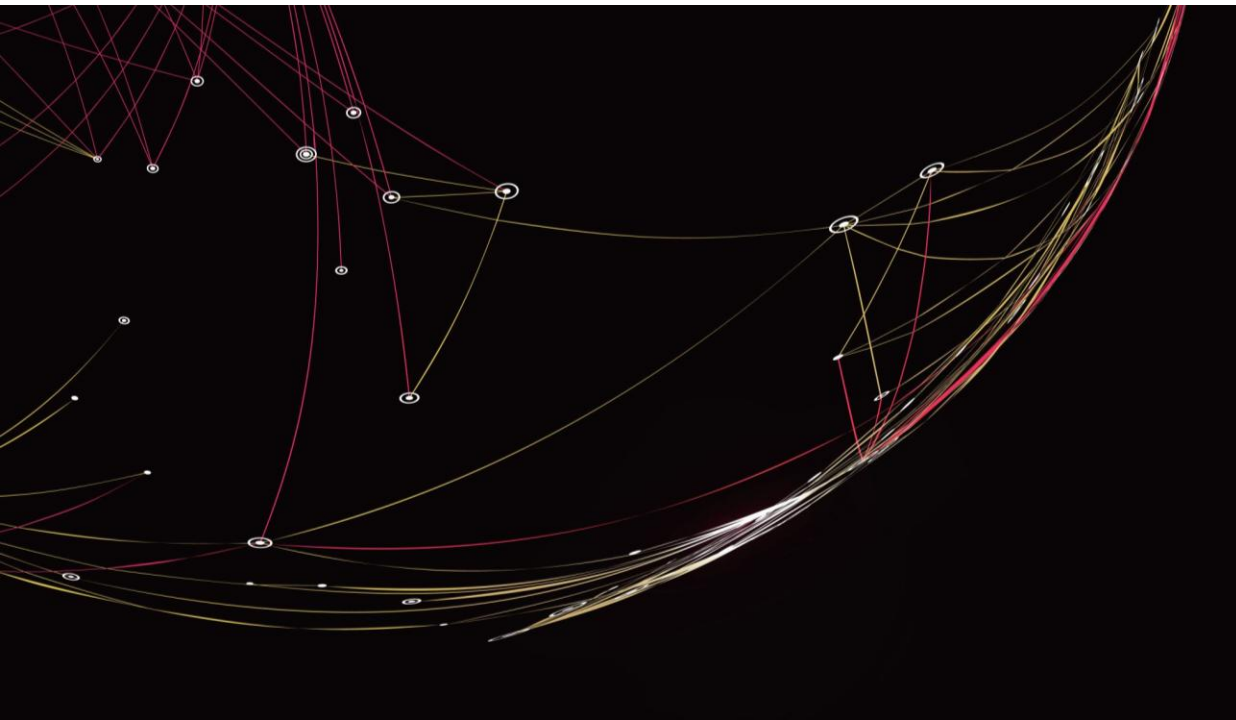



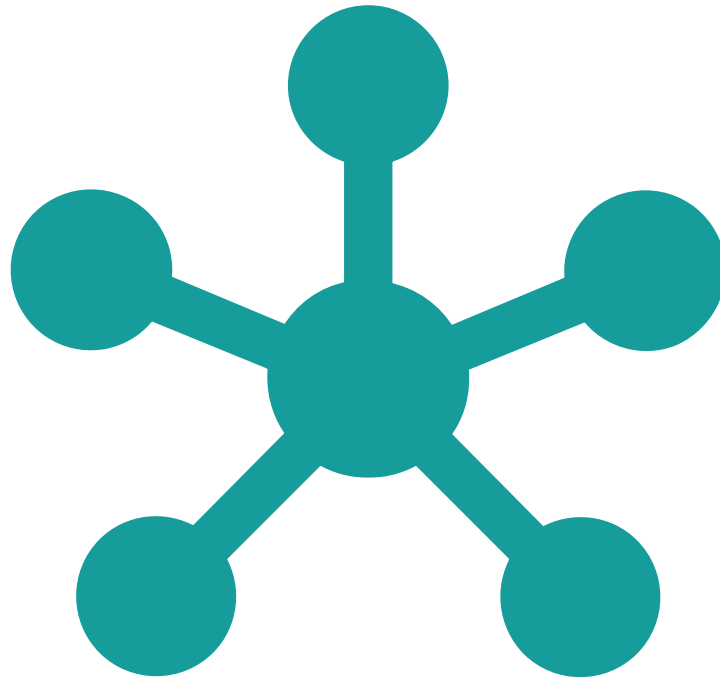
Understanding Graph Neural Networks (GNNs)

- Onoja Anthony, Ph.D.



 DataEdge Academy
 **YouTube** @tonyonoja7880

Introduction to Graph Neural Networks (GNNs)



Processing and Learning from Graph-Structured Data

Learning Objectives

By the end of this session, you'll be able to:

- ✓ Define graphs and graph structures
- ✓ Describe how GNNs differ from CNNs/RNNs
- ✓ Understand message-passing and node embeddings
- ✓ Learn major GNN architectures (GCN, GAT, GRN)
- ✓ Discover real-world applications



What is a Graph?

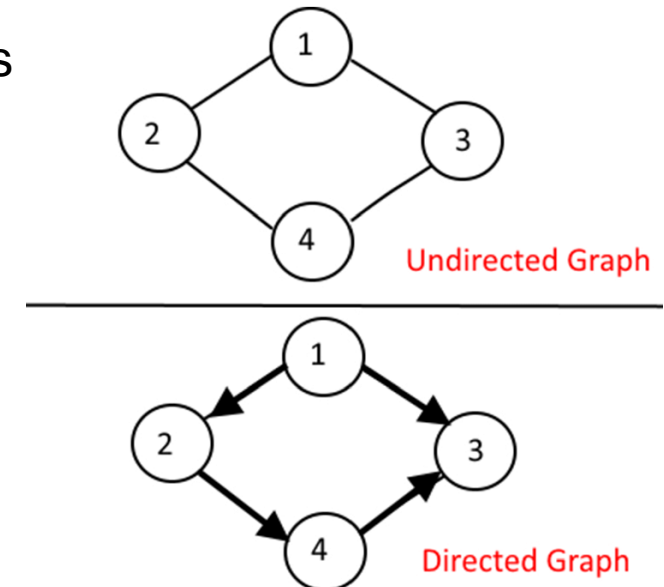
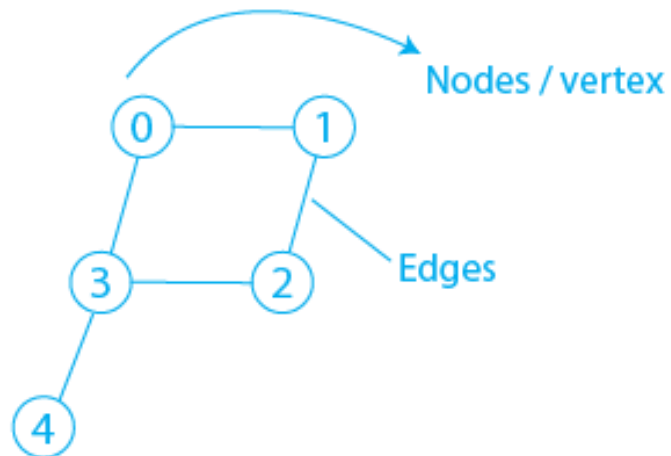
A graph is a non-Euclidean data structure that represents relationships between entities.

Components:

- Nodes (Vertices): Entities or objects in the graph.
- Edges (Links): Connections or relationships between nodes.

Types: Directed vs. Undirected, Homogeneous vs. Heterogeneous, Static vs. Dynamic

Representations: adjacency matrix, list, edge list, feature matrices

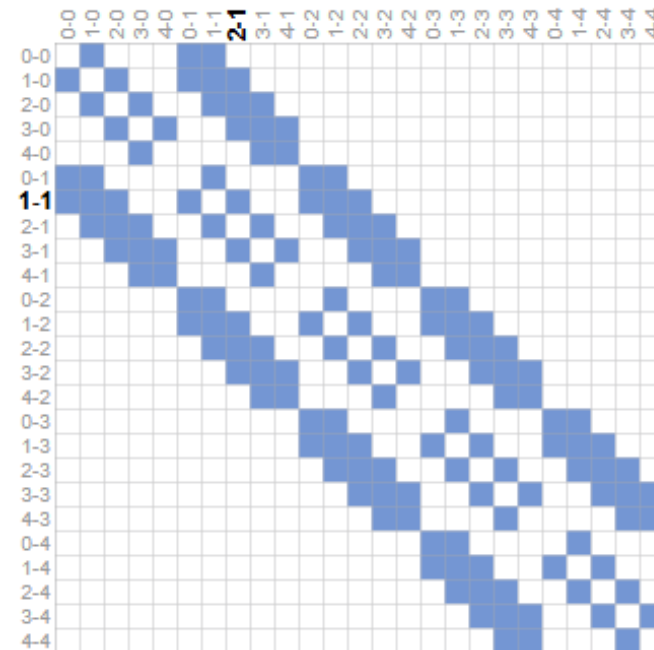


Images as Graphs

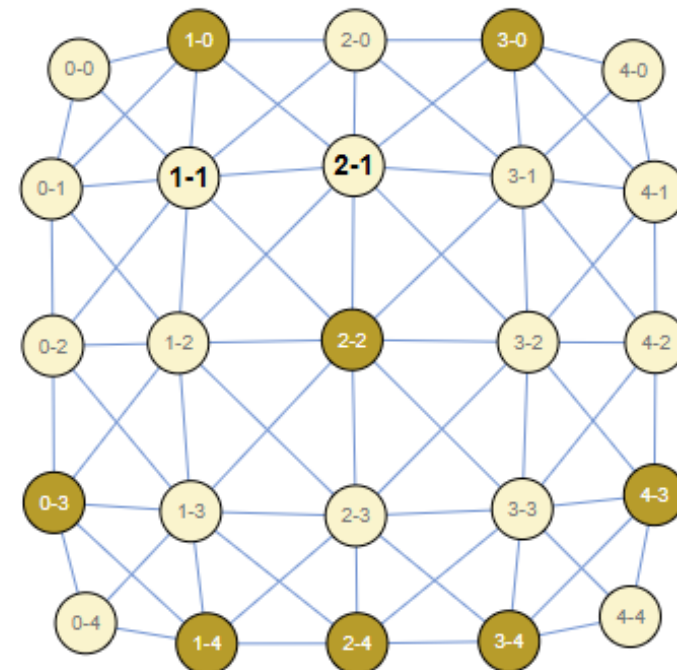
- ❑ Images can be viewed as graphs, not just pixel grids
- ❑ Each pixel = node, connected to up to 8 neighbouring pixels
- ❑ Node features = RGB values (3D vectors)
- ❑ The adjacency matrix shows pixel connectivity
- ❑ Same image data can be represented as:
 - ❑ Grid (array)
 - ❑ Graph structure
 - ❑ Adjacency matrix

0-0	1-0	2-0	3-0	4-0
0-1	1-1	2-1	3-1	4-1
0-2	1-2	2-2	3-2	4-2
0-3	1-3	2-3	3-3	4-3
0-4	1-4	2-4	3-4	4-4

Image Pixels



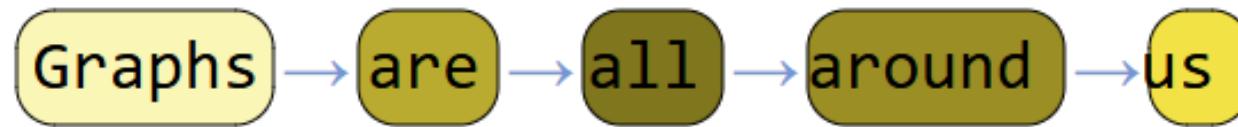
Adjacency Matrix



Graph

Text as Graphs

- ❑ Text can be digitised by assigning indices to characters, words, or tokens
- ❑ Each token = node, forming a directed graph
- ❑ Edges connect each node to the next in sequence
- ❑ Captures structure and order of language in graph form
- ❑ Useful for applying Graph Neural Networks to NLP tasks

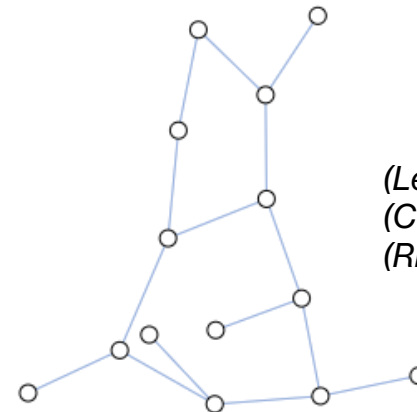
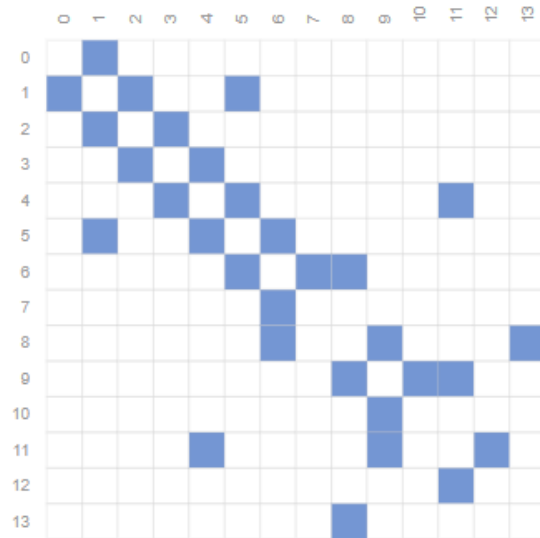
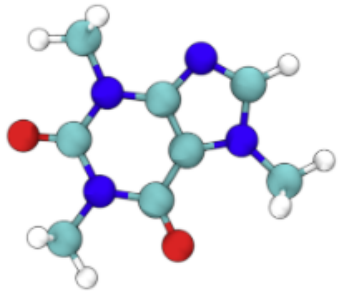


	Graphs	are	all	around	us
Graphs		1			
are			1		
all				1	
around					1
us					

Graph-Valued Data in the Wild

Some data is naturally irregular and heterogeneous. Nodes have variable numbers of neighbours. These cannot be easily represented as grids or sequences → graph is ideal.

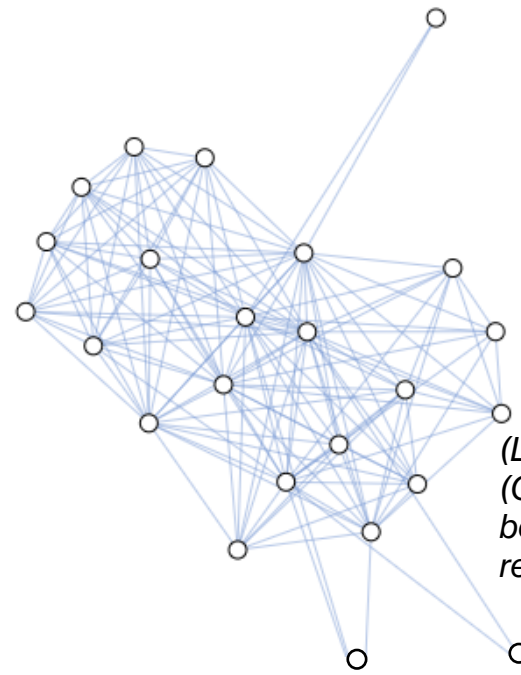
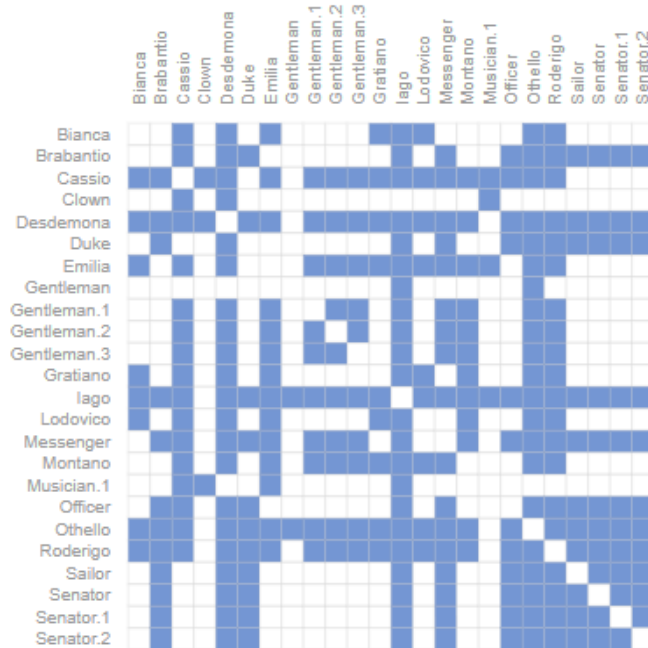
- ❑ Example: Molecules as Graphs
- ❑ Molecules = atoms (nodes) + bonds (edges)
- ❑ Covalent bonds form edges between atoms in stable configurations
- ❑ Edge types vary (e.g. single, double, triple bonds)
- ❑ Graphs capture molecular structure in a compact, flexible form



(Left) 3d representation of the Caffeine molecule
(Centre) Adjacency matrix of the bonds in the molecule
(Right) Graph representation of the molecule.

Social Networks as Graphs

- ✓ Used to study patterns in human and organisational behaviour
- ✓ Each individual = node
- ✓ Edges represent relationships (e.g. friendship, collaboration, communication)
- ✓ Can model complex, large-scale social structures and interactions
- ✓ Useful for tasks like influence analysis, community detection, and recommendation systems

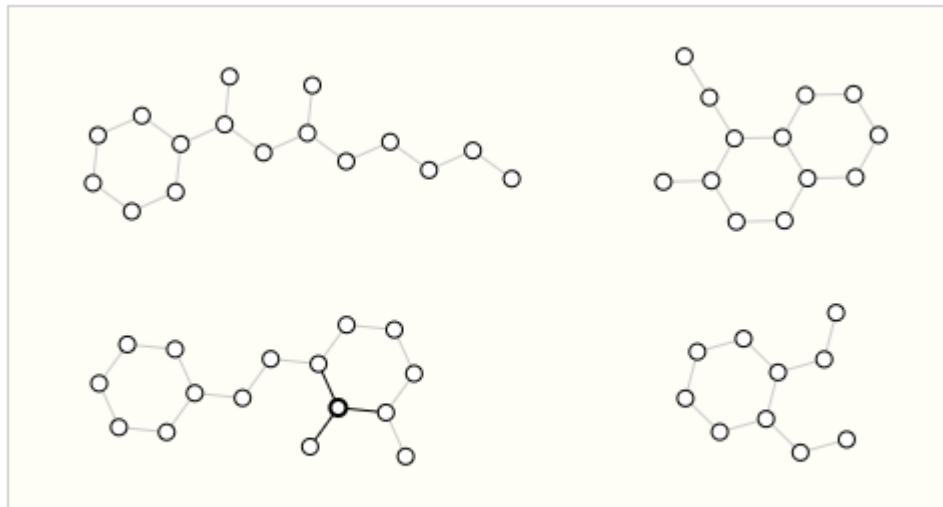


*(Left) Image of a scene from the play “Othello”.
(Centre) Adjacency matrix of the interaction
between characters in the play. (Right) Graph
representation of these interactions.*

Types of Problems in Graph-Structured Data

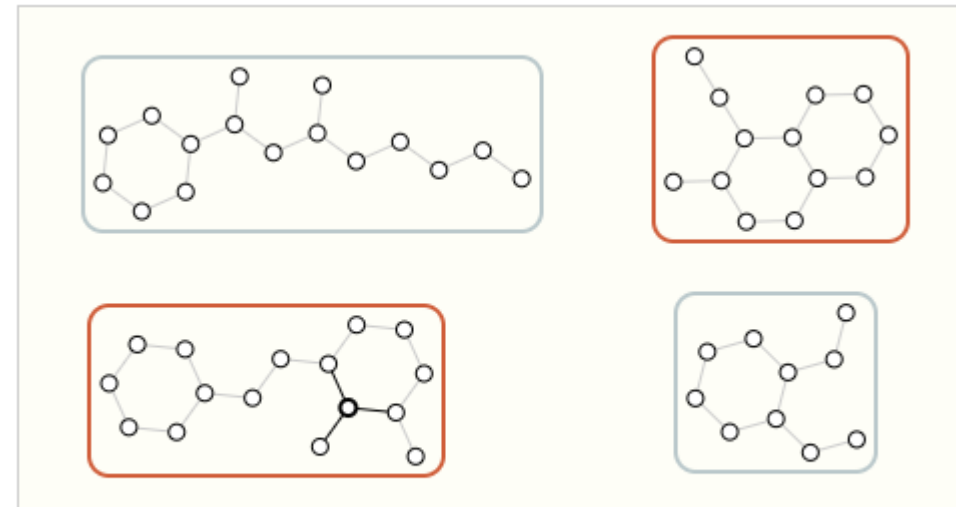
Graph-Level Prediction

- ❑ Predict a property for the entire graph
- ❑ Example: Classify molecules as toxic or non-toxic
- ❑ Task: Graph classification or regression.



Input: graphs

Credit: <https://distill.pub/2021/gnn-intro/>

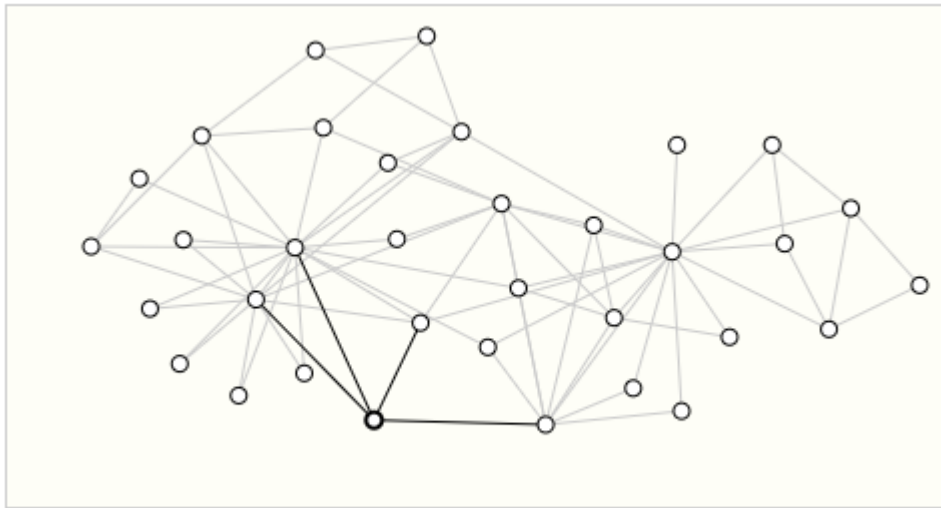


Output: labels for each graph, (e.g., "does the graph contain two rings?")

Types of Problems in Graph-Structured Data

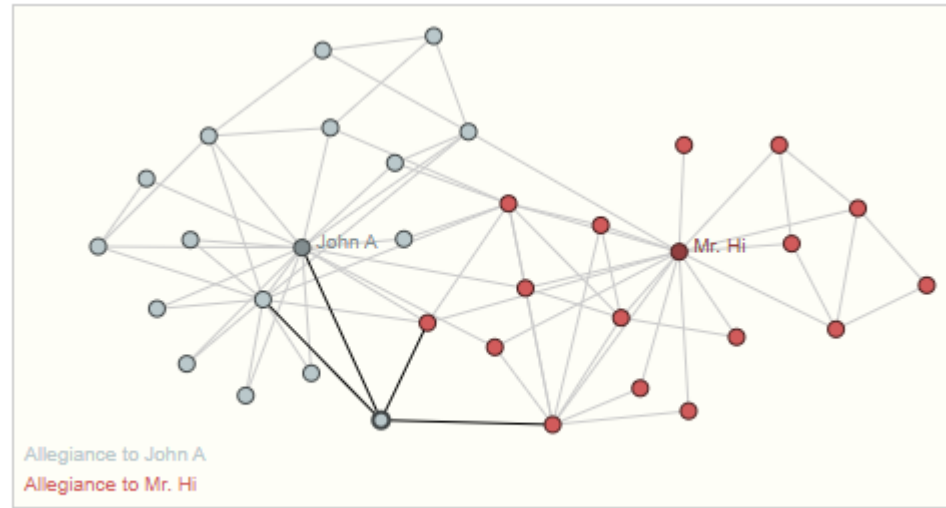
Node-Level Prediction

- ❑ Predict a label or value for each node
- ❑ Example: Classify users in a social network (e.g., bots vs. humans)
- ❑ Task: Node classification or node regression.



Input: graph with unlabeled nodes

→



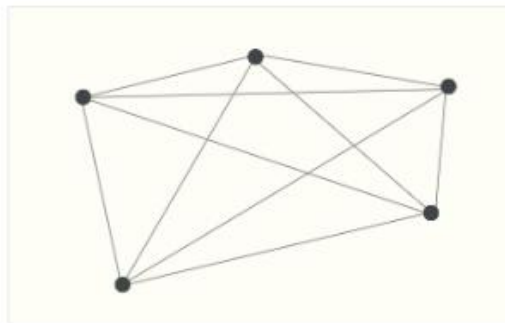
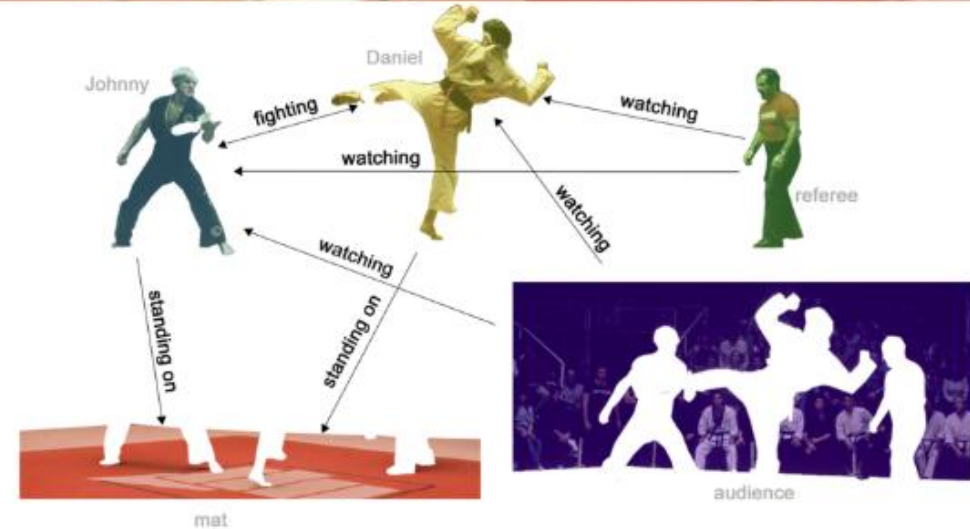
Output: graph node labels

Credit: <https://distill.pub/2021/gnn-intro/>

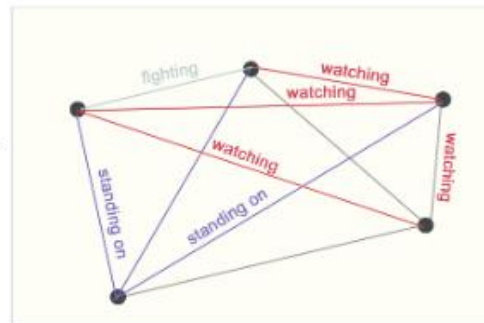
Types of Problems in Graph-Structured Data

Edge-Level Prediction

- ❑ Predict relationships between nodes.
- ❑ Example: Predict if a link exists between two users (link prediction).
- ❑ Task: Edge classification or edge regression.



Input: fully connected graph, unlabeled edges



Output: labels for edges

The Challenge of Graph Data

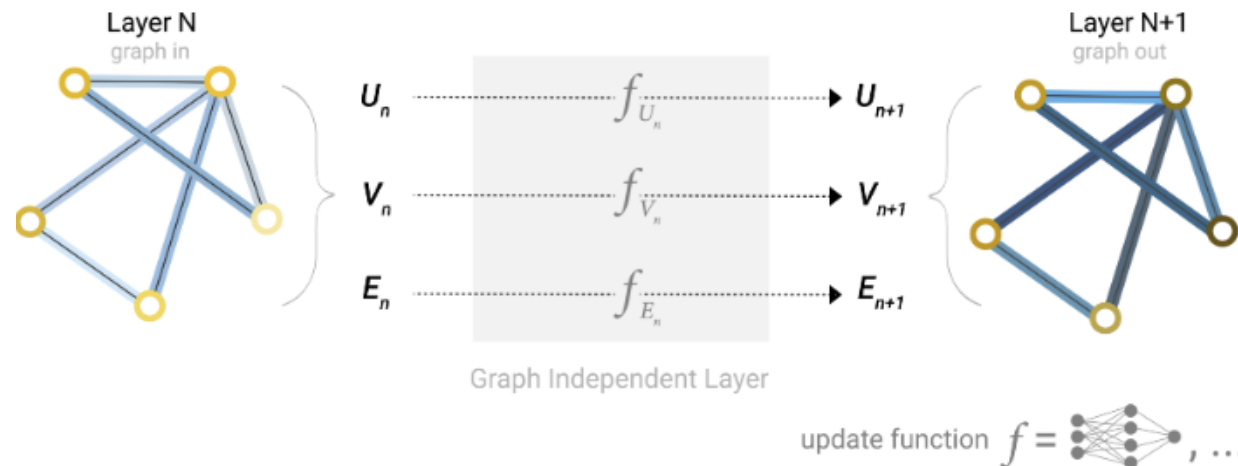
Why traditional NNs struggle:

- ✓ Standard neural networks (CNNs, RNNs) are designed for structured data (grids, sequences).
- ✓ Graphs are dynamic, irregularly structured, and can have varying numbers of neighbours (nodes).
- ✓ Traditional methods lose the crucial relational information.

Introduction to Graph Neural Networks (GNNs)

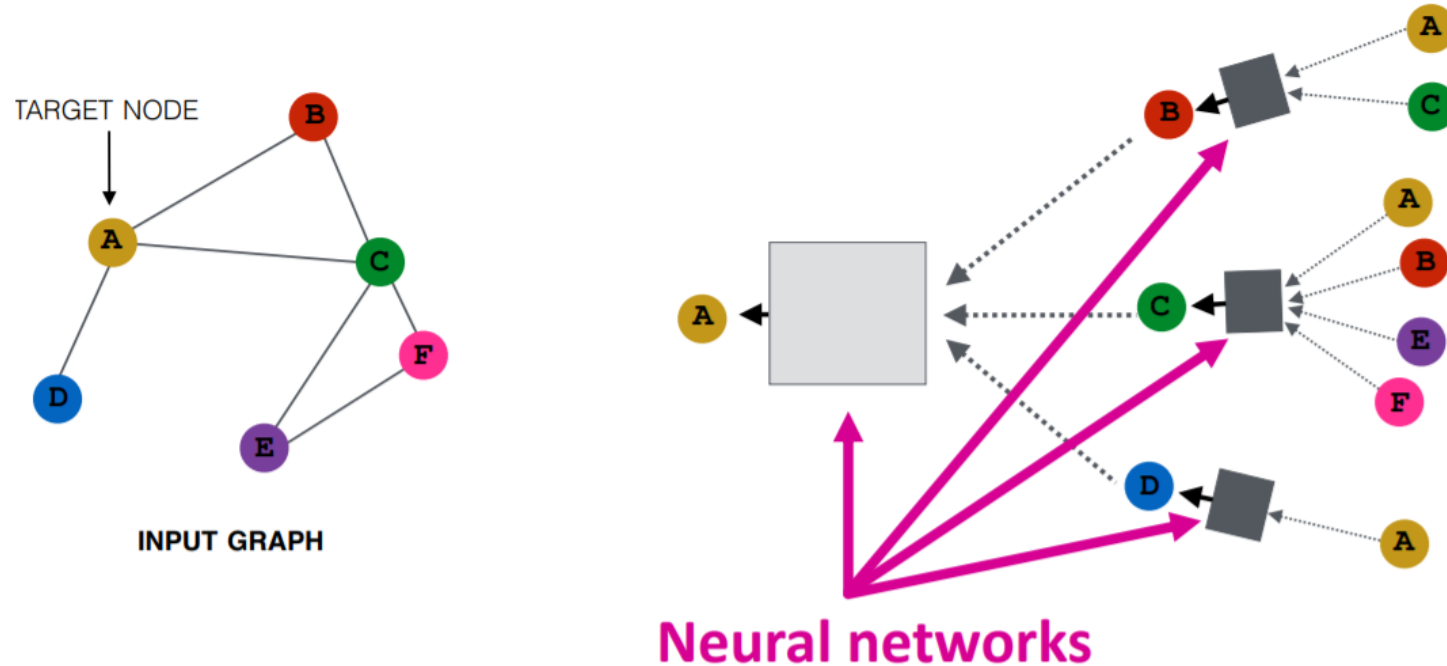
A specialised class of neural networks designed to process and learn from graph data.

- ❑ Key Function: GNNs learn representations (embeddings) for nodes, edges, or the entire graph by considering both node features and the graph structure.
- ❑ Geometric Deep Learning: GNNs are a form of geometric deep learning.
- ❑ GNNs enable learning on social networks, molecules, knowledge graphs.



Key Concepts in GNNs

- ❑ Message Passing: nodes aggregate information from neighbours



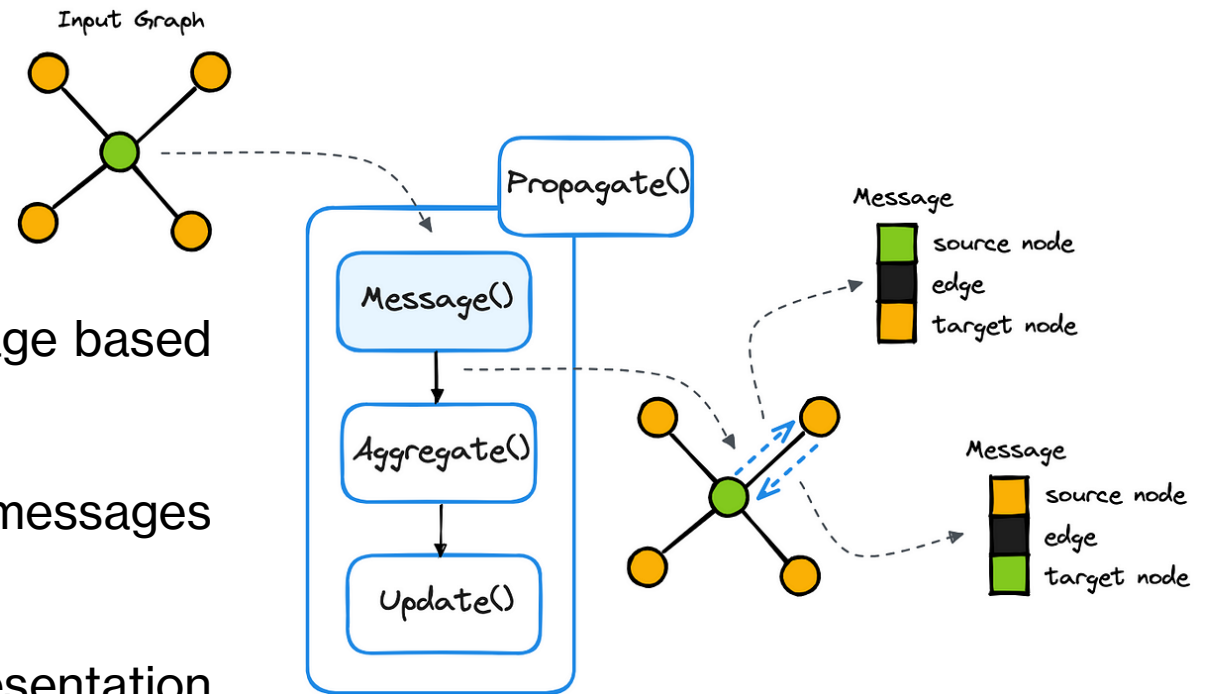
- ❑ Node Embeddings: learned feature vectors per node.
- ❑ Graph-level Outputs: e.g. whole-graph classification or regression.

How GNNs Work: The Message Passing Paradigm

Core Concept: GNNs operate via "message passing."

Process:

- ❑ Message Generation: Each node creates a message based on its own features and those of its neighbours.
- ❑ Aggregation: A node combines (aggregates) the messages received from its neighbours.
- ❑ Update: The node updates its own state or representation using the aggregated information and its previous state.



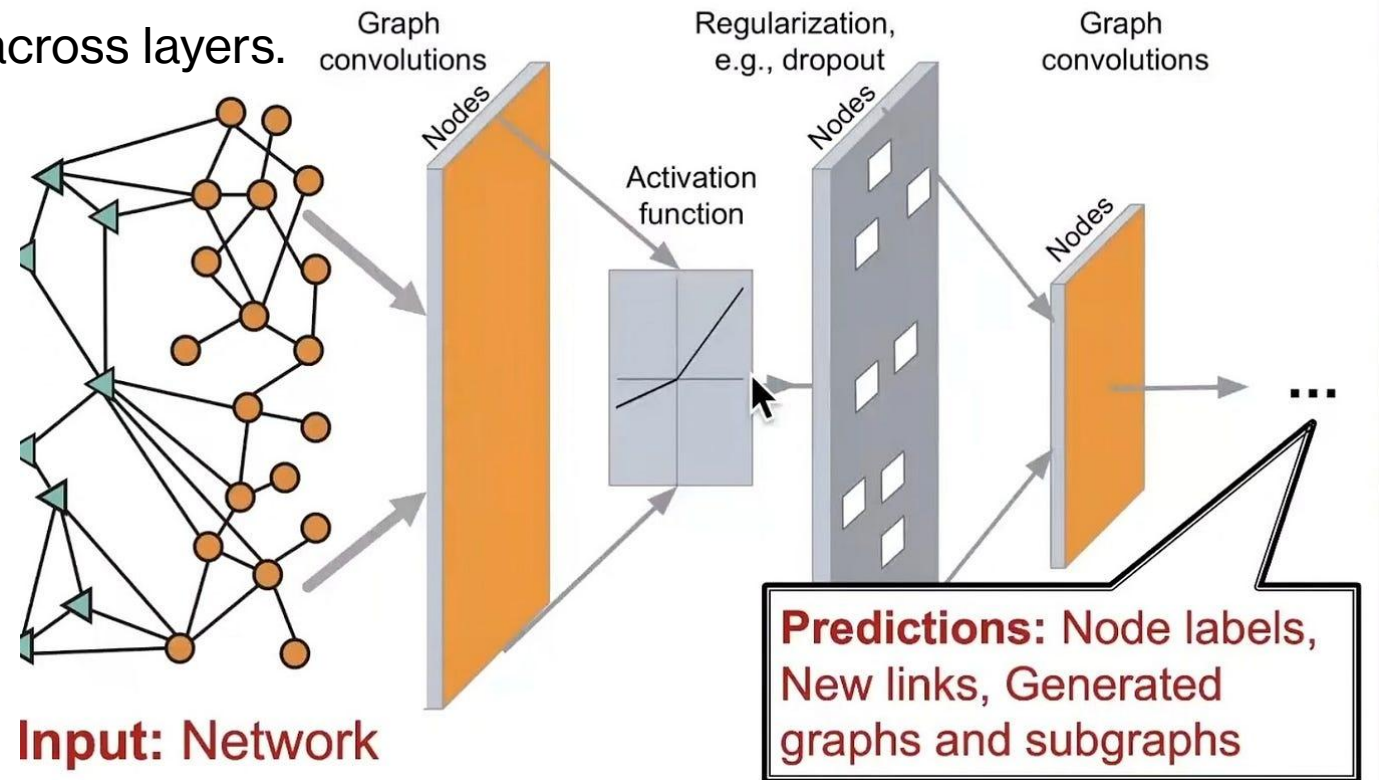
Goal: To iteratively capture local and global graph structure.

Types of GNN Architectures

- ❑ Graph Convolutional Networks (GCNs): Extend CNN concepts to graphs, aggregating neighbour information using convolution operations.
- ❑ Graph Attention Networks (GATs): Introduce attention mechanisms to assign different weights to messages from neighbours, focusing on more relevant information.
- ❑ Graph Autoencoders (GAEs): Used for tasks like graph reconstruction and dimensionality reduction.
- ❑ Recurrent Graph Neural Networks (RecGNNs): Utilise recurrent principles for learning sequential or dynamic graph data.

Graph Convolutional Networks (GCNs)

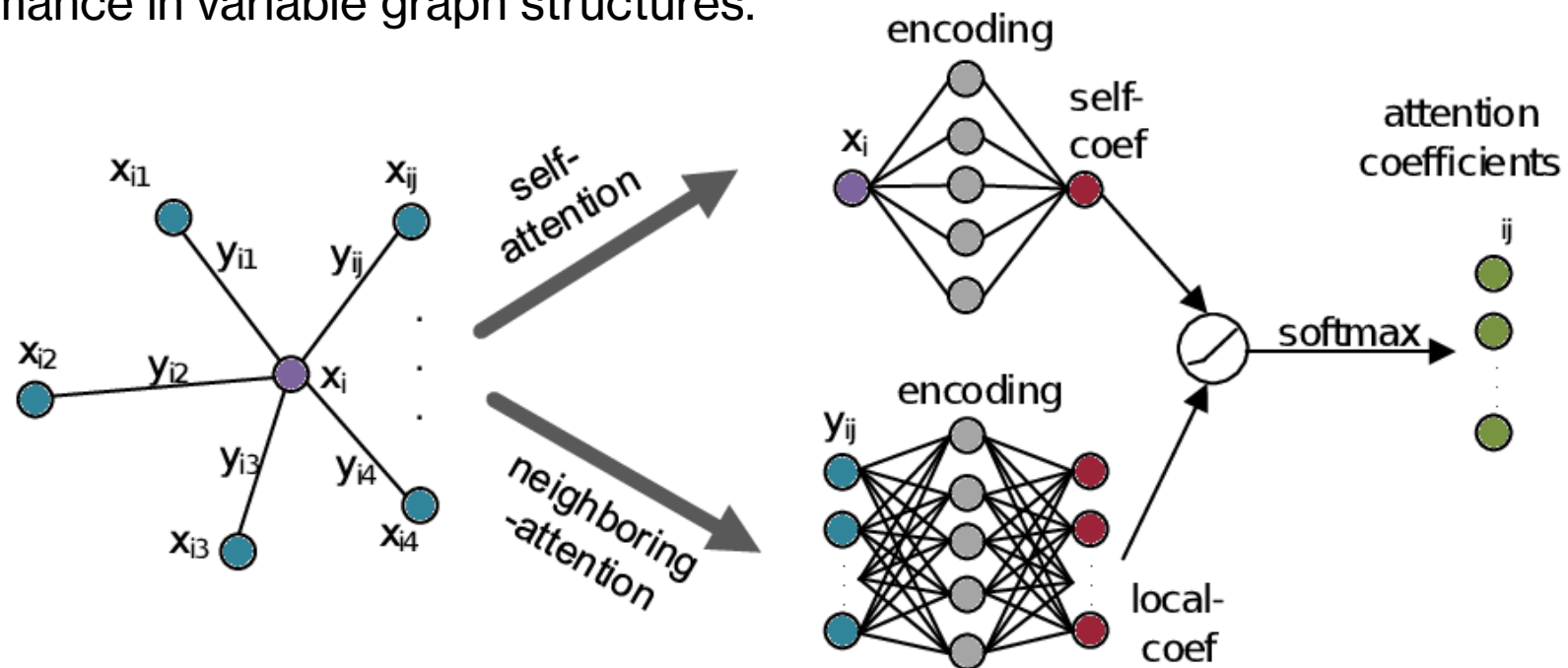
- ✓ Extend CNNs to graph topology.
- ✓ Aggregate neighbour features, refine across layers.
- ✓ Spectral vs. spatial GCNs.



Ideal for semi-supervised node classification tasks

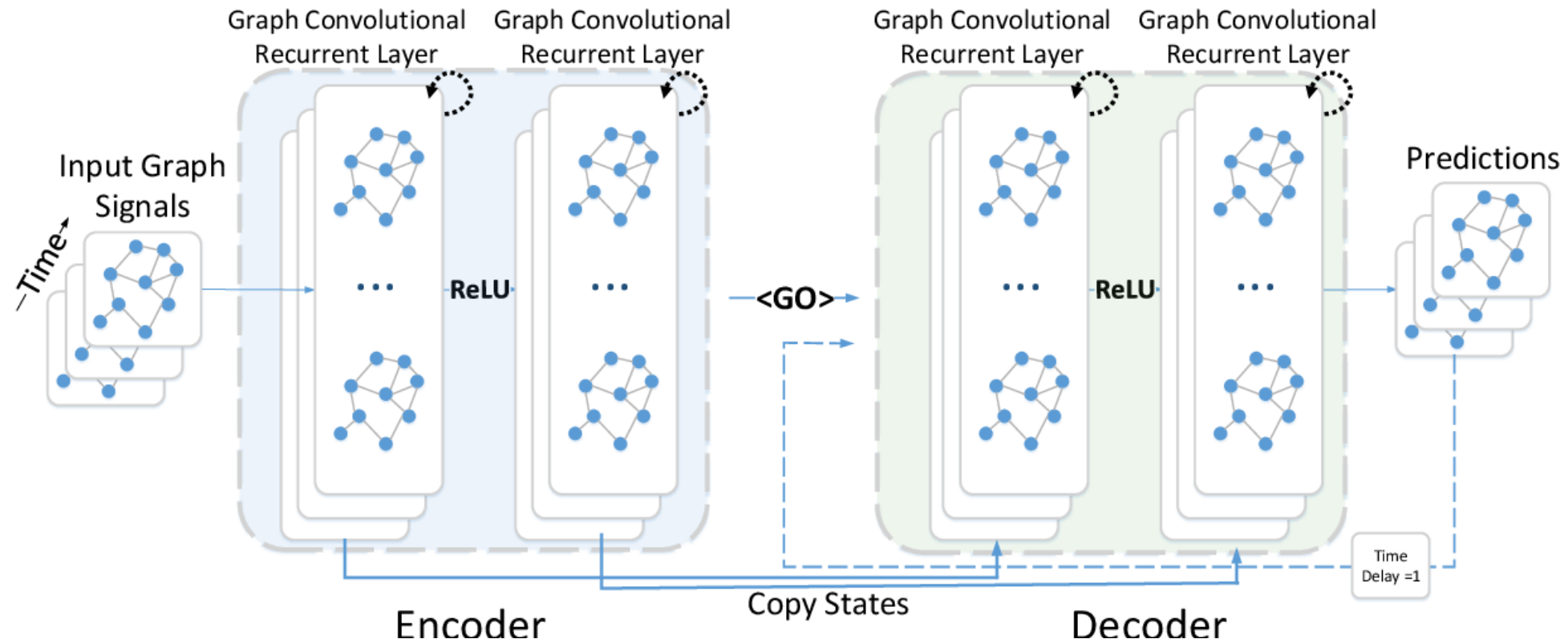
Graph Attention Networks (GATs)

- ❑ Introduce attention weights on edges.
- ❑ Learn which neighbours contribute most.
- ❑ Improve performance in variable graph structures.



Graph Recurrent Networks (GRN)

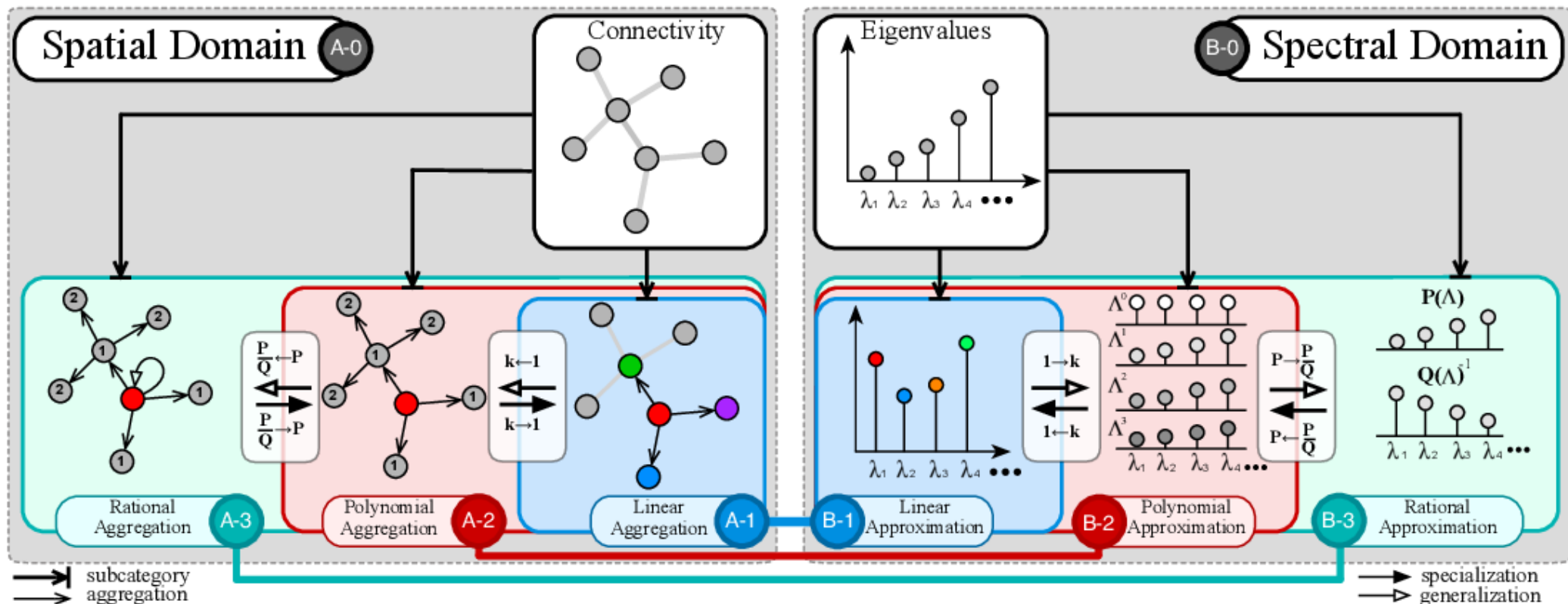
GRNs combine the principles of recurrent neural networks (RNNs) with graph structures.



Designed to handle temporal dynamics in graph data, making them suitable for scenarios where relationships evolve over time.

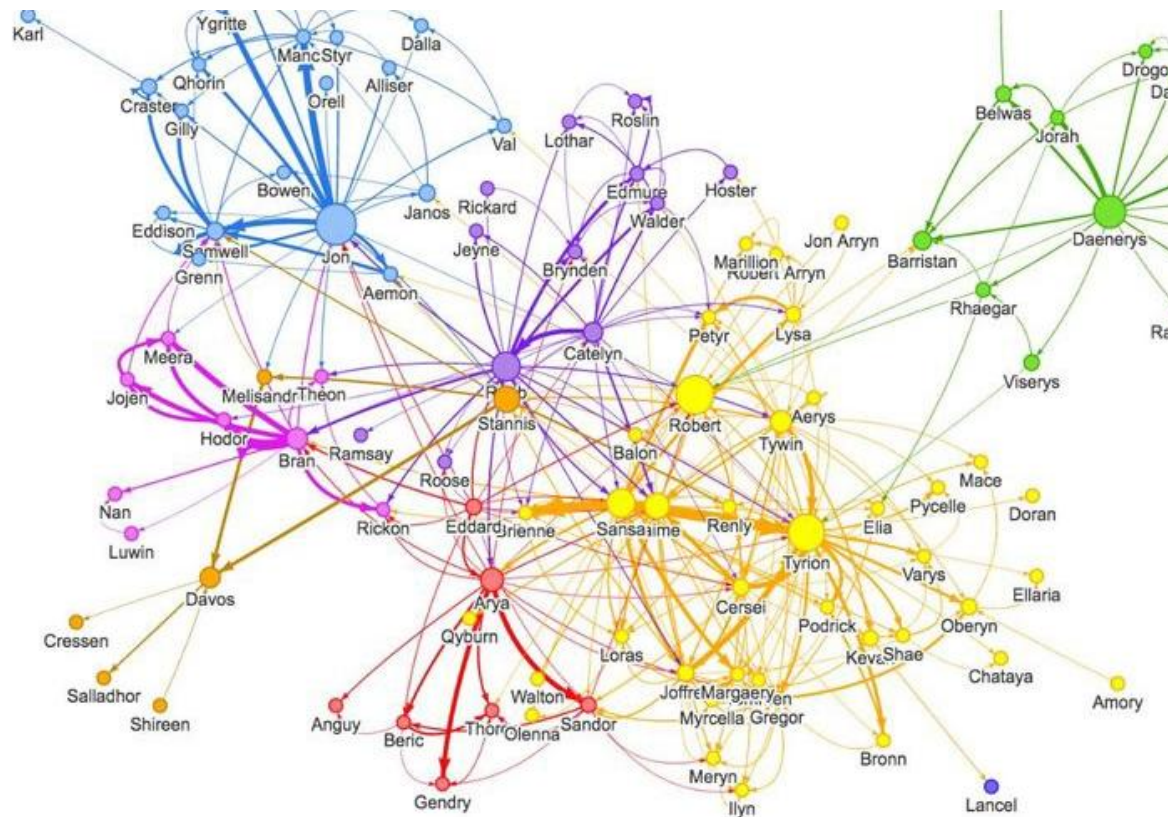
Spatial and Spectral-based GNNs

- ❑ Spatial-based GNNs operate directly on the graph structure, focusing on the spatial relationships between nodes.
- ❑ Spectral-based GNNs utilise spectral graph theory, applying techniques from Fourier analysis to define convolutions on graphs.



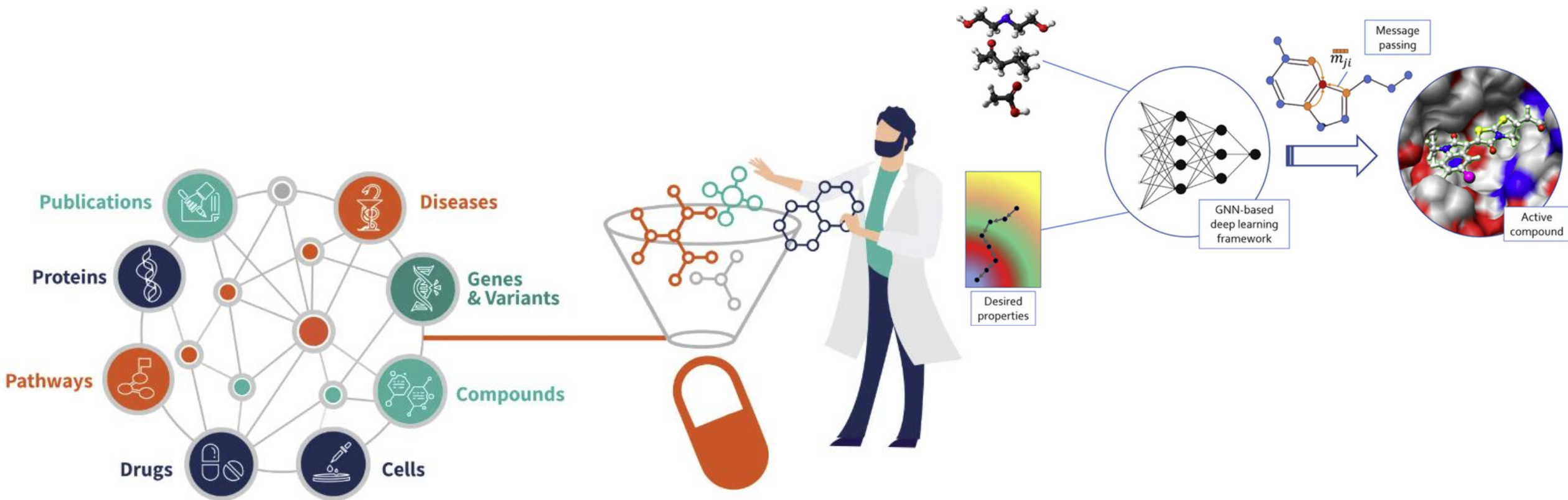
GNN Applications: Social Networks

- User behavior prediction.
- Friend recommendations.
- Community detection.



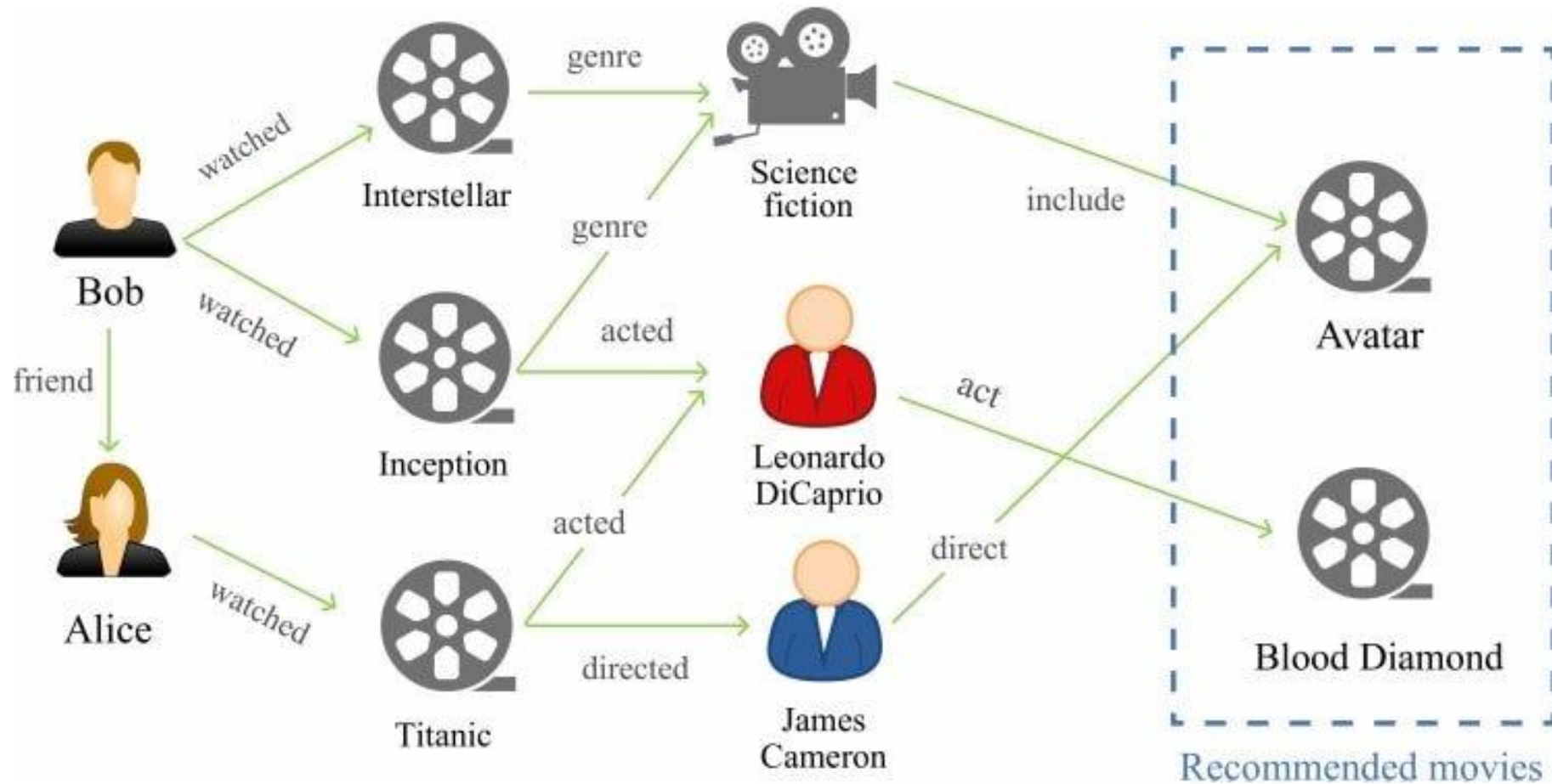
GNN Applications: Drug Discovery and Bioinformatics

- ❑ Predicting molecular properties.
- ❑ Analysing protein structures.

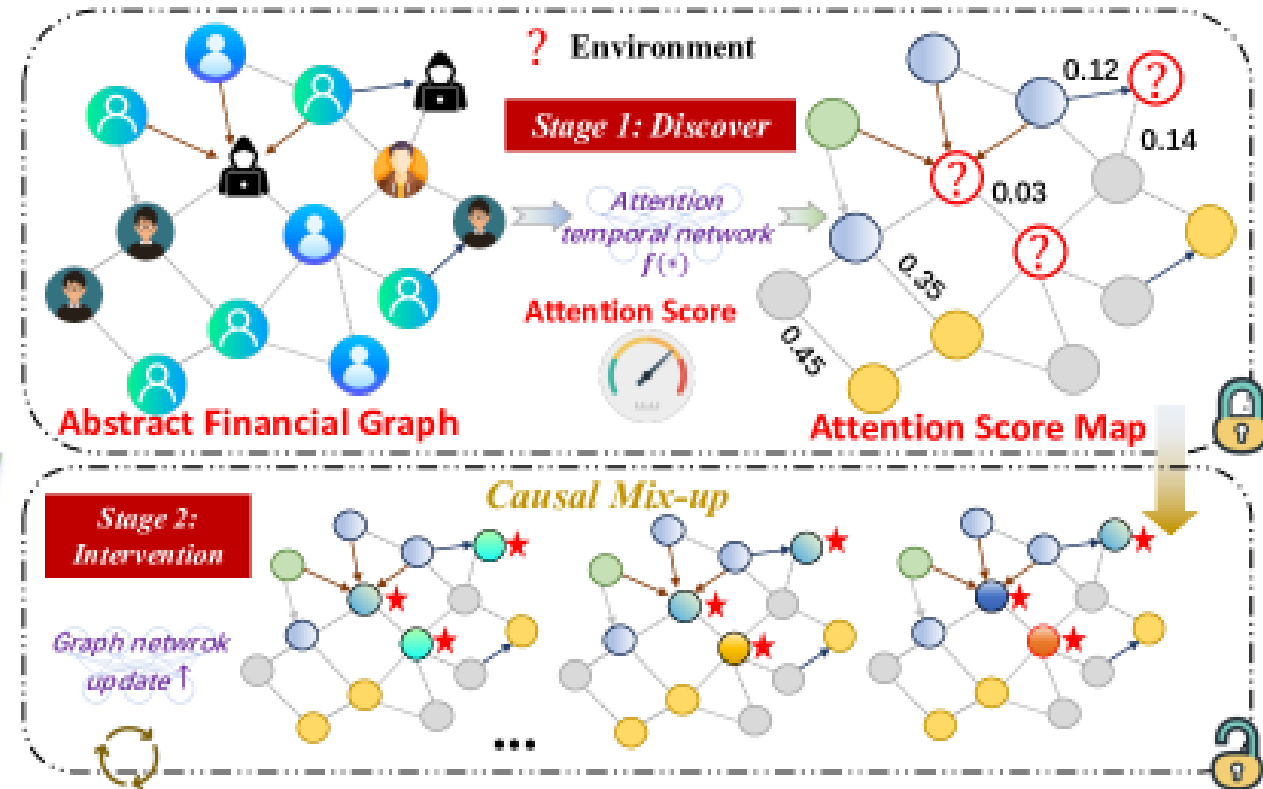
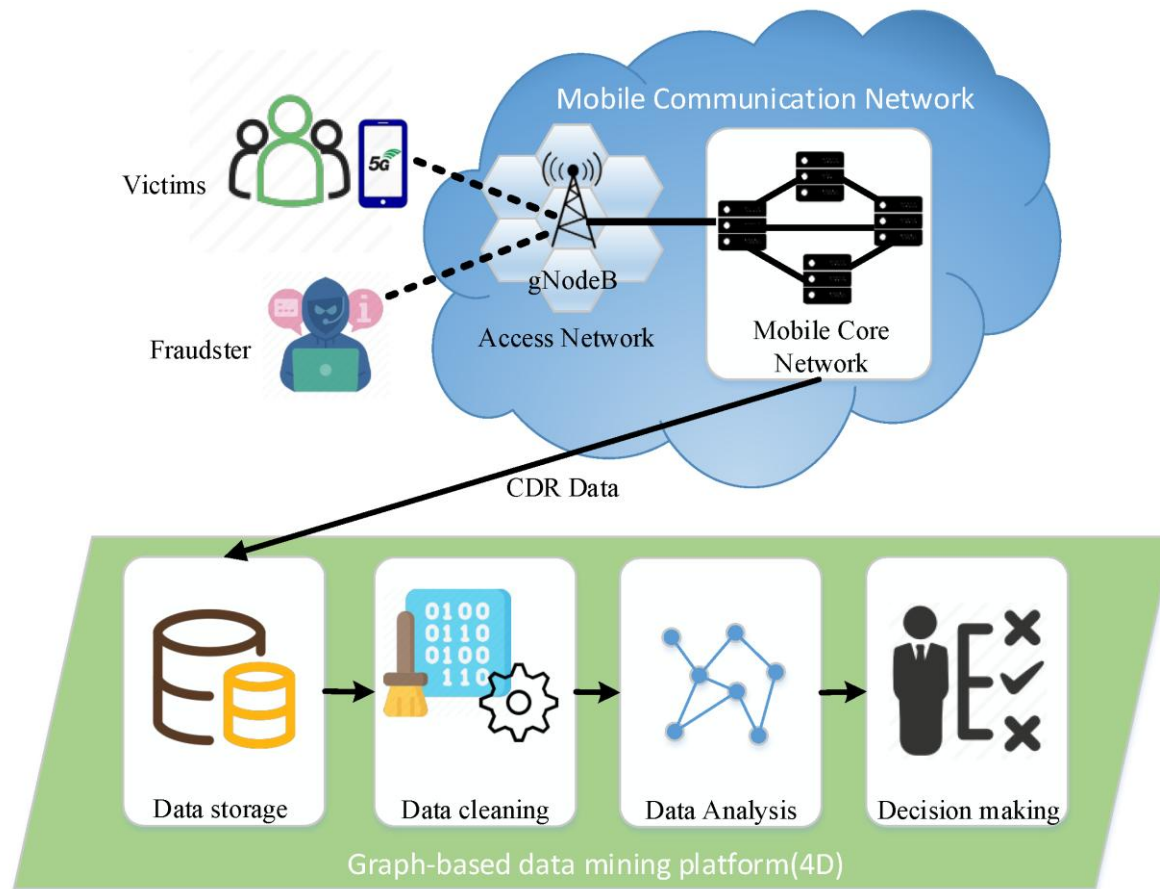


GNN Applications: Recommender Systems

Personalised recommendations based on user-item interaction graphs.



GNN Applications: Fraud Detection

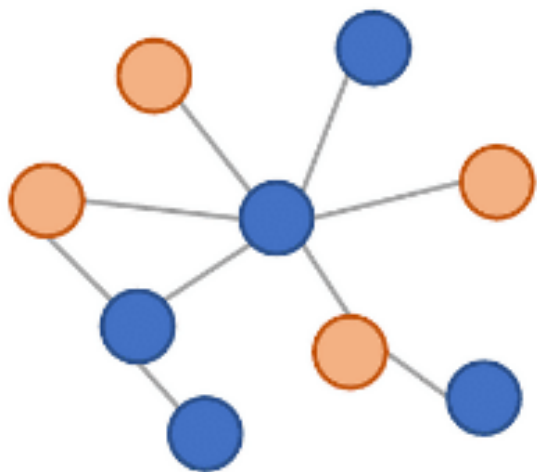


Identifying suspicious transactions or patterns in financial graphs.

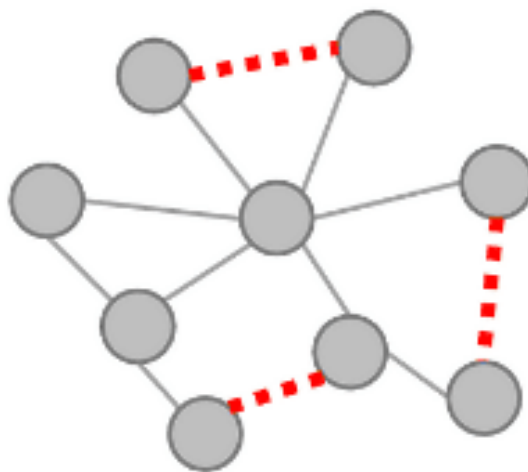
Advantages of using GNNs

- ❑ Excellent at modelling complex relationships and dependencies.
- ❑ Can handle non-Euclidean data.
- ❑ Effective for tasks involving relational data.

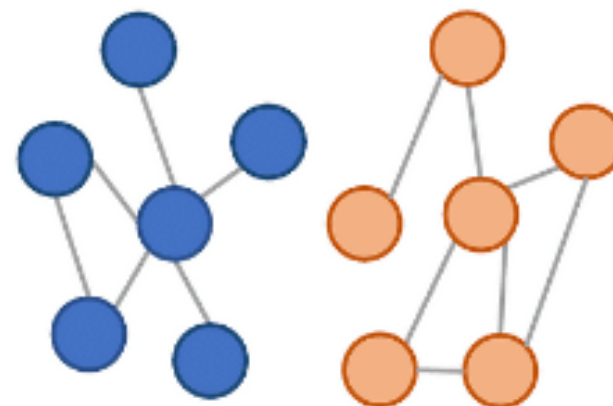
Node Classification



Link Prediction



Graph Classification



Limitations & Challenges

- Scalability: Processing very large graphs can be computationally expensive.
- Over-smoothing: Deep GNNs can sometimes cause node representations to become too similar.
- Dynamic Graphs: Handling rapidly changing graphs remains a challenge.

Conference'17, July 2017, Washington, DC, USA

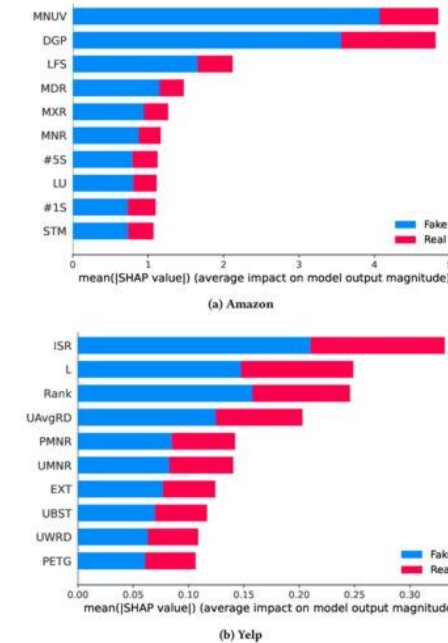


Figure 5: Feature Attributions in the Fake Review Detection Task. We only display top-10 important features attributing to model predictions for two corresponding datasets.

As presented in Fig. 6, nodes in the Cora graph are displayed in clusters corresponding to its embedding vectors, projected into the 2-D dimension using T-SNE. From this global view, we can detect cluster boundaries and specify influential factors on a high level. By selecting a node, we can see how it impacts on or is affected by its neighbors. Furthermore, the global view provides an overall understanding of how neighbors impact nodes located at the boundary lines in different clusters. We also observed that important nodes usually have a high degree connected to numerous nodes within their clusters and are located at extreme corners. However, an influential node in the global view only means that it connects to numerous neighbors and sometimes does not have a decisive impact on them. For instance, a top-cited paper has many links to papers across domains, but downstream nodes' classes depend on their locally connected neighbors.

Similarly, Fig. 7 depicts multiple levels of local explanations for an example node. In local explanation, our software provides additional features such as graph layout selection, k^{th} -hop highlighting,

Tien-Cuong Bui, Wen-syan Li, and Sang-Kyun Cha

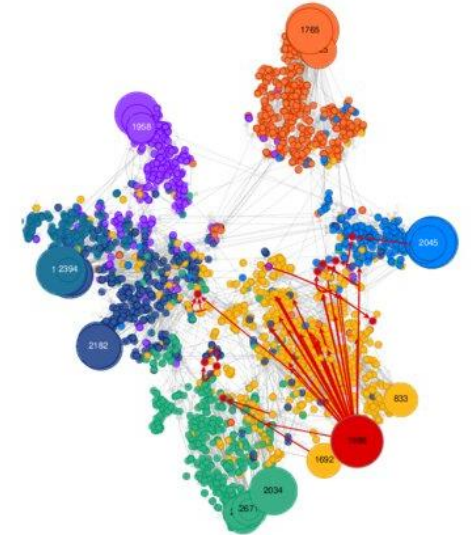


Figure 6: Global Summary Visualization of Cora Graph based on APPNP teacher and SGAT Student Model. We filter out edges with a probability smaller than 0.3 and highlight top-50 nodes (large nodes) ordered by ranks. Red nodes are selected ones, while red edges show the influential flows from/to them. Nodes in different colors are in different clusters.

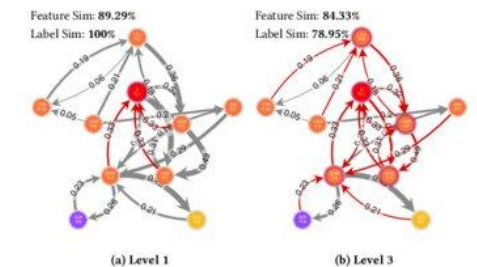


Figure 7: Local Explanations of Node ID 5 (Red Node). Each level shows the contributions of corresponding k^{th} -hop neighbors. Edge weights are normalized probabilities.

edge filtering, feature similarity, and label similarity [17] to support various requirements. First, we can adjust the number of visible

[Read full text here](#)

GNNs in the Data Science Pipeline

- ❑ Feature Engineering: GNNs can automate feature extraction from graphs.
- ❑ Model Training: Utilising GNN architectures for tasks like node classification, link prediction, and graph classification.
- ❑ Evaluation: Assessing performance using appropriate metrics for graph tasks.



Summary and Future Directions

GNNs are powerful tools for analysing structured relational data, unlocking insights that traditional deep learning methods cannot.

Future Directions:

- ✓ Improving scalability and efficiency.
- ✓ Developing methods for dynamic graph learning.
- ✓ Integrating GNNs with other AI techniques (e.g., LLMs).

Questions and Answers