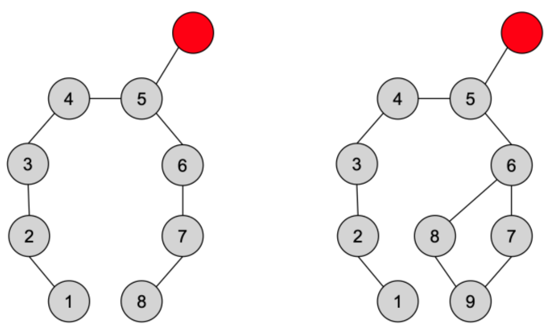
Graph with Machine Learning course assignment

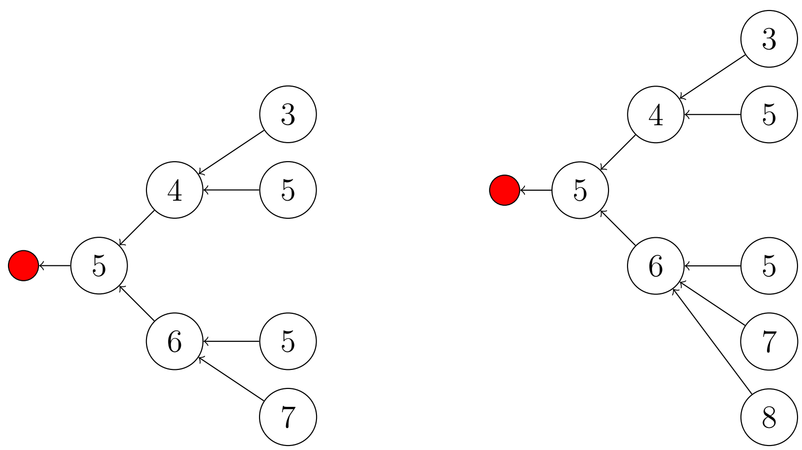
Zahra Tebyanian

2.1 Effect of Depth on Expressiveness (4 points)

Consider the following 2 graphs, where all nodes have 1-dimensional initial feature vector $x = [1]$. We use a simplified version of GNN, with no nonlinearity, no learned linear transformation, and sum aggregation. Specifically, at every layer, the embedding of node v is updated as the sum over the embeddings of its neighbors (N_v) and its current embedding h_v^t to get h_v^{t+1}. We run the GNN to compute node embeddings for the 2 red nodes respectively. Note that the 2 red nodes have different 5-hop neighborhood structure (note this is not the minimum number of hops for which the neighborhood structure of the 2 nodes differs). How many layers of message passing are needed so that these 2 nodes can be distinguished (i.e., have different GNN embeddings)?



Answer: Three layers is needed:



Embedding of red node in the left one is 4 and in the right one is 5.

2.3 Relation to Random Walk (i) (4 points)

Let's explore the similarity between message passing and random walks. Let $h_i^{(l)}$ be the embedding of node i at layer l. Suppose that we are using a mean aggregator for message passing, and omit the learned linear transformation and non-linearity: $h_i^{(l+1)} = \frac{1}{|N_i|}\sum_{j \in N_i} h_j^{(l)}$. If we start at a node u and take a uniform random walk for 1 step, the expectation over the layer-l embeddings of nodes we can end up with is $h_u^{(l+1)}$, exactly the embedding of u in the next GNN layer. What is the transition matrix of the random walk? Describe the transition matrix using the adjacency matrix A, and degree matrix D, a diagonal matrix where $D_{i,i}$ is the degree of node i.

Answer: The transition matrix would be: $D^{-1}A$.

2.6 Learning BFS with GNN (7 points)

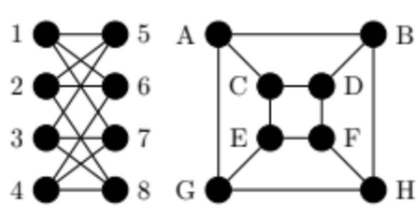
Next, we investigate the expressive power of GNN for learning simple graph algorithms. Consider breadth-first search (BFS), where at every step, nodes that are connected to already visited nodes become visited. Suppose that we use GNN to learn to execute the BFS algorithm. Suppose that the embeddings are 1-dimensional. Initially, all nodes have input feature 0, except a source node which has input feature 1. At every step, nodes reached by BFS have embedding 1, and nodes not reached by BFS have embedding 0. Describe a message function, an aggregation function, and an update rule for the GNN such that it learns the task perfectly.

Answer: We consider identity as massage function, and maximum as aggregation function so the update rule would be result of aggregation function:

$$h^l = max(h^{l-1})$$

4.1 Isomorphism Check(1 point)

Are the following two graphs isomorphic? If so, demonstrate an isomorphism between the sets of vertices. To demonstrate an isomorphism between two graphs, you need to find a 1-to-1 correspondence between their nodes and edges. If these two graphs are not isomorphic, prove it by finding a structure (node and/or edge) in one graph which is not present in the other.



Answer: Desired function would be:

$$\text{A} \longleftrightarrow 6,\ \text{B} \longleftrightarrow 2,\ \text{C} \longleftrightarrow 4,\ \text{D} \longleftrightarrow 8,$$
$$\text{E} \longleftrightarrow 7,\ \text{F} \longleftrightarrow 3,\ \text{G} \longleftrightarrow 1,\ \text{H} \longleftrightarrow 5$$

4.2 Aggregation Choice (3 points)

The choice of the AGGREGATE($\cdot$) is important for the expressiveness of the model above. Three common choices are:

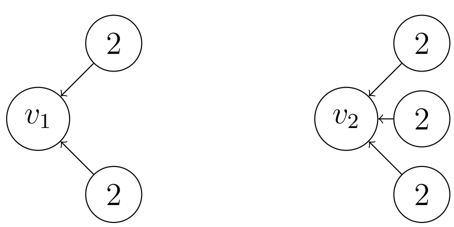
$$\text{AGGREGATE}_{\text{max}}\left(\left\{h_u^{(k-1)}, \forall u \in \mathcal{N}(v)\right\}\right)_i = \max_{u \in \mathcal{N}(v)}\left(h_u^{(k-1)}\right)_i \text{ (element-wise max}\Big)$$

$$\text{AGGREGATE}_{\text{mean}}\left(\left\{h_u^{(k-1)}, \forall u \in \mathcal{N}(v)\right\}\right) = \frac{1}{|\mathcal{N}(v)|}\sum_{u \in \mathcal{N}(v)}\left(h_u^{(k-1)}\right)$$

$$\text{AGGREGATE}_{\text{sum}}\left(\left\{h_u^{(k-1)}, \forall u \in \mathcal{N}(v)\right\}\right) = \sum_{u \in \mathcal{N}(v)}\left(h_u^{(k-1)}\right)$$

Give an example of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ and their initial node features, such that for two nodes $v_1 \in V_1$ and $v_2 \in V_2$ with the same initial features $h_{v_1}^{(0)} = h_{v_2}^{(0)}$, the updated features $h_{v_1}^{(1)}$ and $h_{v_2}^{(1)}$ are equal if we use mean and max aggregation, but different if we use sum aggregation.
Hint: Your node features can be scalars rather than vectors, i.e. one dimensional node features instead of n-dimensional. Also, You are free to arbitrarily choose the number of nodes (e.g. 3 nodes), their connections (i.e. edges between nodes) in your example.

Answer: V_1 and V_2 are the following graphs. Initial features are considered to be scalar 2.



In V_1, embedding of node v_1:
with mean: 2
with max: 2
with sum: 4

In V_2, embedding of node v_2:
with mean: 2

with max: 2
with sum: 6

4.3 Weisfeiler-Lehman Test (6 points)

Answer: Let's start with a definition:

Definition 3.3.2. Two nodes u, v are said to be *unfolding equivalent* $u \backsim_{ue} v$, if $\mathbf{T}_u = \mathbf{T}_v$. Analogously, two graphs $\mathbf{G}_1, \mathbf{G}_2$ are said to be *unfolding equivalent* $\mathbf{G}_1 \backsim_{ue} \mathbf{G}_2$, if there exists a bijection between the nodes of the graphs that respects the partition induced by the unfolding equivalence on the nodes[4]. ■

where T_u is computation graph of node u.
Now in order to answer the question, we prove the following theorem:

Theorem 4.1.1.
Let $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ be a labeled graph. Then, for each $u, v \in \mathbf{V}$, $u \backsim_{ue} v$ if and only if $u \backsim_{WL} v$ holds.

Let's prove the following lemma first:

Lemma A.0.1.
Let $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ be a graph and let $u, v \in \mathbf{V}$, with features ℓ_u, ℓ_v. Then, $\forall t \in \mathbb{N}$

$$\mathbf{T}_u^t = \mathbf{T}_v^t \;\; \textit{iff} \;\; c_u^{(t)} = c_v^{(t)} \tag{4}$$

where $c_u^{(t)}$ and $c_v^{(t)}$ represent the node coloring of u and v at time t, respectively.

Proof. The proof is carried out by induction on t, which represents both the depth of the unfolding trees and the iteration step in the WL colouring.

For $t = 0$, $\mathbf{T}_u^0 = \text{Tree}(\ell_u) = \text{Tree}(\ell_v) = \mathbf{T}_v^0$ if and only if $\ell_u = \ell_v$ and $c_u^{(0)} = \text{HASH}_0(\ell_u) = \text{HASH}_0(\ell_v) = c_v^{(0)}$. Let us suppose that Eq. (4) holds for $t-1$, and prove that it holds also for t.

($\rightarrow$) Assuming that $\mathbf{T}_u^t = \mathbf{T}_v^t$, we have

$$\mathbf{T}_u^{t-1} = \mathbf{T}_v^{t-1} \tag{5}$$

and

$$\text{Tree}(\ell_u, \mathbf{T}_{ne[u]}^{t-1}) = \text{Tree}(\ell_v, \mathbf{T}_{ne[v]}^{t-1}) \tag{6}$$

By induction, Eq. (5) is true if and only if

$$c_u^{(t-1)} = c_v^{(t-1)} \tag{7}$$

Eq. (6) implies that $\ell_u = \ell_v$ and $\mathbf{T}_{ne[u]}^{t-1} = \mathbf{T}_{ne[v]}^{t-1}$, which means that an ordering on $ne[u]$ and $ne[v]$ exists s.t.

$$T_{ne_i(u)}^{t-1} = T_{ne_i(v)}^{t-1} \; \forall\, i = 1, \ldots, |ne[u]| \tag{8}$$

Hence, Eq. (8) holds iff an ordering on $ne[u]$ and $ne[v]$ exists s.t.

$$c_{ne(u)_i}^{t-1} = c_{ne(v)_i}^{t-1} \; \forall i = 1, \ldots, |ne[u]|$$

that is

$$\{c_m^{(t-1)} | m \in ne[u]\} = \{c_n^{(t-1)} | n \in ne[v]\} \tag{9}$$

Putting together Eqs. (7) and (9), we obtain:

$$\text{HASH}((c_u^{(t-1)}, \{c_m^{(t-1)} | m \in ne[u]\})) =$$
$$\text{HASH}((c_v^{(t-1)}, \{c_n^{(t-1)} | n \in ne[v]\}))$$

which implies that $c_u^{(t)} = c_v^{(t)}$

(←) The proof of the converse implication follows a similar reasoning, but some different steps are required in order to reconstruct the unfolding equivalence from the equivalence based on the 1–WL test.

Let us assume that $c_u^{(t)} = c_v^{(t)}$; by definition,

$$\begin{gathered} \text{HASH}((c_u^{(t-1)}, \{c_m^{(t-1)} | m \in ne[u]\})) = \\ \text{HASH}((c_v^{(t-1)}, \{c_n^{(t-1)} | n \in ne[v]\})) \end{gathered} \tag{10}$$

Being the HASH function bijective, Eq. (10) implies that:

$$c_u^{(t-1)} = c_v^{(t-1)} \tag{11}$$

and

$$\{c_m^{(t-1)} | m \in ne[u]\} = \{c_n^{(t-1)} | n \in ne[v]\} \tag{12}$$

Eq. (11) is true if and only if, by induction,

$$\mathbf{T}_u^{t-1} = \mathbf{T}_v^{t-1} \tag{13}$$

which implies

$$\ell_u = \ell_v \tag{14}$$

Moreover, Eq. (12) means that an ordering on $ne[u]$ and $ne[v]$ exists such that

$$c_{ne(u)_i}^{t-1} = c_{ne(v)_i}^{t-1} \; \forall i = 1, \ldots, |ne[u]| \tag{15}$$

Instead, by induction, Eq. (15) holds iff an ordering on $ne[u]$ and $ne[v]$ exists so as $T_{ne_i(u)}^{t-1} = T_{ne_i(v)}^{t-1} \; \forall \, i = 1, \ldots, |ne[u]|$, i.e.

$$\mathbf{T}_{ne[u]}^{t-1} = \mathbf{T}_{ne[v]}^{t-1} \tag{16}$$

Finally, putting together Eqs. (14) and (16), we obtain

$$\text{Tree}(\ell_u, \mathbf{T}_{ne[u]}^{t-1}) = \text{Tree}(\ell_v, \mathbf{T}_{ne[v]}^{t-1})$$

that means $\mathbf{T}_u^t = \mathbf{T}_v^t$

Theorem 4.1.1 is therefore proven, as its statement just rephrases the statement of Lemma A.0.1 in terms of the equivalence notation.

References

[1] Giuseppe Alessio D'Inverno, Monica Bianchini, Maria Lucia Sampoli, Franco Scarselli. A NEW PERSPECTIVE ON THE APPROXIMATION CAPABILITY OF GNN S