On Dimensionality of Feature Vectors in MPNNs

César Bravo [*1,2] **Alexander Kozachinskiy** [*2,3] **Cristóbal Rojas** [*1,2]

Abstract

We revisit the result of Morris et al. (AAAI'19) that message-passing graphs neural networks (MPNNs) are equal in their distinguishing power to the Weisfeiler–Leman (WL) isomorphism test. Morris et al. show their result with ReLU activation function and $O(n)$-dimensional feature vectors, where n is the size of the graph. Recently, by introducing randomness into the architecture, Aamand et al. (NeurIPS'22) improved this bound to $O(\log n)$-dimensional feature vectors, although at the expense of guaranteeing perfect simulation only with high probability. In all these constructions, to guarantee equivalence to the WL test, the dimension of feature vectors in the MPNN has to increase with the size of the graphs. However, architectures used in practice have feature vectors of constant dimension. Thus, there is a gap between the guarantees provided by these results and the actual characteristics of architectures used in practice. In this paper we close this gap by showing that, for *any* non-polynomial analytic (like the sigmoid) activation function, to guarantee that MPNNs are equivalent to the WL test, feature vectors of dimension $d = 1$ is all we need, independently of the size of the graphs.

1. Introduction

A plethora of real-life data is represented as a graph. A common deep-learning architecture for this kind of data is a graph neural network (GNN) (Scarselli et al., 2008). In this paper, we focus on message-passing graph neural networks (MPNN), which are rather simple architectures that have proved to be quite useful in practice.

[*]Equal contribution [1]Instituto de Ingeniería Matemática y Computacional, Universidad Católica de Chile [2]Centro Nacional de Inteligencia Artificial, Chile [3]Instituto Milenio Fundamentos de los Datos, Chile. Correspondence to: Alexander Kozachinskiy <alexander.kozachinskyi@cenia.cl>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

MPNNs work as follows. First, as inputs to them, we consider simple undirected graphs with node labels. MPNNs work in layers, where each node of a graph has its own *feature vector* that is updated at each layer. The initial feature vector of a node is some encoding of the label of this node (typically a one-hot encoding if the number of different labels is not too big). In each layer, the feature vector of a node v is updated as follows. One takes the sum of feature vectors of the neighbors of v, multiplies it by some matrix (with learnable entries), and adds the feature vector of v itself, multiplied by some other matrix (with learnable entries as well). After that, a non-linear activation function is applied coordinate-wise to the resulting vector, yielding the new feature vector of v.

Why do MPNNs perform well in practice? To answer this question, one needs to understand which specific architectural characteristics of MPNNs can provide guarantees on their performance, for example in terms of their trainability, generalization power, or their ability to fit the data. In this paper we will focus on the ability of MPNNs to accurately represent the data, a property commonly referred to as *expressive power*.

A main feature of MPNNs is that, by design, they compute functions on graphs that assign the same output to isomorphic graphs. That is, the functions they compute are always invariant under isomorphisms. Which invariant functions can MPNNs compute? It turns out that this question is related to the MPNNs's ability to distinguish non-isomorphic graphs, in the sense of being able to produce different outputs for graphs that are not isomorphic. An MPNN architecture is said to be *complete* for a class of graphs if it can distinguish all pairs of non-isomorphic graphs within the class. As shown in (Chen et al., 2019), an MPNN architecture is universal (can approximate any invariant function arbitrarily well) within a class of graphs, if and only if it is complete for that class. Therefore, it becomes relevant to identify the characteristics of MPNN architectures that guarantee maximal distinguishing power.

It was observed independently by (Xu et al., 2019) and (Morris et al., 2019) that the distinguishing power of MPNNs is upper bounded by the so-called Weisfeiler-Leman (WL) test (Leman & Weisfeiler, 1968), in the sense that MPNNs are incapable of distinguishing two node-labeled graphs

that are not distinguished by the WL-test. Let us briefly explain how the test works. Given two node-labeled graphs, the test iteratively updates the labels and checks, after each update, whether every label appears equally many times in both graphs. If a discrepancy arises during this process, the graphs are reported as non-isomorphic. In every update, the label of a node is replaced by a pair consisting of its current label together with the multiset of labels of its neighbors. It is known that there are non-isomorphic graphs that cannot be distinguished by this test, meaning that the test will consistently fail to report them as non-isomorphic. This implies that MPNNs are unable to distinguish all pairs of non-isomorphic graphs.

Do architectures that are used in practice attain this WL upper bound? In particular, what characteristics, such as the type of activation function or the dimension of feature vectors, are required to guarantee maximal expressive power?

Morris et al. answered this question by constructing, for any given graph, an MPNN that can perfectly simulate the WL-test on it. The characteristics of their MPNN are, however, graph-dependent: ReLU activation function works, and the dimension of feature vectors has to be $O(n)$, where n is the number of nodes in the graph. According to their result, if one is working with graphs with around 1000 nodes, feature vectors of dimension $d = 1000$ should be used. This is in stark contrast to what is done in practice, where the dimension of feature vectors is rather small (typically a few hundred) and independent of graph size. Furthermore, the parameter values in their construction also depend on the input graph.

An exponential improvement to the construction of Morris et al. was developed in (Aamand et al., 2022), where the dimension of the feature vectors was reduced to $O(\log(n))$. Their construction also has the advantage of being partially uniform, allowing a single MPNN to be used on all graphs of a certain size. Their architecture is, however, randomized, and perfect simulation is only guaranteed with high probability.

Does the dimension of feature vectors necessarily have to grow with the size of the graphs? What guarantees can be provided for MPNNs with continuous feature vectors of constant dimension, such as those used in practice?

In this paper we answer the question above by showing that for *any* non-polynomial analytic activation function (like the sigmoid), MPNNs with one-dimensional feature vectors are already as powerful as the WL-test. In particular, the dimension of the feature vectors does not have to grow with the size of a graph to guarantee full expressive power.

Key contributions.

1. We show that MPNNs with 1-dimensional features already attain the full expressive power of the WL test. Besides that, they require just 1 parameter per iteration. Moreover, for Lebesgue almost any fixation of parameters, we obtain a single MPNN that distinguishes any pair of graphs that can be distinguished by the WL test.

   In more detail, for an activation function $\mathbf{a}\colon \mathbb{R} \to \mathbb{R}$ we define an MPNN architecture with one parameter per layer that we call $1\mathrm{parMPNN}$. It is parametrized by a sequence $(\gamma^{(0)}, \gamma^{(1)}, \gamma^{(2)}, \ldots)$, where $\gamma^{(t)} \in \mathbb{R}$ denotes the parameter used in the t-th layer.

   Assume we give it a graph G, possibly with node labels coming from a fixed countable set. First, the architecture assigns an initial *feature scalar* $f^{(0)}(v) \in \mathbb{R}$ to every node v of G. More specifically, nodes with distinct labels are mapped into square roots of distinct prime numbers. Then, for $t = 0, 1, 2, \ldots$ and for every node v, the architecture computes the feature scalar $f^{(t)}(v) \in \mathbb{R}$ of v after t iterations, by updating them in the following fashion:

$$f^{(t+1)}(v) = \mathbf{a}\left(\frac{\gamma^{(t)}}{2} f^{(t)}(v) + \gamma^{(t)} \sum_{u \in N(v)} f^{(t)}(u)\right),$$

   where $N(v)$ denotes the set of neighbors of v in G. The architecture defines the read-out after T iterations as follows:

$$1\mathrm{parMPNN}^{(T)}(G) = \sum_{v \text{ is a node of } G} f^{(T)}(v).$$

   Theorem 1.1. *Let $\mathbf{a}\colon \mathbb{R} \to \mathbb{R}$ be any analytic non-polynomial function and let L be any countable set of labels. Then for every integer $T \geq 0$ and for Lebesgue almost all $(\gamma^{(0)}, \ldots, \gamma^{(T-1)}) \in \mathbb{R}^T$ the following holds. For any two graphs G_1, G_2 with node labels from the set L, we have*

$$1\mathrm{parMPNN}^{(T)}(G_1) \neq 1\mathrm{parMPNN}^{(T)}(G_2)$$

   if and only if G_1 and G_2 are distinguished by the WL test after T iterations.

2. We experimentally validate this theoretical result by demonstrating that our $1\mathrm{parMPNN}$ architecture is capable of achieving perfect simulation of the WL algorithm simultaneously for all graphs in the benchmark dataset for graph kernels (Kriege et al., 2020), even if we use *the same value of the parameter γ* in all the iterations. We also explored how the minimum number of precision bits required to guarantee perfect simulation depends on the size of the graphs. Our results suggest that logarithmically many bits of precision are enough.

Our technique. To show equivalence between MPNNs and the WL-test, one has to construct, for any given graph, some MPNN with the property that after any number of iterations, two nodes receive the same feature vectors if and only if they receive the same WL labels. The proof goes by induction on the number of iterations. In an MPNN, the multiset of labels of a node is encoded by the sum of the current feature vectors of the neighbors. To ensure the injectivity of this encoding, the construction of Morris et al. maintains *linear independence* of different feature vectors after each iteration. However, since the number of different labels in principle can be as large as n (the number of nodes), this argument requires the dimension of feature vectors to be at least n, as the number of linearly independent vectors in $\mathbb{R}^d$ can not be larger than d.

To overcome this difficulty, we just observe that for the argument it is enough to require linear independence over *rational numbers*. That is because multisets of labels are encoded as linear combinations of feature vectors, with coefficients that correspond to the multiplicities of the labels in these multisets, and multiplicities are always integer numbers. Now, the point is that even in $\mathbb{R}$, we can pick arbitrarily many numbers that are linearly independent over $\mathbb{Q}$, for instance, square roots of prime numbers, $\sqrt{2}, \sqrt{3}, \sqrt{5}$, and so on. Our main technical contribution is a demonstration that any non-polynomial analytic activation function can map arbitrarily many different numbers into a collection of numbers that are linearly independent over $\mathbb{Q}$, under a suitable choice of a single parameter.

Related work. A similar result to our Theorem 1.1 was recently obtained by (Amir et al., 2023), using an architecture also with one-dimensional features but three parameters per layer, whereas ours has only one. Additionally, our proof is shorter and requires only standard properties of analytic functions, easily derivable from definitions. At the same time, due to a recent result of (Khalife & Basu, 2024), one cannot drop the non-polynomiality condition in Theorem 1.1.

The connection between GNNs and the WL test has inspired the development of new, more expressive GNN architectures, mimicking *higher-order* versions of WL test (Cai et al., 1992) and enjoying the same theoretical equivalence to them (Morris et al., 2019; Maron et al., 2019). This connection was also established for more general models of graphs, such as geometric graphs (Joshi et al., 2023; delle Rose et al., 2023) and relational graphs (Barceló et al., 2022). In some cases, the new, WL-inspired architectures, demonstrated an improvement over standard MPNNs as well as over state-of-the-art models, for example, in the case of learning 3-dimensional point clouds (Li et al., 2023). We point out the interested reader to a survey on the use of the WL-test in machine learning (Morris et al., 2023b).

2. Preliminaries

A multiset is a function f from some set S to $\mathbb{N}$ (elements of S are taken with some finite *multiplicities*, specified by f). We use brackets $\{\!\{, \}\!\}$ to denote multisets. More precisely, if A, B are sets (and A is finite) and $\phi\colon A \to B$ is a function, we write $\{\!\{\phi(a) \mid a \in A\}\!\}$ for the multiset over elements of B, where each element $b \in B$ is taken with the multiplicity $|\phi^{-1}(b)|$ (that is, how many times b appears as the value of ϕ when we go through elements of A).

We consider simple undirected graphs with node labels taken from some countable set L. Such a graph can be given as a triple $G = \langle V, E, \ell\rangle$, where V is the set of nodes of the graph, $E \subseteq \binom{V}{2}$ is the set of edges of the graph (some set of 2-element subsets of V), and $\ell\colon V \to L$ is the node-labeling function. For $v \in V$, we denote by $N_G(v)$ the set of neighbors of v in the graph G, that is, $N_G(v) = \{u \in V \mid \{u, v\} \in E\}$. If G is clear from the context, we drop the subscript G.

Weisfieler Leman test. The Weisfeiler–Leman algorithm receives on input a node-labeled graph $G = \langle V, E, \ell\rangle$ and produces a sequence of "node labeling" functions

$$\phi_G^{(0)}, \phi_G^{(1)}, \phi_G^{(2)} \colon V \to ...,$$

defined iteratively as follows:

- $\phi_G^{(0)}(v) = \ell(v)$ for $v \in V$.
- $\phi_G^{(t+1)}(v) = \left(\phi_G^{(t)}(v), \{\!\{\phi_G^{(t)}(u) \mid u \in N(v)\}\!\}\right)$

The Weisfieler Leman test receives on input two labeled graphs $G_1 = \langle V_1, E_1, \ell_1\rangle$ and $G_2 = \langle V_2, E_2, \ell_2\rangle$. It runs the Weisfieler Leman algorithm on both of them, obtaining two sequences of node labeling functions $\{\phi_{G_1}^{(t)}\}_{t\geq 0}$ and $\{\phi_{G_2}^{(t)}\}_{t\geq 0}$. If for some $t \geq 0$ the test finds out that the multisets of WL-labels in G_1 and G_2 after t iterations are different:

$$\{\!\{\phi_{G_1}^{(t)}(v) \mid v \in V_1\}\!\} \neq \{\!\{\phi_{G_2}^{(t)}(v) \mid v \in V_2\}\!\},$$

the test outputs that these graphs can be distinguished. It is standard that whether or not the WL test distinguishes two graphs can be checked in polynomial time.

MPNNs. A T-layer *message-passing graph neural network* (MPNN) $\mathcal{N}$ with d-dimensional feature vectors and activation function $a\colon \mathbb{R} \to \mathbb{R}$ is specified by:

- a label-encoding function $\mathbf{e}\colon L \to \mathbb{R}^d$;
- a sequence of t pairs of $d \times d$ real matrices

$$W_1^{(1)}, W_2^{(1)}, \ldots, W_1^{(T)}, W_2^{(T)}$$

(in practice, elements of these matrices are to be learned).

Given a node-labeled graph $G = \langle V, E, \ell \rangle$, the MPNN $\mathcal{N}$ works on it as follows. First, it computes initial feature vectors using the label-encoding function:

$$f^{(0)}(v) = \mathbf{e}(\ell(v)) \in \mathbb{R}^d, \qquad v \in V.$$

Then for $t = 0, \ldots, T-1$, it does the following updates of the feature vectors:

$$f^{(t+1)}(v) = \mathbf{a}\left(W_1^{(t+1)} f^{(t)}(v) + W_2^{(t+1)} \sum_{u \in N(v)} f^{(t)}(u) \right), \tag{1}$$

$$f^{(t+1)}(v) \in \mathbb{R}^d, v \in V. \tag{2}$$

(we assume that $\mathbf{a}$ is applied component-wise).

The output of $\mathcal{N}$ on the graph G is defined as the sum of all feature vectors after T iterations:

$$\mathcal{N}(G) = \sum_{v \in V} f^{(T)}(v).$$

We say that two graphs G_1, G_2 are distinguished by $\mathcal{N}$ if $\mathcal{N}(G_1) \neq \mathcal{N}(G_2)$.

Non-polynomial analytic activation functions. We recall that an analytic function $f : \mathbb{R} \to \mathbb{R}$ is an infinitely differentiable function such that for every $x_0 \in \mathbb{R}$, its Taylor series at x_0:

$$T(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n} (x - x_0)^n$$

converges to $f(x)$ in a neighborhood of x_0.

We will require our activation functions to be analytic and non-polynomial. Common examples are the sigmoid functions such as the Arctangent function, or the logistic function:

$$\sigma(x) = \frac{e^x}{1 + e^x}.$$

We will need the following well-known fact. For the reader's convenience, we also include a proof.

Proposition 2.1. *Let $f : \mathbb{R} \to \mathbb{R}$ be an analytic function. Suppose there is a point x_0 where all the derivatives of f vanish. Then f is the constant zero function.*

Proof. We consider the set

$$D = \{x \in \mathbb{R} : f^{(n)}(x) = 0 \text{ for all } n \geq 0\}$$

and show that $D = \mathbb{R}$. First, we note that $x_0 \in D$, so D is nonempty. Second, we note that since f is analytic, for any $x \in D$ there is a neighborhood of x where f equals its Taylor series at x. It follows that every point in this neighborhood is also in D, and thus D is open. Finally, since all the $f^{(n)}$ are continuous, the sets $\{x : f^{(n)}(x) = 0\}$ are all closed. Thus, being a countable intersection of closed sets, D is also closed. Being open, closed, and nonempty, it follows that $D = \mathbb{R}$. □

3. Proof of the Main Result

We start with a formal definition of the architecture 1parMPNN. It is parametrized by a sequence $\{\gamma^{(t)} \in \mathbb{R}\}_{t=0}^{+\infty}$, with one parameter per iteration. Recall that we work with a countable set $L = \{z_1, z_2, z_3, \ldots\}$ of labels. We define a label-encoding function $\mathbf{e} \colon L \to \mathbb{R}$ by mapping distinct labels into roots of distinct prime numbers:

$$\mathbf{e}(z_i) = \sqrt{p_i}, \qquad \text{where } p_i \text{ is the } i\text{th prime.}$$

Given a graph $G = \langle V, E, \ell \rangle$ with a node labeling $\ell \colon V \to L$, we define for every $v \in V$ and $t \geq 0$ the feature scalar $f^{(t)}(v) \in \mathbb{R}$ of the node v after t iterations as follows:

$$f^{(0)}(v) = \mathbf{e}(\ell(v)), \tag{3}$$

$$f^{(t+1)}(v) = \mathbf{a}\left(\frac{\gamma^{(t)}}{2} f^{(t)}(v) + \gamma^{(t)} \sum_{u \in N(v)} f^{(t)}(u) \right), \tag{4}$$

$$v \in V, t \geq 0. \tag{5}$$

Finally, we define the read-out on G after t iterations of our architecture as:

$$1\text{parMPNN}^{(t)}(G) = \sum_{v \in V} f^{(t)}(v). \tag{6}$$

Theorem 3.1 (Restatement of Theorem 1.1)**.** *Let $\mathbf{a} \colon \mathbb{R} \to \mathbb{R}$ be any analytic non-polynomial function and let L be any countable set of labels. Then for every integer $T \geq 0$ and for Lebesgue almost all $(\gamma^{(0)}, \ldots, \gamma^{(T-1)}) \in \mathbb{R}^T$ the following holds. For any two graphs G_1, G_2 with node labels from the set L, we have*

$$1\text{parMPNN}^{(T)}(G_1) \neq 1\text{parMPNN}^{(T)}(G_2)$$

if and only if G_1 and G_2 are distinguished by the WL test after T iterations.

Proof. It is well-known and standard that if two graphs can be distinguished by some MPNN, in particular, by 1parMPNN after T iterations, then they can also be distinguished by the WL test after T iterations.

It remains to establish the other direction of the theorem. We say that a set $S \subseteq R$ is linearly independent over $\mathbb{Q}$

if for any $m \geq 1$ and for any distinct $x_1, \ldots, x_m \in S$ we have:

$$\lambda_1 x_1 + \ldots + \lambda_m x_m = 0 \implies \lambda_1 = \ldots = \lambda_m = 0$$

for all $\lambda_1, \ldots, \lambda_m \in \mathbb{Q}$. The theorem can be derived from the following lemma.

Lemma 3.2. *Let* $\mathbf{a}\colon \mathbb{R} \to \mathbb{R}$ *be any analytic non-polynomial function and* $G = \langle V, E, \ell\colon V \to L\rangle$ *be any node-labeled graph. Then for all* $T \geq 0$*, and for Lebesgue almost all* $(\gamma^{(0)}, \ldots, \gamma^{(T-1)}) \in \mathbb{R}^T$ *when* $T > 0$*, we have the following conditions:*

- $f^{(T)}(u) = f^{(T)}(v) \iff \phi^{(T)}(u) = \phi^{(T)}(v)$ *for every* $u, v \in V$*. Here* $\phi^{(T)}(v)$ *is the WL-label of the node* v *after* T *iterations.*
- *the set* $F_T = \{f^{(T)}(v) \mid v \in V\}$ *is linearly independent over* $\mathbb{Q}$*.*

Let us first explain how Lemma 3.2 implies the theorem. First of all, since there are just countably many graphs G with labels from L (because L is countable), we have that for Lebesgue almost all $(\gamma^{(0)}, \ldots, \gamma^{(T-1)}) \in \mathbb{R}^T$ both conditions of Lemma 3.2 are satisfied *for all* graphs G.

We now take any fixation of $(\gamma^{(0)}, \ldots, \gamma^{(T-1)}) \in \mathbb{R}^T$ for which conditions of Lemma 3.2 are true for all G and show for this fixation that $1\text{parMPNN}^{(T)}(G_1) \neq 1\text{parMPNN}^{(T)}(G_2)$ for any two graphs $G_1 = \langle V_1, E_1, \ell_1\rangle$ and $G_2 = \langle V_2, E_2, \ell_2\rangle$ that are distingushed by the WL test after T iterations. That is, our goal is to show that

$$\sum_{v \in V_1} f^{(T)}(v) \neq \sum_{v \in V_2} f^{(T)}(v). \tag{7}$$

Note that these two sums involve feature scalars of 1parMPNN from two different graphs, G_1 and G_2. However, we may equally assume that these are the feature scalars of the nodes of the disjoint union of G_1 and G_2, denoted by $G_1 \sqcup G_2$ (formally, it consists of a copy of G_1 and a copy of G_2, with no edges between the copies). To finish the argument, we use Lemma 3.2 for the graph $G_1 \sqcup G_2$. Since G_1 and G_2 are distinguished by the WL test after T iterations, there will be a WL label that appears a different number of times in G_1 and G_2 after T iterations. By the first condition of Lemma 3.2, this means that there will be a feature scalar that appears a different number of times in the left-hand side and the right-hand side of (7). Thus, the difference between the left-hand and the right-hand side of (7) gives a non-trivial linear combination of feature scalars of the nodes $G_1 \sqcup G_2$ with rational coefficients (in fact, integral). Since the feature scalars are linearly independent over $\mathbb{Q}$ by the second condition of Lemma 3.2, this difference has to be non-zero.

It remains to establish Lemma 3.2, which we do by induction on T. The base case, $T = 0$, holds just because our label-encoding function maps different labels into different numbers that form a set that is linearly independent over $\mathbb{Q}$.

Now, assume that the statement of the lemma is already proved for T. We establish it for $T + 1$. For that, we fix any choice of the first T parameters $(\gamma^{(0)}, \ldots, \gamma^{(T-1)}) \in \mathbb{R}^T$ for which the conditions of Lemma 3.2 are satisfied for T. We show that for all but countably many $\gamma^{(T)} \in \mathbb{R}$, by extending the sequence of parameters with $\gamma^{(T)}$, we satisfy the conditions of Lemma 3.2 for $T + 1$ (this means that if the set of values of the first T parameters that satisfy the lemma was full-measure in $\mathbb{R}^T$, then the set of values of the first $T + 1$ parameters that satisfy the lemma will be full-measure in $\mathbb{R}^{(T+1)}$ as well, as required).

Denoting

$$x_v = \frac{1}{2} \cdot f^{(T)}(v) + \sum_{u \in N(v)} f^{(T)}(u), \qquad v \in V,$$

we obtain by (4) that

$$f^{(T+1)}(v) = \mathbf{a}\left(\gamma^{(T)} \cdot x_v\right).$$

Using the first condition of our lemma for T, we first establish that $x_u = x_v$ if and only if $\phi^{(T+1)}(u) = \phi^{(T+1)}(v)$ for all $u, v \in V$. In other words, the value of x_v uniquely determines the value of $\phi^{(T+1)}(v)$, and vice versa. It is easy to see that

$$\phi^{(T+1)}(v) = \left(\phi^{(T)}(v), \{\!\{\phi^{(T)}(u) \mid u \in N(v)\}\!\}\right)$$

uniquely determines the value of x_v, because x_v is a function of $f^{(T)}(v)$ and $\sum_{u \in N(v)} f^{(T)}(u)$, which in turn are functions of $\phi^{(T)}(v)$ and $\{\!\{\phi^{(T)}(u) \mid u \in N(v)\}\!\}$, correspondingly.

We now establish that

$$x_v = \frac{1}{2} \cdot f^{(T)}(v) + \sum_{u \in N(v)} f^{(T)}(u) \tag{8}$$

uniquely determines $\phi^{(T+1)}(v)$. For that, we use the fact that the set of feature scalars F_T after T iterations is linearly independent over $\mathbb{Q}$. This means that by knowing the value of some linear combination of numbers from F_T with rational coefficients, we can uniquely determine the coefficients of this linear combination. In particular, knowing the value of x_v in (8), we first can determine the value $f^{(T)}(v)$ because it is the only feature whose coefficient is not integral. By subtracting $\frac{1}{2}$ from the coefficient in front of $f^{(T)}(v)$, we get the coefficients for all feature scalars with which they occur in the sum $\sum_{u \in N(v)} f^{(T)}(u)$, and these coefficients

allow us to restore the multiset $\{\!\{f^{(T)}(u) \mid u \in N(v)\}\!\}$. Using the condition that after T iterations, WL labels and feature scalars are in a one-to-one correspondence, we can then uniquely restore $\phi^{(T)}(v)$ and $\{\!\{\phi^{(T)}(u) \mid u \in N(v)\}\!\}$, and thus $\phi^{(T+1)}(v)$, as required.

At this point, it is enough to show that for all but countably many $\gamma^{(T)} \in \mathbb{R}$, the mapping:

$$x_v \mapsto \mathbf{a}(\gamma^{(T)} x_v) = f^{(T+1)}(v)$$

is injective (which establishes the first condition of the lemma for $T+1$) and that its image is linearly independent over $\mathbb{Q}$ (which establishes the second condition). This follows from the following lemma.

Lemma 3.3. *Let $\mathbf{a}\colon \mathbb{R} \to \mathbb{R}$ be any non-polynomial analytic function. Then, for any $m \in \mathbb{N}$, any sequence $x_1, \ldots, x_m$ of m real numbers such that $0 < |x_1| < |x_2| < \ldots < |x_m|$ (in other words, $x_1, \ldots, x_m$ are distinct non-zero real numbers among which no two are opposite to each other). and for all but countably many $\gamma \in \mathbb{R}$, it holds that, first, $\mathbf{a}(\gamma x_1), \ldots, \mathbf{a}(\gamma x_m)$ are distinct, and second, the set $\{\mathbf{a}(\gamma x_1), \ldots, \mathbf{a}(\gamma x_m)\}$ is linearly independent over $\mathbb{Q}$.*

To apply this lemma, we have to check that in the set $\{x_v \in v \in V\}$ all numbers are non-zero and no two are opposite to each other. This is because each x_v is a linear combination with *positive rational* coefficients of some numbers from F_T. This means that any equality of the form $x_v = 0$ or $x_v + x_u = 0$ would lead to a non-trivial linear combination with rational coefficients of some numbers from F_T, which is impossible because F_T is linearly independent over $\mathbb{Q}$. It only remains to prove Lemma 3.3.

Proof of Lemma 3.3. We show that for all but countably many $\gamma \in \mathbb{R}$ there exists no $(\lambda_1, \ldots, \lambda_m) \in \mathbb{Q}^m \setminus \{(0, 0, \ldots, 0)\}$ such that $\lambda_1 \mathbf{a}(\gamma x_1) + \ldots + \lambda_m \mathbf{a}(\gamma x_m) = 0$. For all such γ, we have that $\mathbf{a}(\gamma x_1), \ldots, \mathbf{a}(\gamma x_m)$ are distinct (otherwise, we could have taken $\lambda_i = 1, \lambda_j = -1$ for some $i \neq j$) and that the set $\{\mathbf{a}(\gamma x_1), \ldots, \mathbf{a}(\gamma x_m)\}$ is linearly independent over $\mathbb{Q}$.

Let BAD denote the set of $\gamma \in \mathbb{R}$ for which there exists a sequence $(\lambda_1^\gamma, \ldots, \lambda_m^\gamma) \in \mathbb{Q}^m \setminus \{(0, 0, \ldots, 0)\}$, such that:

$$\lambda_1^\gamma \mathbf{a}(\gamma x_1) + \ldots + \lambda_m^\gamma \mathbf{a}(\gamma x_m) = 0.$$

Our task is to show that BAD is countable. We have a mapping from BAD to a countable set $\mathbb{Q}^m \setminus \{(0, 0, \ldots, 0)\}$:

$$\Phi\colon \gamma \mapsto (\lambda_1^\gamma, \ldots, \lambda_m^\gamma).$$

To establish that the set BAD is countable, it is enough to show that for any $(\lambda_1, \ldots, \lambda_m) \in \mathbb{Q}^m \setminus \{(0, 0, \ldots, 0)\}$ there exists just countably many $\gamma \in \mathbb{R}$ such that

$$\lambda_1 \mathbf{a}(\gamma x_1) + \ldots + \lambda_m \mathbf{a}(\gamma x_m) = 0.$$

Indeed, this would imply that $\Phi^{-1}((\lambda_1, \ldots, \lambda_m))$ is countable for every $(\lambda_1, \ldots, \lambda_m) \in \mathbb{Q}^m \setminus \{(0, 0, \ldots, 0)\}$, meaning that we can represent BAD as a countable union of countable sets:

$$\text{BAD} = \bigcup_{\overline{\lambda} \in \mathbb{Q}^m \setminus \{(0,0,\ldots,0)\}} \Phi^{-1}(\overline{\lambda}),$$

getting that BAD is countable.

To finish the proof, we take an arbitrary tuple of coefficients $(\lambda_1, \ldots, \lambda_m) \in \mathbb{Q}^m \setminus \{(0, 0, \ldots, 0)\}$ and show that the function:

$$g(\gamma) = \lambda_1 \mathbf{a}(\gamma x_1) + \ldots + \lambda_m \mathbf{a}(\gamma x_m)$$

has only countably many roots. Observe that the function g is analytic. It is a well-known fact that any analytic function other than the zero function has at most countably many roots, meaning that it is enough to show that g is not the zero function. For the reader's convenience, we give a simple derivation of this fact about analytic functions from Proposition 2.1. Consider any analytic g with uncountably many roots. Then it has infinitely many roots in a bounded interval $[-n, n]$ for some $n \in \mathbb{N}$. Any bounded infinite set of real numbers has an accumulation point, that is, a point γ_0 such that every neighborhood of γ_0 has a point from the set other than γ_0. We apply this fact to the set of roots of g in $[-n, n]$. We observe that all derivatives of g at γ_0 must be equal to 0, implying that g is the zero function by Proposition 2.1. Indeed, otherwise we can write $g(\gamma) = \frac{g^{(k)}(\gamma_0)}{k!}(\gamma - \gamma_0)^k + o((\gamma - \gamma_0)^k)$ as $\gamma \to \gamma_0$ for the smallest k with $g^{(k)}(\gamma_0) \neq 0$, which would imply that $g(\gamma)$ is different from 0 for all $\gamma \neq \gamma_0$ sufficiently close to γ_0.

Thus, it remains to show that g is not the zero function, and we do this by obtaining a contradiction from the assumption that g is the zero function. Assuming that g is the zero function, we have:

$$g^{(k)}(0) = (\lambda_1 x_1^k + \ldots + \lambda_m x_m^k)\mathbf{a}^{(k)}(0)$$

for every $k \geq 0$. In turn, since $\mathbf{a}$ is a non-polynomial analytic function, we have $\mathbf{a}^{(k)}(0) \neq 0$ for infinitely many[1] k, meaning that

$$\lambda_1 x_1^k + \ldots + \lambda_m x_m^k = 0$$

for infinitely many k.

Take the largest $i \in \{1, \ldots, m\}$ such that $\lambda_i \neq 0$ (it exists

[1] Indeed, if all derivatives of $\mathbf{a}$ of sufficiently large order are equal to 0 at 0, by Proposition 2.1 we have that some derivative of $\mathbf{a}$ is the constant zero function, meaning that $\mathbf{a}$ itself is a polynomial.

because not all λ_i are equal to 0). Then we have:

$$\lim_{k\to\infty} \frac{\lambda_1 x_1^k + \ldots + \lambda_m x_m^k}{x_i^k} = \lim_{k\to\infty} \lambda_1 \left(\frac{x_1}{x_i}\right)^k + \ldots + \lambda_i \left(\frac{x_i}{x_i}\right)^k.$$

Since $0 < |x_1| < |x_2| < \ldots < |x_m|$, this limit is equal to $\lambda_i \neq 0$. However, the expression under the limit is equal to 0 for infinitely many k, a contradiction. □

□

4. Experiments

To precisely describe our experiments, let us start by establishing some terminology. We say that two labelings l_1 and l_2 of the nodes of a given graph G are *equivalent* if for any two nodes u, v in G, it holds that $l_1(u) = l_1(v) \iff l_2(u) = l_2(v)$. For a given graph G, we say that the WL algorithm *converges* in T steps if T is the first time for which the labelings at T and $T+1$ are equivalent. It is not hard to see that in this case the labeling of G at iteration T will in fact be equivalent to the labeling at $T+t$ for all $t > 0$. Given a graph G on which WL converges in T iterations, we say that a given MPNN $\mathcal{M}$ *perfectly simulates* WL on G if the labeling assigned to G by $\mathcal{M}$ after T iterations is equivalent to the one assigned by WL.

In our experiments, we implemented our architecture $1\text{parMPNN}\ \mathcal{M}_\gamma$ with sigmoid activation function and with $\gamma^{(0)} = \gamma^{(1)} = \gamma^{(2)} = \ldots = \gamma$ for a randomly chosen γ, so that *the same value of a parameter was used for all the iterations*.

Uniform perfect simulations To demonstrate the empirical validity of our theoretical findings, we tested our one-dimensional architecture 1parMPNN on the benchmark dataset for graphs kernels introduced in (Kersting et al., 2016; Kriege et al., 2020), which consists of a collection of 26 different datasets with graphs from different domains including molecular biology, social networks, and computer vision. The number of graphs in these datasets ranges from 300 for the smaller dataset up to 9362 graphs in the largest one. The average number of nodes goes from 14 to 430 (we refer to (Kriege et al., 2020) for a more detailed statistical description).

In our first experiment, we generated 10 values of $\gamma \in (0, 0.5)$[2] chosen uniformly at random and ran our architecture on these datasets. Recall that we are using the sigmoid activation function and *the same value of γ in all iterations*.

[2]We experimentally observed that smaller values for γ tend to work better, which explains the chosen range for this experiment.

For each dataset and each value of γ, we computed the *completeness ratio* of 1parMPNN as the fraction of graphs in the dataset on which it perfectly simulated WL:

$$\text{WL-completeness ratio} = \frac{\text{Perfect simulations}}{\text{Number of graphs}}$$

For graphs with node labels, we initialized their scalar features as the square roots of different prime numbers for different labels (suitably normalized). For unlabeled graphs, initial scalar node features were simply set to 1. All the computations were performed with 50 bits of precision. Figure 1 shows the completeness ratio for each chosen γ and for each dataset.

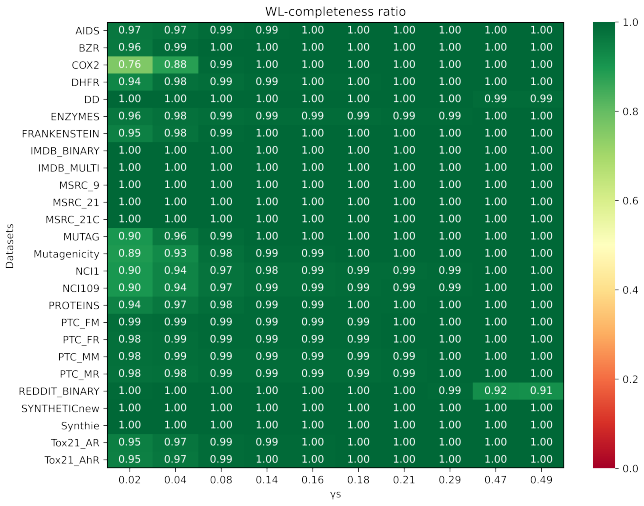

Figure 1. Empirical expressive power of a one-dimensional MPNN with a single parameter γ, for different values of γ and different datasets. The entries correspond to the fraction of graphs in the corresponding dataset on which the corresponding value of γ achieved a perfect simulation.

Dependence on precision bits To study how the number of precision bits required to achieve perfect simulation is affected by the size of the input graphs, we performed the following two experiments. First, we generated random Erdos-Renyi graphs of increasing sizes, from $n = 10$ to $n = 3000$ with a step size of 50, together with 20 values for γ chosen uniformly at random between 0 and 1. For each graph and each value of γ, we computed the minimum number of bits required for perfect simulation. Figure 2 shows, for each n, the average over γ of this minimum precision, together with the standard deviations. The obtained curve is approximately $O(\log(n))$.

Finally, we analyzed the real-world graph from the database

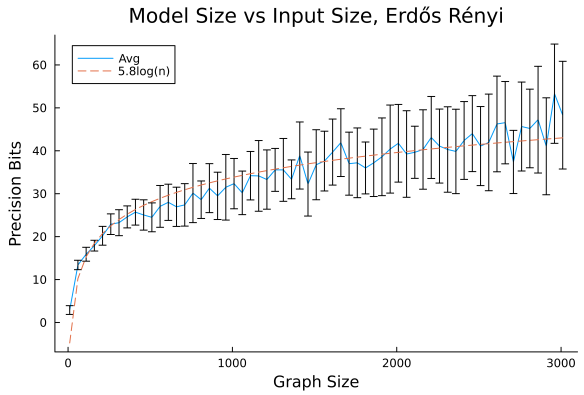

Figure 2. The figure shows the number of precision bits required for perfect simulation as a function of the size of the graph.

CORA[3] as well as a big Erdos-Renyi random graph with 5000 nodes and measured how the performance of the one-dimensional MPNNs improves as a function of the number of precision bits. Figure 3 shows, for 20 randomly chosen values of γ, the number of classes obtained by 1parMPNN as a function of the number of precision bits. We have made available the code of all our experiments[4].

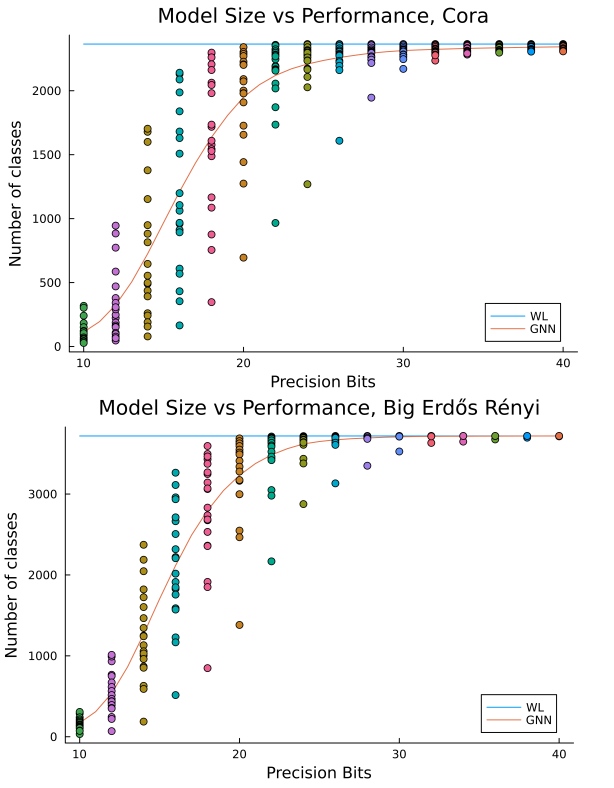

Figure 3. The figures display the quality of the simulation as a function of the number of precision bits. For each precision and each value of γ, the number of classes in the final labeling output by 1parMPNN is plotted. The true number of classes in the case of CORA is 2365, whereas for the big (5000 nodes) Erdos-Renyi graph is 3729.

5. Conclusion and Future Work

In this paper, we have demonstrated that, as far as expressive power is concerned, the dimensionality of feature vectors is not necessarily a restricting factor in the design of MPNNs. However, another restricting factor for practical computations is error-resilience. One can imagine that architectures of bigger size have better error-resilience. This leaves an open question, how error-resilient is our "minimalistic" architecture?

More specifically, assume that the result of any internal computation in our architecture can be "off" by some $\varepsilon > 0$. What is the minimal $\varepsilon > 0$ such that our architecture still has the same distinguishing power as the WL test in this setting? It is natural to conjecture that this time ε can not be chosen independently of the size of the graphs. Ideally, if n denotes the number of nodes, we could hope that $\varepsilon = 1/poly(n)$ is sufficient so that it is enough to do all computations with logarithmically many bits of precision.

Another interesting possible improvement of our main result would be to show that our 1parMPNN architecture has the same distinguishing power as the WL test, even if the value of parameters in all iterations are taken to be the same. Our experiments suggest that it might be true but we could not obtain a proof of this result.

Besides providing some theoretical justification of the behavior already observed in practice for these architectures, there are some consequences related to generalization that may be relevant for practical considerations. As shown in (Morris et al., 2023a), low-dimensional architectures exhibit significantly better generalization performance, as compared to higher-dimensional ones. According to the experiments they report, the difference between training and testing accuracies was systematically below 5% across all data sets for $d = 4$, whereas for $d = 1024$ this difference can become more than 45% for some data sets. The accuracy itself for low-dimensional architectures, however, was observed to vary significantly along different data sets, ranging from 30% of accuracy for some data sets to more than 90% for others.

What is the reason for this discrepancy in accuracy across different data sets?

A natural hypothesis is that such low-dimensional architectures have limited expressive power, and thus there exist

[3] https://graphsandnetworks.com/the-cora-dataset/

[4] https://anonymous.4open.science/r/Single-channel-GNN-B05F/

data sets that are simply impossible to fit for them. Our results, however, suggest that this may not necessarily be the case. The explanation, therefore, could rather be more related to the training process itself (relative to the data), than to the intrinsic limitations of the architecture. How does this trade-off work, and how to find the optimal dimensionality for different practical applications, are interesting topics for future research.

Acknowledgements

Bravo, Kozachinskiy, and Rojas were partially funded by the National Center for Artificial Intelligence CENIA FB210017, Basal ANID. Kozachinskiy was also partially funded by the Millennium Science Initiative Program - Code ICN17002.

We would also like to thank anonymous reviewers for their comments which help us to significantly improve our article. Specifically, we are grateful for pointing out the possibility of a uniform version of our result. In the initial submission, our result was stated in a form that any two graphs can be distinguished by some one-dimensional MPNN, without stressing that the same choice of parameters can be used for all graphs. During the rebuttal we made a claim that the same value of the parameter can be used in all iterations. Unfortunately, while preparing the camera-ready version we realized that our technique, while able to produce a uniform choice of parameters, does not allow us to guarantee that the same value can indeed be used in all iterations.

Impact statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Aamand, A., Chen, J., Indyk, P., Narayanan, S., Rubinfeld, R., Schiefer, N., Silwal, S., and Wagner, T. Exponentially improving the complexity of simulating the weisfeiler-lehman test with graph neural networks. *Advances in Neural Information Processing Systems*, 35: 27333–27346, 2022.

Amir, T., Gortler, S. J., Avni, I., Ravina, R., and Dym, N. Neural injective functions for multisets, measures and graphs via a finite witness theorem. *arXiv preprint arXiv:2306.06529*, 2023.

Barceló, P., Galkin, M., Morris, C., and Orth, M. A. R. Weisfeiler and leman go relational. In Rieck, B. and Pascanu, R. (eds.), *Learning on Graphs Conference, LoG 2022, 9-12 December 2022, Virtual Event*, volume 198 of *Proceedings of Machine Learning Research*, pp. 46. PMLR, 2022. URL https://proceedings.mlr.press/v198/barcelo22a.html.

Cai, J.-Y., Fürer, M., and Immerman, N. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992.

Chen, Z., Villar, S., Chen, L., and Bruna, J. On the equivalence between graph isomorphism testing and function approximation with gnns. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

delle Rose, V., Kozachinskiy, A., Rojas, C., Petrache, M., and Barcelo, P. Three iterations of (d − 1)-WL test distinguish non isometric clouds of d-dimensional points. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=9yhYcjsdab.

Joshi, C. K., Bodnar, C., Mathis, S. V., Cohen, T., and Lio, P. On the expressive power of geometric graph neural networks. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 15330–15355. PMLR, 2023. URL https://proceedings.mlr.press/v202/joshi23a.html.

Kersting, K., Kriege, N. M., Morris, C., Mutzel, P., and Neumann, M. Benchmark data sets for graph kernels. *URL http://graphkernels.cs.tu-dortmund.de.e*, 2016.

Khalife, S. and Basu, A. On the power of graph neural networks and the role of the activation function, 2024.

Kriege, N. M., Johansson, F. D., and Morris, C. A survey on graph kernels. *Applied Network Science*, 2020.

Leman, A. and Weisfeiler, B. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya*, 2(9):12–16, 1968.

Li, Z., Wang, X., Huang, Y., and Zhang, M. Is distance matrix enough for geometric deep learning? *arXiv preprint arXiv:2302.05743*, 2023.

Maron, H., Ben-Hamu, H., Serviansky, H., and Lipman, Y. Provably powerful graph networks. *Advances in neural information processing systems*, 32, 2019.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. Weisfeiler and leman

go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 4602–4609, 2019.

Morris, C., Geerts, F., Tönshoff, J., and Grohe, M. Wl meet vc, 2023a.

Morris, C., Lipman, Y., Maron, H., Rieck, B., Kriege, N. M., Grohe, M., Fey, M., and Borgwardt, K. Weisfeiler and leman go machine learning: The story so far. *Journal of Machine Learning Research*, 24(333): 1–59, 2023b. URL http://jmlr.org/papers/v24/22-0240.html.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. 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. URL https://openreview.net/forum?id=ryGs6iA5Km.