Complex Data Representation with Graphs

ACMS 80770: Deep Learning with Graphs

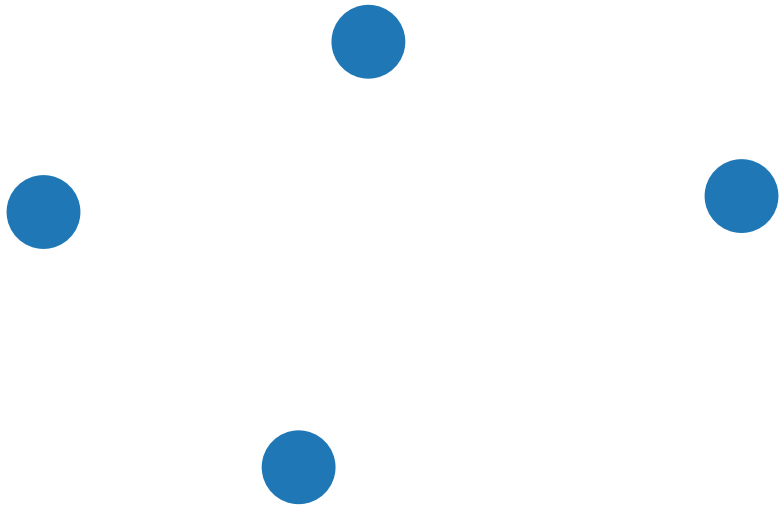
Instructor: Navid Shervani-Tabar

Department of Applied and Comp Math and Stats

UNIVERSITY OF NOTRE DAME

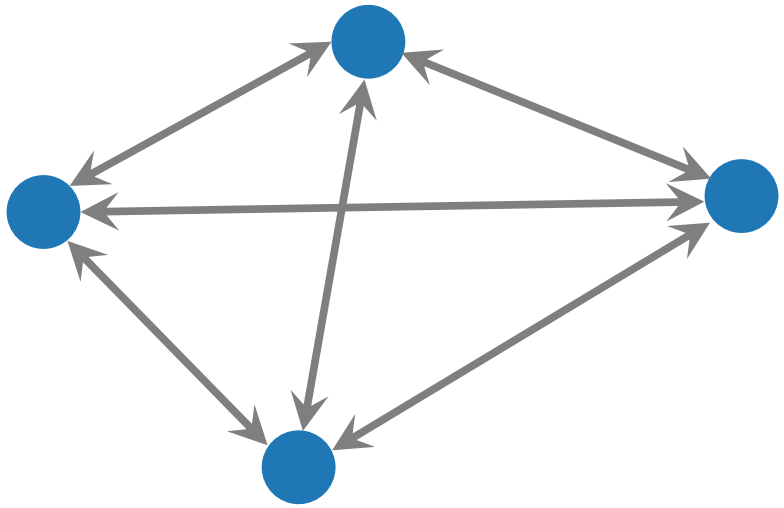
Data Representation

- Some data are naturally defined on a graph.
- multi-component systems.



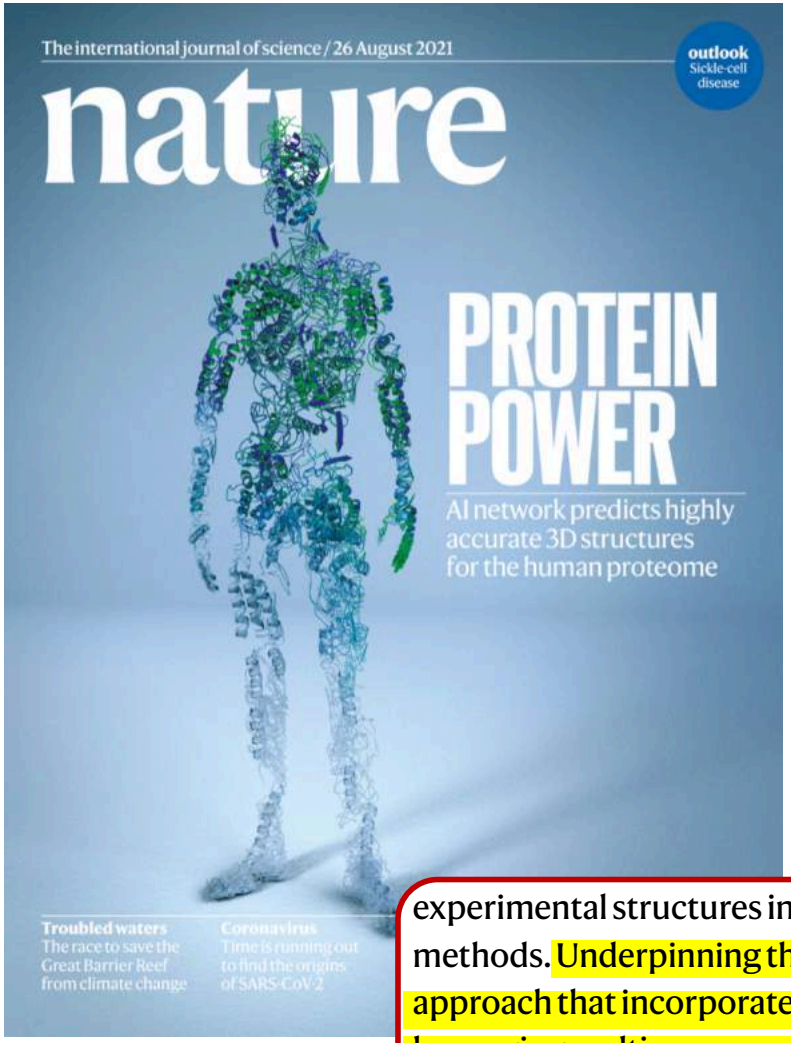
Data Representation

- Some data are naturally defined on a graph.
  - multi-component systems.



- Graph incorporates information about the structure of the data that resides on it.
  - Closeness, strength, or type of the relation between nodes.

Data Representation



Article

Highly accurate protein structure prediction with AlphaFold

https://doi.org/10.1038/s41586-021-03819-2
Received: 11 May 2021
Accepted: 12 July 2021
Published online: 15 July 2021
Open access
Check for updates

John Jumper[1,4✉], Richard Evans[1,4], Alexander Pritzel[1,4], Tim Green[1,4], Michael Figurnov[1,4], Olaf Ronneberger[1,4], Kathryn Tunyasuvunakool[1,4], Russ Bates[1,4], Augustin Žídek[1,4], Anna Potapenko[1,4], Alex Bridgland[1,4], Clemens Meyer[1,4], Simon A. A. Kohl[1,4], Andrew J. Ballard[1,4], Andrew Cowie[1,4], Bernardino Romera-Paredes[1,4], Stanislav Nikolov[1,4], Rishub Jain[1,4], Jonas Adler[1], Trevor Back[1], Stig Petersen[1], David Reiman[1], Ellen Clancy[1], Michal Zielinski[1], Martin Steinegger[2,3], Michalina Pacholska[1], Tamas Berghammer[1], Sebastian Bodenstein[1], David Silver[1], Oriol Vinyals[1], Andrew W. Senior[1], Koray Kavukcuoglu[1], Pushmeet Kohli[1] & Demis Hassabis[1,4✉]

Proteins are essential to life, and understanding their structure can facilitate a mechanistic understanding of their function. Through an enormous experimental effort[1–4], the structures of around 100,000 unique proteins have been determined[5], but this represents a small fraction of the billions of known protein sequences[6,7]. Structural coverage is bottlenecked by the months to years of painstaking effort required to determine a single protein structure. Accurate computational approaches are needed to address this gap and to enable large-scale structural bioinformatics. Predicting the three-dimensional structure that a protein will adopt based solely on its amino acid sequence—the structure prediction component of the 'protein folding problem'[8]—has been an important open research problem for more than 50 years[9]. Despite recent progress[10–14], existing methods fall far short of atomic accuracy, especially when no homologous structure is available. Here we provide the first computational method that can regularly predict protein structures with atomic accuracy even in cases in which no similar structure is known. We validated an entirely redesigned version of our neural network-based model, AlphaFold, in the challenging 14th Critical Assessment of protein Structure Prediction (CASP14)[15], demonstrating accuracy competitive with experimental structures in a majority of cases and greatly outperforming other methods. Underpinning the latest version of AlphaFold is a novel machine learning approach that incorporates physical and biological knowledge about protein structure, leveraging multi-sequence alignments, into the design of the deep learning algorithm.

The development of computational methods to predict three-dimensional (3D) protein structures from the protein sequence has proceeded along two complementary paths that focus on either the physical interactions or the evolutionary history. The physical interaction programme heavily integrates our understanding of molecular driving forces into either thermodynamic or kinetic simulation of protein physics[16] or statistical approximations thereof[17]. Although theoretically very appealing, this approach has proved highly challenging for even moderate-sized proteins due to the computational intractability of molecular simulation, the context dependence of protein stability

the steady growth of experimental protein structures deposited in the Protein Data Bank (PDB)[5], the explosion of genomic sequencing and the rapid development of deep learning techniques to interpret these correlations. Despite these advances, contemporary physical and evolutionary-history-based approaches produce predictions that are far short of experimental accuracy in the majority of cases in which a close homologue has not been solved experimentally and this has limited their utility for many biological applications.

In this study, we develop the first, to our knowledge, computational approach capable of predicting protein structures to near experimental

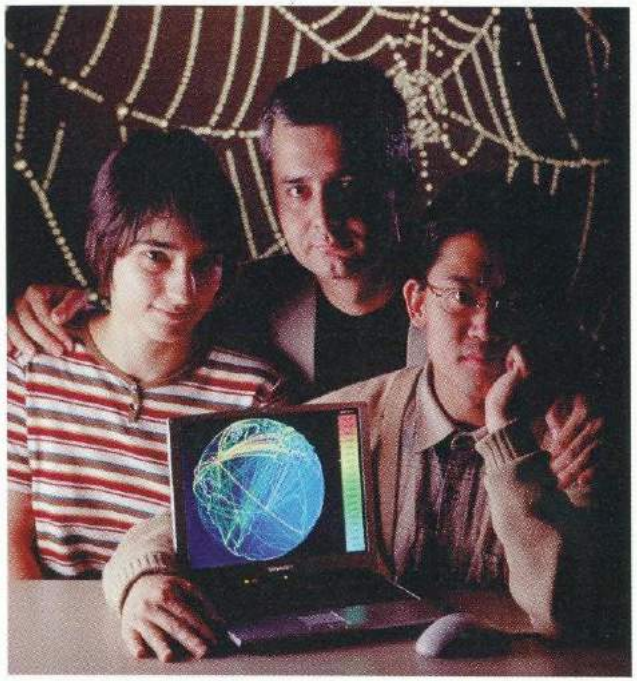
Fold that we
uly 2020;
different
ssment is
have not
blind test

*These
Anna

21 | 583

experimental structures in a majority of cases and greatly outperforming other methods. Underpinning the latest version of AlphaFold is a novel machine learning approach that incorporates physical and biological knowledge about protein structure, leveraging multi-sequence alignments, into the design of the deep learning algorithm.

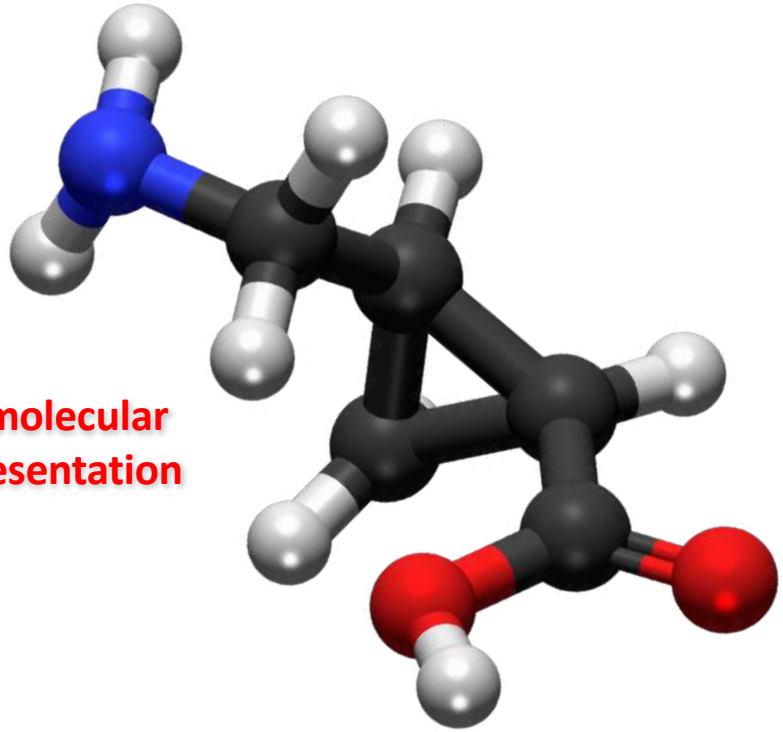
Data Representation



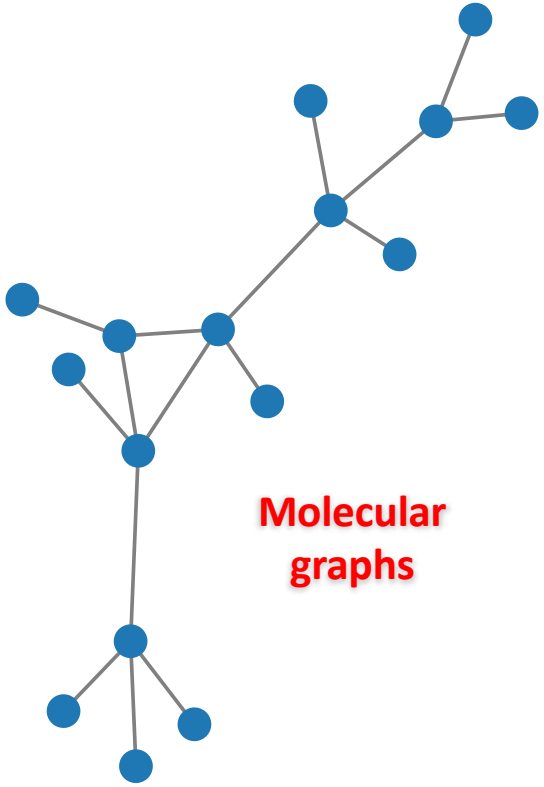
"The monolithic corn fields that surround Route 90 that connects Notre Dame to Chicago allowed my mind to contemplate a whole range of quixotic questions: [...] Could one indeed formulate a set of equations that could predict the future of a system as complex as the society? [...], Asimov kept pulling my mind back to the questions that never stopped fascinating me [...]: networks and complex systems."

Data Representation

- A molecular graph is a representation of the molecules and chemical compounds using graph theory.
- Atoms are shown by the nodes and edges symbolize the covalent bonds.



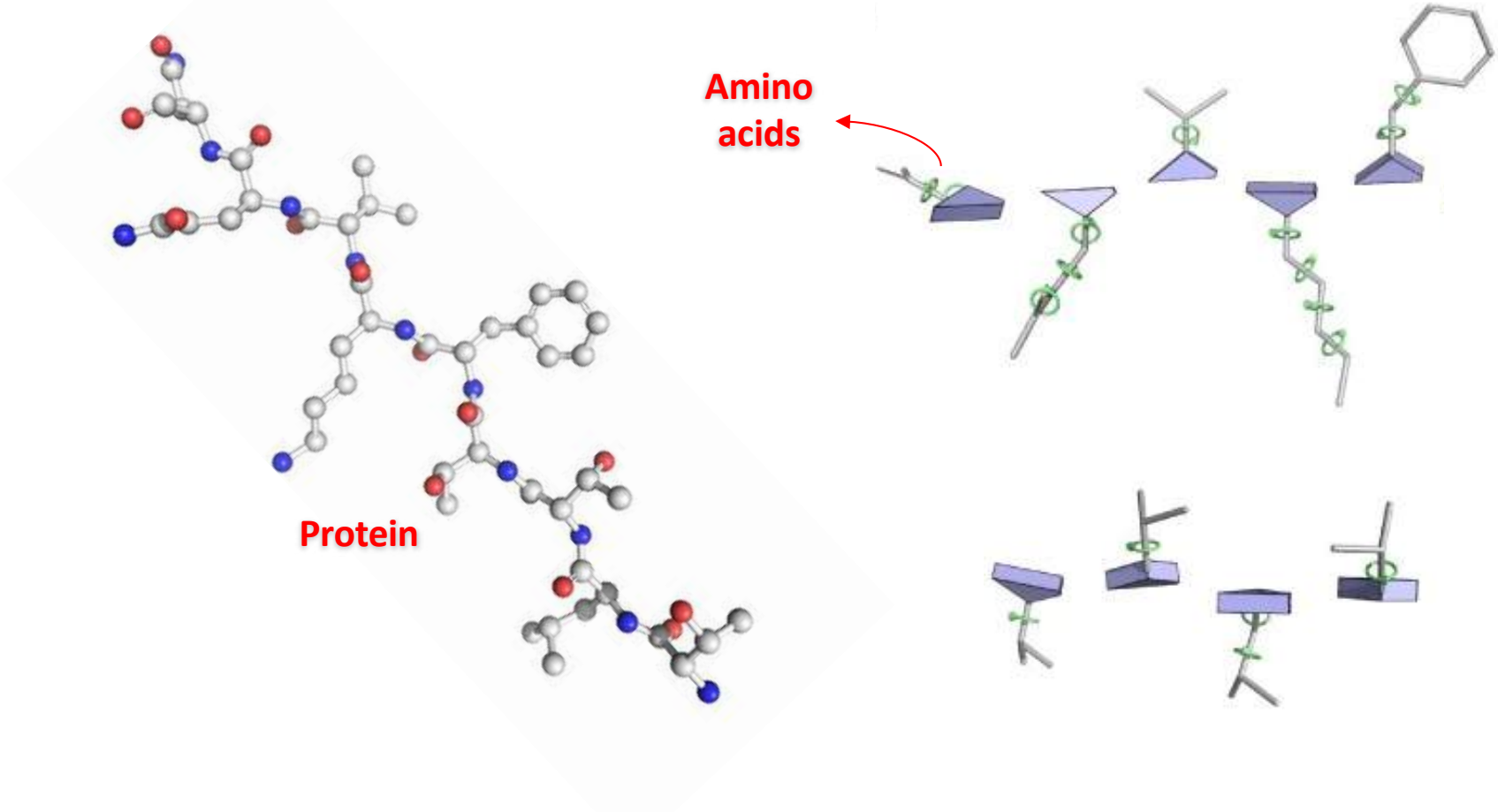
3D molecular representation



Molecular graphs

Data Representation

- Amino acids in a protein.
- Nodes are amino acids and edges show spatial proximity.

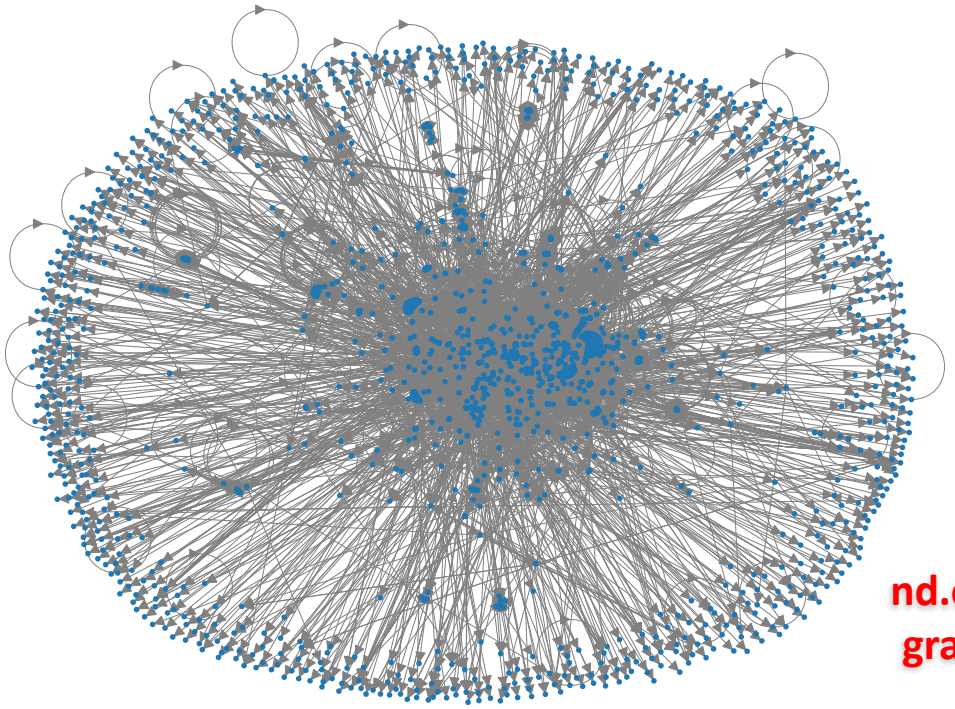


Data Representation

- Social Network.
- Nodes represent people and edges are type of interactions.

Data Representation

- Internet
- Computers are nodes and optical cables are edges.
- World Wide Web
- Websites are nodes and links are edges.



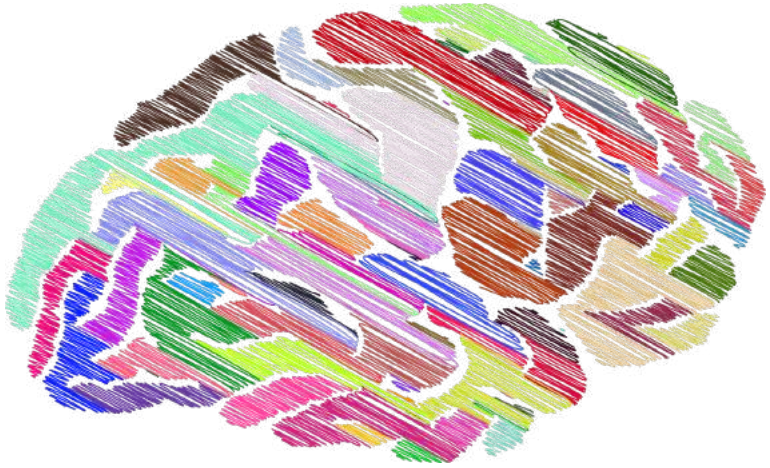
nd.edu graph

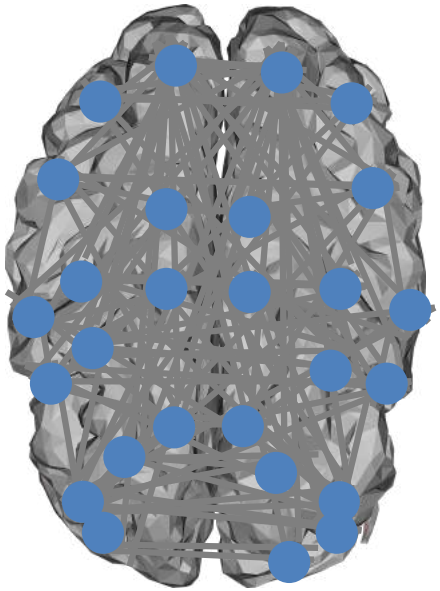
Data Representation

- Transportation network
- Edges are routs and nodes are connections.

Data Representation

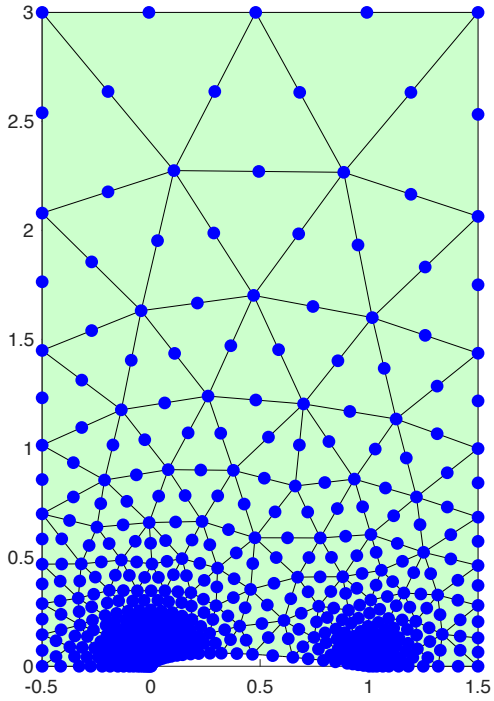
- fMRI.
- Nodes represent brain regions and edges are temporal activity correlation.





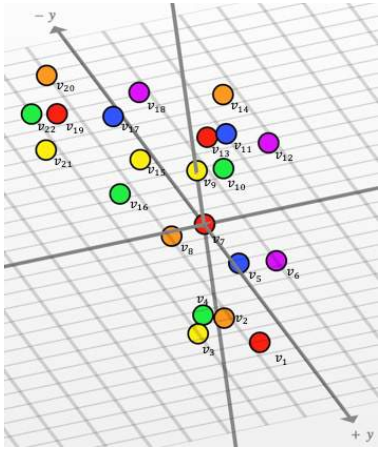
Data Representation

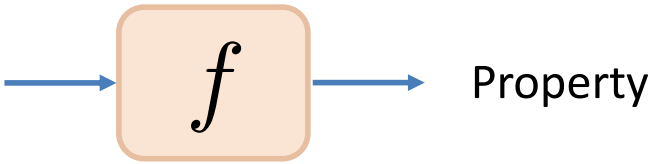
- Simulation:



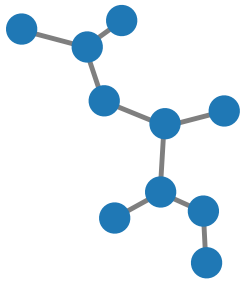
Motivation

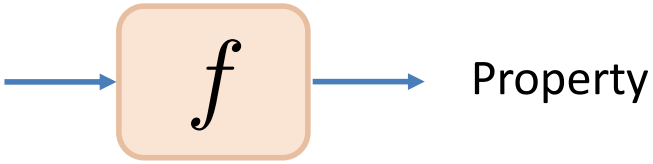
- Conventional ML models work on Euclidian space.





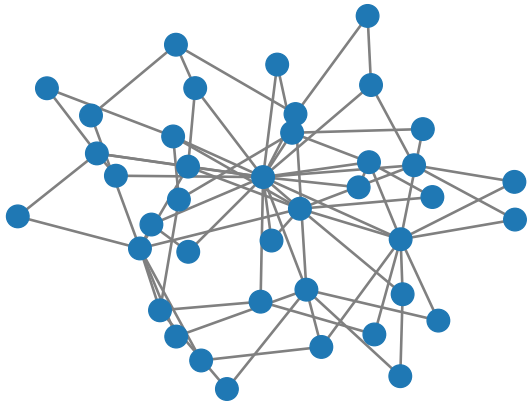
- Graphs contain richer topological information

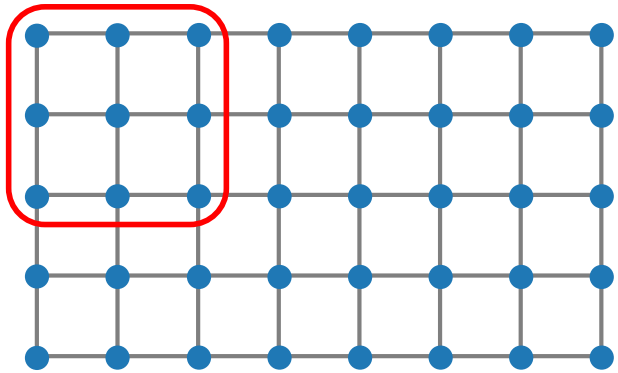




Motivation

- Conventional ML models are not designed for graph analysis.
- Convolution.

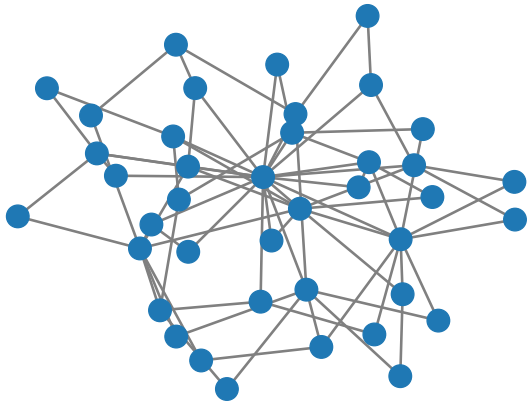


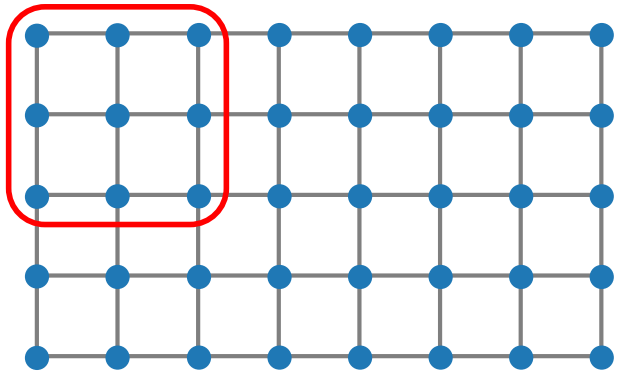


- Extend conventional tools to graph non-Euclidean domain.

Motivation

- Conventional ML models are not designed for graph analysis.
  - Convolution.





- Extend conventional tools to graph non-Euclidean domain.
  - Traditional methods
  - Learning on graphs
  - Graph-based deep learning

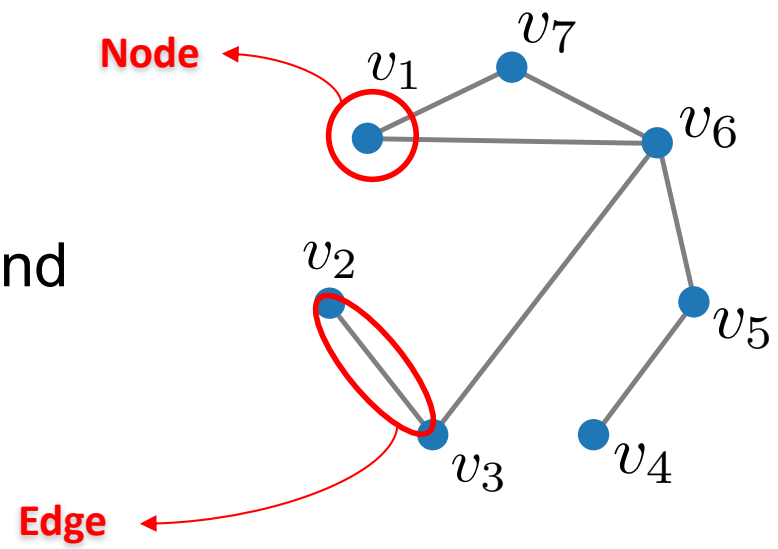
Definition

- Graphs are mathematical objects G that are consisted of a set of nodes V and a set of pairwise connections E, hence defining a graph as

$$G = (V, E)$$

where an edge ε_{mn} is defined with the pair of vertices (v_m, v_n).

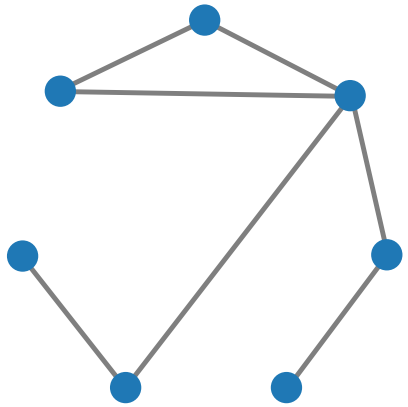
- v: Node, vertex, site, actor.
- (u, v): Edge, link, connection, bond tie, interaction.



Directed and Undirected Graphs

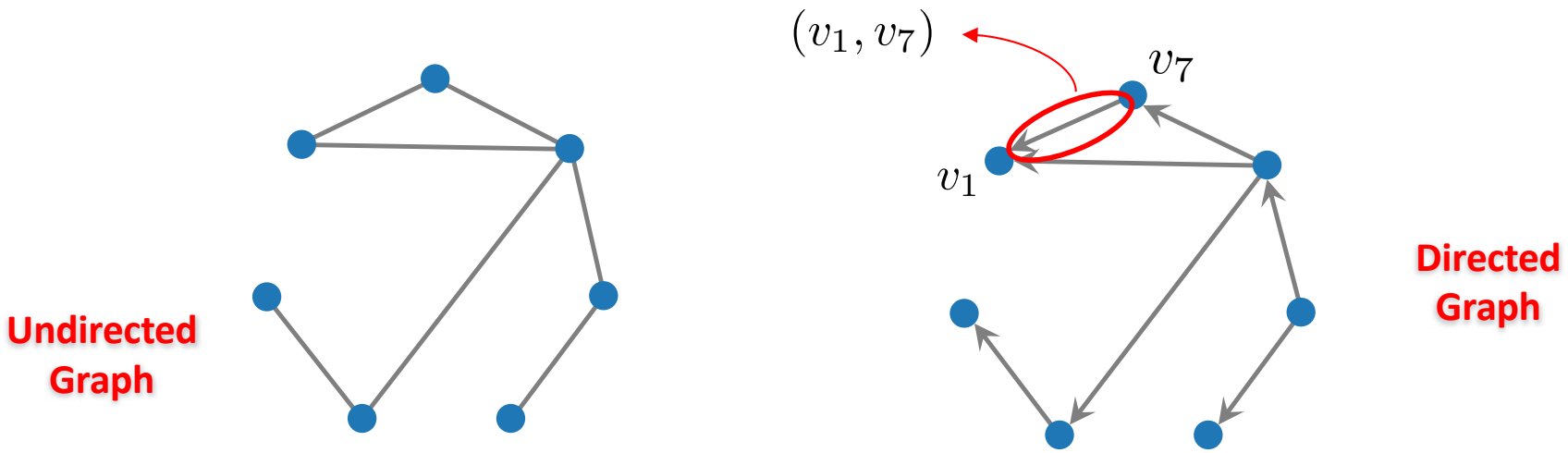
- In an undirected graph, there is no direction assigned to the edges. Mathematically put, $(v_1, v_2) \in E \leftrightarrow (v_2, v_1) \in E$.

Undirected Graph



Directed and Undirected Graphs

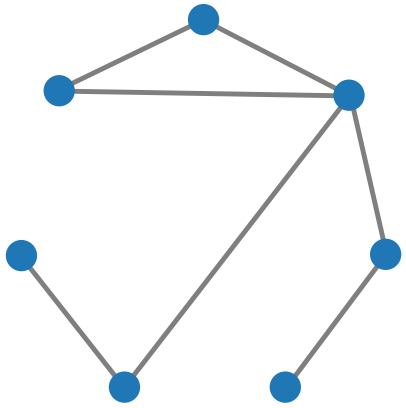
- In an undirected graph, there is no direction assigned to the edges. Mathematically put, $(v_1, v_2) \in E \leftrightarrow (v_2, v_1) \in E$.
- A directed graph or digraph is a graph where each edge $(v_1, v_2) \in E$ starts from one node $v_2 \in V$ and ends at another node $v_1 \in V$.
  - E.g. WWW, citation graph.



Adjacency matrix

- The underlying structure of the graph is typically demonstrated using a matrix representation.
- For a graph $G = (V, E)$ with $|V|$ nodes, an adjacency matrix A is a $|V| \times |V|$ square matrix such that

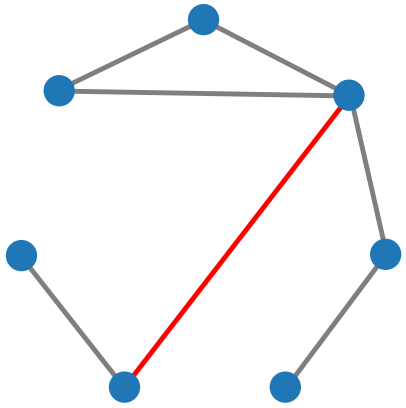
$$\mathbf{A}_{mn} = \begin{cases} 1 & \text{if } \varepsilon_{mn} \in E \\ 0 & \text{otherwise.} \end{cases}$$

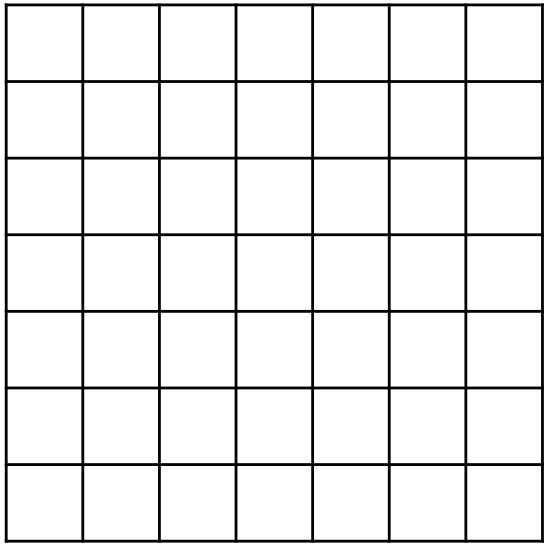


Adjacency matrix

- The underlying structure of the graph is typically demonstrated using a matrix representation.
- For a graph $G = (V, E)$ with $|V|$ nodes, an adjacency matrix A is a $|V| \times |V|$ square matrix such that

$$\mathbf{A}_{mn} = \begin{cases} 1 & \text{if } \varepsilon_{mn} \in E \\ 0 & \text{otherwise.} \end{cases}$$

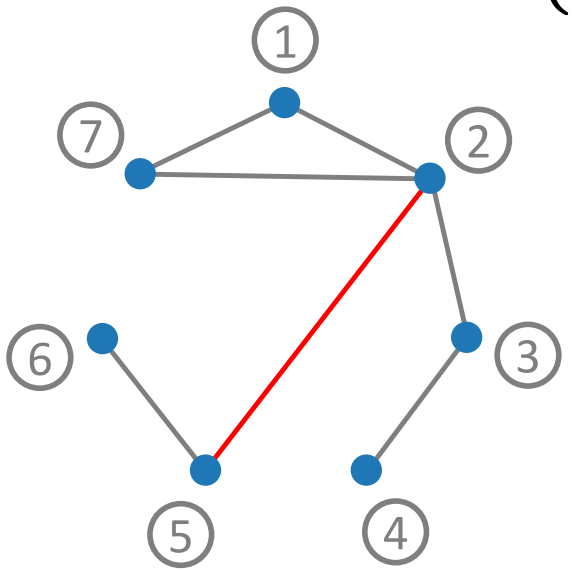


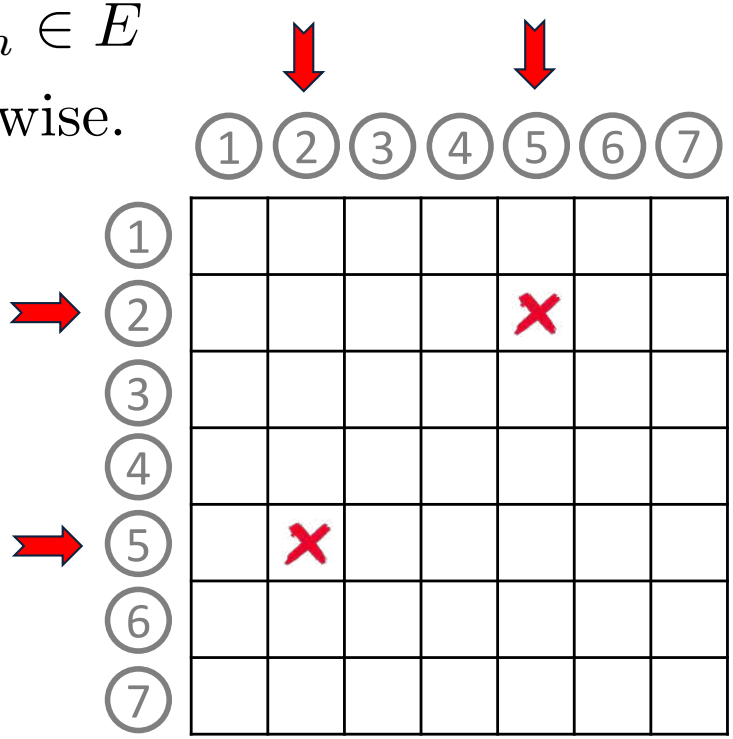


Adjacency matrix

- The underlying structure of the graph is typically demonstrated using a matrix representation.
- For a graph $G = (V, E)$ with $|V|$ nodes, an adjacency matrix A is a $|V| \times |V|$ square matrix such that

$$\mathbf{A}_{mn} = \begin{cases} 1 & \text{if } \varepsilon_{mn} \in E \\ 0 & \text{otherwise.} \end{cases}$$

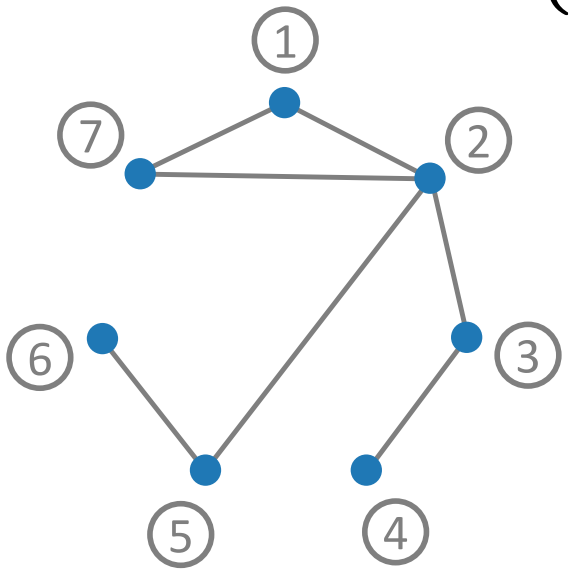




Adjacency matrix

- The underlying structure of the graph is typically demonstrated using a matrix representation.
- For a graph $G = (V, E)$ with $|V|$ nodes, an adjacency matrix A is a $|V| \times |V|$ square matrix such that

$$\mathbf{A}_{mn} = \begin{cases} 1 & \text{if } \varepsilon_{mn} \in E \\ 0 & \text{otherwise.} \end{cases}$$

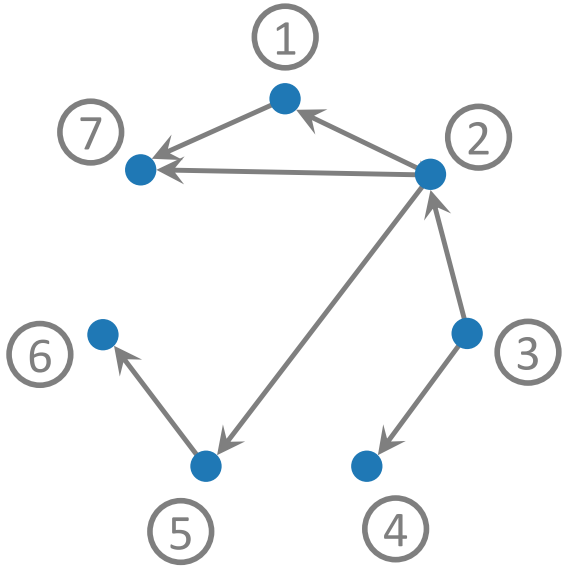


	1	2	3	4	5	6	7
1		1					1
2	1		1		1		1
3		1		1			
4			1				
5		1				1	
6					1		
7	1	1					

Adjacency matrix

- For a directed graph, the adjacency matrix is a $|V|\times|V|$ non-symmetric square matrix such that

$$\mathbf{A}_{mn} = \begin{cases} 1 & \text{if } (v_m, v_n) \in E \\ 0 & \text{otherwise} \end{cases}$$

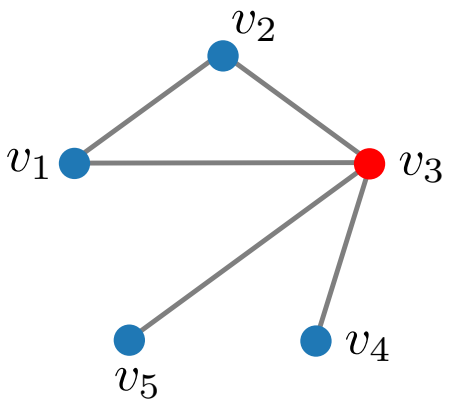


	1	2	3	4	5	6	7
1		1					
2			1				
3							
4			1				
5		1					
6					1		
7	1	1					

Degree

- **Node Degree:** The degree d_m of a vertex v_m is defined as the number of edges connected to the vertex.

$$d_m = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

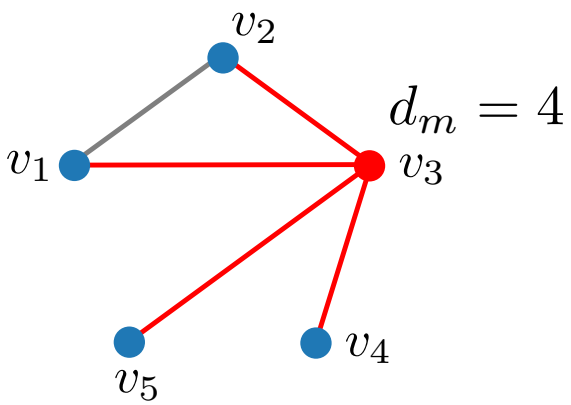


	1	1		
1		1		
1	1		1	1
		1		
		1		

Degree

- **Node Degree:** The degree d_m of a vertex v_m is defined as the number of edges connected to the vertex.

$$d_m = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

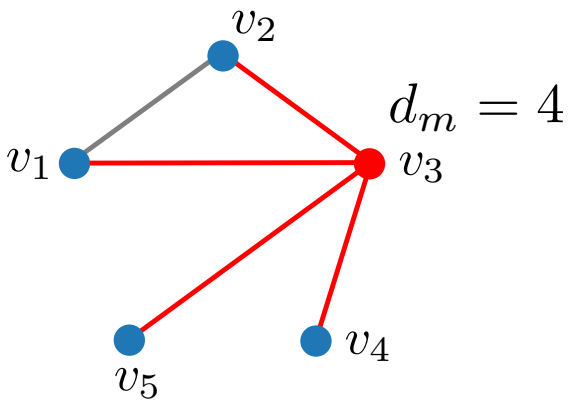


	1	1		
1		1		
1	1		1	1
		1		
		1		

Degree

- **Node Degree:** The degree d_m of a vertex v_m is defined as the number of edges connected to the vertex.

$$d_m = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

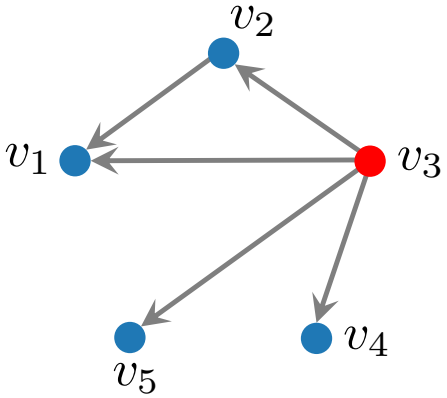


	1	1		
1		1		
1	1		1	1
		1		
		1		

- In directed graphs, node degree is distinguished by its direction.

$$d_m^{(out)} = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

$$d_m^{(in)} = \sum_{n=1}^{|V|} \boldsymbol{A}_{nm}$$

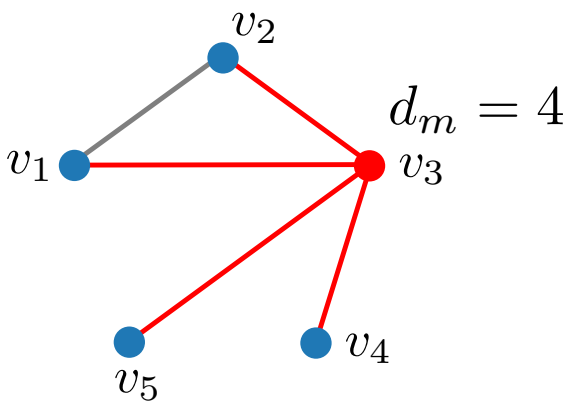


	1	1		
		1		
		1		
		1		

Degree

- **Node Degree:** The degree d_m of a vertex v_m is defined as the number of edges connected to the vertex.

$$d_m = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

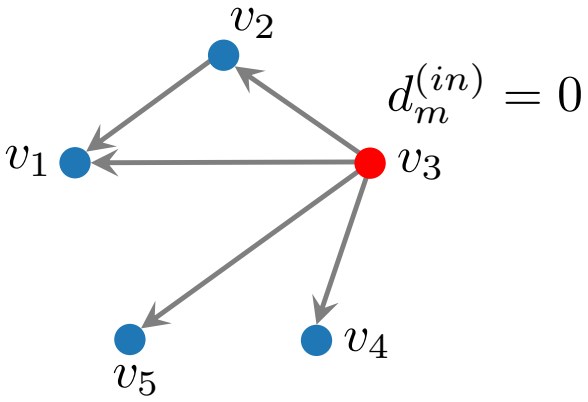


	1	1		
1		1		
1	1		1	1
		1		
		1		

- In directed graphs, node degree is distinguished by its direction.

$$d_m^{(out)} = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

$$d_m^{(in)} = \sum_{n=1}^{|V|} \boldsymbol{A}_{nm}$$

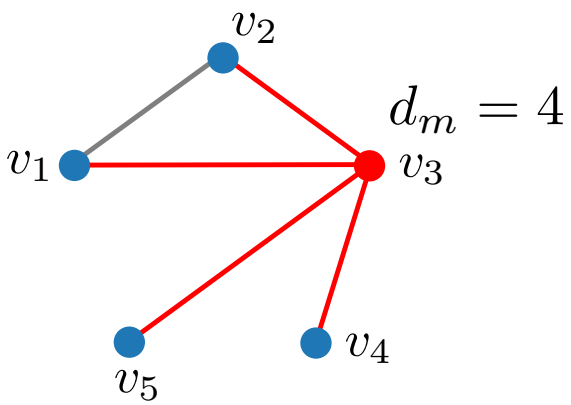


	1	1		
		1		
		1		
		1		

Degree

- **Node Degree:** The degree d_m of a vertex v_m is defined as the number of edges connected to the vertex.

$$d_m = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

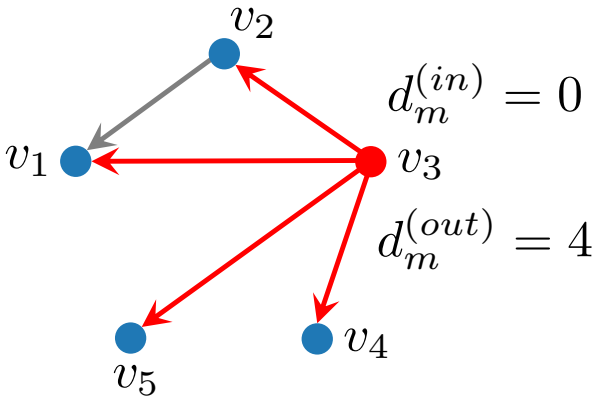


	1	1		
1		1		
1	1		1	1
		1		
		1		

- In directed graphs, node degree is distinguished by its direction.

$$d_m^{(out)} = \sum_{n=1}^{|V|} \boldsymbol{A}_{mn}$$

$$d_m^{(in)} = \sum_{n=1}^{|V|} \boldsymbol{A}_{nm}$$

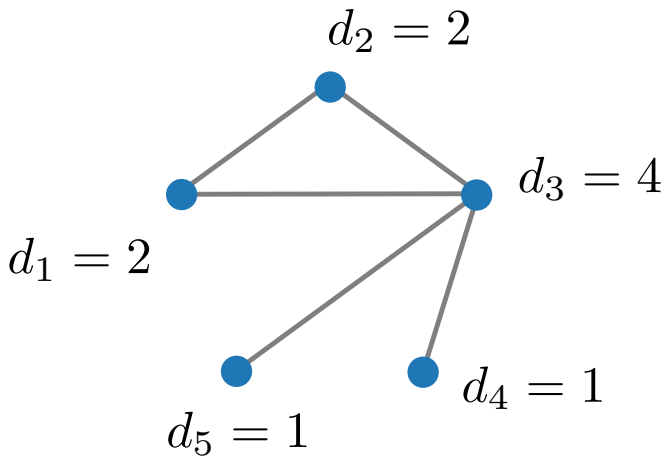


	1	1		
		1		
		1		
		1		

Degree

- **Degree Matrix:** For an undirected graph, we can define vertex-degree matrix D as a diagonal matrix with

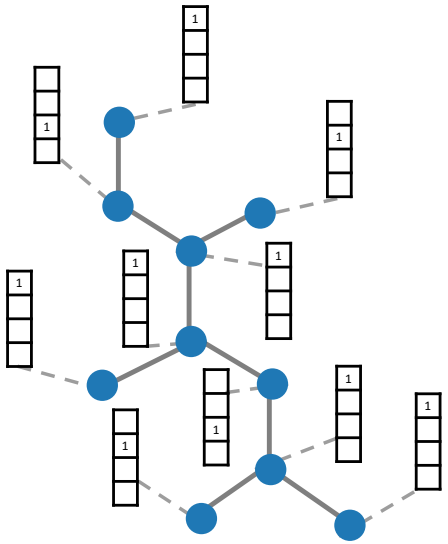
$$D_{mn} := \begin{cases} d_m & \text{if } m = n \\ 0 & \text{if } m \neq n \end{cases}$$



2				
	2			
		4		
			1	
				1

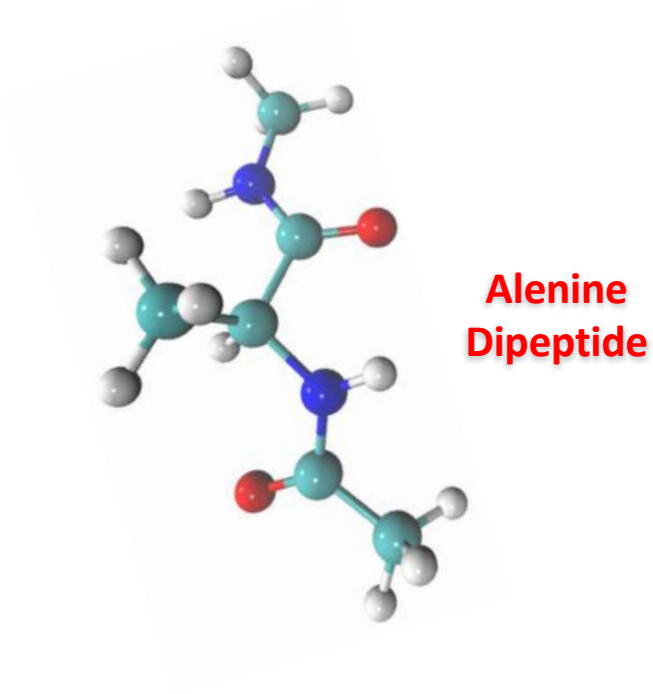
Node Attributes

- **Node attributes:** Graphs provide a domain for node information to reside. These information can be discrete categories of continuous values.



C:
O:
N:
H:

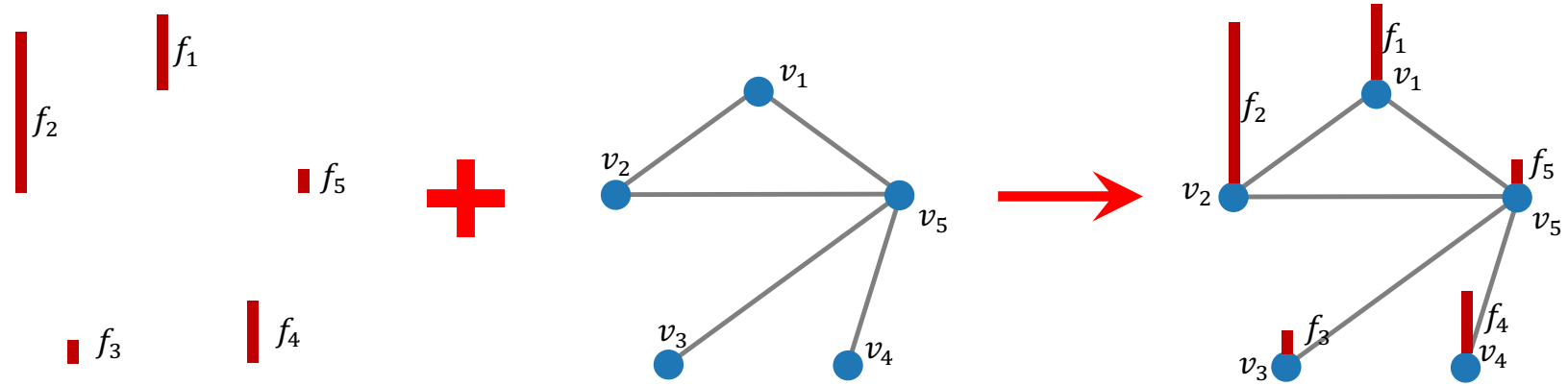
Node Features



Alenine Dipeptide

Node Attributes

- **Node attributes:** Graphs provide a domain for node information to reside. These information can be discrete categories of continuous values.

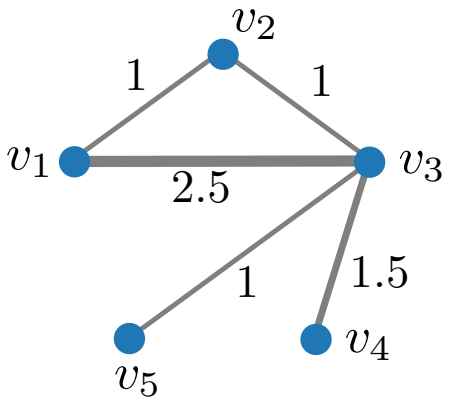


Weighted Graphs

- **Weighted graphs** consist of a set of nodes, a set of pairwise connections, and their corresponding set of weights W

$$G = (V, E, W)$$

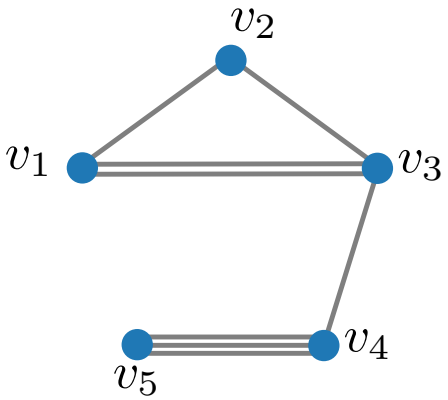
- Relationship strength, spatial proximity.



	1	2.5		
1		1		
2.5	1		1	1.5
		1		
		1.5		

Multi-Relational Graphs

- **Multi-relational graphs** have different types of edges.
- Each edge is defined by $(u, t, v) \in \mathrm{E}$, where t is the type of the edge



	1			
1		1		
	1		1	
		1		

A_I

		1		
1				

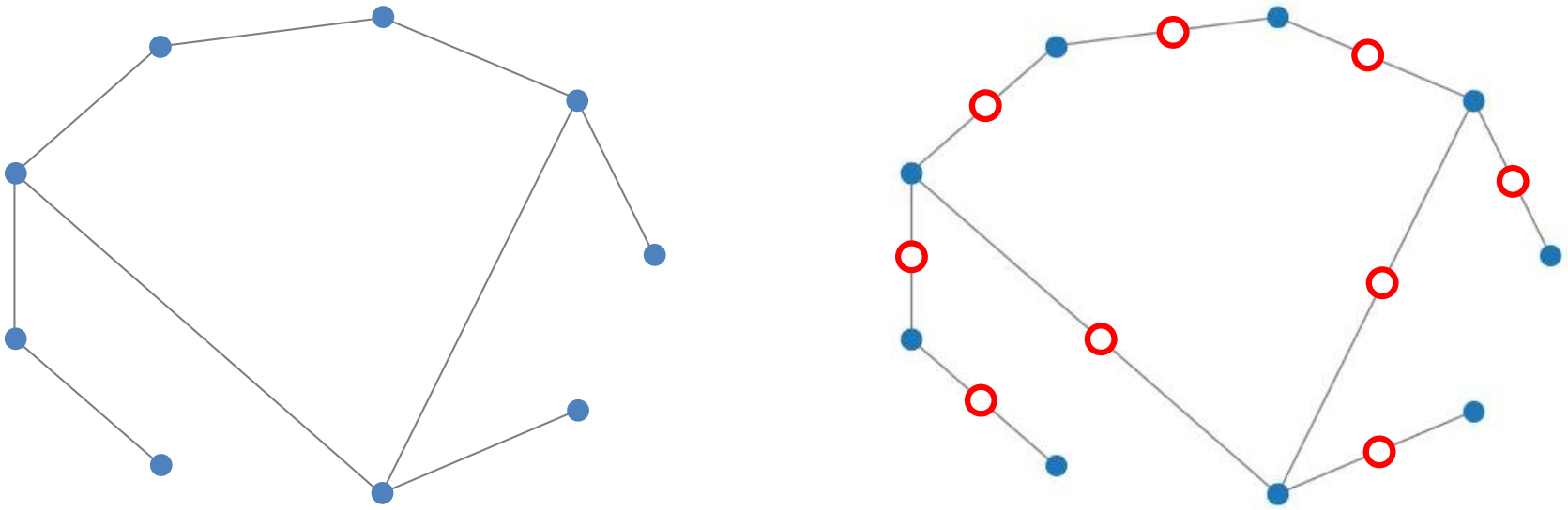
A_{II}

				1
			1	

A_{III}

Line Graph

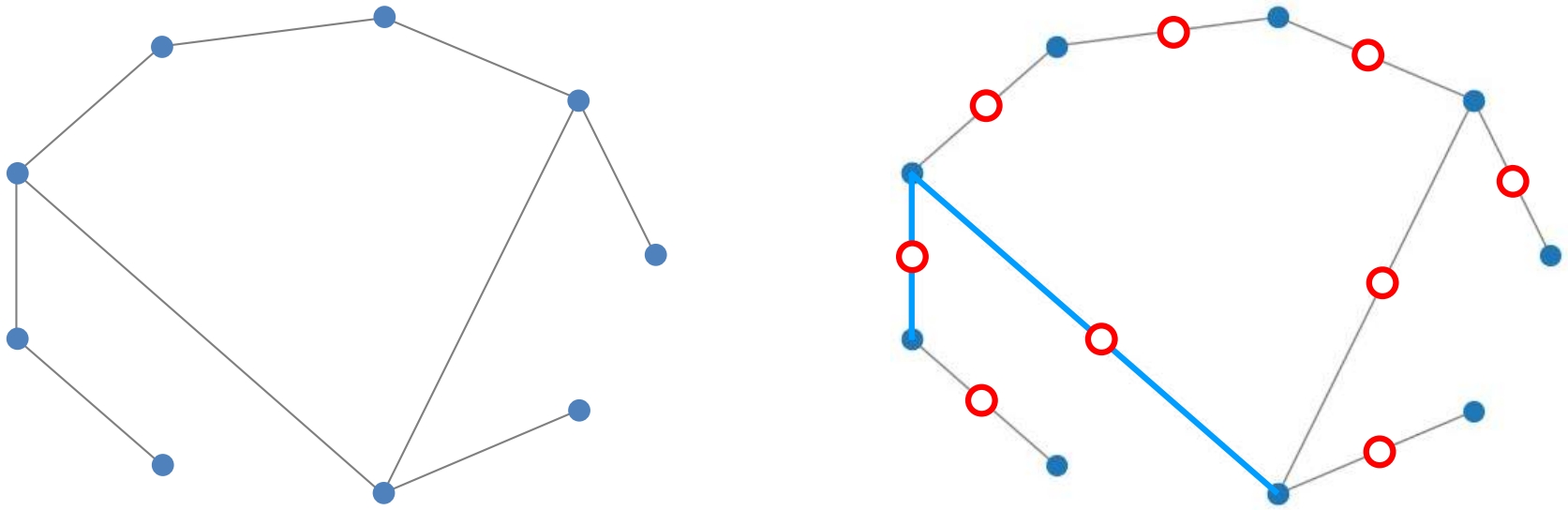
- For a given graph G, a line graph I is defined by assigning a node for every edge in graph G.



Line Graph

- For a given graph G, a line graph I is defined by assigning a node for every edge in graph G.

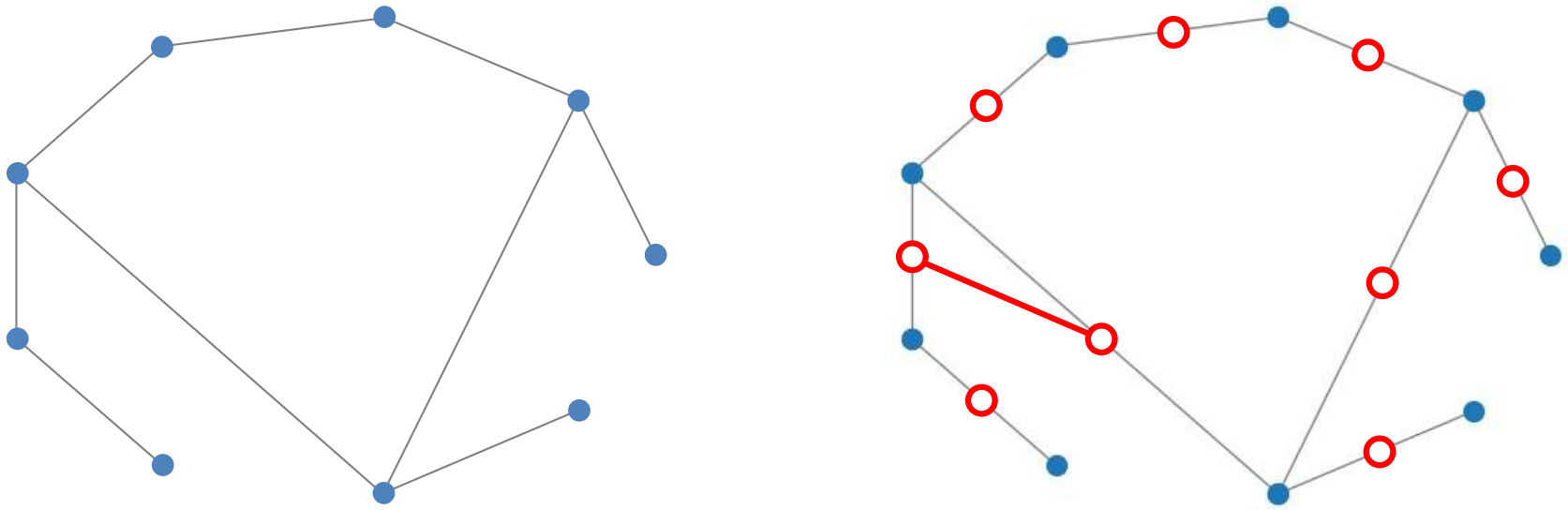
- Two nodes of the line graph I are connected if the corresponding edges in the graph G are neighbors.



Line Graph

- For a given graph G, a line graph I is defined by assigning a node for every edge in graph G.

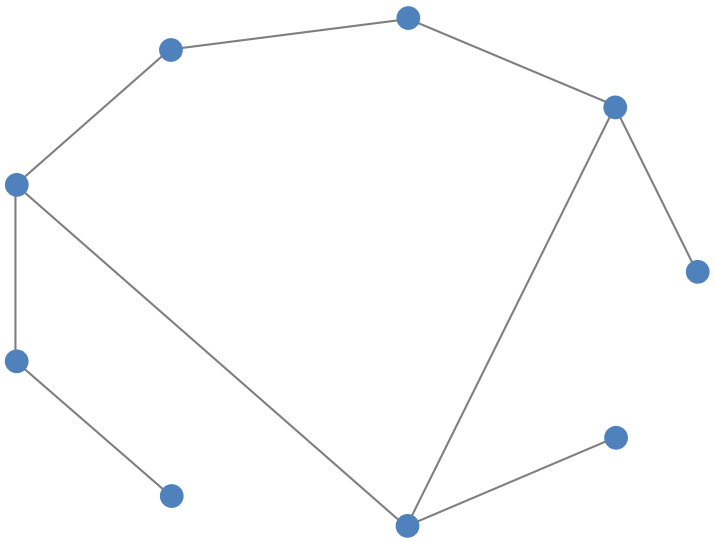
- Two nodes of the line graph I are connected if the corresponding edges in the graph G are neighbors.

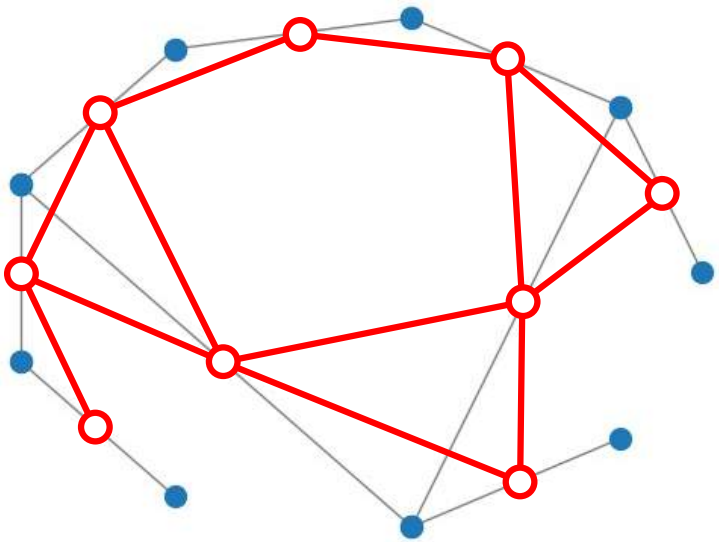


Line Graph

- For a given graph G, a line graph I is defined by assigning a node for every edge in graph G.

- Two nodes of the line graph I are connected if the corresponding edges in the graph G are neighbors.

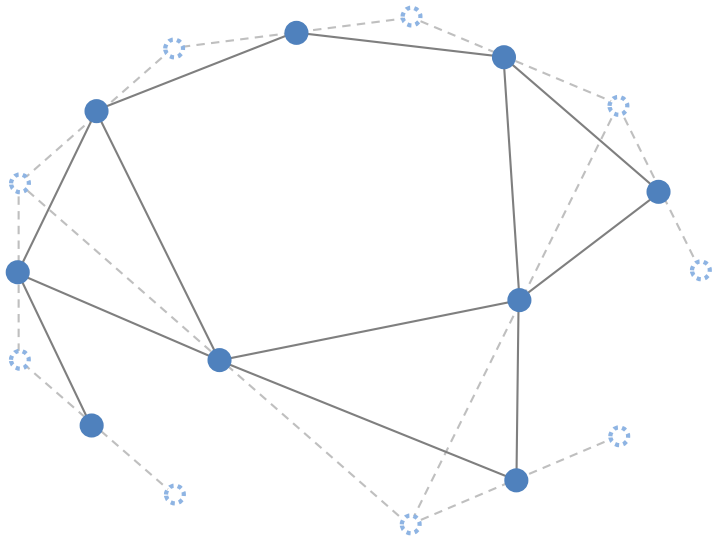




Line Graph

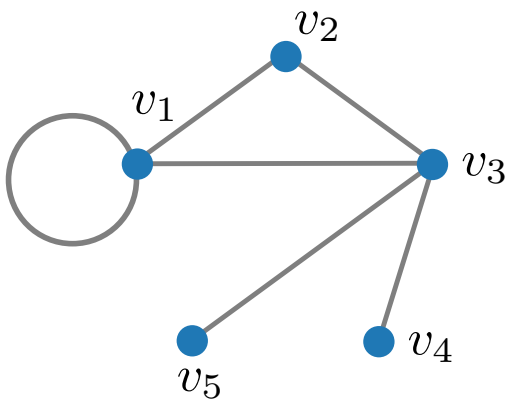
- For a given graph G, a line graph I is defined by assigning a node for every edge in graph G.

- Two nodes of the line graph I are connected if the corresponding edges in the graph G are neighbors.



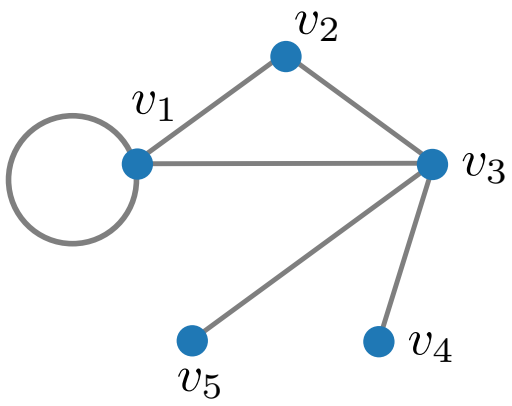
Self-loop

- **Self-loop** or **self-edge** is an edge starting and ending at the same node.
- Self-loop represents interaction of a node with itself.



Self-loop

- **Self-loop** or **self-edge** is an edge starting and ending at the same node.
- Self-loop represents interaction of a node with itself.



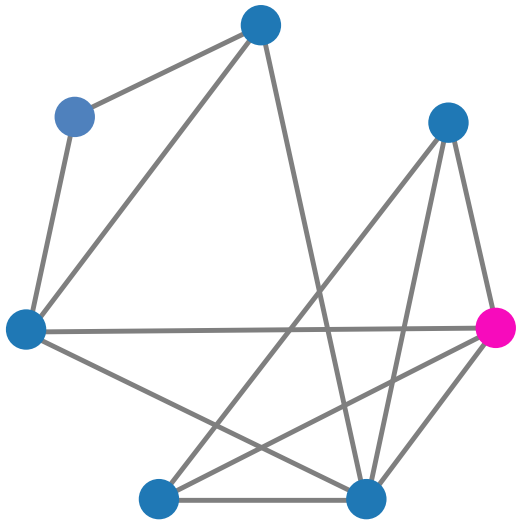
2	1	1		
1		1		
1	1		1	1
		1		
		1		

- A self-edge to node v_i is represented by $A_{ii} = 2$ on the adjacency matrix.

Neighbors

- Neighbors of node $v \in V$ is the set of nodes that are connected to v through an edge, i.e.

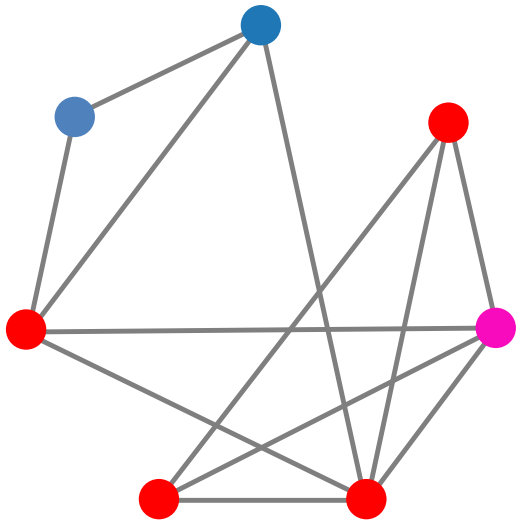
$$N(v) = \{u \in V | (u, v) \in E\}$$



Neighbors

- Neighbors of node $v \in V$ is the set of nodes that are connected to v through an edge, i.e.

$$N(v) = \{u \in V | (u, v) \in E\}$$

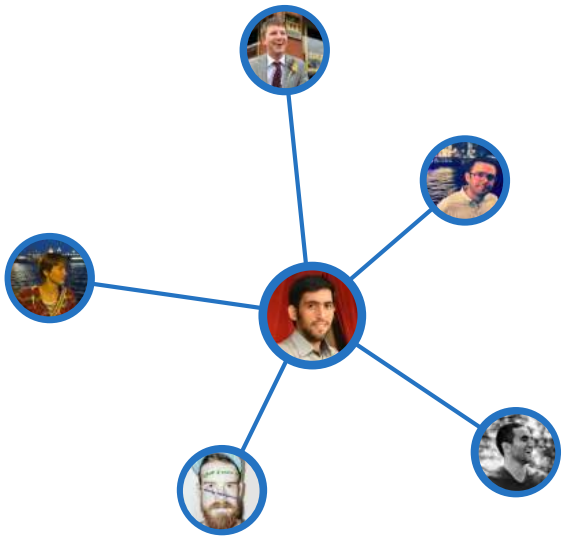


K-hop Neighborhood

- The k-hop neighborhood of v is the set of vertices that are reachable from v in k hops or fewer.

K-hop Neighborhood

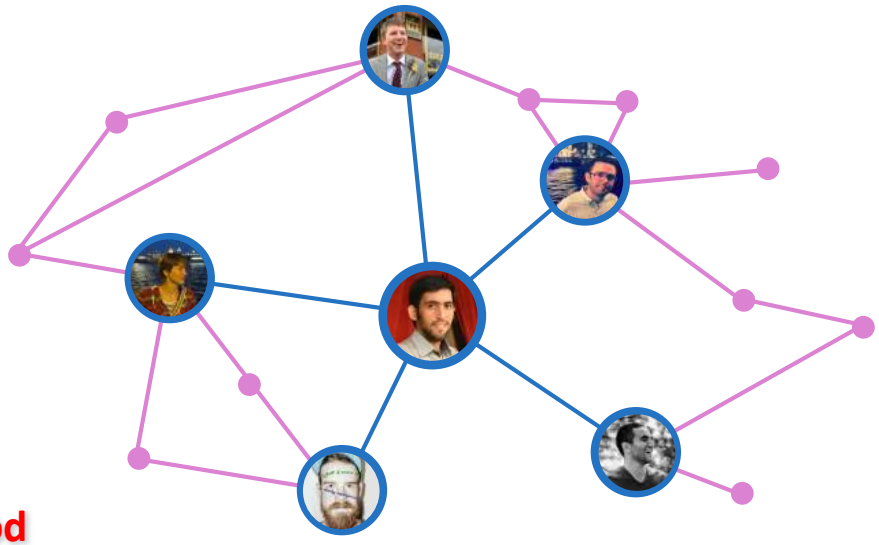
- The k-hop neighborhood of v is the set of vertices that are reachable from v in k hops or fewer.



1-hop Neighborhood

K-hop Neighborhood

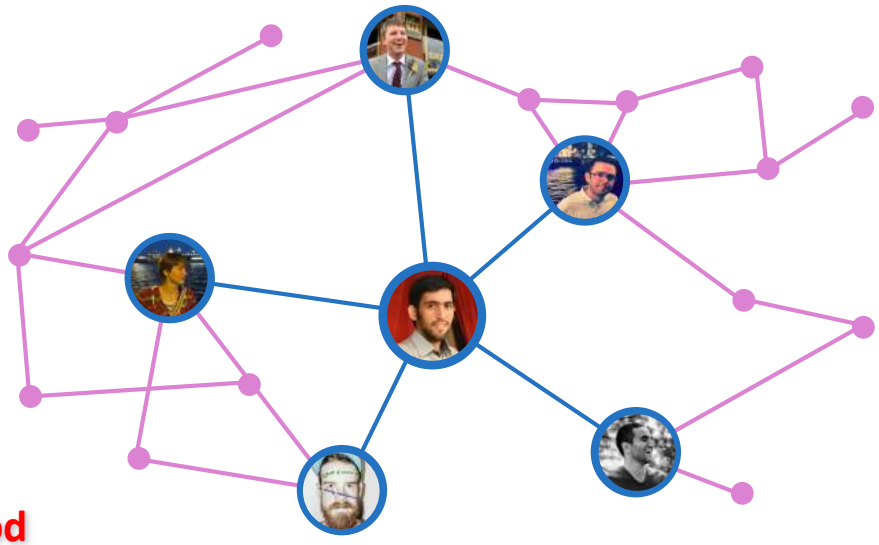
- The k-hop neighborhood of v is the set of vertices that are reachable from v in k hops or fewer.



2-hop Neighborhood

K-hop Neighborhood

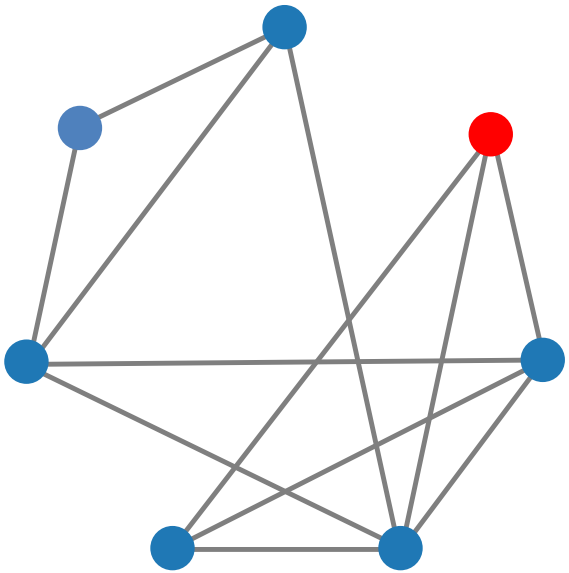
- The k-hop neighborhood of v is the set of vertices that are reachable from v in k hops or fewer.



3-hop Neighborhood

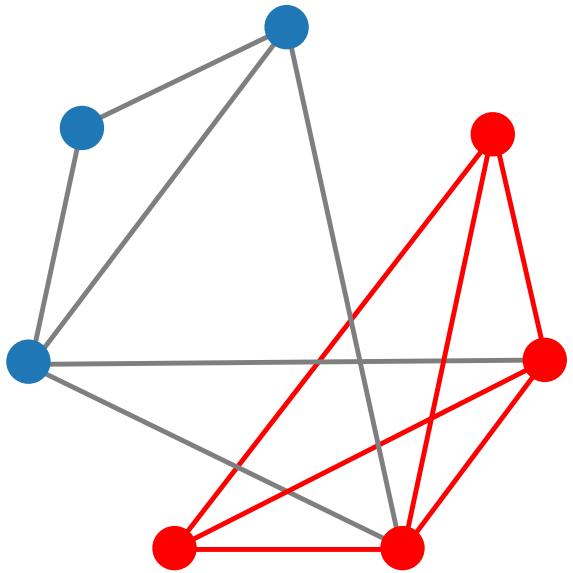
Ego Graph

- The ego graph of a node is the subgraph containing the node, it's neighbors, and all the connections between them



Ego Graph

- The ego graph of a node is the subgraph containing the node, it's neighbors, and all the connections between them

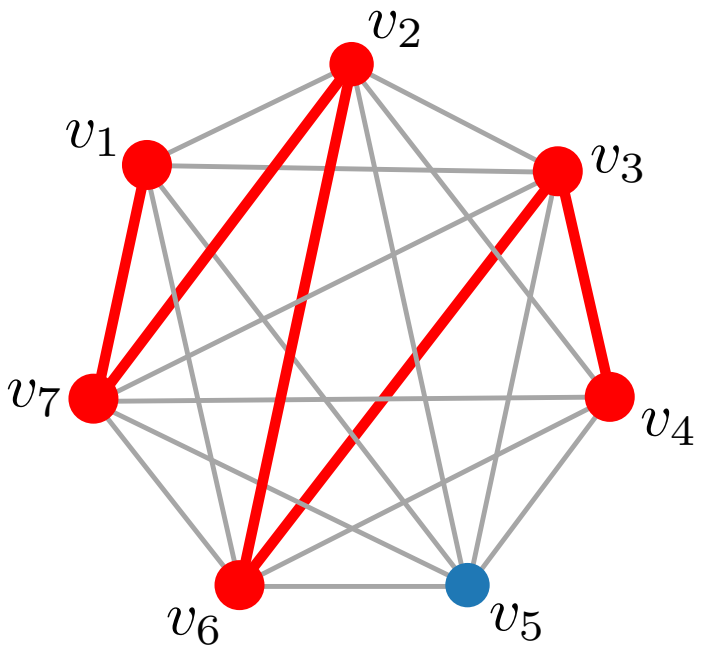


Path

- In graph terminology, a path is defined as a sequence of unique connected edges and vertices.

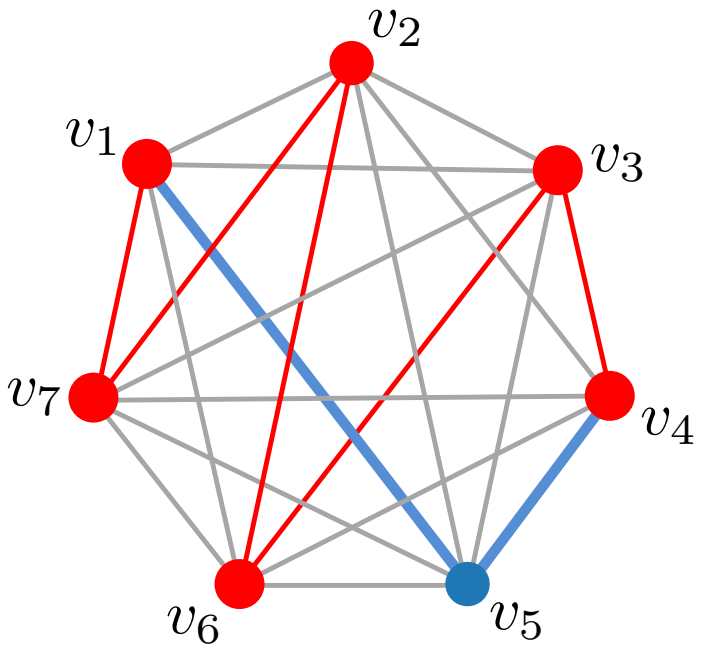
$$P : \{(v_i, v_j), (v_j, v_k), \ldots, (v_l, v_m)\}$$

- Route: $v_1 \rightarrow v_7 \rightarrow v_2 \rightarrow v_6 \rightarrow v_3 \rightarrow v_4$.
- Length: $n = 5$



Shortest Path and Distance

- The shortest path or the geodesic path between v_i and v_j is the path that connects v_i and v_j with the fewest number of edges.

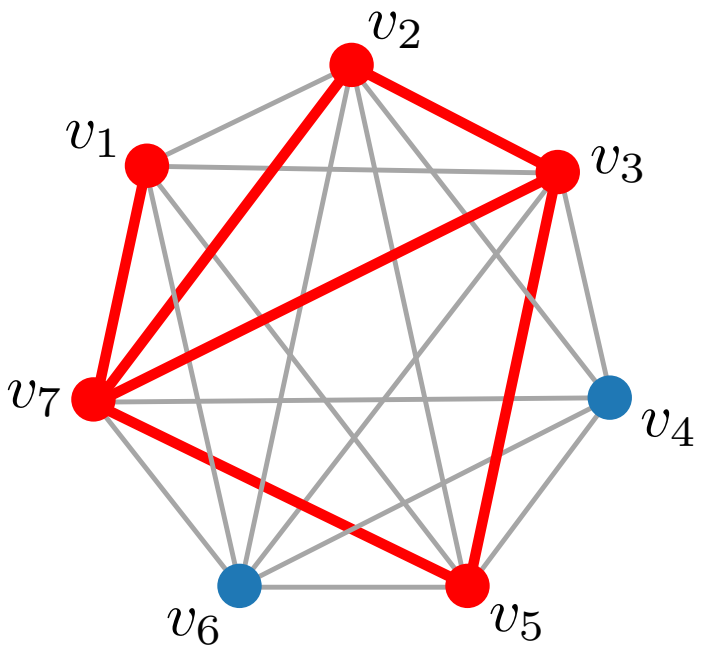


- Distance between node v_i and v_j is the length of the shortest path between v_i and v_j.

Walk

- Any sequence of connected pair of nodes on the graph is a walk.
- A walk can revisit an edge or node more than once.
- Route: $v_1 \rightarrow v_7 \rightarrow v_5 \rightarrow v_3 \rightarrow v_7 \rightarrow v_2 \rightarrow v_3 \rightarrow v_5$.

: Revisited

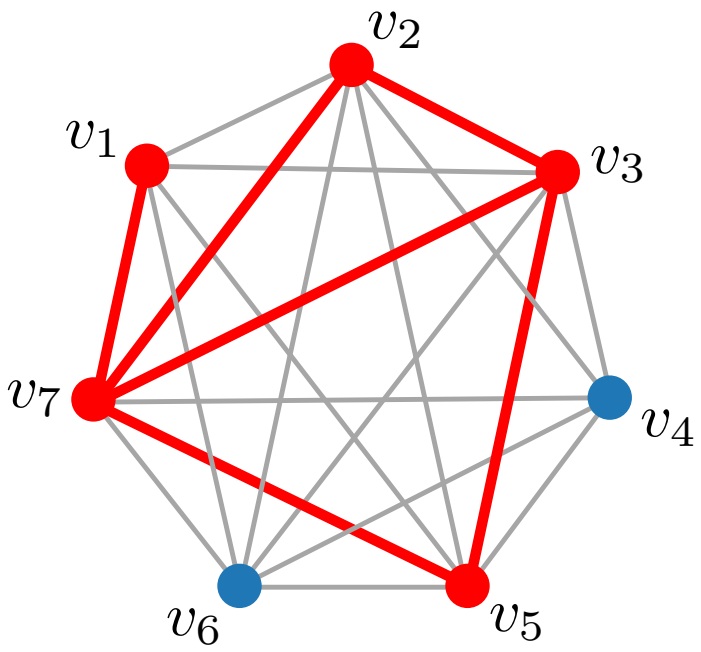


Walk

- Any sequence of connected pair of nodes on the graph is a walk.
- A walk can revisit an edge or node more than once.
  - Route: $v_1 \rightarrow v_7 \rightarrow v_5 \rightarrow v_3 \rightarrow v_7 \rightarrow v_2 \rightarrow v_3 \rightarrow v_5$.

: Revisited

- Length of a walk is the number of the edges visited along the walk.
  - Length = 7



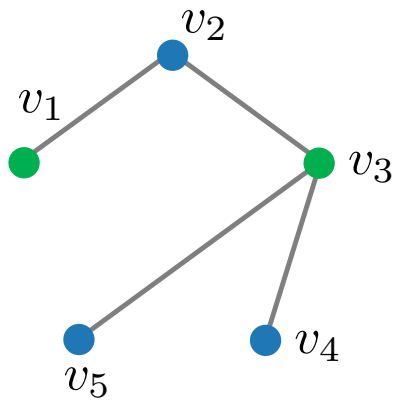
Walk

- The number of walks of length 1 between v_i and v_j is denoted by A_{ij}.

$$N_{ij}^{(1)} = A_{ij}$$

- Number of walks of size 2 from v_i to v_j that go through $v_k \in V$ is

$$N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik} A_{kj}$$
$$= \left[A^2\right]_{ij}$$



	1	1		
1		1		
1	1		1	1
		1		
		1		

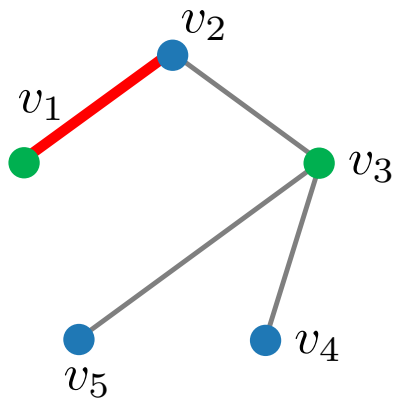
Walk

- The number of walks of length 1 between v_i and v_j is denoted by A_{ij}.

$$N_{ij}^{(1)} = A_{ij}$$

- Number of walks of size 2 from v_i to v_j that go through $v_k \in V$ is

$$N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik} A_{kj}$$
$$= \left[A^2\right]_{ij}$$



	1	1		
1		1		
1	1		1	1
		1		
		1		

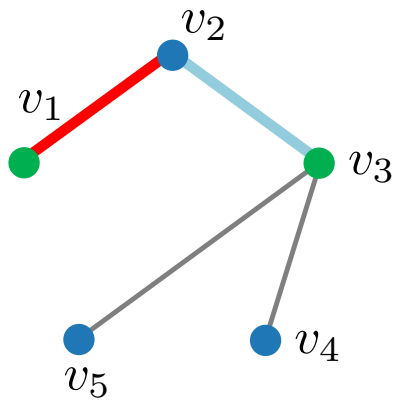
Walk

- The number of walks of length 1 between v_i and v_j is denoted by A_{ij}.

$$N_{ij}^{(1)} = A_{ij}$$

- Number of walks of size 2 from v_i to v_j that go through $v_k \in V$ is

$$N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik} A_{kj}$$
$$= \left[A^2\right]_{ij}$$



	1	1		
1		1		
1	1		1	1
		1		
		1		

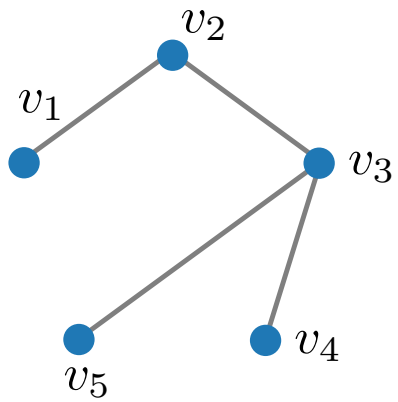
Walk

- The number of walks of length 1 between v_i and v_j is denoted by A_{ij}.

$$N_{ij}^{(1)} = A_{ij}$$

- Number of walks of size 2 from v_i to v_j that go through $v_k \in V$ is

$$N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik} A_{kj} = \left[A^2\right]_{ij}$$



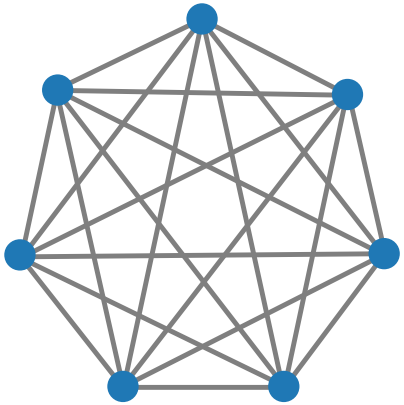
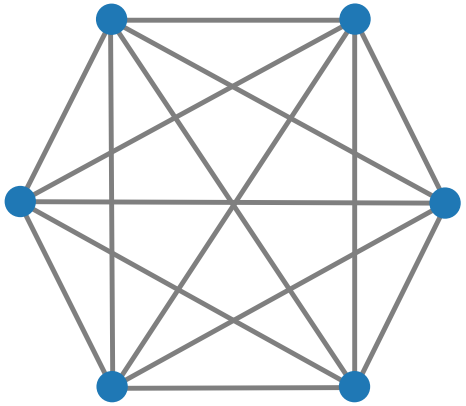
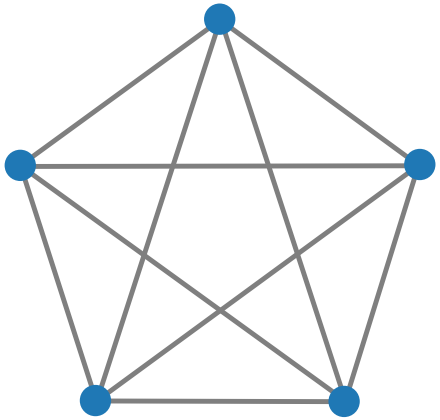
	1	1		
1		1		
1	1		1	1
		1		
		1		

- The number of walks of length r from v_i to v_j is represented by

$$N_{ij}^{(r)} = [\boldsymbol{A}^r]_{ij}$$

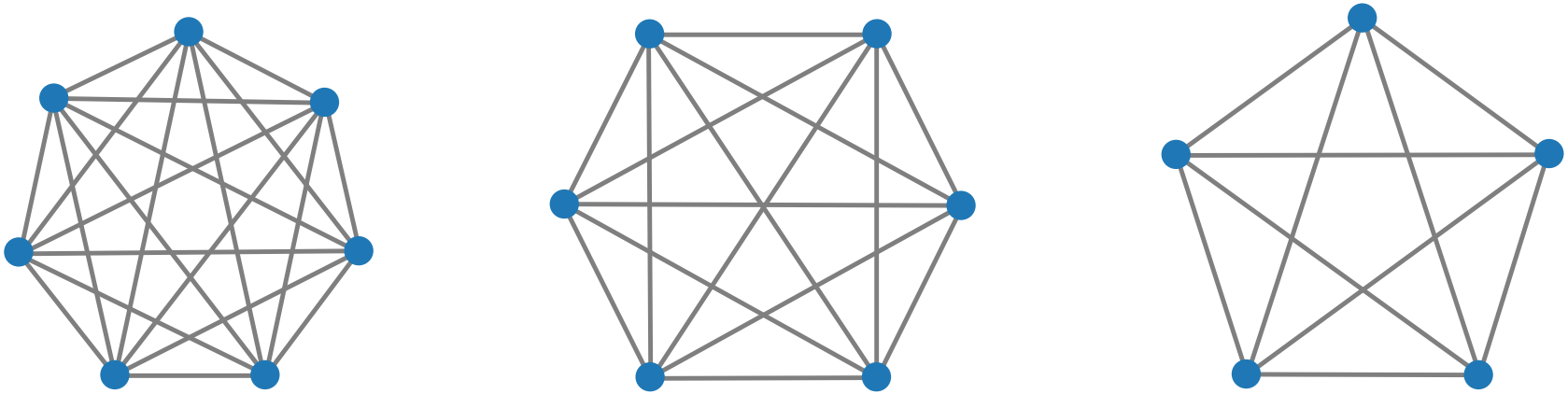
Complete Graph and Clique

- A clique or complete graph is a graph in which every pair of nodes are connected with an edge.

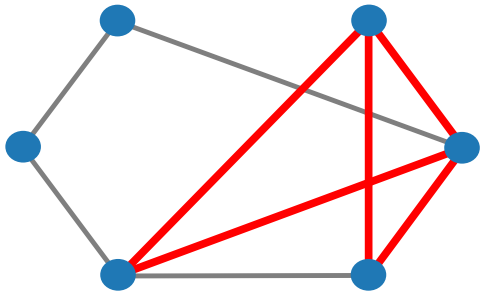




Complete Graph and Clique

- A clique or complete graph is a graph in which every pair of nodes are connected with an edge.



- A subset of nodes in the undirected graph that are all connected with an edge is called a clique.



Summary

- Data representation on graphs
  - Molecules, proteins, social network, WWW, fMRI, simulation.
- Undirected and directed graphs.
- Adjacency matrix.
- Node degree.
- Node attributes.
- Weighted graphs, Multi-relational graphs, and Line graph.
- Self-loop.
- Neighbors, k-hop neighborhood, and Ego graph.
- Path, distance, and walk.
- Complete graph and Clique.