Relationship Modeling: Graph Neural Networks (GNNs)

Ju Sun
Computer Science & Engineering

Apr 10, 2025

Outline

- Why graphs?
- Graphs: basic notions
- Graph neural networks
- Scaling up training

Why graphs?

Graphs are everywhere!

Molecules

Knowledge
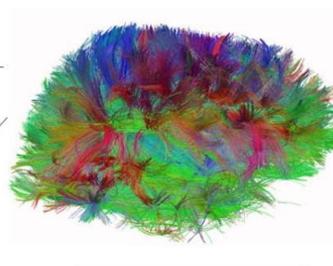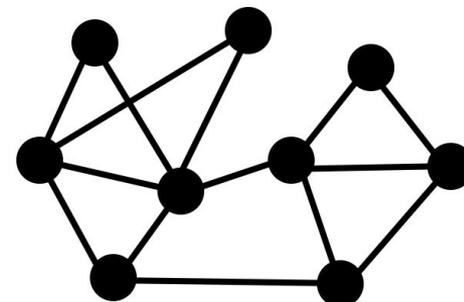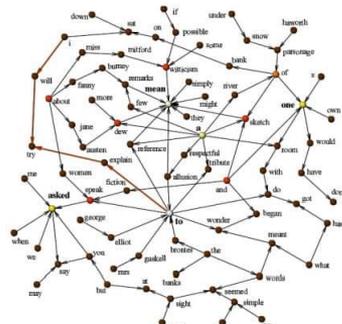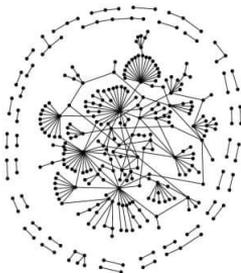
Information

Brain/neurons

Genes

Communication

Software

Social

Image credit: Stanford CS224W

Graphs model **relationships/ interactions**

Image credit: https://blogs.nvidia.com/blog/2022/10/24/what-are-graph-neural-networks/

Different tasks on graphs

The whole graph is all available data

The whole graph is part of all available data, i.e., a data point

Node level → **Node prediction (classification/regression)** given labels over part of the graph

Graph-level prediction, Graph generation

Community (subgraph) level → **Community detection/clustering** Clustering nodes/edges

Edge-level → **Link prediction** Predict links should exists or not

Graph prediction (classification/regression) , graph generation A graph is a data point

Image credit: Stanford CS224W

Example task 1: Protein folding

Image credit: Deep Mind

1	20	100's	20,000	200,000,000
AMINO ACID	AMINO ACIDS IN A STRING	AMINO ACIDS IN A PROTEIN	PROTEINS IN THE HUMAN BODY	DISTINCTIVE PROTEINS FOUNDS ON EARTH

AlphaFold DB today
200M+ Structures

**AlphaFold DB previously**
~1M Structures

Experimental (PDB) today
190K Structures
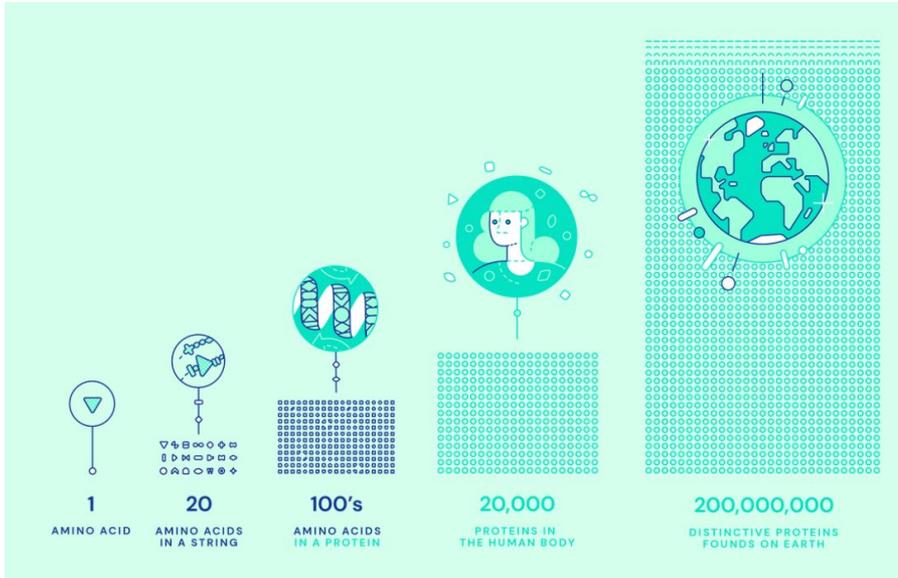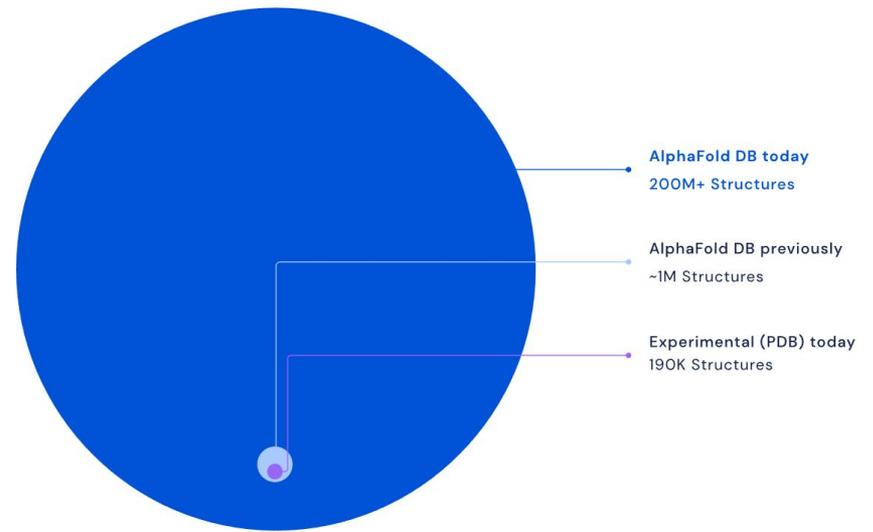
Number of species represented in AlphaFold DB
Total increase from ~10K to ~1M

- Today
- Previously

Animals Plants Bacteria Fungi Other

Example task 1: Protein folding

AlphaFold

Key concept: **Spatial graph**

Complete story: https://www.deepmind.com/research/highlighted-research/alphafold
Paper: https://www.nature.com/articles/s41586-021-03819-2
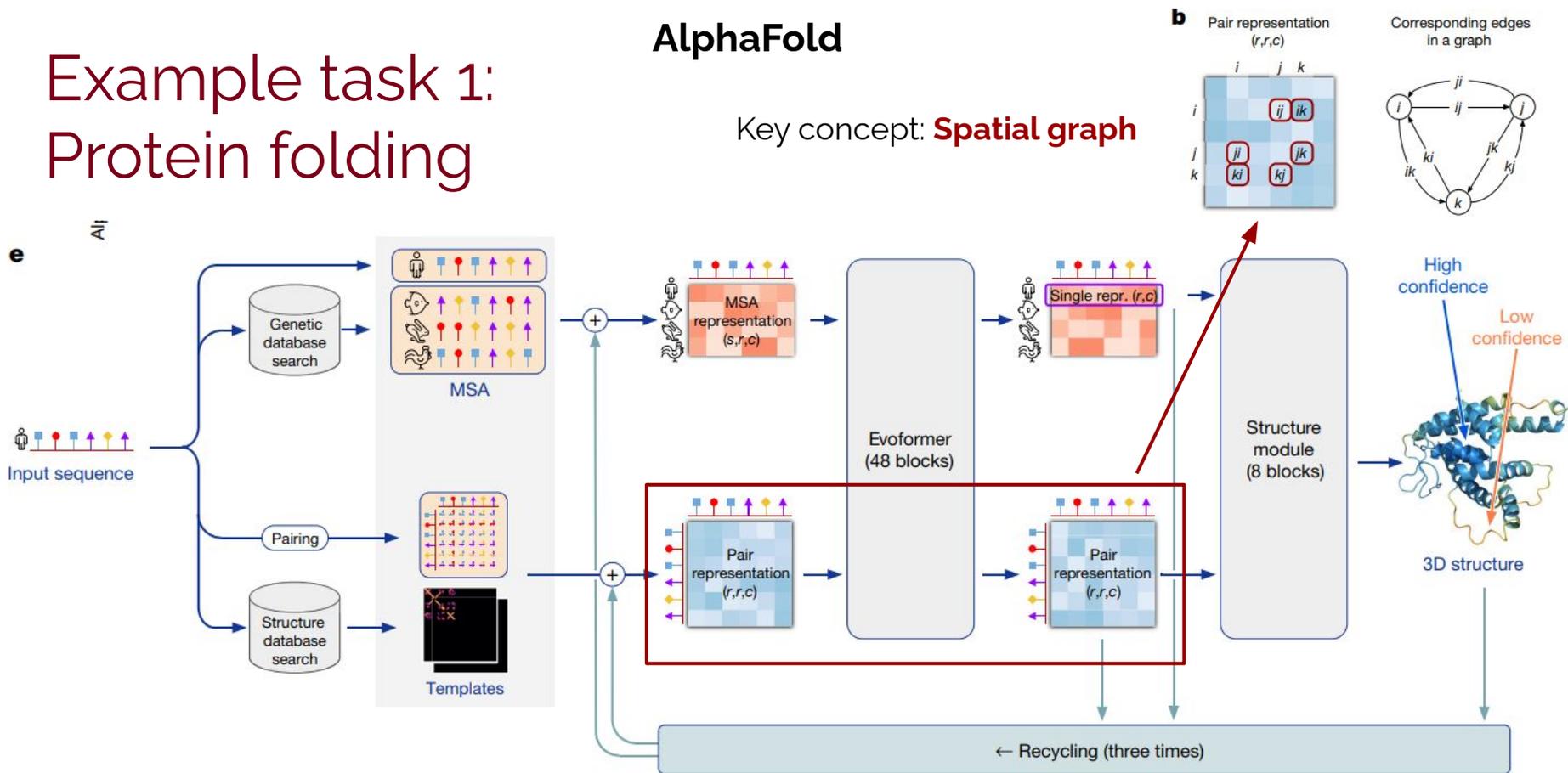
THE NOBEL PRIZE IN CHEMISTRY 2024

Illustrations: Niklas Elmehed

David Baker
"for computational protein design"

Demis Hassabis
John M. Jumper
"for protein structure prediction"

THE ROYAL SWEDISH ACADEMY OF SCIENCES

nature

Explore content ∨   About the journal ∨   Publish with us ∨   Subscribe

nature > news > article

NEWS | 27 March 2025

AlphaFold is running out of data — so drug firms are building their own version

Thousands of 3D protein structures locked up in big-pharma vaults will be used to create a new AI tool that won't be open to academics.

By Ewen Callaway
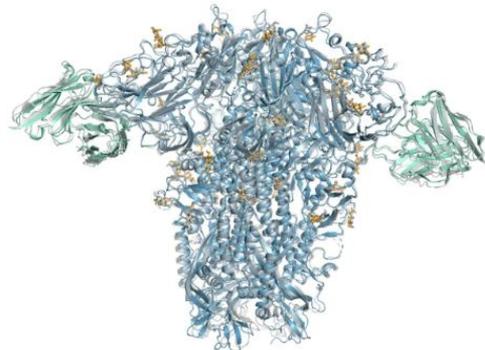
An AlphaFold 3 model of a common cold spike protein (blue) interacting with antibodies (green). Credit: Google DeepMind

Example task 2: Recommendation systems
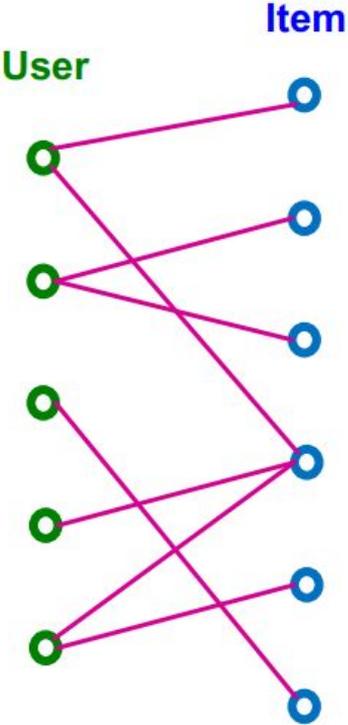
online shopping,
music/movie
recommendation

Nodes:
Users, items
**Edges:**
User-item
interactions

Image credit: Stanford CS224W

Image credit: Stanford CS224W

9

Example task 3: Drug adverse effect discovery

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

△ Drug ⬤ Protein
r_1 Gastrointestinal bleed side effect △—⬤ Drug-protein interaction
r_2 Bradycardia side effect ⬤—⬤ Protein-protein interaction
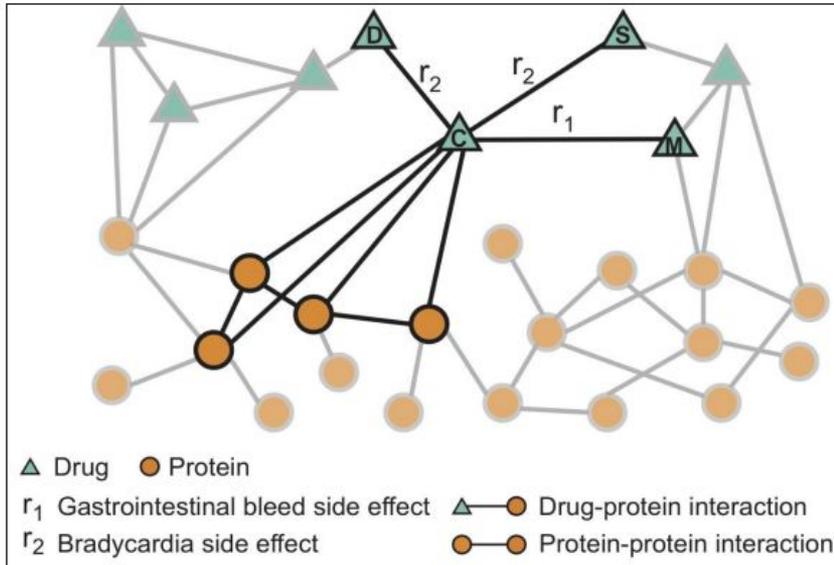
Query: How likely will Simvastatin and Ciprofloxacin, when taken together, break down muscle tissue?

Image credit: Stanford CS224W

10

Example task 4: Traffic prediction

Example task 4: Traffic prediction

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

Image credit: DeepMind
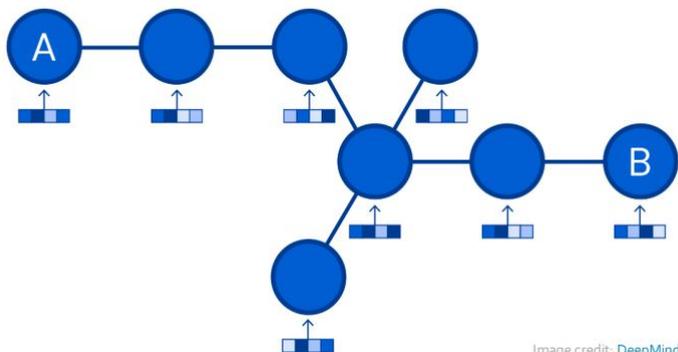
Image credit: Stanford CS224W

Predicting Time of Arrival with Graph Neural Networks

- Used in Google Maps

THE MODEL ARCHITECTURE FOR DETERMINING OPTIMAL ROUTES AND THEIR TRAVEL TIME

Image credit: DeepMind
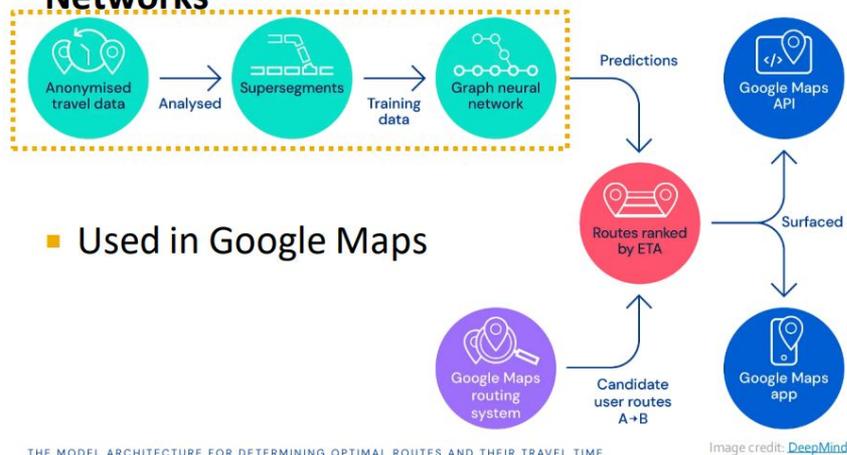
Image credit: Stanford CS224W

Subgraph discovery

Example task 5: Drug discovery

Image credit: https://distill.pub/2021/gnn-intro/
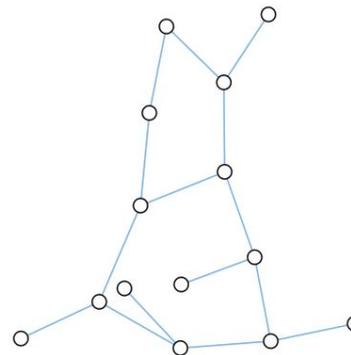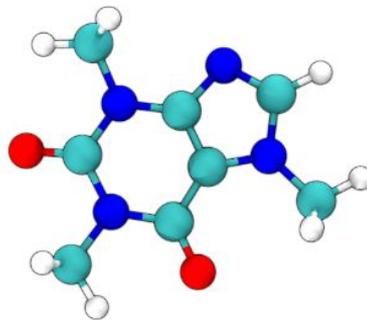
Molecules as graphs:
Nodes: atoms   **Edges**: chemical bounds
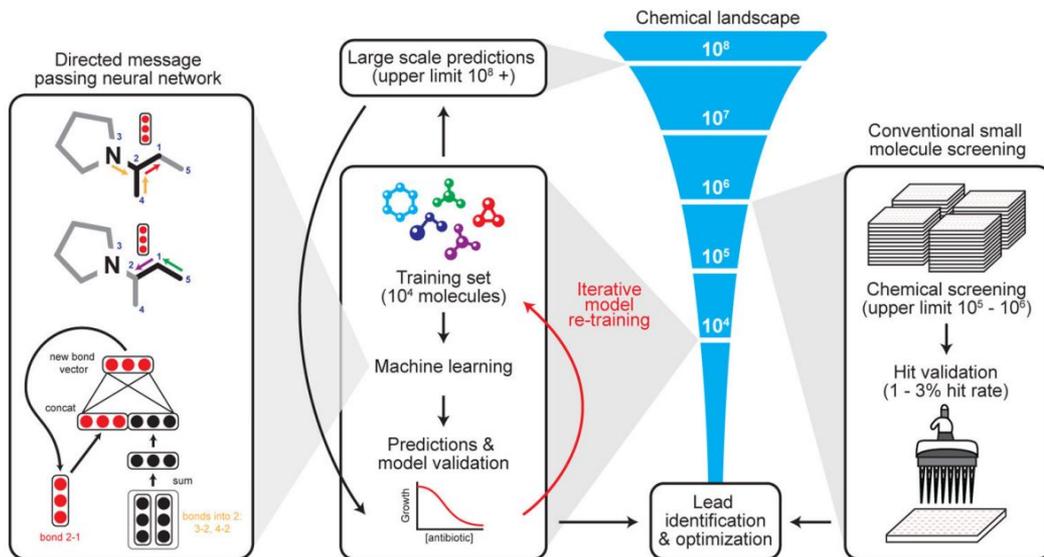
full-graph prediction

13

Graphs: basic notions

Basic objects
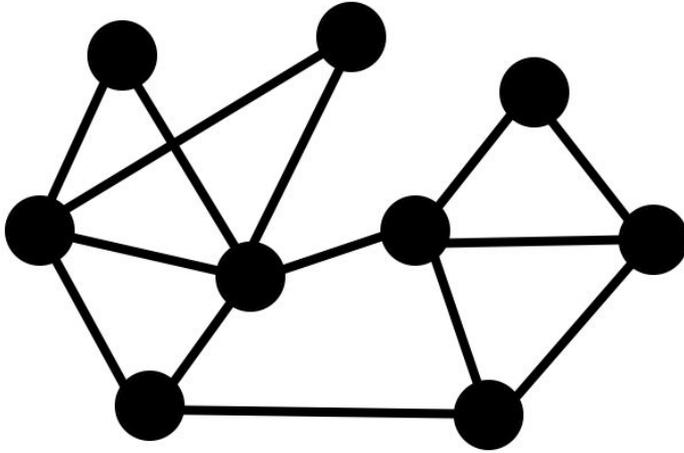
- N : Nodes (also vertices)
- E : Edges (also links)
- $G(N, E)$: Graph

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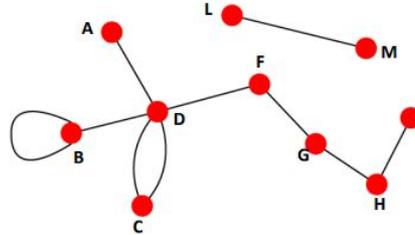

- **Examples:**
  - Collaborations
  - Friendship on Facebook

Directed
- **Links:** directed (arcs)
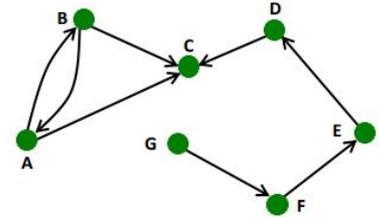
- **Examples:**
  - Phone calls
  - Following on Twitter

Image credit: Stanford CS224W

Heterogeneous graphs

Nodes/Edges are multi-typed

$$G(N, E, T, R)$$

T: types of nodes
R: types of relationships
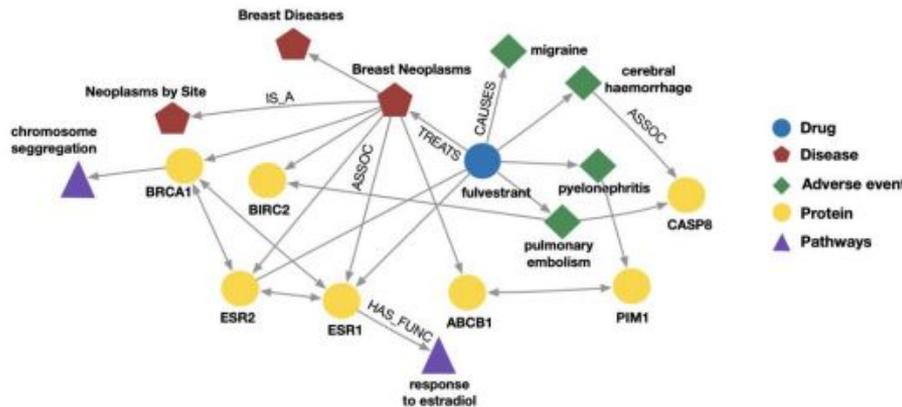
Biomedical Knowledge Graphs

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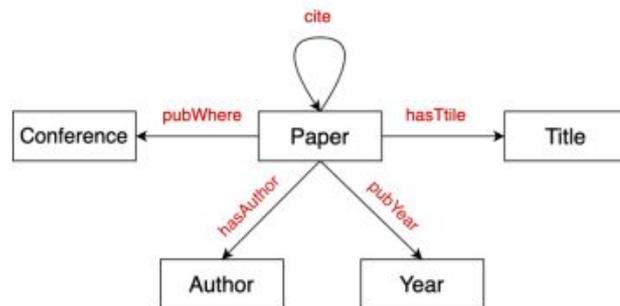
Academic Graphs

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

16

Image credit: Stanford CS224W

Graph representation

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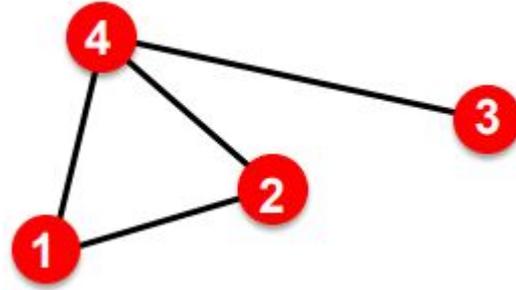
$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

(1, 4), (1, 2)
(2, 1), (2, 4)
(3, 4)
(4, 1), (4, 2), (4, 3)

1: 2, 4
2: 1, 4
3: 4
4: 1, 2, 3

Adjacency matrix   **Edge list**   **Adjacency list**

Graph representation

Directed graphs

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

(1, 4)
(2, 1)

(4, 2), (4, 3)

1: 4
2: 1
3:
4: 2, 3
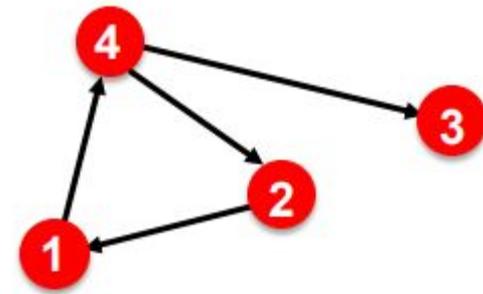
Adjacency matrix   **Edge list**   **Adjacency list**

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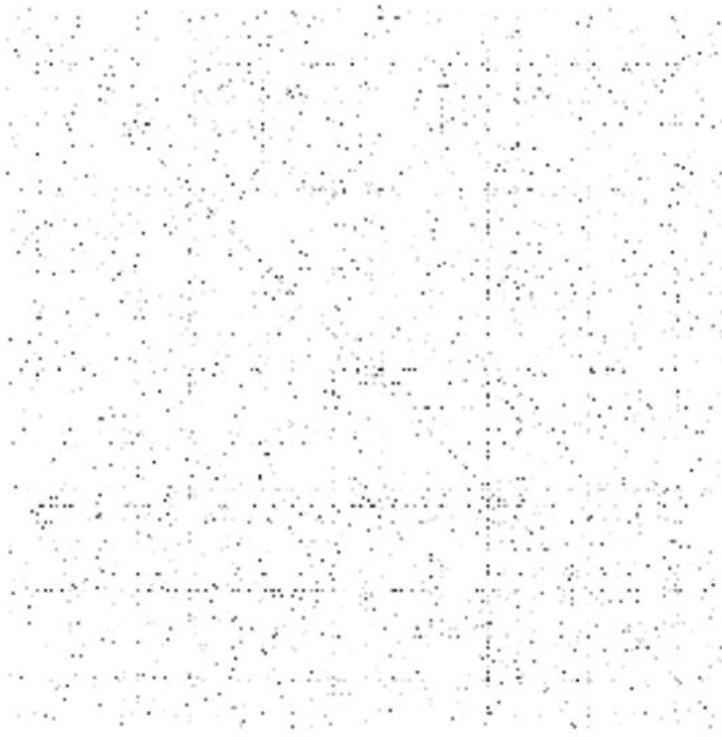
| NETWORK	NODES	LINKS	DIRECTED/ UNDIRECTED	N	E
Internet	Routers	Internet connections	Undirected	192,244	609,066
WWW	Webpages	Links	Directed	325,729	1,497,134
Power Grid	Power plants, transformers	Cables	Undirected	4,941	6,594
Phone Calls	Subscribers	Calls	Directed	36,595	91,826
Email	Email Addresses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

Density = $|E|/|N|^2$

Real-world graphs are often very sparse

19

Weighted graphs

Unweighted (undirected)
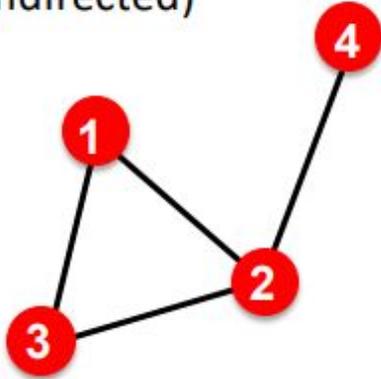
$$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Weighted (undirected)
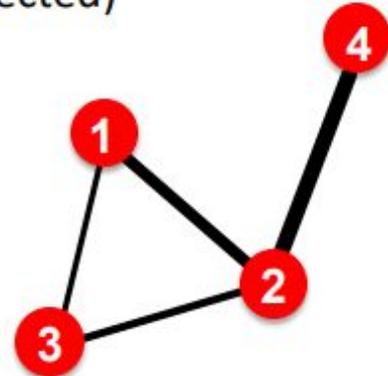
$$\begin{pmatrix} 0 & 2 & 0.5 & 0 \\ 2 & 0 & 1 & 4 \\ 0.5 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 \end{pmatrix}$$

Image credit: Stanford CS224W

Graph isomorphism/equivalence
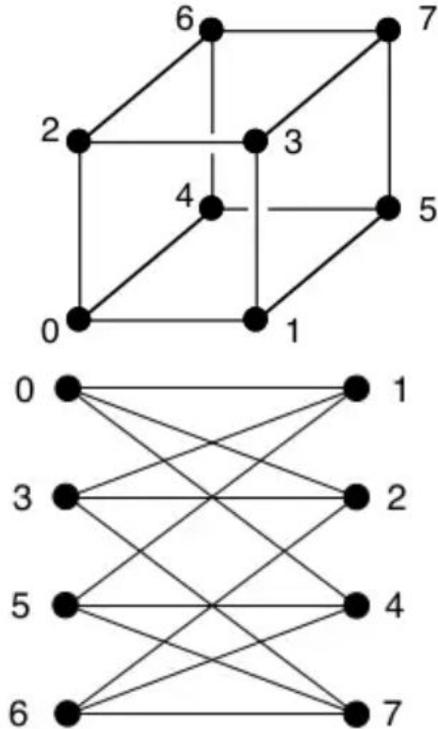
Isomorphism: there exists a bijective mapping, i.e.,
permutation, results in the same neighborhood structure

21

Permutation invariance
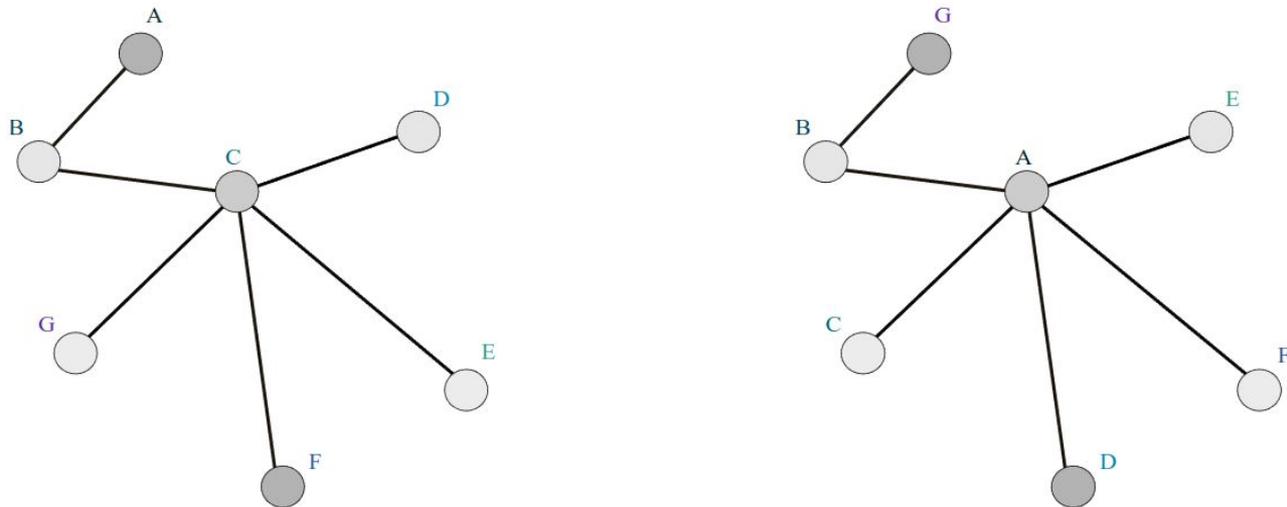
Image credit: Image credit: https://distill.pub/2021/understanding-gnns/

Permutation invariance: permuting the names of the nodes doesn't change the graph, as graph nodes are intrinsically orderless

Mathematically: if A is (the adjacency matrix of) a graph, $\Pi A \Pi^{\mathsf{T}}$ is (the adjacency matrix of) an equivalent graph for **any permutation matrix Π**

Graph neural networks (GNNs)

Representation learning for graphs

traditional learning pipeline

Data → Feature Extraction → Learning Models → Decision
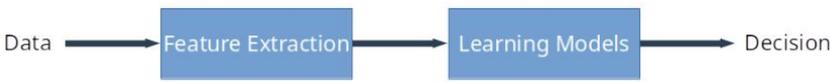
modern learning pipeline
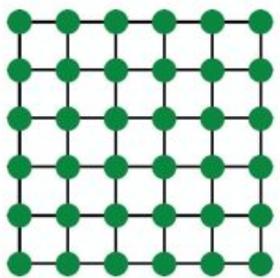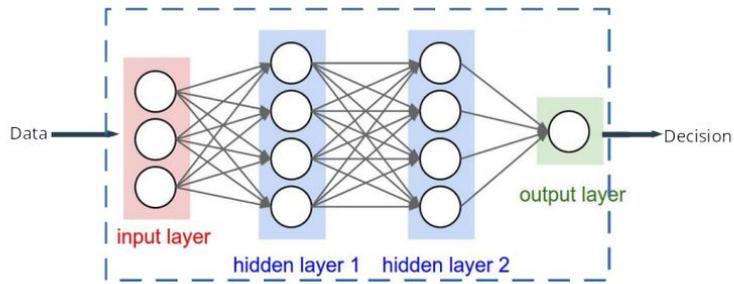
Data → Feature Extraction & Learning Models → Decision

Data → input layer → hidden layer 1 → hidden layer 2 → output layer → Decision

Grid
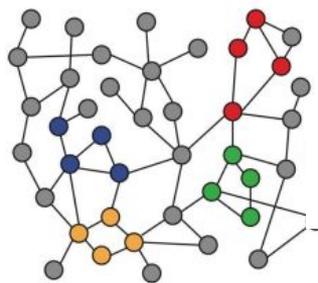
image

video

audio (spectrogram)

time series

$\sim 10^6$

$\sim 10^8 (10s)$

$\sim 10^8 (10s)$

$1/\text{resol}$

List

NLP

natural language processing
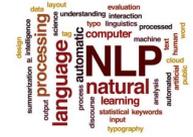
− machine translation, e.g., English ⇆ Chinese

− typing/writing prediction (smart compose)

− semantic classification

Graph

Molecules Knowledge Information Brain/neurons

Genes Communication Software Social

24

Where to put the features?

Link-level ?
Node-level ?
Graph-level ?

$\in \mathbb{R}^D$

Link features
Node features
Graph features
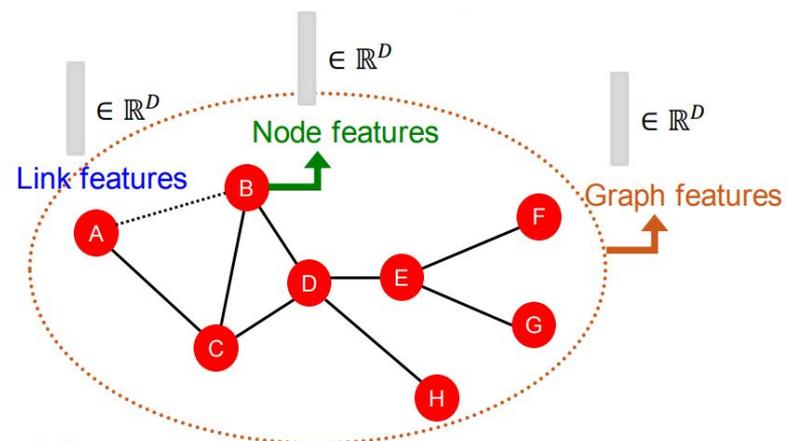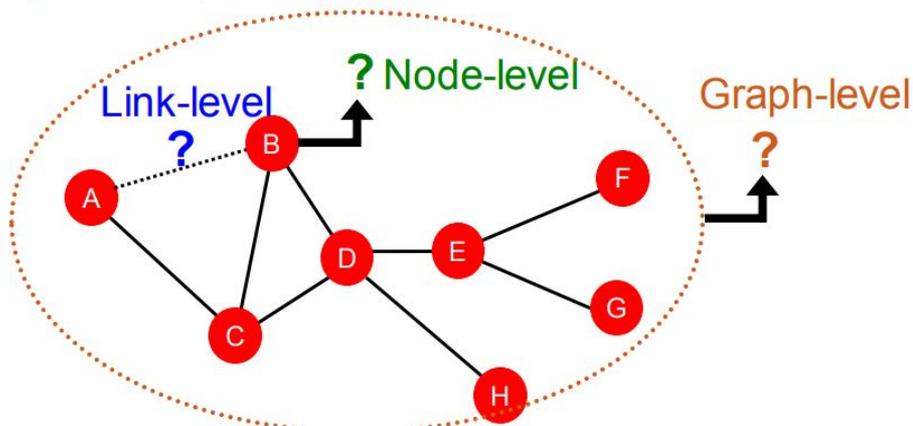
- **Train an ML model:**
  - Random forest
  - SVM
  - Neural network, etc.

$x_1 \rightarrow y_1$

$\vdots$

$x_N \rightarrow y_N$

- **Apply the model:**
  - Given a new node/link/graph, obtain its features and make a prediction

$x \rightarrow y$

Image credit: Stanford CS224W

25

Node embedding
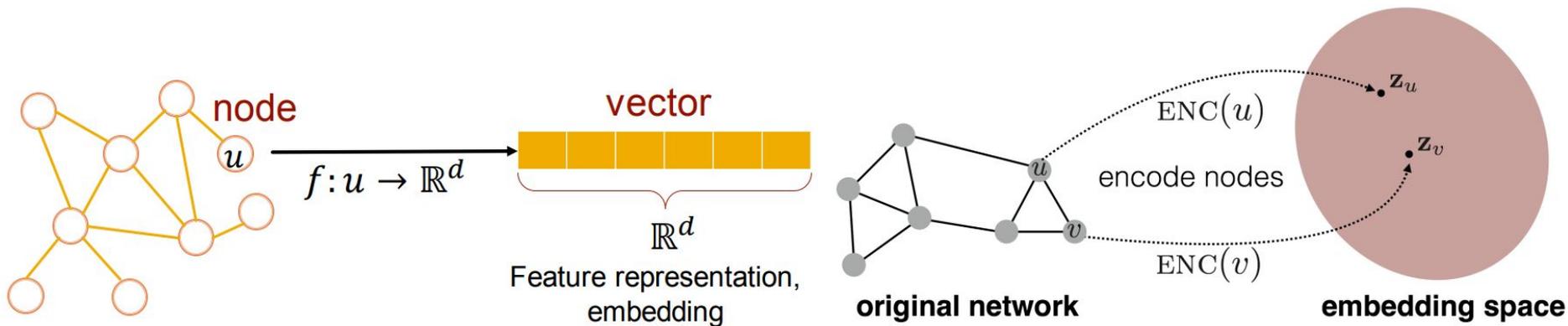
Image credit: Stanford CS224W

N : set of nodes
A : adjacency matrix
$\boldsymbol{X} \in \mathbb{R}^{|N| \times d}$: node (raw) features
u, v : nodes in N
$\mathcal{N}(u)$: neighbors of u

Node raw features: e.g.,
- Biomedical graphs: patient's EHR
- Social network graphs: user profile and images
- When no features: node indicator vector, constant vector

How to define the f ?

We'll bypass fully connected networks directly
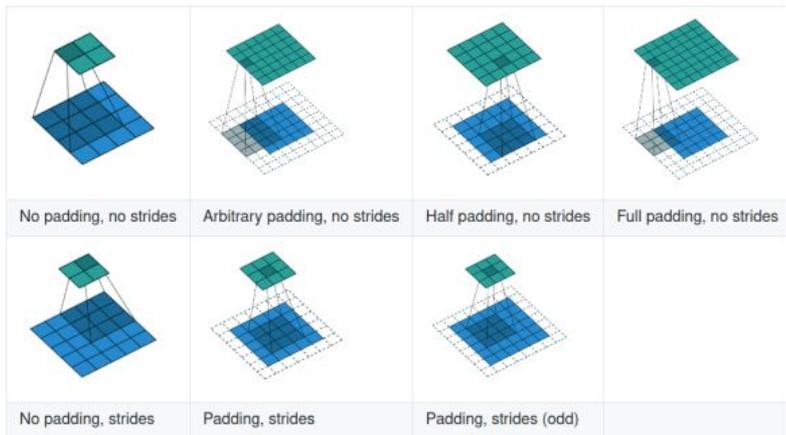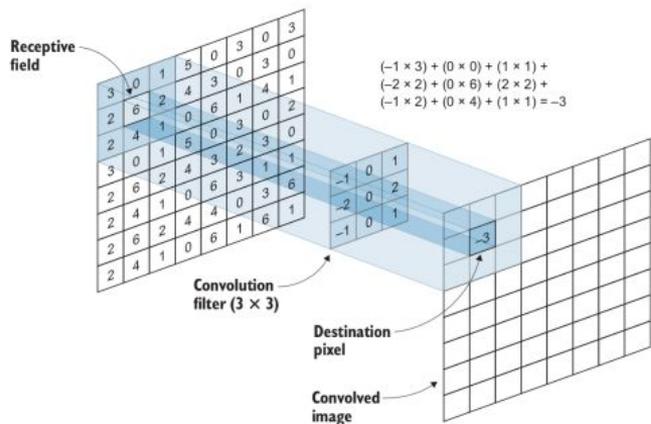
(Credit: [Elgendy, 2020])

https://github.com/vdumoulin/conv_arithmetic

Convolution as performing **local info aggregation** (or **message passing**):
- Each time, the conv window focus on a **local neighborhood** of the current pixel
- Conv effectively **aggregates** the local info by **weighted summation**

How to define the f ?

Convolution as performing **local info aggregation** (or **message passing**):
- **Neighborhood**: Each time, the conv window focus on a **local neighborhood** of the current pixel
- **Aggregation**: Conv effectively **aggregates** the local info by **weighted summation**
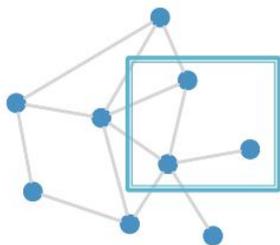
or this:
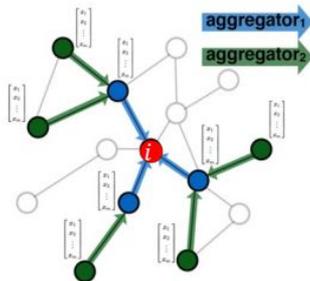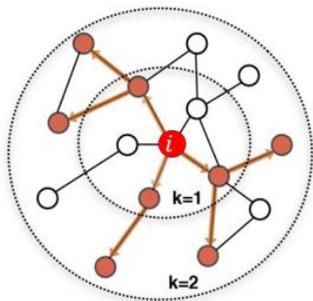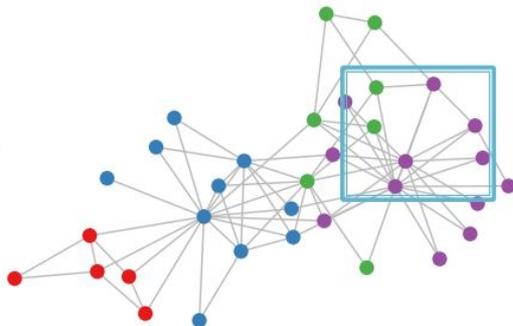
We need neighbors on the graph!

The rigid, grid-based neighborhood doesn't work!

One layer of a graph neural network (GNN)!

Image credit: Stanford CS224W

Graph in, graph out

Input Graph | GNN blocks | Transformed Graph | Classification layer | Prediction
node, edge, or global predictions

Y
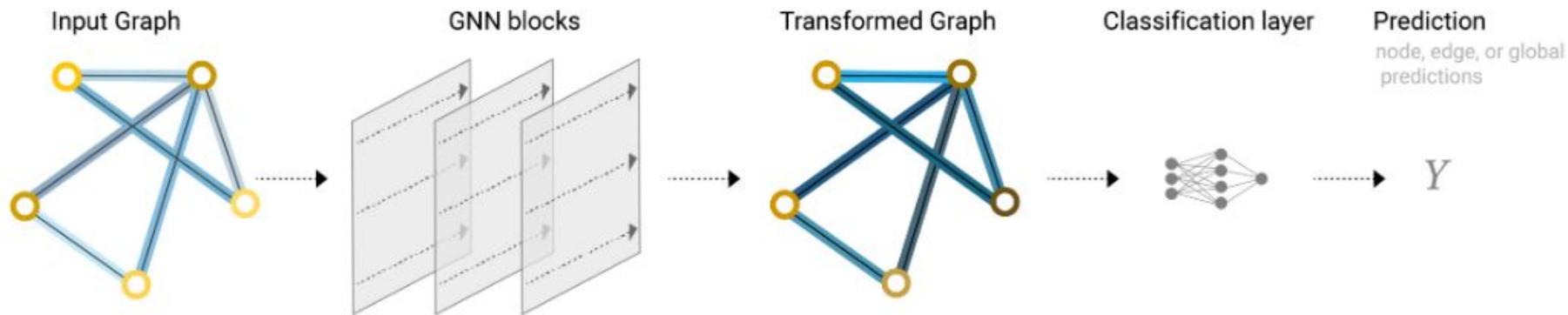
An end-to-end prediction task with a GNN model.

Image credit: https://distill.pub/2021/gnn-intro/

29

How to make supervised predictions?

Image credit: https://distill.pub/2021/gnn-intro/

Predictor on the node directly

Predictor on pairs of nodes

Predictor on pooled feature on whole graph
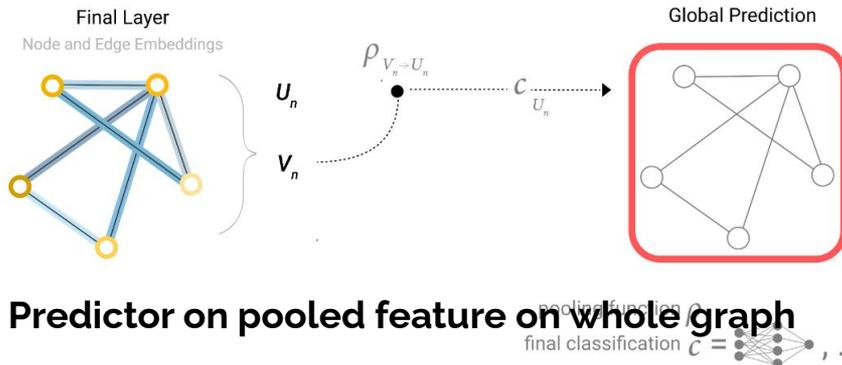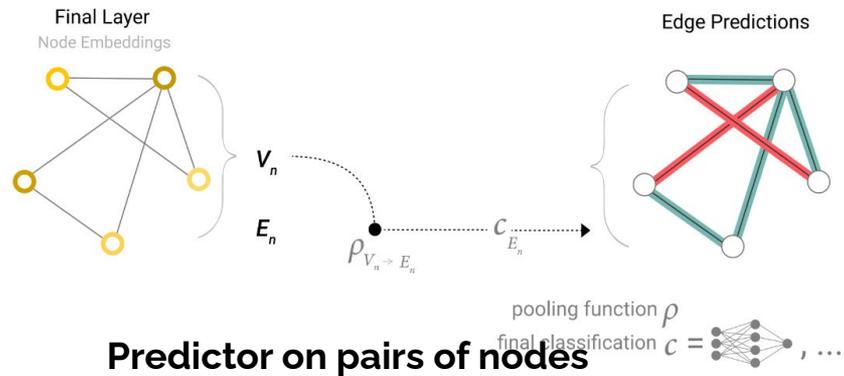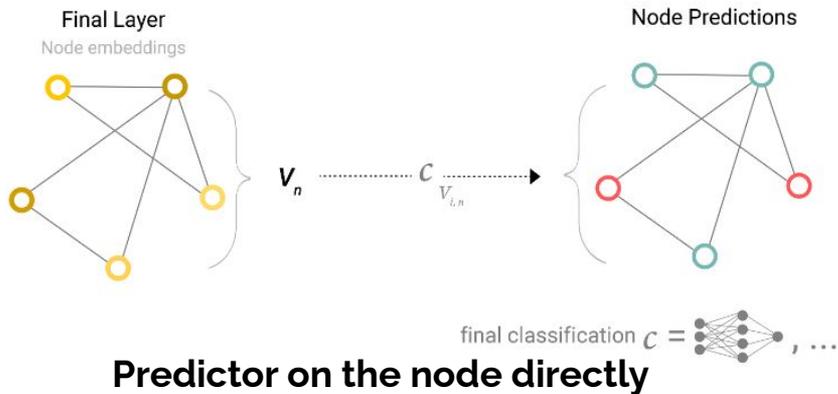
How to perform unsupervised learning?

Goal: $\text{similarity}(u, v) \approx \mathbf{z}_v^{\mathrm{T}} \mathbf{z}_u$

in the original network — Similarity of the embedding

$$\mathcal{L} = \sum_{z_u, z_v} \text{CE}(y_{u,v}, \text{DEC}(z_u, z_v))$$

Need to define!

$\text{ENC}(u)$

encode nodes

$\text{ENC}(v)$

$\mathbf{z}_u$

$\mathbf{z}_v$

original network

embedding space

Image credit: Stanford CS224W

Look into GNN layers

Layer-0

Layer-1

Layer-2

$\mathbf{x}_A$

$\mathbf{x}_C$

$\mathbf{x}_A$

$\mathbf{x}_B$

$\mathbf{x}_E$

$\mathbf{x}_F$

$\mathbf{x}_A$

Image credit: Stanford CS224W

TARGET NODE

INPUT GRAPH
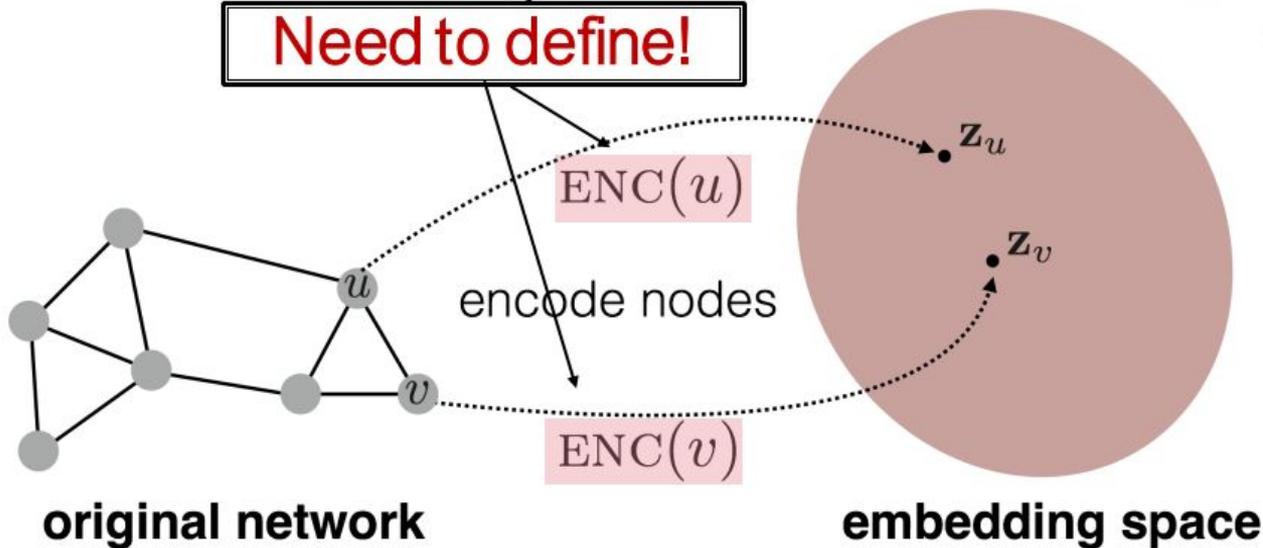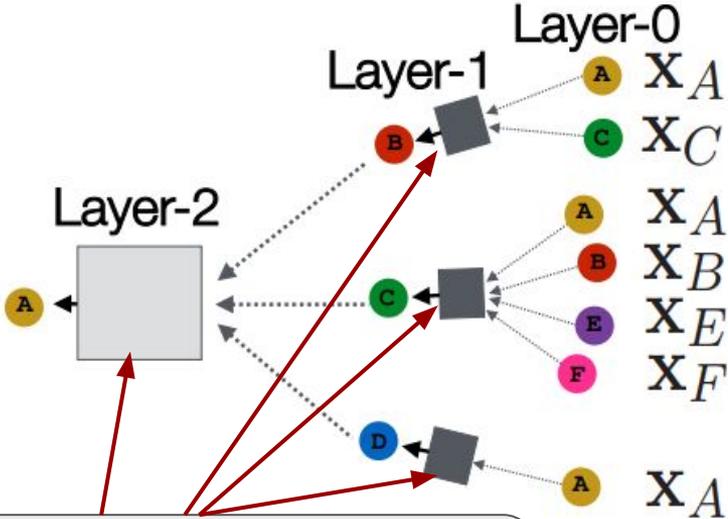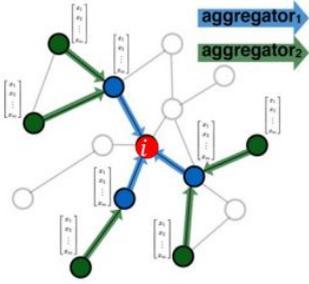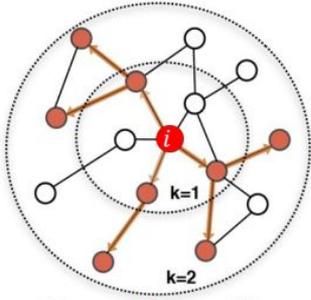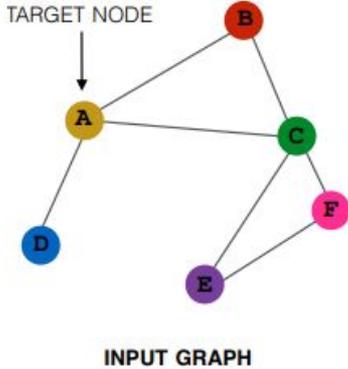
These **aggregators** sum up neighboring info, and can be represented by **DNNs**

aggregator₁

aggregator₂

k=1

k=2

32

Permutation invariance

Permutation invariance: permuting the names of the nodes doesn't change the graph, as graph nodes are intrinsically orderless

Feature vector for the **whole graph**

For $f : G(\boldsymbol{A}, \boldsymbol{X}) \mapsto \boldsymbol{h}$, $f(\boldsymbol{A}_1, \boldsymbol{X}_2) = f(\boldsymbol{A}_2, \boldsymbol{X}_2)$

Order plan 1: A_1, X_1
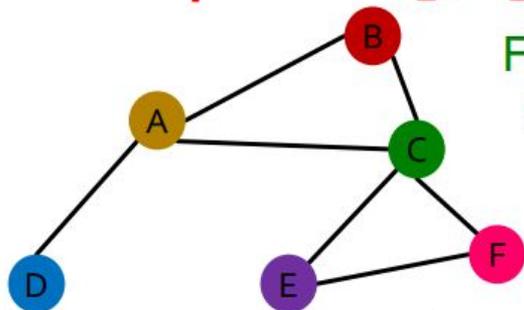
Order plan 2: A_2, X_2

For two order plans, output of f should be the same!
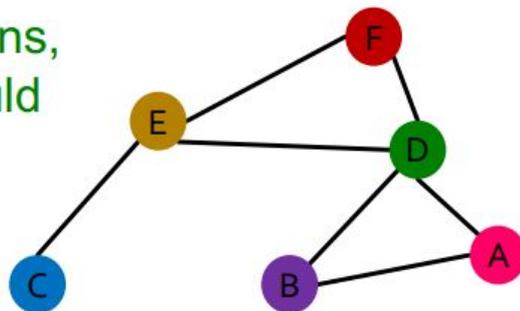
Permutation equivariance

For $f : G(\boldsymbol{A}, \boldsymbol{X}) \mapsto \boldsymbol{H} \in \mathbb{R}^{|N| \times d}$

$$f(\boldsymbol{\Pi A \Pi}^{\mathsf{T}}, \boldsymbol{\Pi X}) = \boldsymbol{\Pi} f(\boldsymbol{A}, \boldsymbol{X})$$

for any permutation $\boldsymbol{\Pi}$

Collection of feature vectors on all nodes

Order plan 1: A_1, X_1

Order plan 2: A_2, X_2

$f(\boldsymbol{A_1}, \boldsymbol{X_1}) =$
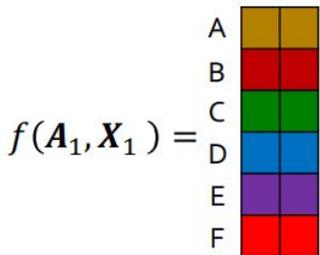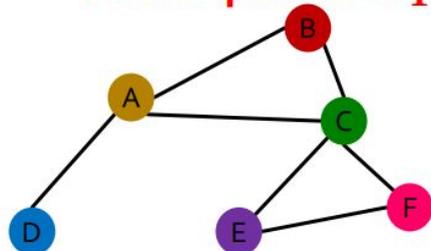
$f(\boldsymbol{A_2}, \boldsymbol{X_2}) =$
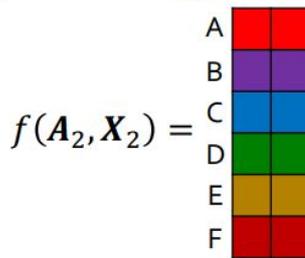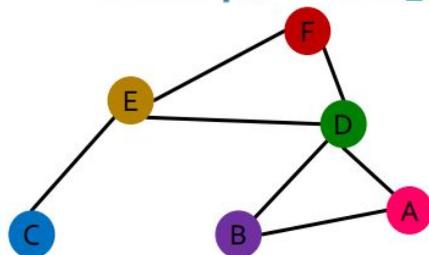
In other words, if the nodes are re-ordered, the learned features are re-ordered accordingly, **so features are attached to nodes not their names**

Image credit: Stanford CS224W

34

A typical GNN consists of multiple permutation equivariant/invariant layers

Image credit: Stanford CS224W
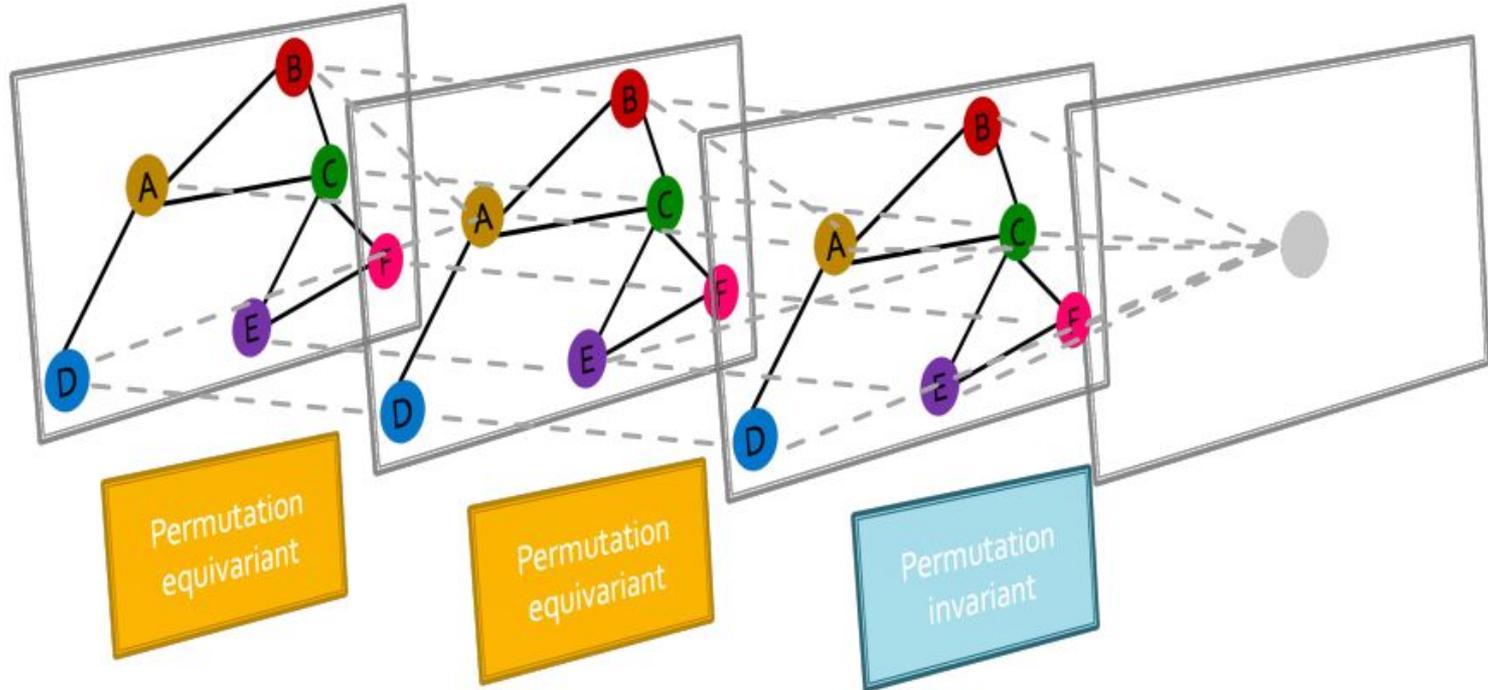
Graph convolutional networks (GCNs)

$$h_v^{(0)} \quad = \quad x_v \qquad \text{for all } v \in V.$$

Node v's initial embedding.

... is just node v's original features.

and for $k = 1, 2, \ldots$ upto K:

$$h_v^{(k)} \quad = \quad f^{(k)} \left(W^{(k)} \cdot \frac{\sum\limits_{u \in \mathcal{N}(v)} h_u^{(k-1)}}{|\mathcal{N}(v)|} + B^{(k)} \cdot h_v^{(k-1)} \right) \qquad \text{for all } v \in V.$$

Node v's embedding at step k.

Mean of v's neighbour's embeddings at step $k-1$.

Node v's embedding at step $k-1$.

Color Codes:

- ■ Embedding of node v.
- ■ Embedding of a neighbour of node v.
- ■ (Potentially) Learnable parameters.

Image credit: https://distill.pub/2021/understanding-gnns/

36
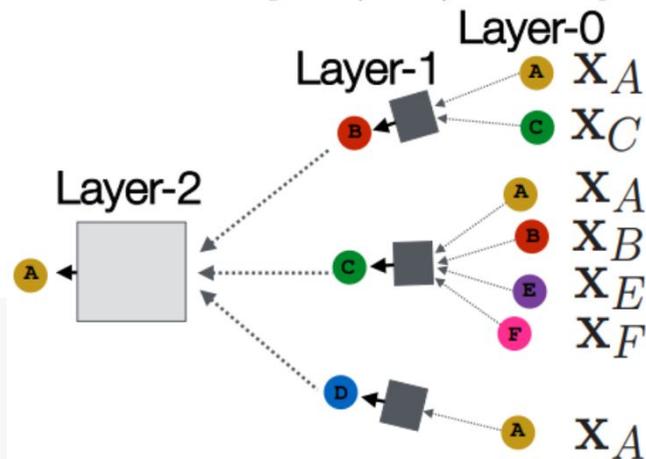
Graph attention networks (GATs)

$$h_v^{(0)} \quad = \quad x_v \qquad \text{for all } v \in V.$$

Node v's initial embedding.

... is just node v's original features.

and for $k = 1, 2, \ldots$ upto K:

$$h_v^{(k)} \quad = \quad f^{(k)}\left(W^{(k)} \cdot \left[\sum_{u \in \mathcal{N}(v)} \alpha_{vu}^{(k-1)} h_u^{(k-1)} + \alpha_{vv}^{(k-1)} h_v^{(k-1)} \right] \right) \qquad \text{for all } v \in V.$$

Node v's embedding at step k.

Weighted mean of v's neighbour's embeddings at step $k - 1$.

Node v's embedding at step $k - 1$.

where the attention weights $\alpha^{(k)}$ are generated by an attention mechanism $A^{(k)}$, normalized such that the sum over all neighbours of each node v is 1:

$$\alpha_{vu}^{(k)} \quad = \quad \frac{A^{(k)}\left(h_v^{(k)}, h_u^{(k)}\right)}{\sum_{w \in \mathcal{N}(v)} A^{(k)}\left(h_v^{(k)}, h_w^{(k)}\right)} \qquad \text{for all } (v, u) \in E.$$

Color Codes:

■ Embedding of node v.

■ Embedding of a neighbour of node v.

■ (Potentially) Learnable parameters.

Image credit: https://distill.pub/2021/understanding-gnns/

Graph sample and aggregate (GraphSAGE)

$$h_v^{(0)} \quad = \quad x_v \quad \text{for all } v \in V.$$

Node v's initial embedding.   ... is just node v's original features.

and for $k = 1, 2, \ldots$ upto K:

$$h_v^{(k)} \quad = \quad f^{(k)}\left(W^{(k)} \cdot \left[\underset{u \in \mathcal{N}(v)}{\mathrm{AGG}}(\{h_u^{(k-1)}\}), \; h_v^{(k-1)} \right] \right) \quad \text{for all } v \in V.$$

Node v's embedding at step k.

Aggregation of v's neighbour's embeddings at step $k - 1$ ...

... Node v's embedding at step $k - 1$.

... concatenated with ...

Color Codes:

■ Embedding of node v.

■ Embedding of a neighbour of node v.

■ (Potentially) Learnable parameters.

Image credit: https://distill.pub/2021/understanding-gnns/

Graph isomorphism networks (GINs)

$$h_v^{(0)} = x_v \quad \text{for all } v \in V.$$

Node v's initial embedding.

... is just node v's original features.

and for $k = 1, 2, \ldots$ upto K:

$$h_v^{(k)} = f^{(k)}\left(\sum_{u \in \mathcal{N}(v)} h_u^{(k-1)} + (1 + \epsilon^{(k)}) \cdot h_v^{(k-1)}\right) \quad \text{for all } v \in V.$$

Node v's embedding at step k.

Sum of v's neighbour's embeddings at step $k - 1$.

Node v's embedding at step $k - 1$.

Color Codes:

- ■ Embedding of node v.
- ■ Embedding of a neighbour of node v.
- ■ (Potentially) Learnable parameters.

Image credit: https://distill.pub/2021/understanding-gnns/

Connection to CNNs and Transformers

filter:
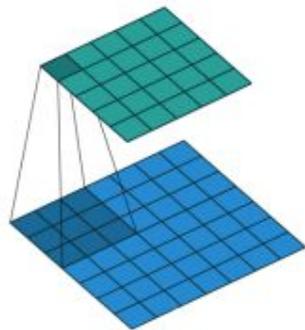
Image
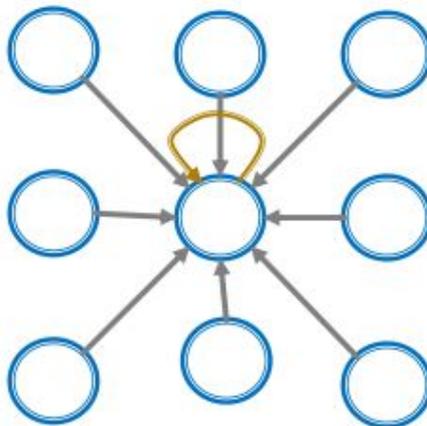
Graph

- CNN is GNN that keeps local ordering
- CNN not permutation-invariant

GNN formulation: $h_v^{(l+1)} = \sigma(\mathbf{W_l} \sum_{u \in N(v)} \frac{h_u^{(l)}}{|N(v)|} + B_l h_v^{(l)}), \forall l \in \{0, \dots, L-1\}$

CNN formulation: $h_v^{(l+1)} = \sigma(\sum_{u \in N(v)} \mathbf{W_l^u} h_u^{(l)} + B_l h_v^{(l)}), \forall l \in \{0, \dots, L-1\}$

Image credit: Stanford CS224W

# Connection to CNNs and Transformers
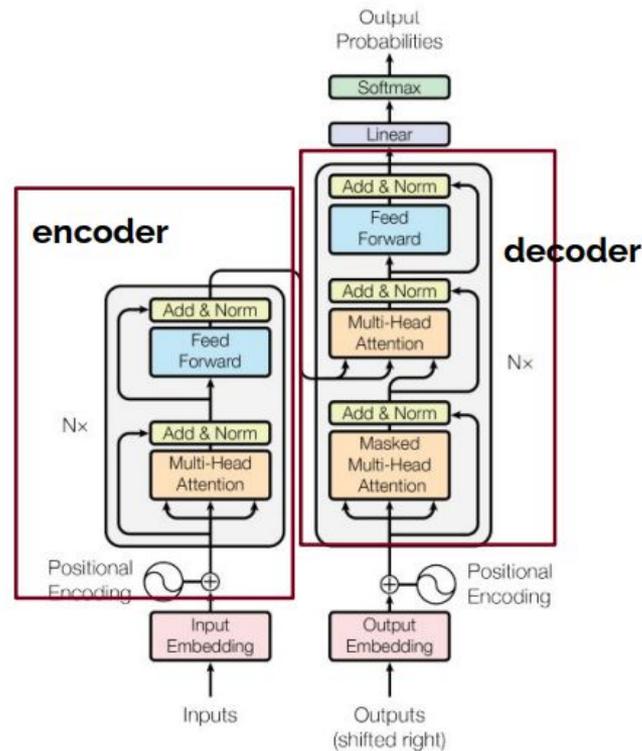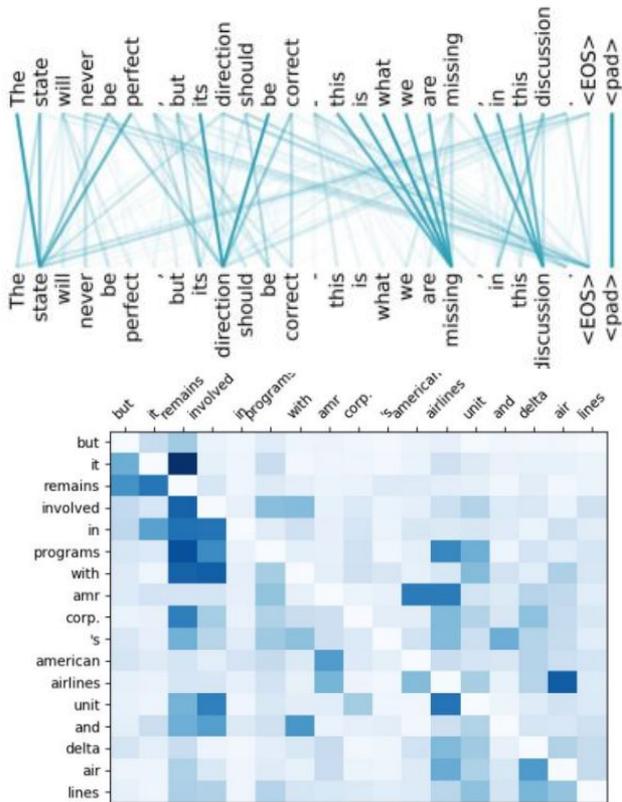
Figure 1: The Transformer - model architecture.

Self-attention (plus feed forward) is a layer of GAT on a complete graph!

Scaling up training

Practical graphs
are large yet sparse

How to perform mini-batch training?

o: Sampled nodes

All the nodes in G
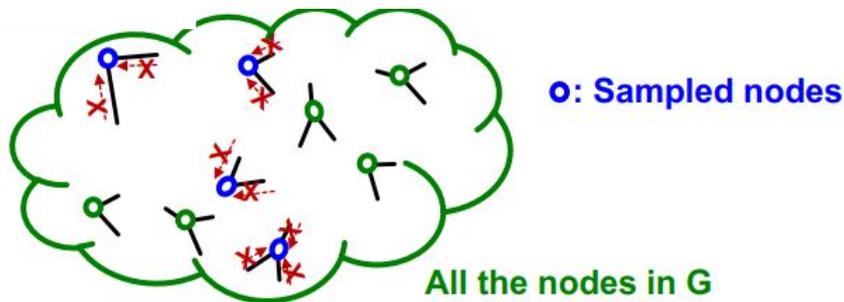
- Mini-batch subsampling induces isolated nodes
- No info to aggregate inside the mini-batch for most nodes

- **Knowledge Graphs (KGs):**
 - Wikidata
  - Freebase
- **ML tasks:**
  - KG completion
  - Reasoning

Entities
80M—90M
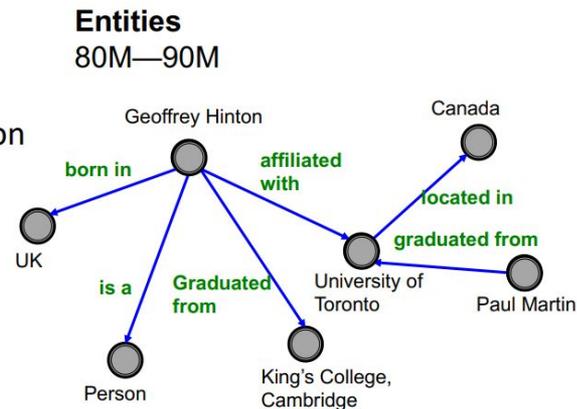
Solution: structured subsampling
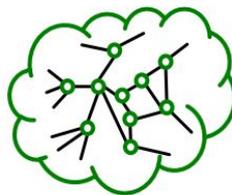
Large graph	Sampled subgraph (small enough to be put on a GPU)	Layer-wise node embeddings update on the GPU
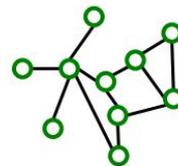
Two structured sub-sampling strategies—I

2-hop neighborhood

Neighbor aggr

Comp. graph for 1-st node

Comp. graph for 2-nd node

Comp. graph for M-th node

Mini-batch

Jure Leskovec, Stanford CS224W: Machine Learning with Graphs
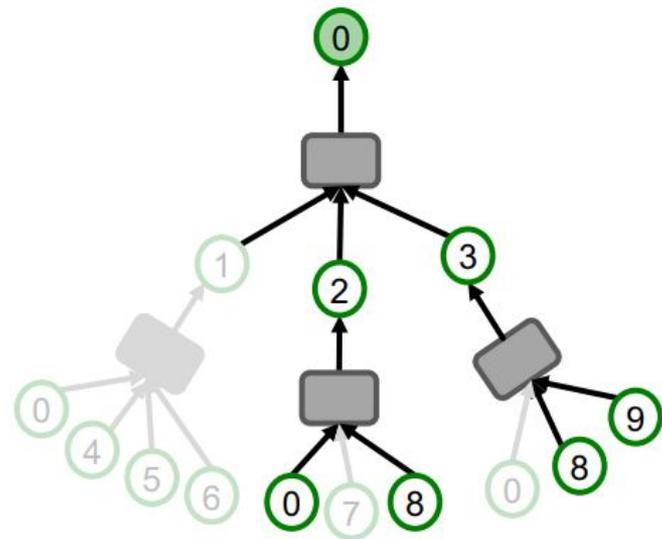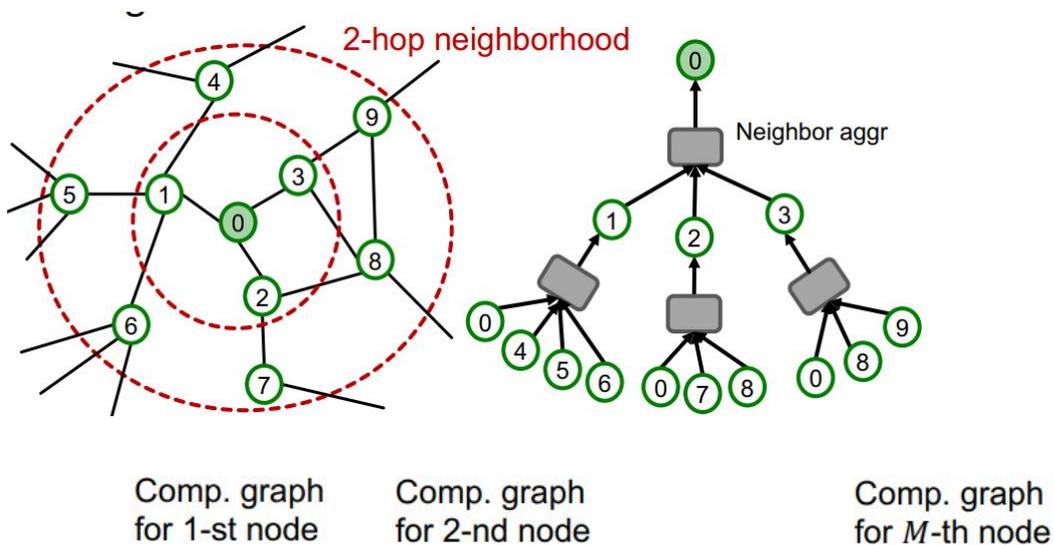
i.e., **neighborhood sampling**, instead of iid uniform sampling

Image credit: Stanford CS224W
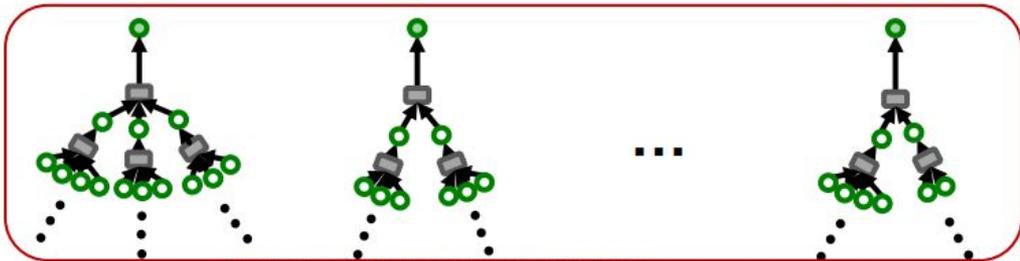
Two structured sub-sampling strategies—II

**Cluster-GCN** consists of two steps:

- **Pre-processing**: Given a large graph, partition it into groups of nodes (i.e., subgraphs).

- **Mini-batch training**: Sample one node group at a time. Apply GNN's message passing over the **induced subgraph**.

Rationale: important to keep the community structures, i.e., keep the "backbone" nodes

Input large graph — Partitioning — Sample — Mini-batch training Message-passing over induced subgraph to compute the loss

Image credit: Stanford CS224W
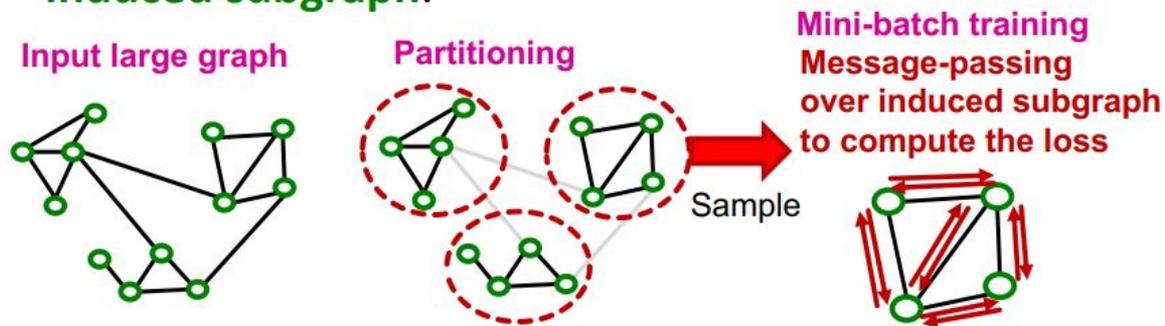
Software

PyTorch Geometric (PyG)

Deep Graph Library (DGL)

https://pytorch-geometric.readthedocs.io/en/latest/

https://www.dgl.ai/
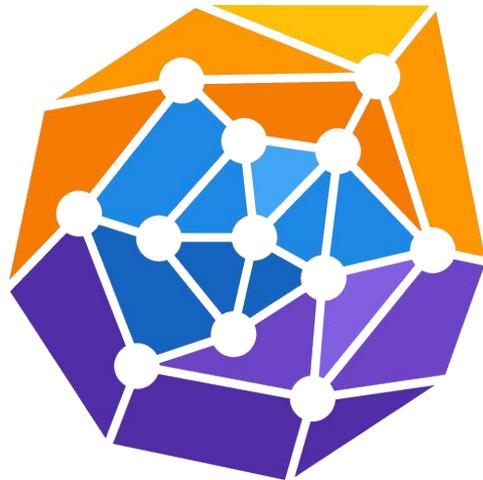
Further reading

- What are graph neural networks?
  https://blogs.nvidia.com/blog/2022/10/24/what-are-graph-neural-networks/
- A Gentle Introduction to Graph Neural Networks
  https://distill.pub/2021/gnn-intro/
- Understanding Convolutions on Graphs
  https://distill.pub/2021/understanding-gnns/
- Graph Neural Networks: A Review of Methods and Applications
  https://arxiv.org/abs/1812.08434
- Stanford CS224W: Machine Learning with Graphs
  https://web.stanford.edu/class/cs224w/index.html
- Graph Representation Learning   https://www.cs.mcgill.ca/~wlh/grl_book/