School of COMPUTING
Clemson University

Introduction to Graph Neural Networks

Aaron J. Masino, PhD
Associate Professor, School of Computing

School of COMPUTING Clemson University

Outline

- Graph terminology
- Graph neural networks

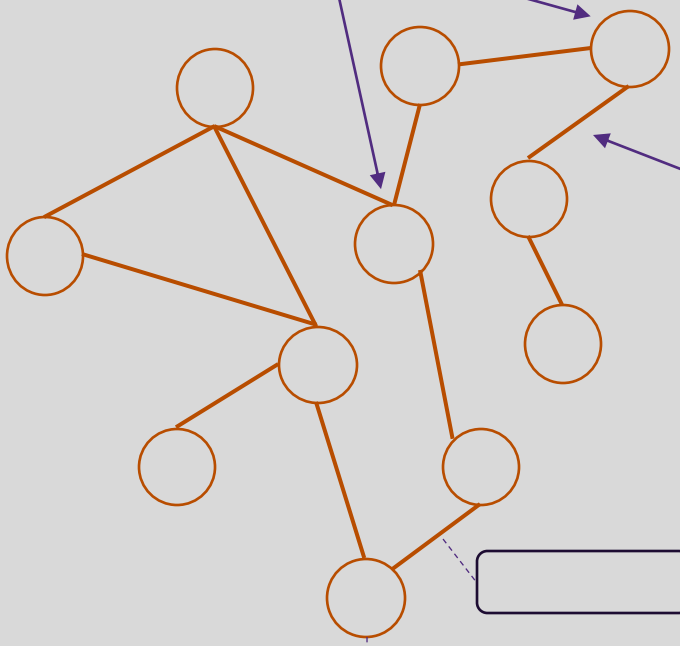
School of COMPUTING
Clemson University

Introduction to graphs

School of COMPUTING Clemson University

Terminology

Neighbors – two nodes connected by an edge
Neighborhood – all neighbors of a node



Vertices (nodes) usually represent entities (e.g., a patient, symptom, disease, cell type, atom/molecule, gene)

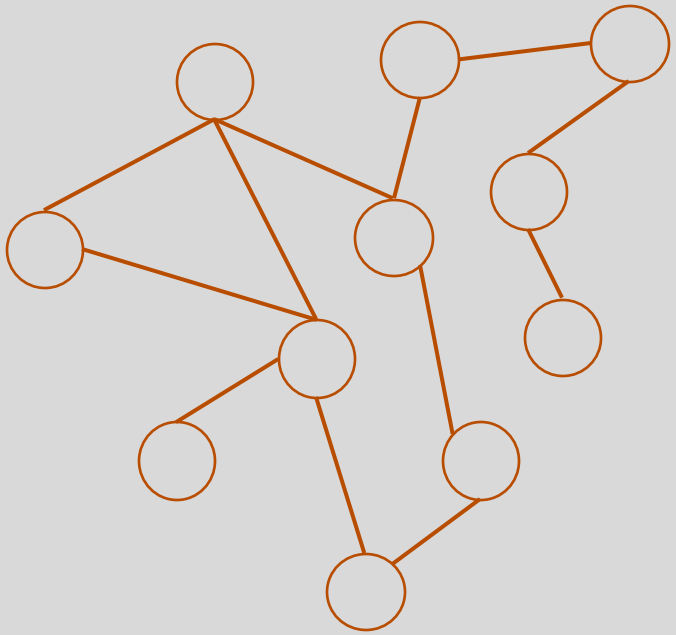
Edges usually represent relationships between nodes (e.g., “has symptom”, “causes disease”, “bonds with”, “regulates expression of”).

Graph representations may also include *attributes* (vertex / edge properties, e.g., text description).
Why not just represent these as additional vertices?

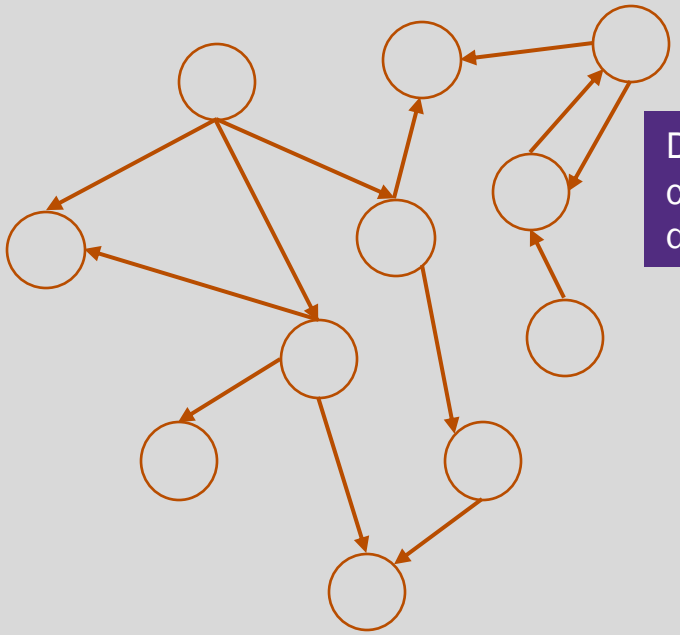
- Tends to lead to very dense networks for properties that are common to most nodes.
- Such properties usually don’t describe relations between nodes
- Computational efficiency

Terminology

Undirected Graph – edges have no direction and define relations that are inherently bidirectional.

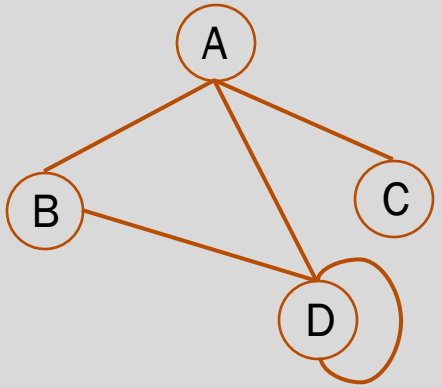


Directed Graph – edges have a specified direction and define relations that are inherently unidirectional.

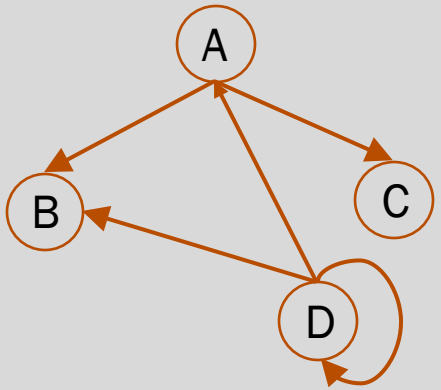


Directed graphs may contain edges in both directions between nodes.

Adjacency matrix



	A	B	C	D
A	0	1	1	1
B	1	0	0	1
C	1	0	0	0
D	1	1	0	1

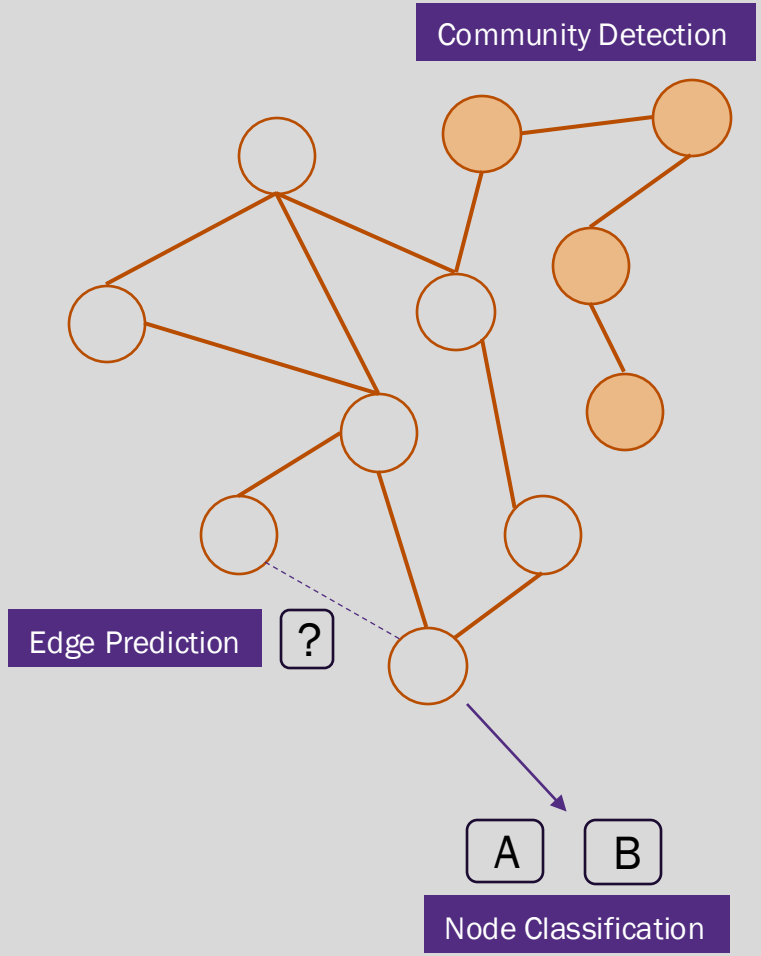


	A	B	C	D
A	0	1	1	0
B	0	0	0	0
C	0	0	0	0
D	1	1	0	1

- Adjacency matrix, A, summarizes the edges in a graph where the element $a_{i,j} \in A$ is the weight of the edge $e_{i,j} \in E$. For unconnected vertices, v_i, v_j, $a_{i,j} = 0$
- Adjacency matrix is symmetric for undirected graphs and usually asymmetric for directed graphs
- For graphs without self loops, $a_{i,i} = 0$, i.e., the diagonal elements are all 0

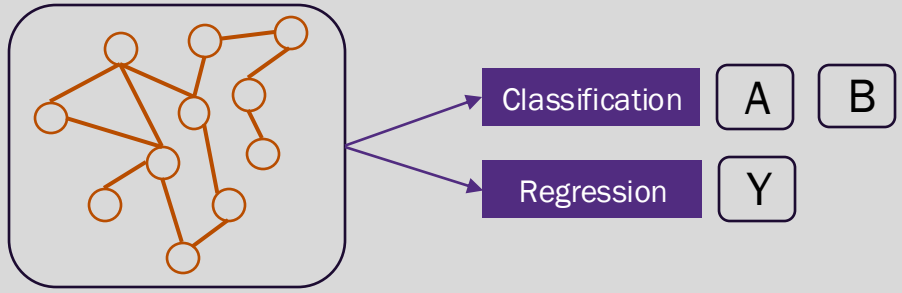
Intra-graph machine learning tasks

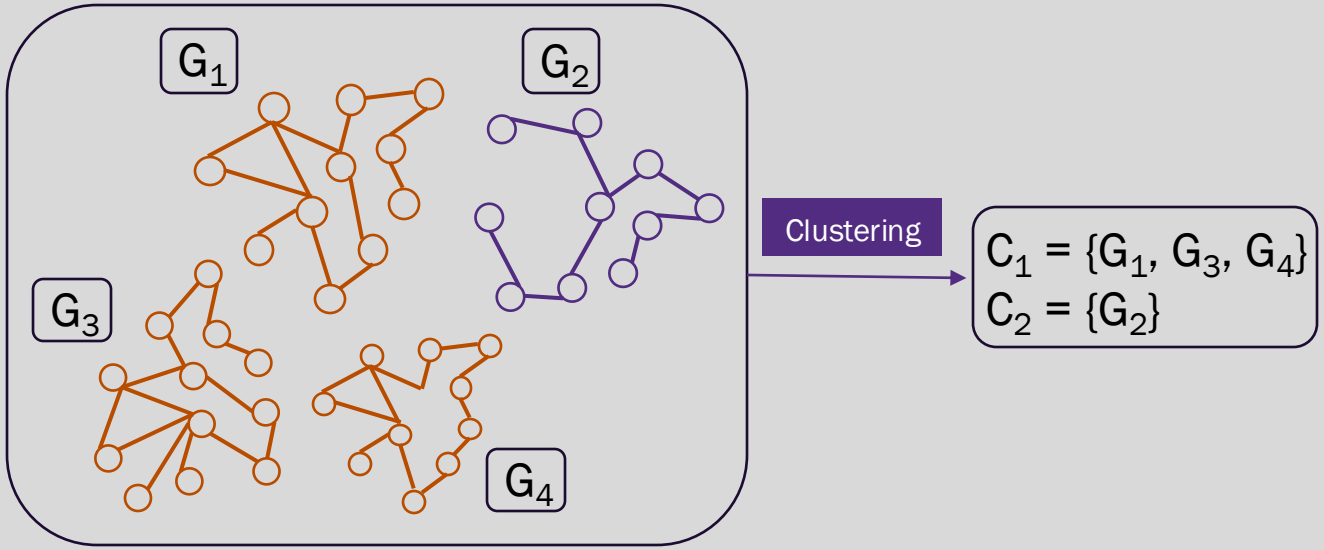
- Node classification – given information about the node (may include neighbors, relationships, attributes), infer class membership (usually an attribute)
- Edge prediction – given information about two nodes, determine if a given relation exists between them
- Community detection (node clustering) – detection of “meaningful” node groups



Whole graph and inter-graph machine learning tasks

- Whole graph classification and regression – given one entire graph, infer class membership or estimate (regress) a related numerical property
- Inter graph tasks – given multiple graphs estimate the similarity between graphs and/or identify clusters



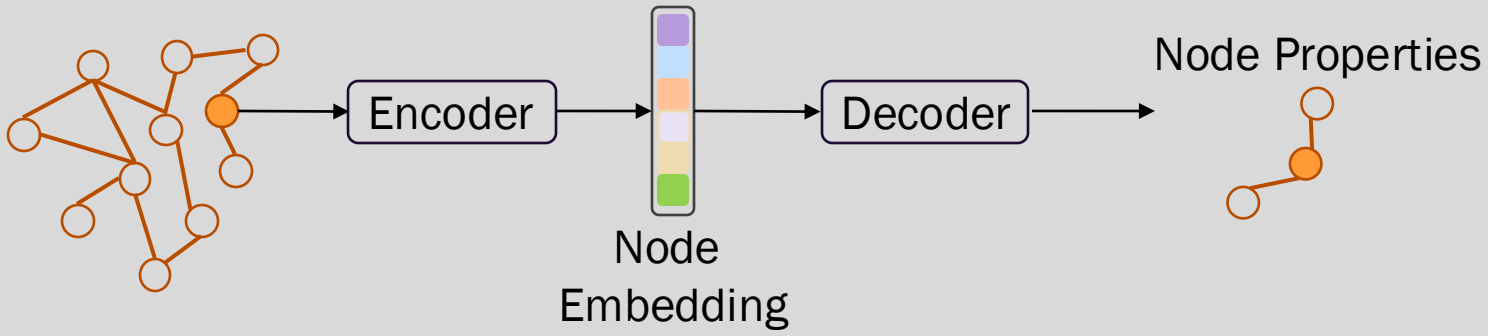


School of COMPUTING
Clemson University

Graph neural networks

Graph representation for machine learning

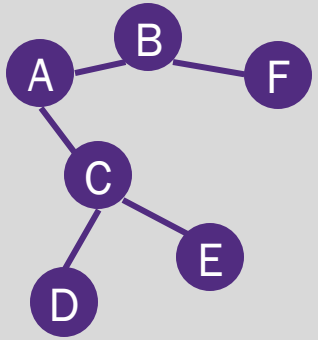
- Traditional approaches utilize engineered features (e.g., node degree, adjacency matrix eigenvalues) to represent nodes, edges, and graphs for analysis
- Graph neural network (GNN) approaches adopt an *encoder – decoder* perspective
  - *encoder* transforms the graph elements (nodes, edges, entire graph) into a latent space
  - *decoder* estimates values of interest (class membership, node attributes, relationships) from the encoded representation



Random walk methods

- Shallow embedding approaches that attempt to encode random walk statistics
- Goal is for the decoder to approximate the probability distribution of visiting a node v on a path of length T starting at node u on graph G

$$P_{G,T}(v|u)$$

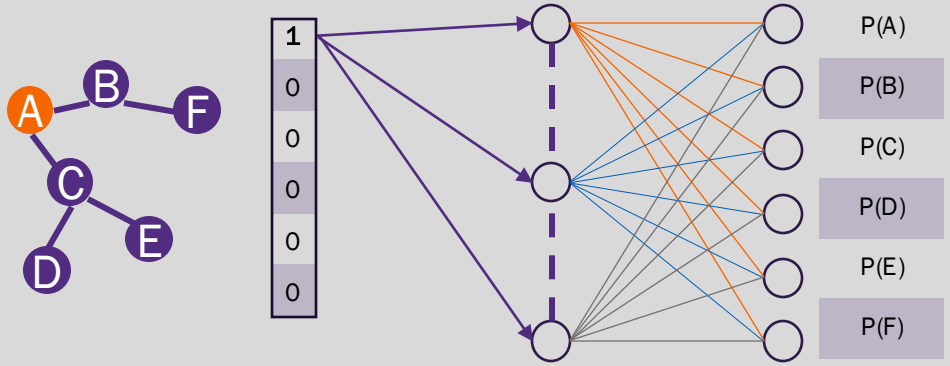


Random Walks

- Assume we have transition states, $p(u, v \mid v \epsilon N(u))$ that define the probability of traversing from node u to v
- Random walks are generated by starting at a randomly selected node, u , and probabilistically selecting the next node from $N(u)$ and repeating
- Sample paths from the graph above include
  - ABF
  - ABACE
  - ABADC

School of COMPUTING Clemson University

+node2vec



Input Node Represented as a 1-hot encoding

Hidden linear layer. Effectively a look up for the weight row corresponding to the input node.

Softmax output represents probabilities of encountering each node in a random walk

Training with Negative Sampling

- Assume we've sampled many random walks
- From those we can generate sample node pairs *(u,v)*
- Computing normalized probabilities over all nodes is expensive
- Instead, for each sample, the input is *u* and the target output is a 1 hot encoding where the value at the index for node *v* is set to 1 and 0 for indices corresponding to a small number of randomly selected nodes

	T=1	T=2
ABF	AB	AF
	BF	
ABACE	AB	AA
	BA	BC
	AC	AE
	CE	
ABACD	AB	AA
	BA	BC
	AC	AD
	CD	

+Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016.

Message passing methods

- Intended to address limitations of shallow embeddings - especially inability to use node feature information and transductive nature (i.e., can only learn embeddings for nodes present during training)
- Methods are motivated by the need to satisfy *permutation invariance,* i.e., the results should not depend on node ordering as there is no natural ordering of nodes in a graph
- Defining characteristic of most recent graph neural networks (GNNs) is the use of *message passing* – vector messages sent between nodes

Convolutional graph neural networks

- Convolutional in the sense that parameters are shared across the graph
- Assume node features are represented by $N \times D$ matrix X (N nodes, D features). For graphs where nodes have no features, set $X = I$
- Every layer, $H^{(l)}$, can be written as

$$H^{(l+1)} = f\left(H^{(l)}, A\right)$$

  where A is the adjacency matrix, $H^{(0)} = X$.
- For *vanilla* convolutional neural network

$$f\left(H^{(l)}, A\right) = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$$

Diagonal node degree matrix (used to normalize rows of A)

Augmented adjacency matrix (diagonal elements all set to 1)

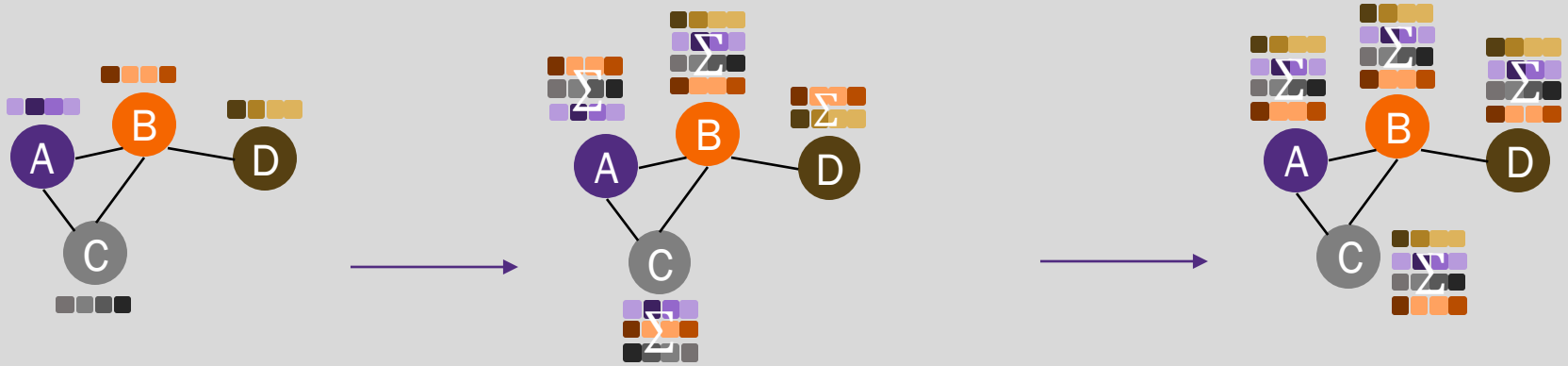
Weight matrix for layer l. Importantly, the dimension of W is independent of the size of the graph.

School of COMPUTING Clemson University

Convolutional graph neural networks

After the first layer, each node representation is now the weighted sum of its input feature representation and its neighbors input features (**the messages**)

After the second layer, each node representation is now the weighted sum of its input feature representation and its 1 and 2 hop neighbors



$$H^{(0)} = X$$

$$H^{(1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}XW^{(1)}\right)$$

$$H^{(2)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(1)}W^{(2)}\right)$$

$\hat{A} =$

	A	B	C	D
A	1	1	1	0
B	1	1	1	1
C	1	1	1	0
D	0	1	0	1

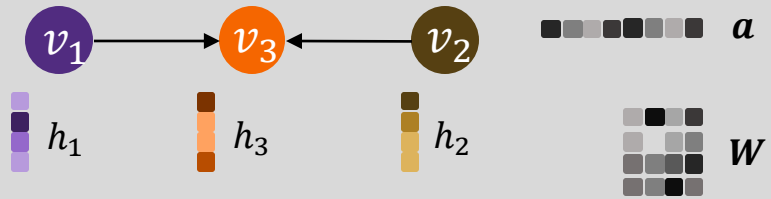
Consider $\hat{A}X$. This updates the feature row for each node as the sum of its features and its neighbor's features

Consider $\hat{A}H^1$. This updates the feature row for each node as in the first layer, but now the representation of each node neighbor contains information from their neighbors as well.

School of COMPUTING Clemson University

Graph attention

- Similar attention mechanism to transformers. Recall the QKV paradigm.
- The key (K) and value (V) are the transformed message from a neighboring node (e.g., output of an MLP applied to message).
- The receiving node uses the transformed version of its feature representation as the query (Q)



Message transformation MLP weights

GCN weights

$$\alpha_{ij} = \frac{exp\left(\sigma\left(\boldsymbol{a}\left[\boldsymbol{W}h_i || \boldsymbol{W}h_j\right]\right)\right)}{\sum_{k \in N_i} exp\left(\sigma\left(\boldsymbol{a}\left[\boldsymbol{W}h_i || \boldsymbol{W}h_k\right]\right)\right)}$$

Attention weight applied to message from node j to node i , where N_i are the neighbors of node i (which includes node i)

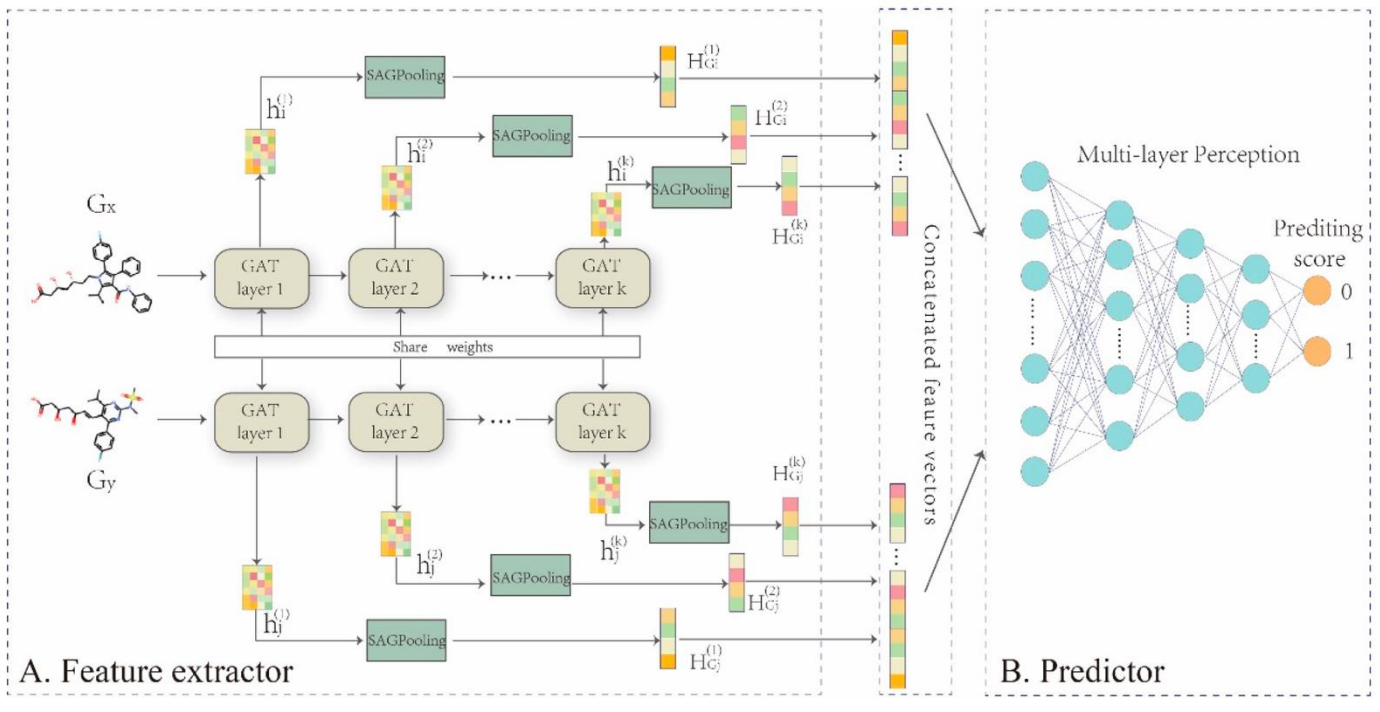
$$h'_i = \sigma\left(\sum_{j \in N_i} \alpha_{ij} \boldsymbol{W}h_j\right)$$

Attention based representation of node i

School of COMPUTING
Clemson University

Applications

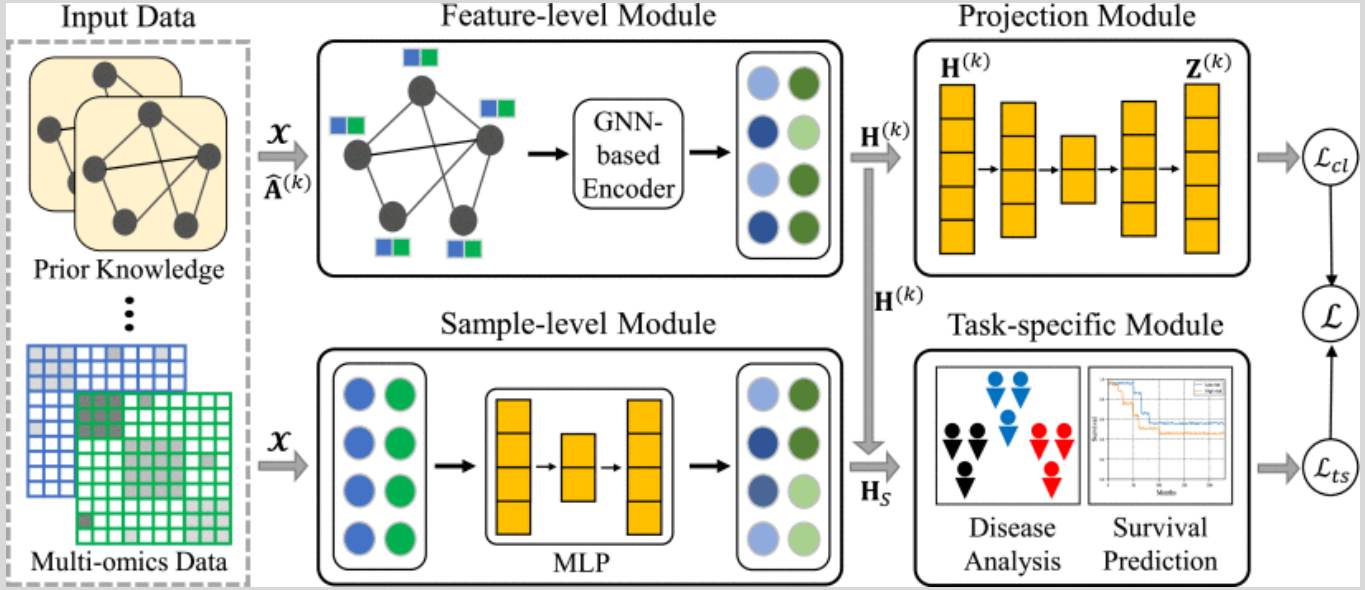
GNN drug-drug interaction prediction



- Each drug is represented by a graph
  - Nodes represent atoms
  - Edges represent bonds
  - Node features (78 dims) include atomic symbol, valence, formal charge, number of free radical electrons, hybridization, number of hydrogen bonds, and aromaticity
- Each drug in a candidate DDI pair is passed through a 5-layer GCN with attention
- Output node representations are down sampled via Self-attention graph (SAG) pooling to form an overall representation of each drug
- Drug representations are concatenated and passed through a FFN for DDI classification

Feng, Y.-H.; Zhang, S.-W. Prediction of Drug-Drug Interaction Using an Attention-Based Graph Neural Network on Drug Molecular Graphs. Molecules 2022, 27, 3004

Integrating prior knowledge for cancer subtype prediction



- Prior knowledge is represented as a graph
  - Nodes represent genes (selected based on omics)
  - Edges represent relationships between genes (bespoke rules based on the knowledge base)
  - Graph representation from GNN encoder
  - Projection model attempts to maximally contrast gene representations from knowledge base
- Patient level omics data processed in MLP
- Output representations of prior knowledge and patient omic data combined and passed to task-specific module

S. Xiao, H. Lin, C. Wang, S. Wang and J. C. Rajapakse, "Graph Neural Networks With Multiple Prior Knowledge for Multi-Omics Data Analysis," in IEEE Journal of Biomedical and Health Informatics, vol. 27, no. 9, pp. 4591-4600, Sept. 2023

School of COMPUTING Clemson University

Biology informed AI – improving explainability

Central hypothesis: AI that utilizes abductive reasoning through integration of formal biology knowledge with observational data will demonstrate improved inference performance and explainability in limited data settings.

Our Methodological Research

- Selective attention mechanisms to identify knowledge graph concepts relevant to AI system output
- Pathway analysis methods to trace the paths in the knowledge graph that lead to a particular prediction