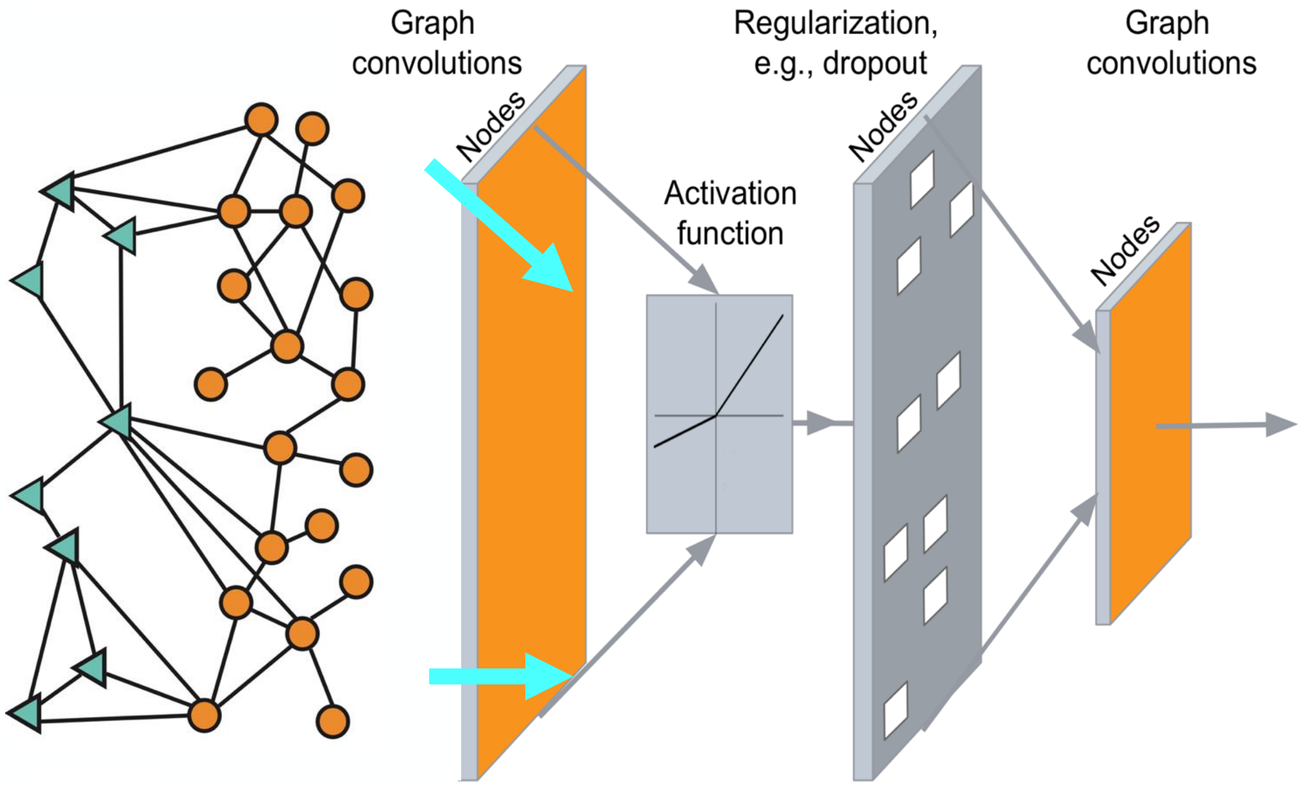
  - No fixed node ordering or reference point
  - Often dynamic and have multimodal features

This Course

- How can we develop neural networks that are much more broadly applicable?
- **Graphs** are the new frontier of deep learning.

Deep Learning with Graphs

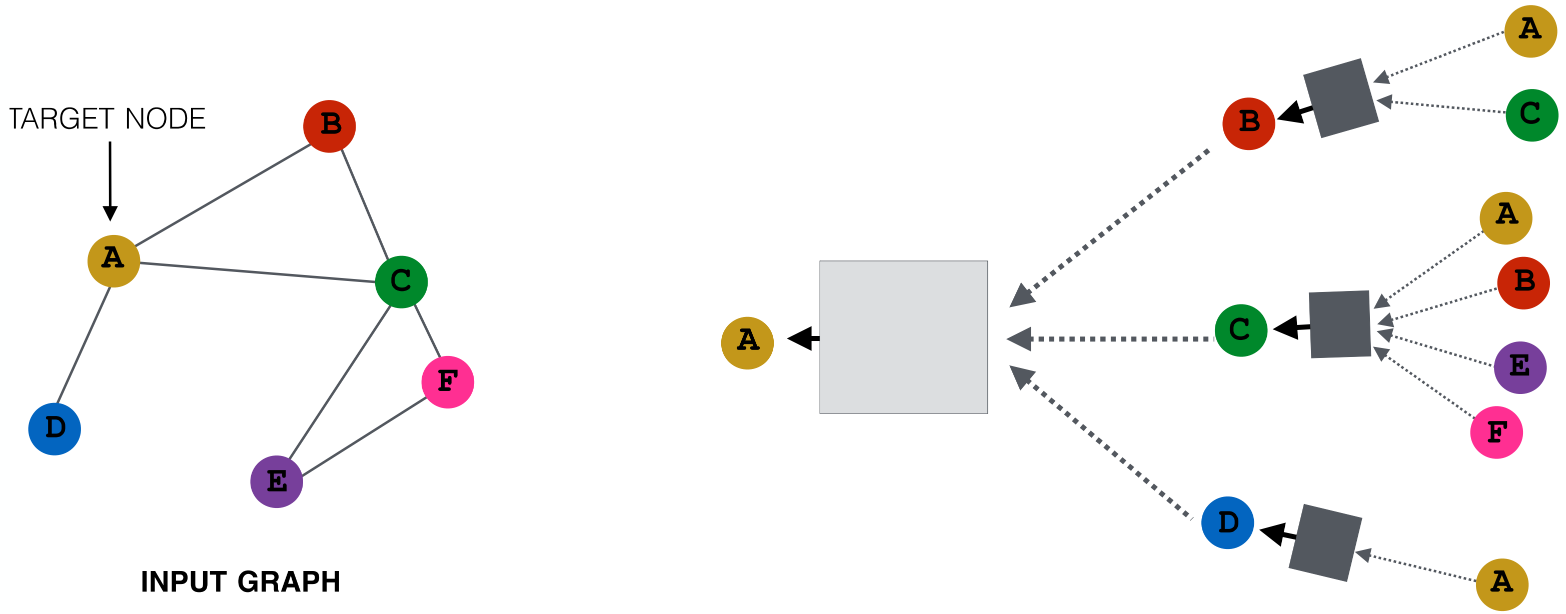
Input:
Network



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

Graph Neural Networks

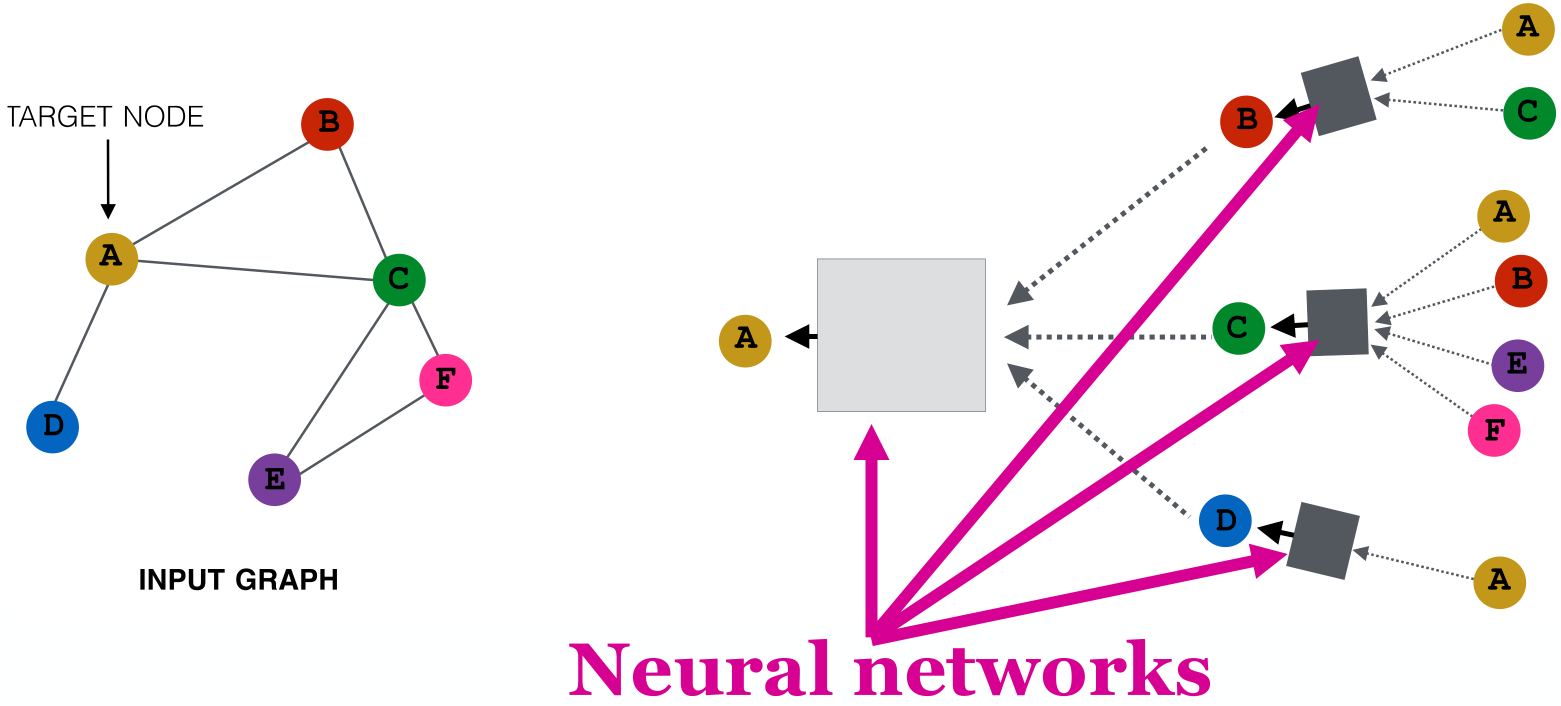
- Each node defines a computation graph.
  - Each edge in this graph is a transformation/aggregation function.



Scarselli et al. 2005. The Graph Neural Network Model. *IEEE Transactions on Neural Networks*.

Graph Neural Networks

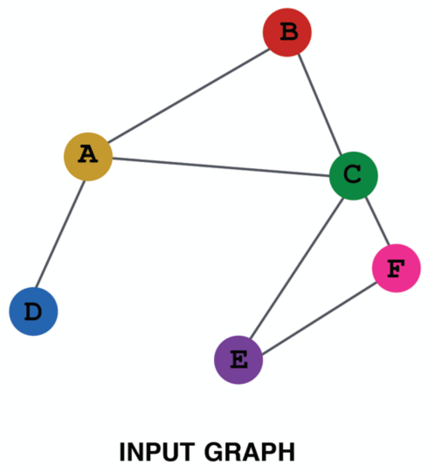
- Intuition: Nodes aggregate information from their neighbors using neural networks.



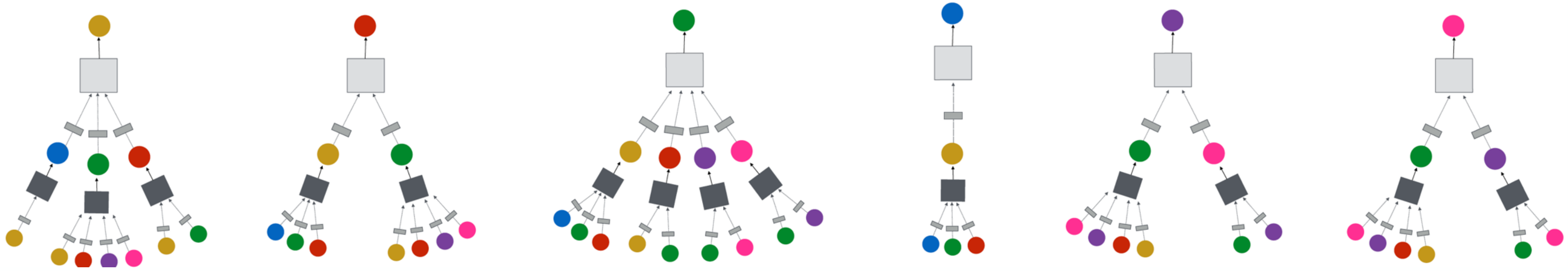
Inductive Representation Learning on Large Graphs. W. Hamilton, R. Ying, J. Leskovec. NIPS, 2017.

Idea: Aggregate Neighbors

- Intuition: Network neighborhood defines a computation graph.

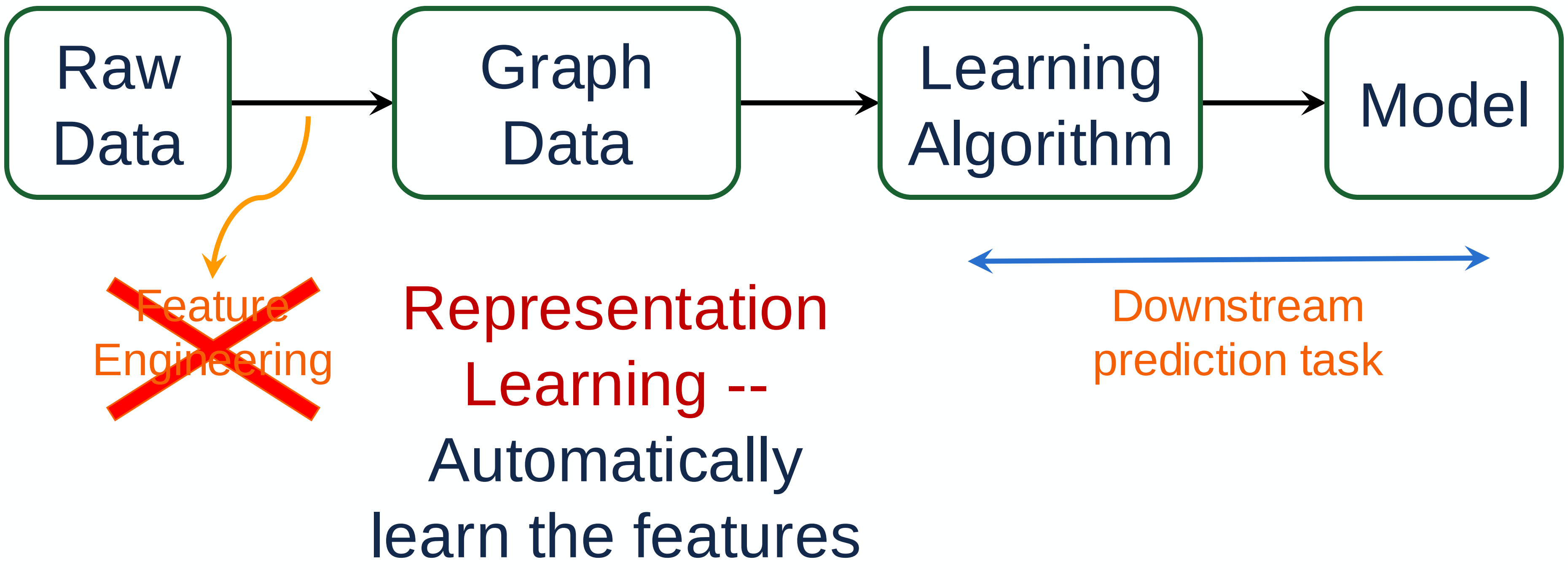


Every node defines a computation graph based on its neighborhood!



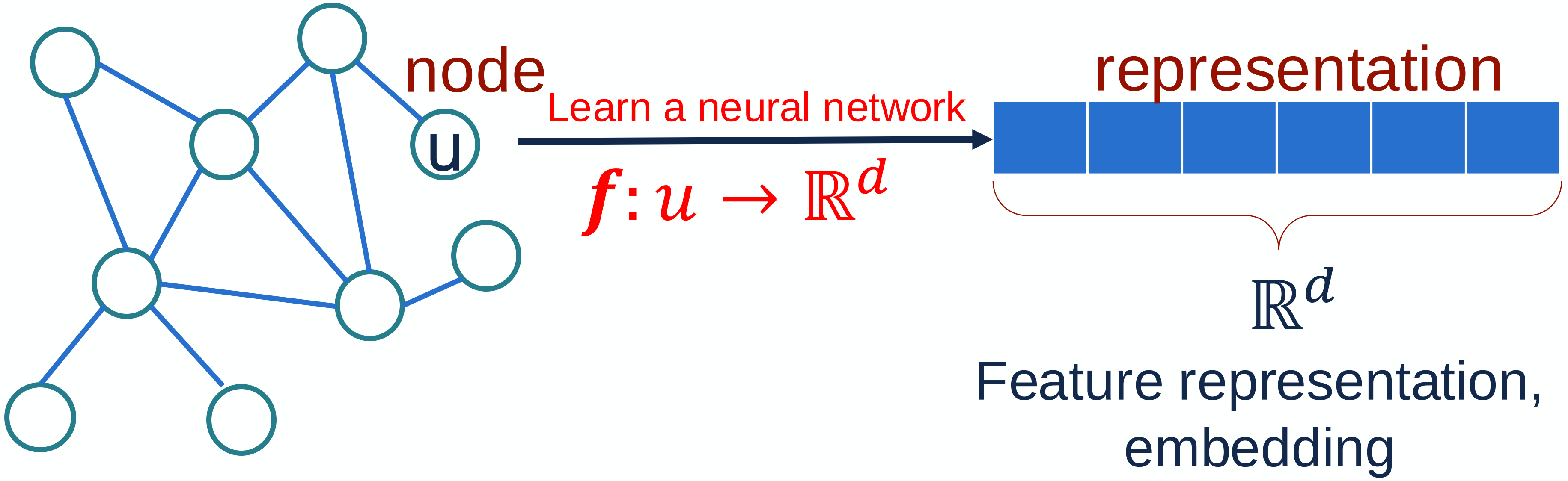
Representation Learning

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



Representation Learning

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



Course Outline

- We are going to explore Machine Learning and Representation Learning for graph 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, Technology

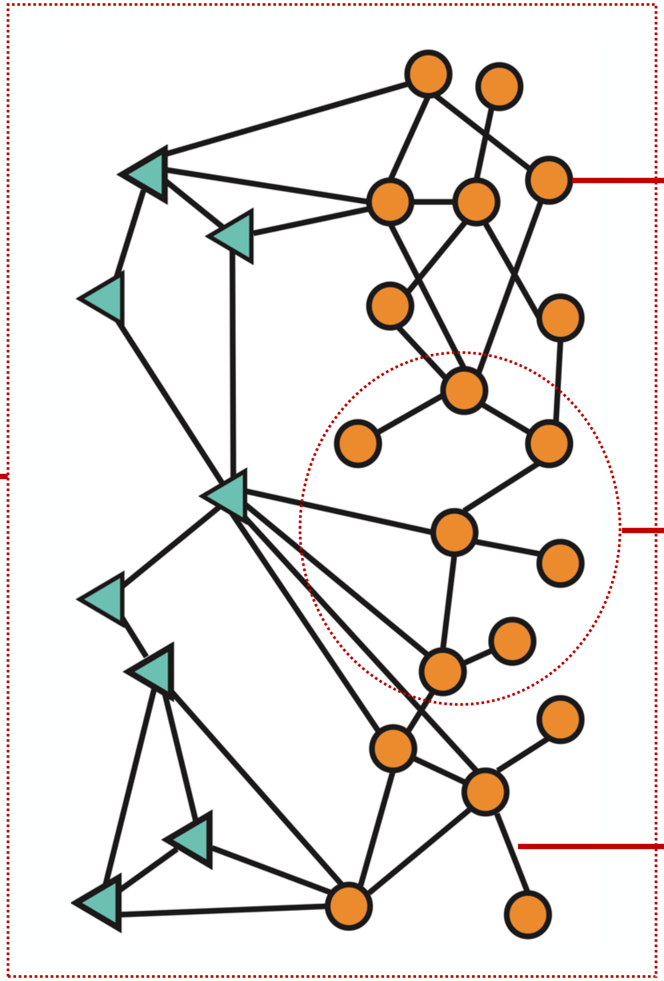
2

Applications of Graph ML

Different Types of Tasks

Graph-level prediction,

Graph generation



Node level

Community (subgraph) level

Edge-level

Classic Graph ML Tasks

- **Node classification**: Predict a property of a node
  - Example: Cat
- **Link predicti** een two nodes
  - Example: Kno
- **Graph classif**
  - Example: Mol
- **Clustering**: De
  - Example: Soc
- Other tasks:
  - **Graph generation**: Drug discovery
  - **Graph evolution**: Physical simulation

These Graph ML tasks lead to high-impact applications!

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

2

Applications of Graph ML

Example of Node-level ML Tasks

Example (1): Protein Folding

- A protein chain acquires its native 3D structure.

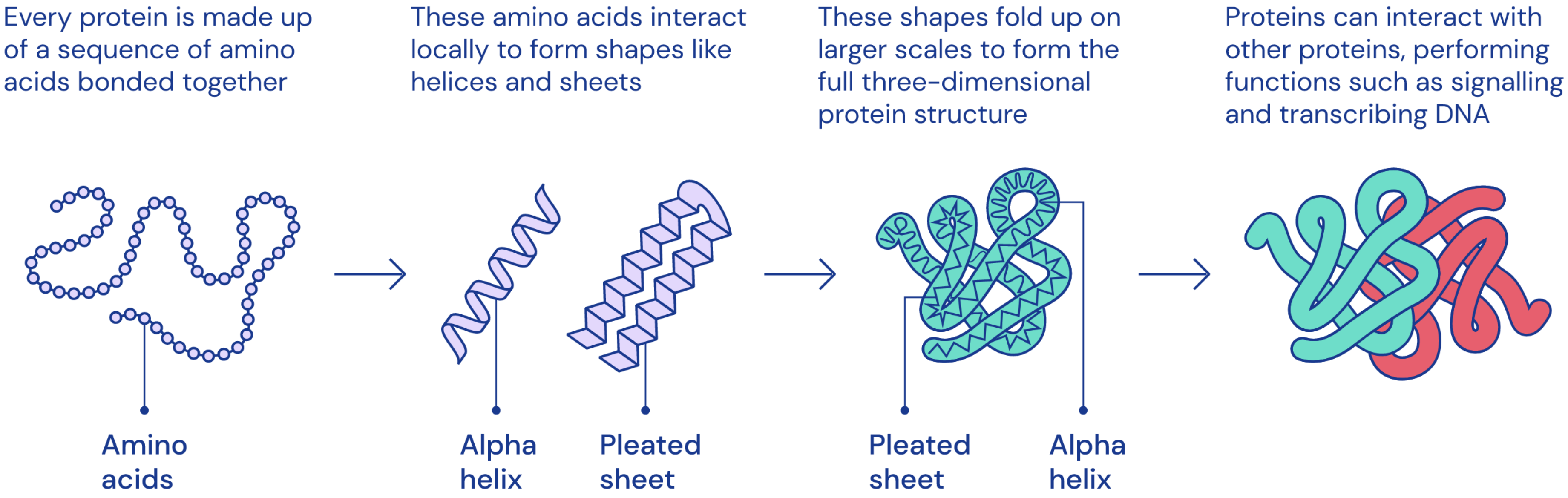


Image credit: DeepMind

Example (1): Protein Folding

- The Protein Folding Problem: Computationally predict a protein's 3D structure based solely on its amino acid sequence.

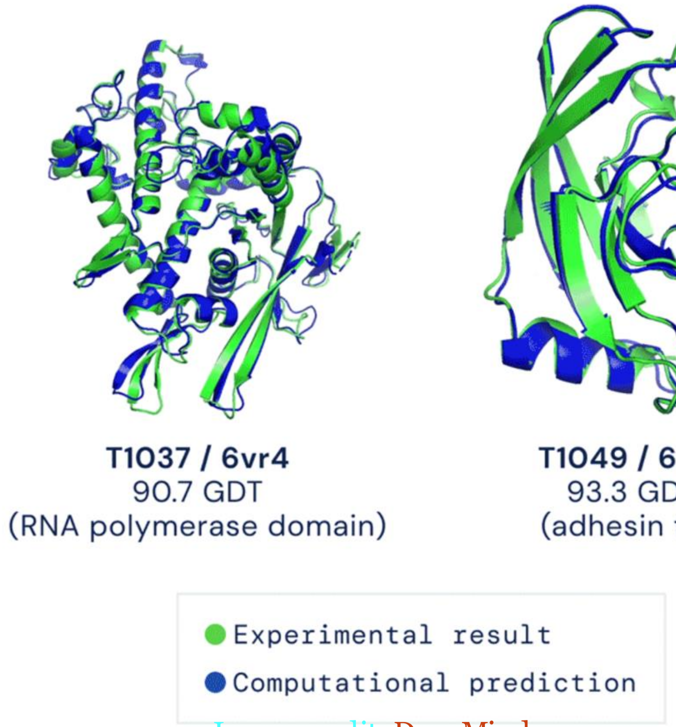


Image credit: DeepMind

AlphaFold: Impact

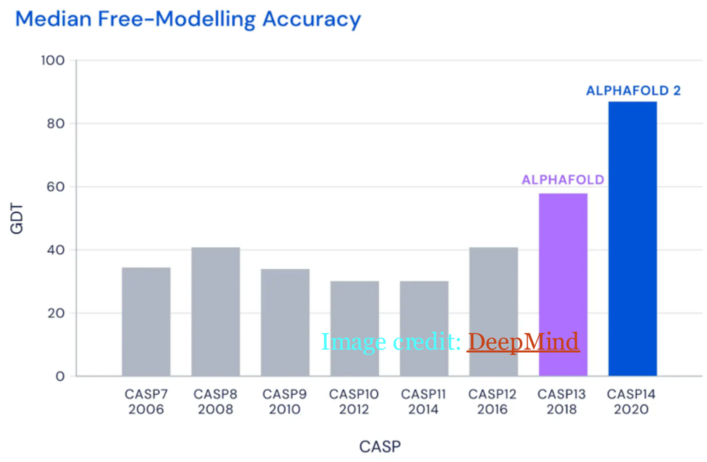


Image credit: DeepMind

Image credit: SingularityHub

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

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

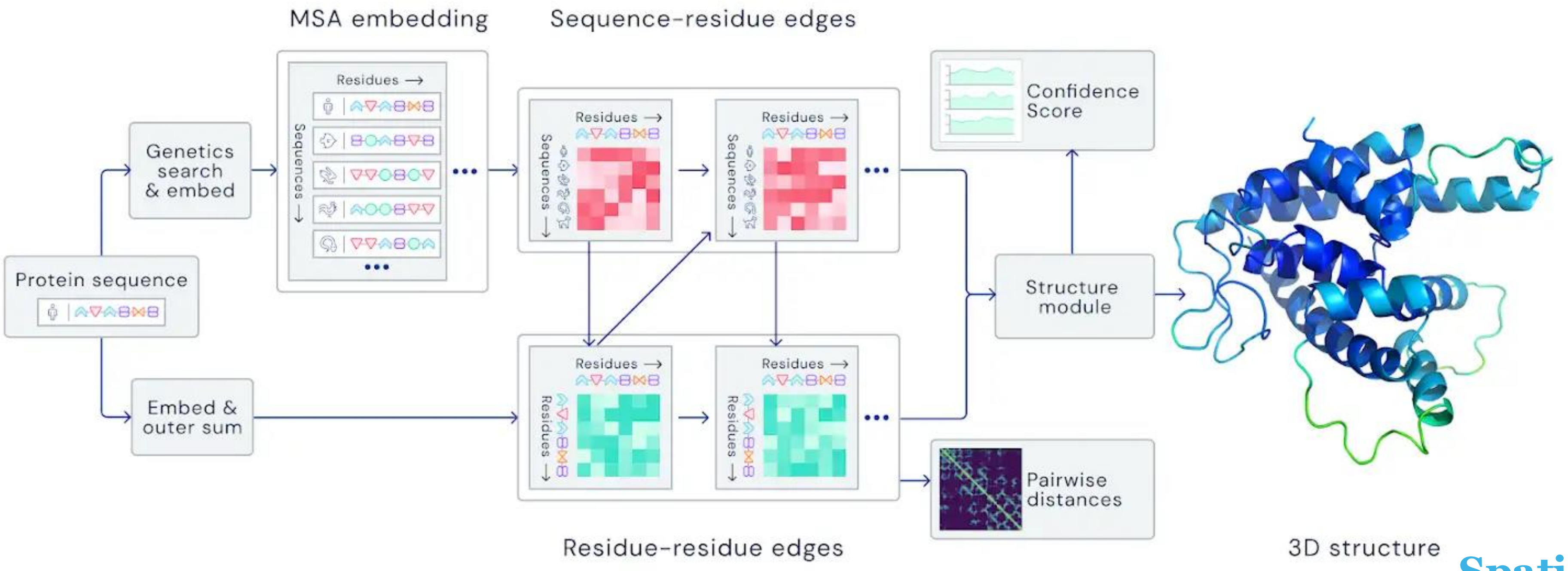
12-14-20

DeepMind's latest AI breakthrough could turbocharge drug discovery

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

AlphaFold: Solving Protein Folding

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



Spatial graph

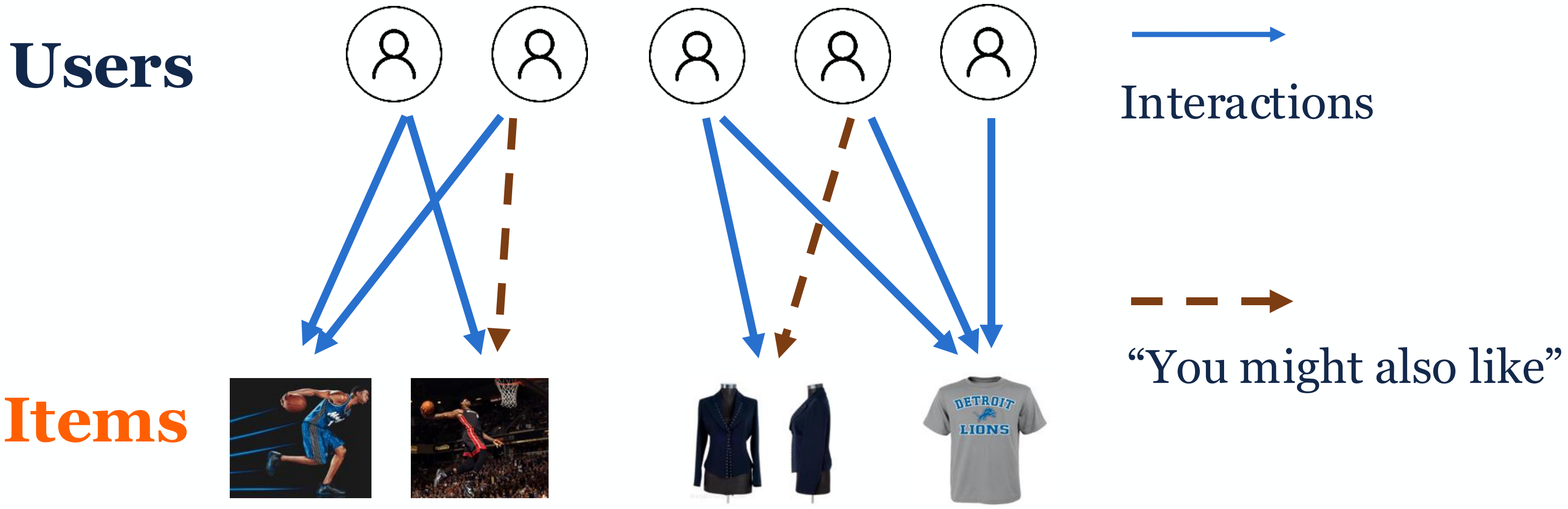
2

Applications of Graph ML

Example of Edge-level ML Tasks

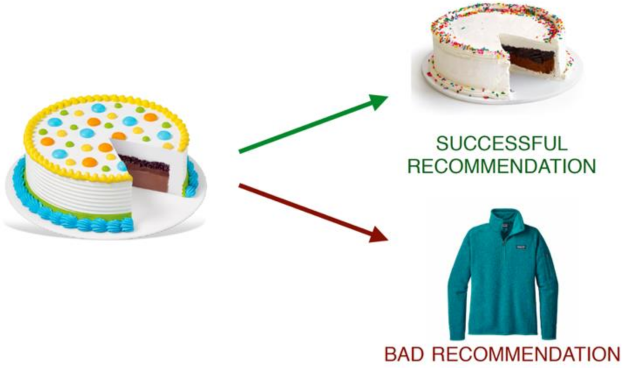
Example (2): Recommender Systems

- **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**



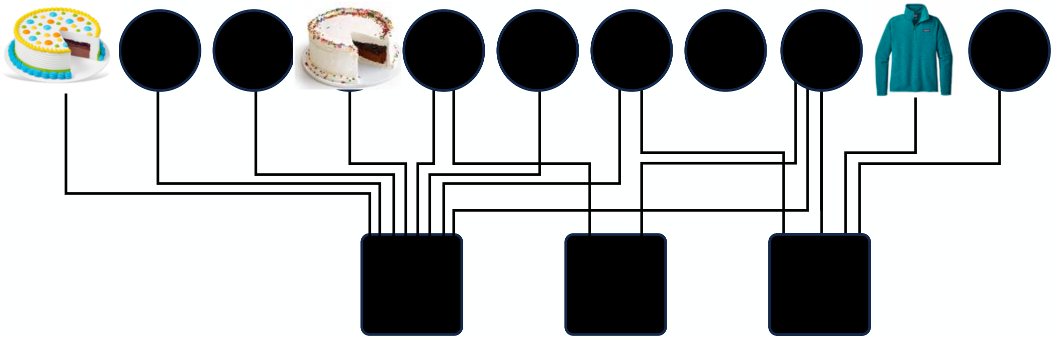
PinSage: Graph-based Recommender

- Task: Recommend related pins to users



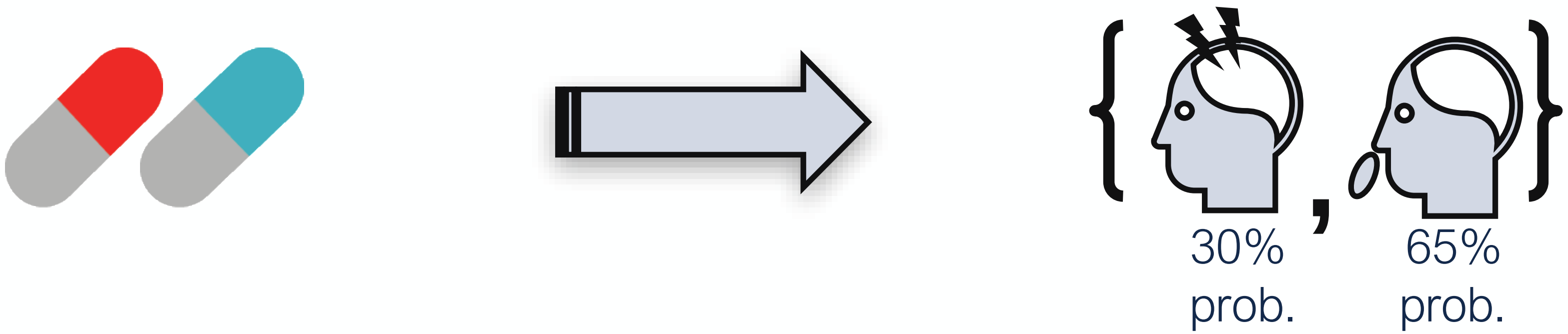
Task: Learn node embeddings z_i such that $d(z_{cake1}, z_{cake2}) < d(z_{cake1}, z_{sweater})$

Predict whether two nodes in a graph are related



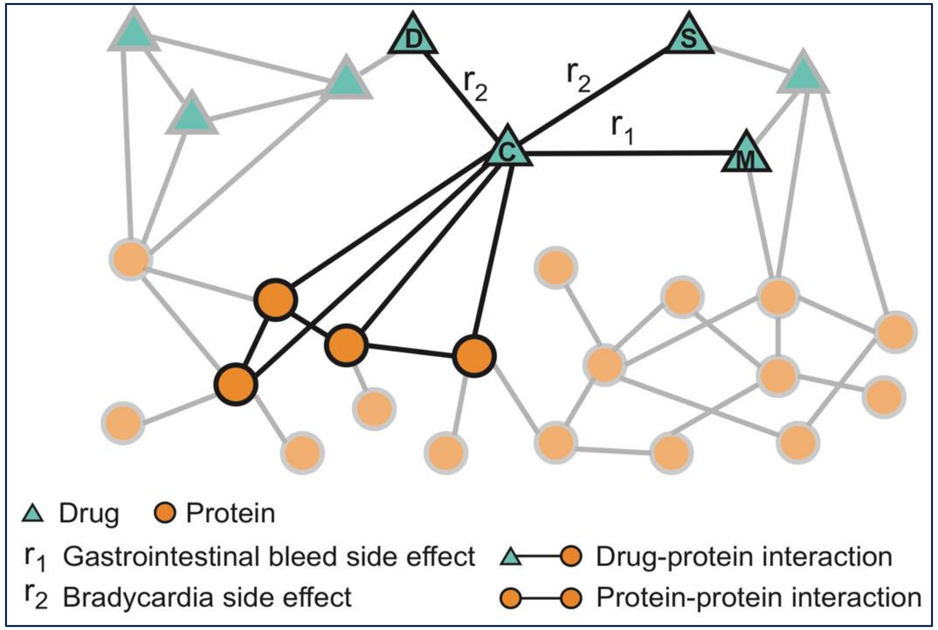
Example (3): Drug Side Effects

- 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.**

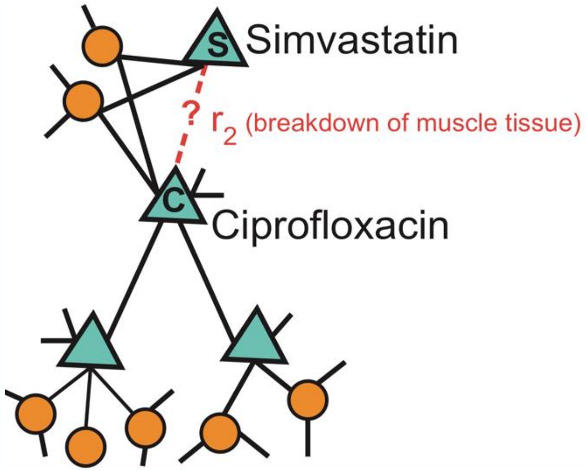


Biomedical Graph Link Prediction

- **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

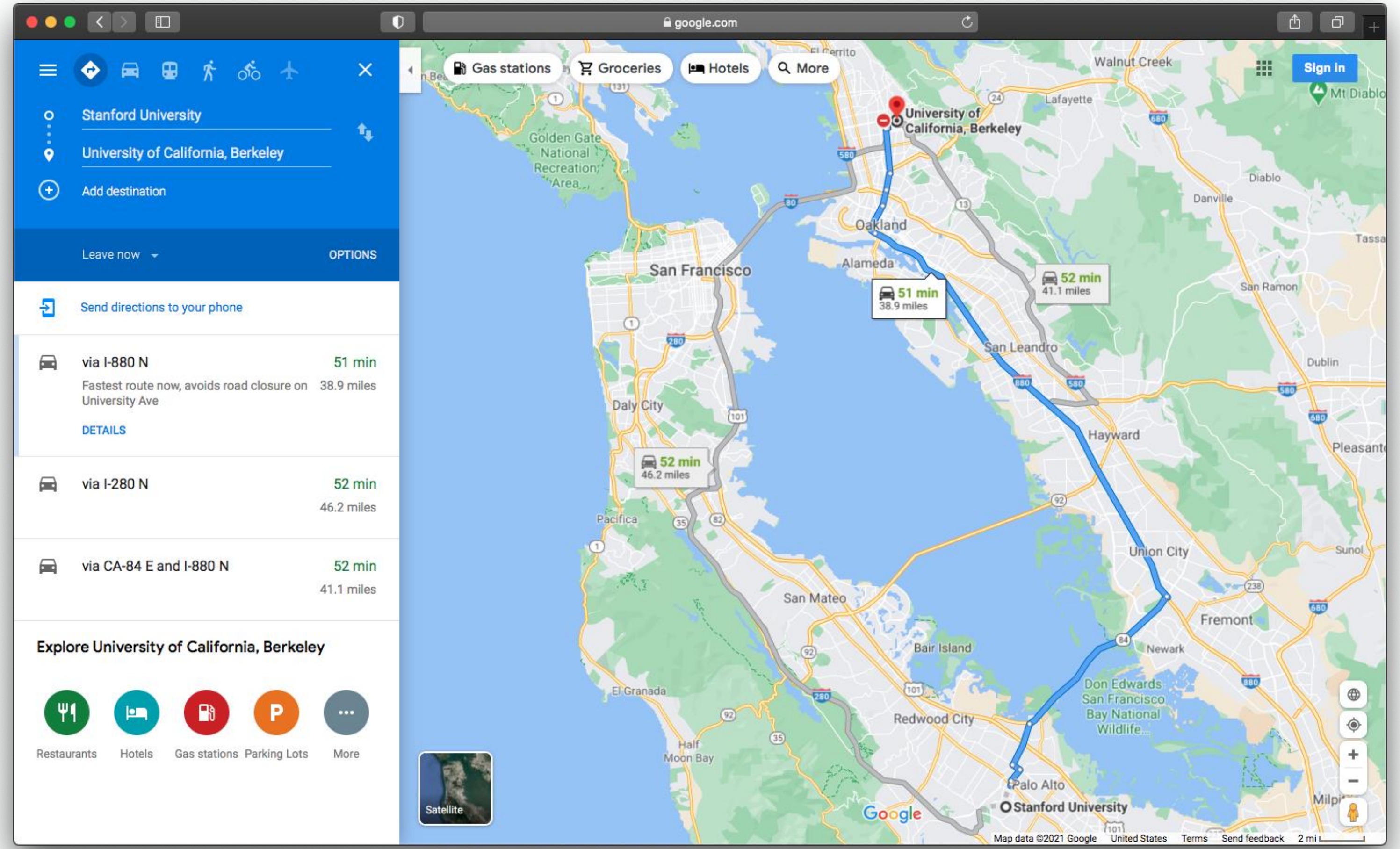
Severe Rhabdomyolysis due to Presumed Drug Interactions between Atorvastatin with Amlodipine and Ticagrelor

2

Applications of Graph ML

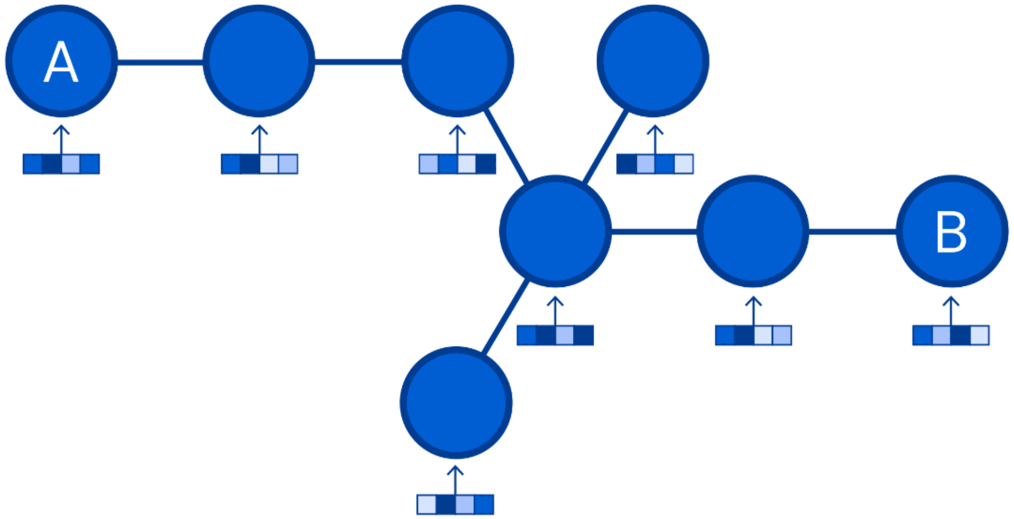
Example of Subgraph-level ML Tasks

Example (4): Traffic Prediction



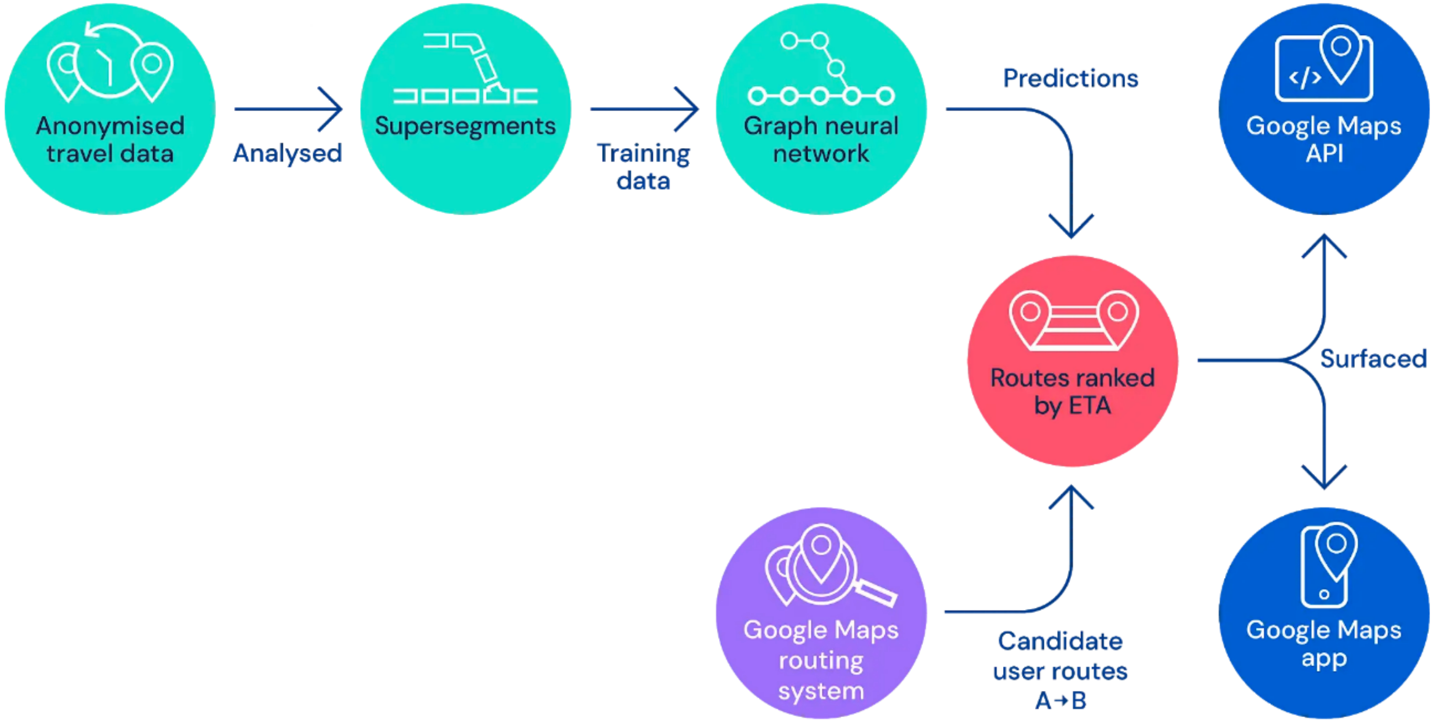
Road Network as a Graph

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



Traffic Prediction via GNN

- Predicting Time of Arrival with Graph Neural Networks



THE MODEL ARCHITECTURE FOR DETERMINING OPTIMAL ROUTES AND THEIR TRAVEL TIME.

Used in Google Maps

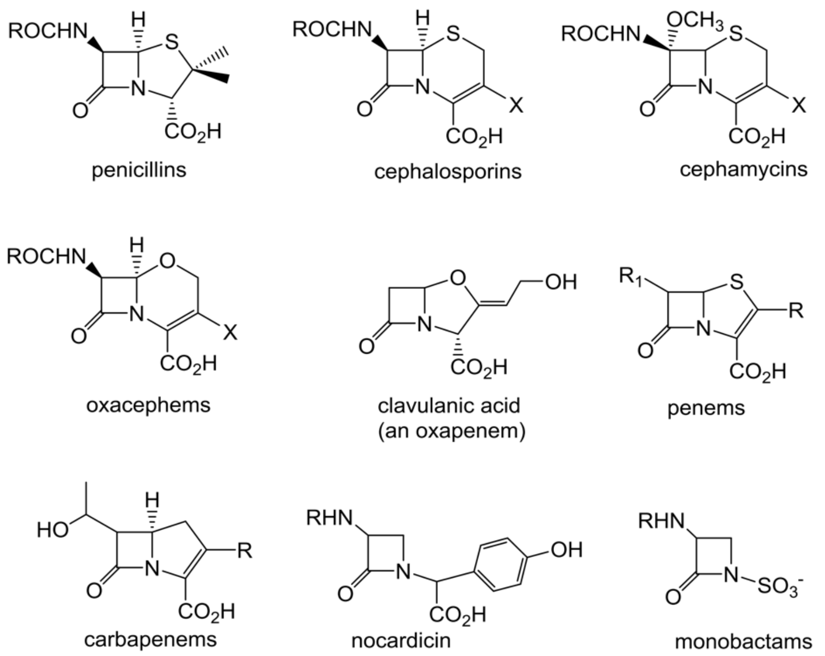
2

Applications of Graph ML

Example of Graph-level ML Tasks

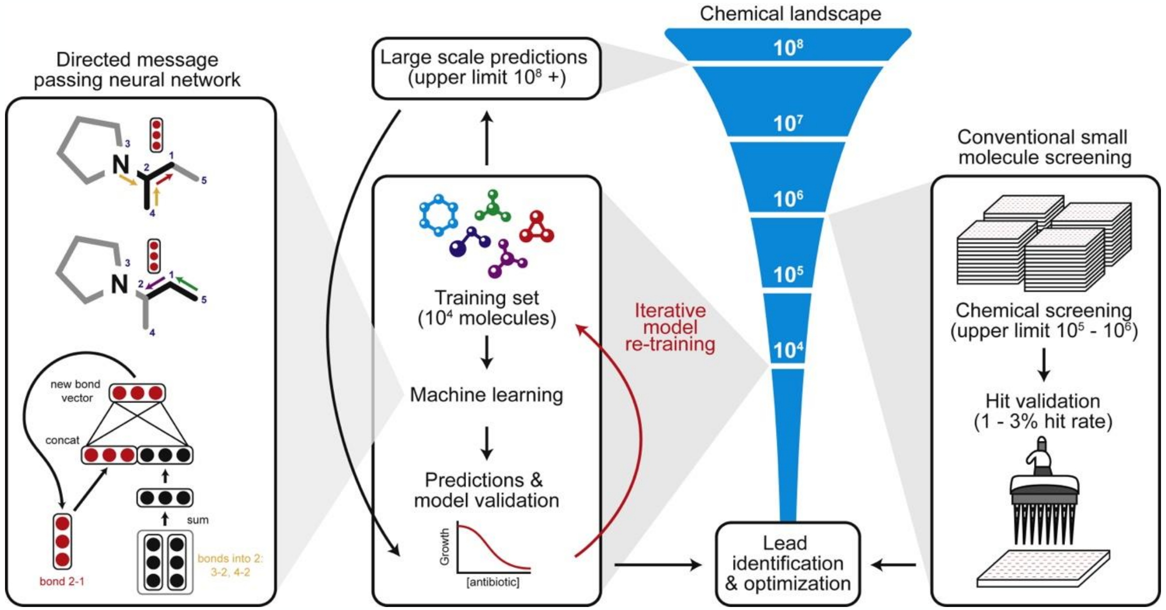
Example (5): Drug Discovery

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



Deep Learning for Antibiotic Discovery

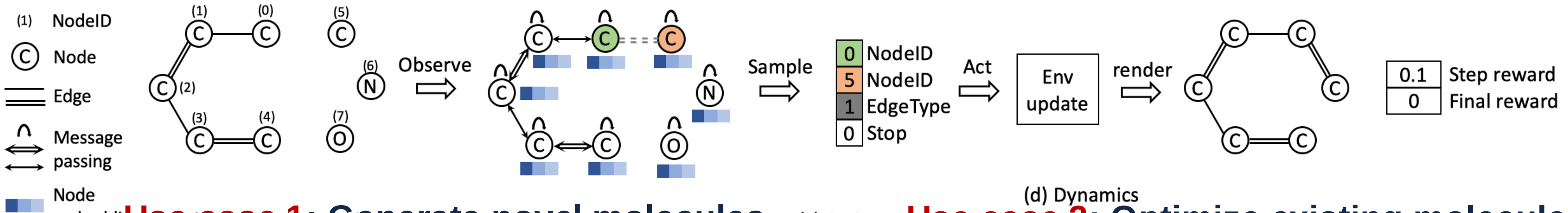
- A Graph Neural Network **graph classification model**
- Predict promising molecules from a pool of candidates



Stokes et al., A Deep Learning Approach to Antibiotic Discovery, Cell 2020

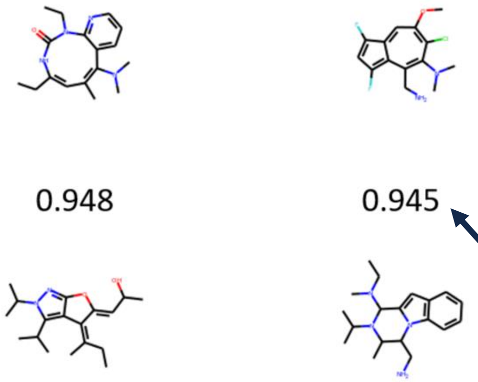
Molecule Generation / Optimization

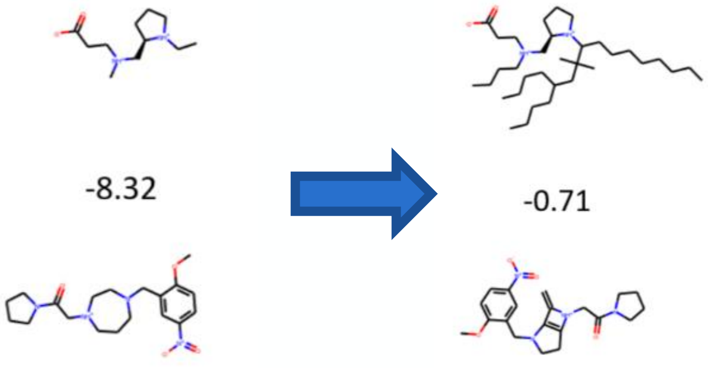
- **Graph generation**: Generating novel molecules



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

Use case 2: Optimize existing molecules to have desirable properties



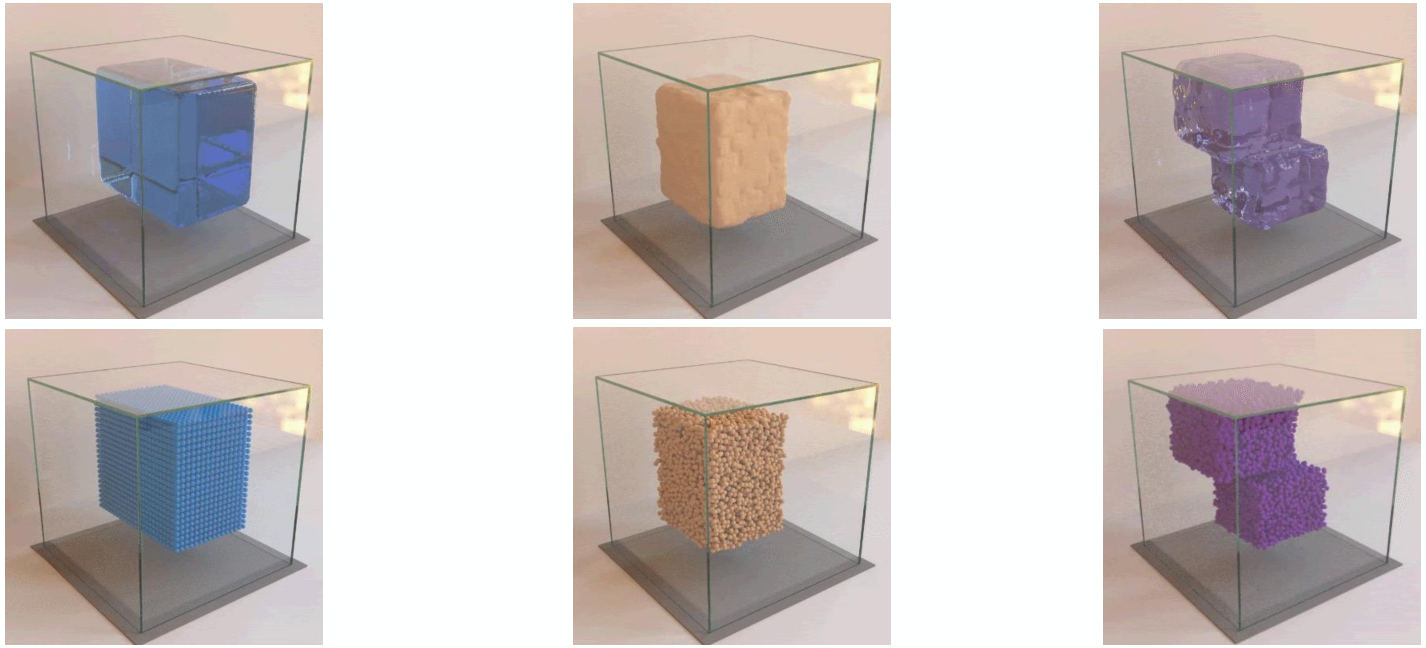


Drug likeness

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

Example (6): Physics Simulation

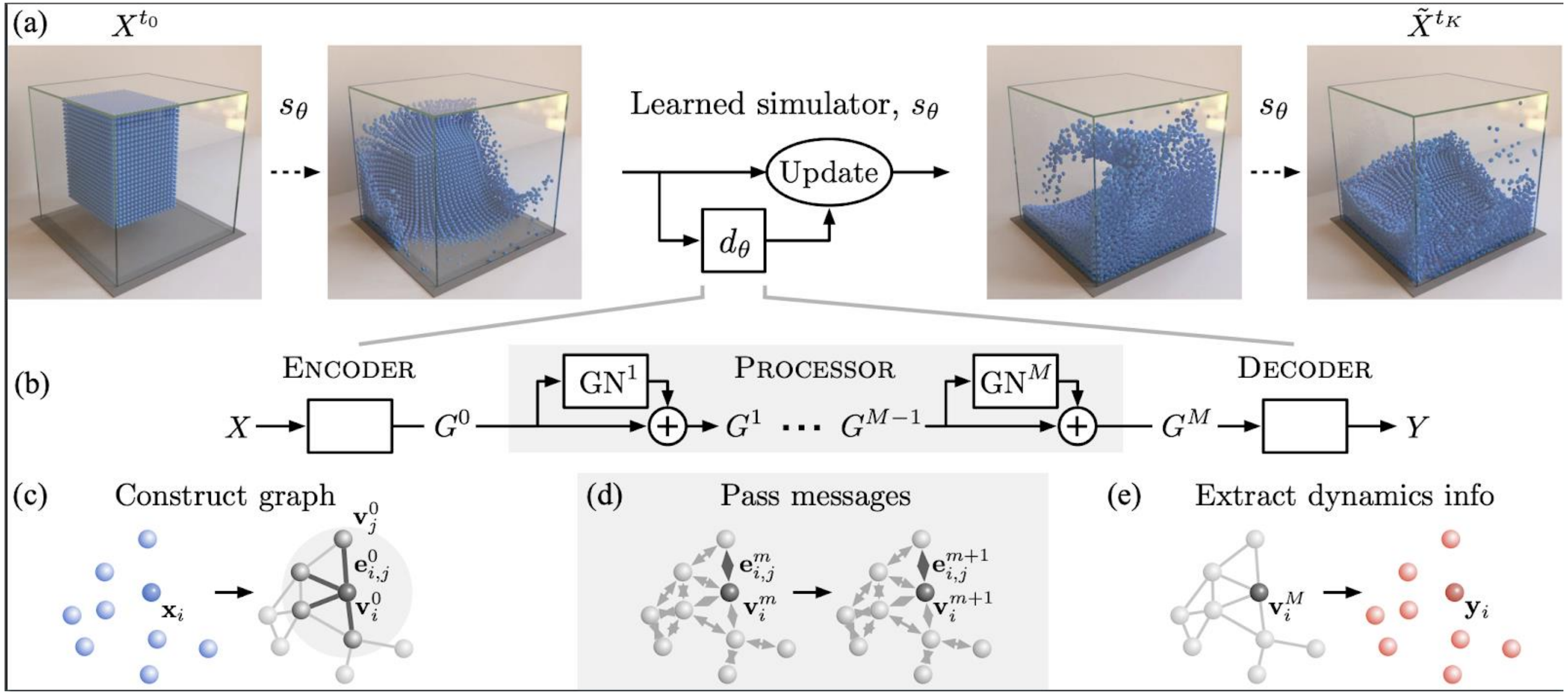
- Physical simulation as a graph:
  - **Nodes**: Particles
  - **Edges**: Interaction between particles



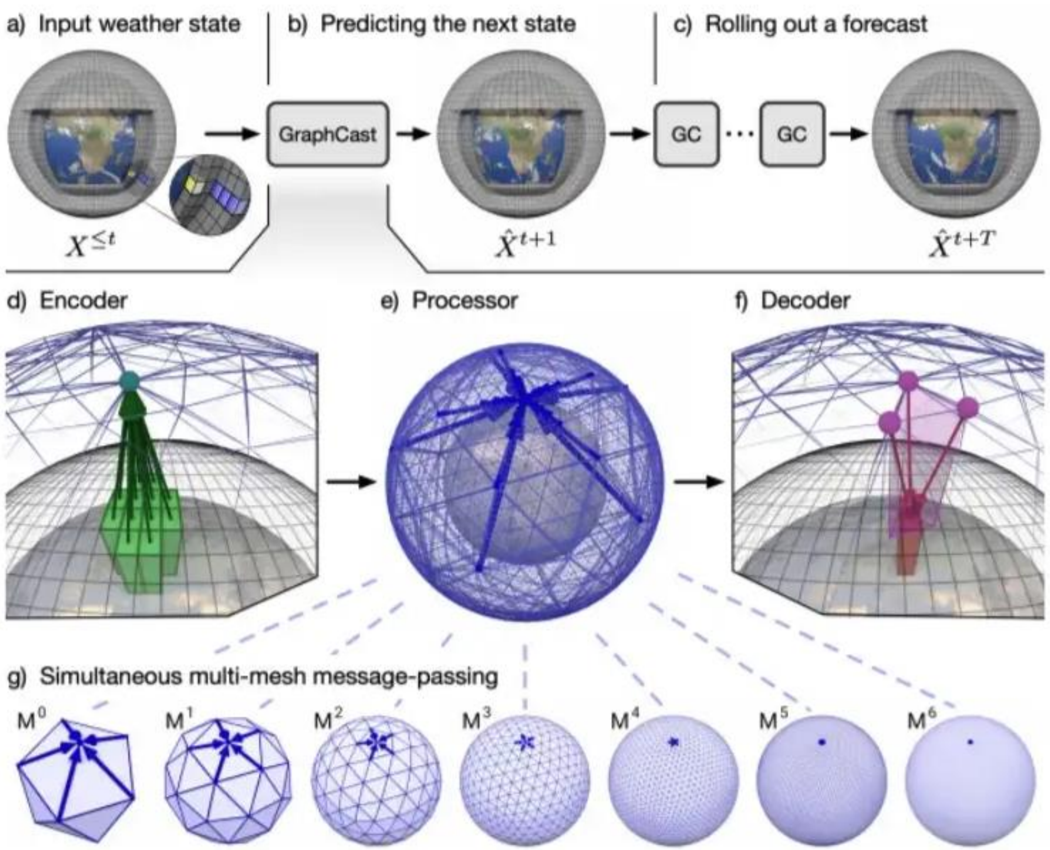
Sanchez-Gonzalez et al., Learning to simulate complex physics with graph networks, ICML 2020

Simulation Learning Framework

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



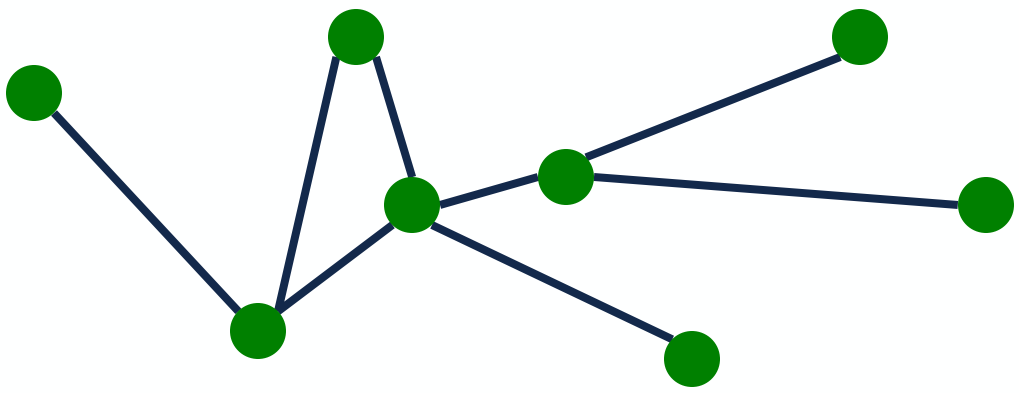
Application: DeepMind weather forecasting



3

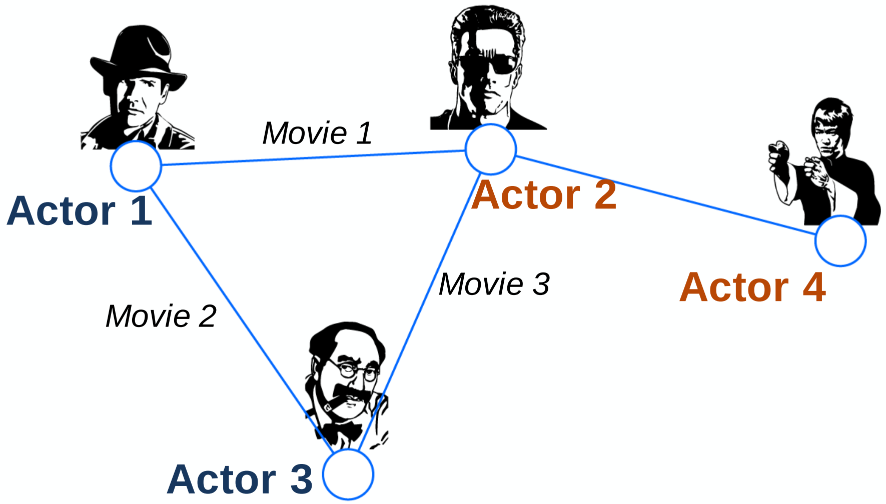
Choices of Graph Representation

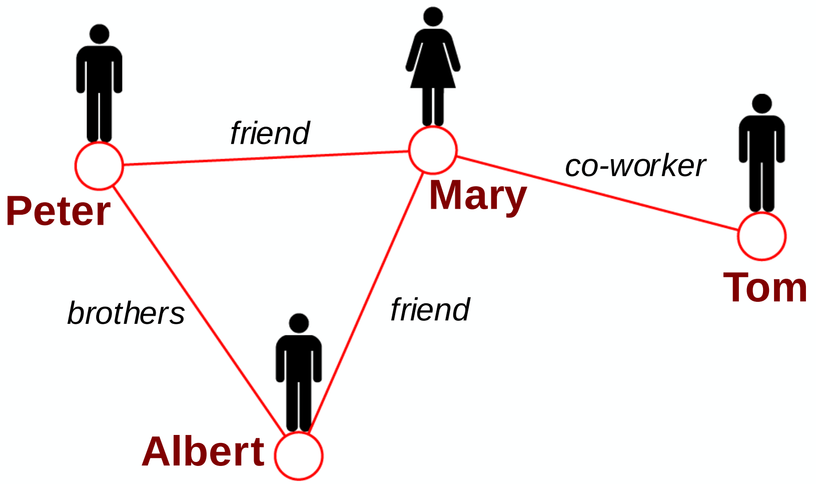
Components of a Network

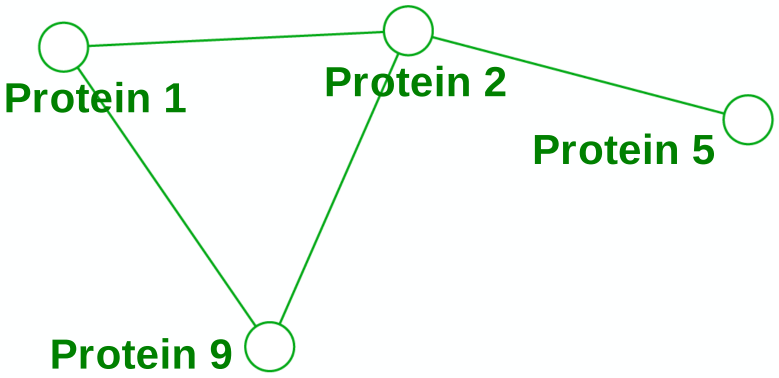


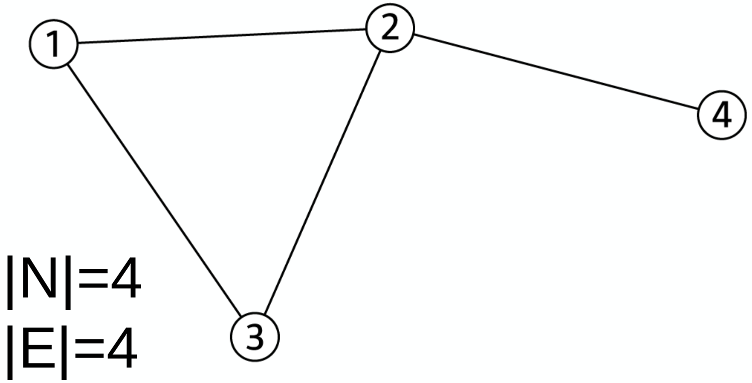
- **Objects**: nodes, vertices — N
- **Interactions**: links, edges — E
- **System**: network, graph — $G(N,E)$

Graphs: A Common Language









Choosing a Proper Representation

- If you connect individuals that work with each other, you will explore a **professional network**.
- If you connect those that have a sexual relationship, you will be exploring **sexual networks**.
- 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.

Image credit: Euro Scientists

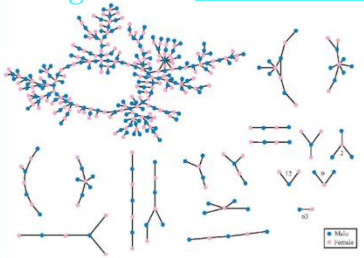
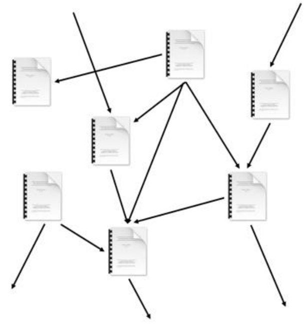


Image credit: ResearchGate

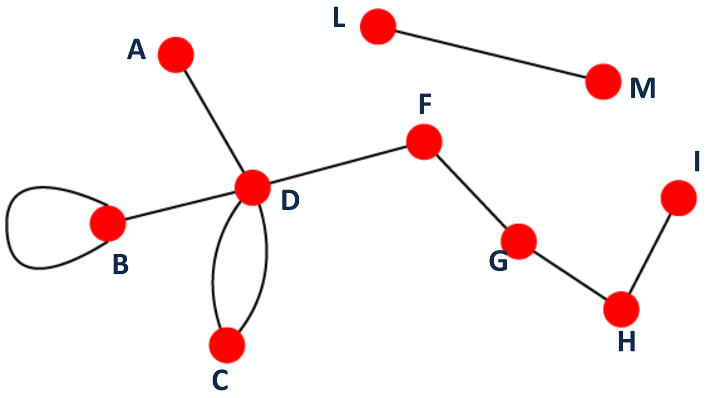


How do you define a graph?

- **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

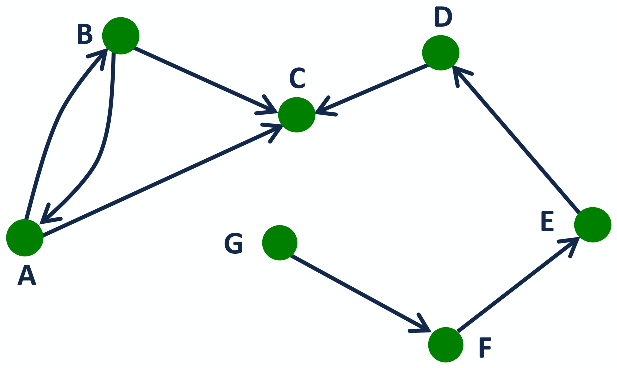
Directed vs. Undirected Graphs

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



  - Examples:
    - Collaborations
    - Friendship on Facebook

- **Directed**
  - Links: directed (arcs)



  - Examples:
    - Phone calls
    - Following on Twitter

Heterogeneous Graphs

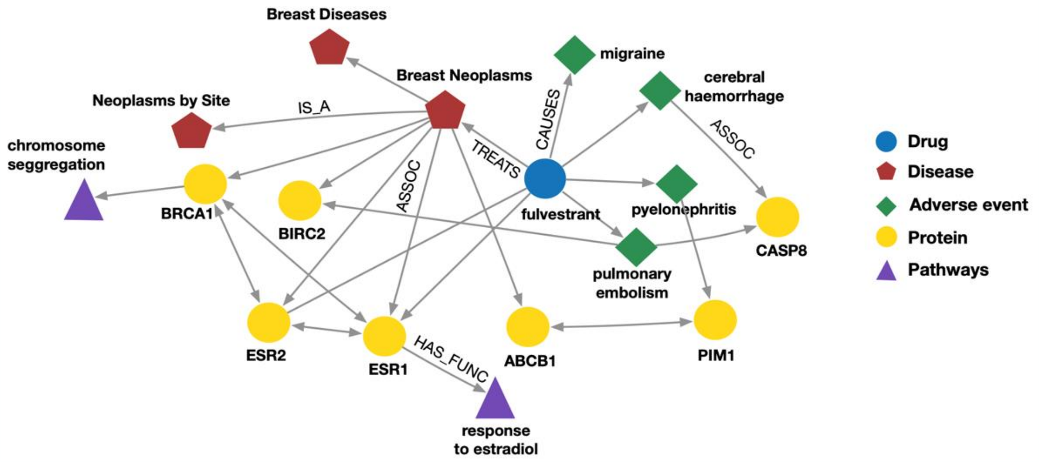
- **A heterogeneous graph is defined as**

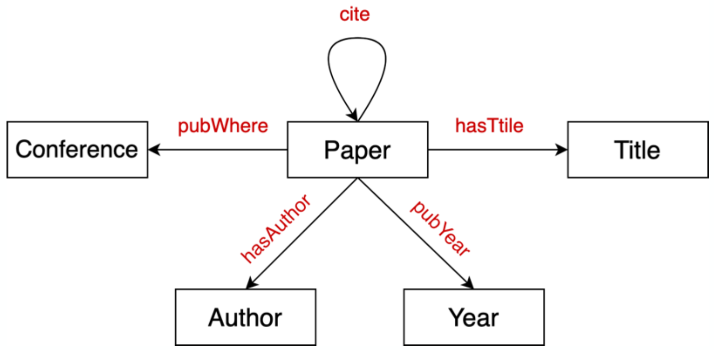
$$\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{R}, \boldsymbol{T})$$

  - 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

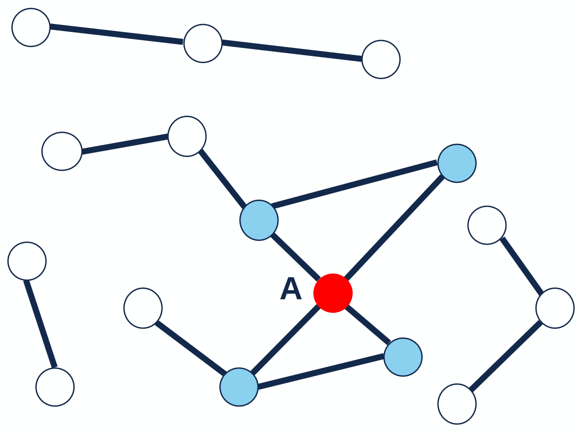
- **Biomedical Knowledge Graphs**
  - **Example node: Migraine**
  - **Example edge: (fulvestrant, Treats, Breast Neoplasms)**
  - **Example node type: Protein**
  - **Example edge type (relation): Causes**
- **Academic Graphs**
  - **Example node: ICML**
  - **Example edge: (GraphSAGE, NeurIPS)**
  - **Example node type: Author**
  - **Example edge type (relation): pubYear**



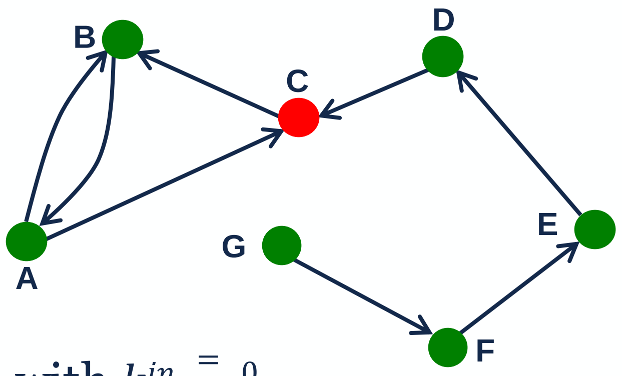


Node Degrees

- **Undirected**
  - **Node degree, k_i:** the number of edges adjacent to node $\boldsymbol{i}$, $\boldsymbol{k_A = 4}$
  - $\boldsymbol{Avg.\ degree:\ \bar{k} = \langle k \rangle = \frac{1}{N}\sum_{i=1}^{N} k_i = \frac{2E}{N}}$



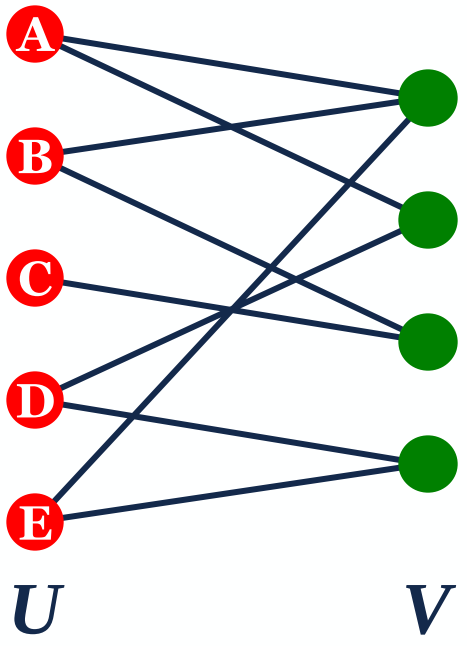
- **Directed**
  - In directed networks we define an **in-degree** and **out-degree**.
    The (total) degree of a node is the sum of in- and out-degrees.
  - $k_C^{in} = 2,\ k_C^{out} = 1,\ k_C = 3$
    $\bar{k} = \frac{E}{N},\ \overline{k^{in}} = \overline{k^{out}}$



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

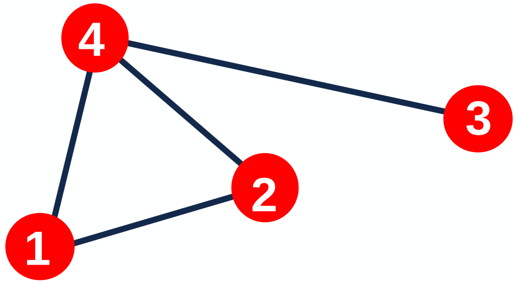
Bipartite Graph

- **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

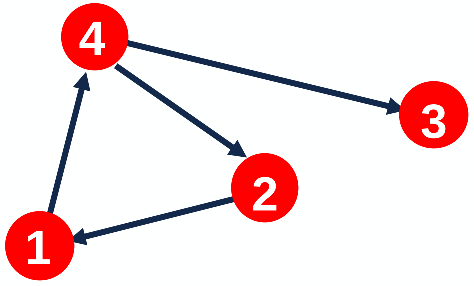


Representing Graphs: Adjacency Matrix

- $A_{ij} = 1$ if there is a link from node i to node j
- $A_{ij} = 0$ otherwise



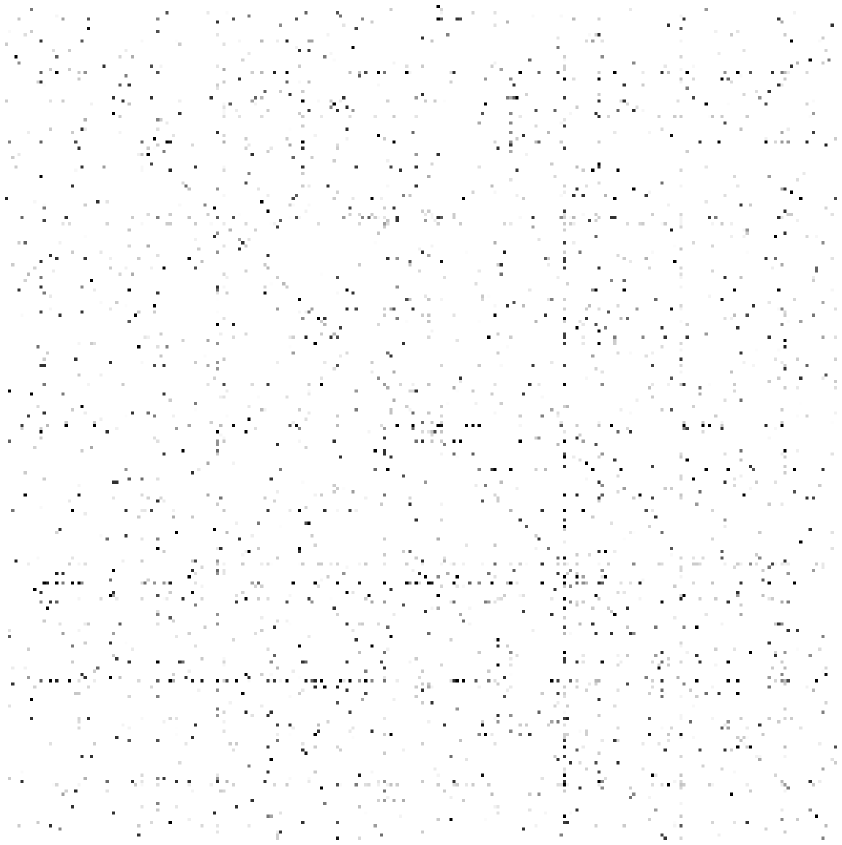
$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$



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

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

Adjacency Matrices are Sparse



Networks are Sparse Graphs

- **Most real-world networks are sparse**
- **$E << E_{max}$ (or $k << N\text{-}1$)**

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})

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

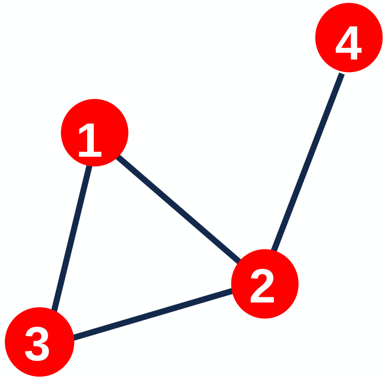
Unweighted (undirected)

- Examples: Friendship, Hyperlink

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

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

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



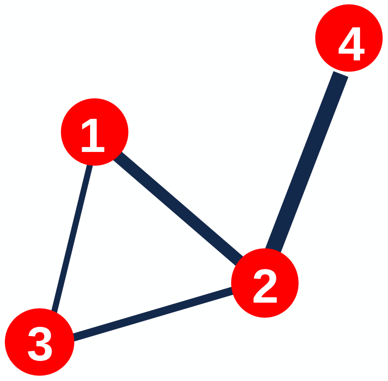
Weighted (undirected)

- Examples: Collaboration, Internet, Roads

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

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

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



More Types of Graphs

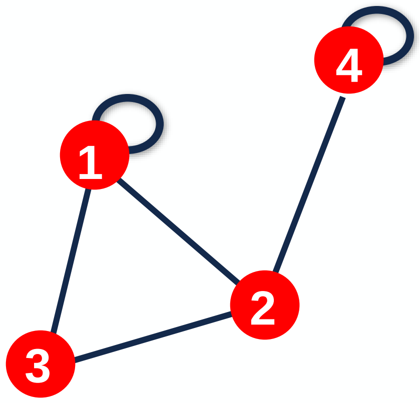
Self-edges (self-loops) (undirected)

- Examples: Proteins, Hyperlinks

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

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

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



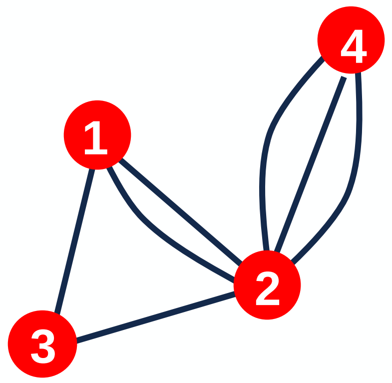
Multigraph (undirected)

- Examples: Communication, Collaboration

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

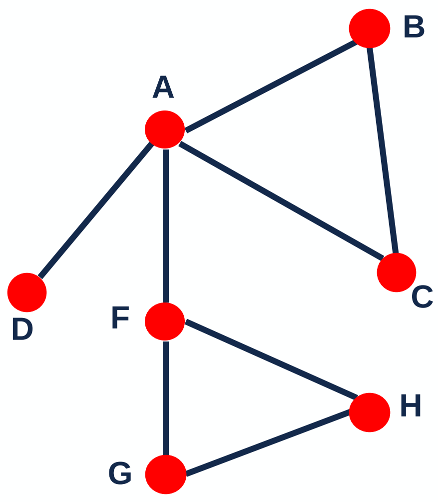
$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

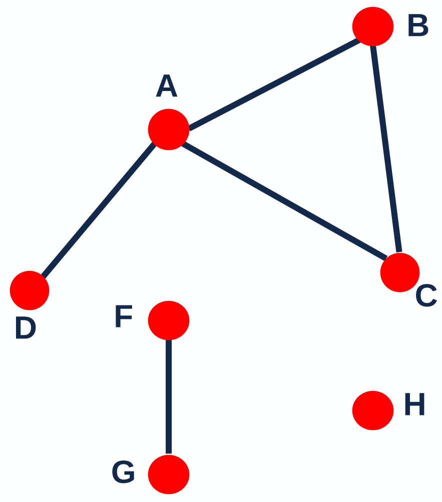
$$E = \frac{1}{2} \sum_{i,j=1}^{N} nonzero(A_{ij}) \qquad \bar{k} = \frac{2E}{N}$$



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



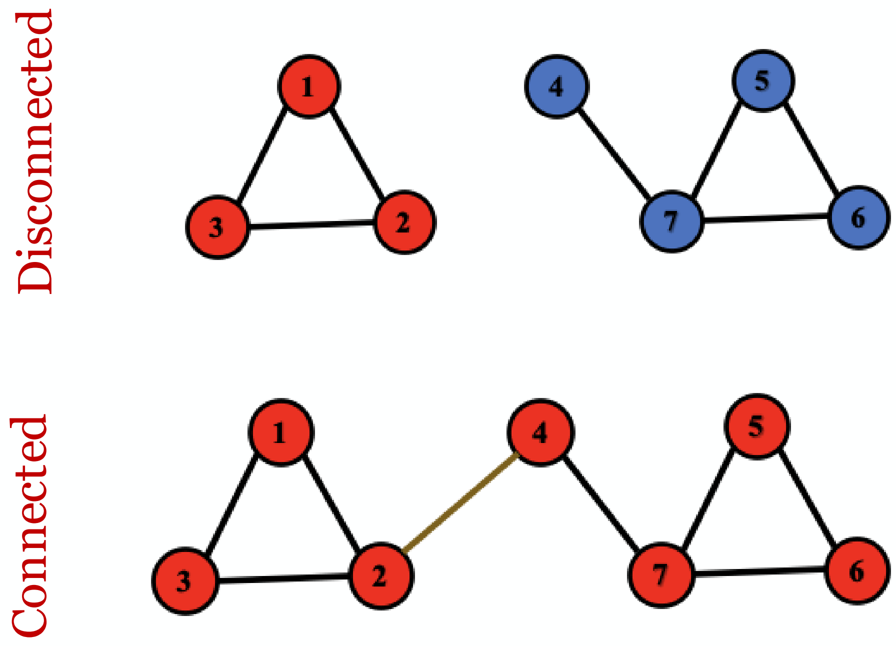


Largest Component:
Giant Component

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:

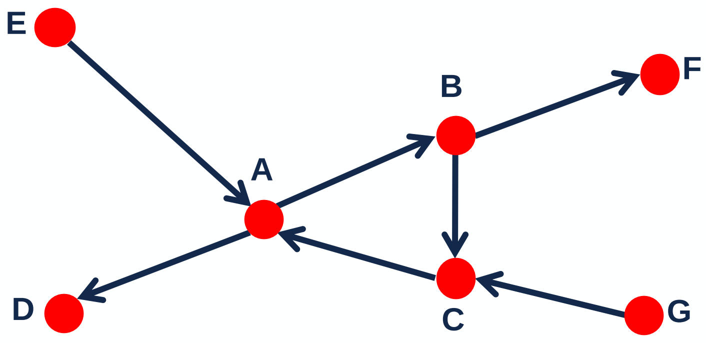


$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

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

Connectivity of Directed Graphs

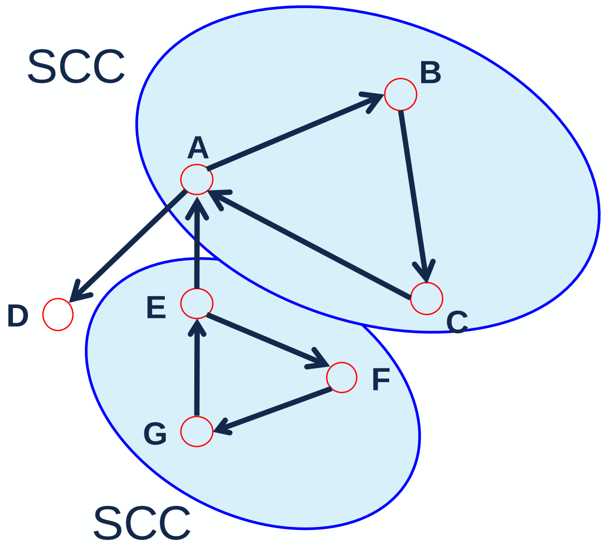
- **Strongly connected directed graph**
  - has a path from each node to every other node and vice versa (e.g., A-B path and B-A path)
- **Weakly connected directed graph**
  - is connected if we disregard the edge directions

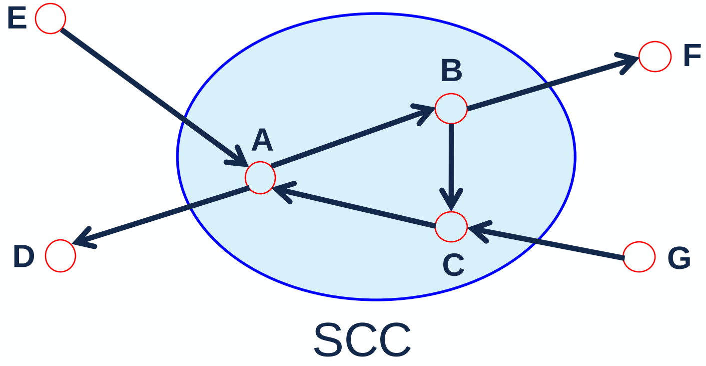


Graph on the left is connected but not strongly connected (e.g., there is no way to get from F to G by following the edge directions).

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

Course Logistics

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

Course Logistics (1)

- Meeting Times:
  - 12:30 PM - 01:45 PM, Wednesday & Friday
  - Urbana-Champaign Campus | Siebel Center for Comp Sci | Room 1304
  - Videos of the lectures will be recorded and posted on Canvas
- Structure of lectures:
  - 70-80 minutes of a lecture
  - During this time, you can ask questions
  - 10 minutes of a live Q&A/discussion session at the end of the lecture
- Teaching Staff
  - Instruction: Jiaxuan You
  - Course TA: Zirui Cheng

Course Logistics (2)

- 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
- Communication
- Office Hours

Course Logistics (3)

- Final grade will be composed of:
  - Coding assignments: 20%
    - 5 coding assignments using Google Colab, each worth 4%
  - Paper presentation: 20%
  - Course project: 60%
    - Proposal: 20%; Final report: 70%; Poster: 10%
  - Extra credit: Ed participation
    - Used if you are on the boundary between grades

Course Outline

Date	Topic	Date	Topic
Tue, 1/10	1. Introduction; Machine Learning for Graphs	Tue, 2/14	11. Community Structure in Networks
Thu, 1/12	2. Node Embeddings	Thu, 2/16	12. Traditional Generative Models for Graphs
Tue, 1/17	3. Label Propagation for Node Classification	Tue, 2/21	13. Deep Generative Models for Graphs
Thu, 1/19	4. Graph Neural Networks 1: GNN Model	Thu, 2/23	14. Advanced Topics on GNNs
Tue, 1/24	5. Graph Neural Networks 2: Design Space	Tue, 2/28	15. Scaling up GNNs
Thu, 1/26	6. Applications of Graph Neural Networks	Thu, 3/2	16. Explainability
Tue, 1/31	7. Theory of Graph Neural Networks	Tue, 3/7	**EXAM**
Thu, 2/2	8. Knowledge Graph Embeddings	Thu, 3/9	17. Guest lecture: TBD
Tue, 2/7	9. Reasoning over Knowledge Graphs	Tue, 3/14	18. GNNs for Science
Thu, 2/9	10. Frequent Subgraph Mining with GNNs	Thu, 3/16	19. Special topics in GNNs

Thanks!

Deep Learning with Graphs
Jiaxuan You