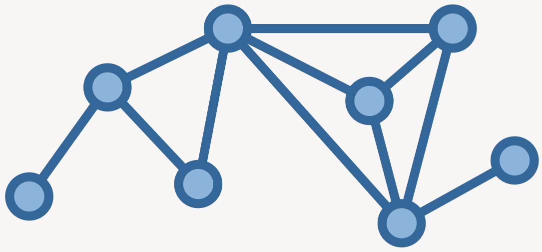
Cheatsheet: Graphs

Graph Neural Networks and Transformers: Neural Networks for Sets

graph

a set of nodes and edges

G = (V, E)



---

node / vertex

what do the nodes represent?
problem dependent (manual choice)

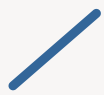
properties:
id, label (optional), feature (optional)

---

edge

What do edges represent?
problem dependent (manual choice)

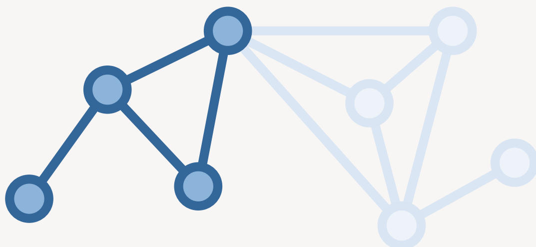
properties:
id, weight (optional), direction (optional), label (optional), feature (optional)



---

component

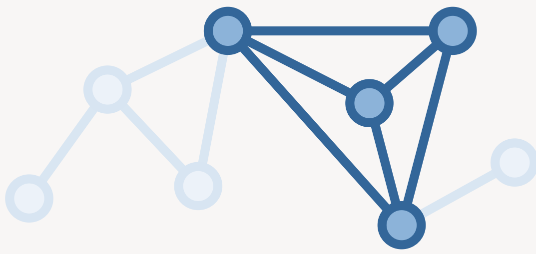
any subgraph consisting of nodes that by themselves would represent a connected graph



---

clique

subgraph that is complete (all possible connections exist between the nodes)

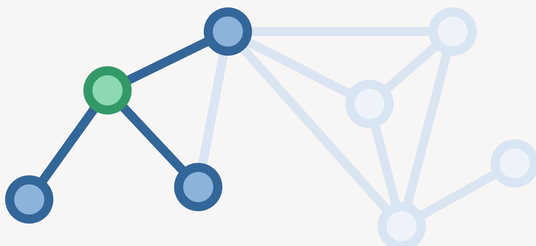


---

neighborhood

relative to a defined node/ vertex v; N(v)

subgraph that contains all nodes connected to node v (usually not containing v itself)



ENCCS EuroCC National Competence Centre Sweden
EURO
EuroHPC Joint Undertaking
UPPSALA UNIVERSITET
RISE
Swedish Research Council
VINNOVA Sweden's Innovation Agency

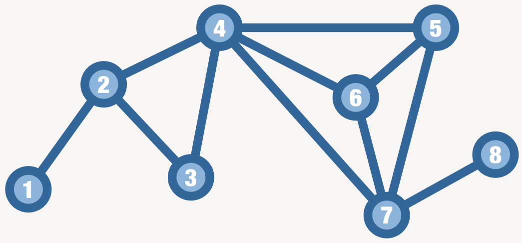
Cheatsheet: Graphs

Graph Neural Networks and Transformers: Neural Networks for Sets

ID

integer number

unique for each node / edge



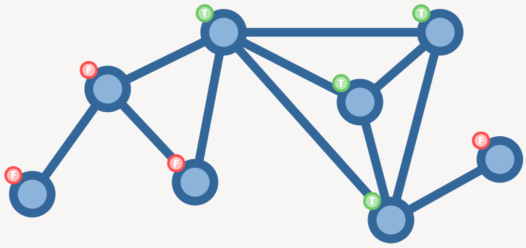
---

label

typically boolean or integer

not unique per node / edge

examples: toxicity, group assignment

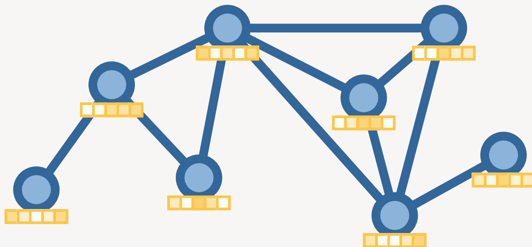


---

feature

contains information about a node or edge

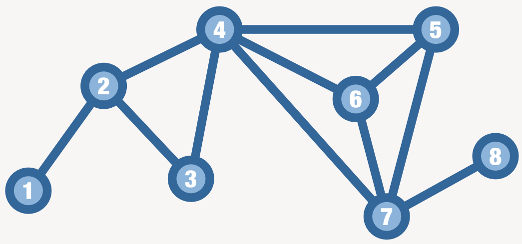
d-dimensional vector (often of real numbers)



---

adjacency matrix

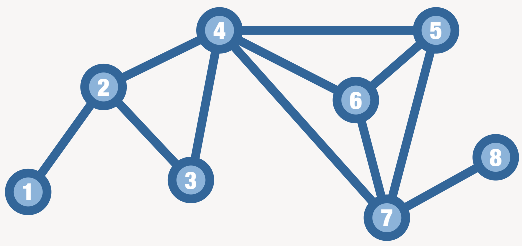
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$



---

list of edges

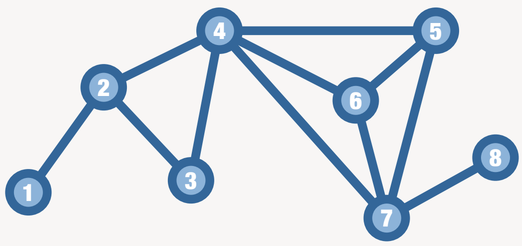
(1, 2)
(2, 3)
(2, 4)
(3, 4)
(4, 5)
(4, 6)
(4, 7)
(5, 6)
(5, 7)
(6, 7)
(7, 8)



---

adjacency list

1: 2
2: 1, 3, 4
3: 2, 4
4: 2, 3, 5, 6, 7
5: 4, 6, 7
6: 4, 5, 7
7: 4, 5, 6, 8
8: 7



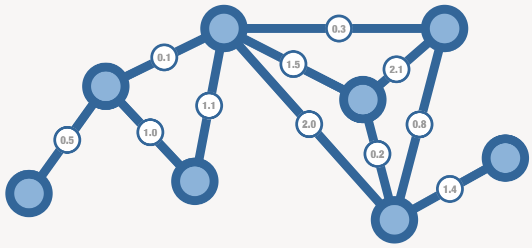
Cheatsheet: Graphs

Graph Neural Networks and Transformers: Neural Networks for Sets

edge weights

unweighted:
all edges represented as “1” in adjacency matrix

weighted:
all real numbers possible

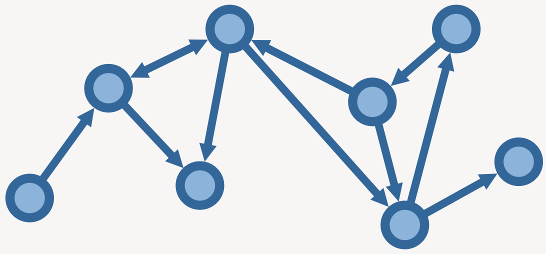


---

edge directions (directed / undirected graph)

examples undirected:
collaborations, friendships on facebook

examples directed:
phone calls, Instagram followers

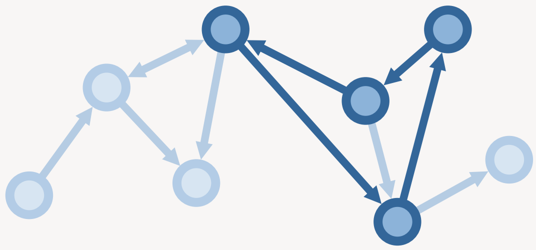


---

cycle (cyclic / acyclic graph)

(simplified) a set of adjacent edges that end on the same node they started on

directed graphs: travelling only allowed in the direction of the edges



---

self-loops

node connected to itself via an edge

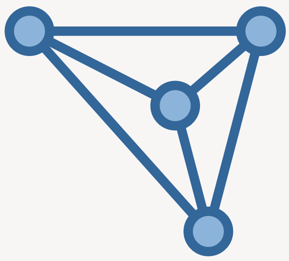
can either be allowed or not

example:
hyperlink networks

---

completeness

complete graph: a graph in which all possible connections exist



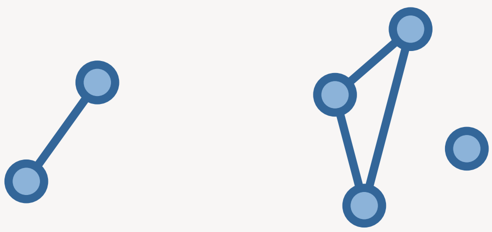
---

disconnected graphs

largest connected group = giant component

can feature isolated nodes

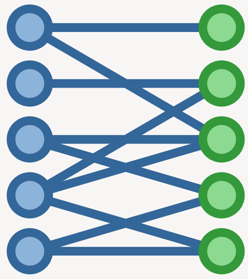
leads to block-sparse adjacency matrix



bipartite graphs

nodes can be divided in two sets: all edges only between the two sets, never within a set

example:
actors and movies, authors and papers

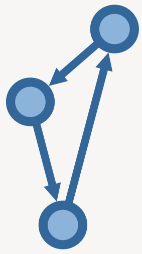


---

strongly connected graph / component

applies to (components in) directed graphs

strongly connected = there is a path from every node to every other node

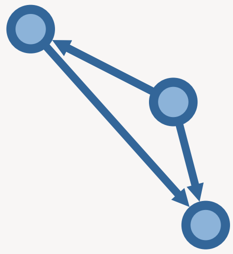


---

weakly connected graph / component

applies to (components in) directed graphs

weakly connected = there only is a path from every node to every other node if we disregard the edge directions



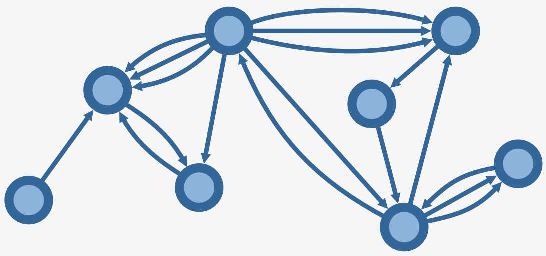
---

multigraph

multiple edges allowed between two nodes

difference to weighted graphs: each edge can have its own label / feature / properties

example:
collaborations, phone calls



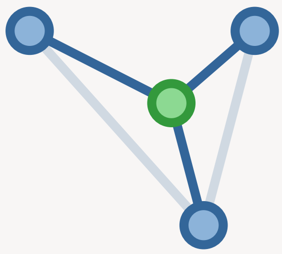
---

node degrees

number of edges adjacent (i.e., connected) to a given node

example:
node degree of the green node = 3

average n. d. in a network: k = 2E / N



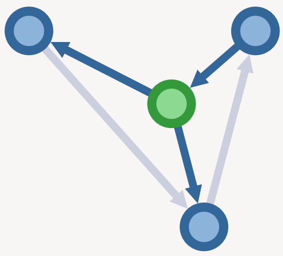
---

in- / out-degree

applies only to directed graphs

number of incoming / outgoing edges to / from a given node

example:
in degree of the green node = 1
out degree of the green node = 2



Cheatsheet: Graphs

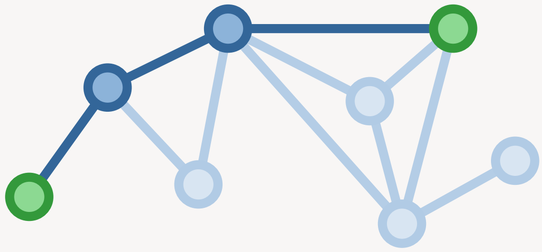
Graph Neural Networks and Transformers: Neural Networks for Sets

shortest path

the shortest path between two nodes

example:
shortest path between the green nodes = 3

for weighted graphs, the sum of the weights along the path is used



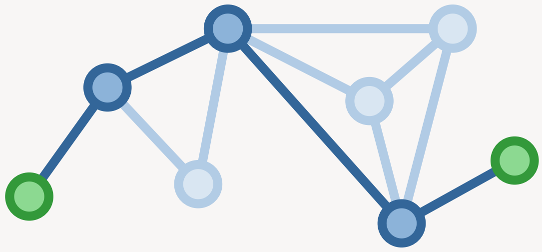
---

graph diameter

longest shortest path between two nodes

calculate the shortest path for each pair of nodes in the graph, the longest one of these defines the graph parameter

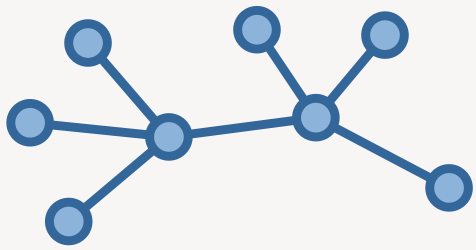
example:
graph diameter = 4



---

tree

a graph without any cycles



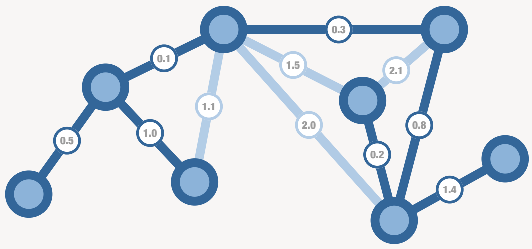
---

minimum spanning tree

applies to connected, weighted, undirected graphs

spanning tree = tree connecting all of the graph's nodes

MST = spanning tree with the smallest sum of edge weights

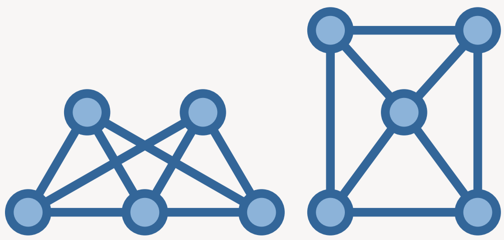


---

graph isomorphism

an edge-preserving bijection between two graphs G and H exists

in other words:
each node in G has a corresponding node in H, and any two nodes in G are adjacent if and only if the corresponding nodes in H are also adjacent



---

subgraph isomorphism

given two graphs G and H, a subgraph of G exists that is isomorphic to H