How Powerful are Graph Neural Networks?

Authors: Keyulu Xu, Weihua Hu, Jure Leskovec, Stefanie Jegelka
Presenters: Yu Yang, Yihe Deng, Mingyu Derek Ma

Motivation

- The design of new GNNs is mostly based on empirical intuition, heuristics, and experimental trial-and-error.
- There is little theoretical understanding of the properties and limitations of GNNs, and formal analysis of GNNs' representational capacity is limited.

Contribution

- A theoretical framework for analyzing the representational power of GNNs.
- *How expressive are different GNN variants in learning to represent and distinguish between different graph structures?*
- Inspired by the close connection between GNNs and the Weisfeiler-Lehman (WL) graph isomorphism test (Weisfeiler & Lehman, 1968)
  - A powerful test known to distinguish a broad class of graphs.

Results

1. GNNs are *at most* as powerful as the WL test in distinguishing graph structures.
2. Established conditions on the neighbor aggregation and graph pooling functions under which the resulting GNN is *as powerful as* the WL test.
3. Identified graph structures that cannot be distinguished by popular GNN variants, such as GCN and GraphSAGE and characterized the kinds of graph structures such GNN-based models can capture.
4. Developed a neural architecture, **Graph Isomorphism Network (GIN)** and showed that its discriminative/representational power *is equal to* the power of the WL test.

Preliminaries

$G = (V, E)$: Graph

X_v : Node feature vector for node v

h_v : Node representation vector (for node classification $y_v = f(h_v)$) for node v

h_G : Graph representation vector (for graph classification $y_G = g(h_G)$) for graph G

The *k*-th layer of a GNN is

$$a_v^{(k)} = \text{AGGREGATE}^{(k)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right) \quad h_v^{(0)} = X_v$$

After *k* iterations of aggregation, a node's representation captures the structural information within its *k*-hop network neighborhood.

Preliminaries

$$a_v^{(k)} = \boxed{\text{AGGREGATE}}^{(k)} \left(\left\{ h_u^{(k-1)} : u \in \mathcal{N}(v) \right\} \right)$$

$$h_v^{(k)} = \boxed{\text{COMBINE}}^{(k)} \left(h_v^{(k-1)}, a_v^{(k)} \right) \qquad h_v^{(0)} = X_v$$

- GraphSAGE
  - **AGGREGATE** (element-wise max pooling)

$$a_v^{(k)} = \text{MAX} \left(\left\{ \text{ReLU} \left(W \cdot h_u^{(k-1)} \right), \forall u \in \mathcal{N}(v) \right\} \right)$$

  - **COMBINE** $W \cdot \left[h_v^{(k-1)}, a_v^{(k)} \right]$
- GCN
  - **AGGREGATE** & **COMBINE** (element-wise mean pooling)

$$h_v^{(k)} = \text{ReLU} \left(W \cdot \text{MEAN} \left\{ h_u^{(k-1)}, \forall u \in \mathcal{N}(v) \cup \{v\} \right\} \right)$$

Preliminaries

$$a_v^{(k)} = \text{AGGREGATE}^{(k)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right) \qquad h_v^{(0)} = X_v$$

For graph classification,

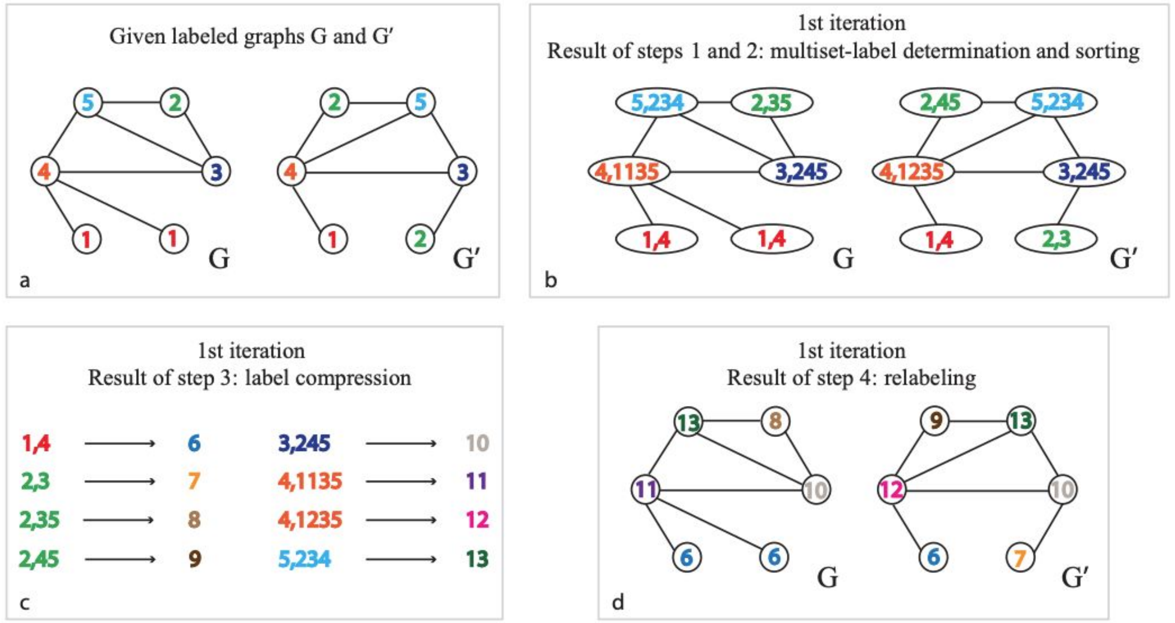
$$h_G = \text{READOUT}\big(\{h_v^{(K)} \mid v \in G\}\big)$$

READOUT can be a simple **permutation invariant** function (such as summation).

Weisfeiler-Lehman Test

- The graph isomorphism problem: Are two graphs topologically identical?
  - NP-Hard
- **The Weisfeiler-Lehman (WL) test of graph isomorphism** (Weisfeiler & Lehman, 1968) is an effective and computationally efficient test.
  - Though it fails on some corner cases.
- Its 1-dimensional form, “naïve vertex refinement”, is analogous to neighbor aggregation in GNNs.

Weisfeiler-Lehman Test


Given labeled graphs G and G′
a
G
G′
1st iteration
Result of steps 1 and 2: multiset-label determination and sorting
b
1st iteration
Result of step 3: label compression
1,4 → 6
2,3 → 7
2,35 → 8
2,45 → 9
3,245 → 10
4,1135 → 11
4,1235 → 12
5,234 → 13
c
1st iteration
Result of step 4: relabeling
d

$$l_v^{(k)} = g\left(l_v^{(k-1)}, \left\{l_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

$$a_v^{(k)} = \text{AGGREGATE}^{(k)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right)$$

An Overview of the Theoretical Framework

- Assume node input features X_v and node feature vectors at deeper layers of any fixed model h_v are from a countable universe.
- Then, feature vectors of a set of neighboring nodes form a **multiset** $X = (S, m)$

  S : the underlying set of *X* that is formed from its distinct elements.

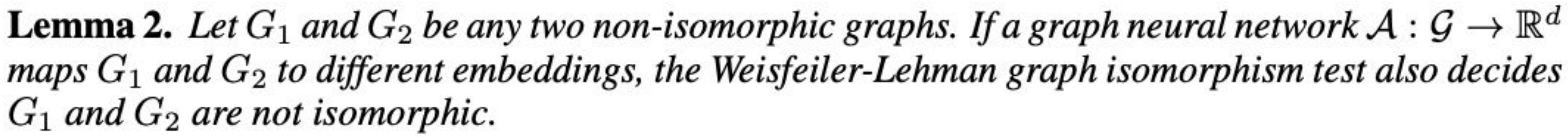
  $m : S \rightarrow \mathbb{N}_{\geq 1}$: the multiplicity of the elements.

- Intuitively, a maximally powerful GNN would never map two different neighborhoods, i.e., multisets of feature vectors, to the same representation.
- => Its aggregation scheme must be injective.

Representational Capacity of General GNN-based Models

Characterizing the representational capacity implies solving the graph isomorphism problem.

We want isomorphic graphs to be mapped to the same representation and non-isomorphic ones to different representations.



Lemma 2. *Let G_1 and G_2 be any two non-isomorphic graphs. If a graph neural network $\mathcal{A} : \mathcal{G} \rightarrow \mathbb{R}^d$ maps G_1 and G_2 to different embeddings, the Weisfeiler-Lehman graph isomorphism test also decides G_1 and G_2 are not isomorphic.*

Proof of Lemma 2

Lemma 2. *Let G_1 and G_2 be any two non-isomorphic graphs. If a graph neural network $\mathcal{A} : \mathcal{G} \to \mathbb{R}^d$ maps G_1 and G_2 to different embeddings, the Weisfeiler-Lehman graph isomorphism test also decides G_1 and G_2 are not isomorphic.*

- **Proof by contradiction.** Suppose after k iterations, $\mathcal{A}(G_1) \neq \mathcal{A}(G_2)$ but the WL test cannot decide G_1 and G_2 are non-isomorphic.
- Then for i=0,...,k, G_1 and G_2 have the same multiset of WL node labels $\left\{l_v^{(i)}\right\}$ and the same multiset of node neighborhoods $\left\{\left(l_v^{(i)}, \left\{l_u^{(i)} : u \in \mathcal{N}(v)\right\}\right)\right\}$.
- By **induction**, we will prove that on the same graph $G = G_1$ **or** G_2 , if WL node labels $l_v^{(i)} = l_u^{(i)}$ always have GNN node features $h_v^{(i)} = h_u^{(i)}$ ny iteration i.

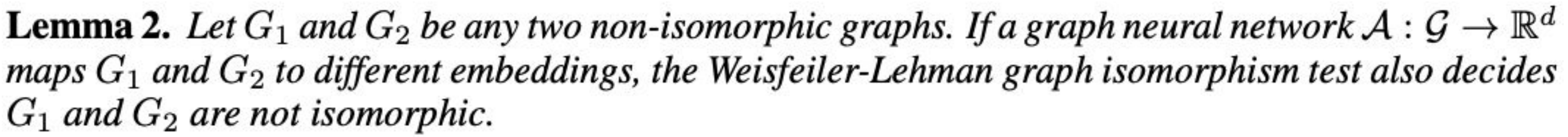
Proof of Lemma 2

- By induction, we prove that on the same graph $G = G_1$ or G_2, if WL node labels $l_v^{(i)} = l_u^{(i)}$, we always have GNN node features $h_v^{(i)} = h_u^{(i)}$ for any iteration i. $\boxed{l_v^{(i)} = l_u^{(i)} \Rightarrow h_v^{(i)} = h_u^{(i)} \text{ (1)}}$
  - (1) apparently holds for i = 0 because WL and GNN starts with the same node features.
  - Suppose (1) holds for iteration j. For any u, v that have $l_v^{(j+1)} = l_u^{(j+1)}$

$$\left(l_v^{(j)}, \left\{l_w^{(j)} : w \in \mathcal{N}(v)\right\}\right) = \left(l_u^{(j)}, \left\{l_w^{(j)} : w \in \mathcal{N}(u)\right\}\right)$$
$$\left(h_v^{(j)}, \left\{h_w^{(j)} : w \in \mathcal{N}(v)\right\}\right) = \left(h_u^{(j)}, \left\{h_w^{(j)} : w \in \mathcal{N}(u)\right\}\right)$$

  - The same input generates the same output. Thus, $h_v^{(j+1)} = h_u^{(j+1)}$

Proof of Lemma 2



Lemma 2. *Let G_1 and G_2 be any two non-isomorphic graphs. If a graph neural network $\mathcal{A} : \mathcal{G} \to \mathbb{R}^d$ maps G_1 and G_2 to different embeddings, the Weisfeiler-Lehman graph isomorphism test also decides G_1 and G_2 are not isomorphic.*

- Proof by contradiction. Suppose after k iterations, $\mathcal{A}(G_1) \neq \mathcal{A}(G_2)$ but the WL test cannot decide G_1 and G_2 are non-isomorphic.
- Then for i=0,...,k, G_1 and G_2 have the same multiset of WL node labels $\{l_v^{(i)}\}$ and the same multiset of node neighborhoods $\left\{\left(l_v^{(i)}, \left\{l_u^{(i)} : u \in \mathcal{N}(v)\right\}\right)\right\}$.

➢ By induction, we proved that on the same graph $G = G_1$ or G_2, if WL node labels $l_v^{(i)} = l_u^{(i)}$, we always have GNN node features $h_v^{(i)} = h_u^{(i)}$ for any iteration i.

- A **valid** mapping ϕ such that $h_v^{(i)} = \phi(l_v^{(i)})$ for any v ∈ G.

Proof of Lemma 2

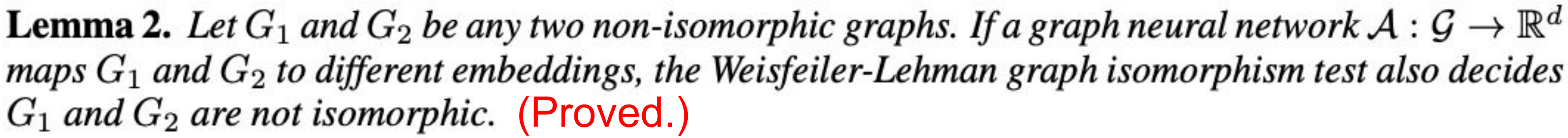
Lemma 2. *Let G_1 and G_2 be any two non-isomorphic graphs. If a graph neural network $\mathcal{A} : \mathcal{G} \to \mathbb{R}^d$ maps G_1 and G_2 to different embeddings, the Weisfeiler-Lehman graph isomorphism test also decides G_1 and G_2 are not isomorphic.*

- Proof by contradiction. Suppose after k iterations, $\boxed{\mathcal{A}(G_1) \neq \mathcal{A}(G_2)}$ but the WL test cannot decide G_1 and G_2 are non-isomorphic.
- Then for i=0,...,k, G_1 and G_2 have the same multiset of WL node labels $\left\{ l_v^{(i)} \right\}$ and the same multiset of node neighborhoods $\left\{ \left(l_v^{(i)}, \left\{ l_u^{(i)} : u \in \mathcal{N}(v) \right\} \right) \right\}$.
- This creates a **valid** mapping φ such that $h_v^{(i)} = \phi(l_v^{(i)})$ for any v ∈ G.

➢ Then because G_1 and G_2 have the same multiset of WL neighborhood labels $\left\{ \left(l_v^{(i)}, \left\{ l_u^{(i)} : u \in \mathcal{N}(v) \right\} \right) \right\}$, they must also have the same collection of GNN neighborhood features. $\boxed{\mathcal{A}(G_1) = \mathcal{A}(G_2).}$ Contradiction. □

Representational Capacity of General GNN-based Models

We want isomorphic graphs to be mapped to the same representation and non-isomorphic ones to different representations.

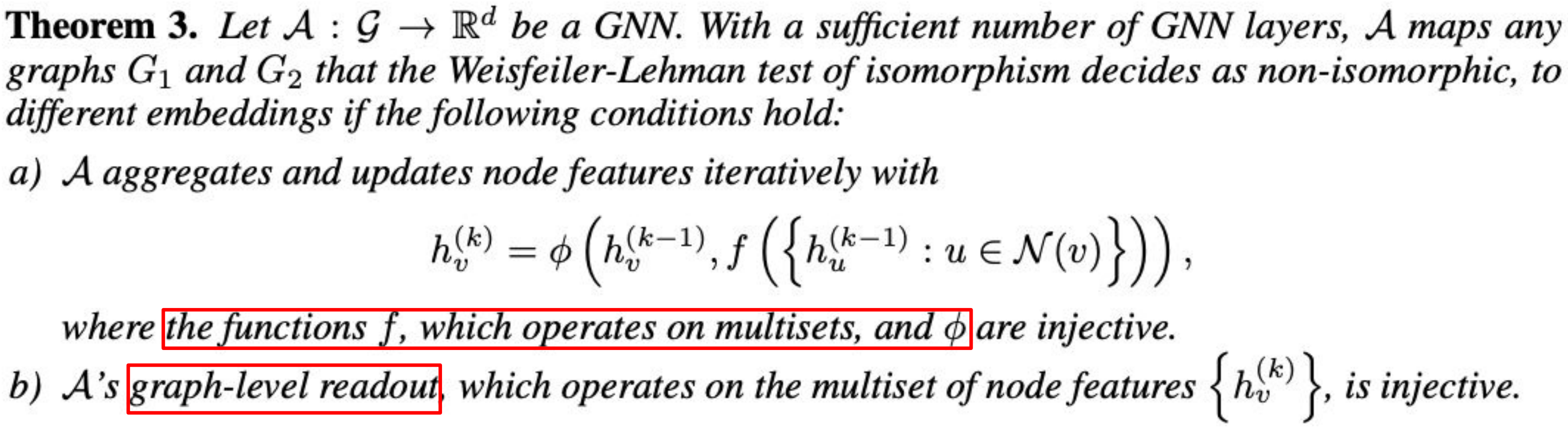


Lemma 2. *Let G_1 and G_2 be any two non-isomorphic graphs. If a graph neural network $\mathcal{A} : \mathcal{G} \rightarrow \mathbb{R}^d$ maps G_1 and G_2 to different embeddings, the Weisfeiler-Lehman graph isomorphism test also decides G_1 and G_2 are not isomorphic.* (Proved.)

Hence, any aggregation-based GNN is **at most** as powerful as the WL test in distinguishing different graphs.

A natural follow-up question is: Do there exist GNNs that are, in principle, **as powerful as** the WL test?

Building Powerful Graph Neural Networks



Theorem 3. *Let $\mathcal{A} : \mathcal{G} \rightarrow \mathbb{R}^d$ be a GNN. With a sufficient number of GNN layers, $\mathcal{A}$ maps any graphs G_1 and G_2 that the Weisfeiler-Lehman test of isomorphism decides as non-isomorphic, to different embeddings if the following conditions hold:*

a) $\mathcal{A}$ aggregates and updates node features iteratively with

$$h_v^{(k)} = \phi\left(h_v^{(k-1)}, f\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)\right),$$

where the functions f, which operates on multisets, and ϕ are injective.

b) $\mathcal{A}$'s graph-level readout, which operates on the multiset of node features $\left\{h_v^{(k)}\right\}$, is injective.

Because the injective graph-level readout function (condition (b)) naturally maps distinct multiset of node features into unique embeddings, it suffices to show that A's neighborhood aggregation process, with sufficient iterations, embeds G1 and G2 into different multisets of node features.

Proof of Theorem 3

GNN updates:

$$h_v^{(k)} = \phi\left(h_v^{(k-1)}, f\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)\right)$$

WL updates:

$$l_v^{(k)} = g\left(l_v^{(k-1)}, \left\{l_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

f and ϕ are **injective** functions. g is a predetermined **injective** hash function.

By induction, we will show that for any iteration k, there always exists an **injective** function φ such that $h_v^{(k)} = \varphi\left(l_v^{(k)}\right)$. (2)

Again, this apparently holds for k = 0 because the initial node features are the same for WL and GNN.

Suppose this holds for iteration k − 1, we show that it also holds for k.

Proof of Theorem 3

$$h_v^{(k)} = \varphi\left(l_v^{(k)}\right) \quad (2)$$

GNN updates:

$$h_v^{(k)} = \phi\left(h_v^{(k-1)}, f\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)\right)$$

WL updates:

$$l_v^{(k)} = g\left(l_v^{(k-1)}, \left\{l_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

Substituting $h_v^{(k-1)}$ with $\varphi\left(l_v^{(k-1)}\right)$ gives us

$$h_v^{(k)} = \phi\left(\varphi\left(l_v^{(k-1)}\right), f\left(\left\{\varphi\left(l_u^{(k-1)}\right) : u \in \mathcal{N}(v)\right\}\right)\right)$$

Since the composition of injective functions is injective, there exists some injective function ψ so that

$$h_v^{(k)} = \psi\left(l_v^{(k-1)}, \left\{l_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

$$h_v^{(k)} = \psi \circ g^{-1} g\left(l_v^{(k-1)}, \left\{l_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right) = \psi \circ g^{-1}\left(l_v^{(k)}\right)$$

$\varphi = \psi \circ g^{-1}$ is injective because the composition of injective functions is injective.

Proof of Theorem 3

GNN updates:

$$h_v^{(k)} = \phi\left(h_v^{(k-1)}, f\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)\right)$$

WL updates:

$$l_v^{(k)} = g\left(l_v^{(k-1)}, \left\{l_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

f and φ are **injective** functions. g is a predetermined **injective** hash function.

- By induction, we showed that for any iteration k, there always exists an injective function φ such that $h_v^{(k)} = \varphi\left(l_v^{(k)}\right)$.

At the K-th iteration, the WL test decides that G1 and G2 are non-isomorphic, that is the multisets $\left\{l_v^{(K)}\right\}$ are different for G1 and G2. Then,

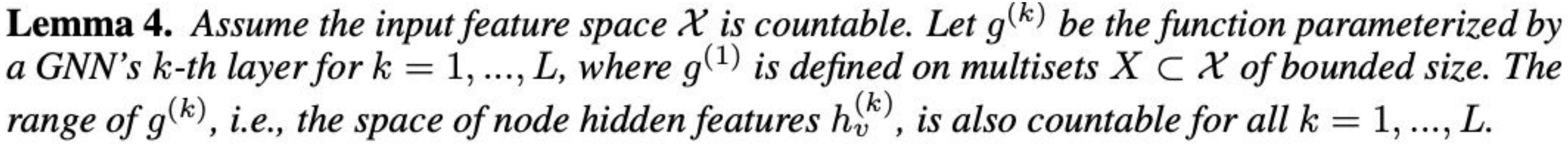
$\left\{h_v^{(K)}\right\} = \left\{\varphi\left(l_v^{(K)}\right)\right\}$ must also be different for G1 and G2 because of the injectivity of φ. □

Building Powerful Graph Neural Networks

Note: We assumed that node input features and node feature vectors at deeper layers of any fixed model are from a **countable** universe.

For countable sets, injectiveness well characterizes whether a function preserves the distinctness of inputs.

Uncountable sets, where node features are continuous, need some further considerations.



Lemma 4. *Assume the input feature space $\mathcal{X}$ is countable. Let $g^{(k)}$ be the function parameterized by a GNN's k-th layer for $k = 1, ..., L$, where $g^{(1)}$ is defined on multisets $X \subset \mathcal{X}$ of bounded size. The range of $g^{(k)}$, i.e., the space of node hidden features $h_v^{(k)}$, is also countable for all $k = 1, ..., L$.*

Building Powerful Graph Neural Networks

- Graph Isomorphism Network (GIN)
  - Having developed conditions for maximally powerful GNN, the authors propose a simple architecture that satisfies the condition in **Theorem 3 (condition a)**:

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

a) $\mathcal{A}$ aggregates and updates node features iteratively with

$$h_v^{(k)} = \phi\left(h_v^{(k-1)}, f\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)\right),$$

where the functions f, which operates on multisets, and ϕ are injective.

- The authors want prove that the chosen multiset functions are injective for **the neighbor aggregation** (f) and for the functions **over a node and the multiset of its neighbors** (Φ).
- There may exist many other maximally powerful GNNs, where GIN is one such example.

Graph Isomorphism Network (GIN)

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

$$f(u) = h_u^{(k-1)}$$

- Proof of GIN satisfying the condition (a) in Theorem 3:
  - **Step 1: f is injective, i.e. the sum aggregator is injective**
  - Step 2: ϕ is injective, i.e. the combine function is injective

Lemma 5. *Assume $\mathcal{X}$ is countable. There exists a function $f : \mathcal{X} \rightarrow \mathbb{R}^n$ so that* $\boxed{h(X) = \sum_{x \in X} f(x)}$ *is unique for each multiset $X \subset \mathcal{X}$ of bounded size. Moreover, any multiset function g can be decomposed as $g(X) = \phi\left(\sum_{x \in X} f(x)\right)$ for some function ϕ.*

Lemma 5. (Part 1) Given a countable domain $\mathcal{X}$, there always exists a function $f : \mathcal{X} \rightarrow \mathbb{R}^n$ such that

- for each unique input multiset $X \subset \mathcal{X}$ of **bounded** size,
- the summation output $h(X) = \sum_{x \in X} f(x)$ is unique.

Proof of Lemma 5 (part 1)

- There exists a mapping f so that for each unique multiset X of bounded size, a $\sum_{x \in X} f(x)$.
  a. Because domain $\mathcal{X}$ is countable, we could map each node feature vector x to a unique natural number. Let the mapping $Z : \mathcal{X} \to \mathbb{N}$
  b. Because the size of all multiset X is bounded, there exist a natural number N such that, for any multiset X, $|X| < N$
  c. An example of such mapping f would be $f(x) = N^{-Z(x)}$
     - Maps each finite multiset uniquely to a rational scalar with N-digit-expansion representation
     - Another example would be one-hot encoding.

Proof of Lemma 5 (part 1)

A concrete example of $f(x) = N^{-Z(x)}$

- Take N = 10 for example, then by definition |X| < 10 for all multisets $X \subset \mathcal{X}$
- Have Z(X) = 1, 2, …, n, … for all $x \in \mathcal{X}$ countable
- For a multiset $Z(X_1) = \{1, 1, 1, 3, 4, 4\}$, $\sum_{x \in X} f(x)$ = 0.301200…
- The mapping would always result in a unique sum for a unique multiset

Lemma 5 (part 2)

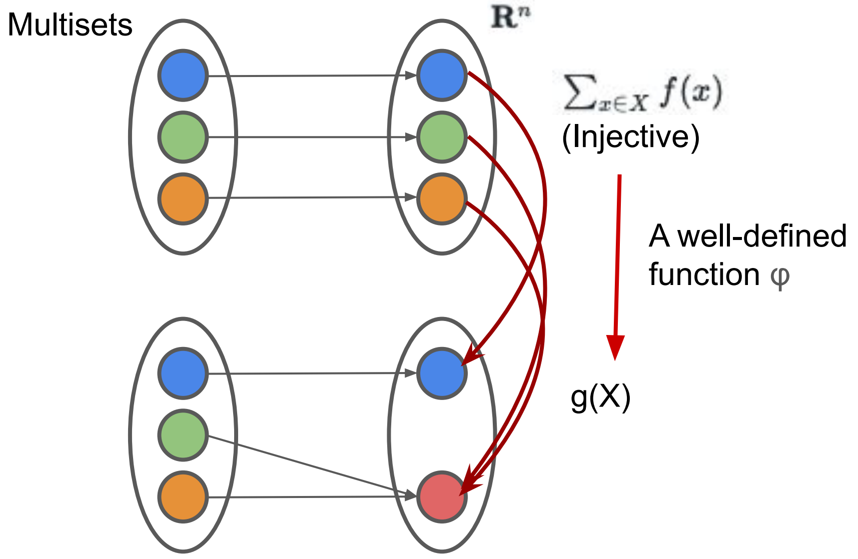
Lemma 5. (Part 2) Moreover, any multiset function g can be decomposed as $g(X) = \phi\left(\sum_{x \in X} f(x)\right)$ for some function ϕ.

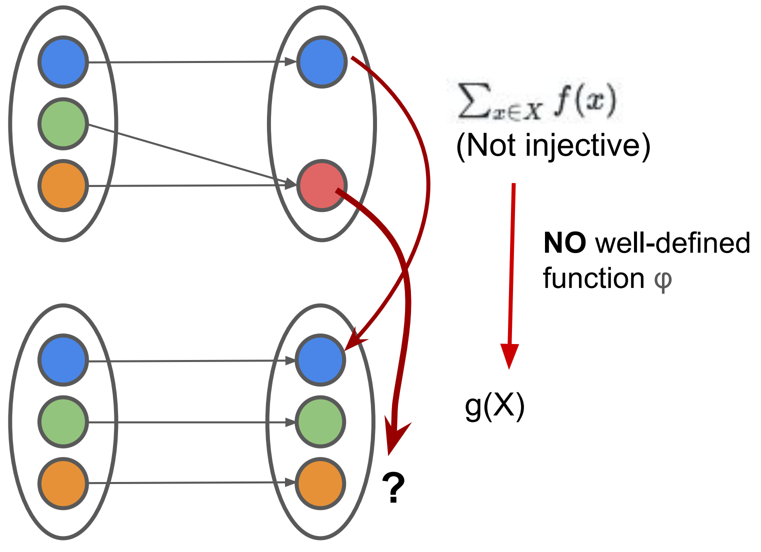
- Implication: we can model any multiset functions with the sum aggregator.
  - i.e. the sum aggregator is one of the most expressive multiset functions.
- Rephrasing the claim
  - For any multiset function $g : \mathcal{X} \to \mathbf{R}^n$ that could exist, there exists a function $\phi : \mathbf{R}^n \to \mathbf{R}^n$ such that for all input X, we have $g(X) = \phi(\sum_{x \in X} f(x))$
- Proof
  - Because $\sum_{x \in X} f(x)$ is injective, we would always have a such well-defined function φ.

Lemma 5 (part 2) $g(X) = \phi(\sum_{x \in X} f(x))$

Illustration of the proof:

- Because $\sum_{x \in X} f(x)$ is injective, we would always have a well-defined function φ.

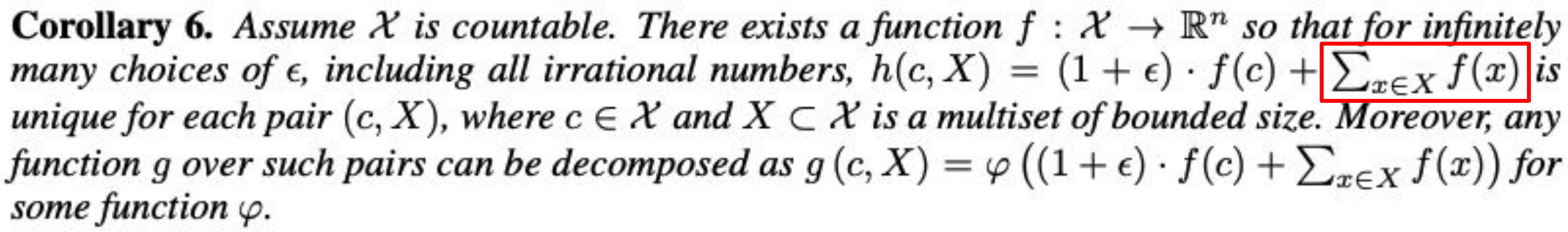




$f : A \to B$ is well-defined if for each $a \in A$ there is a unique $b \in B$ with $f(a) = b$.

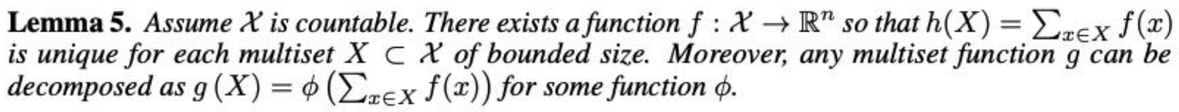
Graph Isomorphism Network (GIN)

- Proof of GIN satisfying the condition (a) in Theorem 3:
  - Step 1: f is injective, i.e. the sum aggregator is injective
  - **Step 2: ϕ is injective, i.e. the combine function is injective**



Corollary 6. *Assume $\mathcal{X}$ is countable. There exists a function $f : \mathcal{X} \to \mathbb{R}^n$ so that for infinitely many choices of ϵ, including all irrational numbers, $h(c, X) = (1 + \epsilon) \cdot f(c) + \boxed{\sum_{x \in X} f(x)}$ is unique for each pair (c, X), where $c \in \mathcal{X}$ and $X \subset \mathcal{X}$ is a multiset of bounded size. Moreover, any function g over such pairs can be decomposed as $g(c, X) = \varphi\left((1 + \epsilon) \cdot f(c) + \sum_{x \in X} f(x)\right)$ for some function φ.*

- Consider $f(x) = N^{-Z(x)}$ following Lemma 5. Want to show that for any input $(c', X') \neq (c, X)$ the result $h(c, X) \neq h(c', X')$ holds, with ϵ as an irrational number.



Lemma 5. *Assume $\mathcal{X}$ is countable. There exists a function $f : \mathcal{X} \to \mathbb{R}^n$ so that $h(X) = \sum_{x \in X} f(x)$ is unique for each multiset $X \subset \mathcal{X}$ of bounded size. Moreover, any multiset function g can be decomposed as $g(X) = \phi\left(\sum_{x \in X} f(x)\right)$ for some function ϕ.*

Proof of Corollary 6

$h(c, X) = (1 + \epsilon) \cdot f(c) + \sum_{x \in X} f(x)$

- Proof by contradiction.
  - Given an input pair (c, X), assume there exists $(c', X') \neq (c, X)$ but $h(c, X) = h(c', X')$ holds. Consider two cases
    - $c' = c$ but $X' \neq X$
    - $c' \neq c.$
  - First case proved by Lemma 5.
  - Second case: rewrite the equation $h(c, X) = h(c', X')$ as
    - $$\epsilon \cdot (f(c) - f(c')) = \left(f(c') + \sum_{x \in X'} f(x)\right) - \left(f(c) + \sum_{x \in X} f(x)\right).$$

Graph Isomorphism Network (GIN)

- Proven:
  - There exists function f such that $h(c, X) = (1+\epsilon)\cdot f(c) + \sum_{x\in X} f(x)$ is injective.
  - There exists function φ such that $g(c, X) = \varphi\left((1+\epsilon)\cdot f(c) + \sum_{x\in X} f(x)\right)$ for any g.
- (Universal Approximation Theorem) Using MLPs to model and learn f and φ.
- In practice, we can use one MLP to model $f^{(k+1)} \circ \varphi^{(k)}$
- Let ε be a learnable parameter or a fixed scalar. GIN then update node representations as

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

Graph-level Readout of GIN

- Node embeddings learned by GIN can be directly used for node classification or link prediction.
- For graph classification, need a readout function to produce embedding for the entire graph.

$$h_G = \text{CONCAT}\Big(\text{READOUT}\Big(\Big\{h_v^{(k)} | v \in G\Big\}\Big) \mid k = 0, 1, \ldots, K\Big).$$

  - CONCAT: to consider all structural information, we use information from all depths/iterations of the model.
  - READOUT: by Theorem 3 and Corollary 6, GIN can replace READOUT with summing all node features from the same iterations.

Ablation Studies: Less Powerful But Still Interesting GNNs

GNNs that do not satisfy the conditions in Theorem 3:

1. 1-Layer perceptrons are not sufficient (instead of MLPs)
2. Structures that confuse mean and max-pooling (instead of sum)
    a. Mean learns distributions
    b. Max-pooling learns sets with distinct elements
3. Remarks on other aggregators for future works
    a. Attention
    b. LSTM pooling

1-Layer perceptrons are not sufficient

Lemma 7. *There exist finite multisets $X_1 \neq X_2$ so that for any linear mapping W,* $\sum_{x \in X_1} \mathrm{ReLU}(Wx) = \sum_{x \in X_2} \mathrm{ReLU}(Wx)$.

- Consider two different multisets $X_1 = \{1, 1, 1, 1, 1\}$ and $X_2 = \{2, 3\}$ that sum up to the same value.

$$\sum_{x \in X} \mathrm{ReLU}(Wx) = \mathrm{ReLU}\left(W \sum_{x \in X} x\right) \longrightarrow \sum_{x \in X_1} \mathrm{ReLU}(Wx) = \sum_{x \in X_2} \mathrm{ReLU}(Wx)$$

- Main idea: 1-layer perceptrons can behave much like linear mappings, so the GNN layers degenerate into simply summing over neighborhood features.
  - With the bias term and sufficiently large output dimensionality, 1-layer perceptrons might be able to distinguish different multisets.
  - Not a universal approximator of multiset functions.

Structures that confuse mean and max-pooling

- What happens if we replace the sum in $h(X) = \sum_{x \in X} f(x)$ with mean or max-pooling as in GCN and GraphSAGE?
  - They are not injective aggregators for **multisets**.

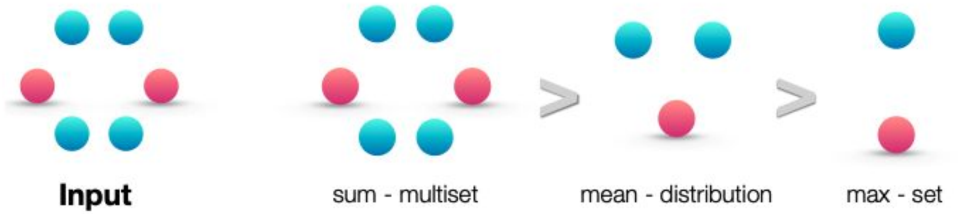
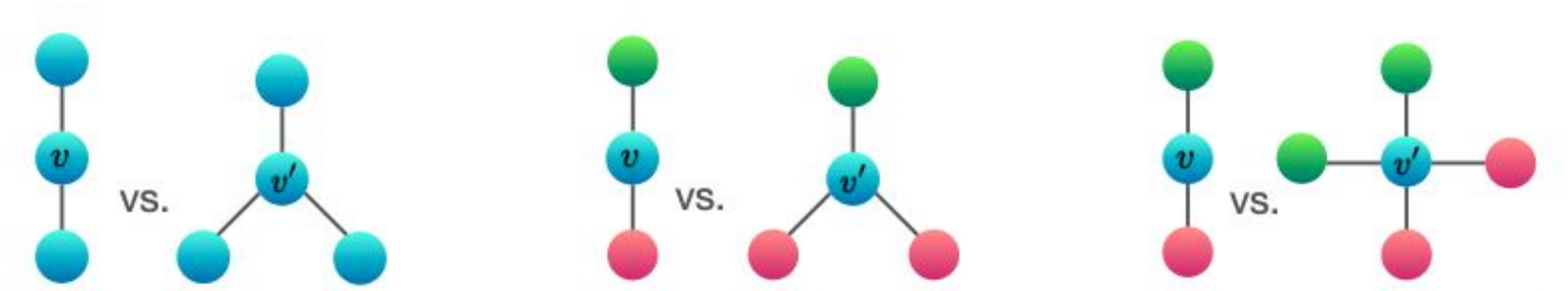


Figure 2: **Ranking by expressive power for sum, mean and max aggregators over a multiset**. Left panel shows the input multiset, *i.e.*, the network neighborhood to be aggregated. The next three panels illustrate the aspects of the multiset a given aggregator is able to capture: sum captures the full multiset, mean captures the proportion/distribution of elements of a given type, and the max aggregator ignores multiplicities (reduces the multiset to a simple set).

Structures that confuse mean and max-pooling



(a) Mean and Max both fail (b) Max fails (c) Mean and Max both fail

Figure 3: **Examples of graph structures that mean and max aggregators fail to distinguish.** Between the two graphs, nodes v and v' get the same embedding even though their corresponding graph structures differ. Figure 2 gives reasoning about how different aggregators "compress" different multisets and thus fail to distinguish them.

Mean and max-pooling

1. Mean learns distributions.

Corollary 8. *Assume $\mathcal{X}$ is countable. There exists a function $f : \mathcal{X} \rightarrow \mathbb{R}^n$ so that for $h(X) = \frac{1}{|X|}\sum_{x \in X} f(x)$, $h(X_1) = h(X_2)$ if and only if multisets X_1 and X_2 have the same distribution. That is, assuming $|X_2| \geq |X_1|$, we have $X_1 = (S, m)$ and $X_2 = (S, k \cdot m)$ for some $k \in \mathbb{N}_{\geq 1}$.*

$$\frac{1}{|X_2|}\sum_{x \in X_2} f(x) = \frac{1}{k \cdot |X_1|} \cdot k \cdot \sum_{x \in X_1} f(x) = \frac{1}{|X_1|}\sum_{x \in X_1} f(x)$$

2. Max-pooling learns sets with distinct elements.

Corollary 9. *Assume $\mathcal{X}$ is countable. Then there exists a function $f : \mathcal{X} \rightarrow \mathbb{R}^\infty$ so that for $h(X) = \max_{x \in X} f(x)$, $h(X_1) = h(X_2)$ if and only if X_1 and X_2 have the same underlying set.*

$$\max_{x \in X_1} f(x) = \max_{x \in S} f(x) = \max_{x \in X_2} f(x)$$

Experiments

Task and Datasets

Task: graph classification

Datasets

- 4 bioinformatics datasets: MUTAG, PTC, NCI1, PROTEINS
- 5 social network datasets

Name	Graph	Node	Edge	Classification Target
IMDB-BINARY/MULTI	Ego network for actor/actress	Actor/actress	They appear in the same movie	Genre
REDDIT-BINARY/MULTI5K	Online discussion thread	Users	Respond to comment	Subreddit
COLLAB	Ego network of researchers	Researchers	Collaboration	Subfield of researchers

Training Set Performance

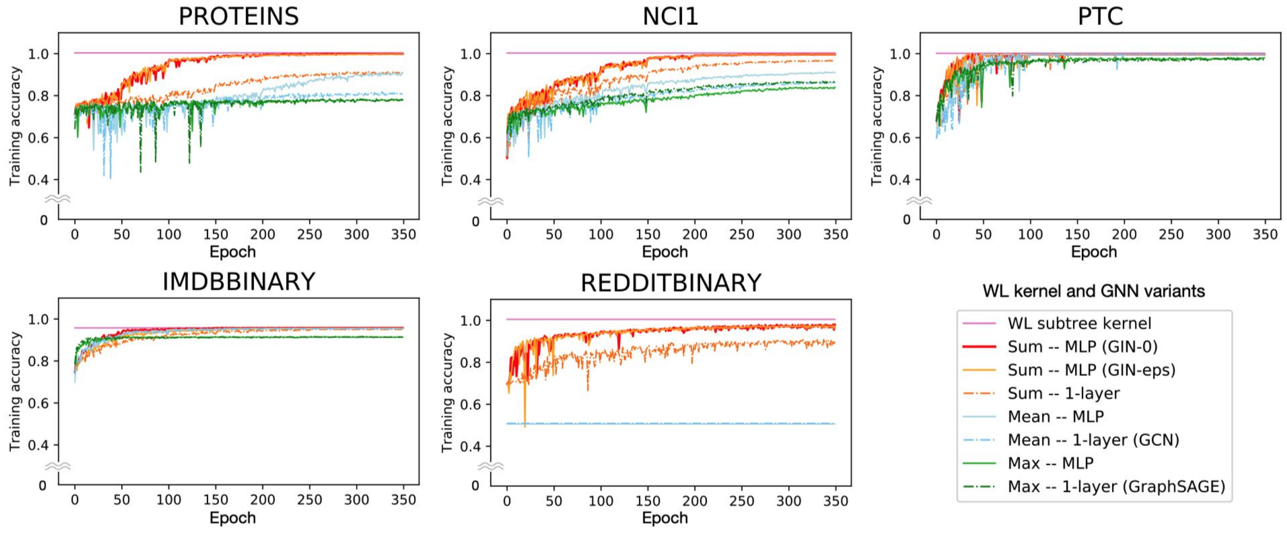


Figure 4: Training set performance of GINs, less powerful GNN variants, and the WL subtree kernel.

- Higher representational power -> higher training set accuracy
- GINs are able to almost perfectly fit all the training sets
- Explicit learning of ε yields no gain compared to fixing ε to 0

Training Set Performance

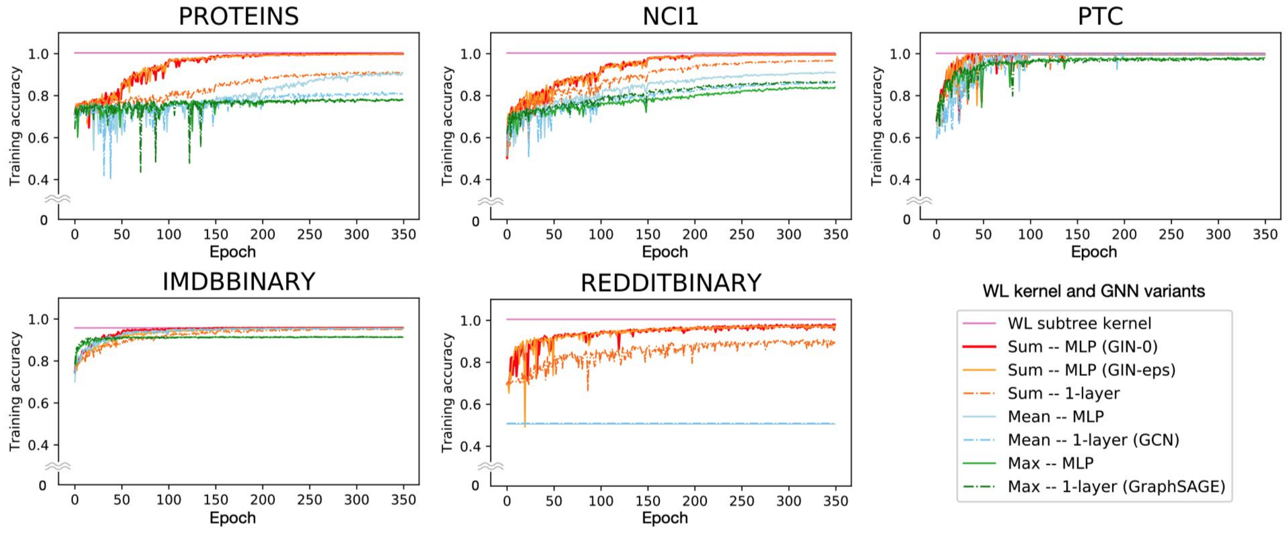


Figure 4: Training set performance of GINs, less powerful GNN variants, and the WL subtree kernel.

- GNNs with MLPs > GNNs with 1 layer perceptrons
- GNNs with sum aggregators > GNNs with mean/max pooling aggregators

Training Set Performance

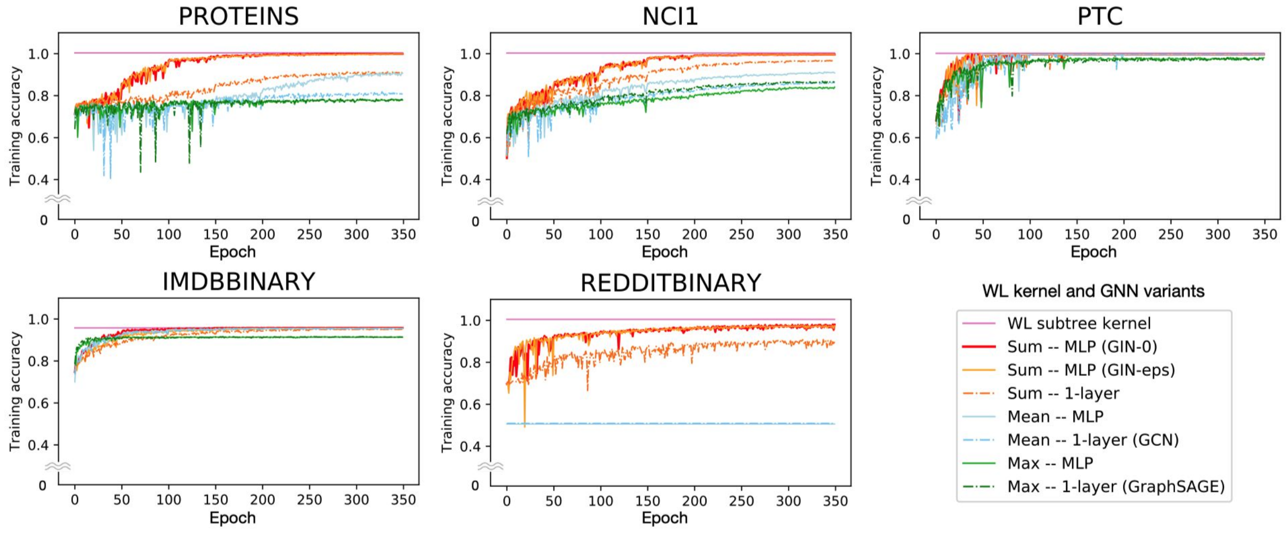
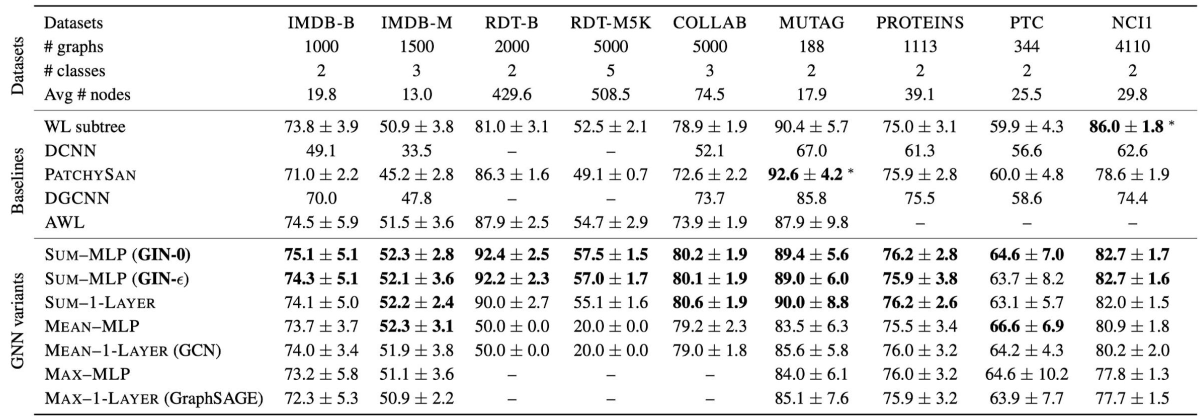


Figure 4: Training set performance of GINs, less powerful GNN variants, and the WL subtree kernel.

- WL subtree kernel is the upper bound of any methods, especially in IMDBBINARY

Test Set Accuracies: Discriminative Power -> Generalization ?



	Datasets	IMDB-B	IMDB-M	RDT-B	RDT-M5K	COLLAB	MUTAG	PROTEINS	PTC	NCI1
Datasets	# graphs	1000	1500	2000	5000	5000	188	1113	344	4110
	# classes	2	3	2	5	3	2	2	2	2
	Avg # nodes	19.8	13.0	429.6	508.5	74.5	17.9	39.1	25.5	29.8
Baselines	WL subtree	73.8 ± 3.9	50.9 ± 3.8	81.0 ± 3.1	52.5 ± 2.1	78.9 ± 1.9	90.4 ± 5.7	75.0 ± 3.1	59.9 ± 4.3	**86.0 ± 1.8** *
	DCNN	49.1	33.5	–	–	52.1	67.0	61.3	56.6	62.6
	PatchySan	71.0 ± 2.2	45.2 ± 2.8	86.3 ± 1.6	49.1 ± 0.7	72.6 ± 2.2	**92.6 ± 4.2** *	75.9 ± 2.8	60.0 ± 4.8	78.6 ± 1.9
	DGCNN	70.0	47.8	–	–	73.7	85.8	75.5	58.6	74.4
	AWL	74.5 ± 5.9	51.5 ± 3.6	87.9 ± 2.5	54.7 ± 2.9	73.9 ± 1.9	87.9 ± 9.8	–	–	–
GNN variants	Sum–MLP (**GIN-0**)	**75.1 ± 5.1**	**52.3 ± 2.8**	**92.4 ± 2.5**	**57.5 ± 1.5**	**80.2 ± 1.9**	**89.4 ± 5.6**	**76.2 ± 2.8**	**64.6 ± 7.0**	**82.7 ± 1.7**
	Sum–MLP (**GIN-ϵ**)	**74.3 ± 5.1**	**52.1 ± 3.6**	**92.2 ± 2.3**	**57.0 ± 1.7**	**80.1 ± 1.9**	**89.0 ± 6.0**	**75.9 ± 3.8**	63.7 ± 8.2	**82.7 ± 1.6**
	Sum–1-Layer	74.1 ± 5.0	**52.2 ± 2.4**	90.0 ± 2.7	55.1 ± 1.6	**80.6 ± 1.9**	**90.0 ± 8.8**	**76.2 ± 2.6**	63.1 ± 5.7	82.0 ± 1.5
	Mean–MLP	73.7 ± 3.7	**52.3 ± 3.1**	50.0 ± 0.0	20.0 ± 0.0	79.2 ± 2.3	83.5 ± 6.3	75.5 ± 3.4	**66.6 ± 6.9**	80.9 ± 1.8
	Mean–1-Layer (GCN)	74.0 ± 3.4	51.9 ± 3.8	50.0 ± 0.0	20.0 ± 0.0	79.0 ± 1.8	85.6 ± 5.8	76.0 ± 3.2	64.2 ± 4.3	80.2 ± 2.0
	Max–MLP	73.2 ± 5.8	51.1 ± 3.6	–	–	–	84.0 ± 6.1	76.0 ± 3.2	64.6 ± 10.2	77.8 ± 1.3
	Max–1-Layer (GraphSAGE)	72.3 ± 5.3	50.9 ± 2.2	–	–	–	85.1 ± 7.6	75.9 ± 3.2	63.9 ± 7.7	77.7 ± 1.5

Table 1: **Test set classification accuracies (%).** The best-performing GNNs are highlighted with boldface. On datasets where GINs' accuracy is not strictly the highest among GNN variants, we see that GINs are still comparable to the best GNN because a paired t-test at significance level 10% does not distinguish GINs from the best; thus, GINs are also highlighted with boldface. If a baseline performs significantly better than all GNNs, we highlight it with boldface and asterisk.

- Strong expressive power -> generalize
- GIN-0 is the best on all 9 datasets
- GIN-0 > GIN-eps
  - GIN-0 is more simple yet fit training data well
- Mean-based GNNs perform much worse than sum-based GNNs

What about Node Classification?

Node features are rich and diverse

- Existing GNNs model can already fit training data well
- Inductive bias of GNNs also plays a key role in performance
- Statistical and distributional information of neighborhood features provide strong signals

Pivot from the main goal of the paper: investigate expressive power of GNNs to capture graph **structures**

Demo

Reproduction Observation

- Can reproduce the majority of the result
- If taking testing set performance when training set achieve best accuracy/loss, the number is worse than claimed (MUTAG and RDT-M5K)
- Large variance, especially for smaller datasets (MUTAG, PTC, IMDB-B)

Pros and Cons

Pros

- Developed theoretical foundations for reasoning about the expressive power of GNNs
- Designed provably maximally powerful GNN under neighborhood aggregation framework

Cons

- Beyond neighborhood aggregation framework?
- Discriminative power is not the only thing about representation power, most applications are not doing graph isomorphism tests

Not sure

- The main assumption is the input node features are from a countable universe. But do real-world graph classification applications' input node features come from a countable universe?