Graph Neural Networks

And a Myriad of Things that are Relevant

Author: Viet Nhat Nguyen

University of Information Technology
21520378@gm.uit.edu.vn

Table of Contents

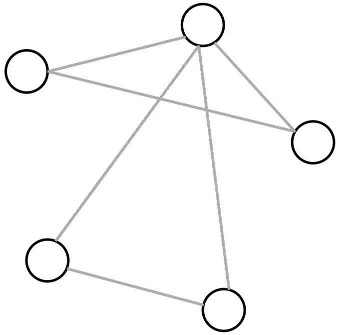
What is a Graph?

$$G = \{V, E\}$$

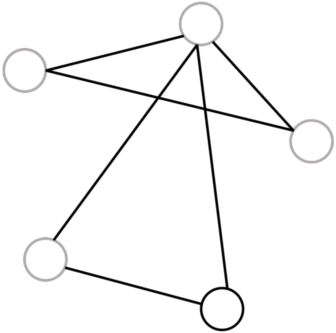
- A graph represents the relation (*edges*) between a collection of entities (*nodes*).

- V Vertices or nodes
- E Edges or links
- G Graph

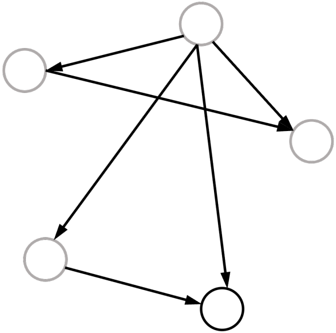
What is a Graph?



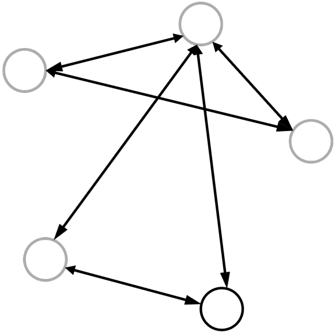
What is a Graph?



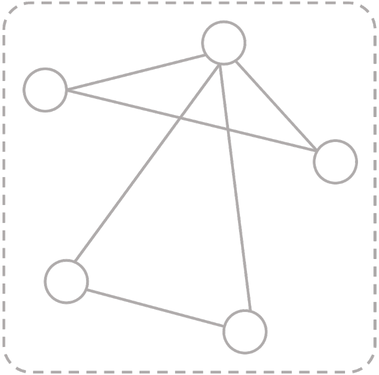
What is a Graph?



What is a Graph?



Attributes of Graph



V Vertex (or node) attributes

e.g., node identity, number of neighbors

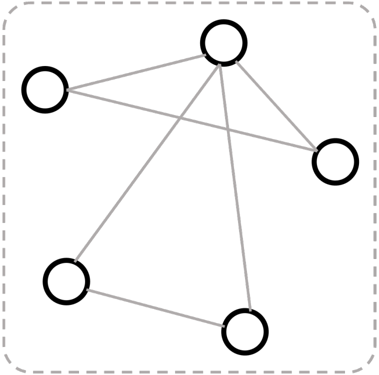
E Edge (or link) attributes and directions

e.g., edge identity, edge weight

U Global (or master node) attributes

e.g., number of nodes, longest path

Attributes of Graph



V Vertex (or node) attributes

e.g., node identity, number of neighbors

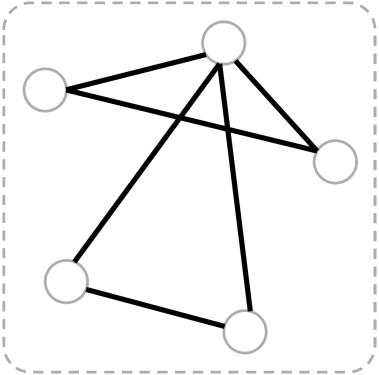
E Edge (or link) attributes and directions

e.g., edge identity, edge weight

U Global (or master node) attributes

e.g., number of nodes, longest path

Attributes of Graph



V Vertex (or node) attributes

e.g., node identity, number of neighbors

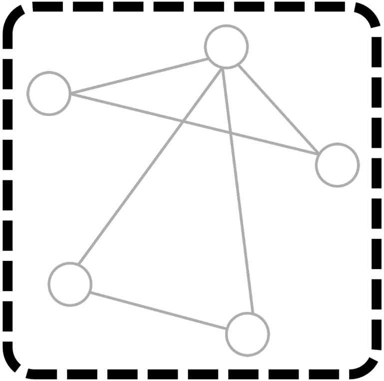
E Edge (or link) attributes and directions

e.g., edge identity, edge weight

U Global (or master node) attributes

e.g., number of nodes, longest path

Attributes of Graph



V Vertex (or node) attributes

e.g., node identity, number of neighbors

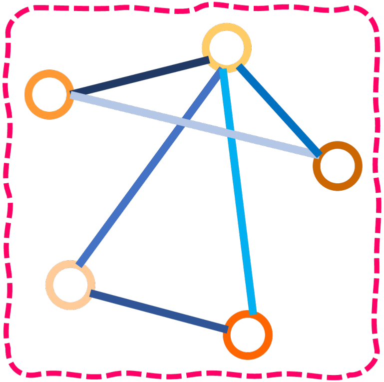
E Edge (or link) attributes and directions

e.g., edge identity, edge weight

U Global (or master node) attributes

e.g., number of nodes, longest path

Attributes of Graph



V Vertex (or node) embedding

~~e.g., node identity, number of neighbors~~

E Edge (or link) embedding

~~e.g., edge identity, edge weight~~

U Global (or master node) embedding

~~e.g., number of nodes, longest path~~

Where to Find them?

Graphs are all around us; real world objects are often defined in terms of their connections to other things.

Images as Graphs

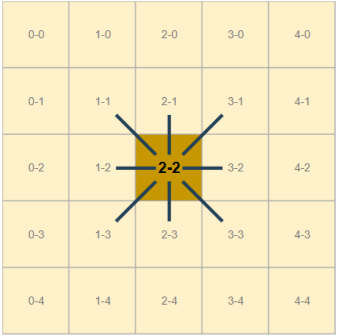
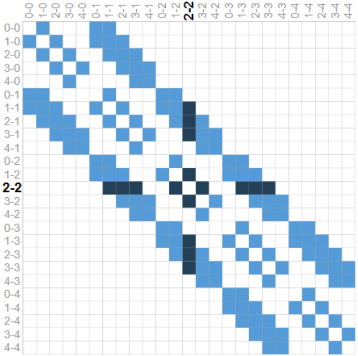
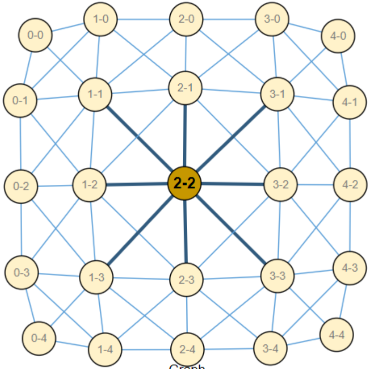


Image Pixels

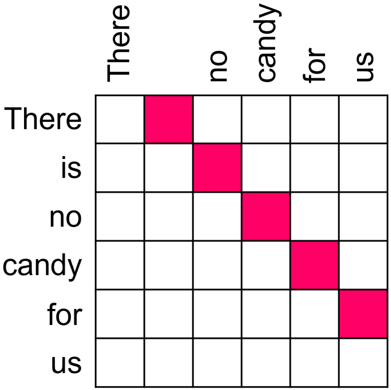


Adjacency Matrix



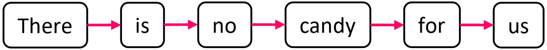
Graph

Text as Graphs



Adjacency Matrix

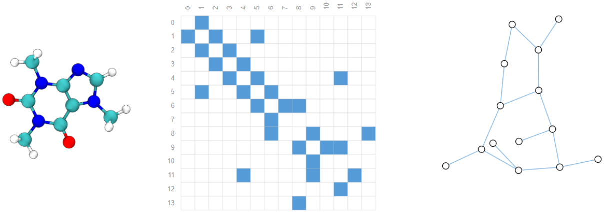
Text: There is no candy for us



Graph

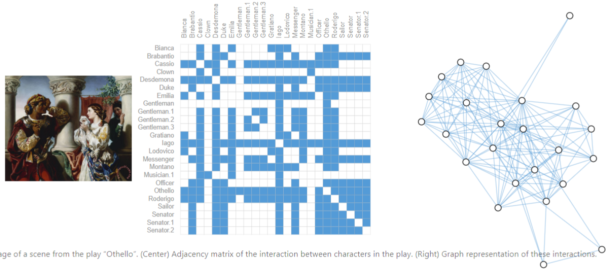
Graph-valued Data in the Wild

- Molecules



Graph-valued Data in the Wild

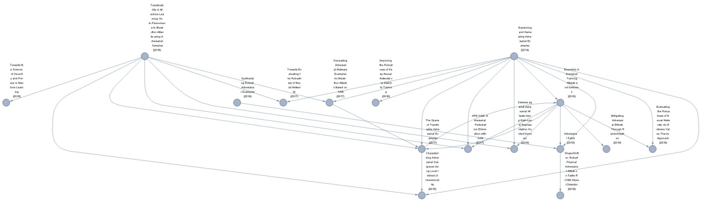
- Molecules
- Social networks



(Left) Image of a scene from the play "Othello". (Center) Adjacency matrix of the interaction between characters in the play. (Right) Graph representation of these interactions.

Graph-valued Data in the Wild

- Molecules
- Social networks
- Citation networks



Graph-valued Data in the Wild

- Molecules
- Social networks
- Citation networks
- etc

Dataset	Domain	graphs	nodes	edges	Edges per node (degree) min	mean	max
karate club	Social network	1	34	78		4.5	17
qm9	Small molecules	134k	≤ 9	≤26	1	2	5
Cora	Citation network	1	23,166	91,500	1	7.8	379
Wikipedia links, English	Knowledge graph	1	12M	378M		62.24	1M

Summary statistics on graphs found in the real world. Numbers are dependent on featurization decisions. More useful statistics and graphs can be found in KONECT [14]

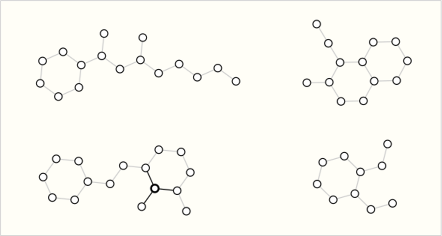
What types of problems have graph structured data?

Graph-level Task

- Predicting the property of an entire graph

Graph-level Task

- Predicting the property of an entire graph

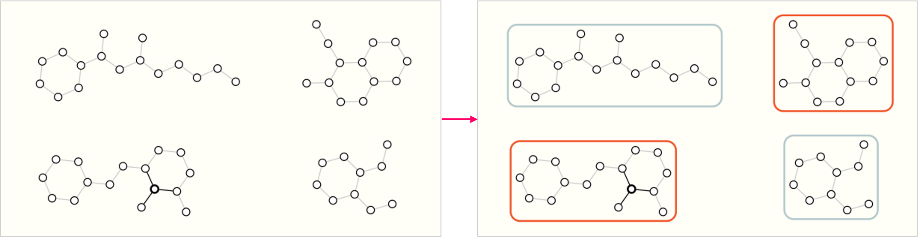


- **Input**: graphs

Graph-level Task

- Predicting the property of an entire graph

- **Input**: graphs
- **Output**: labels for each graph



Graph-level Task

- Predicting the property of an entire graph
  - the smell of the molecular
  - image classification
  - sentiment analysis of text

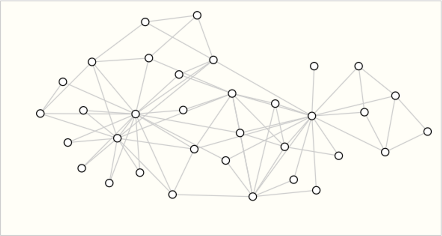
- **Input**: graphs
- **Output**: labels for each graph

Node-level Task

- Predicting the identity or role of each node

Node-level Task

- Predicting the identity or role of each node

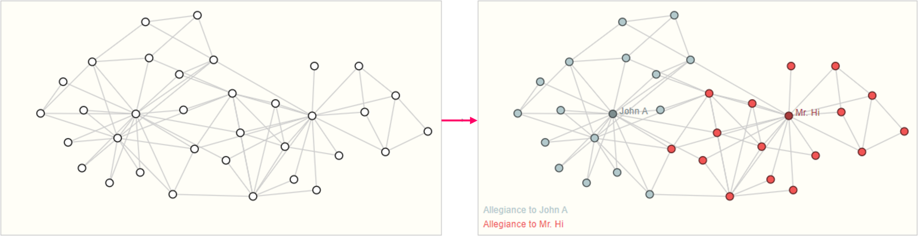


- **Input**: graphs (unlabeled nodes)

Node-level Task

- Predicting the identity or role of each node

- **Input**: graphs (unlabeled nodes)
- **Output**: graph node labels



Node-level Task

- Predicting the identity or role of each node
    - image segmentation
    - PoS of each word

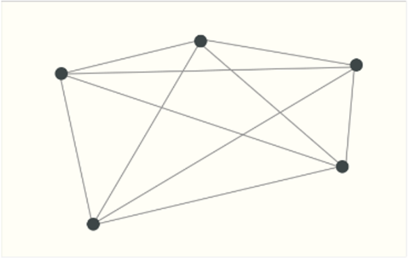
- **Input**: graphs (unlabeled nodes)
- **Output**: graph node labels

Edge-level Task

- Predicting the identity or role of each edge

Edge-level Task

- Predicting the identity or role of each edge

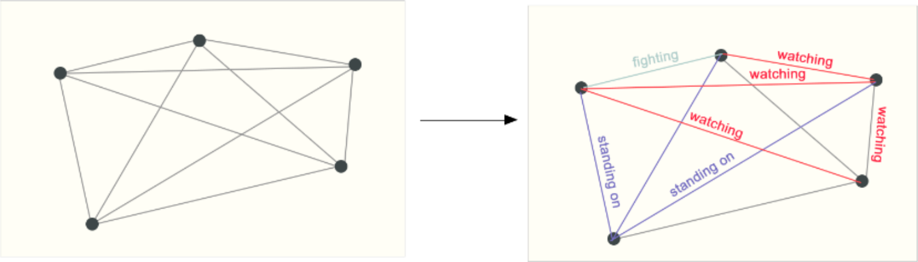


- **Input**: graphs (unlabeled edges)

Edge-level Task

- Predicting the identity or role of each edge

- **Input**: graphs (unlabeled edges)
- **Output**: labels for edges

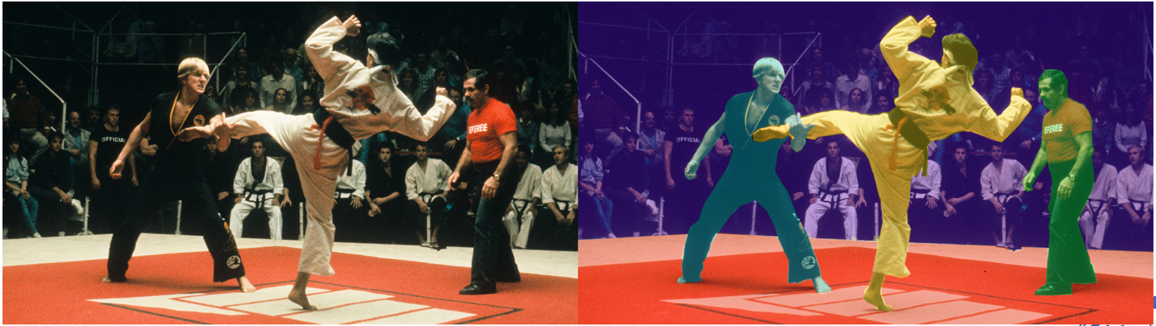


Edge-level Task

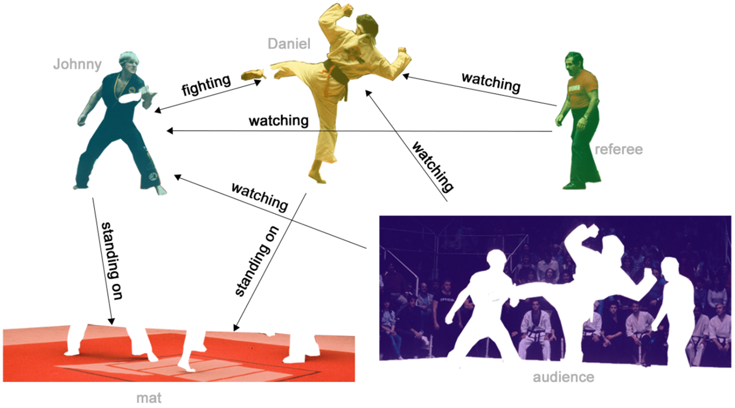
- Predicting the identity or role of each edge
  - scence understanding

- **Input**: graphs (unlabeled edges)
- **Output**: labels for edges

Edge-level Task



Edge-level Task



How do we go about solving these different graph tasks with neural networks?

How to Represent Graphs

Graphs have 4 types of information:

- nodes,
- edges,
- global-context,
- and connectivity!

How to Represent Graphs

Graphs have 4 types of information:

- nodes,
- edges,
- global-context,
- and **connectivity**!

$$N = \{node_i\} \quad [n_{nodes}, node_{dim}]$$
$$E = \{edge_i\} \quad [n_{edges}, edge_{dim}]$$
$$U = master\ node's\ embedding\ [, global_{dim}]$$

Representing Graph's Connectivity

- One elegant way to represent graphs is as **Adjacency lists**: These describe the connectivity of edge e_k between nodes n_i and n_j as a tuple (i, j) in the k-th entry of an adjacency list.
  - memory-efficient (even with sparse matrices),
  - permutation invariant.

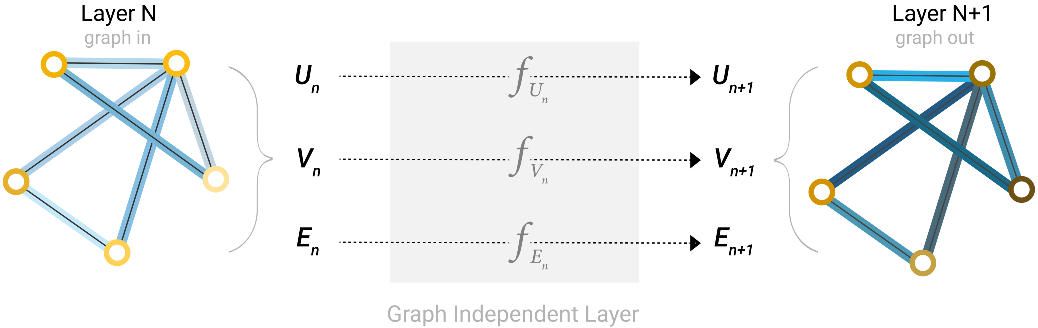
Representing Graph's Connectivity

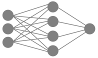
Images/adjacency-list.png

The simplest GNN

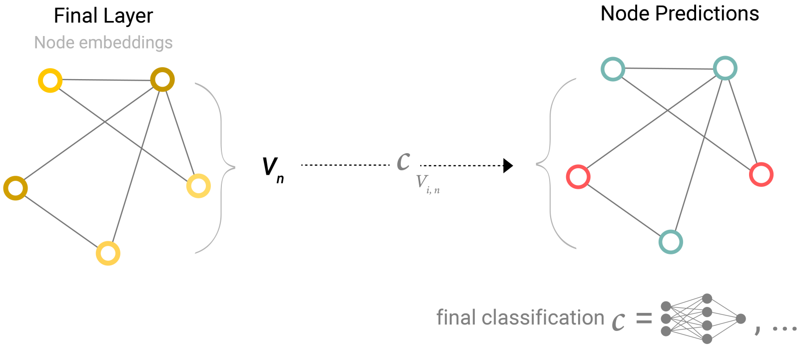
- **GNN**s = **G**raph **N**eural **N**etworks
- **A GNN is an optimizable transformation on all attributes of the graph** (i.e., nodes, edges, global-context) **that preserves graph symmetries** (connectivity or permutation invariances)
- *"graph-in, graph-out"* architecture

The Simplest GNN



update function $f =$  , ...

GNN Predictions



Pooling Information

- **Pooling**: Collecting information from 1 component and giving them to another.

Pooling Information

- **Pooling**: Collecting information from 1 component and giving them to another.
  1. For each item to be pooled, *gather* each of their embeddings and concatenate them into a matrix.

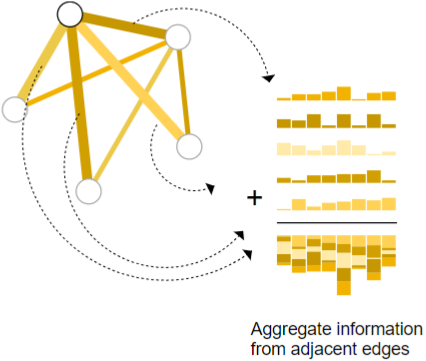
Pooling Information

- **Pooling**: Collecting information from 1 component and giving them to another.
  1. For each item to be pooled, *gather* each of their embeddings and concatenate them into a matrix.
  2. The gathered embeddings are then *aggregated*, using via a operation (e.g., sum, mean).

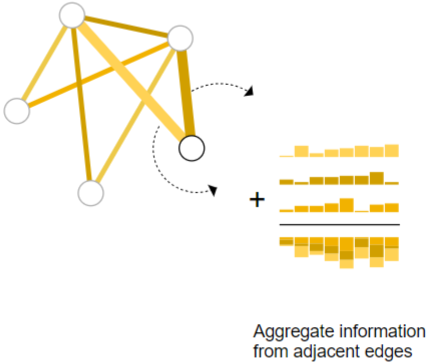
Pooling Information

- **Pooling**: Collecting information from 1 component and giving them to another.
  1. For each item to be pooled, *gather* each of their embeddings and concatenate them into a matrix.
  2. The gathered embeddings are then *aggregated*, using via a operation (e.g., sum, mean).
- Abbreviation: ρ e.g., gathering information from edges to nodes $\rho_{E_n \to V_n}$.

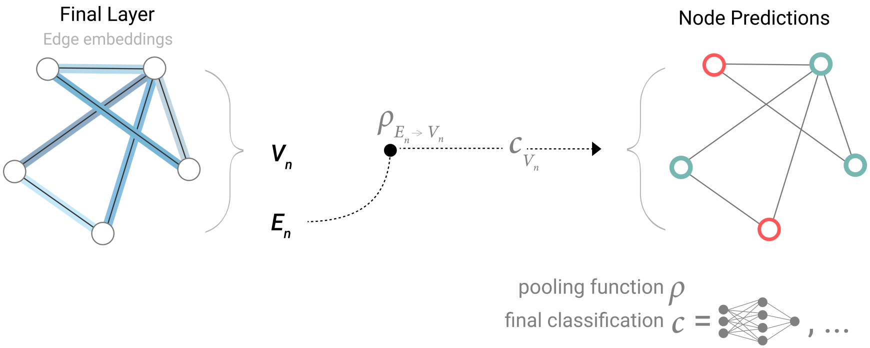
Pooling Information



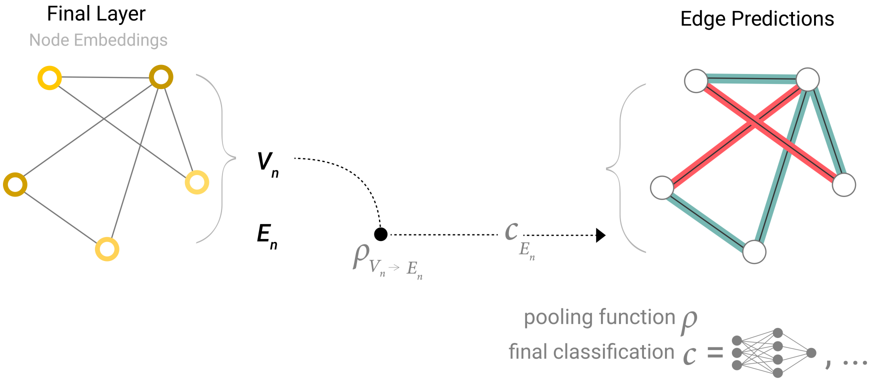
Pooling Information



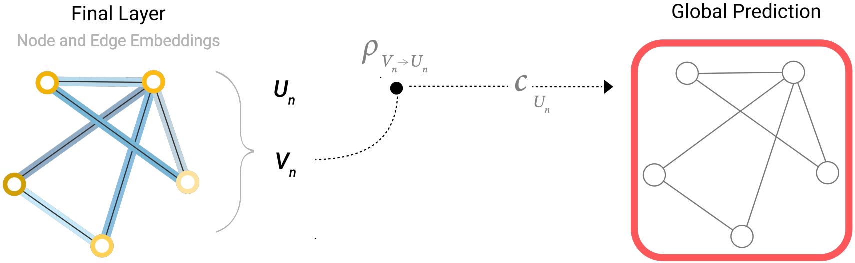
GNN Predictions by Pooling Information



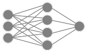
GNN Predictions by Pooling Information



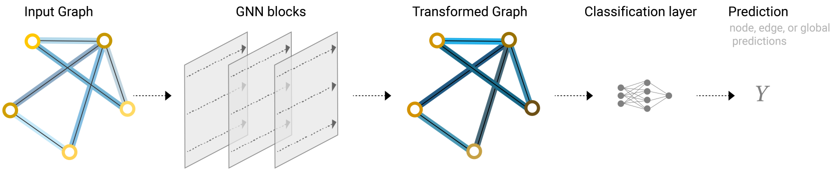
GNN Predictions by Pooling Information



pooling function ρ

final classification c = , ...

End2End Prediction Task with a GNN Model



End2End Prediction Task with a GNN Model

- We still don’t use the connectivity of the graph
  - Each component is processed independently
  - ONLY use connectivity when prediction

Passing messages between parts of the graph

...What if we try using pooling inside GNN Layers?

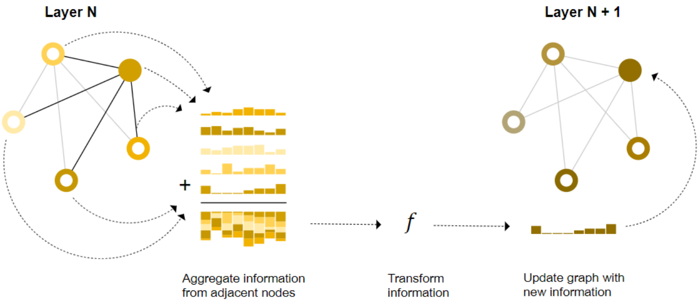
...What if we try using pooling inside GNN Layers?

Or to make our learned embeddings aware of graph **connectivity**?

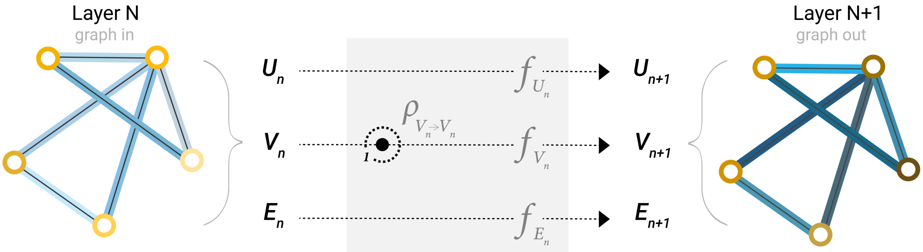
Message Passing

- **Message Passing:**
  1. For each node in the graph, *gather* all the neighboring node embedding (or messages), which is the g function described above.
  2. *Aggregate* all messages via an aggregate function (e.g, sum).
  3. All pooled messages are passed through an *update function*, usually a learned neural network.
- Just as pooling can be applied to either nodes or edges, message passing can occur between either nodes or edges.

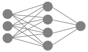
Message Passing



GCN architecture

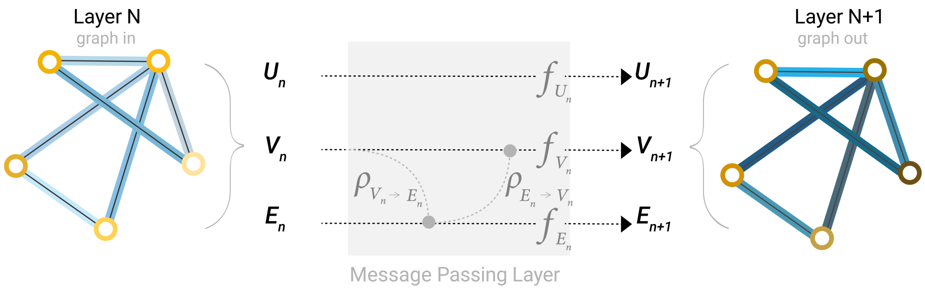


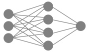
Graph Convolutional Layer

update function $f =$  , ...

pooling function ρ

Learning Edge Representations



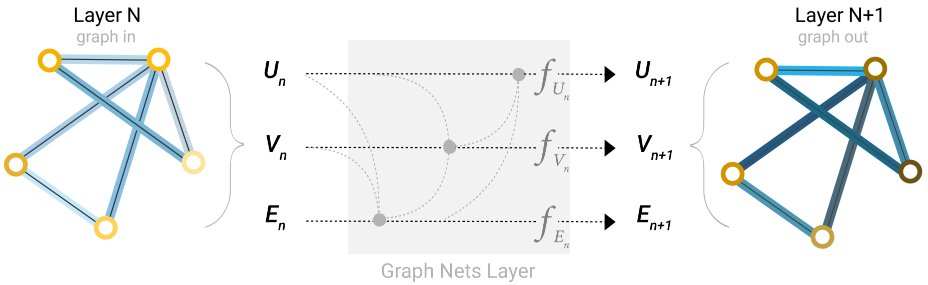
update function $f =$  , ...

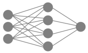
pooling function ρ

Adding Global Representations

- So far, there is still one flaw:
  - Nodes that far away from each other in the graph may never be able to efficiently transfer information to one another, even which several layers.
  - e.g., if we have *k*-layers, information will propagate at most *k*-steps away.
- One solution is by using the global representations or **master node**.

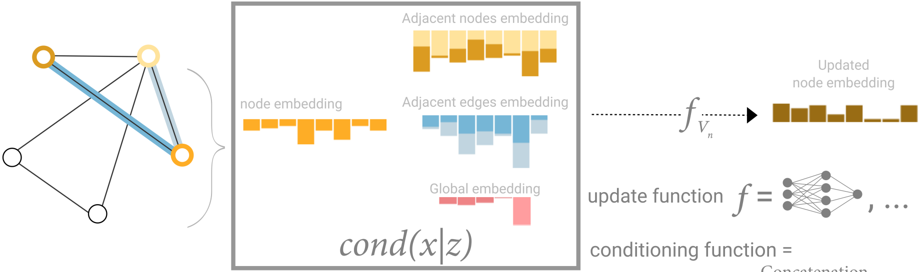
Adding Global Representations



update function $f =$  , ...

pooling function ρ

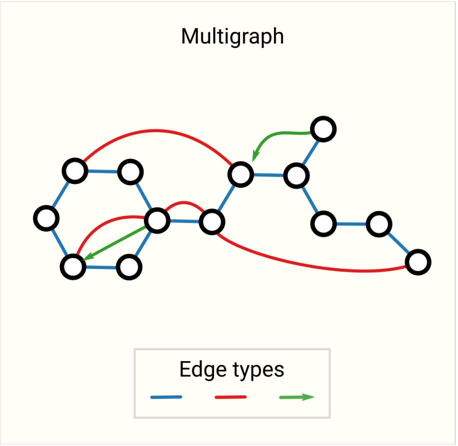
Adding Global Representations

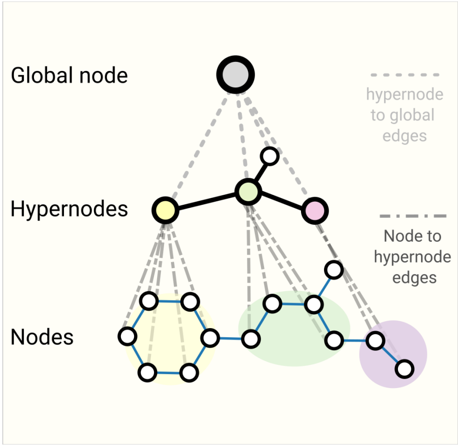


conditioning function =
Concatenation,
Linear Layer and Add,
FiLM Layer..

Into the Weeds

Other types of graphs





Sampling Graphs and Batching in GNNs

- For training neural networks, it's common to:

- For training neural networks, it's common to:
  - calculate on randomized but **constant** size subsets of the training data (mini-batches).

- For training neural networks, it's common to:
  - calculate on randomized but **constant** size subsets of the training data (mini-batches).
- Sometimes, a graph is too large that it cannot be fit in memory.

Sampling Graphs and Batching in GNNs

- For training neural networks, it's common to:
  - calculate on randomized but **constant** size subsets of the training data (mini-batches).
- Sometimes, a graph is too large that it cannot be fit in memory.

Sampling Graphs and Batching in GNNs

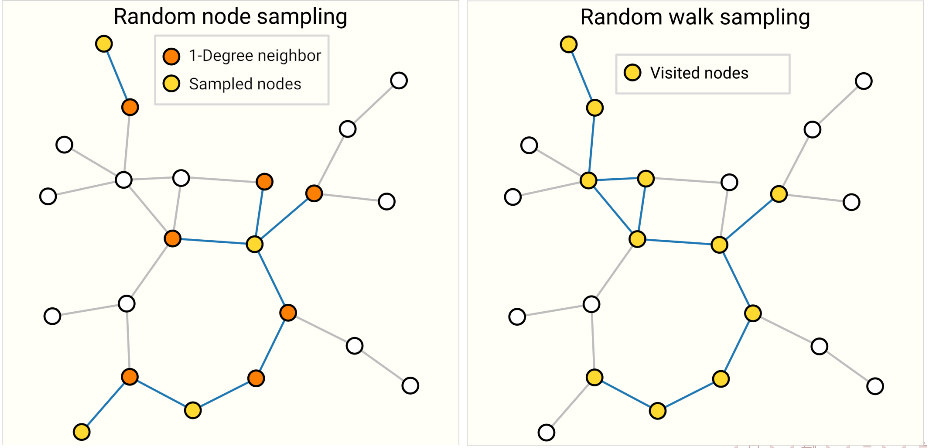
- For training neural networks, it's common to:
  - calculate on randomized but **constant** size subsets of the training data (mini-batches).
- Sometimes, a graph is too large that it cannot be fit in memory.
- However, the number of nodes and edges adjacent to each other are variable.

Sampling Graphs

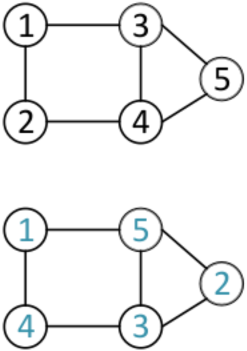
- To create subgraphs that preserve essential properties of the larger graph.
- This graph sampling operation is highly dependent on context,
  - some might make sense in some contexts (citation networks),
  - but might be too strong of an operation in others (molecule graph).

- Cluster-GCN, GraphSaint, Metropolis algorithm, etc...

Sampling Graphs



Generative Modelling



0	1	1	0	0
1	0	0	1	0
1	0	0	1	1
0	1	1	0	1
0	0	1	1	0

0	0	0	1	1
0	0	1	0	1
0	1	0	1	1
1	0	1	0	0
1	1	1	0	0

Same graph

Very different representations!

Generative Modelling

Example: Generate a ring graph on 6 nodes: