Topics in Deep Learning Assignment

Aaryan Ajay Sharma
IIIT Hyderabad
aaryan.s@research.iiit.ac.in

Preprocess and Analyze Graph Data

Analysis of Dataset

- **Graph 1:** Comprising 1174 vertices and 1417 links, Graph 1 has an average vertex degree of 2.41. Degrees span from a minimum of one to a maximum of 10. With a graph density of 0.0020, this points to a mere 0.2 percent of all possible linkages being utilized, suggesting a rather loose interlinking between vertices. The graph contains 32 triads and lacks any 4-node cliques, signifying the existence of triple-vertex interconnected substructures. Measures of node centrality highlight the significance of specific vertices within this network. Moreover, a clustering coefficient of 0.017 denotes a tendency against tight clustering among nodes, pointing to minimal local interlinking. The positive degree assortativity coefficient of 0.13 reflects a modest propensity for vertices to connect with others of a similar degree. These characteristics, among other measures, afford a detailed insight into the network's structural and connective nuances.
- **Graph 2:** With 3212 vertices and 3423 links, the mean vertex degree in Graph 2 is 2.13, ranging from a low of one to a high of 267. A density of 0.00066 indicates only about 0.07 percent of all possible linkages are actualized, which indicates a network with sparse interconnections. The presence of 14 triads and an absence of 4-node cliques emphasize the presence of small-sized interconnected substructures. Calculations of node centrality underscore the prominence of particular vertices within the network. A notably low clustering coefficient of 0.0037 infers a limited inclination for nodes to cluster, reflecting scant local interconnectivity. With a degree assortativity of -0.28, there is an inclination for vertices to link with others of dissimilar degree. These metrics, along with other parameters, help provide a comprehensive picture of the network's structure and its connections.

Comparing Node2Vec and DeepWalk

- **Graph 1 Performance:**
  - **Node2Vec AUC:** 88.18%
  - **DeepWalk AUC:** 86.09%

  Node2Vec Insights:
  - **Pros:** Achieves higher AUC, demonstrating effective structure capture due to its biased random walk (p = 0.5, q = 2).
  - **Cons:** May miss some complex structures or community patterns in Graph-1.

  DeepWalk Insights:
  - **Pros:** High AUC score signifies meaningful structure capture; algorithm's simplicity aids broad applicability.
  - **Cons:** Its unbiased random walk might overlook some complex patterns.
- **Graph 2 Performance:**
  - **Node2Vec AUC:** 86.39%

- **DeepWalk AUC:** 81.35%

Node2Vec Insights:

- **Pros:** Effective at capturing local structures within Graph-2 despite a lower AUC than for Graph-1.
- **Cons:** Performance possibly constrained by Graph-2's specific features (e.g., size, connectivity).

DeepWalk Insights:

- **Pros:** Slightly better AUC score than Node2Vec for Graph-2, indicating capability in capturing certain patterns; benefits from simplicity and scalability.
- **Cons:** May not fully capture complex structures, especially in densely connected subgraphs.

Conclusion:

- Node2Vec generally outperforms DeepWalk in representing graph structures for both Graph-1 and Graph-2, attributed to its effective local and global information capture.
- The choice between Node2Vec and DeepWalk should be based on the specific graph characteristics and the embedding task's requirements, considering DeepWalk's advantages in simplicity and scalability.

Analyzing Node Centrality Measures

Eigenvector Centrality Analysis:

1. **Problem Statement:**
   - **Objective:** Identify influential papers in a citation network to guide literature review processes.
   - **Data:** A network where nodes represent scientific papers and edges represent citations.
   - **Result:** A ranked list of papers based on their influence and importance in the field.
2. **Rationale for Eigenvector Centrality:**
   - Eigenvector centrality not only considers the quantity of citations (connections) but also the quality, by factoring in the influence of the citing papers. This captures the notion that being cited by highly influential papers is more valuable.
   - This measure is more appropriate than betweenness or closeness centrality because the focus is on the influence within the network, rather than the role of papers in connecting different parts of the network or their closeness to all other papers.
3. **Model Application:** A recommendation system for researchers could use eigenvector centrality scores to suggest key readings. Integrating with machine learning, a classifier such as a Random Forest could predict the future eigenvector centrality of recent papers based on features like the centrality of references, topics, and the publishing journal's impact factor.

Betweenness Centrality Analysis:

1. **Problem Statement:**
   - **Objective:** Optimize logistics and delivery routes within a supply chain network.
   - **Data:** A network where nodes are distribution centers or stores and edges represent transportation routes.
   - **Result:** Identification of critical nodes (centers or routes) essential for efficient logistics.
2. **Rationale for Betweenness Centrality:**
   - **Fit:** Betweenness centrality identifies nodes that serve as bridges between other nodes. This is crucial for understanding choke points in logistics networks.
   - **Reason:** This centrality is chosen over eigenvector and closeness centrality as it highlights the nodes crucial for the flow of goods across the entire network, rather than their influence or accessibility.

3. **Model Application:** A supply chain optimization tool could use betweenness centrality to simulate disruptions (e.g., at highly central nodes) and assess their impact. Decision tree classifiers could help predict the resilience of different network configurations, enabling proactive restructuring for robustness against disruptions.

Closeness Centrality Analysis:

1. **Scenario:**
   - **Objective:** Improve response times in emergency services within a city's road network.
   - **Data:** A network where nodes represent intersections and edges are roads connecting them.
   - **Result:** Strategic locations for emergency services based on access to the entire city.
2. **Rationale for Closeness Centrality:**
   - **Fit:** Closeness centrality measures the average shortest path from a node to all other nodes, identifying the most accessible locations.
   - **Reason:** This measure is ideal over eigenvector and betweenness centralities for emergency services, as the primary concern is minimizing the distance to any potential incident location, ensuring rapid response times.
3. **Model Application:** To support decision-making on where to locate new services, a GIS (Geographic Information System) based model incorporating closeness centrality can be used. Combining this with a k-Nearest Neighbors classifier could predict areas of high demand based on historical incident data, optimizing both location and resource allocation.

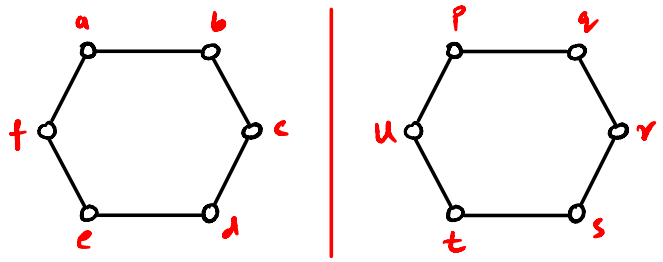
Understanding WL Kernels through Color Refinement

Color refinement in WL Kernels involves iteratively applying hash functions to refine graph node labels, crucial for efficiently capturing graph structural details.

Hashing Process:

- **Initial Step:** Assign initial labels or colors to nodes based on their attributes.
- **Iterative Refinement:** Apply hash functions iteratively, merging node and neighbor labels to encapsulate the graph's structure.
- **Stopping Criteria:** Terminate on reaching stability, convergence, hitting the maximum iteration limit, or when changes drop below a predefined threshold.
- **Implementation Nuances:** Criteria selection depends on balancing accuracy with computational demands, ensuring efficient processing of large graphs.

Q. 2. Isomorphic Pairs :



G_1 G_2

WL-Test

Iteration 1 :

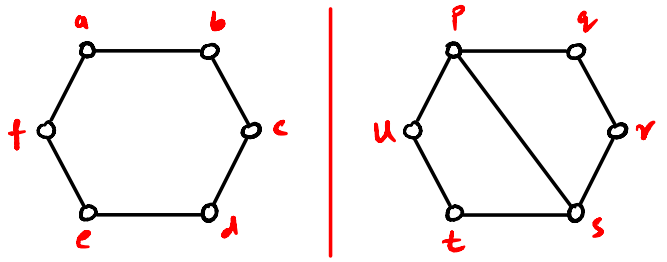
Graph 1			Graph 2		
Node	hash	colour	Node	hash	colour
a	111	3	p	111	3
b	111	3	q	111	3
c	111	3	r	111	3
d	111	3	s	111	3
e	111	3	t	111	3
f	111	3	u	111	3

Iteration 2 :

Graph 1			Graph 2		
Node	hash	colour	Node	hash	colour
a	833	9	p	833	9
b	333	9	q	333	9
c	333	9	r	333	9
d	333	9	s	333	9
e	333	9	t	333	9
f	333	9	u	333	9

Since colours have not changed from iteration 1 & 2, the test has converged and we get the graphs to be isomorphic.

Non-Isomorphic Pairs:



G_1 G_2

WL-Test

Iteration 1:

Graph 1			Graph 2		
Node	hash	colour	Node	hash	colour
a	111	3	p	1111	4
b	111	3	q	111	3
c	111	3	r	111	3
d	111	3	s	1111	4
e	111	3	t	111	3
f	111	3	u	111	3

Iteration 2:

Graph 1			Graph 2		
Node	hash	colour	Node	hash	colour
a	333	9	p	3344	14
b	333	9	q	334	10
c	333	9	r	334	10
d	333	9	s	3344	14
e	333	9	t	334	10
f	333	9	u	334	14

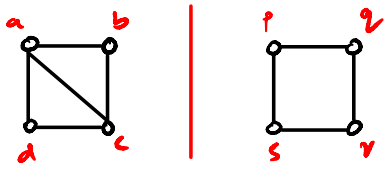
Since colours have not changed from iteration 1 & 2, the test has converged and we get the graphs to be isomorphic.

Expressivity in GNN

1. Provide five examples of pairs of graphs (of at least 3 nodes in each graph) where GCN and GraphSAGE fail to distinguish between the nodes in the graphs in each pair, but GIN is able to distinguish. Show GCN (mean pool), GraphSAGE (max pool) and GIN (sum) operations on these five pairs of graphs. Also run WL test to show that the graphs in each pair are different from each other.
2. Provide five examples of pairs of graphs (of at least 3 nodes in each graph) where GIN fails to distinguish between the nodes in the graphs in each pair. Run WL test to prove that the GIN fails.

Q. 3. a)

1.



Mean : a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

Max : a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

Sum : a, b, c, d = 3, 2, 3, 2

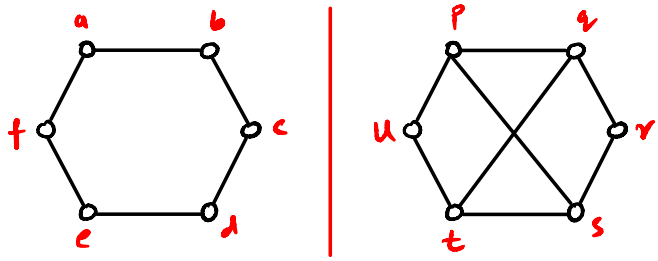
p, q, r, s = 2, 2, 2, 2

WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	1111	p	111
b	111	q	111
c	1111	r	111
d	111	s	111

∴ The Graphs are non-isomorphic

2.



Mean : a, b, c, d, e, f = 1, 1, 1, 1, 1, 1

p, q, r, s, t, u = 1, 1, 1, 1, 1, 1

Max : a, b, c, d, e, f = 1, 1, 1, 1, 1, 1

p, q, r, s, t, u = 1, 1, 1, 1, 1, 1

Sum : a, b, c, d, e, f = 2, 2, 2, 2, 2, 2

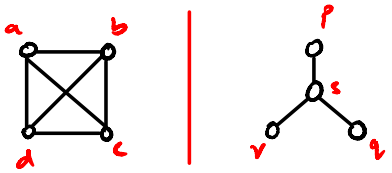
p, q, r, s, t, u = 3, 3, 2, 3, 3, 2

WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	111	p	1111
b	111	q	1111
c	11	r	111
d	111	s	1111
e	111	t	1111
f	111	u	111

∴ The Graphs are non-isomorphic.

3.



Mean : a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

Max : a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

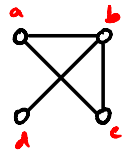
WL-Test

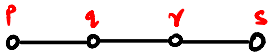
Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	1111	p	11
b	1111	q	11
c	1111	r	11
d	1111	s	1111

∴ The Graphs are non-isomorphic

Sum: a, b, c, d = 3, 3, 3, 3

p, q, r, s = 1, 1, 1, 3

4. 



Mean: a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

Max: a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

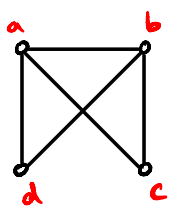
Sum: a, b, c, d = 2, 3, 2, 1

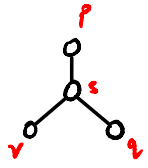
p, q, r, s = 1, 2, 2, 1

WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	111	p	11
b	1111	q	111
c	111	r	111
d	11	s	11

∴ The Graphs are non-isomorphic

5. 



Mean: a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

Max: a, b, c, d = 1, 1, 1, 1

p, q, r, s = 1, 1, 1, 1

Sum: a, b, c, d = 3, 3, 2, 2

p, q, r, s = 1, 1, 1, 3

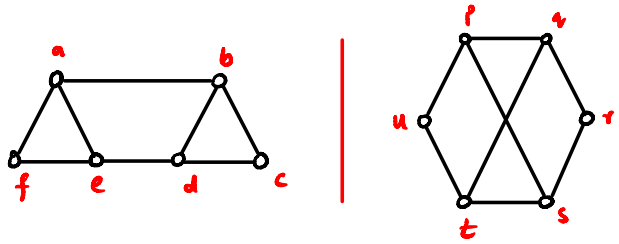
WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	1111	p	11
b	1111	q	11
c	111	r	11
d	111	s	1111

∴ The Graphs are non-isomorphic

b) GIN fails to differentiate

1)



sum: a, b, c, d, e, f = 3, 3, 2, 3, 3, 2

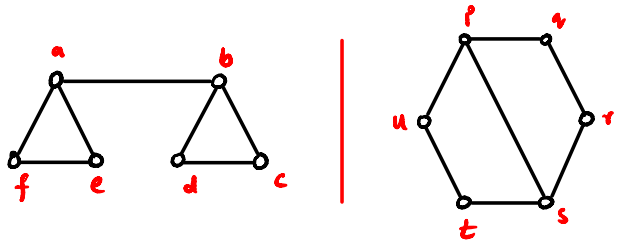
p, q, r, s, t, u = 3, 3, 2, 3, 3, 2

WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	1111	p	1111
b	1111	q	1111
c	111	r	111
d	1111	s	1111
e	1111	t	1111
f	111	u	111

∴ WL-Test fails

2)



Sum: a, b, c, d, e, f = 3, 3, 2, 2, 2, 2

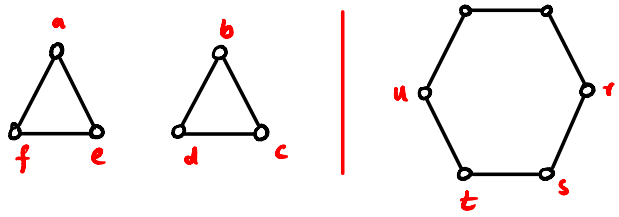
p, q, r, s, t, u = 3, 2, 2, 3, 2, 2

WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	1111	p	1111
b	1111	q	111
c	111	r	111
d	111	s	1111
e	111	t	111
f	111	u	111

∴ WL-Test fails

3)



Sum: a, b, c, d, e, f = 2, 2, 2, 2, 2, 2

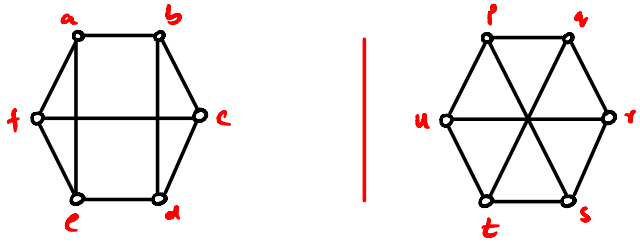
p, q, r, s, t, u = 2, 2, 2, 2, 2, 2

WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	111	p	111
b	111	q	111
c	111	r	111
d	111	s	111
e	111	t	111
f	111	u	111

∴ WL-Test fails

4)



sum: a, b, c, d, e, f = 3, 3, 3, 3, 3, 3

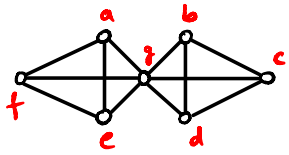
p, q, r, s, t, u = 3, 3, 3, 3, 3, 3

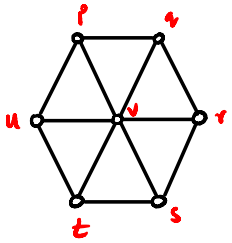
WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	1111	p	1111
b	1111	q	1111
c	1111	r	1111
d	1111	s	1111
e	1111	t	1111
f	1111	u	1111

∴ WL-Test fails

5)





Sum: a, b, c, d, e, f, g = 3, 3, 3, 3, 3, 3, 6

p, q, r, s, t, u, v = 3, 3, 3, 3, 3, 3, 6

WL-Test

Graph 1		Graph 2	
Node	Embedding	Node	Embedding
a	11111	p	11111
b	11111	q	11111
c	11111	r	11111
d	11111	s	11111
e	11111	t	11111
f	11111	u	11111
g	11111	g	11111

∴ WL-Test fails