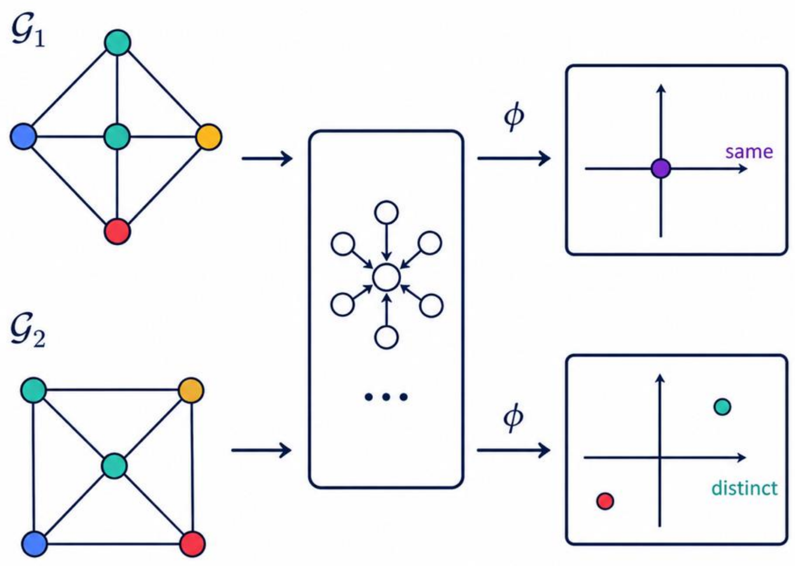
Expressivity is a collision problem.

01 Require invariance to graph relabeling

02 Preserve useful structural distinctions

03 Use 1-WL as a testable benchmark

04 Understand the assumptions behind GIN



Expressivity asks which structural differences a model can preserve.

Three questions guide us

01 **WHAT MAKES TWO GRAPHS THE SAME?**

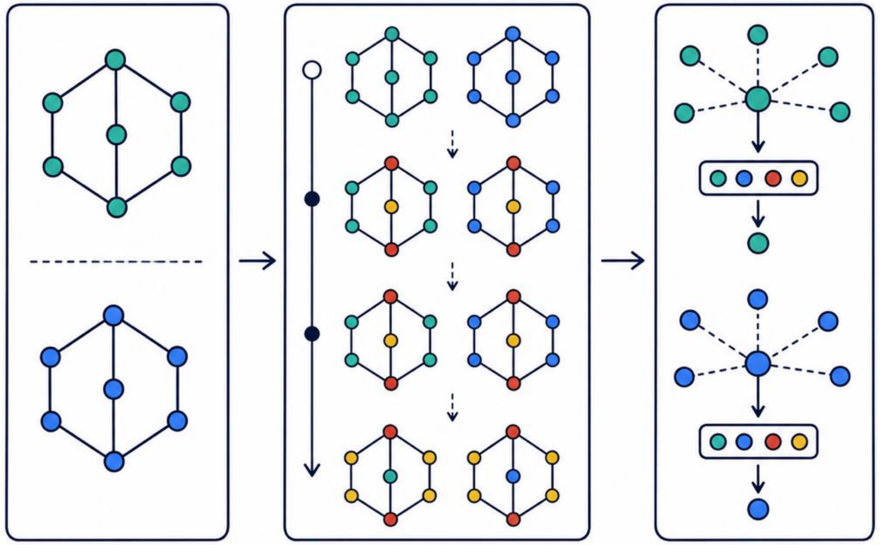
Formalize graph isomorphism and permutation-invariant graph readout.

02 **WHAT CAN 1-WL CERTIFY?**

Refine neighborhood colors, compare histograms, and expose failure cases.

03 **WHEN DOES A GNN MATCH 1-WL?**

Require injective aggregation, combination, and readout -- derive GIN.



Graph tasks need a graph-level readout

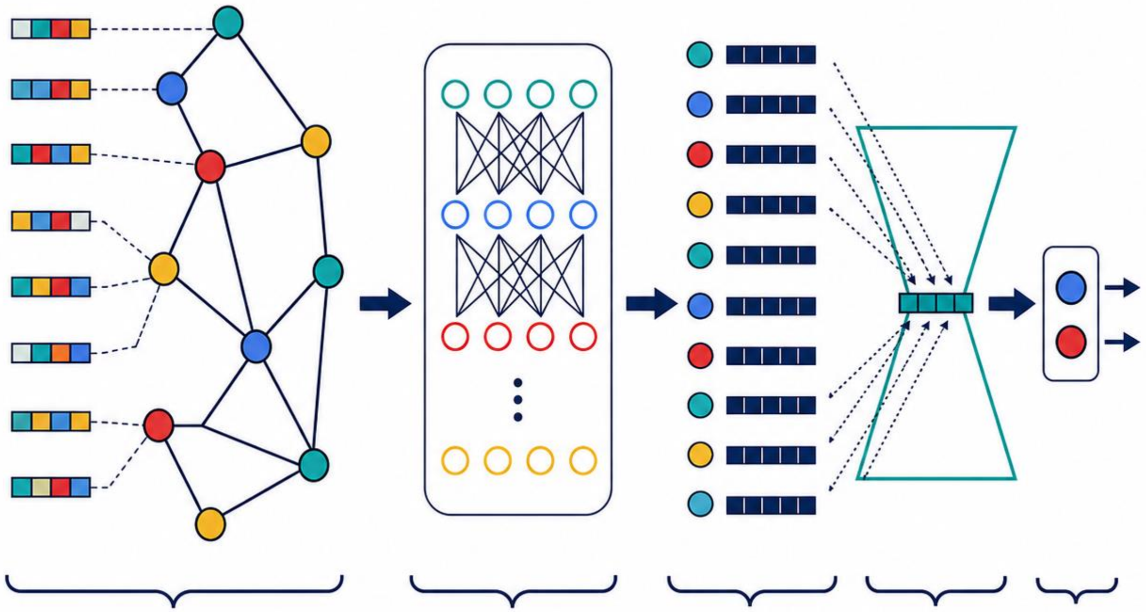
$$H^{(L)} \in \mathbb{R}^{n \times d}, \qquad z_G \in \mathbb{R}^q$$

Node layers produce one embedding per node;

graph tasks require one fixed-length representation.

GRAPH-LEVEL MODEL

$$z_G = \mathrm{READOUT}\Big(\{\{h_v^{(L)} : v \in V\}\}\Big), \quad \hat{y} = g(z_G)$$



Input: $G = (V, E)$ and node features X.
Output: a class or scalar for the whole graph.

Might lose information, e.g. averaging can be the same for duplicated or single collection.

Readout must ignore node order without discarding useful structure.

Graph readout must ignore node order

VALID SYMMETRY

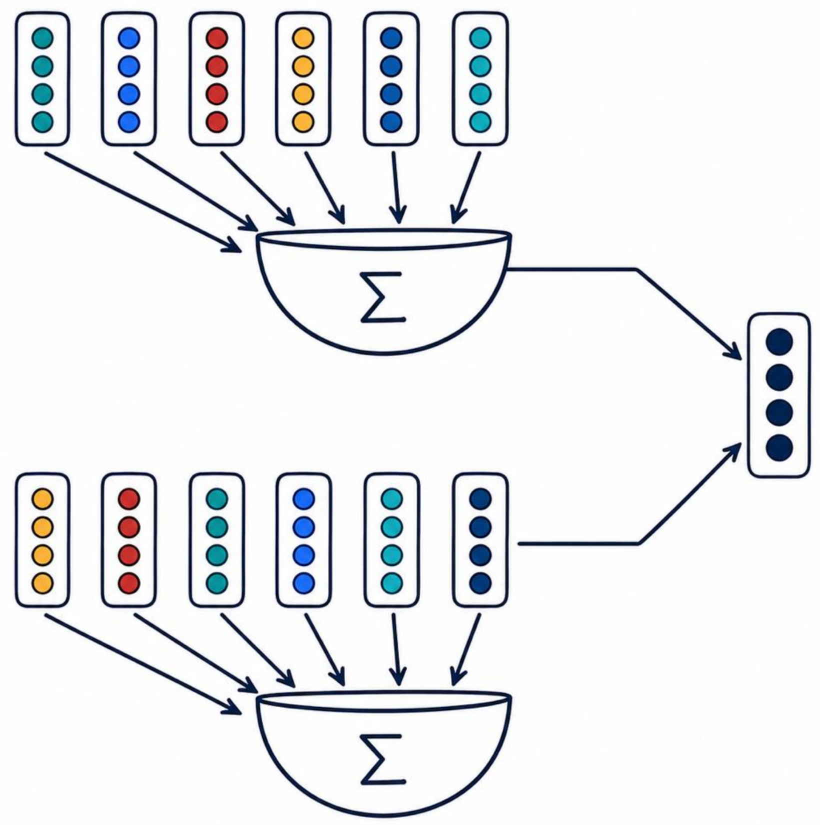
Does the representation change when I merely relabel nodes?

SUM, MEAN, and MAX return the same value after any permutation of the node rows.

CAPACITY TO SEPARATE

Can the representation change when the graph structure meaningfully changes?

The representation should be able to retain structural differences that the task needs.



Invariance prevents false differences; expressivity prevents false matches.

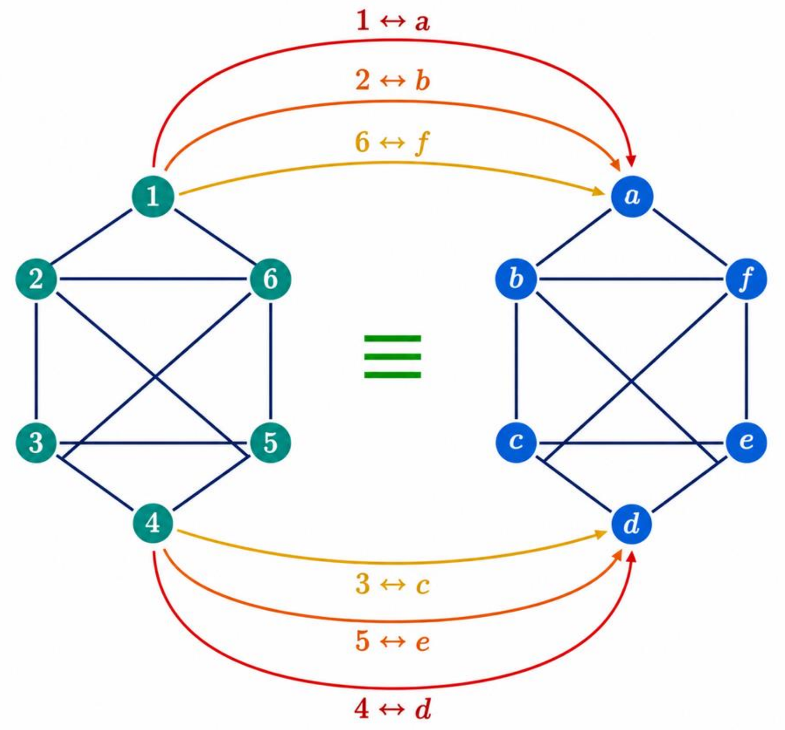
Isomorphism is edge-preserving relabeling

BIJECTION

$\pi: V \rightarrow V'$ is a one-to-one correspondence between all nodes of G and G'.

EDGE PRESERVATION

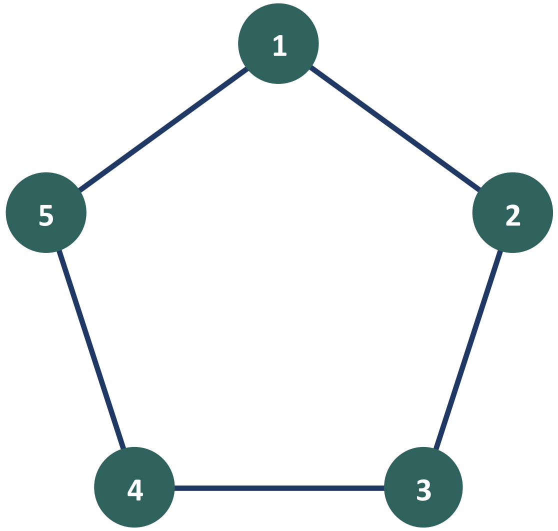
$$(u, v) \in E \quad \Longleftrightarrow \quad (\pi(u), \pi(v)) \in E'$$



For attributed graphs, the correspondence also preserves node and edge attributes.

Isomorphic drawings are the same abstract graph under relabeling.

Example: pentagon and pentagram are one graph



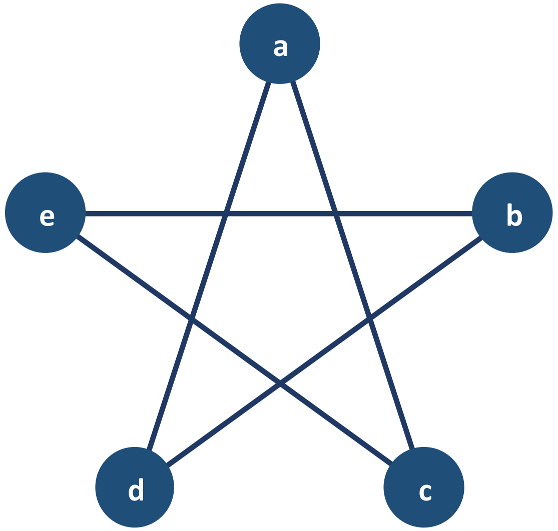
G: drawn as a pentagon
Edges 1-2, 2-3, 3-4, 4-5, 5-1

$\cong$

RELABELING π

$1 \to \mathrm{a}$
$2 \to \mathrm{c}$
$3 \to \mathrm{e}$
$4 \to \mathrm{b}$
$5 \to \mathrm{d}$

Every edge (u, v) of $\boldsymbol{G}$ lands on an edge $\big(\pi(u), \pi(v)\big)$ of $\boldsymbol{G}'$, and vice versa.



G': drawn as a pentagram
Edges a-c, c-e, e-b, b-d, d-a

Same 5 nodes, 5 edges, every node of degree 2: relabel with π and the edge sets coincide.

Invariance and separation are different requirements

Relabeling must not change a graph representation.

$$f_\theta(PAP^\top, PX) = f_\theta(A, X)$$

A distinguishable pair can be separated by some parameter choice:

$$\exists\theta: f_\theta(G) \neq f_\theta(G')$$

Expressivity ≠ successful training ≠ generalization.

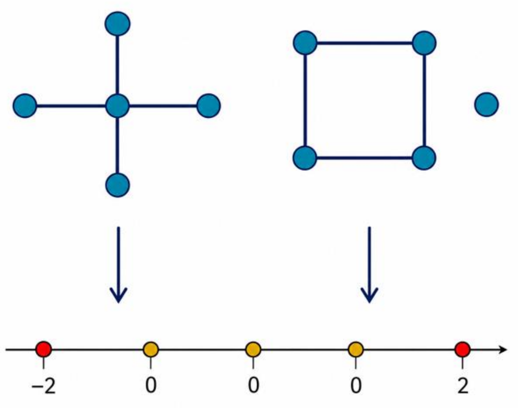
Different graph representations may still lead to the same class label.

An architectural limitation is a collision that persists for every parameter choice.

The spectrum is invariant, but not complete

$$\mathrm{spec}(A) = \mathrm{spec}(PAP^{\mathsf{T}})$$

Relabeling preserves eigenvalues, yet non-isomorphic graphs can be adjacency-cospectral.



ADJACENCY SPECTRA

$$\mathrm{spec}(K_{1,4}) = \mathrm{spec}(C_4 \sqcup K_1) = \{-2, 0, 0, 0, 2\}$$

One graph is connected; the other has an isolated node.

While their Laplacian spectra differ.

$A(K_{1,4})$

$$\begin{matrix} 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 \end{matrix}$$

$A(C_4 \sqcup K_1)$

$$\begin{matrix} 0 & \mathbf{1} & 0 & \mathbf{1} & 0 \\ \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\ 0 & \mathbf{1} & 0 & \mathbf{1} & 0 \\ \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Different matrices, same eigenvalues $\{-2, 0, 0, 0, 2\}$

A spectral signature is an invariant, not a complete isomorphism test.

Two graphs have different spectra → They are non-isomorphic

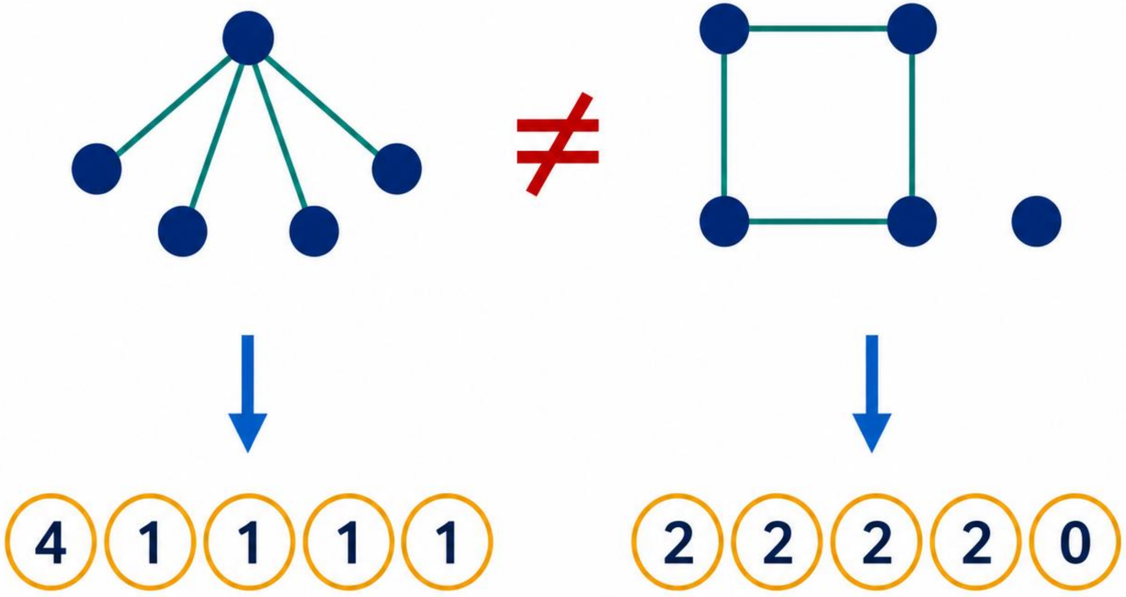
Degree multisets resolve the cospectral pair

STAR $K_{1,4}$

Degree multiset: {4,1,1,1,1}

$C_4 \sqcup K_1$

Degree multiset: {2,2,2,2,0}



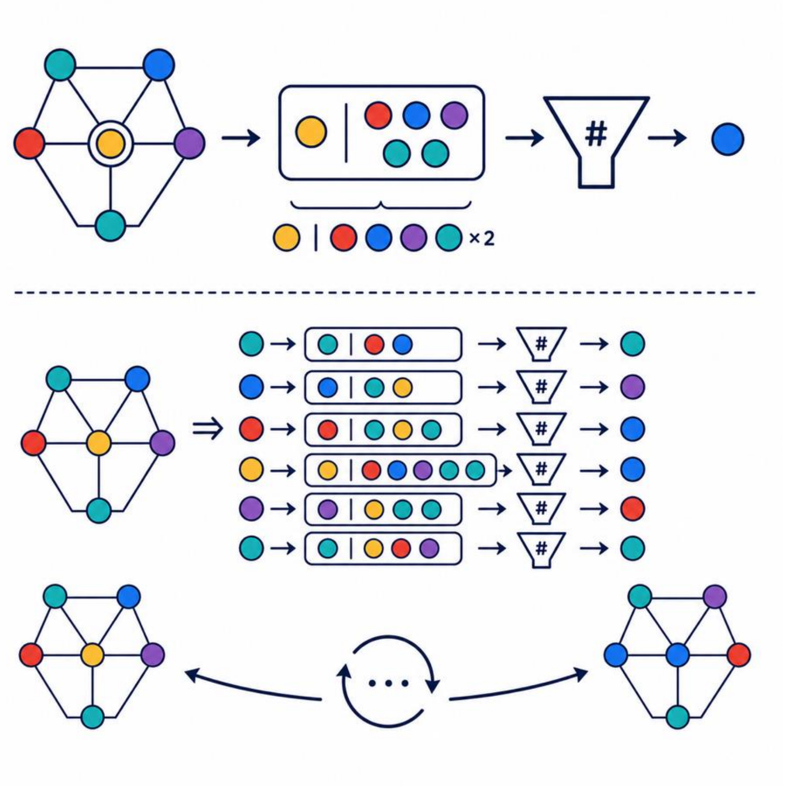
Local counts can separate graphs that a global spectral summary collides.

Local counts collides $G = C_6$ $G' = C_3 \bigcup C_3$

"How many neighbors do I have?" → "What kinds of neighbors do I have?" → "What kinds of neighbors those neighbors have?"

1-WL repeatedly hashes colored neighborhoods

Weisfeiler-Leman graph isomorphism test



INITIALIZE

Use node labels, or one shared initial color.
Update all nodes synchronously.

REFINE

$$c_v^{(t+1)} = \mathrm{HASH}\Big(c_v^{(t)}, \{\{c_u^{(t)} : u \in N(v)\}\}\Big)$$

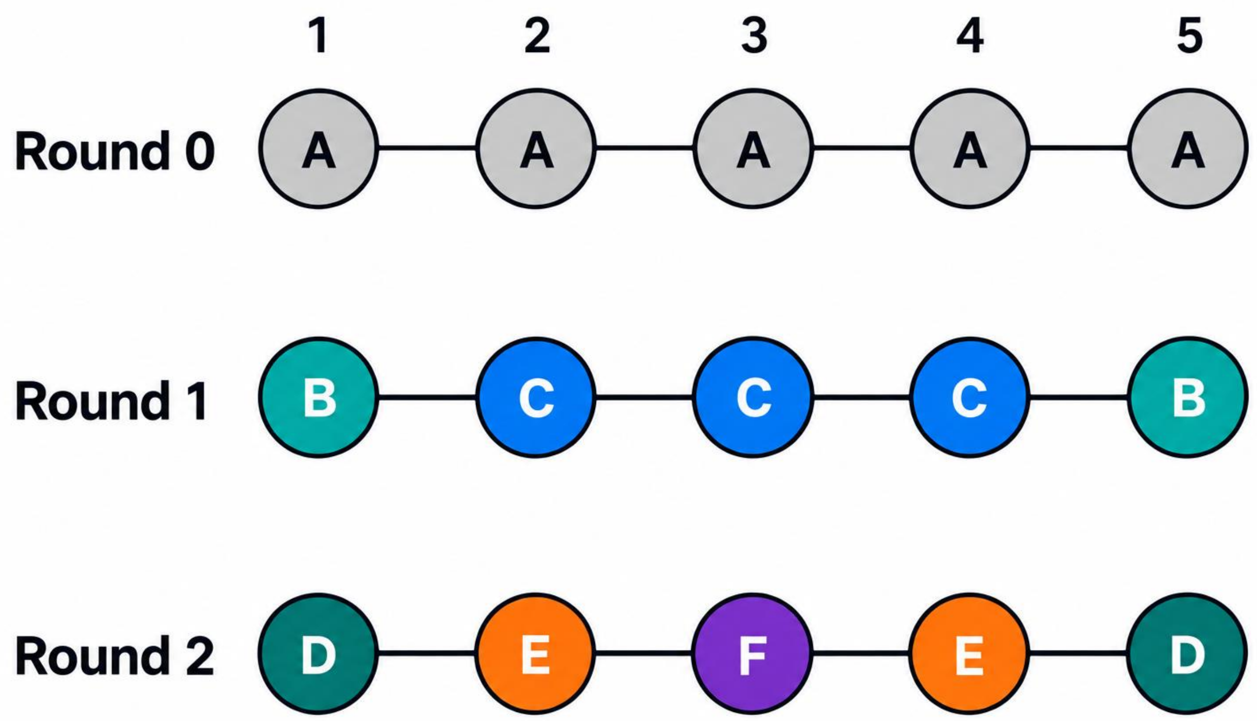
Updates are synchronous

Use the same collision-free color dictionary in both graphs.

Stop when color classes no longer split, stop when a round produces the same partition as the round before.

1-WL on a five-node path

Synchronous updates use the old self color and the old neighbor-color multiset.



ROUND 1: TWO COLOR CLASSES

$$(A, \{A\}) \rightarrow B$$
$$(A, \{A, A\}) \rightarrow C$$

ROUND 2: THREE COLOR CLASSES

$$(B, \{C\}) \rightarrow D$$
$$(C, \{B, C\}) \rightarrow E$$
$$(C, \{C, C\}) \rightarrow F$$

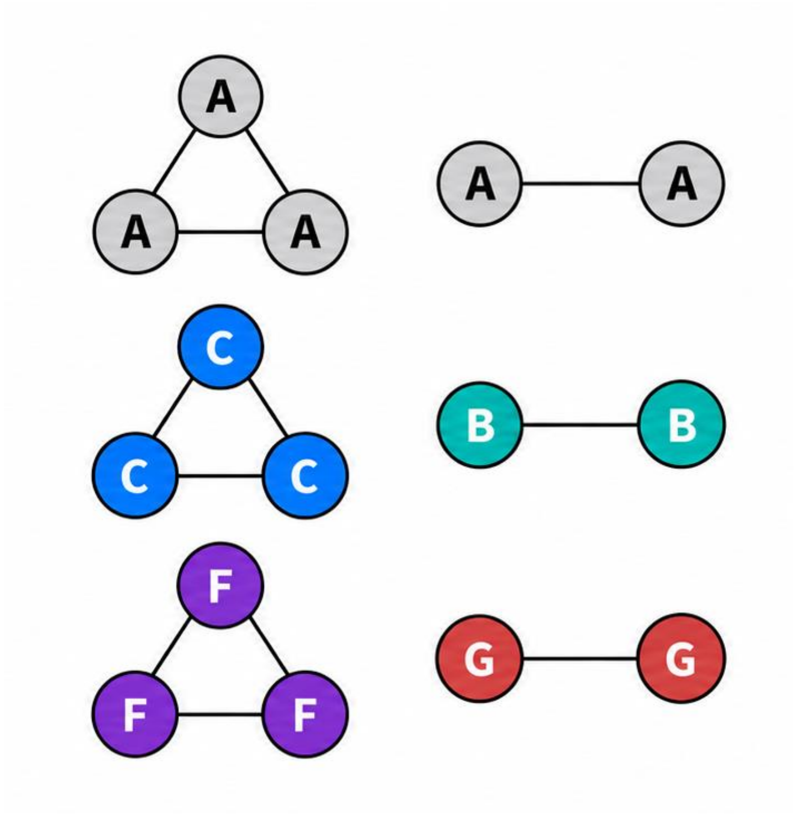
Nodes 2, 3 and 4 stay identical after round 1. The center separates in round 2. Pause at Round 3.

1-WL on $K_3 \cup K_2$

Round 0

Round 1

Round 2



ROUND 1: THREE COLOR CLASSES

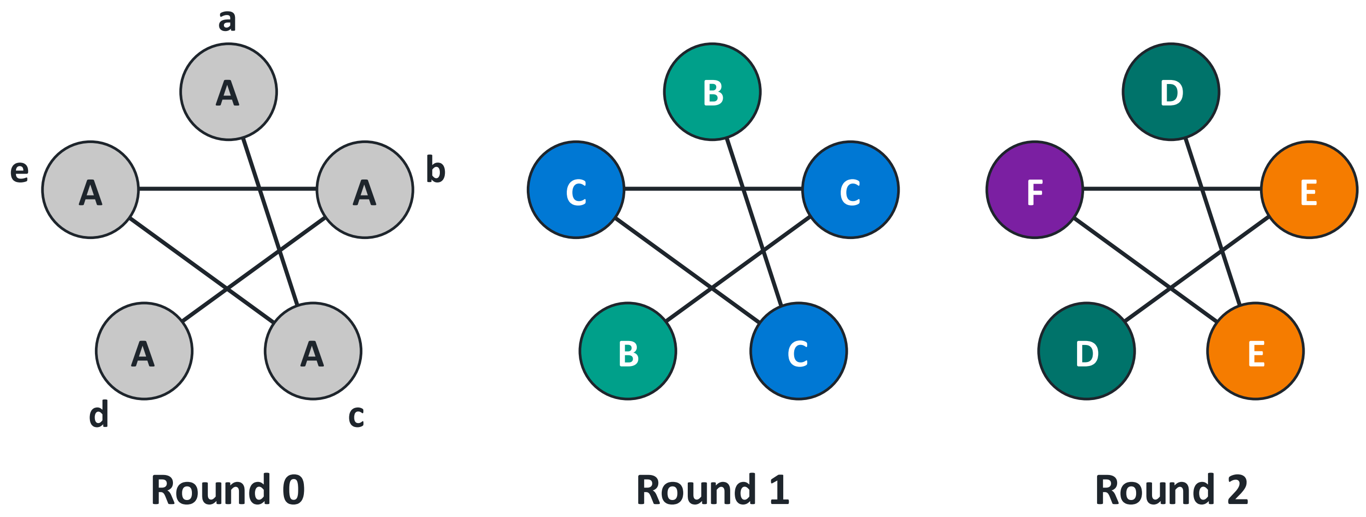
(A, {A}) → B
(A, {A, A}) → C

ROUND 2: FOUR COLOR CLASSES

(B, {B}) → G
$(C, \{C, C\}) \to F$

Color histograms already differ after round 1, so 1-WL declares they are non-isomorphic.

1-WL on a disguised path



ROUND 1: TWO COLOR CLASSES

(A, {A}) → B
(A, {A, A}) → C

ROUND 2: THREE COLOR CLASSES

(B, {C}) → D
(C, {B, C}) → E
(C, {C, C}) → F

Path: {D, D, E, E, F}
Star drawing: {D, D, E, E, F} → same
π: 1→a, 2→c, 3→e, 4→b, 5→d

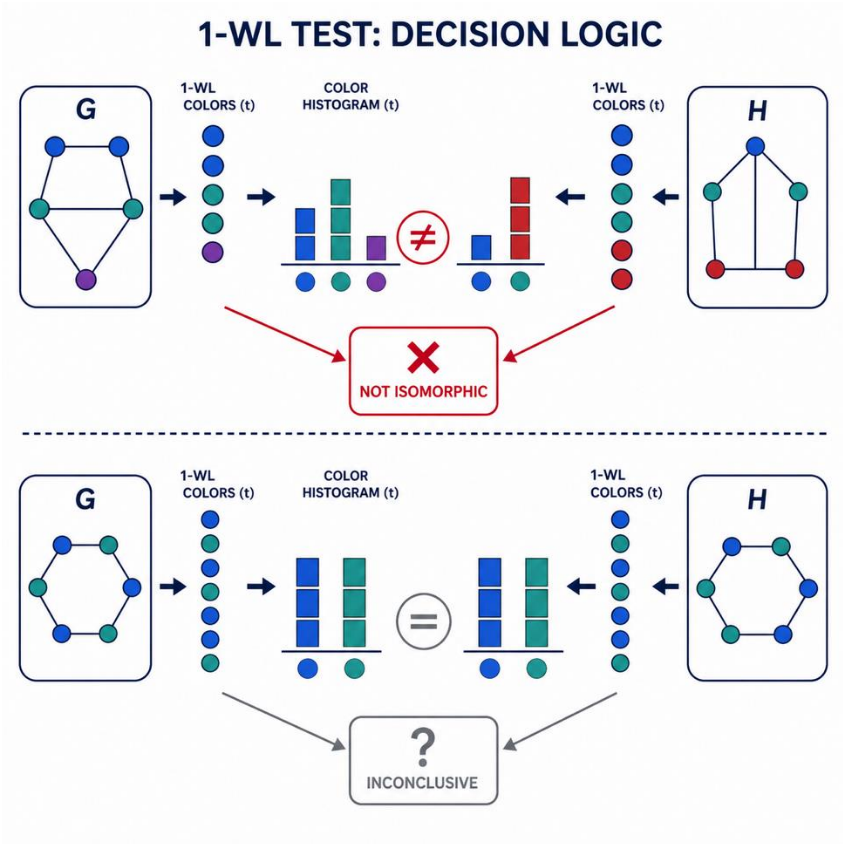
Same histogram every round: 1-WL says "possibly isomorphic", and π confirms they are.

Different color histograms certify non-isomorphism

Compare shared-color histograms **after each round.**

Different histograms ⇒ non-isomorphic

Equal histograms ⇒ not distinguished yet



Matching colors are not a proof of isomorphism; they mean this test did not find a difference.

1-WL fails on some regular graphs

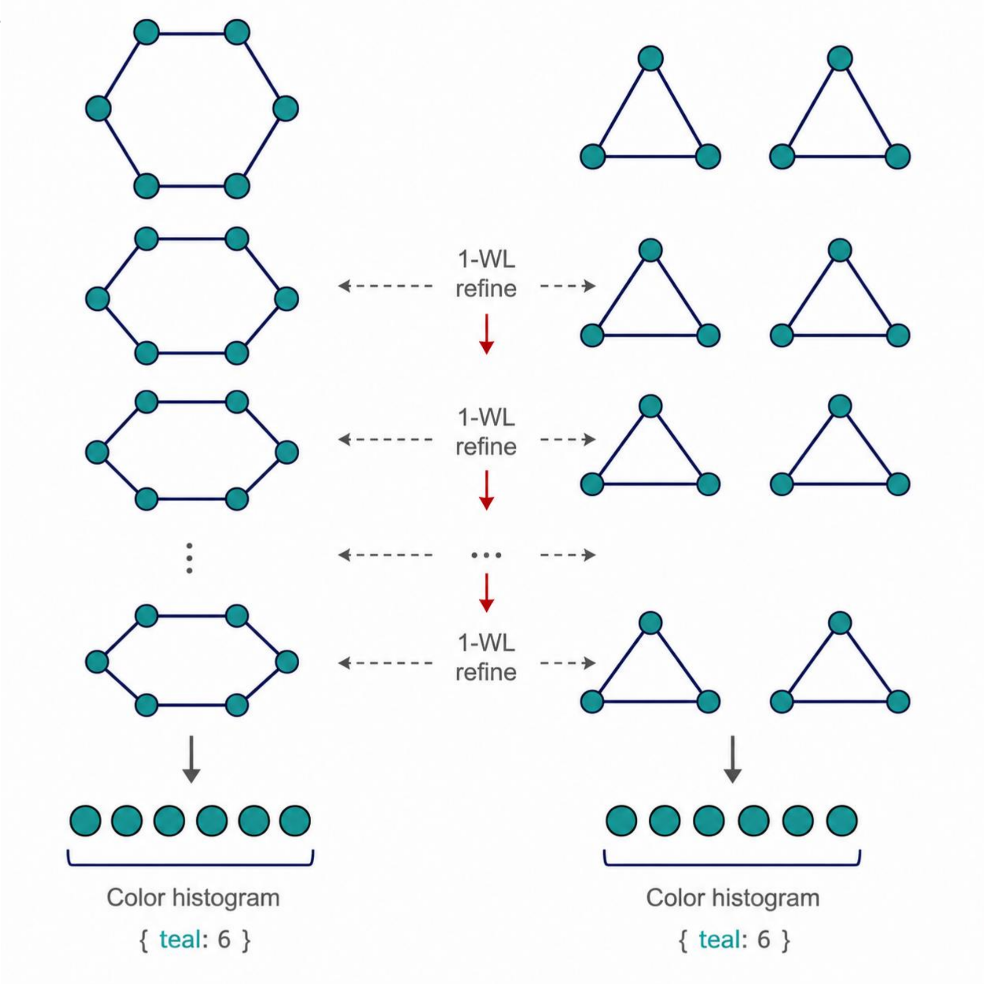
CYCLE C_6

Every node always sees two nodes of the same current color.

TWO TRIANGLES

The same local multiset occurs at every node and every iteration.

A recursively aggregated summary is not the full induced neighborhood.

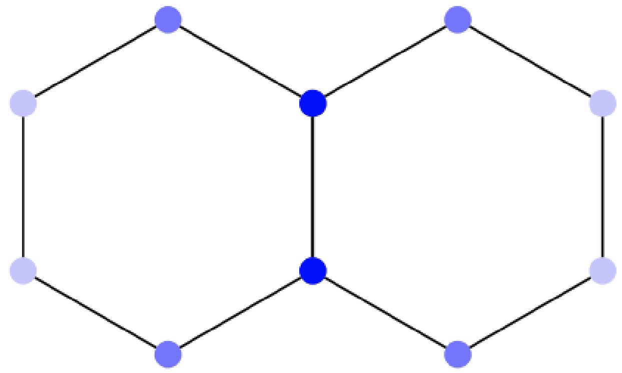


Uniform initial features: six copies of the same color at every round in both graphs.

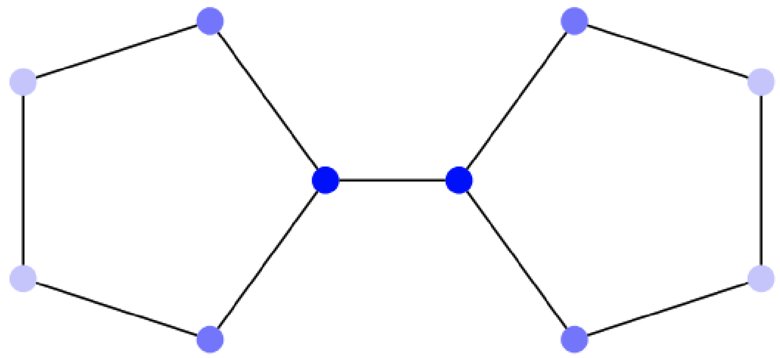
A classical 1-WL counterexample

Uniform initial node features. Colors below show the stable partition from round 2.

Decalin



Bicyclopentyl



Neighbor-color multisets match in both graphs:

Dark D: {D, M, M} **Medium M: {D, L}** **Light L: {M, L}**

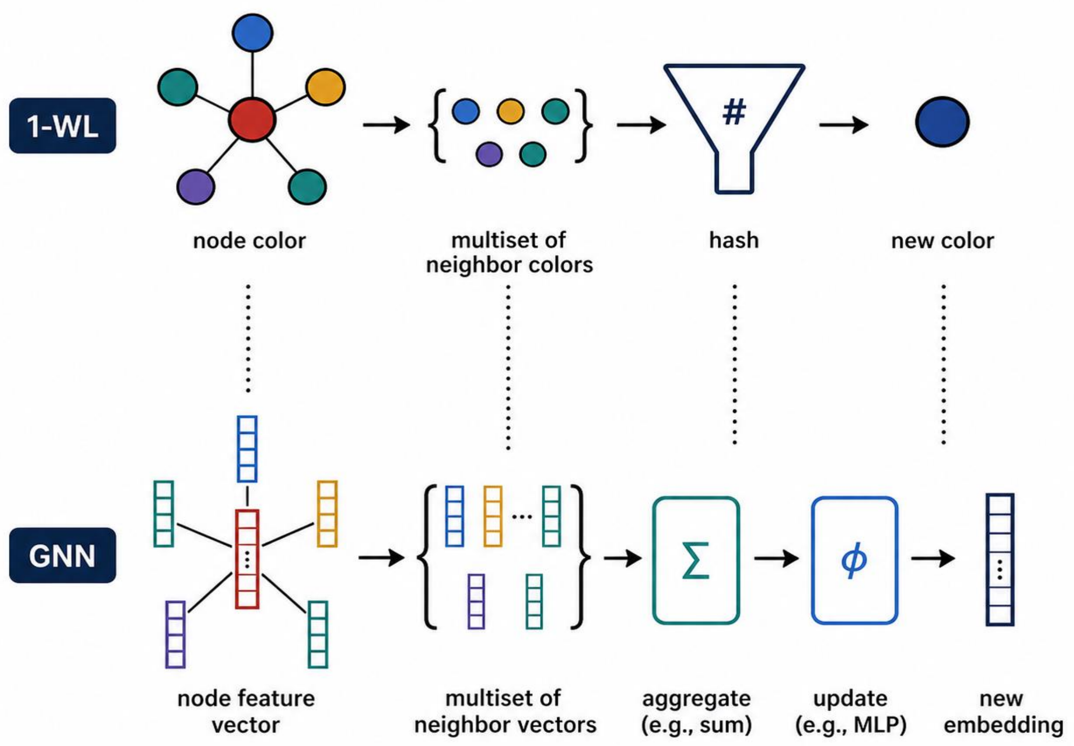
Stable class counts: 2 dark, 4 medium, 4 light. More rounds cannot separate the graphs.

Non-isomorphic: removing the dark–dark edge disconnects only the right graph.

Message passing mirrors color refinement

Both update a node from its current state and a permutation-invariant multiset of neighbor states.

$$m_v^{(k)} = \mathrm{AGG}\Big(\{\{h_u^{(k-1)} : u \in N(v)\}\}\Big)$$
$$h_v^{(k)} = \mathrm{COMBINE}(h_v^{(k-1)}, m_v^{(k)})$$



WL encodes descriptions without collisions, a neural update may lose distinctions.

Message passing also returns the same output for all nodes for C_6 and $K_{3,3}$

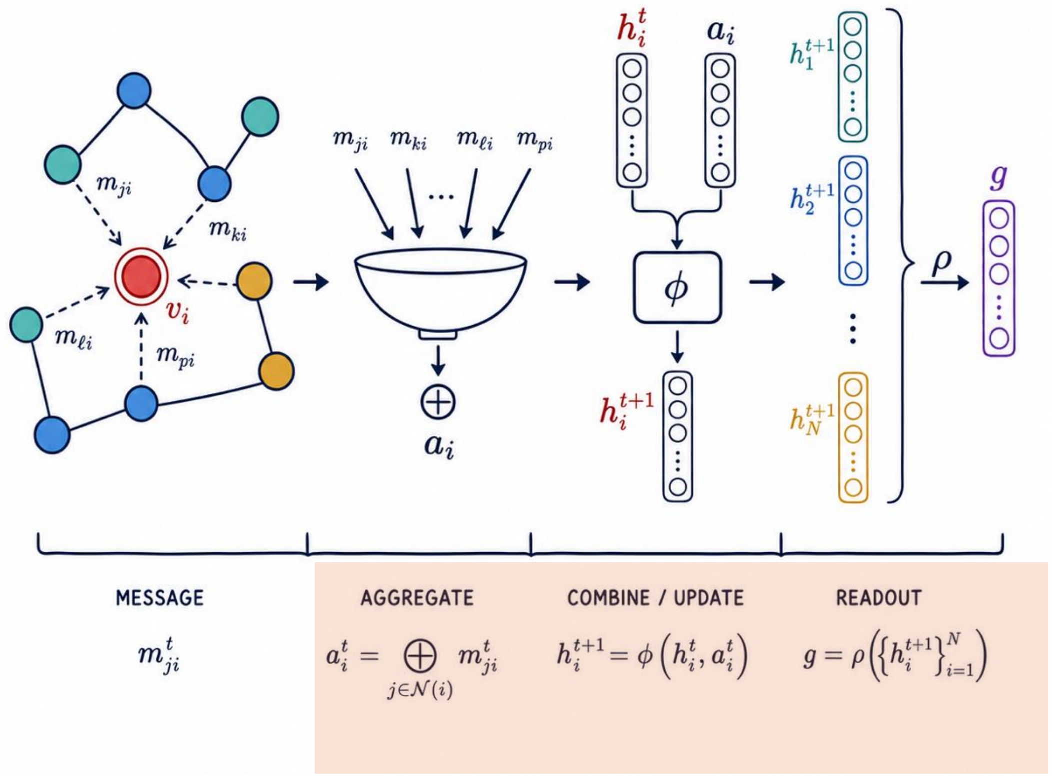
The graph-level task has three functions

NODE UPDATE

$$m_v^{(k)} = \mathrm{AGG}\Big(\{\{h_u^{(k-1)} : u \in N(v)\}\}\Big)$$
$$h_v^{(k)} = \mathrm{COMBINE}(h_v^{(k-1)}, m_v^{(k)})$$

GRAPH READOUT

$$z_G = \mathrm{READOUT}\Big(\{\{h_v^{(L)} : v \in V\}\}\Big), \quad \hat{y} = g(z_G)$$



Expressivity depends on whether AGG, COMBINE, and READOUT are injective.

Standard message passing is bounded by 1-WL

EXPRESSIVITY UPPER BOUND

1-WL indistinguishable
⇒ GNN graph representation cannot distinguish

- Same initial information.
- Shared deterministic node updates.

Invariant aggregation and graph readout.
No extra IDs or global structural encodings. → Node-message-aggregation-combine-readout

GNN tells two graphs apart ⇒ 1-WL tells them apart

Example: C_6 and two triangles

Uniform features
Two identical neighbor states
Same update at every depth

The upper bound concerns standard message passing—not every architecture called a GNN.

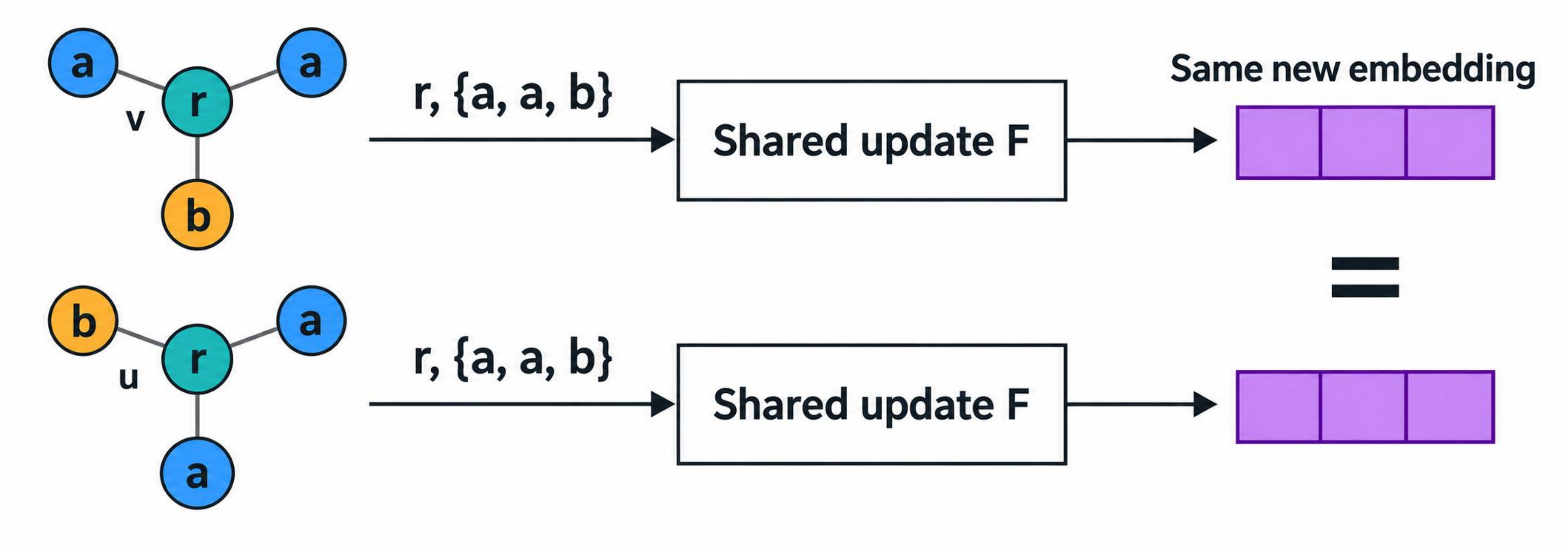
The proof is an induction on refinement depth

INDUCTION HYPOTHESIS

$$c_v^{(k)} = c_u^{(k)} \quad \Longrightarrow \quad h_v^{(k)} = h_u^{(k)}$$

SAME UPDATE

Equal old self-states + equal neighbor-state multisets → equal new states under a shared update.



Base case: matching input features. Inductive step: equal inputs give equal outputs.

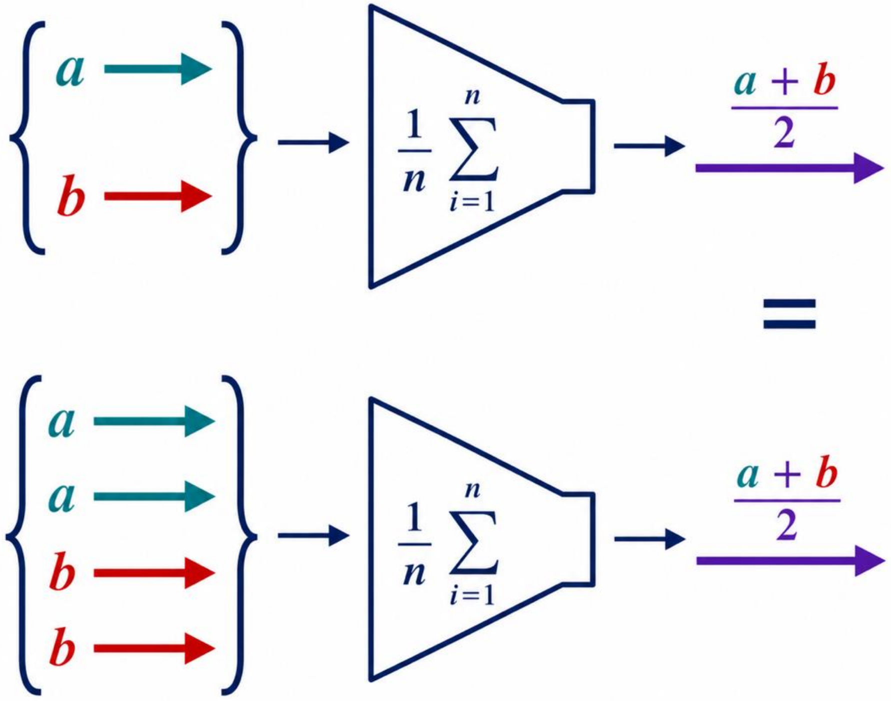
Mean aggregation forgets multiplicity

MULTISET 1

$$\text{mean}\{\{a, b\}\} = \frac{a+b}{2}$$

MULTISET 2

$$\text{mean}\{\{a, a, b, b\}\} = \frac{a+b}{2}$$



Scaling every count by the same factor creates the same mean.

Max aggregation forgets counts

01 **INPUT MULTISETS**

{1,0} and {1,1,0}

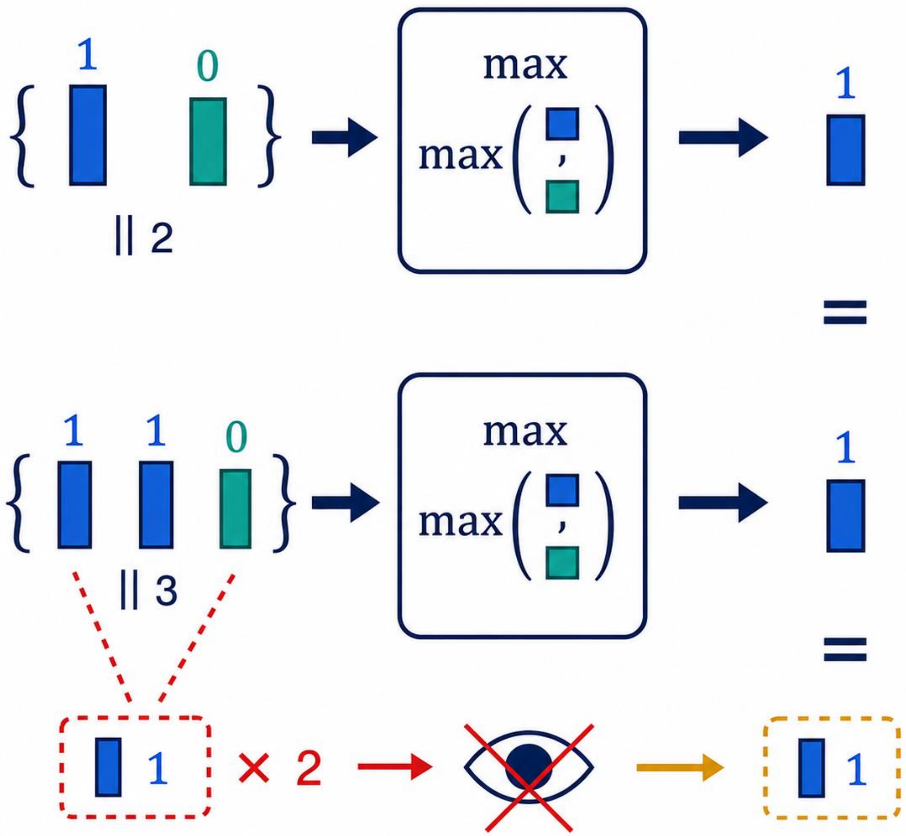
02 **COORDINATEWISE MAX**

Both return 1.

03 **COLLISION**

The repeated maximum is invisible.

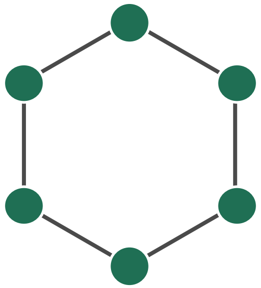
Repeated elements are invisible to max.
Different sets can collide too.

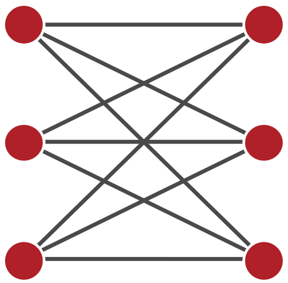


An exact GCN collision on regular graphs

$$S = \widetilde{D}^{-1/2}(A+I)\widetilde{D}^{-1/2}$$

$$\text{For a } d\text{-regular graph:} \qquad S = \frac{A+I}{d+1}, \qquad S\mathbf{1} = \mathbf{1}$$





C_6: six nodes, degree 2

3 identical vectors / 3 = x

$K_{3,3}$: six nodes, degree 3

4 identical vectors / 4 = x

Identical initial features → identical node states at every layer.
Both graphs have six nodes → the same invariant graph readout.

1-WL separates degrees 2 and 3 in round 1.
A standard GCN being strictly weaker than 1-WL under uniform initialization

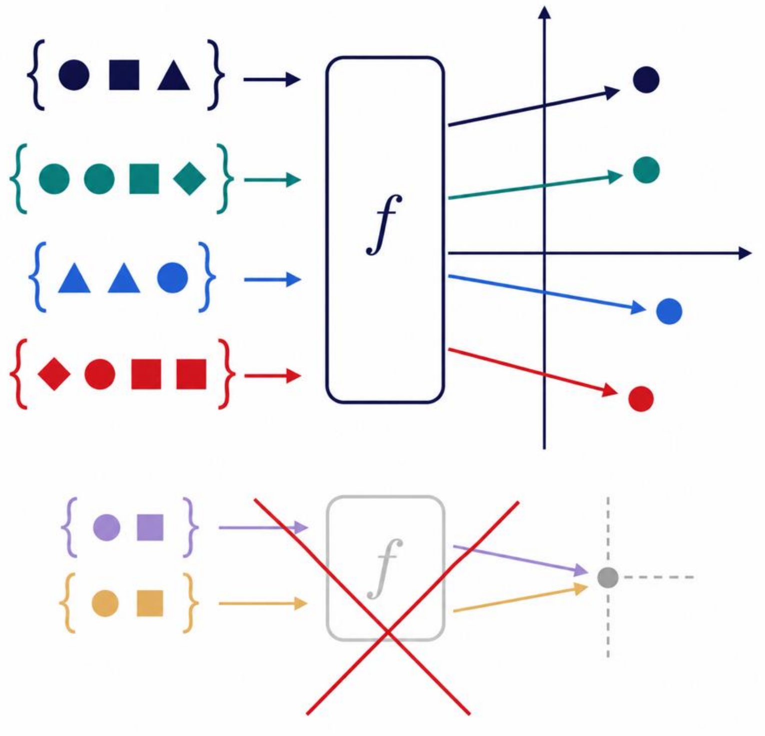
The design target is injective multiset encoding

INJECTIVE AGGREGATION

Different neighbor-feature multisets map to different aggregated vectors.

INJECTIVE READOUT

Different node-embedding multisets map to different graph vectors.



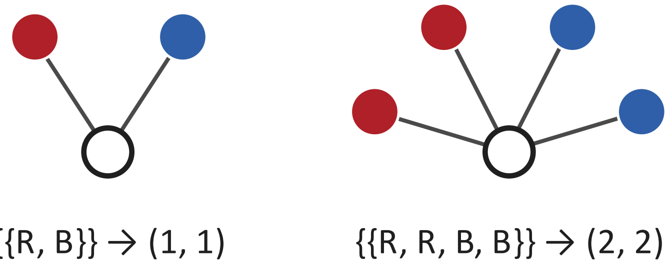
Injectivity is relative to a domain; aggregation, combination, and readout all matter.

A feature map plus sum can preserve multiplicity

$$\phi(R) = (1, 0), \quad \phi(B) = (0, 1)$$

$$\sum_{x \in \{\{R, B\}\}} \phi(x) = (1, 1) \neq (2, 2) = \sum_{x \in \{\{R, R, B, B\}\}} \phi(x)$$

Rooted neighborhoods (root in white)



One-hot categories → sum is a count vector.

Raw sums can collide: {1,3} and {2,2} both sum to 4.
Formal existence result: countable feature universe, bounded multiset size, and a suitable feature map. A trained model is not automatically injective.

Preserving multiplicity requires the right encoding—not just the word “sum.”

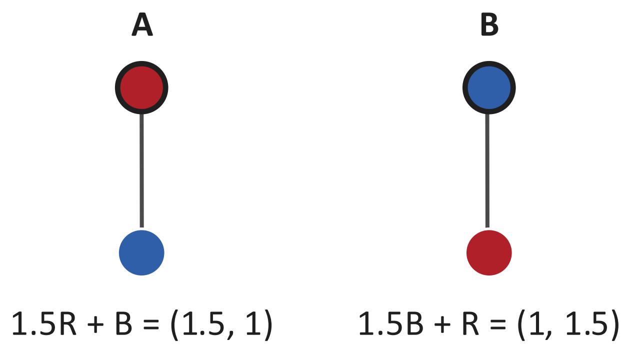
The center needs a distinguishable role

$$(1+\epsilon)h_v + \sum_{u \in N(v)} h_u$$

A: root R, neighbor {B}

B: root B, neighbor {R}

Use R = (1,0) and B = (0,1).



Root has the bold outline

$$\epsilon = 0: \quad R + B = B + R = (1,1)$$
$$\epsilon = 0.5: \quad 1.5R + B = (1.5,1) \neq (1,1.5) = 1.5B + R$$

ε = 0.5 separates this pair. A nonzero ε alone is not a universal guarantee.

The formal construction also depends on the domain and feature map.

GIN uses a sum, a self coefficient, and an MLP

$$h_v^{(k)} = \mathrm{MLP}^{(k)}\left((1+\epsilon^{(k)})h_v^{(k-1)} + \sum_{u\in N(v)} h_u^{(k-1)}\right)$$

Neighbors are summed; the center is added separately.

The feature map f : the previous layer's MLP output $h^{(k-1)}$ plays that role, and one-hot inputs supply it at layer 1.

To match 1-WL, use suitable injective updates, sufficient depth,
and an injective graph readout on the relevant domain.

An MLP cannot undo a collision that already occurred before it.
Training does not guarantee that these ideal conditions are met.

Use ordinary neighbors $N(v)$ here; do not accidentally count the self-loop twice.

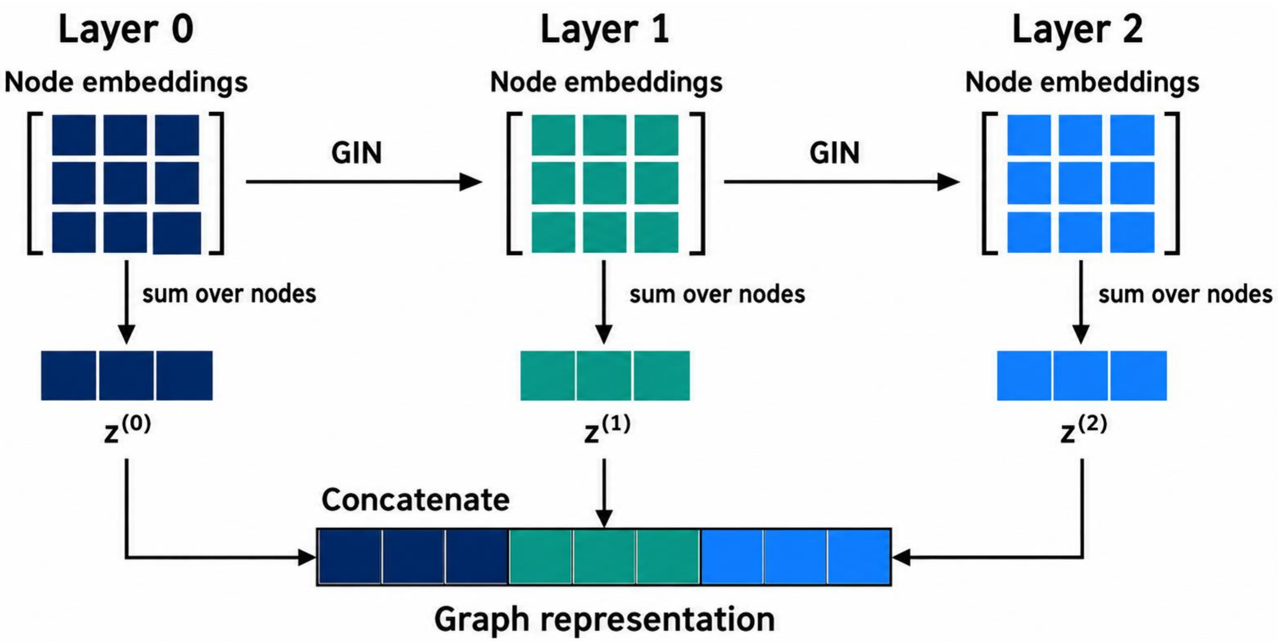
Layerwise readout preserves multiple scales

LAYERWISE SUMS

$$z_G^{(k)} = \sum_{v \in V} h_v^{(k)}$$

MULTI-SCALE VECTOR

$$z_G = \mathrm{CONCAT}\left(\sum_{v \in V} h_v^{(0)}, \ldots, \sum_{v \in V} h_v^{(K)} \right)$$



Concatenation retains earlier summaries; it does not make arbitrary sums injective.

Worked example: mean collides, GIN can separate

Same root r; A has neighbors {t,b}; B has neighbors {t,t,b,b}.

r = (1,0,0) t = (0,1,0) b = (0,0,1) ε = 0

$$\mathrm{mean}(A) = \mathrm{mean}(B) = (0, \tfrac{1}{2}, \tfrac{1}{2})$$

$$r + \mathrm{sum}(A) = (1, 1, 1)$$

$$r + \mathrm{sum}(B) = (1, 2, 2)$$

An injective MLP retains this difference. Arbitrary learned weights may not.

This is a one-layer rooted-neighborhood comparison, not a universal graph-level claim.

GIN can match 1-WL—and inherits its limits

01 **WHAT GIN MATCHES**

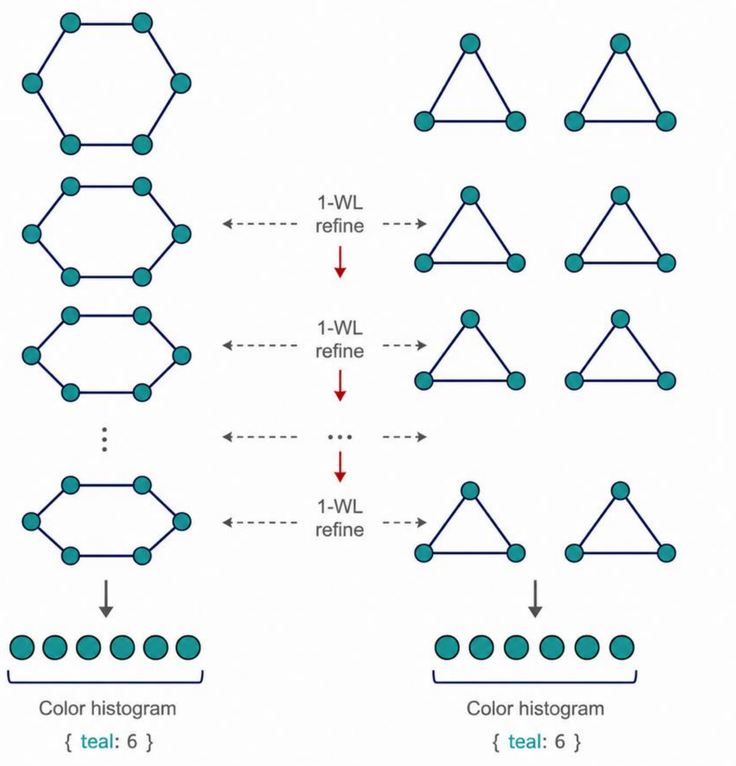
Under suitable injective updates and readout, enough depth, and matched input information.

02 **WHAT GIN MISSES**

C_6 and two triangles remain indistinguishable with uniform features.

03 **WHAT CAN EXTEND IT**

Triangle counts, subgraphs, or higher-order states can add information. Handle positional encodings carefully.



More expressive ≠ automatically better generalization.

GCN vs. GraphSAGE vs. GIN

Criterion	GCN	GraphSAGE mean / max	GIN
Neighborhood map	Symmetric degree-weighted sum	Mean or learned max-pool	Sum + weighted self + MLP
Always injective?	No	No for these aggregators	No; conditions matter
Counts	Can lose degree distinctions	Mean loses replication Max loses repeats	Suitable sum encodings retain counts
Center role	Mixed through self-loop	Usually combined separately	Coefficient $1 + \varepsilon$
Versus 1-WL	Can be strictly weaker	Can be strictly weaker	Can match under assumptions
Graph readout	Pooling can lose information	Pooling can lose information	Injective on relevant states
Key caution	Not always an ordinary mean	Variant and sampling matter	Capacity is not a training guarantee

Comparison assumes the stated variants and the same input information.

Which distinctions can your model keep?

01 Isomorphism demands relabeling invariance

02 1-WL refines neighbor-color multisets

03 Mean and max erase multiset information

04 GIN can reach the 1-WL bound

A practical design check

INPUTS

What structural information enters?

UPDATE

Which neighbor multisets collide?

READOUT

Which graph summaries collide?

Ask what information each aggregation or readout irreversibly loses.