Stanford CS224W: Machine Learning with Graphs

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu
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
<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tue, Jan 12</td>
<td>Introduction; Machine Learning for Graphs</td>
<td>Tue, Feb 16</td>
<td>Reasoning over Knowledge Graphs</td>
</tr>
<tr>
<td>Thu, Jan 14</td>
<td>Traditional Methods for ML on Graphs</td>
<td>Thu, Feb 18</td>
<td>Frequent Subgraph Mining with GNNs</td>
</tr>
<tr>
<td>Tue, Jan 19</td>
<td>Node Embeddings</td>
<td>Tue, Feb 23</td>
<td>Community Structure in Networks</td>
</tr>
<tr>
<td>Thu, Jan 21</td>
<td>Link Analysis: PageRank</td>
<td>Thu, Feb 25</td>
<td>Traditional Generative Models for Graphs</td>
</tr>
<tr>
<td>Tue, Jan 26</td>
<td>Label Propagation for Node Classification</td>
<td>Tue, Mar 2</td>
<td>Deep Generative Models for Graphs</td>
</tr>
<tr>
<td>Thu, Jan 28</td>
<td>Graph Neural Networks 1: GNN Model</td>
<td>Thu, Mar 4</td>
<td>Scaling Up GNNs</td>
</tr>
<tr>
<td>Tue, Feb 2</td>
<td>Graph Neural Networks 2: Design Space</td>
<td>Tue, Mar 9</td>
<td>Learning on Dynamic Graphs</td>
</tr>
<tr>
<td>Thu, Feb 4</td>
<td>Applications of Graph Neural Networks</td>
<td>Thu, Mar 11</td>
<td>GNNs for Computational Biology</td>
</tr>
<tr>
<td>Tue, Feb 9</td>
<td>Theory of Graph Neural Networks</td>
<td>Tue, Mar 16</td>
<td>GNNs for Science</td>
</tr>
<tr>
<td>Thu, Feb 11</td>
<td>Knowledge Graph Embeddings</td>
<td>Thu, Mar 18</td>
<td>Industrial Applications of GNNs</td>
</tr>
</tbody>
</table>
Logistics: Website

- [http://cs224w.stanford.edu](http://cs224w.stanford.edu)
  - Slides posted before the class
- **Readings:**
  - [Graph Representation Learning Book](http://cs224w.stanford.edu) 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:**
    - 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](mailto:cs224w-win2021-staff@lists.stanford.edu)

- We will post course announcements to Piazza (make sure you check it regularly)
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)
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](http://cs224w.stanford.edu/info.html)
<table>
<thead>
<tr>
<th>Week</th>
<th>Assignment</th>
<th>Due on (11:59pm PT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Colab 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Colab 1</td>
<td>Thu, Jan 28</td>
</tr>
<tr>
<td>2</td>
<td>Homework 1</td>
<td>Thu, Feb 4</td>
</tr>
<tr>
<td>3</td>
<td>Colab 2</td>
<td>Thu, Feb 11</td>
</tr>
<tr>
<td></td>
<td>Project Proposal</td>
<td>Thu, Feb 11</td>
</tr>
<tr>
<td>4</td>
<td>Homework 2</td>
<td>Thu, Feb 18</td>
</tr>
<tr>
<td>5</td>
<td>Colab 3</td>
<td>Thu, Feb 25</td>
</tr>
<tr>
<td>6</td>
<td>Homework 3</td>
<td>Thu, Mar 4</td>
</tr>
<tr>
<td>7</td>
<td>Colab 4</td>
<td>Thu, Mar 11</td>
</tr>
<tr>
<td>8</td>
<td>Colab 5</td>
<td>Thu, Mar 18</td>
</tr>
<tr>
<td></td>
<td>Project Report</td>
<td>Sun, Mar 21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(No Late Periods!)</td>
</tr>
</tbody>
</table>
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
Why Graphs?

Graphs are a general language for describing and analyzing entities with relations/interactions.
Many Types of Data are Graphs (1)

- **Event Graphs**
- **Computer Networks**
- **Disease Pathways**
  - Image credit: SalientNetworks

- **Food Webs**

- **Particle Networks**
  - Image credit: Pinterest

- **Underground Networks**
  - Image credit: visitlondon.com

Jure Leskovec, Stanford CS224W: Machine Learning with Graphs

1/13/21
Many Types of Data are Graphs (2)

- Social Networks
- Economic Networks
- Communication Networks
- Citation Networks
- Internet
- Networks of Neurons
Many Types of Data are Graphs (3)

Knowledge Graphs
- Spock played characterIn Star Trek
- Science Fiction
- Leonard Nimoy
- Star Wars
- Alec Guinness

Image credit: Maximilian Nickel et al

Regulatory Networks

Image credit: ese.wustl.edu

Scene Graphs

Image credit: math.hws.edu

Code Graphs

Image credit: ResearchGate

Molecules

Image credit: MDPI

3D Shapes

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
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

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

Knowledge Graphs
Image credit: Maximilian Nickel et al

3D Shapes

Code Graphs
Image credit: ResearchGate

Molecules
Image credit: MDPI

Scene Graphs
Image credit: math.hws.edu

Regulatory Networks
Image credit: ese.wustl.edu

1/13/21
Complex domains have a rich relational structure, which can be represented as a relational graph.

By explicitly modeling relationships we achieve better performance!
Modern deep learning toolbox is designed for simple sequences & grids
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
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**

**Input:** Network

**Predictions:** Node labels, New links, Generated graphs and subgraphs
(Supervised) Machine Learning Lifecycle: This feature, that feature. Every single time!
Map nodes to d-dimensional embeddings such that similar nodes in the network are embedded close together.

Learn a neural network

\[ f: u \rightarrow \mathbb{R}^d \]

Feature representation, embedding
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

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

Jure Leskovec, Stanford CS224W: Machine Learning with Graphs
Stanford CS224W: Applications of Graph ML
Different Types of Tasks

- **Node level**
- **Community (subgraph) level**
- **Edge-level**

**Graph-level**
- prediction,
- Graph generation
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
  - **Example**: Knowledge graph completion

- **Graph classification**: Categorize different graphs
  - **Example**: Molecule property prediction

- **Clustering**: Detect if nodes form a community
  - **Example**: Social circle detection

- **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**

![Protein structures](image)

- **T1037 / 6vr4**
  - 90.7 GDT
  - (RNA polymerase domain)

- **T1049 / 6y4f**
  - 93.3 GDT
  - (adhesin tip)

Image credit: DeepMind
AlphaFold: Impact

Medan Free-Modelling Accuracy

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?

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)
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
**Task:** Recommend related pins to users

Query pin

**Task:** Learn node embeddings $z_i$ such that

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

**Predict whether two nodes in a graph are related**
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

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

**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
Traffic Prediction via GNN

Predict via Graph Neural Networks

- Used in Google Maps

Image credit: DeepMind
Examples of Graph-level ML Tasks
Antibiotics are small molecular graphs

- **Nodes:** Atoms
- **Edges:** Chemical bonds


Image credit: CNN
A Graph Neural Network graph classification model
Predict promising molecules from a pool of candidates

### 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

---

**Diagram:**

- **State:** $G_t$
- **Scaffold:** $C$
- **GCPN:** $\pi_G(a_t | G_t \cup C)$
- **Action:** $a_t \sim \pi_G$
- **Dynamics:** $p(G_{t+1} | G_t, a_t)$
- **State:** $G_{t+1}$
- **Reward:** $r_t$

---

**Table: Use case 1**

<table>
<thead>
<tr>
<th>Molecule</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Molecule 1" /></td>
<td>0.948</td>
</tr>
<tr>
<td><img src="image2" alt="Molecule 2" /></td>
<td>0.945</td>
</tr>
<tr>
<td><img src="image3" alt="Molecule 3" /></td>
<td>0.944</td>
</tr>
<tr>
<td><img src="image4" alt="Molecule 4" /></td>
<td>0.941</td>
</tr>
</tbody>
</table>

**Table: Use case 2**

<table>
<thead>
<tr>
<th>Molecule</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image5" alt="Molecule 1" /></td>
<td>-8.32</td>
</tr>
<tr>
<td><img src="image6" alt="Molecule 2" /></td>
<td>-5.55</td>
</tr>
<tr>
<td><img src="image7" alt="Molecule 3" /></td>
<td>-0.71</td>
</tr>
<tr>
<td><img src="image8" alt="Molecule 4" /></td>
<td>-1.78</td>
</tr>
</tbody>
</table>
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
A graph evolution task:

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

![Graph Evolution Diagram]

Sanchez-Gonzalez et al., *Learning to simulate complex physics with graph networks*, ICML 2020
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
- **Interactions**: links, edges
- **System**: network, graph

$$G(N,E)$$
Graphs: A Common Language

Actors and Movies:
- Actor 1
- Actor 2
- Actor 3
- Actor 4
- Peter
- Mary
- Albert
- Tom

Movies:
- Movie 1
- Movie 2
- Movie 3

Proteins:
- Protein 1
- Protein 2
- Protein 5
- Protein 9

Graph Properties:
- $|N|=4$
- $|E|=4$
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
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)

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

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

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

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

\[
k_A = 4
\]

**Avg. degree:**

\[
\overline{k} = \langle k \rangle = \frac{1}{N} \sum_{i=1}^{N} k_i = \frac{2E}{N}
\]

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.

\[
k_C^{in} = 2 \quad k_C^{out} = 1 \quad k_C = 3
\]

**Source:** Node with \( k^{in} = 0 \)

**Sink:** Node with \( k^{out} = 0 \)

\[
\overline{k} = \frac{E}{N} \quad \overline{k^{in}} = \overline{k^{out}}
\]
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

Projection U

Projection V

U

V

1

A

2

B

3

C

4

D

5

6

7

Projection V

A

B

C

D
Representing Graphs: Adjacency Matrix

\[ A_{ij} = 1 \quad \text{if there is a link from node } i \text{ to node } j \]
\[ A_{ij} = 0 \quad \text{otherwise} \]

\[
A = \begin{pmatrix}
0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 \\
\end{pmatrix}
\]

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

Undirected

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

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

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

\[L = \frac{1}{2} \sum_{i=1}^{N} k_i = \frac{1}{2} \sum_{i,j}^{N} A_{ij}\]

Directed

\[
A = \begin{pmatrix}
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0
\end{pmatrix}
\]

\[k_{out}^{\text{out}} = \sum_{j=1}^{N} A_{ij}\]

\[k_{in}^{\text{in}} = \sum_{i=1}^{N} A_{ij}\]

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

\[A_{ij} \neq A_{ji}\]

\[A_{ii} = 0\]
Adjacency Matrices are Sparse
Most real-world networks are **sparse**

\[ E \ll E_{\text{max}} \text{ (or } k \ll N-1) \]

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

**Consequence:** Adjacency matrix is filled with zeros!

(Density of the matrix \( \frac{E}{N^2} \): WWW=1.51\times10^{-5}, MSN IM = 2.27\times10^{-8})
Represent graph as a list of edges:

- (2, 3)
- (2, 4)
- (3, 2)
- (3, 4)
- (4, 5)
- (5, 2)
- (5, 1)
**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)

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

\[A_{ii} = 0 \quad A_{ij} = A_{ji}\]

\[E = \frac{1}{2} \sum_{i,j=1}^{N} A_{ij} \quad \bar{k} = \frac{2E}{N}\]

**Examples:** Friendship, Hyperlink

- **Weighted** (undirected)

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

\[A_{ii} = 0 \quad A_{ij} = A_{ji}\]

\[E = \frac{1}{2} \sum_{i,j=1}^{N} \text{nonzero}(A_{ij}) \quad \bar{k} = \frac{2E}{N}\]

**Examples:** Collaboration, Internet, Roads
More Types of Graphs

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

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

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

\[A_{ii} \neq 0 \quad A_{ij} = A_{ji}\]

**Examples:** Proteins, Hyperlinks

- **Multigraph**
  (undirected)

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

\[E = \frac{1}{2} \sum_{i,j=1}^{N} \text{nonzero}(A_{ij})\]

\[\bar{k} = \frac{2E}{N}\]

**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

![Graph Diagram]

- 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:
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).
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.
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