Stanford CS224W: Machine Learning with Graphs

CS224W: Machine Learning with Graphs

Jure Leskovec, Stanford University

http://cs224w.stanford.edu

CS224W Course Logistics

- **The class meets Tue and Thu 10:30-11:50am Pacific Time on Zoom.**
  - Videos of the lectures will be recorded and posted on Canvas.
- **Structure of lectures:**
  - 60 minutes of a prerecorded lecture.
  - During this time we will be using Piazza Live Q&A
  - 20 minutes of a live Q&A session

Stanford CS224W: Course Logistics

CS224W: Machine Learning with Graphs

Jure Leskovec, Stanford University

http://cs224w.stanford.edu

Logistics: Teaching Staff

Instructor

Jure Leskovec

Advisor

Michele Catasta

Teaching Assistants

Jiaxuan You
Head TA

Div Garg

Jonathan Gomes-Selman

Weihua Hu

Natasha Sharp
Course Coordinator

Jingjing Tian

Zhitao (Rex) Ying

Zecheng Zhang

Course Outline

Date	Topic	Date	Topic
Tue, Jan 12	Introduction; Machine Learning for Graphs	**Tue, Feb 16**	Reasoning over Knowledge Graphs
Thu, Jan 14	Traditional Methods for ML on Graphs	**Thu, Feb 18**	Frequent Subgraph Mining with GNNs
Tue, Jan 19	Node Embeddings	**Tue, Feb 23**	Community Structure in Networks
Thu, Jan 21	Link Analysis: PageRank	**Thu, Feb 25**	Traditional Generative Models for Graphs
Tue, Jan 26	Label Propagation for Node Classification	**Tue, Mar 2**	Deep Generative Models for Graphs
Thu, Jan 28	Graph Neural Networks 1: GNN Model	**Thu, Mar 4**	Scaling Up GNNs
Tue, Feb 2	Graph Neural Networks 2: Design Space	**Tue, Mar 9**	Learning on Dynamic Graphs
Thu, Feb 4	Applications of Graph Neural Networks	**Thu, Mar 11**	GNNs for Computational Biology
Tue, Feb 9	Theory of Graph Neural Networks	**Tue, Mar 16**	GNNs for Science
Thu, Feb 11	Knowledge Graph Embeddings	**Thu, Mar 18**	Industrial Applications of GNNs

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

- **Piazza Q&A website:**
  - http://piazza.com/stanford/win2021/cs224w
    - Register with your @stanford.edu email
  - **Please participate and help each other!**
    - Don't post code, annotate your questions, search for answers before you ask
    - Given COVID/virtual class, this will be the main mode of communication
- **To reach course staff (prof/TAs), always use:**
  - cs224w-win2021-staff@lists.stanford.edu
- We will post course announcements to Piazza (make sure you check it regularly)

Work for the Course & Grading

- **Final grade will be composed of:**
  - **Homework: 30%**
    - Homework 1, 2, 3, each worth 10%
  - **Coding assignment: 30%**
    - 5 Coding assignments using Google Colab, each worth 6%
  - **Course project: 40%**
    - Proposal: 30%
    - Final report: 70%
  - **Extra credit: Piazza participation, code contribution**
    - Used if you are on the boundary between grades

Homework, Write-ups

- **Assignments are long and take time (~10h) Start early!**
  - A combination of data analysis, algorithm design, and math
  - Generally due on Thursdays 23:59 Pacific Time
- **How to submit?**
  - **Upload via Gradescope (http://gradescope.com)**
    - **You will be automatically registered to Gradescope once you officially enroll in CS224W**
    - **Each answer must start on a new page. Read carefully the course info page!**
  - **Both homework (including code) and project deliverables** must be uploaded to Gradescope!
- **Total of 2 Late Periods (LP) per student:**
  - Max 1 late period per assignment (no LP for final report)

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.

Course Projects

- **Course project:**
  - Make predictions on a network dataset
- **Performed in groups of up to 3 students:**
  - Fine to have groups of 1 or 2. The team size will be taken under consideration when evaluating the scope of the project in breadth and depth. But 3 person teams can be more efficient.
  - Project is the **important work** for the class
  - Teaching staff will help with problems and data
  - More details to follow.
- **Read: http://cs224w.stanford.edu/info.html**

Course Schedule

Week	Assignment	Due on (11:59pm PT)
1	**Colab 0**	
1	**Colab 1**	Thu, Jan 28
2	**Homework 1**	Thu, Feb 4
3	**Colab 2**	Thu, Feb 11
	Project Proposal	Thu, Feb 11
4	**Homework 2**	Thu, Feb 18
5	**Colab 3**	Thu, Feb 25
6	**Homework 3**	Thu, Mar 4
7	**Colab 4**	Thu, Mar 11
8	**Colab 5**	Thu, Mar 18
	Project Report	Sun, Mar 21 **(No Late Periods!)**

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)

Graph Machine Learning Tools

- **We use PyTorch Geometric (PyG)**
- **We further recommend:**
  - **DeepSNAP:** Library that assists deep learning on graphs.
    - Flexible graph manipulation, standard data split pipeline, …
  - **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

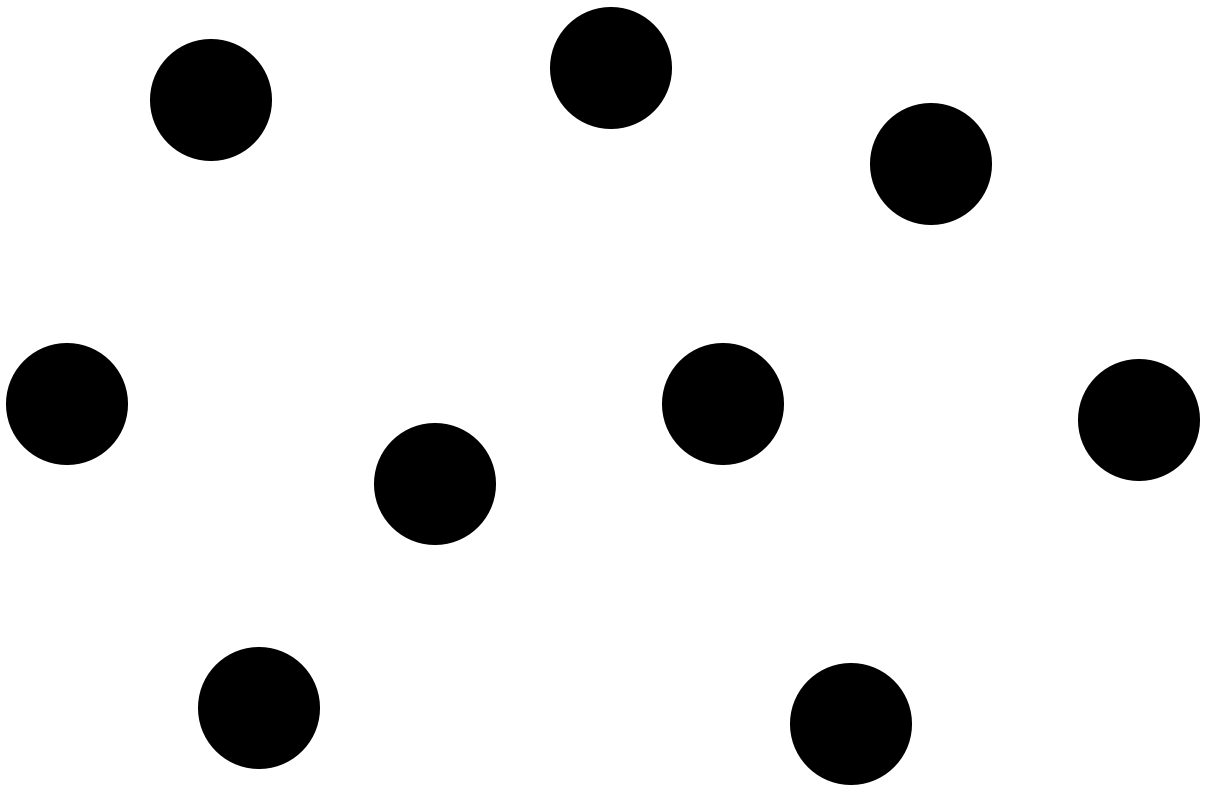
CS224W: Machine Learning with Graphs

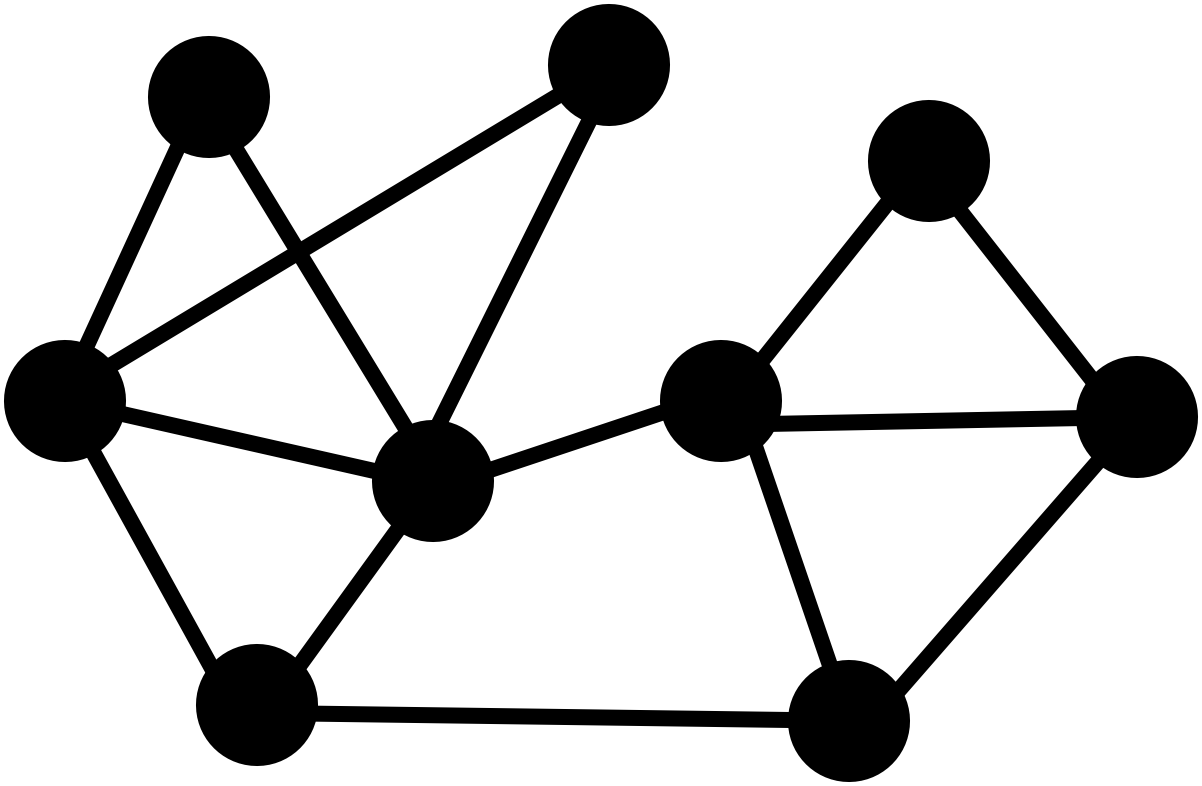
Jure Leskovec, Stanford University

http://cs224w.stanford.edu

Why Graphs?

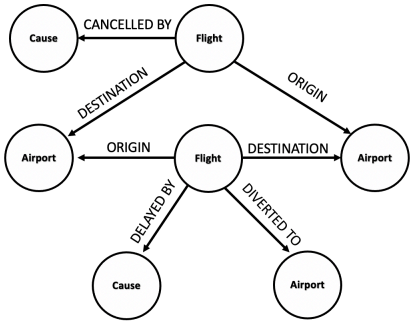
Graphs are a general language for describing and analyzing entities with relations/interactions





Graph

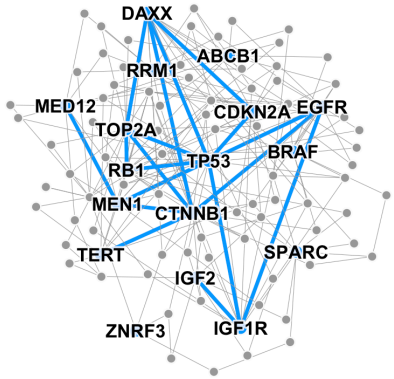
Many Types of Data are Graphs (1)



Event Graphs

Image credit: SalientNetworks

Computer Networks



Disease Pathways

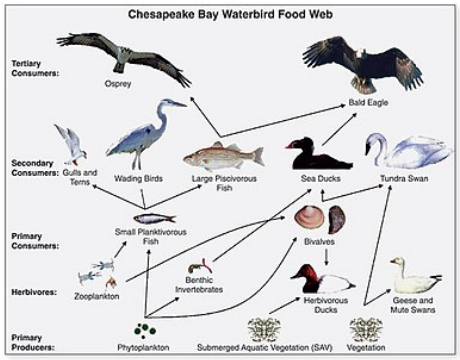


Image credit: Wikipedia

Food Webs

Image credit: Pinterest

Particle Networks

Image credit: visitlondon.com

Underground Networks

Many Types of Data are Graphs (2)

Image credit: Medium

Social Networks

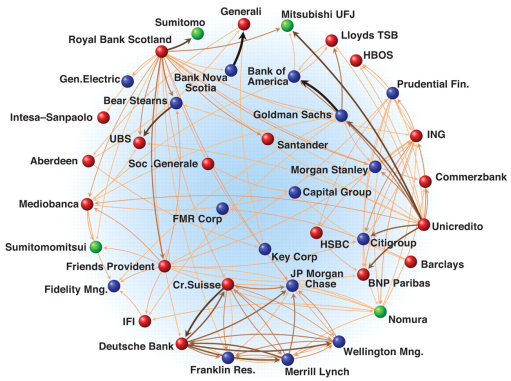
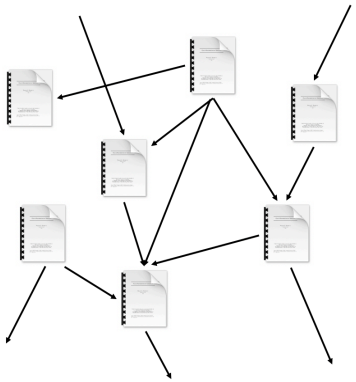


Image credit: Science

Economic Networks

Image credit: Lumen Learning

Communication Networks



Citation Networks

Image credit: Missoula Current News

Internet

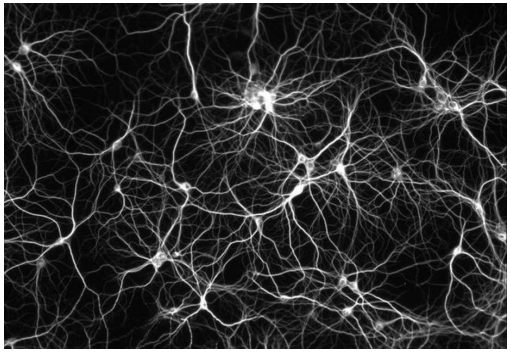


Image credit: The Conversation

Networks of Neurons

Many Types of Data are Graphs (3)

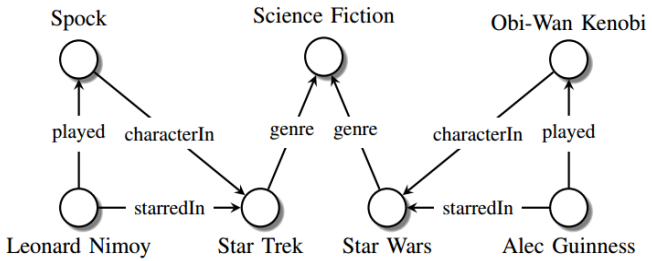


Image credit: Maximilian Nickel et al

Knowledge Graphs

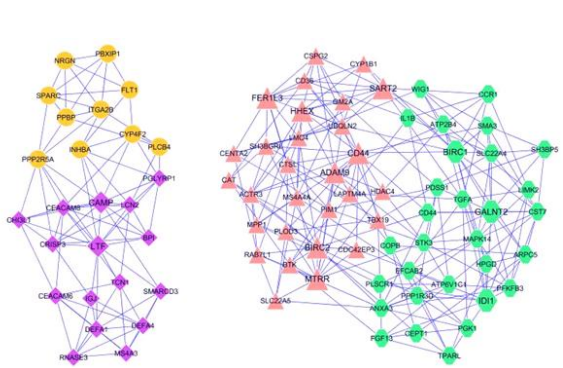


Image credit: ese.wustl.edu

Regulatory Networks

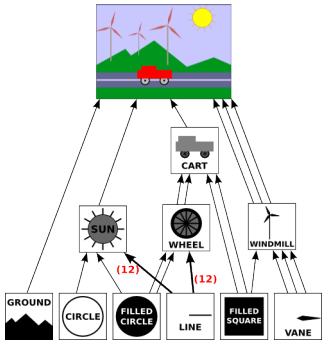


Image credit: math.hws.edu

Scene Graphs

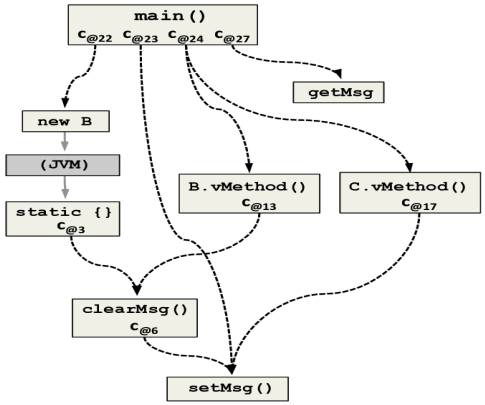


Image credit: ResearchGate

Code Graphs

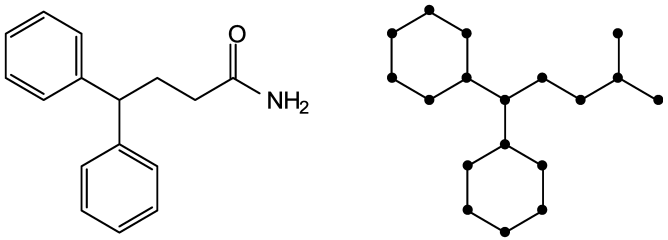


Image credit: MDPI

Molecules

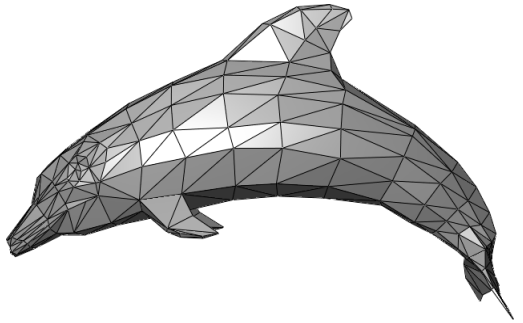


Image credit: Wikipedia

3D Shapes

Types of Networks and Graphs

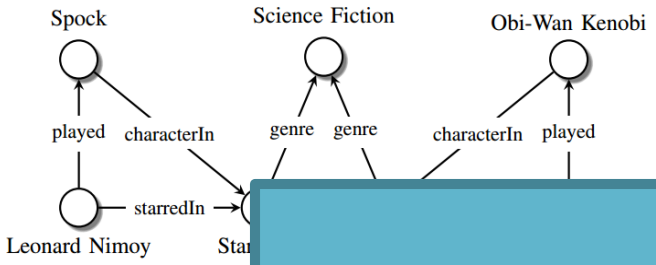
- **Networks (also known as Natural Graphs):**
  - **Social networks:**
    - **Society** is a collection of 7+ billion individuals
  - **Communication and transactions:**
    - Electronic devices, phone calls, financial transactions
  - **Biomedicine:**
    - Interactions between **genes/proteins** regulate life
  - **Brain connections:**
    - Our **thoughts** are hidden in the connections between billions of neurons

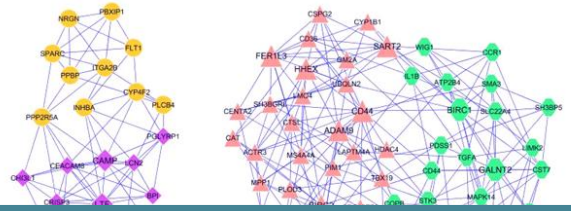
Types of Networks and Graphs

- **Graphs (as a representation):**
  - **Information/knowledge** are organized and linked
  - **Software** can be represented as a graph
  - **Similarity networks:** Connect similar data points
  - **Relational structures:** Molecules, Scene graphs, 3D shapes, Particle-based physics simulations

Sometimes the distinction between networks & graphs is blurred

Graphs and Relational Data





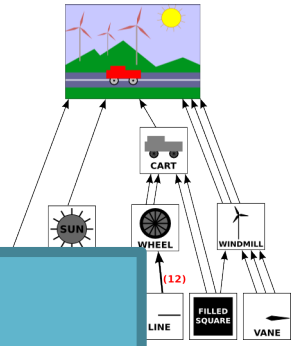


Image cred

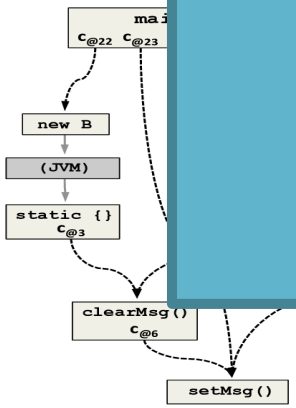
ath.hws.edu

Know

Graphs

Main question:

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



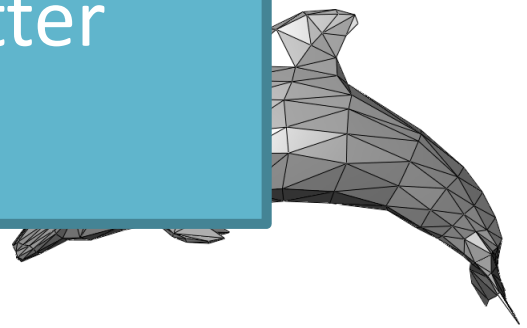


Image credit: ResearchGate

Image credit: MDPI

Image credit: Wikipedia

Code Graphs

Molecules

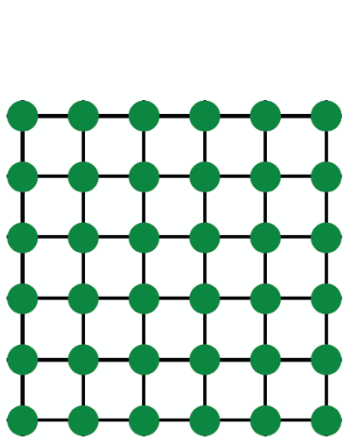
3D Shapes

Graphs: Machine Learning

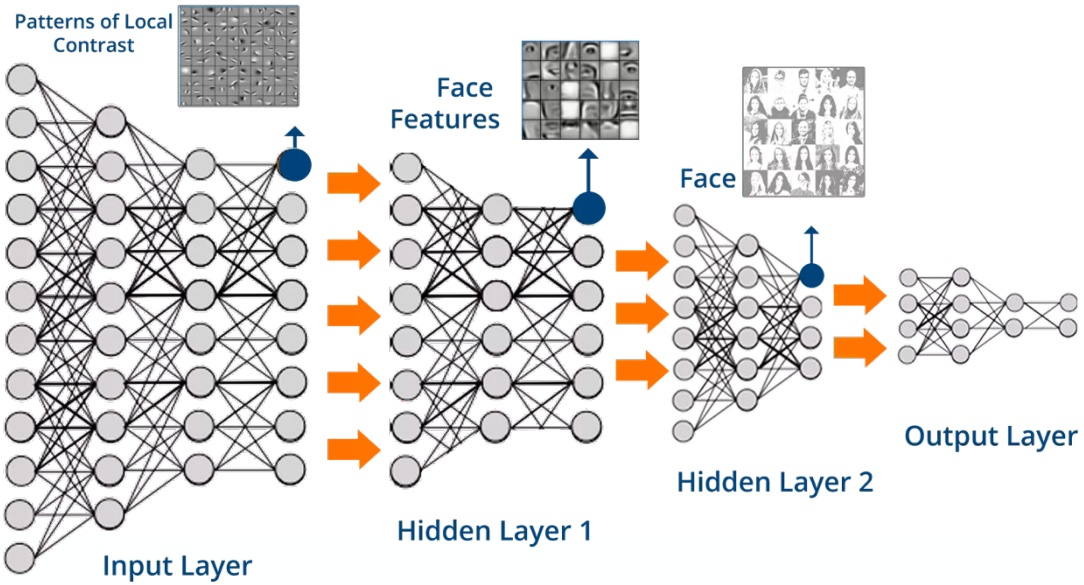
Complex domains have a rich relational structure, which can be represented as a **relational graph**

By explicitly modeling relationships we achieve better performance!

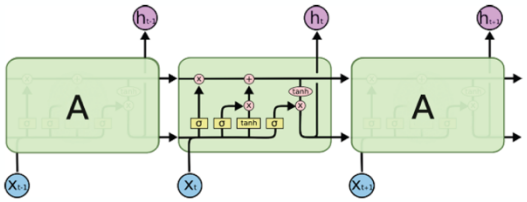
Modern ML Toolbox



Images



Text/Speech

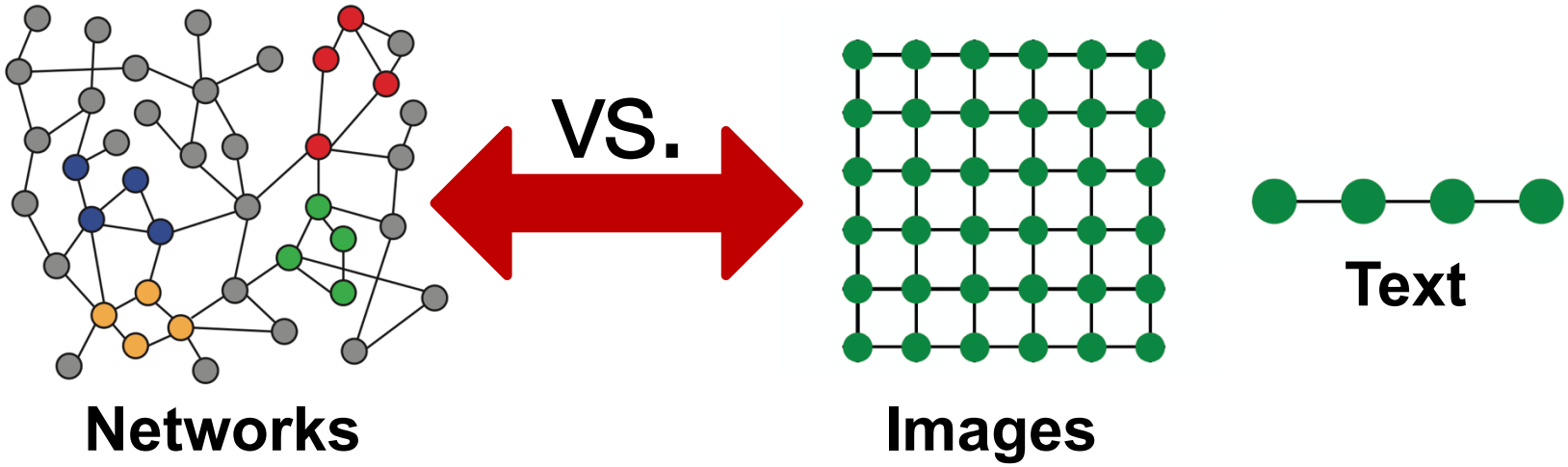


Modern deep learning toolbox is designed for simple sequences & grids

Why is it Hard?

Networks are complex.

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



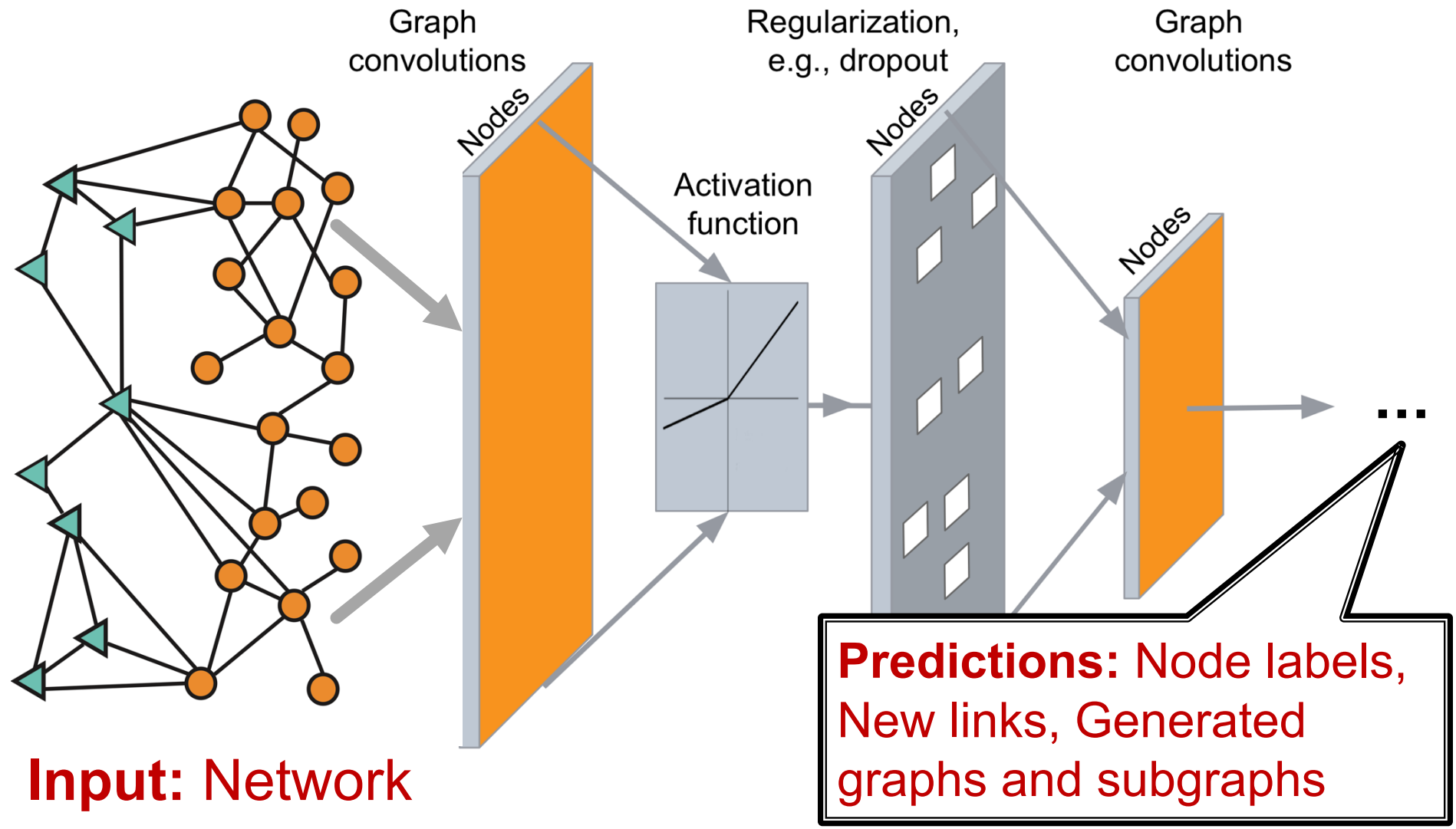
- No fixed node ordering or reference point
- Often dynamic and have multimodal features

This Course

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

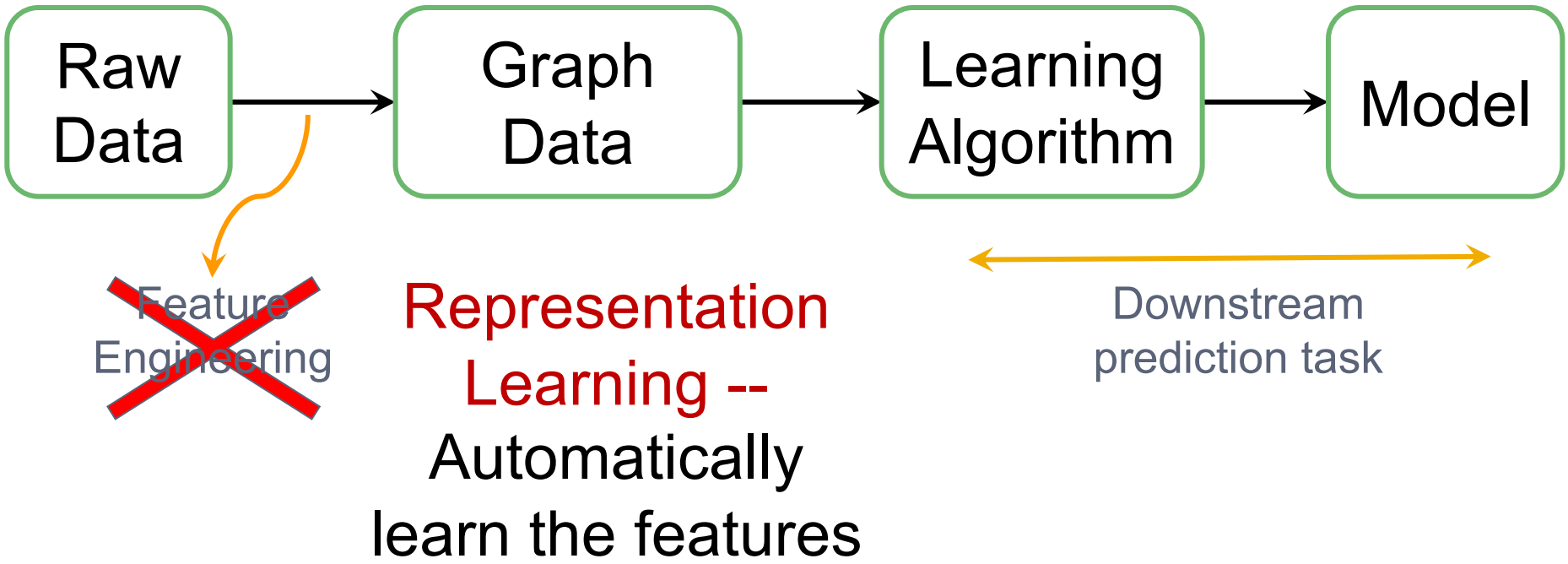
Graphs are the new frontier of deep learning

CS224W: Deep Learning in Graphs



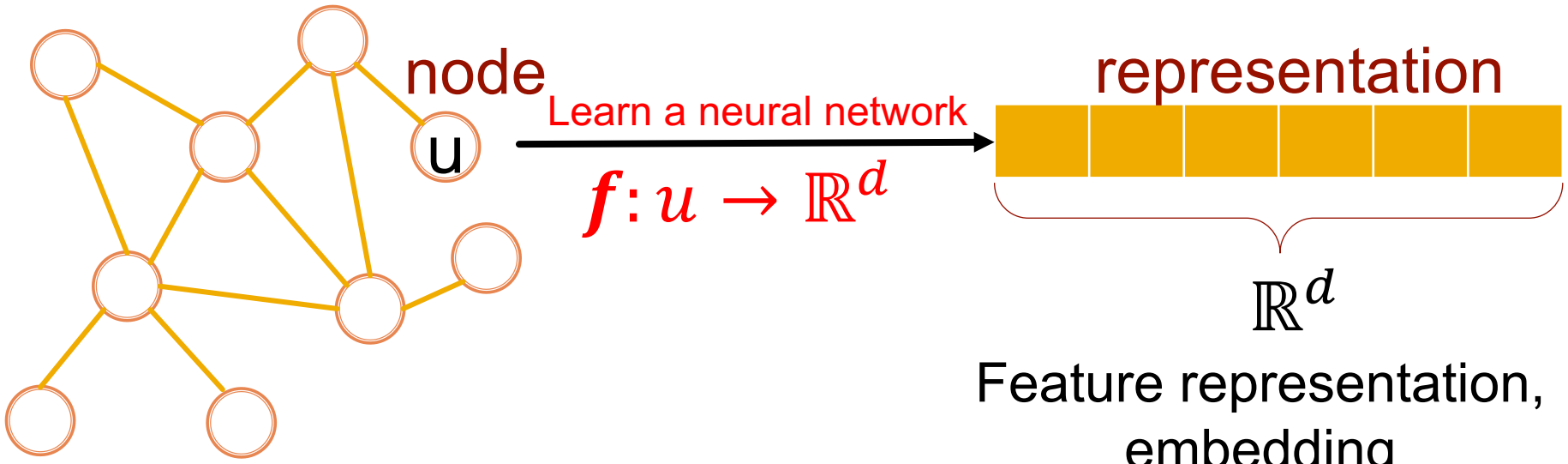
CS224W & Representation Learning

(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
- Applications to Biomedicine, Science, Industry

Topics Covered in CS224W

Date	Topic	Date	Topic
Tue, Jan 12	Introduction; Machine Learning for Graphs	**Tue, Feb 16**	Reasoning over Knowledge Graphs
Thu, Jan 14	Traditional Methods for ML on Graphs	**Thu, Feb 18**	Frequent Subgraph Mining with GNNs
Tue, Jan 19	Node Embeddings	**Tue, Feb 23**	Community Structure in Networks
Thu, Jan 21	Link Analysis: PageRank	**Thu, Feb 25**	Traditional Generative Models for Graphs
Tue, Jan 26	Label Propagation for Node Classification	**Tue, Mar 2**	Deep Generative Models for Graphs
Thu, Jan 28	Graph Neural Networks 1: GNN Model	**Thu, Mar 4**	Scaling Up GNNs
Tue, Feb 2	Graph Neural Networks 2: Design Space	**Tue, Mar 9**	Learning on Dynamic Graphs
Thu, Feb 4	Applications of Graph Neural Networks	**Thu, Mar 11**	GNNs for Computational Biology
Tue, Feb 9	Theory of Graph Neural Networks	**Tue, Mar 16**	GNNs for Science
Thu, Feb 11	Knowledge Graph Embeddings	**Thu, Mar 18**	Industrial Applications of GNNs

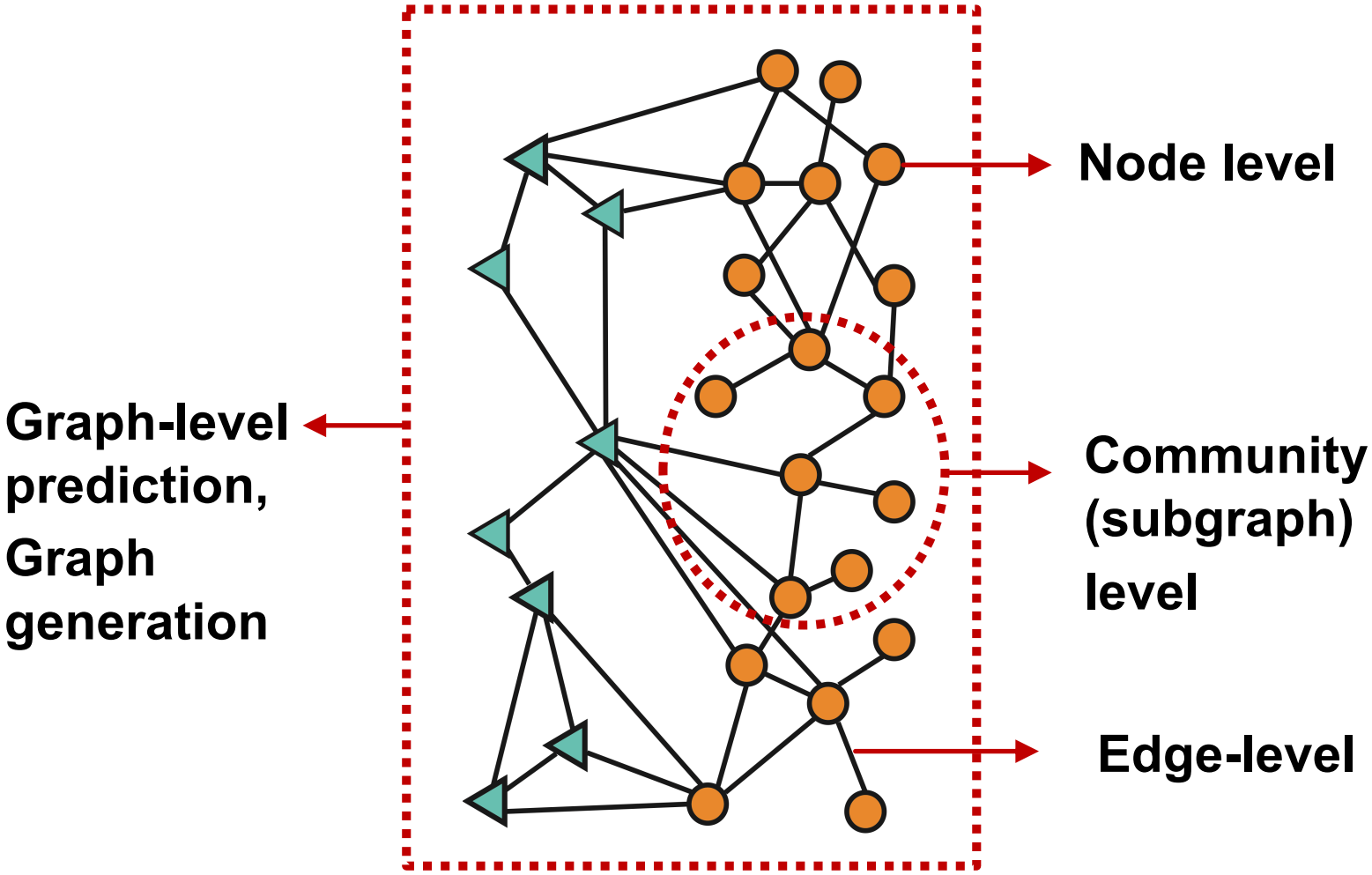
Stanford CS224W: Applications of Graph ML

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

Every protein is made up of a sequence of amino acids bonded together

These amino acids interact locally to form shapes like helices and sheets

These shapes fold up on larger scales to form the full three-dimensional protein structure

Proteins can interact with other proteins, performing functions such as signalling and transcribing DNA

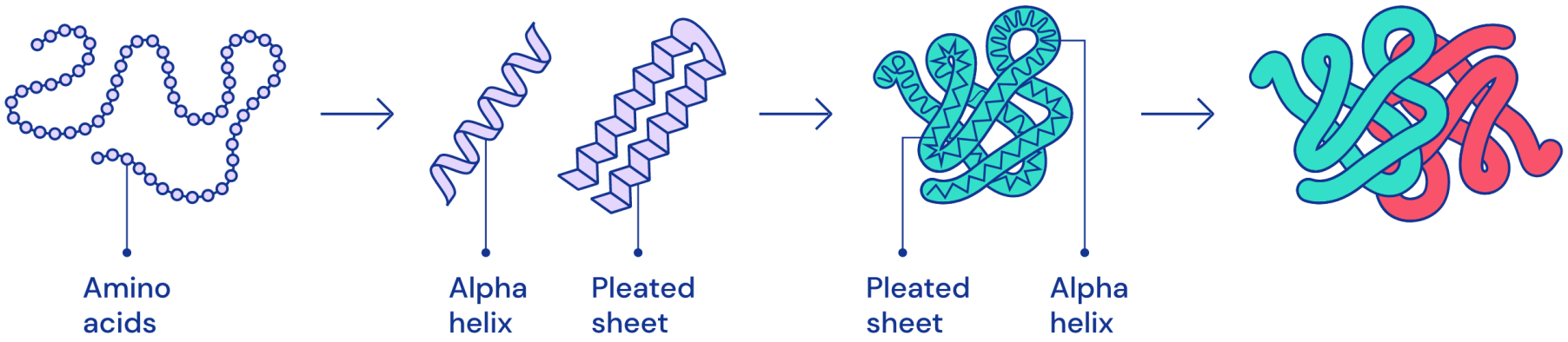
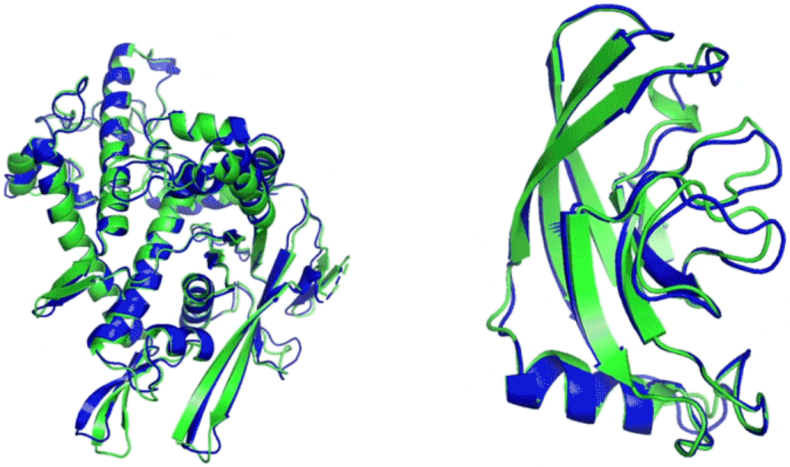


Image credit: DeepMind

The Protein Folding Problem

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



T1037 / 6vr4
90.7 GDT
(RNA polymerase domain)

T1049 / 6y4f
93.3 GDT
(adhesin tip)

Experimental result
Computational prediction

Image credit: DeepMind

AlphaFold: Impact

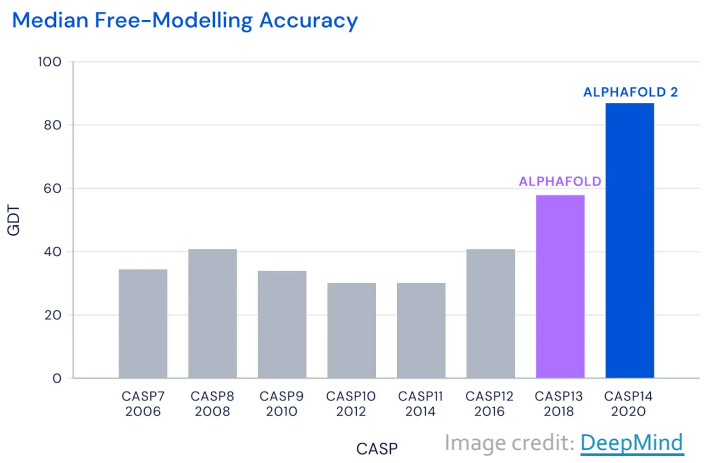


Image credit: DeepMind

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

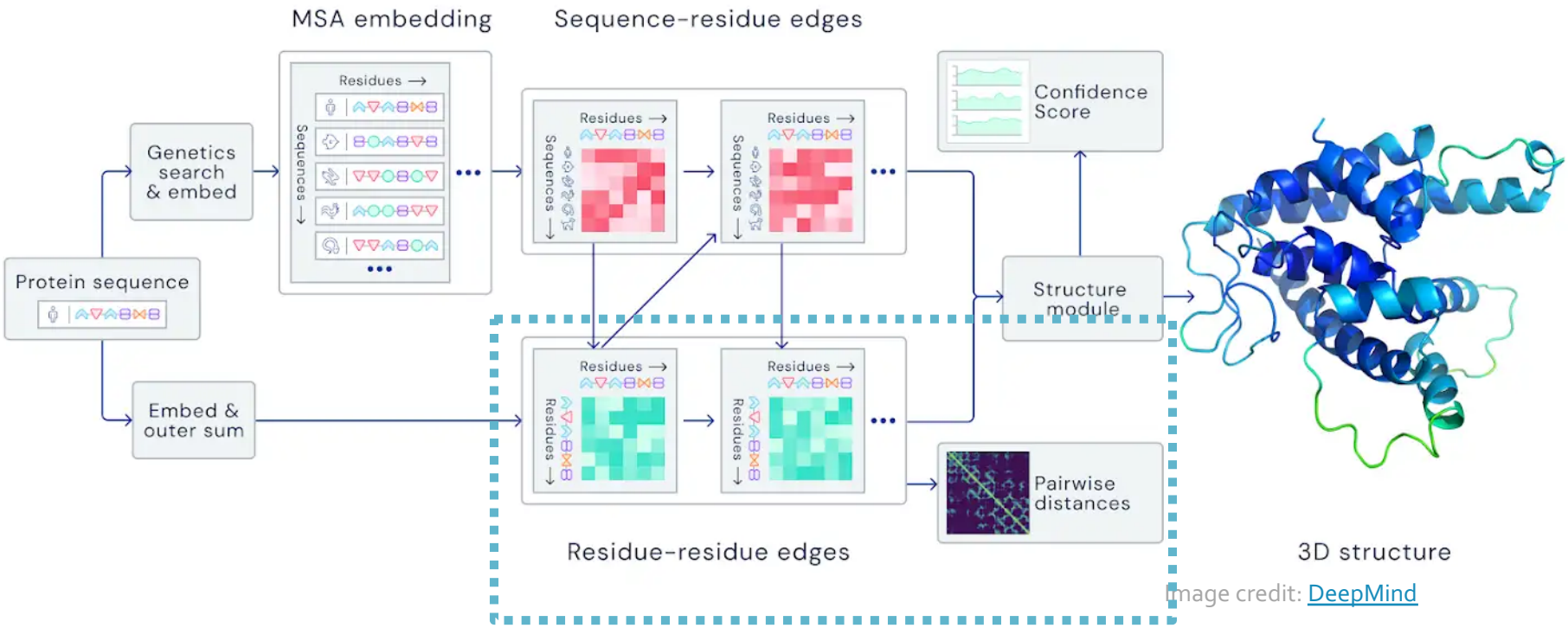
Has Artificial Intelligence 'Solved' Biology's Protein-Folding Problem?

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)

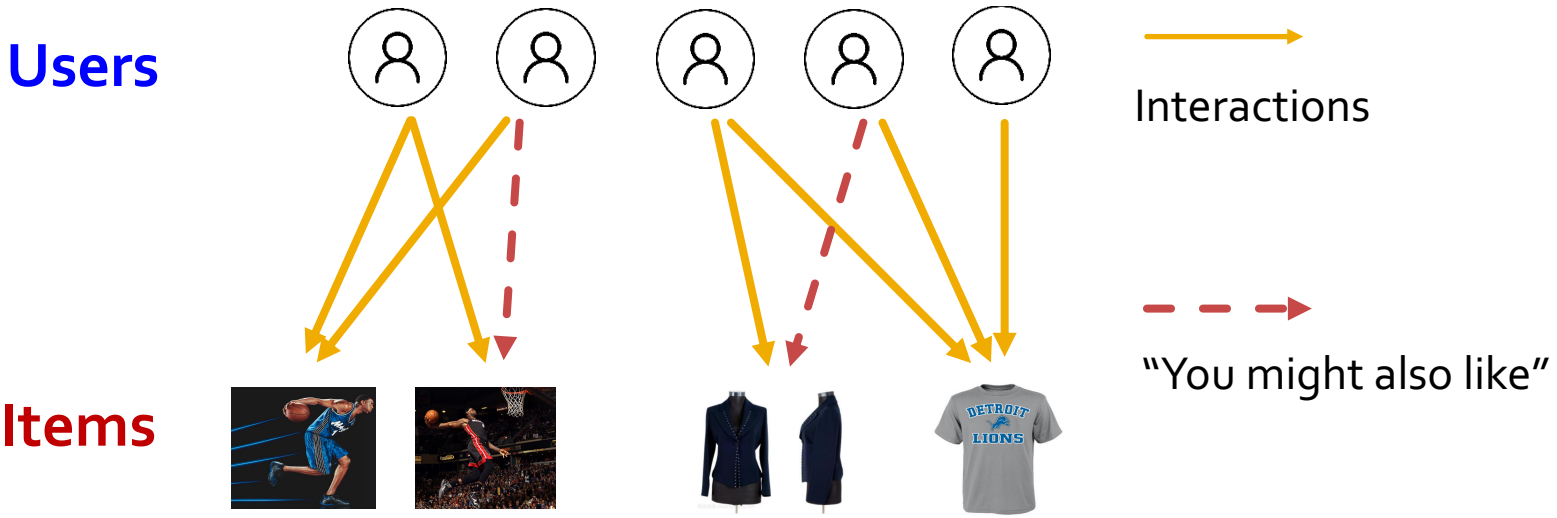


Spatial graph

Examples of Edge-level ML Tasks

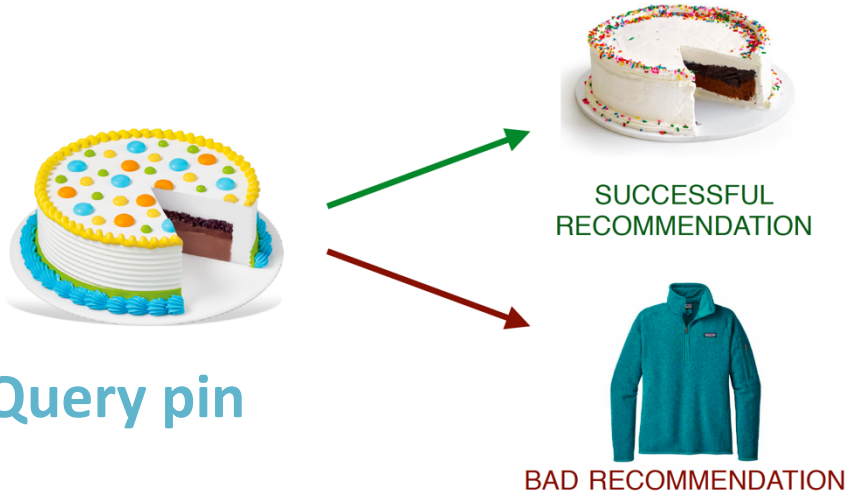
Example (2): Recommender Systems

- **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**



PinSage: Graph-based Recommender

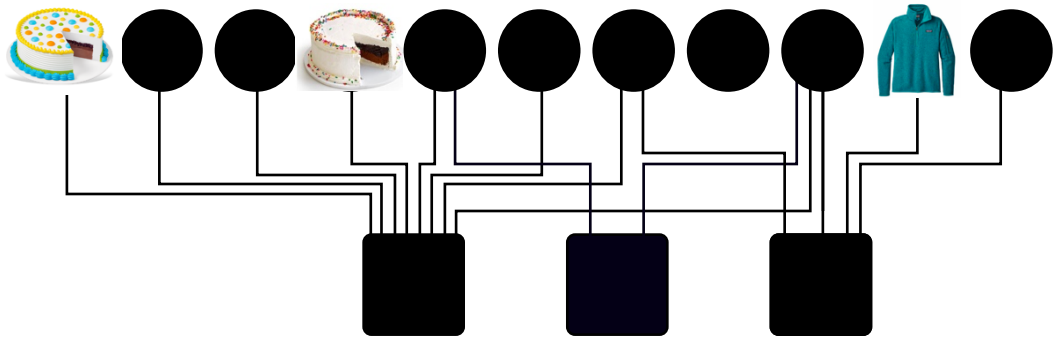
Task: Recommend related pins to users

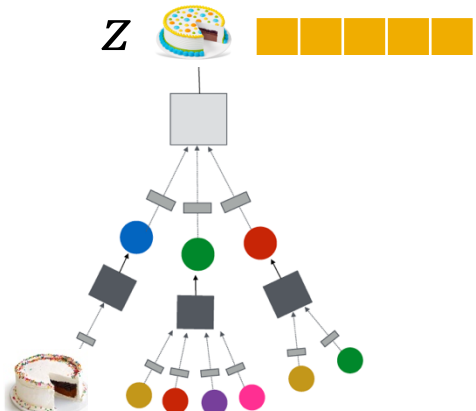


Task: Learn node embeddings z_i such that

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

Predict whether two nodes in a graph are related



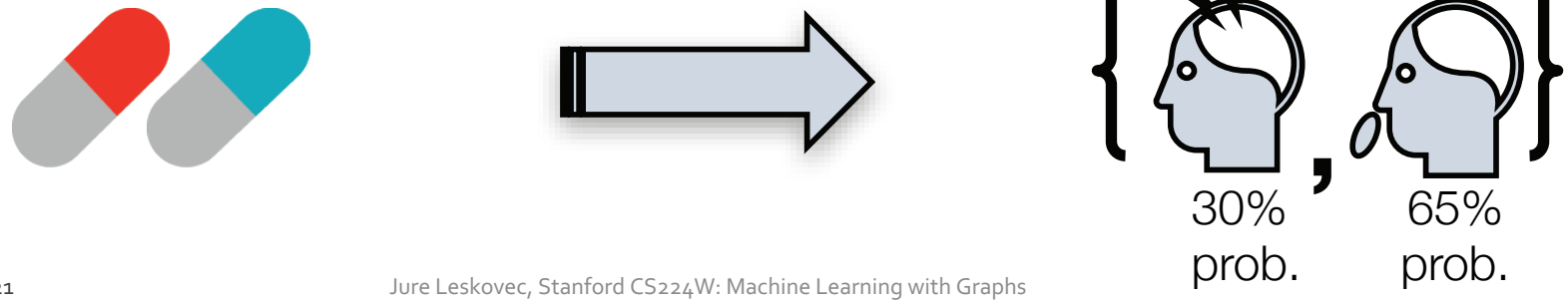


Example (3): Drug Side Effects

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.

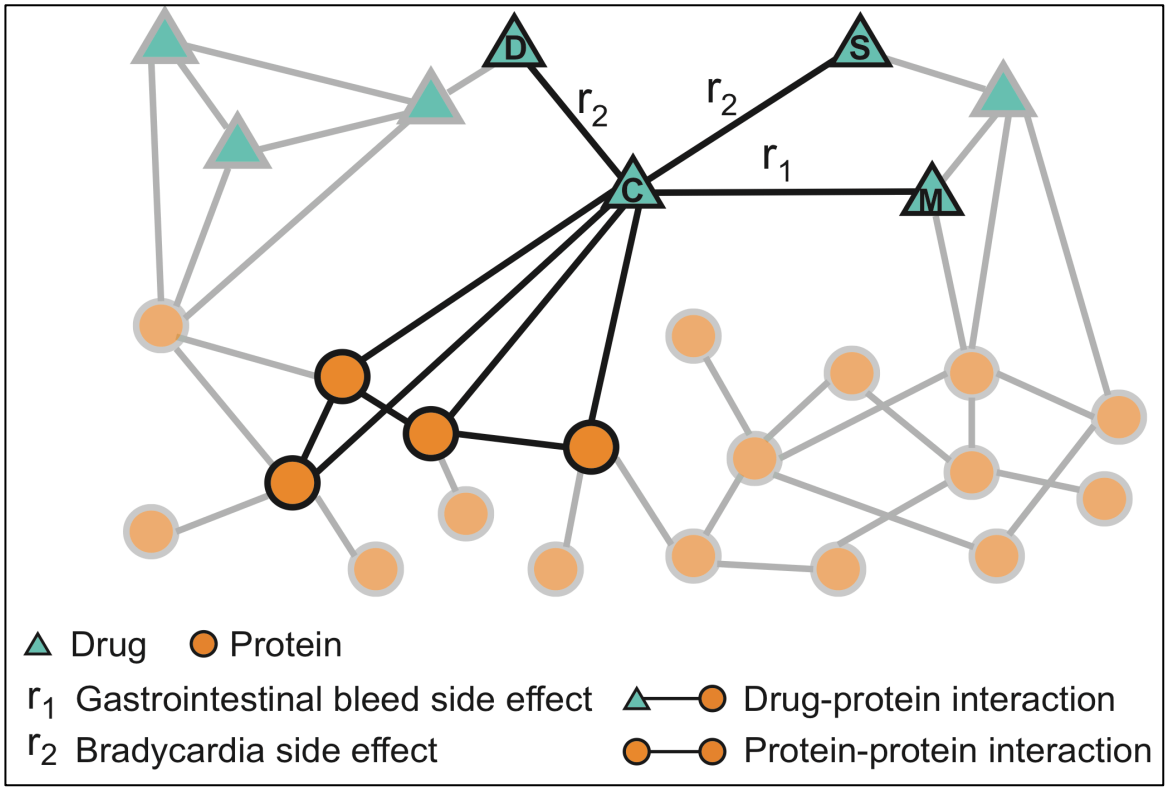
Task: Given a pair of drugs predict adverse side effects



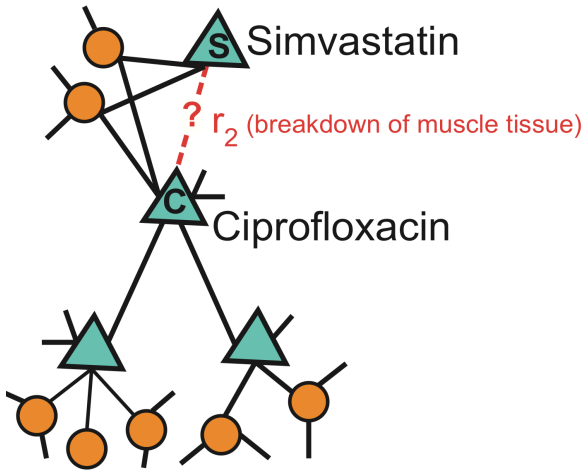
Zitnik et al., Modeling Polypharmacy Side Effects with Graph Convolutional Networks, Bioinformatics 2018

Biomedical Graph Link Prediction

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



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



Results: *De novo* Predictions

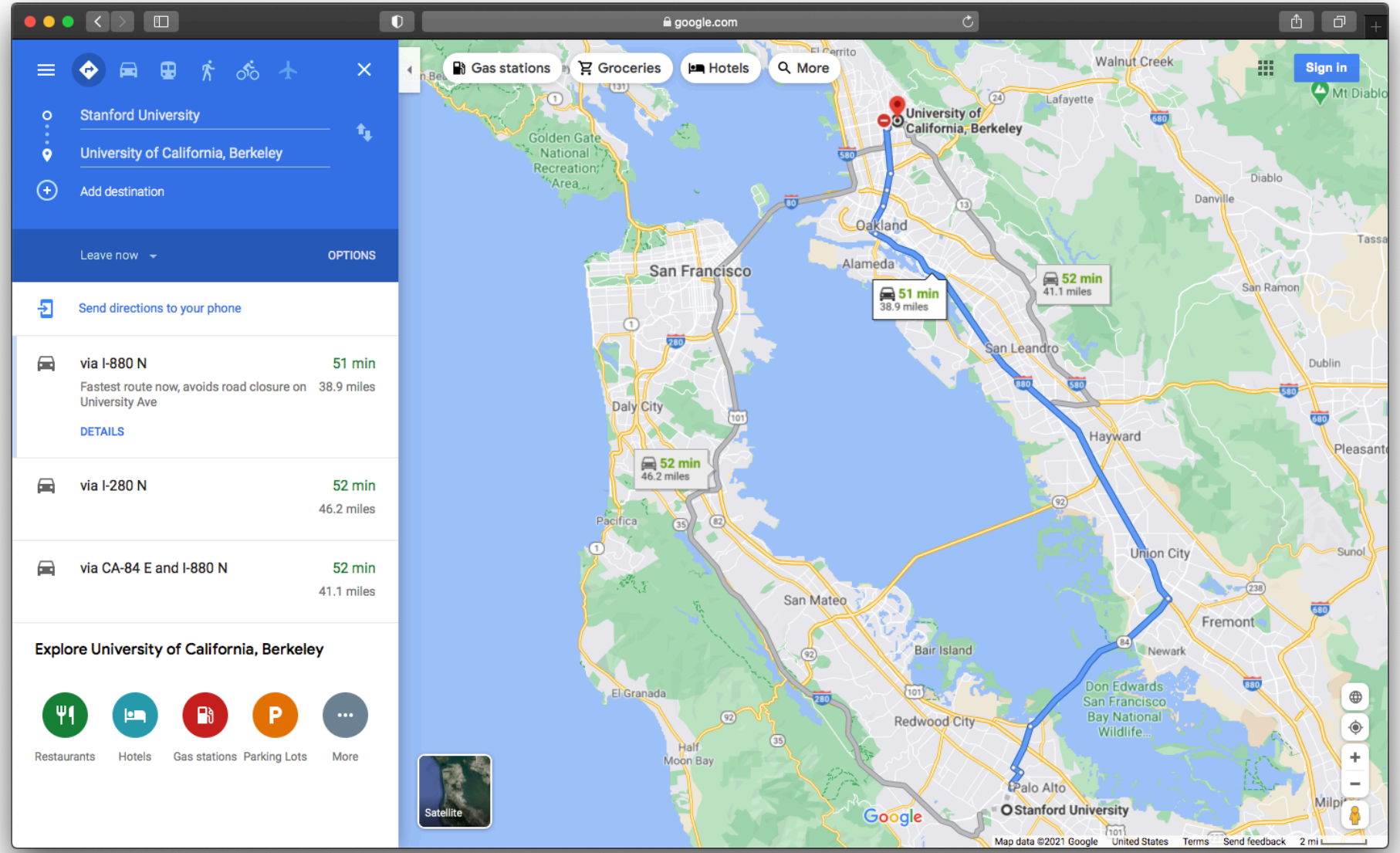
Rank	Drug c	Drug d	Side effect r	Evidence found
1	Pyrimethamine	Aliskiren	Sarcoma	Stage *et al.* 2015
2	Tigecycline	Bimatoprost	Autonomic neuropathy	
3	Omeprazole	Dacarbazine	Telangiectases	
4	Tolcapone	Pyrimethamine	Breast disorder	Bicker *et al.* 2017
5	Minoxidil	Paricalcitol	Cluster headache	
6	Omeprazole	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

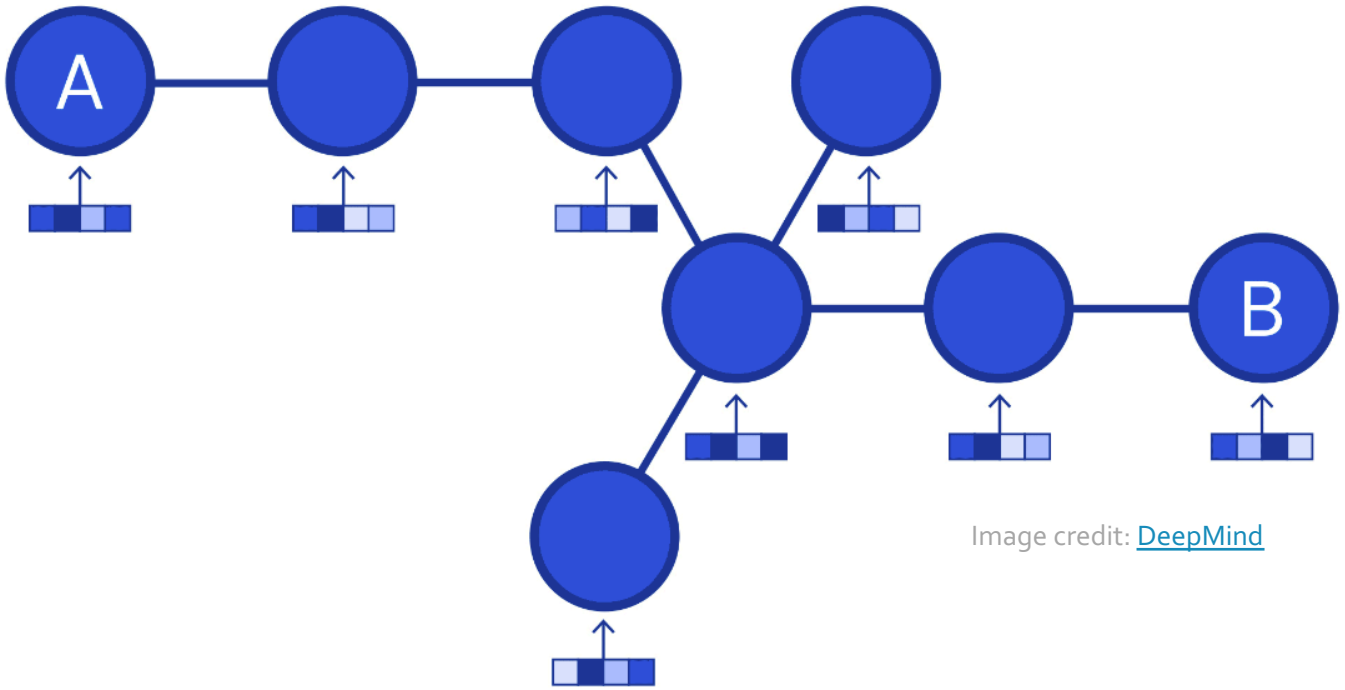
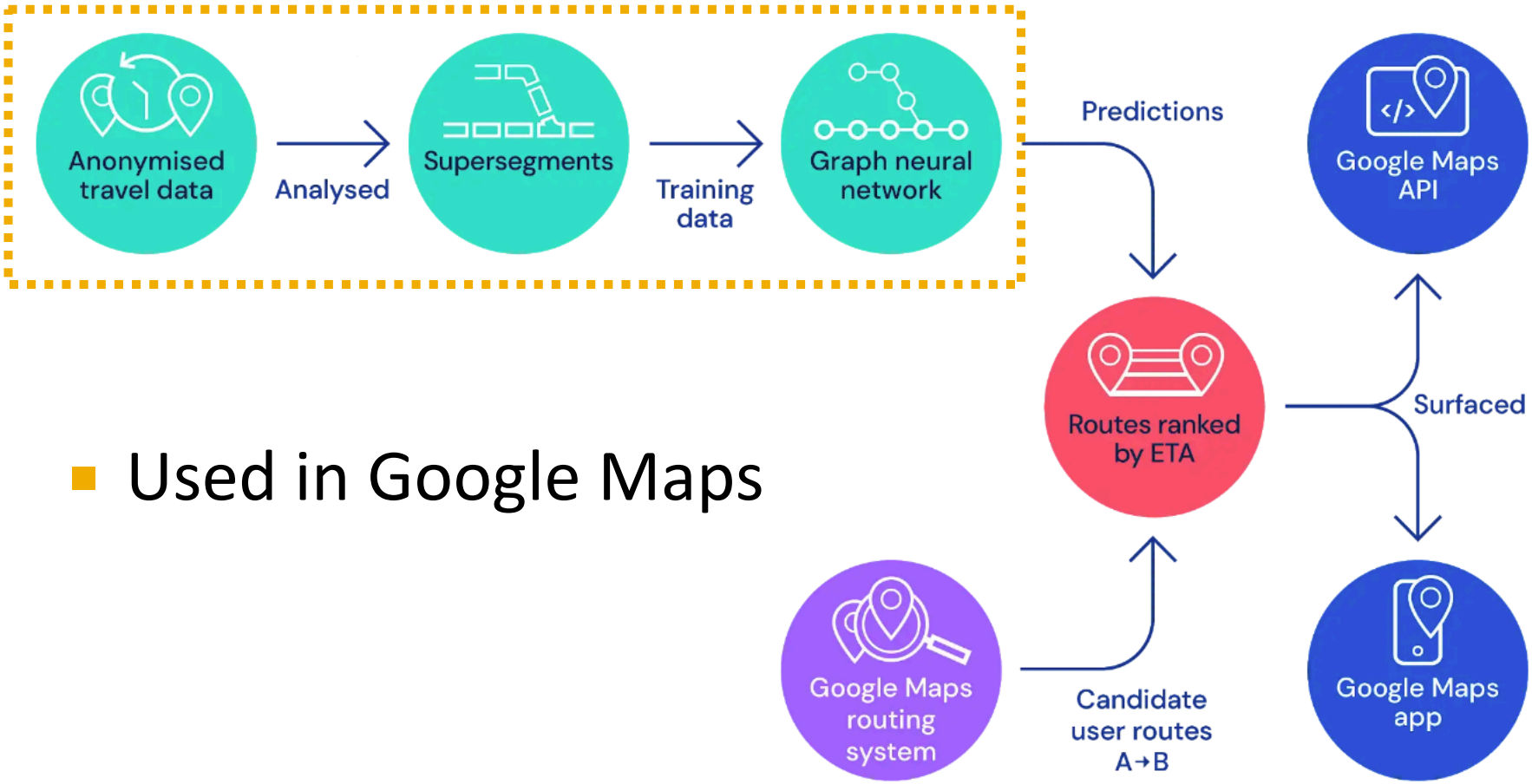


Image credit: DeepMind

Traffic Prediction via GNN

Predict via Graph Neural Networks



- Used in Google Maps

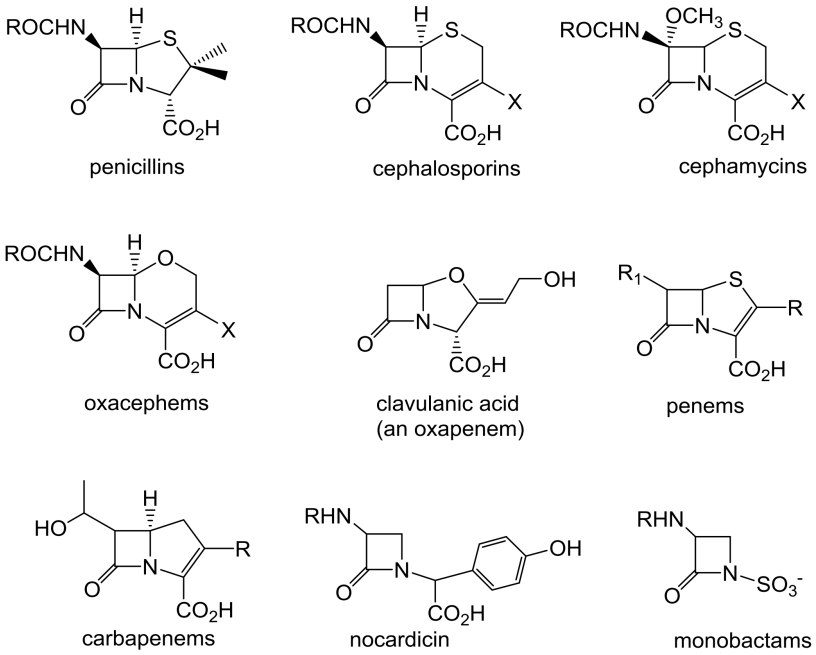
THE MODEL ARCHITECTURE FOR DETERMINING OPTIMAL ROUTES AND THEIR TRAVEL TIME.

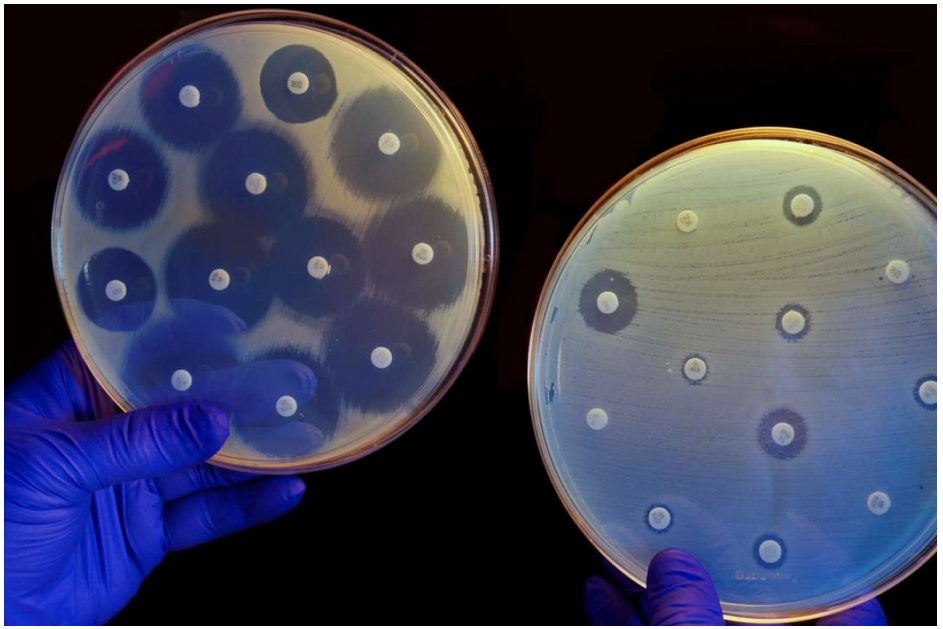
Image credit: DeepMind

Examples of Graph-level ML Tasks

Example (5): Drug Discovery

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



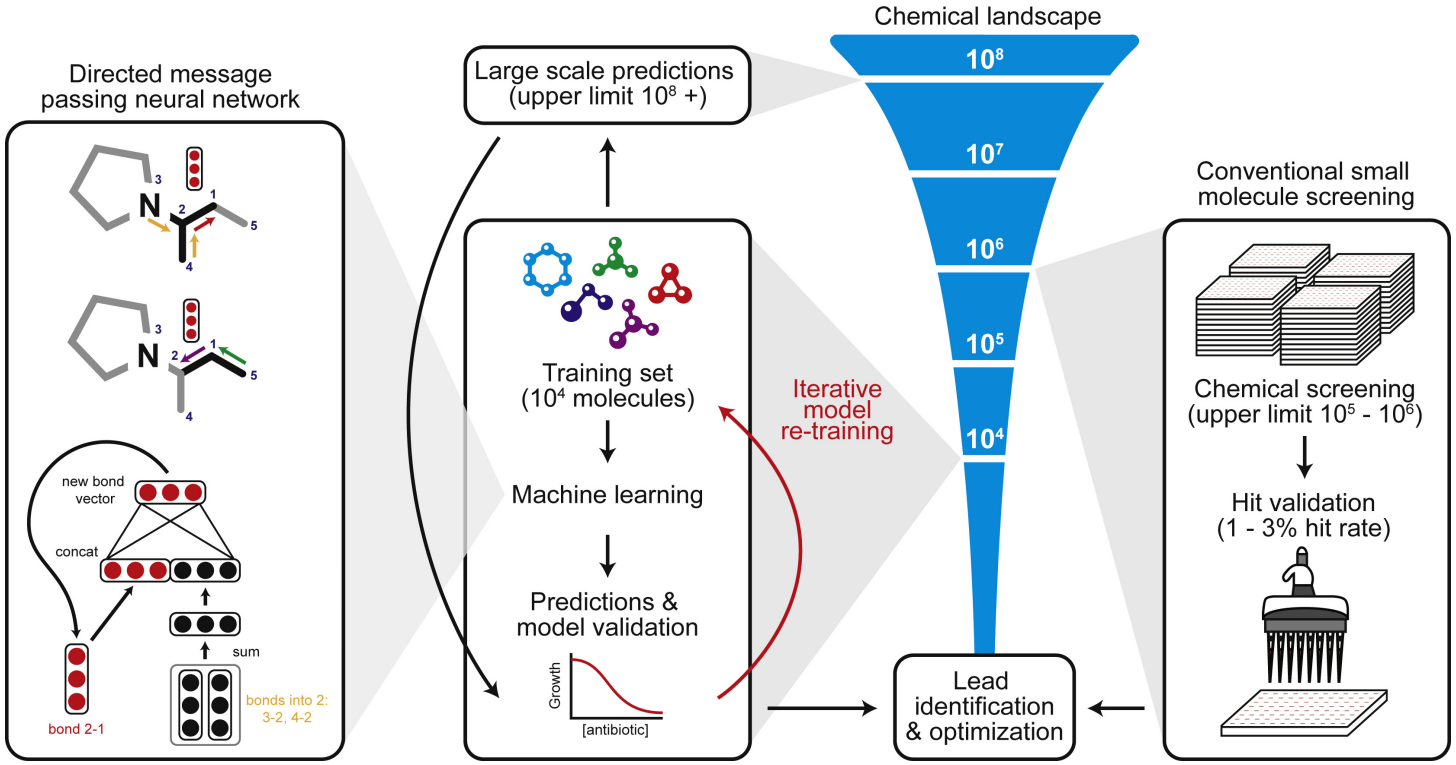


Konaklieva, Monika I. "Molecular targets of β-lactam-based antimicrobials: beyond the usual suspects." Antibiotics 3.2 (2014): 128-142.

Image credit: CNN

Deep Learning for Antibiotic Discovery

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

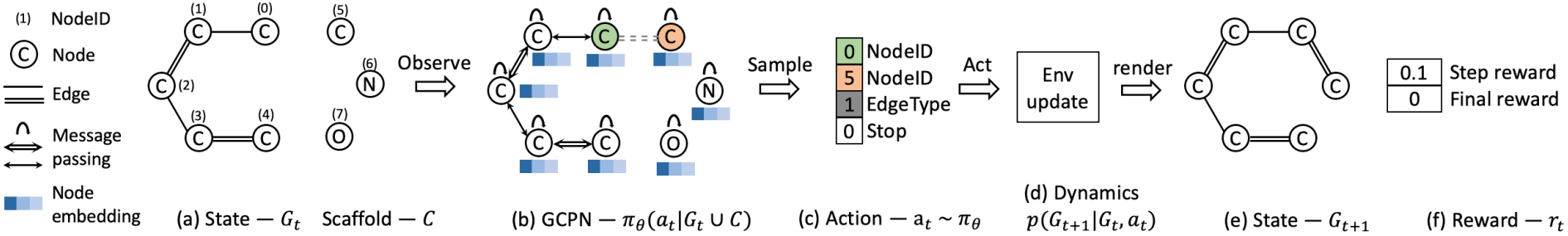


Stokes, Jonathan M., et al. "A deep learning approach to antibiotic discovery." Cell 180.4 (2020): 688-702.

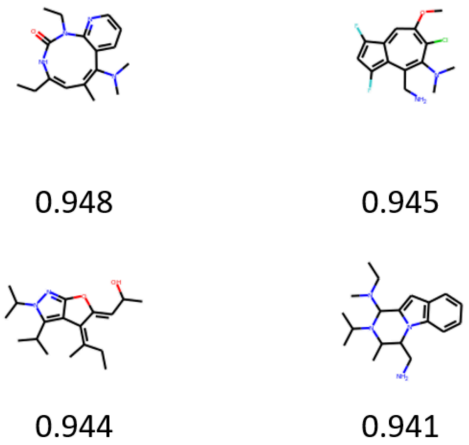
You et al., Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation, NeurIPS 2018

Molecule Generation / Optimization

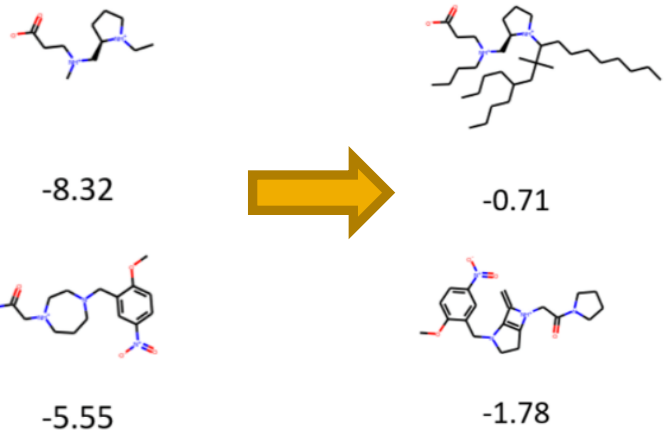
Graph generation: Generating novel molecules



Use case 1: Generate novel molecules with high drug likeness



Use case 2: Optimize existing molecules to have desirable properties



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

Example (6): Physics Simulation

Physical simulation as a graph:

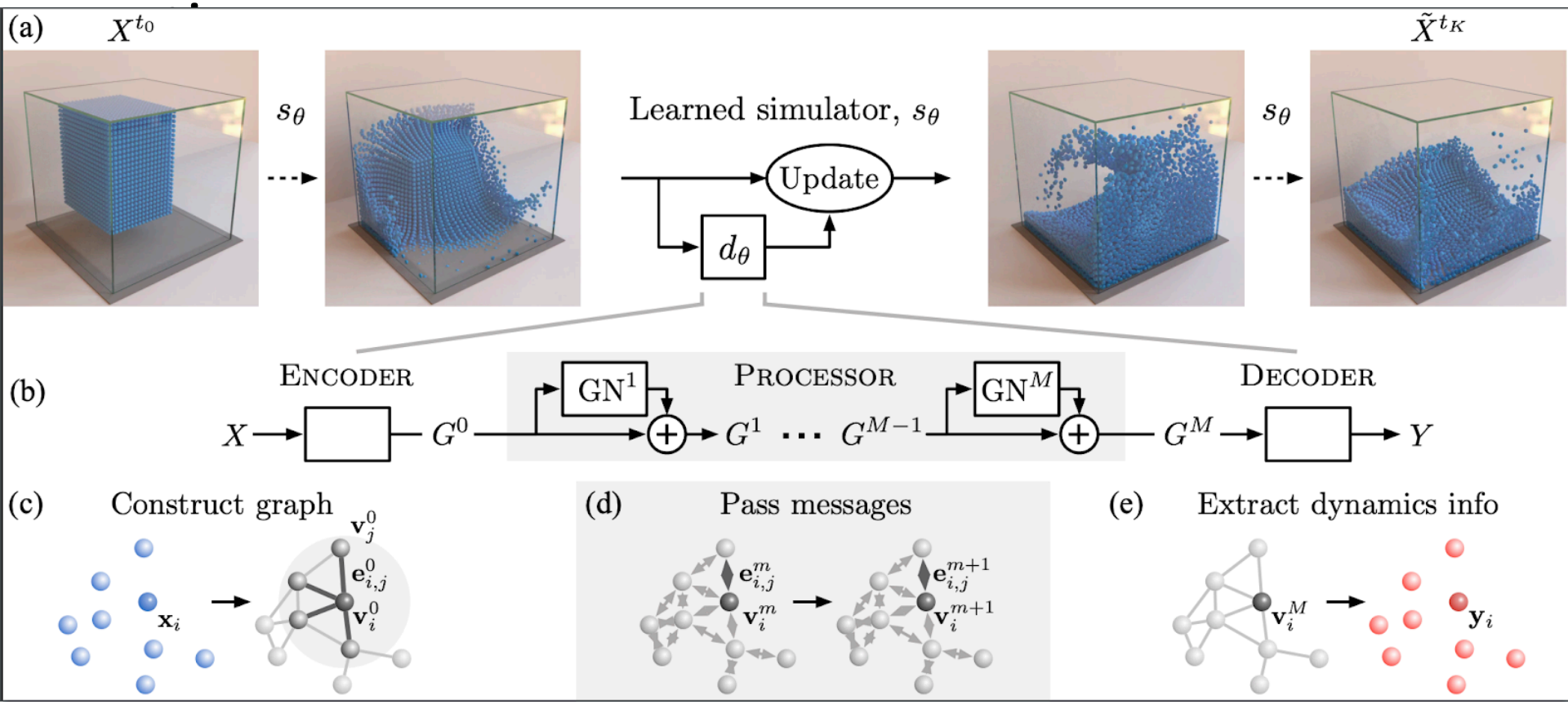
- **Nodes**: Particles
- **Edges**: Interaction between particles

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

Simulation Learning Framework

A graph evolution task:

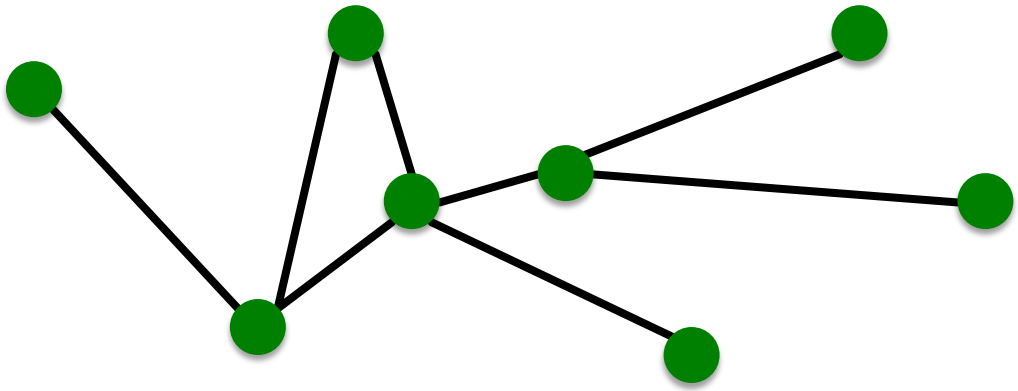
- **Goal**: Predict how a graph will evolve over



Stanford CS224W: Choice of Graph Representation

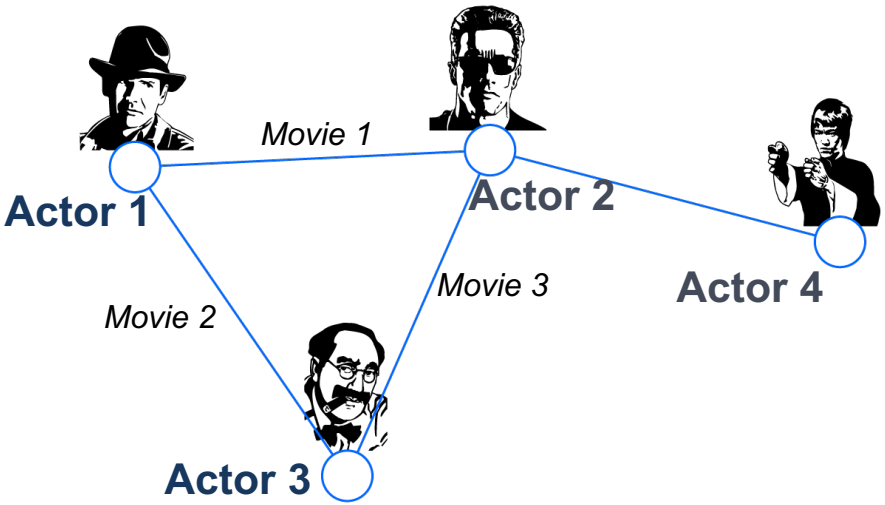
CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

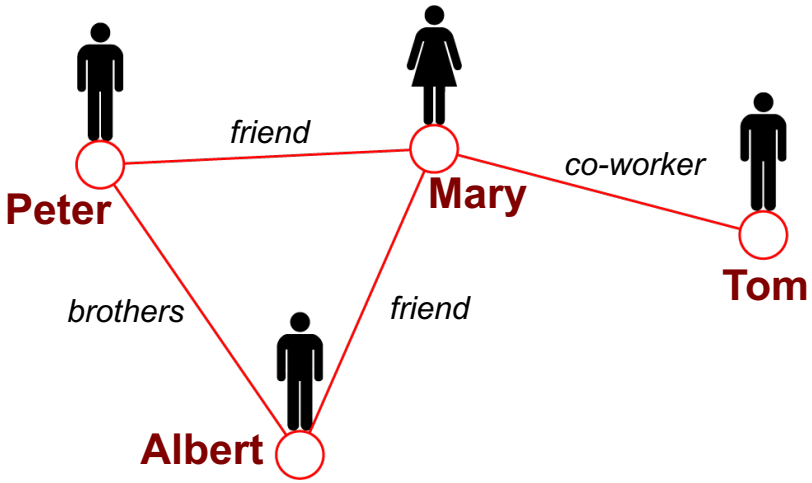
Components of a Network

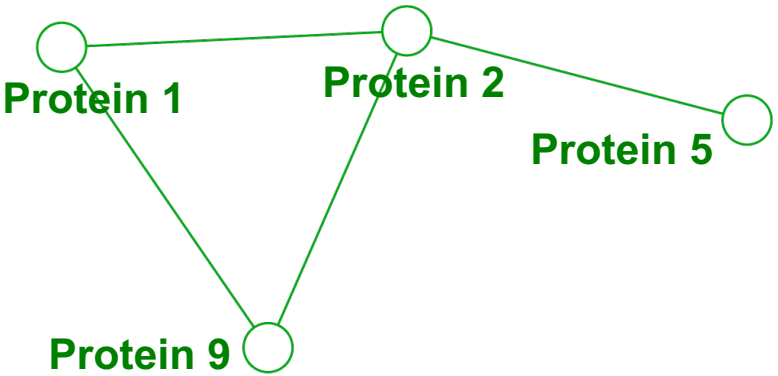


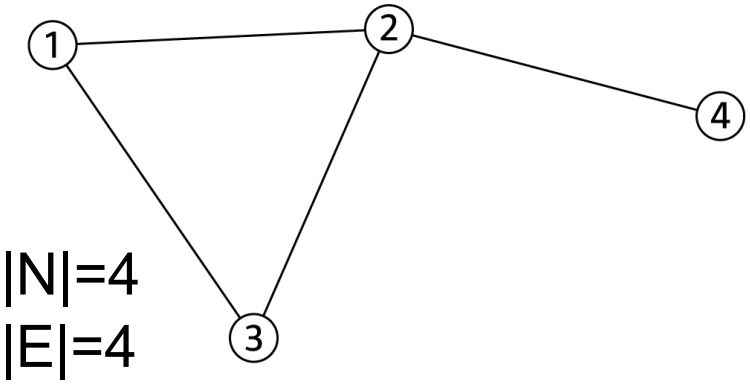
- **Objects:** nodes, vertices — N
- **Interactions:** links, edges — E
- **System:** network, graph — $G(N,E)$

Graphs: A Common Language









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**

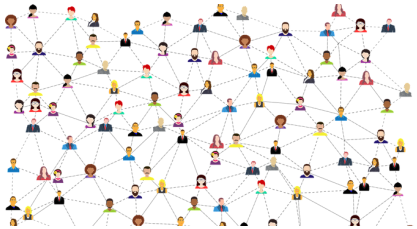


Image credit: Euro Scientists

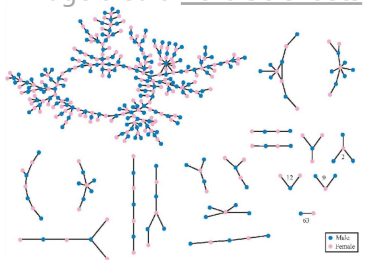
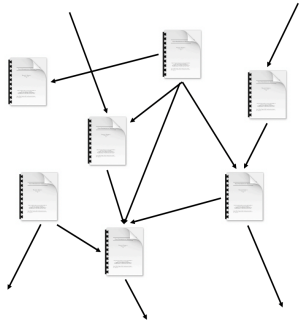


Image credit: ResearchGate



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

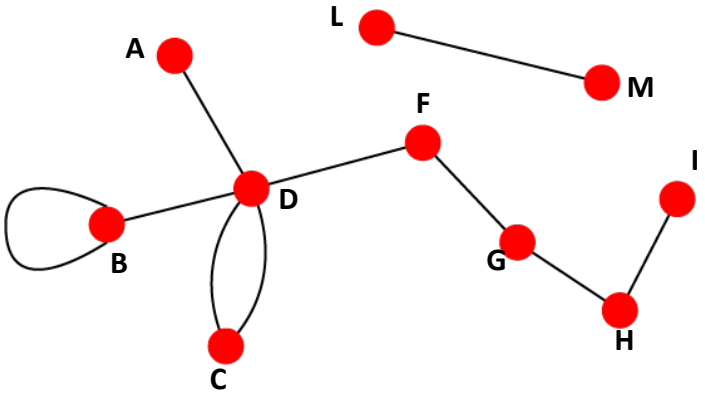
How do you define a graph?

- **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

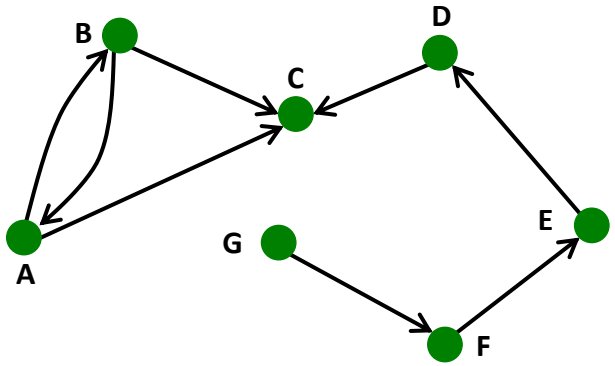
- **Links:** undirected (symmetrical, reciprocal)



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

Directed

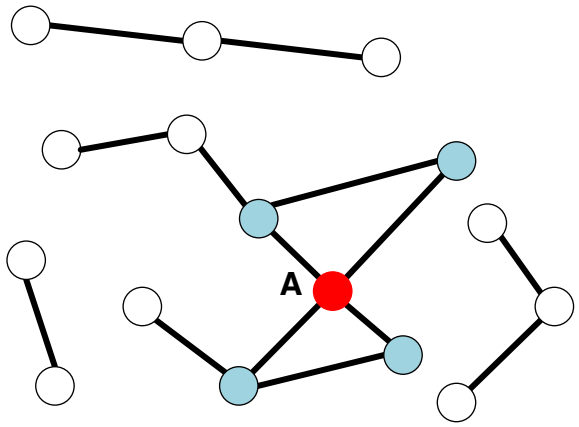
- **Links:** directed (arcs)



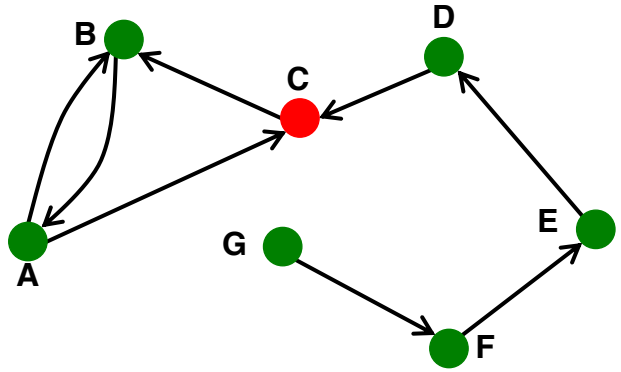
- **Examples:**
  - Phone calls
  - Following on Twitter

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.

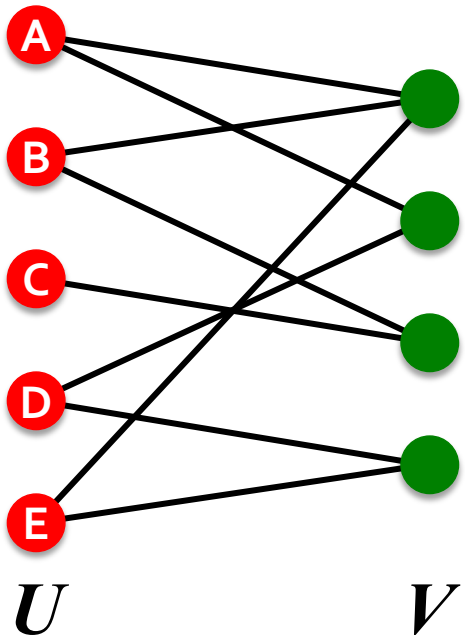
$$k_C^{in} = 2 \qquad k_C^{out} = 1 \qquad k_C = 3$$

$$\bar{k} = \frac{E}{N}$$

$$\overline{k^{in}} = \overline{k^{out}}$$

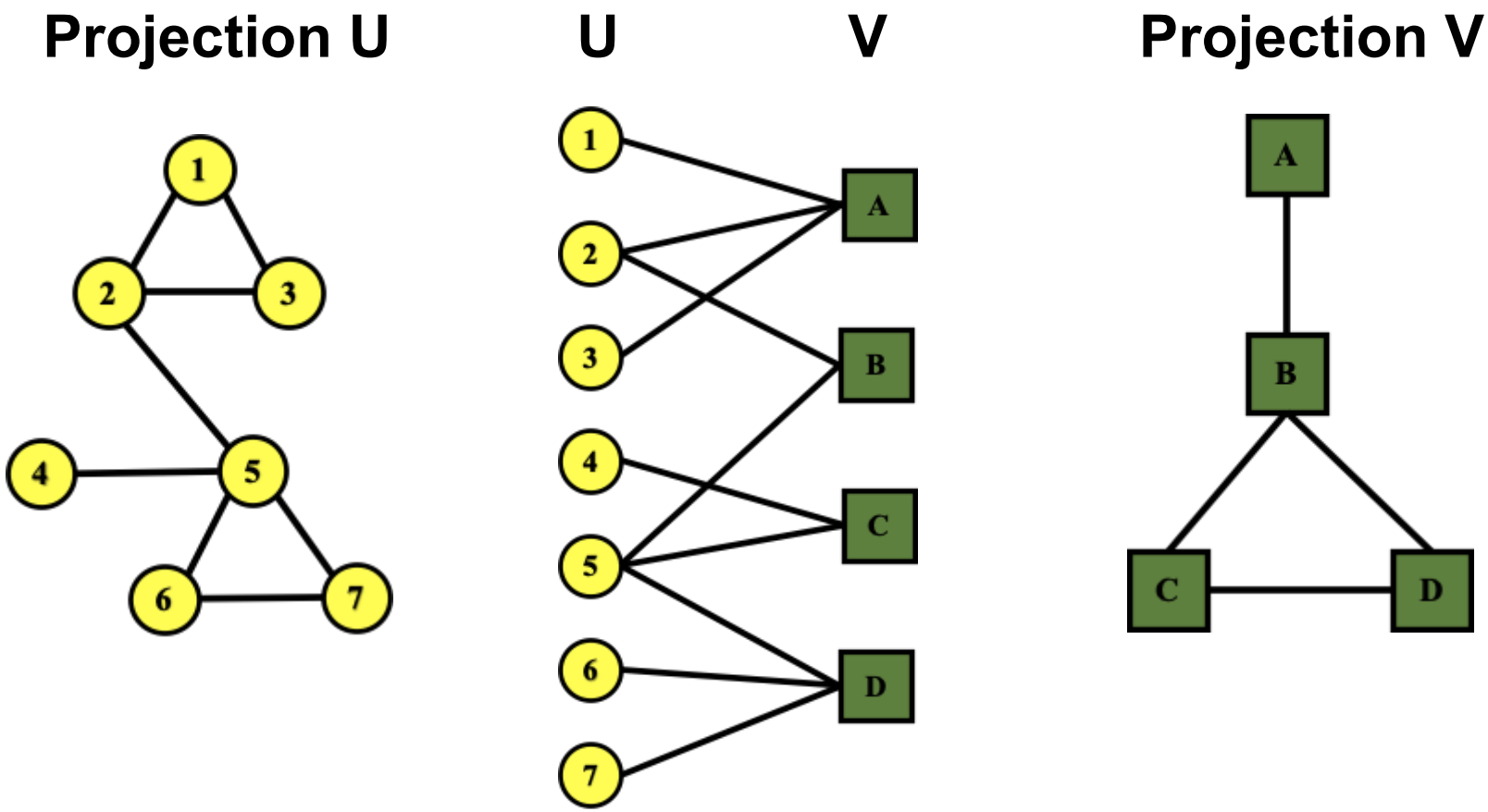
Bipartite Graph

- **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**

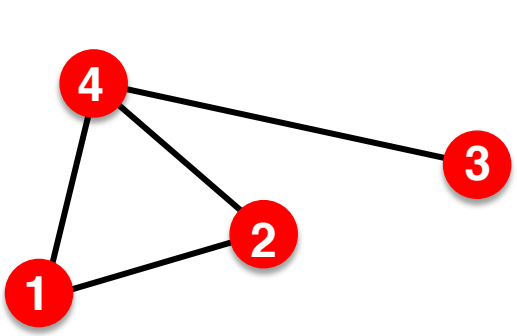


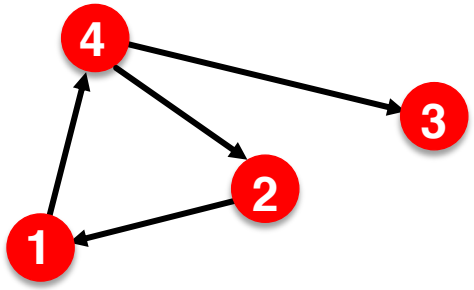
- **Examples:**
  - Authors-to-Papers (they authored)
  - Actors-to-Movies (they appeared in)
  - Users-to-Movies (they rated)
  - Recipes-to-Ingredients (they contain)
- **"Folded" networks:**
  - Author collaboration networks
  - Movie co-rating networks

Folded/Projected Bipartite Graphs



Representing Graphs: Adjacency Matrix





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

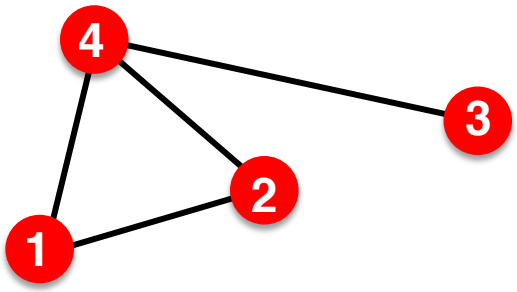
$A_{ij} = 0$ otherwise

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

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

Adjacency Matrix

Undirected



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

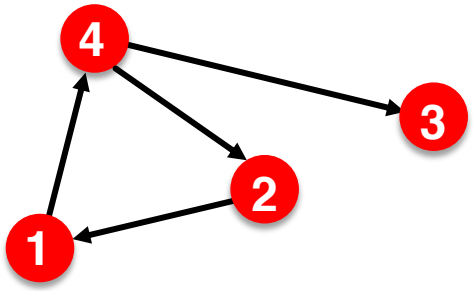
$$A_{ij} = A_{ji}$$
$$A_{ii} = 0$$

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

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

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

Directed



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

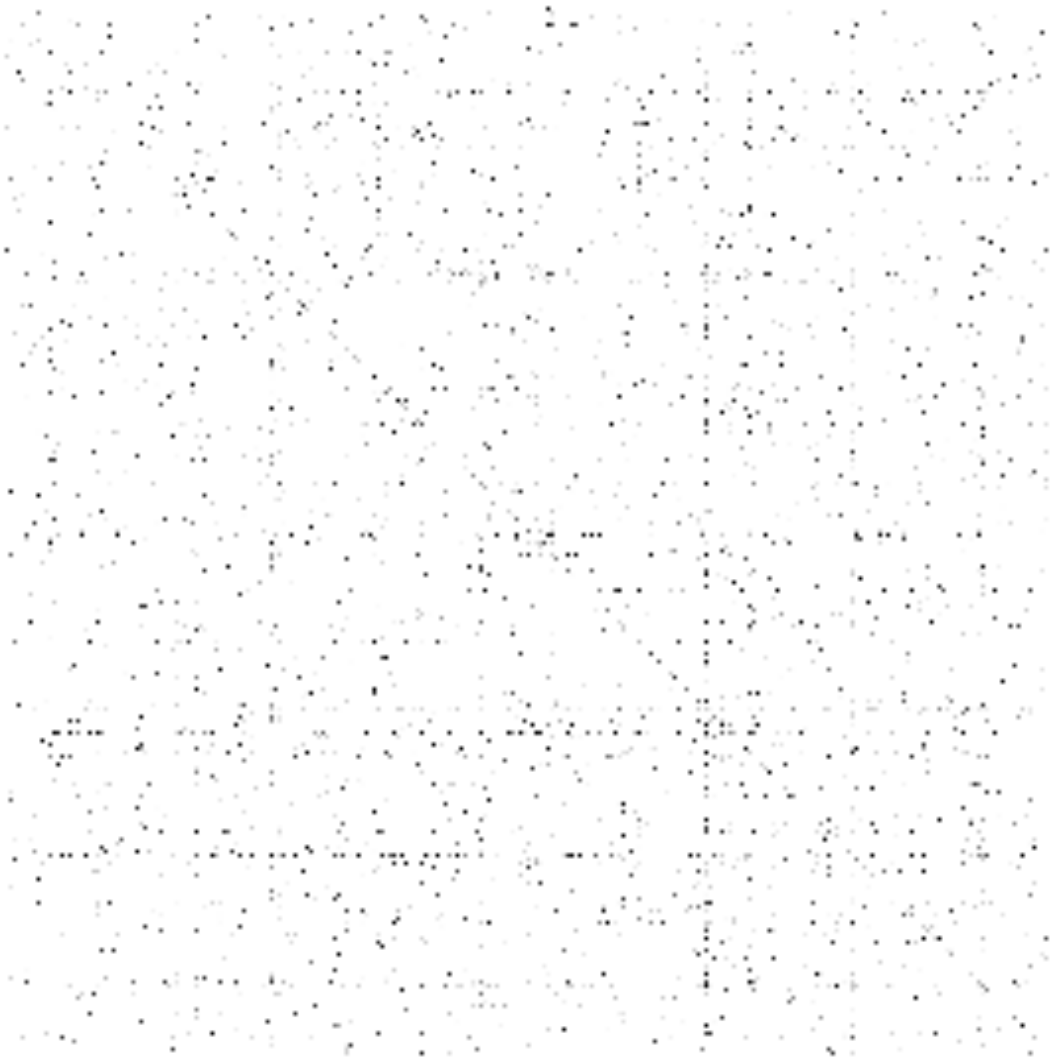
$$A_{ij} \neq A_{ji}$$
$$A_{ii} = 0$$

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

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

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

Adjacency Matrices are Sparse



Networks are Sparse Graphs

Most real-world networks are sparse

$E << E_{max}$ (or $\bar{k} << N-1$)

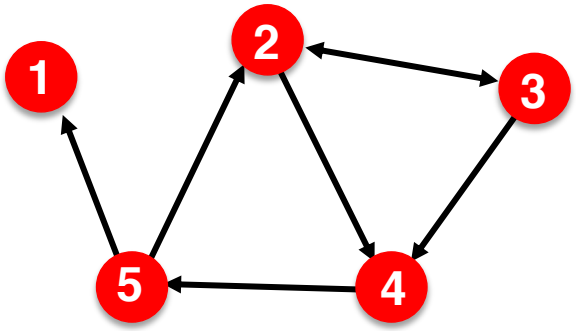
NETWORK	NODES	LINKS	DIRECTED/ UNDIRECTED	N	L	<k>
Internet	Routers	Internet connections	Undirected	192,244	609,066	6.33
WWW	Webpages	Links	Directed	325,729	1,497,134	4.60
Power Grid	Power plants, transformers	Cables	Undirected	4,941	6,594	2.67
Phone Calls	Subscribers	Calls	Directed	36,595	91,826	2.51
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})

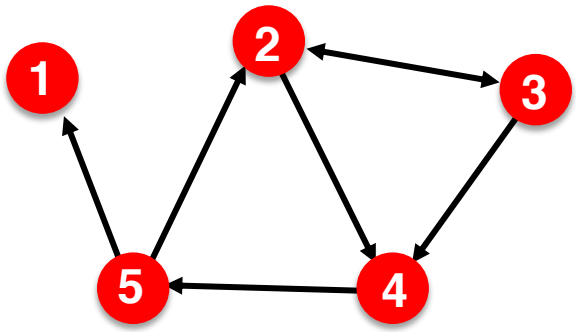
Representing Graphs: Edge list

- **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



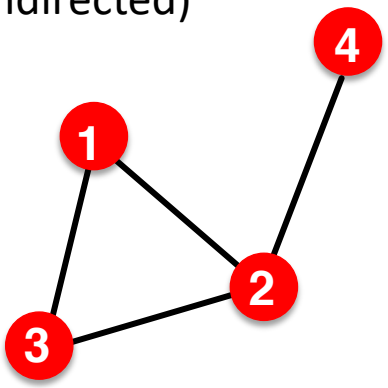
Node and Edge Attributes

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)



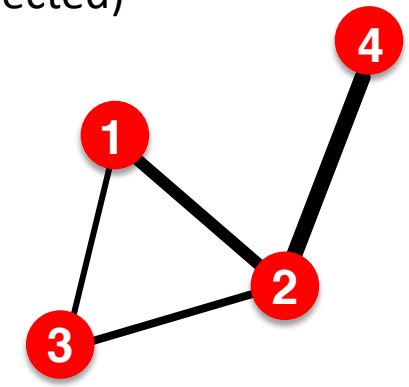
$$A_{ij} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

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

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

Examples: Friendship, Hyperlink

- **Weighted**
  (undirected)



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

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

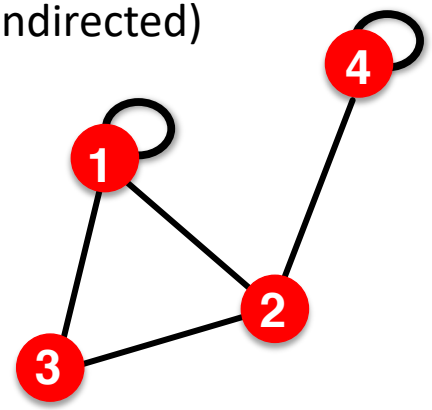
$$E = \frac{1}{2}\sum_{i,j=1}^{N} nonzero(A_{ij}) \qquad \bar{k} = \frac{2E}{N}$$

Examples: Collaboration, Internet, Roads

More Types of Graphs

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

(undirected)



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

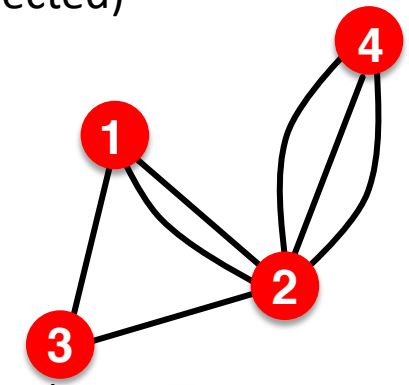
$$A_{ii} \neq 0 \qquad A_{ij} = A_{ji}$$

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

Examples: Proteins, Hyperlinks

- **Multigraph**

(undirected)



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

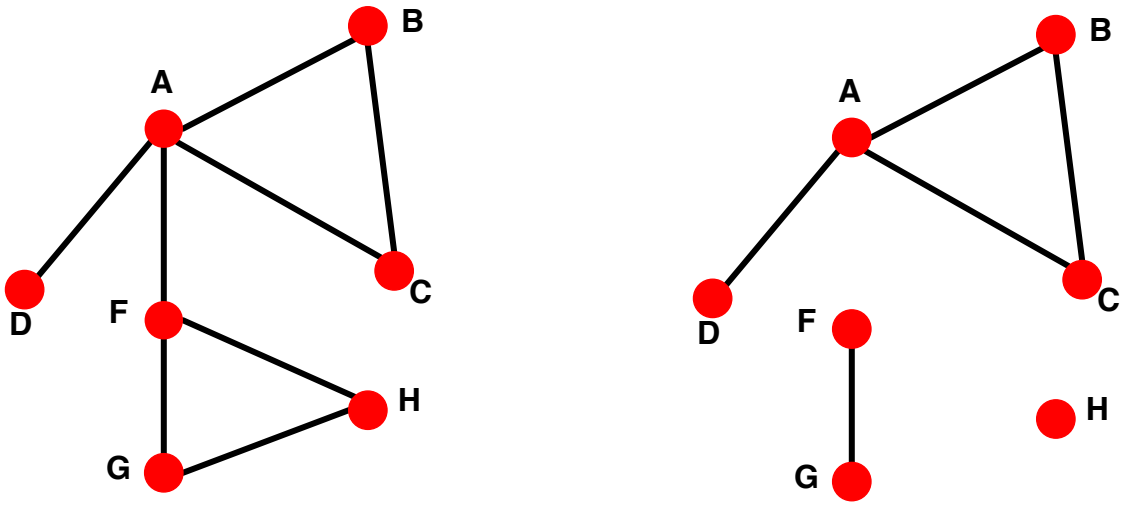
$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

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

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



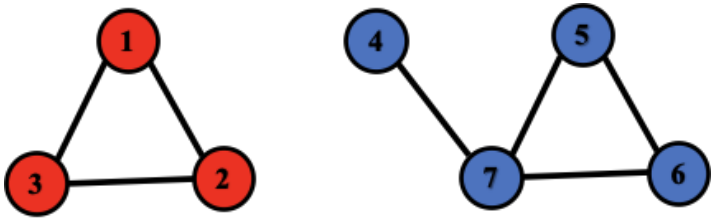
Largest Component:
Giant Component

Isolated node (node H)

Connectivity: Example

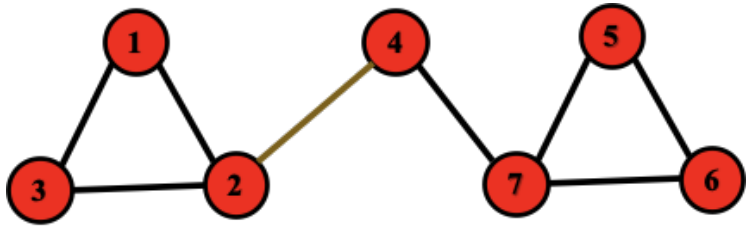
The adjacency matrix of a network with several components can be written in a block- diagonal form, so that nonzero elements are confined to squares, with all other elements being zero:

Disconnected



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

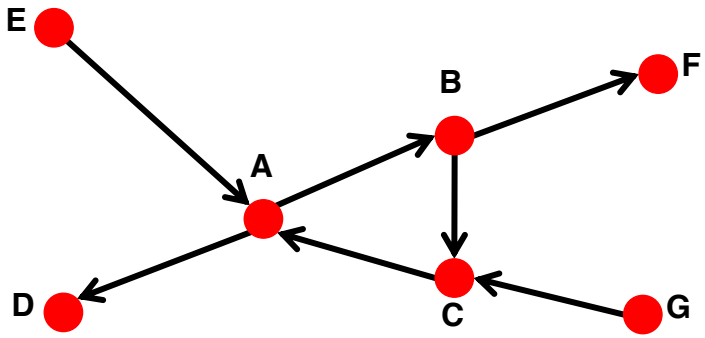
Connected



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

Connectivity of Directed Graphs

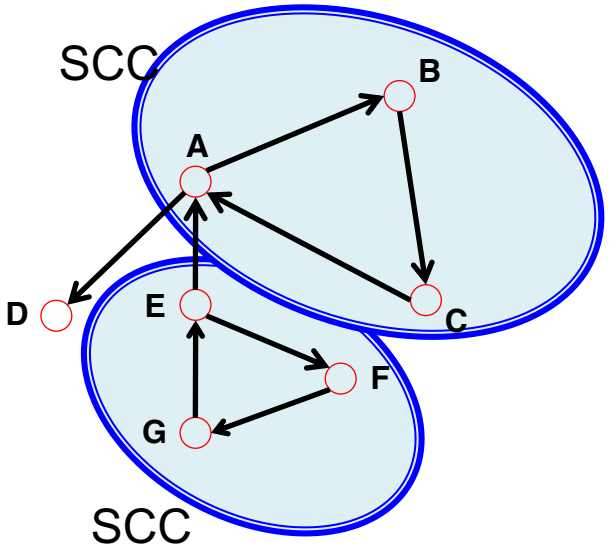
- **Strongly connected directed graph**
  - has a path from each node to every other node and vice versa (e.g., A-B path and B-A path)
- **Weakly connected directed graph**
  - is connected if we disregard the edge directions

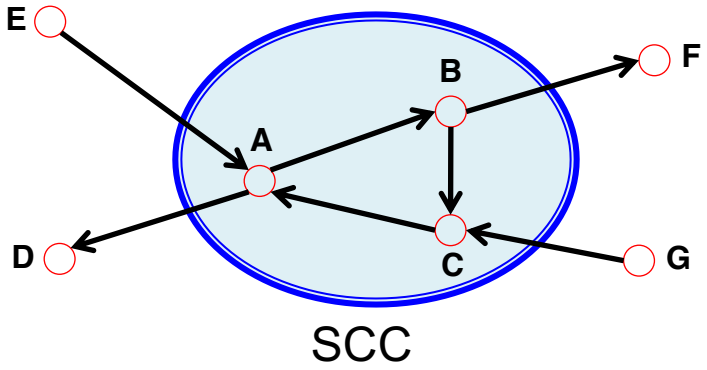


Graph on the left is connected but not strongly connected (e.g., there is no way to get from F to G by following the edge directions).

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