Some Might Say All You Need Is Sum

Eran Rosenbluth[*] , **Jan Toenshoff**[†] and **Martin Grohe**
RWTH Aachen University
{rosenbluth, toenshoff, grohe}@informatik.rwth-aachen.de

Abstract

The expressivity of Graph Neural Networks (GNNs) is dependent on the aggregation functions they employ. Theoretical works have pointed towards Sum aggregation GNNs subsuming every other GNNs, while certain practical works have observed a clear advantage to using Mean and Max. An examination of the theoretical guarantee identifies two caveats. First, it is size-restricted, that is, the power of every specific GNN is limited to graphs of a specific size. Successfully processing larger graphs may require an other GNN, and so on. Second, it concerns the power to distinguish non-isomorphic graphs, not the power to approximate general functions on graphs, and the former does not necessarily imply the latter.

It is desired that a GNN's usability will not be limited to graphs of any specific size. Therefore, we explore the realm of unrestricted-size expressivity. We prove that basic functions, which can be computed exactly by Mean or Max GNNs, are inapproximable by any Sum GNN. We prove that under certain restrictions, every Mean or Max GNN can be approximated by a Sum GNN, but even there, a combination of (Sum, [Mean/Max]) is more expressive than Sum alone. Lastly, we prove further expressivity limitations for GNNs with a broad class of aggregations.

1 Introduction

Message passing graph neural networks (GNNs) are a fundamental deep learning architecture for machine learning on graphs. Most state-of-the-art machine learning techniques for graphs are based on GNNs. It is therefore worthwhile to understand their theoretical properties. Expressivity is one important aspect: which functions on graphs or their vertices can be computed by GNN models? To start with, functions computed by GNNs are always isomorphism invariant, or equivariant for node-level functions. A second important feature of GNNs is that a GNN can operate on input graphs of every size, since it is defined as a series of node-level computations with an optional graph-aggregating readout computation. These are desirable features that motivated the introduction of GNNs in the first place and may be seen as a crucial factor for their success. Research on the expressivity of GNNs has had a considerable impact in the field.

A GNN computation transforms a graph with an initial *feature map* (a.k.a. *graph signal* or *node embedding*) into a new feature map. The new map can represent a node-level function or can be "read out" as a function of the whole graph. The computation is carried out by a finite sequence of separate *layers*. On each layer, each node sends a real-valued message vector which depends on its current feature vector, to all its neighbours. Then each node aggregates the messages it receives, using an order-invariant multiset function, typically being entrywise summation (Sum), mean (Mean), or maximum (Max). Finally, the node features are updated using a neural network which receives as arguments the aggregation value and the node's current feature. In the eyes of a GNN all vertices are euqal: the message, aggregation and update functions of every layer are identical for every node, making GNNs auto-scalable and isomorphism-invariant.

By now, numerous works have researched the expressivity of GNNs considering various variants of them. However, many of the theoretical results have the following caveats:
1. The expressivity considered is *non-uniform*: for a function that is defined on graphs of all sizes, it is asked if for every n there exists a GNN that expresses the function on graphs of size n. The expressing GNN may depend on n, and it may even be exponentially large in n. For some proofs, this exponential blow-up is necessary [Abboud *et al.*, 2021; Xu *et al.*, 2019]. This notion of expressivity is in contrast to *uniform* expressivity: for a function that is defined on graphs of all sizes, asking whether there exists one GNN that expresses the function on graphs of all sizes. In addition to being a significantly weaker theoretical notion, non-uniform expressivity leaves much to be desired also from a practical standpoint: It implies that a GNN may be no good for graphs of sizes larger than the sizes well-represented in the training data. This means that training may have to be done on very

[*]Funded by the German Research Council (DFG), RTG 2236 (UnRAVeL)
[†]Funded by the German Research Council (DFG), grants GR 1492/16-1; KI 2348/1-1 "Quantitative Reasoning About Database Queries"

large graphs, and may have to be often repeated.
2. The expressivity considered is the power to distinguish non-isomorphic graphs. A key theoretical result is the characterisation of the power of GNNs in terms of the Weisfeiler-Leman (WL) isomorphism test [Morris *et al.*, 2019; Xu *et al.*, 2019], and subsequent works have used WL as a yardstick (see 'Related Work'). In applications of GNNs though, the goal is not to distinguish graphs but to regress or classify them or their nodes. There seem to be a hidden assumption that higher distinguishing power implies better ability to express general functions. While this is indeed the case in some settings [Chen *et al.*, 2019], it is not the case with uniform expressivity notion.

Our goal is to better understand the role that the aggregation function plays in the expressivity of GNNs. Specifically, we ask: Do Sum aggregation GNNs subsume Mean and Max GNNs, in terms of uniform expressivity of general functions? A common perception is that an answer is already found in [Xu *et al.*, 2019]: Sum-GNNs strictly subsume all other aggregations GNNs. Examining the details though, what is actually proven there is: in the non-uniform notion, considering a finite input domain, the distinguishing power of Sum-GNNs subsume the distinguishing power of all other aggregations GNNs. Furthermore, in practice it has been observed that for certain tasks there is a clear advantage to using Mean and Max aggregations [Cappart *et al.*, 2021; Hamilton *et al.*, 2017; Tönshoff *et al.*, 2022], with one of the most common models in practice using a variation of Mean aggregation [Kipf and Welling, 2017]. While the difference between theoretical belief and practical evidence may be attributed to learnability rather than expressivity, it calls for better theoretical understanding of expressivity.

1.1 Our Contribution

All our results are in the uniform expressivity notion. Mainly, we prove that Sum-GNNs do not subsume Mean-GNNs nor Max-GNNs (and vice versa), in terms of vertices-embedding expressivity as well as graph-embedding expressivity. The statements in this paper consider additive approximation, yet the no-subsumption ones hold true also for multiplicative approximation.

- Advantage Sum. For the sake of completeness, in Section 3 we prove that even with single-value input features, the neighbors-sum function which can be trivially exactly computed by a Sum-GNN cannot be approximated by any Mean-GNN or Max-GNN.
- Sum subsumes. In Section 4 we prove that if the input features are bounded, Sum-GNNs can approximate all Mean-GNNs or Max-GNNs, though not without an increase in size which depends polynomially on the required accuracy, and exponentially on the depth of the approximated Mean-GNNs or Max-GNNs.
- Advantage Mean and Max. In Section 5.1 we show that if we allow unbounded input features then functions that are exactly computable by Mean-GNNs ; Max-GNNs; and others, cannot be approximated by Sum-GNNs.
- Essential also with finite input-features domain. In Section 5.2 we prove that even with just single-value input features, there are functions that can be exactly computed by a (Sum, Mean)-GNN (a GNN that use both Sum-aggregation and Mean-aggregation) or by a (Sum, Max)-GNN, but cannot be approximated by Sum-GNNs.
- The world is not enough. In Section 6, we examine GNNs with any finite combination of Sum; Mean; Max and other aggregations, and prove upper bounds on their expressivity already in the single-value input features setting.

Lastly, in Section 7 we experiment with synthetic data and observe that what we proved to be expressible is to an extent also learnable, and that in practice inexpressivity is manifested in a significantly higher error than implied in theory.

All proofs, some of the lemmas, and extended illustration and analysis of the experimentation, are found in the full version[1].

1.2 Related Work

The term Graph Neural Network, along with one of the basic models of GNNs, was introduced in [Scarselli *et al.*, 2008]. Since then, more than a few works have explored aspects of expressivity of GNNs. Some have explored the distinguishing power of different models of GNNs [Abboud *et al.*, 2021; Barceló *et al.*, 2021; Geerts and Reutter, 2022; Maron *et al.*, 2019; Morris *et al.*, 2019; Morris *et al.*, 2020; Sato *et al.*, 2021], and some have examined the expressivity of GNNs depending on the aggregations they use [Corso *et al.*, 2020; Xu *et al.*, 2019]. In [Chen *et al.*, 2019], a connection between distinguishing power and function approximation is described. In all of the above, the non-uniform notion was considered. In the uniform notion, it was proven that Sum-GNNs can express every logical formula in Guarded Countable Logic with 2 variables (GC2) [Barceló *et al.*, 2020b; Barceló *et al.*, 2020a]. A theoretical survey of the expressivity of GNNs is found in [Grohe, 2021], and a practical survey of different models of GNNs is found in [Wu *et al.*, 2020].

2 Preliminaries

By $\mathbb{N}, \mathbb{N}_{>0}, \mathbb{Q}, \mathbb{R}$ we denote the sets of nonnegative integers, positive integers, rational numbers, an d real numbers, respectively. For $a, b \in \mathbb{N} : a \leq b$ we denote the set $\{n \in \mathbb{N} : a \leq n \leq b\}$ by $[a..b]$. For $b \in \mathbb{N}_{>0}$ we denote the set $[1..b]$ by $[b]$. For $a, b \in \mathbb{R} : a \leq b$, we denote the set $\{r \in \mathbb{R} : a \leq r \leq b\}$ by $[a, b]$. We may use the terms "average" and "mean" interchangeably to denote the arithmetic mean. We use "{}" as notation for a multiset. Let $x \in \mathbb{R}, b \in \mathbb{N}_{>0}$, we define $\binom{\{x\}}{b} := \{x, \ldots, x\}$ the multiset consisting of b instances of x. Let $d \in \mathbb{N}_{>0}$ and let a vector $v \in \mathbb{R}^d$, we define $|v| := max(|v_i|_{i \in [d]})$. Let two vectors $u, v \in \mathbb{R}^d$, we define $' \leq '$: $u \leq v \Leftrightarrow \forall i \in [d] u_i \leq v_i$.

2.1 Graphs

An *undirected graph* $G = \langle V(G), E(G) \rangle$ is a pair, $V(G)$ being a set of vertices and $E(G) \subseteq \{\{u, v\} \mid u, v \in V(G)\}$ being a set of undirected edges. For a vertex $v \in V(G)$ we denote by $N(v) :=$

[1]see https://arxiv.org/abs/2302.11603

$\{w \in V(G) \mid \{w, v\} \in E(G)\}$ the neighbourhood of v in G, and we denote the size of it by $n_v := |N(v)|$.

A (vertex) *featured graph* $G = \langle V(G), E(G), S^d, Z(G)\rangle$ is a 4-tuple being a graph with a *feature map* $Z(G) : V(G) \to S^d$, mapping each vertex to a d-tuple over a set S. We denote the set of graphs featured over S^d by $\mathcal{G}_{S^d}$, we define $\mathcal{G}_S := \bigcup_{d\in\mathbb{N}} \mathcal{G}_{S^d}$, and we denote the set of all featured graphs by $\mathcal{G}_*$. The special set of graphs featured over $\{1\}$ is denoted $\mathcal{G}_1$. We denote the set of all feature maps that map to S^d by $\mathcal{Z}_{S^d}$, we denote $\bigcup_{d\in\mathbb{N}} \mathcal{Z}_{S^d}$ by $\mathcal{Z}_S$, and we denote the set of all feature maps by $\mathcal{Z}_*$. Let a featured-graph domain $D \subseteq \mathcal{G}_*$, a mapping $f : \mathcal{G}_D \to \mathcal{Z}_*$ to new feature maps is called a *feature transformation*.

For a featured graph G and a vertex $v \in V(G)$ we define $\text{sum}(v) := \Sigma_{w\in N(v)} Z(G)(w)$, $\text{avg}(v) := \frac{1}{n_v}\,\text{sum}(v)$, and $\max(v) := \max(Z(G)(w) : w \in N(v))$. In this paper, we consider the size of a graph G to be its number of vertices, that is, $|G| := |V(G)|$.

2.2 Feedforward Neural Networks

A *feedforward neural network (FNN)* $\mathfrak{F}$ is directed acyclic graph where each edge e carries a *weight* $w_e^{\mathfrak{F}} \in \mathbb{R}$, each node v of positive in-degree carries a *bias* $b_v^{\mathfrak{F}} \in \mathbb{R}$, and each node v has an associated continuous *activation function* $\mathfrak{a}_v^{\mathfrak{F}} : \mathbb{R} \to \mathbb{R}$. The nodes of in-degree 0, usually $X_1, \ldots, X_p$, are the *input nodes* and the nodes of out-degree 0, usually $Y_1, \ldots, Y_q$, are the *output nodes*. We denote the underlying directed graph of an FNN $\mathfrak{F}$ by $(V(\mathfrak{F}), E(\mathfrak{F}))$, and we call $(V(\mathfrak{F}), E(\mathfrak{F}), (\mathfrak{a}_v^{\mathfrak{F}})_{v\in V(\mathfrak{F})})$ the *architecture of* $\mathfrak{F}$, notated $A(\mathfrak{F})$. We drop the indices $^{\mathfrak{F}}$ at the weights and the activation function if $\mathfrak{F}$ is clear from the context.

The *input dimension* of an FNN is the number of input nodes, and the *output dimension* is the number of output nodes. The *depth* $\text{depth}(\mathfrak{F})$ of an FNN $\mathfrak{F}$ is the maximum length of a path from an input node to an output node.

To define the semantics, let $\mathfrak{F}$ be an FNN of input dimension p and output dimension q. For each node $v \in V(\mathfrak{F})$, we define a function $f_{\mathfrak{F},v} : \mathbb{R}^p \to \mathbb{R}$ by $f_{\mathfrak{F},X_i}(x_1, \ldots, x_p) := x_i$ for the ith input node X_i and

$$f_{\mathfrak{F},v}(\vec{x}) := \mathfrak{a}_v\left(b_v + \sum_{j=1}^{k} f_{\mathfrak{F},u_j}(\vec{x}) \cdot w_{e_j}\right)$$

for every node v with incoming edges $e_j = (u_j, v)$. Then $\mathfrak{F}$ computes the function $f_{\mathfrak{F}} : \mathbb{R}^p \to \mathbb{R}^q$ defined by

$$f_{\mathfrak{F}}(\vec{x}) := (f_{\mathfrak{F},Y_1}(\vec{x}), \ldots, f_{\mathfrak{F},Y_q}(\vec{x}))$$

Let $\mathfrak{F}$ an FNN, we consider the size of $\mathfrak{F}$ to be the size of its underlying graph. That is, $|\mathfrak{F}| = |V(\mathfrak{F})|$.

A common activation function is the ReLU activation, defined as $ReLU(x) := max(0, x)$. In this paper, we assume all FNNs to be ReLU activated. ReLU activated FNNs subsume every finitely-many-pieces piecewise-linear activated FNN, thus the results of this paper hold true for every such FNNs. Every ReLU activated FNN $\mathfrak{F}$ is Lipschitz-Continuous. That is, there exists a minimal $a_{\mathfrak{F}} \in \mathbb{R}_{\geq 0}$ such that for every input and output coordinates (i, j), for every specific input arguments $x_1, \ldots, x_n$, and for every $\delta > 0$, it holds that

$$\left| f_{\mathfrak{F}}(x_1, \ldots, x_n)_j - f_{\mathfrak{F}}(x_1, \ldots x_{i-1}, x_i + \delta, \ldots, x_n)_j \right| / \delta \leq a_{\mathfrak{F}}$$

We call $a_{\mathfrak{F}}$ the *Lipschitz-Constant* of f.

2.3 Graph Neural Networks

Several GNN models are described in the literature. In this paper, we define and consider the Aggregate-Combine (AC-GNN) model [Xu *et al.*, 2019; Barceló *et al.*, 2020b]. Some of our results extend straightforwardly to the messaging scheme of MPNN [Gilmer *et al.*, 2017], yet such extensions are out of scope of this paper.

A *GNN layer*, of input and output (I/O) dimensions $p; q$, is a pair $(\mathfrak{F}, agg)$ such that: $\mathfrak{F}$ is an FNN of I/O dimensions $2p; q$, and agg is an order-invariant p-dimension multiset-to-one aggregation function. An m-layer GNN $\mathcal{N} = ((\mathfrak{F}_1, agg_1), \ldots, (\mathfrak{F}_m, agg_m))$, of I/O dimensions $p; q$, is a sequence of m GNN layers of I/O dimensions $p^{(i)}; q^{(i)}$ such that: $p^{(1)} = p$, $q^{(m)} = q$ and $\forall i \in [m-1]\ p^{(i+1)} = q^{(i)}$. It determines a series of m feature transformations as follows: Let a graph $G \in \mathcal{G}_{\mathbb{R}^p}$ and vertex $v \in V(G)$, then $\mathcal{N}^{(0)}(G, v) := Z(G)(v)$, and for $i \in [m]$ we define a transformation

$$\mathcal{N}^{(i)}(G, v) := f_{\mathfrak{F}_i}(\mathcal{N}^{(i-1)}(G, v), agg_i(\mathcal{N}^{(i-1)}(G, w) : w \in N(v)))$$

We notate by $\mathcal{N}(G, v) := \mathcal{N}^{(m)}(G, v)$ the final output of $\mathcal{N}$ for v. We define the size of $\mathcal{N}$ to be $|\mathcal{N}| := \Sigma_{i\in[m]} |\mathfrak{F}_i|$ the sum of its underlying FNNs' sizes. We call $((A(\mathfrak{F}_1), agg_1), \ldots, (A(\mathfrak{F}_m), agg_m))$ the *architecture* of $\mathcal{N}$, notated $A(\mathcal{N})$, and say that $\mathcal{N}$ *realizes* $A(\mathcal{N})$. For an aggregation function agg, we denote by agg-GNNs the class of GNNs for which $\forall i \in [m]\ agg_i = agg$. For aggregation functions agg_1, agg_2, we denote by (agg_1, agg_2)-GNNs the class of GNNs with $m = 2n$ layers such that $\forall i \in [n]\ agg_{2i-1} = agg_1, agg_{2i} = agg_2$.

2.4 Expressivity

Let $p, q \in \mathbb{N}$, and a set S. Let $F = \{f : \mathcal{G}_{S^p} \to \mathcal{Z}_{\mathbb{R}^q}\}$ a set of feature transformations, and let a feature transformation $h : \mathcal{G}_{S^p} \to \mathcal{Z}_{\mathbb{R}^q}$. We say F *uniformly additively approximates* h, notated $F \approx h$, if and only if

$$\forall \varepsilon > 0 \exists f \in F : \forall G \in \mathcal{G}_{S^p} \forall v \in V(G)\ \ |f(G)(v) - h(G)(v)| \leq \varepsilon$$

The essence of uniformity is that one function "works" for graphs of all sizes, unlike non-uniformity where it is enough to have a specific function for each specific size of input graphs. The proximity measure is additive - as opposed to multiplicative where it is required that $\left|\frac{f(G)(v)-h(G)(v)}{h(G)(v)}\right| \leq \varepsilon$. In this paper, approximation always means uniform additive approximation and we use the term "approximates" synonymously with *expresses*. Although our no-approximation statements consider additive approximation, they hold true also for multiplicative approximation, and the respective proofs (in the full version) require not much additional argumentation to show that.

Let F, H be sets of feature transformations $f : \mathcal{G}_{S^p} \to \mathcal{Z}_{\mathbb{R}^q}$, we say F *subsumes* H, notated $F \geq H$ if and only if for every $h : \mathcal{G}_{S^p} \to \mathcal{Z}_{\mathbb{R}^q}$ it holds that $H \approx h \Rightarrow F \approx h$. If the subsumption holds only for graphs featured with a subset $T^p \subset S^p$ we notate it as $F \geq^T H$.

Let $p, q \in \mathbb{N}$. We call an order-invariant mapping $f : \mathcal{Z}_{\mathbb{R}^p} \to \mathbb{R}^q$, from feature maps to q-tuples, a *readout function*. Both sum and avg are commonly used to aggregate feature maps, possibly followed by an FNN that maps the aggregation value to a final output. We call a mapping $f : \mathcal{G}_{S^p} \to \mathbb{R}^q$, from featured graphs to q-tuples, a *graph embedding*. Let $w \in \mathbb{N}$, let a set of feature transformations $F = \{f : \mathcal{G}_{S^p} \to \mathcal{Z}_{\mathbb{R}^q}\}$, and let a readout $r : \mathcal{Z}_{\mathbb{R}^q} \to \mathbb{R}^w$, we notate the set of embeddings $\{r \circ f : f \in F\}$ by $r \circ F$. We use the expressivity terms and notations defined for feature transformations, for graph embeddings as well.

3 Mean and Max Do Not Subsume

It has already been stated that Sum-GNNs can express functions that Mean-GNNs and Max-GNNs cannot [Xu *et al.*, 2019]. For the sake of completeness we provide formal proofs that Mean-GNNs and Max-GNNs subsume neither Sum-GNNs nor each other.

3.1 Mean and Max Do Not Subsume Sum

Neither Mean-GNNs nor Max-GNNs subsume Sum-GNNs, even when the input-feature domain is a single value.

We define a featured star graph with (a parameter) k leaves, G_k (see Figure 1): For every $k \in \mathbb{N}_{>0}$:

- $V(G_k) = \{u\} \cup \{v_1, \dots, v_k\}$
- $E(G_k) = \bigcup_{i \in [k]} \{\{u, v_i\}\}$
- $Z(G_k) = \{(u, 1)\} \bigcup_{i \in [k]} \{(v_i, 1)\}$

Let $\mathcal{N}$ be an m-layer GNN. We define $u_k^{(t)} := \mathcal{N}^{(t)}(G_k, u)$, the feature of $u \in V(G_k)$ after operating the first t layers of $\mathcal{N}$. Note that $u_k^{(m)} = \mathcal{N}(G_k, u)$.

Lemma 3.1. *Assume $\mathcal{N}$ is a* Mean-*GNN or a* Max-*GNN . Let the maximum input dimension of any layer be d, and let the maximum Lipschitz-Constant of any FNN of $\mathcal{N}$ be a. Then, for every k it holds that* $\left|u_k^{(m)}\right| \le (da)^m$.

Theorem 3.2. *Let $f : \mathcal{G}_1 \to \mathcal{Z}_{\mathbb{R}}$ a feature transformation such that for every k it holds that $f(G_k)(u) = k$. Then,* Mean-*GNNs* $\not\approx f$ *and* Max-*GNNs* $\not\approx f$.

Note that by Theorem 3.2, a function such as neighbors-count is inexpressible by Mean-GNNs and Max-GNNs .

Corollary 3.3. *We have that* Mean-*GNNs* $\not\succeq^{\{1\}}$ Sum-*GNNs,* Max-*GNNs* $\not\succeq^{\{1\}}$ Sum-*GNNs.*

3.2 Mean and Max Do Not Subsume Each Other

Mean-GNNs and Max-GNNs do not subsume each other, even in a finite input-feature domain setting. We define a parameterized graph in which, depending on the parameters' arguments, the average of the center's neighbors is in $[0, \frac{1}{2}]$ while their max can be either 0 or 1. For every $k \in \mathbb{N}$ and $b \in \{0, 1\}$:

- $V(G_{k,b}) = \{u\} \cup \{v_1, \dots, v_k\} \cup \{w\}$
- $E(G_{k,b}) = \bigcup_{i \in [k]} \{\{u, v_i\}\} \cup \{\{u, w\}\}$
- $Z(G_{k,b}) = \{(u, 0)\} \bigcup_{i \in [k]} \{(v_i, 0)\} \cup \{(w, b)\}$

Theorem 3.4. *Let $f : \mathcal{G}_{\{0,1\}} \to \mathcal{Z}_{\mathbb{R}}$ a feature transformation such that for every k it holds that $f(G_{k,b})(u) = \frac{b}{k+1}$. Then,* Max-*GNNs* $\not\approx f$.

Theorem 3.5. *Let $f : \mathcal{G}_{\{0,1\}} \to \mathcal{Z}_{\mathbb{R}}$ a feature transformation such that for every k it holds that $f(G_{k,b})(u) = b$. Then,* Mean-*GNNs* $\not\approx f$.

Corollary 3.6. *We have that* Mean-*GNNs* $\not\succeq^{\{0,1\}}$ Max-*GNNs* , Max-*GNNs* $\not\succeq^{\{0,1\}}$ Mean-*GNNs* .

4 Sometimes Sum Subsumes

In a bounded input-feature domain setting, Sum-GNNs can express every function that Mean-GNNs and Max-GNNs can. The bounded input-feature domain results in a bounded range for Mean and Max, a fact which can be exploited to approximate the target GNN with a Sum-GNN. The approximating Sum-GNNs, that we describe, come at a size cost. We do not know if an asymptotically-lower-cost construction exist.

4.1 Mean by Sum

Sum-GNNs subsume Mean-GNNs in a bounded input-feature domain setting.

Lemma 4.1. *For every $\varepsilon > 0$ and $d \in \mathbb{N}_{>0}$, there exists a* Sum-*GNN $\mathcal{N}$ of size $O(d\frac{1}{\varepsilon})$ such that for every featured graph $G \in \mathcal{G}_{[0,1] \subset \mathbb{R}^d}$ it holds that $\forall v \in V(G)$* $\left|\mathcal{N}(G, v) - \text{avg}(v)\right| \le \varepsilon$.

Theorem 4.2. *Let a* Mean-*GNN $\mathcal{N}_M$ consisting of m layers, let the maximum input dimension of any layer be d, and let the maximum Lipschitz-Constant of any FNN of $\mathcal{N}_M$ be a. Then, for every $\varepsilon > 0$ there exists a* Sum-*GNN $\mathcal{N}_S$ such that:*

1. $\forall G \in \mathcal{G}_{[0,1]^d}\ \forall v \in V(G) \quad |\mathcal{N}_M(G, v) - \mathcal{N}_S(G, v)| \le \varepsilon$.
2. $|\mathcal{N}_S| \le O(|\mathcal{N}_M| + \frac{d \cdot m \cdot ad(1-(2ad)^m)}{\varepsilon(1-(2ad))})$.

Corollary 4.3. Sum-*GNNs* $\ge^{[0,1]}$ Mean-*GNNs.*

4.2 Max by Sum

Sum-GNNs subsume Max-GNNs in a bounded input-feature domain setting.

Lemma 4.4. *For every $\varepsilon > 0$ and $d \in \mathbb{N}_{>0}$, there exists a* Sum-*GNN $\mathcal{N}$ of size $O(d\frac{1}{\varepsilon})$ such that for every featured graph $G \in \mathcal{G}_{[0,1]^d}$ and vertex $v \in V(G)$ it holds that* $|\mathcal{N}(G, v) - \max(v)| \le \varepsilon$.

Theorem 4.5. *Let a* Max-*GNN $\mathcal{N}_M$ consisting of m layers, let the maximum input dimension of any layer be d, and let the maximum Lipschitz-Constant of any FNN of $\mathcal{N}_M$ be a. Then, for every $\varepsilon > 0$ there exists a* Sum-*GNN $\mathcal{N}_S$ such that:*

1. $\forall G \in \mathcal{G}_{[0,1]^d}\ \forall v \in V(G) \quad |\mathcal{N}_M(G, v) - \mathcal{N}_S(G, v)| \le \varepsilon$.
2. $|\mathcal{N}_S| \le O(|\mathcal{N}_M| + \frac{d \cdot m \cdot ad(1-(2ad)^m)}{\varepsilon(1-(2ad))})$.

Corollary 4.6. Sum-*GNNs* $\ge^{[0,1]}$ Max-*GNNs.*

5 Mean and Max Have Their Place

In two important settings, Mean and Max aggregations enable expressing functions that cannot be expressed with Sum alone. As in Section 3, we define a graph G_θ parameterized by θ over domain Θ. We define a feature transformation f on that graph and prove that it cannot be approximated by Sum-GNNs. The line of proofs (in the full version) is as follows:

1. We show that for every Sum-GNN $\mathcal{N}$ there exists a finite set $F_{\mathcal{N}}$ of polynomials of θ, those polynomials obtain a certain property φ, and it holds that:
$$\forall \theta \in \Theta\ \exists u_\theta \in V(G_\theta)\ \exists p \in F_{\mathcal{N}} :\ \mathcal{N}(G_\theta, u_\theta) = p(\theta)$$
2. We show that for every finite set F of polynomials (of θ) that obtain φ, it holds that:
$$\forall \varepsilon > 0\ \exists \theta \in \Theta :\ \forall p \in F\ \ |p(\theta) - f(G_\theta)(u_\theta)| > \varepsilon$$

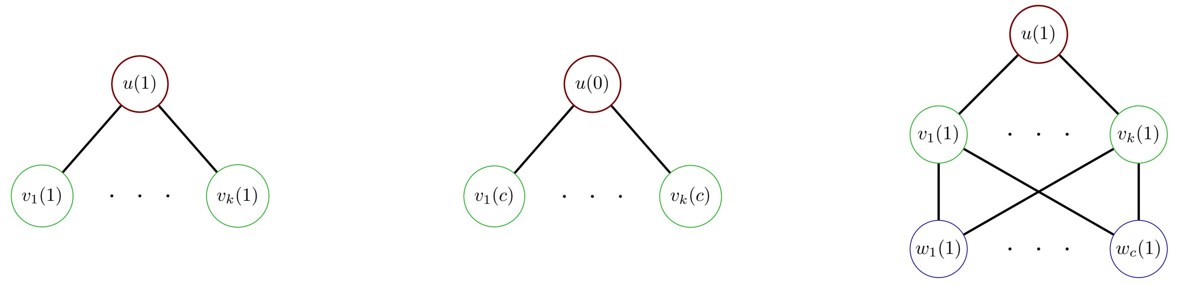


Figure 1: A star graph with k leaves, featured over a single-value input-feature domain (left); a star graph with k leaves, featured over $\mathbb{N}_{>0}$ (middle); a tripartite graph, with k intermediates fully connected to c leaves, featured over a single-value input-feature domain (right).

5.1 Unbounded, Countable, Input-Feature Domain

In an unbounded input-feature domain setting, Mean;Max and other GNNs are not subsumed by Sum-GNNs. We define a graph $G_{k,c}$ (see Figure 1): For $(k,c) \in \mathbb{N}^2_{>0}$,

- $V(G_{k,c}) = \{u\} \cup \{v_1,\ldots,v_k\}$
- $E(G_{k,c}) = \bigcup_{i\in[k]}\{\{u,v_i\}\}$
- $Z(G_{k,c}) = \{(u,0)\} \bigcup_{i\in[k]}\{(v_i,c)\}$

Theorem 5.1. *Let $f : \mathcal{G}_{\mathbb{N}^1} \to \mathcal{Z}_{\mathbb{R}}$ a feature transformation, such that for every k,c it holds that $f(G_{k,c})(u) = c$. Then,* Sum-*GNNs* $\not\approx f$.

Corollary 5.2. *Denote by S the set of all multisets over $\mathbb{N}_{>0}$. Let $g : S \to \mathbb{R}$ an aggregation such that $\forall a,b \in \mathbb{N}_{>0}\ g(\binom{\{a\}}{b}) = a$, that is, g aggregates every homogeneous multiset to its single unique value. Then,* Sum-*GNNs* $\not\succsim^{\mathbb{N}}$ *g-aggregation GNNs.*

Corollary 5.2 implies a limitation of Sum-GNNs compared to GNNs that use Mean; Max; or many other aggregations.

Graph Embedding

Sum-GNNs are limited compared to Mean; Max; and other GNNs, not only when used to approximate vertices' feature transformations but also when used in combination with a readout function to approximate graph embeddings. Consider another variant of $G_{k,c}$: For $(k,c) \in \mathbb{N}^2_{>0}$,

- $V(G_{k,c}) = \{u_1,\ldots,u_{k^2}\} \cup \{v_1,\ldots,v_k\}$
- $E(G_{k,c}) = \bigcup_{i\in[k^2],j\in[k]}\{\{u_i,v_j\}\}$
- $Z(G_{k,c}) = \bigcup_{i\in[k^2]}\{(u_i,0)\} \bigcup_{i\in[k]}\{(v_i,c)\}$

Theorem 5.3. *Let $f : \mathcal{G}_{\mathbb{N}^1} \to \mathbb{R}$ a graph embedding such that $\forall k,c\ f(G_{k,c}) = \frac{kc}{k+1}$. Let an aggregation $\mathfrak{a} \in \{\text{sum}, \text{avg}\}$ and an FNN $\mathfrak{F}$, and define a readout $\mathfrak{ro} := f_{\mathfrak{F}} \circ \mathfrak{a}$. Then,* $\mathfrak{ro} \circ$ Sum-*GNNs* $\not\approx f$.

Corollary 5.4. *Denote by S the set of all multisets over $\mathbb{N}_{>0}$. Let $g : S \to \mathbb{R}$ an aggregation such that $\forall a,b \in \mathbb{N}_{>0}\ g(\binom{\{a\}}{b}) = a$. Let an aggregation $\mathfrak{a} \in \{\text{sum}, \text{avg}\}$ and an FNN $\mathfrak{F}$, and define a readout $\mathfrak{ro} := f_{\mathfrak{F}} \circ \mathfrak{a}$. Then,* $\mathfrak{ro} \circ$ Sum-*GNNs* $\not\succsim^{\mathbb{N}}$ avg $\circ$ *g-GNNs.*

We have shown that Sum-GNNs do not subsume Mean and Max (and many other) GNNs. The setting though, consisted of an input-feature domain $\mathbb{N}_{>0}$, that is, countable unbounded.

5.2 Finite Input-Feature Domain

Mean and Max aggregations are essential also when the input-feature domain is just a single value i.e. when the input is featureless graphs. We define a new graph $G_{k,c}$ (see Figure 1): For every $(k,c) \in \mathbb{N}^2_{>0}$,

- $V(G_{k,c}) = \{u\} \cup \{v_1,\ldots,v_k\} \cup \{w_1,\ldots,w_c\}$
- $E(G_{k,c}) = \bigcup_{i\in[k]}\{\{u,v_i\}\} \bigcup_{i\in[k],j\in[c]}\{\{v_i,w_j\}\}$
- $Z(G_{k,c}) = \{(u,1)\} \bigcup_{i\in[k]}\{(v_i,1)\} \bigcup_{i\in[c]}\{(w_i,1)\}$

Theorem 5.5. *Let $f : \mathcal{G}_1 \to \mathcal{Z}_{\mathbb{R}}$ a feature transformation, such that for every k,c it holds that $f(G_{k,c})(u) = c$. Then,* Sum-*GNNs* $\not\approx f$.

Corollary 5.6. *Denote by S the set of all multisets over $\mathbb{N}_{>0}$, and let $g : S \to \mathbb{R}$ an aggregation such that $\forall a,b \in \mathbb{N}_{>0}\ g(\binom{\{a\}}{b}) = a$. Then,* Sum-*GNNs* $\not\succsim^{\{1\}}$ *(Sum, g)-GNNs.*

Corollary 5.6 implies a limitation of Sum-GNNs compared to stereo aggergation GNNs that combine Sum with Mean; Max; or many other aggregations. The limitation exists even when the input-feature domain consists of only a single value.

Graph Embedding

Completing the no-subsumption picture, Sum-GNNs are not subsuming, in a 2-values input-feature domain setting, also when used in combination with a readout function to approximate graph embeddings. We define $G_{k,c}$: For every $(k,c) \in \mathbb{N}^2_{>0}$,

- $V(G_{k,c}) = \{u_1,\ldots,u_{k^2}\} \cup \{v_1,\ldots,v_{k^3}\} \cup \{w_1,\ldots,w_{kc}\}$
- $E(G_{k,c}) = \bigcup_{j\in[k^2],i\in[k^3]}\{\{u_j,v_i\}\} \bigcup_{i\in[k^3],j\in[kc]}\{\{v_i,w_j\}\}$
- $Z(G_{k,c}) = \bigcup_{i\in[k^2]}\{(u_i,0)\} \bigcup_{i\in[k^3]}\{(v_i,0)\} \bigcup_{i\in[kc]}\{(w_i,1)\}$

Theorem 5.7. *Let $f : \mathcal{G}_{\{0,1\}^1} \to \mathbb{R}$ a graph embedding such that $\forall k,c\ f(G_{k,c}) = \frac{(k^2+kc)kc}{k^3+k^2+kc}$. Let an aggregation $\mathfrak{a} \in \{\text{sum}, \text{avg}\}$ and an FNN $\mathfrak{F}$, and define a readout $\mathfrak{ro} := f_{\mathfrak{F}} \circ \mathfrak{a}$. Then,* $\mathfrak{ro} \circ$ Sum-*GNNs* $\not\approx f$.

Corollary 5.8. *Denote by S the set of all multisets over $\mathbb{N}_{>0}$. Let $g : S \to \mathbb{R}$ an aggregation such that $\forall a,b \in \mathbb{N}_{>0}\ g(\binom{\{a\}}{b}) = a$. Let an aggregation $\mathfrak{a} \in \{\text{sum}, \text{avg}\}$ and an FNN $\mathfrak{F}$, and define a readout $\mathfrak{ro} := f_{\mathfrak{F}} \circ \mathfrak{a}$. Then,* $\mathfrak{ro} \circ$ Sum-*GNNs* $\not\succsim^{\{0,1\}}$ avg $\circ$ *(Sum, g)-GNNs.*

6 Sum and More Are Not Enough

In previous sections we showed that Sum-GNNs do not subsume Mean-GNNsand Max-GNNs , by proving that they cannot express specific functions. In this section, rather than comparing different GNNs classes we focus on one broad GNNs class and show that it is limited in its ability to express any one of a certain range of functions.

Denote by S the set of all multisets over $\mathbb{R}$, and let an aggregation $\mathfrak{a} : S \to \mathbb{R}$. We say that $\mathfrak{a}$ is a *uniform polynomial aggregation* (UPA) if and only if for every homogeneous multiset $\binom{\{x\}}{b}$, $x \in \mathbb{R}, b \in \mathbb{N}_{>0}$ it holds that $\mathfrak{a}(\binom{\{x\}}{b})$ is either a polynomial of x or a polynomial of (bx). Note that Sum; Mean; and Max are all UPAs. We say that a GNN $\mathcal{N} = (\mathcal{L}^{(1)}, \ldots, \mathcal{L}^{(m)})$ is an MUPA-GNN (Multiple UPA) if and only if the aggregation input to each of its layers is defined by a series of UPAs. That is, $\mathcal{L}^{(i)} = (\mathfrak{F}^{(i)}, (\mathfrak{a}_1^{(i)}, \ldots, \mathfrak{a}_{b_i}^{(i)}))$, for some b_i UPAs.

We define a parameterized graph G_k (see Figure 1): For every $k \in \mathbb{N}_{>0}$:

- $V(G_k) = \{u\} \cup \{v_1, \ldots, v_k\}$
- $E(G_k) = \bigcup_{i \in [k]} \{\{u, v_i\}\}$
- $Z(G_k) = \{(u, 1)\} \bigcup_{i \in [k]} \{(v_i, 1)\}$

Lemma 6.1. *Let $\mathcal{A}$ an m-layer* MUPA-*GNN architecture, let l be the maximum depth of any FNN in $\mathcal{A}$, and let d be the maximum in-degree of any node in any FNN in $\mathcal{A}$. Then, there exists $r \in \mathbb{N}$ such that: for every GNN $\mathcal{N}$ that realizes $\mathcal{A}$ it holds that $\mathcal{N}(G_k, u)$ is piecewise-polynomial (of k) with at most $((d+1)^l)^m$ pieces, and each piece is of degree at most r.*

Lemma 6.1 implies that the architecture bounds (from above) the number of polynomial pieces, and their degrees, that make the function computed by any particular realization of the architecture. With Lemma 6.1 at our disposal, we consider any feature transformation that does not converge to a polynomial when applied to $u \in V(G_k)$ and viewed as a function of k. We show that such a function is inexpressible by MUPA-GNNs.

Theorem 6.2. *Let $f : \mathcal{G}_1 \to \mathcal{Z}_\mathbb{R}$ a feature transformation, and define $g(k) := f(G_k)(u)$. Assume that g does not converge to any polynomial, that is, there exists $\varepsilon > 0$ such that for every polynomial p, for every K_0, there exists $k \geq K_0$ such that $|g(k) - p(k)| \geq \varepsilon$. Then,* MUPA-*GNNs* $\not\approx f$.

The last inexpressivity property we prove, concerns a class of functions which we call *PIL* (Polynomial-Intersection Limited). For $n \in \mathbb{N}$ denote by P_n the set of all polynomials of degree $\leq n$. We say that a function $f : \mathbb{N} \to \mathbb{R}$ is PIL if and only if for every $n \in \mathbb{N}$ there exists $k_n \in \mathbb{N}$ such that for every polynomial $p \in P_n$ there exist at most $k_n - 1$ consecutive integer points on which p and f assume the same value. Formally,

$$\sup\left(k : \forall p \in P_n \, \forall x \in \mathbb{N} \, \forall y \in [x..(x+k-1)] \, f(y) = p(y)\right) \in \mathbb{N}$$

We consider every feature transformation f such that for $g(k) := f(G_k)(u)$ it holds that g is PIL. This is a different characterization than "no polynomial-convergence" (in Theorem 6.2), and neither one implies the other. The result though, is weaker for the current characterization. We show that every MUPA-GNN architecture can approximate such a function only down to a certain $\varepsilon > 0$. That is, every GNN that realizes the architecture - no matter the specific weights of its FNNs - is far from the function by at least ε (at least in one point). The following lemma is an adaptation of the Polynomial of Best Approximation theorem [Mayans, 2006; Golomb, 1962] which is a step in the proof of the Equioscillation theorem attributed to Chebyshev.

Lemma 6.3. *For $x, k \in \mathbb{N}$ define $I_{x,k} := \{x, x+1, \ldots, x+k-1\}$ the set of consecutive k integers starting at x. Let $f : \mathbb{N} \to \mathbb{R}$ be a PIL, let $n \in \mathbb{N}$, and define $k_n :=$*

$$1 + \max(k : \forall p \in P_n \, \forall x \in \mathbb{N} \, \forall y \in [x..(x+k-1)] \, f(y) = p(y))$$

Then, for every $x \in \mathbb{N}$ there exists $\varepsilon_{x,k_n} > 0$ such that: for every $p \in P_n$ there exists $y \in I_{x,k_n}$ for which $|p(y) - f(y)| \geq \varepsilon_{x,k_n}$. That is, for every starting point x there is a bounded interval I_{x,k_n}, and a gap ε_{x,k_n}, such that no polynomial of degree $\leq n$ can approximate f on that interval below that gap.

Lemma 6.4. *For every $q, n \in \mathbb{N}$ there exists a point $T_{q,n} \in \mathbb{N}$ and a gap $\delta_{T_{q,n}} > 0$ such that: for every PIL $f : \mathbb{N} \to \mathbb{R}$, and every piecewise-polynomial g with q many pieces of degree $\leq n$, there exists $y \in \mathbb{N}$, $0 \leq y \leq T_{q,n}$ for which $|g(y) - f(y)| \geq \delta_{T_{q,n}}$. That is, the number of pieces and the max degree of a piecewise-polynomial g determine a guaranteed minimum gap by which g misses f within a guaranteed interval.*

Theorem 6.5. *Let $f : \mathcal{G}_1 \to \mathcal{Z}_\mathbb{R}$ a feature transformation, let $g(k) := f(G_k)(u)$, and assume that g is PIL. Then, for every* MUPA-*GNN architecture $\mathcal{A}$, there exists $\varepsilon_\mathcal{A} > 0$ such that for every* MUPA-*GNN $\mathcal{N}$ that realizes $\mathcal{A}$ there exists k such that $|\mathcal{N}(G_k, u) - f(G_k)(u)| \geq \varepsilon$.*

7 Experimentation

We experiment with vertex-level regression tasks. In previous sections we formally proved certain expressivity properties of Sum; Mean; and Max GNNs. Our goal in experimentation is to examine how these properties may affect practical learnability: searching for an approximating GNN using stochastic gradient-descend. With training data ranging over only a small subsection of the true-distribution range, does the existence of a uniformly-expressing GNN increase the chance that a well-generalizing GNN will be learned?

Specific details concerning training and architecture, as well additional illustrations and extended analysis, can be found in the full version[2],[3].

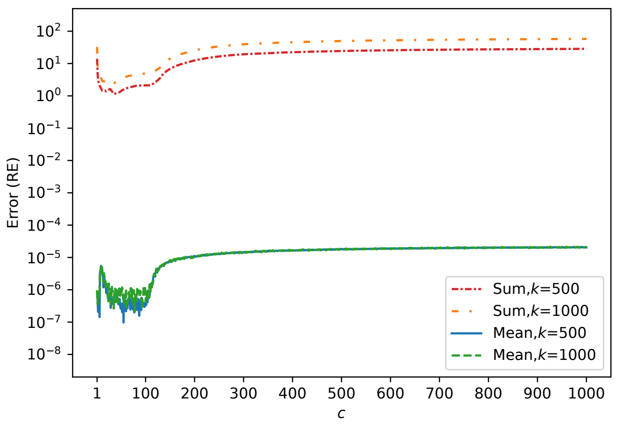
7.1 Data and Setup

For the graphs in the experiments, and with our GNN architecture consisting of two GNN layers (see full version), Mean and Max aggregations output the same value for every vertex, up to machine precision. Thus, it is enough to experiment with Mean and assume identical results for Max.

We conduct experiments with two different datasets, one corresponds to the approximation task in Section 5.1, and the other to the task in Section 5.2:

[2]see https://arxiv.org/abs/2302.11603

[3]code for running the experiments is found at https://github.com/toenshoff/Uniform_Graph_Learning



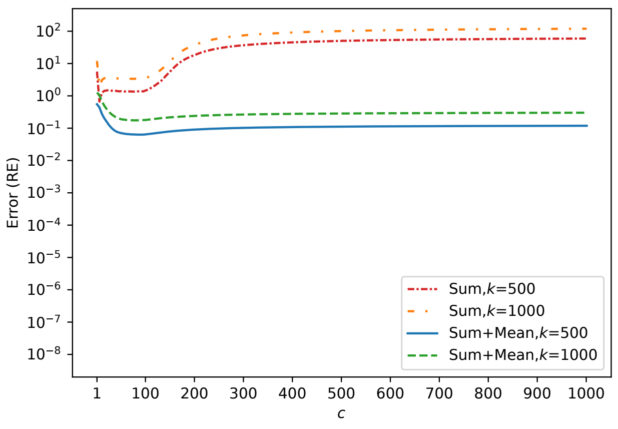


Figure 2: Relative Error of different aggregations on Unbounded Countable Features (left) and Single Value Features (right)

1. **U**nbounded **C**ountable Feature Domain (UC): This dataset consists of the star graphs $\{G_{k,c}\}$ from Section 5.1, for $k, c \in [1..1000]$. The center's ground truth value is c, and it is the only vertex whose value we want to predict.
2. **S**ingle-**V**alue Feature Domain (SV): This dataset consists of the graphs $\{G_{k,c}\}$ from Section 5.2, for $k, c \in [1..1000]$. Again, the center's ground truth value is c, and we do not consider the other vertices' predicted values.

As training data, we vary $k \in [1..100]$ and $c \in [1..100]$. We therefore train on 10K graphs in each experiment. Afterwards, we test each GNN model on larger graphs with $k \in [101..1000\}$ and $c \in [101..1000]$. Here, we illustrate our results for two representing values of k: 500, 1000, for all values of c. Illustrations of the full results can be found in the full version. The increased range of k and c in testing simulates the scenario of unbounded graph sizes and unbounded feature values, allowing us to study the performance in terms of uniform expressivity with unbounded features.

7.2 Results

Our primary evaluation metric is the relative error. Formally, if y_{pred} is the prediction of the GNN for the center vertex of an input graph G, with truth label c, we define the relative error as

$$\text{RE}\left(y_{\text{pred}}, c\right) = \frac{|y_{\text{pred}} - c|}{|c|}.$$

A relative error greater or equal to 1 is a strong evidence for inability to approximate, as the assessed approximation is no-better than an always-0 output. It is also reasonable that in practice, when judging the regression of a function whose range vary by a factor of 1000, relative error would be the relevant measure.

Unbounded, Countable, Feature Domain

Figure 2 provides the test results for UC. We plot the relative error against different values of c. Note that the error has a logarithmic scale. Mean-GNNs achieve very low relative errors of less than 10^{-4} across all considered combinations of k and c. Their relative error falls to less than 10^{-6} when c is within the range seen during training (≤ 100), Therefore, Mean-GNNs do show some degree of overfitting. Notably, the value of k has virtually no effect on the error of Mean-GNNs . This is expected, since mean aggregation should not be affected by the degree k of a center vertex whose neighbors are identical, up to machine precision. Sum-GNNs yield a substantially higher relative error. For $k = 500$ and $c \leq 100$ the relative error is roughly 1, but this value increases as c grows beyond the training range. Crucially, the relative error of Sum-GNNs also increases with k. For $k = 1000$, the relative error is above 1 even when c is within the range seen during training. Therefore, Sum-GNNs do generalize significantly worse than Mean-GNNs in both parameters k and c. '

Single-Value Feature Domain

Figure 2 provides the test results for SV. Again, we plot the relative error against different values of c. Sum-GNNs yield similar relative errors as in the UC experiment. As expected, learned (Sum,Mean)-GNNs do perform significantly better than Sum-GNNs. However, the learning of (Sum,Mean)-GNNs is not as successful as the learning of Mean-GNNs in the UC experiment: relative error is around 10^{-1} for $k = 500$, and slightly larger for $k = 1000$, clearly worse than the UC-experiment performance. In particular, the learned (Sum,Mean)-GNN is sensitive to increases in k. Note that each (Sum,Mean)-GNN layer receives both Sum and Mean aggregations arguments and needs to choose the right one, thus it is a different learning challenge than in the first experiment.

References

[Abboud *et al.*, 2021] Ralph Abboud, İsmail İlkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 2112–2118. ijcai.org, 2021.

[Barceló *et al.*, 2020a] Pablo Barceló, Egor V Kostylev, Mikael Monet, Jorge Pérez, Juan Reutter, and Juan-Pablo Silva. The logical expressiveness of graph neural networks. In *8th International Conference on Learning Representations (ICLR 2020)*, 2020.

[Barceló *et al.*, 2020b] Pablo Barceló, Egor V Kostylev, Mikaël Monet, Jorge Pérez, Juan L Reutter, and Juan-Pablo Silva. The expressive power of graph neural networks as a query language. *ACM SIGMOD Record*, 49(2):6–17, 2020.

[Barceló *et al.*, 2021] Pablo Barceló, Floris Geerts, Juan Reutter, and Maksimilian Ryschkov. Graph neural networks with local graph parameters. *Advances in Neural Information Processing Systems*, 34:25280–25293, 2021.

[Cappart *et al.*, 2021] Quentin Cappart, Didier Chételat, Elias B. Khalil, Andrea Lodi, Christopher Morris, and Petar Velickovic. Combinatorial optimization and reasoning with graph neural networks. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4348–4355. ijcai.org, 2021.

[Chen *et al.*, 2019] Zhengdao Chen, Soledad Villar, Lei Chen, and Joan Bruna. On the equivalence between graph isomorphism testing and function approximation with gnns. *Advances in neural information processing systems*, 32, 2019.

[Corso *et al.*, 2020] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *Advances in Neural Information Processing Systems*, 33:13260–13271, 2020.

[Geerts and Reutter, 2022] Floris Geerts and Juan L. Reutter. Expressiveness and approximation properties of graph neural networks. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[Gilmer *et al.*, 2017] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[Golomb, 1962] Michael Golomb. *Lectures on theory of approximation*. Argonne National Laboratory, Applied Mathematics Division, 1962.

[Grohe, 2021] Martin Grohe. The logic of graph neural networks. In *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–17. IEEE, 2021.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[Maron *et al.*, 2019] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *Advances in neural information processing systems*, 32, 2019.

[Mayans, 2006] Robert Mayans. The polynomial of best approximation. https://www.maa.org/sites/default/files/images/upload_library/4/vol6/Mayans/Best.html, 2006. Accessed: 2023-06-03.

[Morris *et al.*, 2019] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4602–4609, 2019.

[Morris *et al.*, 2020] Christopher Morris, Gaurav Rattan, and Petra Mutzel. Weisfeiler and leman go sparse: Towards scalable higher-order graph embeddings. *Advances in Neural Information Processing Systems*, 33:21824–21840, 2020.

[Sato *et al.*, 2021] Ryoma Sato, Makoto Yamada, and Hisashi Kashima. Random features strengthen graph neural networks. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 333–341. SIAM, 2021.

[Scarselli *et al.*, 2008] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[Tönshoff *et al.*, 2022] Jan Tönshoff, Berke Kisin, Jakob Lindner, and Martin Grohe. One model, any csp: Graph neural networks as fast global search heuristics for constraint satisfaction. *arXiv preprint arXiv:2208.10227*, 2022.

[Wu *et al.*, 2020] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[Xu *et al.*, 2019] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.