Lecture 11 EN.553.744 Prof. Luana Ruiz

▶ Graph isomorphism network

A GIN layer is defined as

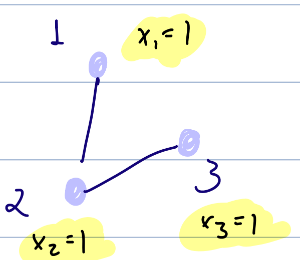
$$[x_\ell]_v = \sigma\left(W_\ell\left((1+\epsilon_\ell)[x_{\ell-1}]_v + \sum_{u \in \mathcal{N}(v)} [x_{\ell-1}]_u\right)\right)$$

$\phi_\ell \circ \sum \varphi_{\ell-1}$: a perceptron or MLP over the neighborhood multiset

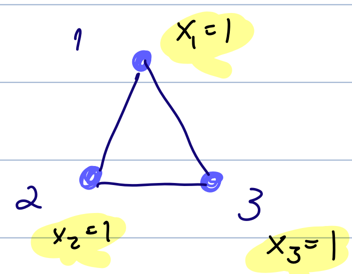
(Hornik 1989) an MLP can model any injective function due to the universal approx. theorem & injective $\phi_\ell \circ \sum_X \varphi_{\ell-1}$ exist for multiset X;

E.g.:

G_1

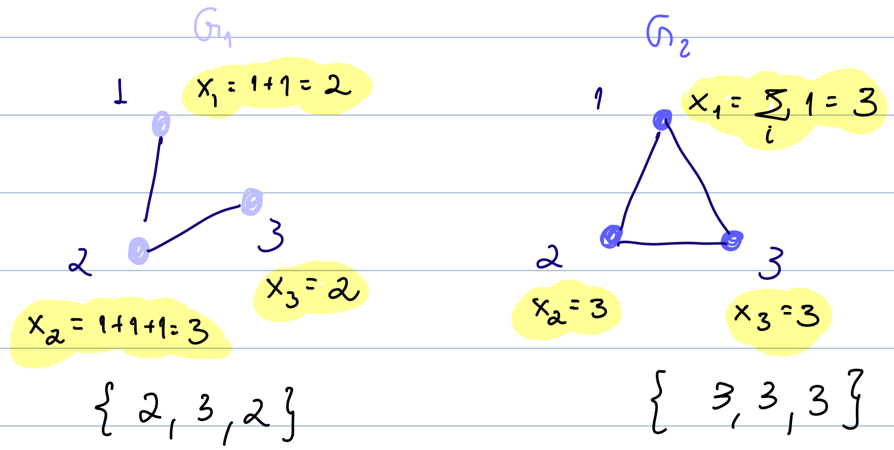


G_2



$(\ell = 0)$

→ Can GIN tell them apart? (assume $W_\ell = 1$, $\epsilon_\ell = 0$)



$(l = 1)$

▶ Graph level readout of GIN

$$X_G = \text{CONCAT}(\text{READOUT}(\{[X_l]_v, v \in V\}) \mid l = 1, \dots, L)$$

$G_1 : [7 \mid 3]$ $\qquad$ $G_2 : [9 \mid 3]$

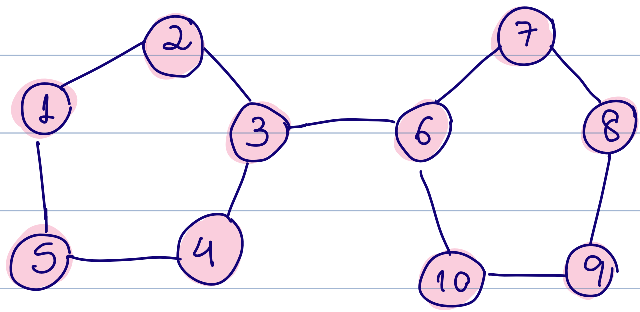
→ not needed, but authors claim better generalization

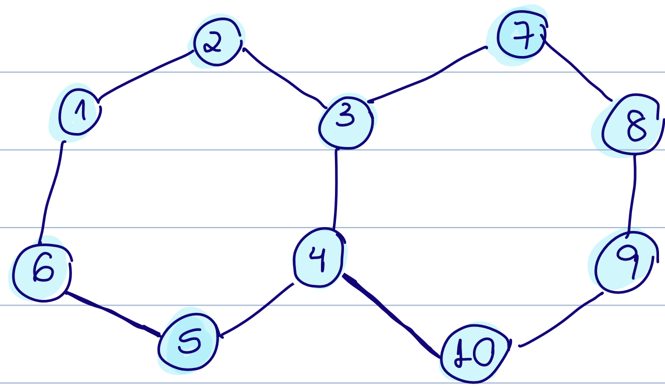
→ with Σ as readout, GIN generalizes the WL test

↳ GIN is a maximally powerful GNN
(for anonymous inputs $\vec{x} = \mathbb{1}$) !

▶ Is being as powerful as WL enough?

E.g.: Can WL distinguish between the two graphs below?





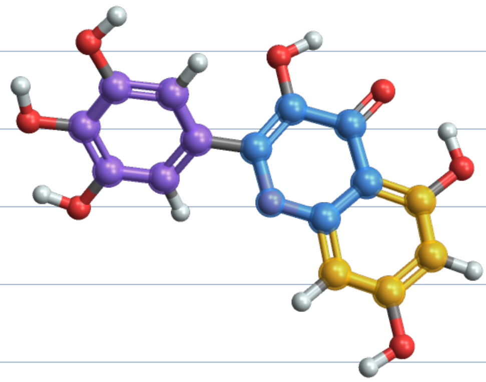
No! The computational graphs are the same (check)

One easy way to discriminate these graphs would be to count cycles:

2 & 3

But since GNNs can't discriminate them, they can't count cycles

Why do we care?



Many biological compounds have cycles in their network representations.

But there is another way to count cycles: densities of cycle homomorphisms

(DEF) Graph homomorphisms: Let $G=(V,E)$ & $F=(V',E')$. A homomorphism from F to G is a map $\gamma: V' \to V$:

$$(i,j) \in E' \Rightarrow (\gamma(i), \gamma(j)) \in E.$$

I.e., homomorphisms are adjacency preserving maps (of F)

Ex.: how is a homomorphism $\neq$ than an isomorphism?

We will denote the total number of homomorphisms from F to G hom(F, G)

(DEF) Homomorphism density : the hom. density from F to G, denoted $t(F, G)$:

$$t(F, G) = \frac{\hom(F, G)}{|V|^{|V'|}}$$

(Claim) Let C_k is the k-cycle. Let G is a graph with adjacency eigenvalues $\lambda_1, \lambda_2, \ldots$ For $k \geq 2$: we have:

$$\boxed{t(C_k, G) = \left(\sum_i \lambda_i^k\right) / n^k}$$

Pf. $$\hom(C_k, G) = \sum_{i,j,k,l\ldots} (A_{ij} A_{jk}) A_{kl} A_{lm} \cdots A_{3i}$$

$$= \sum_{i,k,l\ldots} \boxed{\sum_j (A_{ij} A_{jk})} A_{kl} A_{lm} \cdots A_{3i}$$

$$\sum_{i,k,l\ldots} [A^2]_{ik} A_{kl} A_{lm} \cdots A_{3i}$$

$$= \sum_i [A^k]_{ii} = tr(A^k) = \sum_i \lambda_i^k$$

$$\therefore t(C_k, G) = \sum_i \lambda_i^k / n^k$$

⇒ Hence cycles can be counted using a conv. GNN with white input (see Lec. 9) with GSO $S = A/n$

↳ $E[\hat{x}_i]_i = 1 \quad \forall i$

↳ white inputs help improve expressivity (as opposed to anonymous $x_i = 1 \quad \forall i$)

The "best of both worlds" can be achieved by merging the two methods: GIN with white inputs. Since designing a white input requires computing eigendecompositions, we relax this to random inputs, which almost surely satisfy $[\hat{x}]_i \neq 0$. → TBC in HW2