FSW-GNN: A Bi-Lipschitz WL-Equivalent Graph Neural Network

Yonatan Sverdlov
Technion – Israel Institute of Technology
yonatans@campus.technion.ac.il

Yair Davidson
Technion – Israel Institute of Technology
yairdavidson@campus.technion.ac.il

Nadav Dym
Technion – Israel Institute of Technology
nadavdym@technion.ac.il

Tal Amir
Technion – Israel Institute of Technology
talamir@technion.ac.il

Abstract

Famously, the ability of Message Passing Neural Networks (MPNN) to distinguish between graphs is limited to graphs separable by the Weisfeiler-Lemann (WL) graph isomorphism test, and the strongest MPNNs, in terms of separation power, are WL-equivalent. However, it was demonstrated that the quality of separation provided by standard WL-equivalent MPNN can be very low, resulting in WL-separable graphs being mapped to very similar, hardly distinguishable outputs. This phenomenon can be explained by the recent observation that standard MPNNs are not lower-Lipschitz. This paper addresses this issue by introducing FSW-GNN, the first MPNN that is fully bi-Lipschitz with respect to standard WL-equivalent graph metrics. Empirically, we show that our MPNN is competitive with standard MPNNs for several graph learning tasks and is far more accurate in long-range tasks, due to its ability to avoid oversmoothing and oversquashing. Our code is available at https://github.com/yonatansverdlov/Over-squashing.

1 Introduction

Graph neural networks are a central research topic in contemporary machine learning. As shown by Gilmer et al. [34], many of the most popular models can be seen as instantiations of Message Passing Neural Networks (MPNNs).

A well-known limitation of MPNNs is that they cannot differentiate between all distinct pairs of graphs. In fact, a pair of distinct graphs that cannot be separated by the Weisfeiler-Lehman (WL) graph isomorphism test will not be separated by *any* MPNN [61]. Accordingly, the most expressive MPNNs are those that are *WL-equivalent*, which means they can separate all pairs of graphs that are separable by WL. WL-equivalent MPNNs were proposed in the seminal works of Morris et al. [54] and Xu et al. [61], and the complexity of these constructions was later improved in [1, 5].

While separation is theoretically guaranteed with WL-equivalent MPNNs, in some cases, their separation in practice is so weak that it cannot be observed with 32-bit floating-point number; see [18]. Moreover, using many MPNN iterations often leads to almost-identical node features (oversmoothing), or features that are barely affected by changes in far-off nodes (oversquashing). These observations motivate the development of *quantitative* estimates of MPNN separation by means of bi-Lipschitz stability guarantees. These guarantees would ensure that Euclidean distances in the MPNN feature space are neither much larger nor much smaller than distances in the original graph space, which are defined by a suitable metric on graphs. Consequently, as we shall see, detrimental phenomena like oversmoothing and oversquashing can be fundamentally mitigated.

This paper introduces a novel MPNN, called *FSW-GNN* (*Fourier Sliced-Wasserstein GNN*), which is bi-Lipschitz with respect to two WL-equivalent graph metrics: (a) the Doubly Stochastic (DS) metric

Sverdlov et al., FSW-GNN: A Bi-Lipschitz WL-Equivalent Graph Neural Network. *Proceedings of the Fourth Learning on Graphs Conference (LoG 2025)*, PMLR 269, Hybrid Event, December 10–12, 2025.

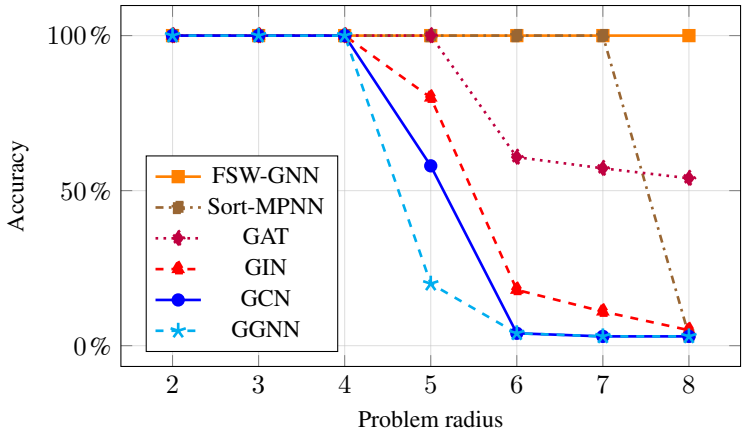

Figure 1: FSW-GNN handles tasks with large radius better than standard MPNNs, which are more prone to oversmoothing and oversquashing. Task taken from [3]

of [36], and (b) the Tree Mover's Distance (TMD) metric of Chuang and Jegelka [24]. Empirically, we show that FSW-GNN performs comparably or better than prevalent MPNNs on standard learning tasks, and achieves significantly superior performance in long-range tasks, i.e., tasks that require a large number of message-passing iterations (for example see Figure 1). This can be attributed to the bi-Lipschitzness of FSW-GNN, in contrast to standard MPNNs, which are not bi-Lipschitz [26].

1.1 Related Works

Bi-Lipschitzness. Bi-Lipschitzness arises naturally in many domains, including frames [7], phase retrieval [11, 23], group-invariant learning [19, 20] and multisets [4, 5]. In the context of MPNNs, upper Lipschitzness was discussed in [24, 48]. A recent survey by Morris et al. [51] identifies *bi-Lipschitzness* as a significant future goal for theoretical GNN research.

SortMPNN. First steps towards a Bi-Lipschitz MPNN have recently been made by Davidson and Dym [26]. Their work analyzes a weaker notion of Lipschitz and Hölder guarantees—*in expectation* over the model parameters. They show that essentially all popular MPNN models fail to be lower-Lipschitz, but are lower-Hölder in expectation, with an exponent that grows worse as the MPNN depth increases. In contrast, they propose a novel MPNN, called SortMPNN, and prove that it satisfies this weaker notion of bi-Lipschitzness in expectation.

As we discuss below, the FSW-GNN satisfies the standard, stronger, notion of bi-Lipschitzness. Interestingly, the proof technique we present here for FSW-GNN also applies to SortMPNN, allowing us to establish that SortMPNN is bi-Lipschitz as well. Nonetheless, SortMPNN has a key limitation in its message aggregation mechanism: it handles neighborhoods of different sizes by padding them to a predetermined maximal size. This requires *a priori* knowledge of the maximal neighborhood size in all future input graphs—an inherent constraint that significantly limits its applicability. In contrast, FSW-GNN does not share this constraint, since it treats vertex neighborhoods as distributions. Moreover, due to this padding-based approach, SortMPNN only accommodates neighborhoods that are multisets, thus making it unsuitable for graphs with non-integer edge weights. In contrast, FSW-GNN naturally supports edge weights.

MPNNs with advanced pooling mechanisms. In addition to SortMPNN, our approach is conceptually related to other MPNNs that replace the basic max- or sum-pooling, used for message aggregation, with more advanced pooling mechanisms, such as sorting [8, 63], standard deviation [25], or Sliced-Wasserstein embeddings via template distributions [47]. However, these methods lack the bi-Lipschitzness guarantees that our model provides.

WL-equivalent metrics. Bi-Lipschitz analysis of MPNNs requires a WL-equivalent graph metric. Several such metrics have been proposed, with notable examples being the *Doubly-Stochastic (DS)*

Table 1: Learning accuracy comparison across different benchmarks and models.

Model	Cora	Cite.	Pubm.	Cham.	Squi.	Actor	Corn.	Texa.	Wisc.
GCN	85.77	73.68	88.13	28.18	23.96	26.86	52.70	52.16	45.88
GAT	**86.37**	74.32	87.62	42.93	30.03	28.45	54.32	58.38	49.41
FSW-GNN	86.35	**75.44**	**88.17**	51.18	36.38	**34.66**	**72.43**	**75.68**	**81.56**
Sort-MPNN	83.46	72.69	85.15	**78.11**	**74.69**	31.32	67.03	70.54	73.92
SOTA	**90.16**	**82.07**	**91.31**	**79.71**	**76.71**	**51.81**	**92.72**	**88.38**	**94.99**

metric (also known as the *tree metric*) [36]; the *Tree Mover's Distance (TMD)* [24]; and the *WL metric* [21]. In this paper, we prove that the graph embeddings computed by our FSW-GNN model are bi-Lipschitz with respect to both the DS and TMD metrics. This analysis applies to graphs of bounded size with continuous, bounded node-features. Weaker notions of equivalence between these metrics, in the context of graphs with unbounded cardinality and without node features, are discussed in [15, 16].

2 Problem Setting

In this section, we outline the problem setting, first providing the theoretical background of the problem and then stating our objectives.

Vertex-featured graphs. Our main objects of study are graphs with vertex features, represented as triplets $G = (V, E, X)$, where $V = \{v_i\}_{i=1}^n$ is the set of vertices, $E \subseteq \{\{v_i, v_j\} \mid i, j \in [n]\}$ are the undirected edges in G, and $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]$ is a matrix containing the vertex feature vectors $\boldsymbol{x}_i \in \Omega$, with $\Omega \subseteq \mathbb{R}^d$ being the *feature domain*. We denote by $\mathcal{G}_{\leq N}(\Omega)$ the set of all vertex-featured graphs with at most N vertices and corresponding features in Ω. Throughout the paper, we use $\{\}$ to denote multisets. We note that our results readily extend to graphs with edge features.

Weisfeiler-Lemann Graph Isomorphism test. Two graphs are *isomorphic* if they are identical up to relabeling of their nodes. Perhaps surprisingly, the problem of determining whether two given graphs are isomorphic is rather challenging. To date, no known algorithm can solve it in polynomial time [6]. However, there exist various heuristics that provide an incomplete but often adequate method to test whether a given pair of graphs is isomorphic. The most notable example is the Weisfeiler-Leman (WL) graph isomorphism test.

The WL test can be described as assigning to each graph $G = (V, E, X)$ a feature vector c_G^T according to the formula

$$\begin{aligned} c_v^0 &:= \mathbf{X}_v, \quad v \in V; \quad \text{for } 1 \leq t \leq T: \\ c_v^t &:= \text{Combine}\big(c_v^{t-1}, \text{Aggregate}\big(\big\{c_u^{t-1} \mid u \in \mathcal{N}_v\big\}\big)\big); \\ c_G^T &:= \text{Readout}\big(\big\{c_{v_1}^T, \ldots, c_{v_n}^T\big\}\big), \end{aligned} \tag{1}$$

where Aggregate and Readout are functions that injectively map multisets of vectors in Euclidean space into another Euclidean space, Combine is an injective function from one Euclidean space to another, and $\mathcal{N}_v$ denotes the neighborhood of the vertex v in G.

Definition (WL graph equivalence)**.** Two vertex-featured graphs G and $\tilde{G}$ are said to be *WL-equivalent*, denoted by $G \overset{\text{WL}}{\sim} \tilde{G}$, if $c_G^T = c_{\tilde{G}}^T$ for all $T \geq 0$. Otherwise, they are said to be *WL-separable*.

It is a known fact [35, 53] that for $G, \tilde{G} \in \mathcal{G}_{\leq N}\big(\mathbb{R}^d\big)$, if the equality $c_G^T = c_{\tilde{G}}^T$ is satisfied for $T = N$, then it is satisfied for all $T \geq 0$, and thus $G \overset{\text{WL}}{\sim} \tilde{G}$.

While the WL test can separate most pairs of non-isomorphic graphs, there exist examples of non-isomorphic graph pairs that WL cannot separate; see [64].

Message passing neural networks. Message Passing Neural Networks (MPNNs) operate on a similar principle to the WL test, but with the purpose of performing predictions on single graphs rather than determining if pairs of them are isomorphic. Their core mechanism is the message-passing

procedure, which maintains a hidden feature for each vertex and iteratively updates it as a function of the neighbors' features. This process is outlined as follows:

1. **Initialization:** The hidden feature $\boldsymbol{h}_v^{(0)}$ of each node is initialized by its input feature $\boldsymbol{x}_v$.
2. **Message aggregation:** Each node $v \in V$ aggregates messages from its neighbors by
$$\boldsymbol{m}_v^{(t)} \coloneqq \text{Aggregate}\Big(\Big\{\boldsymbol{h}_u^{(t-1)} \;\Big|\; u \in \mathcal{N}_v\Big\}\Big) \tag{2}$$
Where Aggregate is a multiset-to-vector function.
3. **Update step:** Each node updates its own hidden feature according to its aggregated messages and its previous hidden feature, using a vector-to-vector *update function*:
$$\boldsymbol{h}_v^{(t)} \coloneqq \text{Update}\Big(\boldsymbol{m}_v^{(t)}, \boldsymbol{h}_v^{(t-1)}\Big), \tag{3}$$
4. **Readout:** After T iterations of steps 2-3, a graph-level feature $\boldsymbol{h}_G$ is computed by a multiset-to-vector *readout function*:
$$\boldsymbol{h}_G \coloneqq \text{Readout}\Big(\Big\{\boldsymbol{h}_v^{(T)} \;\Big|\; v \in V\Big\}\Big).$$

Numerous MPNNs were proposed in recent years, including GIN [61], GraphSage [38], GAT [60], and GCN [45], the main differences between them being the specific choices of the aggregation, update, and readout functions.

An MPNN computes an *embedding* of a graph G to a vector $F(G) = \boldsymbol{h}_G$. The obtained embedding is often further processed by standard machine-learning tools for vectors, such as multi-layer perceptrons (MLPs), to obtain a final graph prediction. The ability of such a model to approximate functions on graphs is closely related to the separation properties of F: if F can differentiate between any pair of non-isomorphic graphs, then a model of the form $\text{MLP} \circ F$ would be able to approximate any functions on graphs [22].

Unfortunately, MPNN cannot separate any pair of WL-equivalent graphs, even if they are not truly isomorphic [54, 61]. Accordingly, the best we can hope for from an MPNN, in terms of separation, is *WL equivalence*: for every pair of graphs $G, G' \in \mathcal{G}_{\leq N}(\Omega)$, $F(G) = F(G')$ *if and only if* $G \overset{\text{WL}}{\sim} G'$. While MPNNs based on max- or mean-pooling cannot be WL-equivalent [61], it is possible to construct WL-equivalent MPNNs based on sum-pooling, as discussed in [1, 5, 17, 54, 61]. Theoretically, a properly tuned graph model based on a WL-equivalent MPNN should be capable of perfectly solving any binary classification task, provided that no two WL-equivalent graphs have different ground-truth labels. However, this separation does not always manifest in practice. One reason is that WL-equivalent functions may map two input graphs that are far apart in the input space to outputs that are numerically indistinguishable in the output Euclidean space. In fact, Davidson and Dym [26] provide an example of graph pairs that are not WL-equivalent, yet are mapped to near-identical outputs by standard sum-based MPNNs. Consequently, these MPNNs fail on binary classification tasks for such graphs.

This paper aims to address this limitation by devising an MPNN whose embeddings preserve distances in the bi-Lipschitz sense. To state our goal formally, we first need to define a notion of distance on the input space of graphs.

WL metric for graphs. WL metrics quantify the *extent* to which two graphs are not WL-equivalent:

Definition 2.1 (WL metric)**.** A *WL metric on* $\mathcal{G}_{\leq N}(\Omega)$ is a function $\rho : \mathcal{G}_{\leq N}(\Omega) \times \mathcal{G}_{\leq N}(\Omega) \to \mathbb{R}_{\geq 0}$ that satisfies the standard requirements for a metric, with the exception that $\rho(G_1, G_2) = 0$ if and only if $G_1 \overset{\text{WL}}{\sim} G_2$.

For convenience, we use the term WL metric, despite the fact that strictly speaking, WL metrics are *pseudometrics* on $\mathcal{G}_{\leq N}\big(\mathbb{R}^d\big)$.

Tree Mover's Distance. The first WL metric we consider is the Tree Mover's Distance (TMD), defined in [24]. This metric is based on building computation trees $\mathbf{T}_v^{(t)}$, which simulate the WL procedure used to create the node features $h_v^{(t)}$, and calculating distances
$$\text{TD}(\mathbf{T}_v^{(t)}, \mathbf{T}_u^{(t)}) \tag{4}$$

recursively between sub-trees using optimal transport. These node-level distances are zero if, and only if, the features $c_v^{(t)}$ and $c_u^{(t)}$, constructed by the WL test, are equal. A graph-level distance $\mathrm{TMD}(G, G')$ is obtained by aggregating all node-level distances. For the full definition, see appendix C and [24]. Under the assumption that the feature domain Ω does not contain the zero vector, Chuang and Jegelka [24] proved that $\mathrm{TMD}(G, G')$ is a WL-metric.

The second WL metric we consider is the DS metric [35]. Originally, this metric was defined only for featureless graphs of fixed cardinality. In the next section, we extend this metric to the more general case of $\mathcal{G}_{\leq N}\left(\mathbb{R}^d\right)$.

Bi-Lipschitzness. Once a WL-metric is defined to measure distances between graphs, one can bound the distortion incurred by a graph embedding with respect to that metric, using the notion of bi-Lipschitzness:

Definition (Bi-Lipschitz embedding)**.** Let ρ be a WL-metric on $\mathcal{G}_{\leq N}(\Omega)$. An embedding $E : \mathcal{G}_{\leq N}(\Omega) \to \mathbb{R}^m$ is said to be *bi-Lipschitz with respect to ρ on $\mathcal{G}_{\leq N}(\Omega)$* if there exist constants $0 < c \leq C < \infty$ such that $\forall G_1, G_2 \in \mathcal{G}_{\leq N}(\Omega)$,

$$c \cdot \rho(G_1, G_2) \leq \|E(G_1) - E(G_2)\|_2 \leq C \cdot \rho(G_1, G_2). \tag{5}$$

If E only satisfies the left- or right-hand side of eq. (5), it is said to be *lower-* or *upper-Lipschitz* respectively.

Bi-Lipschitzness ensures that the embedding maps the input space $\mathcal{G}_{\leq N}(\Omega)$ into the output Euclidean space with bounded distortion, with the ratio $\frac{C}{c}$ serving as an upper bound on the distortion, akin to the condition number of a matrix.

3 Main Contributions

In this section, we discuss our main contributions. We begin by defining our generalized DS metric for vertex-featured graphs. We then discuss our proposed MPNN and show that it is bi-Lipschitz with respect to the DS and TMD metrics.

The DS metric. The DS metric originates from a relaxation of the *graph isomorphism problem*. Two graphs G and $\tilde{G}$, each with n vertices, and corresponding adjacency matrices A, $\tilde{A}$ are isomorphic if and only if there exists a *permutation matrix* P such that $AP = P\tilde{A}$. Since checking whether graphs are isomorphic is intractable, an approximate solution can be sought by considering the equation $AS = S\tilde{A}$ with $S \in \mathcal{D}_n$, the collection of $n \times n$ *doubly stochastic* matrices, which is the convex hull of all permutation matrices. Remarkably, this equation admits a solution $S \in \mathcal{D}_n$ if and only if the graphs G and $\tilde{G}$ are WL-equivalent [59]. Accordingly, a WL-metric between featureless graphs with the same number of vertices n can be defined by the minimization problem

$$\rho_{\mathrm{DS}}\left(G, \tilde{G}\right) = \min_{S \in \mathcal{D}_n} \left\|AS - S\tilde{A}\right\|, \tag{6}$$

where $\|\cdot\|$ could denote any p norm. The fact that this is indeed a pseudometric was established in [12]. The optimization problem eq. (6) can be solved by off-the-shelf convex optimization solvers and was considered as a method for finding the correspondence between two graphs in several papers, including [2, 13, 29, 31, 50].

The idea of using the DS metric for MPNN stability analysis was introduced in [36] and further discussed by Böker [15]. To apply this idea to our setting, we need to adapt this metric to vertex-featured graphs with varying numbers of vertices. We do this by augmenting it as follows:

$$\rho_{\mathrm{DS}}\left(G, \tilde{G}\right) = |n - \tilde{n}| + \min_{S \in \Pi(n, \tilde{n})} \left\|AS - S\tilde{A}\right\|_1 + \sum_{i \in [n], j \in [\tilde{n}]} S_{ij} \|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_j\|_1, \tag{7}$$

where n and $\tilde{n}$ denote the number of vertices in G and $\tilde{G}$, $\boldsymbol{x}_i$ and $\tilde{\boldsymbol{x}}_j$ denote the vertex features of G and $\tilde{G}$, and $\Pi(n, \tilde{n})$ is the set of $n \times \tilde{n}$ matrices S with non-negative entries, whose rows and columns sum to n and $\tilde{n}$, respectively.

Theorem 3.1. (Proof in appendix D.1) *ρ_{DS} is a WL-equivalent metric on $\mathcal{G}_{\leq N}\left(\mathbb{R}^d\right)$.*

Bi-Lipschitz MPNN. We now present our main contribution: a novel MPNN that is not only WL-equivalent, but also bi-Lipschitz, with respect to both ρ_{DS} and TMD.

FSW Embedding. The core innovation in our MPNN lies in its message aggregation method. Specifically, we employ the *Fourier Sliced-Wasserstein (FSW) Embedding*—a method for embedding multisets of vectors into Euclidean space, proposed by Amir and Dym [4], where it was also shown to be bi-Lipschitz. This property makes it plausible, a priori, that an MPNN based on FSW aggregation will be bi-Lipschitz for *graphs*. In this work, we formally establish that this is indeed the case. We begin by describing the FSW embedding and then introduce our FSW-GNN architecture.

The FSW embedding maps an input multiset $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$, to an output vector $\boldsymbol{z} = (z_1, \ldots, z_m) \in \mathbb{R}^m$. It is denoted by

$$\boldsymbol{z} = E_{\text{FSW}}(\boldsymbol{X}; (\boldsymbol{v}_k, \xi_k)_{k=1}^m), \quad \boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}.$$

In addition to the input multiset $\boldsymbol{X}$, it depends on parameters $(\boldsymbol{v}_k, \xi_k)_{k=1}^{m-1}$, where each $\boldsymbol{v}_k \in \mathcal{S}^{d-1}$ represents a direction vector and $\xi_k \in \mathbb{R}$ represents a frequency. The embedding is computed in three steps: First, the direction $\boldsymbol{v}_k$ is used to project the original multiset of vectors to a multiset of scalars $\{\langle \boldsymbol{v}_k, \boldsymbol{x}_i \rangle\}_{i=1}^n$, which is then sorted: $\boldsymbol{y}_k = \text{sort}(\{\langle \boldsymbol{v}_k, \boldsymbol{x}_i \rangle\}_{i=1}^n)$. This step is similar to the sort-type embedding used in SortMPNN, and was shown to be bi-Lipschitz on multisets of fixed size [8]. However, taking $\boldsymbol{y}_k$ directly as the embedding leads to an output dimension dependent on the input multiset's size, thus making the embedding unsuitable for varying-size multisets. The next steps address and resolve this limitation.

In the second step, the vector $\boldsymbol{y}_k$ is identified with a step function $Q_{\boldsymbol{y}_k} : [0, 1] \to \mathbb{R}$, namely the *quantile function* of the multiset $\{\langle \boldsymbol{v}_k, \boldsymbol{x}_i \rangle\}_{i=1}^n$; see illustration in fig. 2. Then, in the third step, the *cosine transform*, a variant of the Fourier transform, is applied to $Q_{\boldsymbol{y}_k}$, and sampled at the given frequency ξ_k, to obtain the final output coordinates z_k, $k = 1, \ldots, m-1$. This is summarized by:

$$\boldsymbol{y}_k = (y_{k1}, \ldots, y_{kn}) = \text{sort}(\langle \boldsymbol{v}_k, \boldsymbol{x}_1 \rangle, \ldots, \langle \boldsymbol{v}_k, \boldsymbol{x}_n \rangle) \tag{8}$$

$$Q_{\boldsymbol{y}_k}(t) = y_{ki} \text{ with } i \text{ such that } t \in [\tfrac{i-1}{n}, \tfrac{i}{n}) \tag{9}$$

$$z_k = 2(1 + \xi_k) \int_0^1 Q_{\boldsymbol{y}_k}(t) \cos(2\pi \xi_k t) dt \tag{10}$$

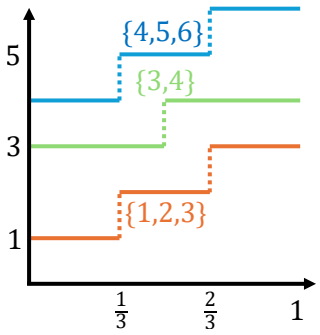

Figure 2: The quantile function for three different multisets

Lastly, note that eqs. (8) to (10) treat input multisets as uniform distributions over their elements, and thus are agnostic to the multiset size by design. To address this and ensure that multisets of different sizes but identical element-proportions are mapped to distinct outputs, the last output coordinate z_m is set to the cardinality $|\boldsymbol{X}|$ of the input multiset $\boldsymbol{X}$. Further details appear in [4, Appendix A.1].

Note that the integral in eq. (10) has a closed-form solution, and the whole embedding can be computed with a complexity of $\mathcal{O}(mnd + mn \log n)$, similarly to simple aggregation functions such as sum pooling. Furthermore, the embedding parameters and output dimension do not depend on n, making this method suitable for multisets of different sizes.

FSW-GNN. The FSW-GNN model processes input graphs $G = (V, E, X)$ by T message-passing iterations:

$$\begin{aligned} \boldsymbol{h}_v^{(0)} &:= \boldsymbol{x}_v, \\ \boldsymbol{q}_v^{(t)} &:= E_{\text{FSW}}^{(t)}\left(\left\{\boldsymbol{h}_u^{(t-1)} \;\middle|\; u \in \mathcal{N}_v\right\}\right), \quad 1 \le t \le T, \\ \boldsymbol{h}_v^{(t)} &:= \Phi^{(t)}\left(\left[\boldsymbol{h}_v^{(t-1)}; \boldsymbol{q}_v^{(t)}\right]\right), \end{aligned} \tag{11}$$

where the functions $E_{\text{FSW}}^{(t)}$ are all instances of the FSW embedding, $\Phi^{(t)}$ are MLPs, and $[\boldsymbol{x}; \boldsymbol{y}]$ denotes column-wise concatenation of column vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. Finally, a graph-level output is computed by:

$$\boldsymbol{h}_G := \Psi \circ E_{\text{FSW}}^{\text{Glob}}\left(\left\{\boldsymbol{h}_v^{(T)} \;\middle|\; v \in V\right\}\right), \tag{12}$$

where, again, $E^{\text{Glob}}_{\text{FSW}}$ is an FSW embedding, and Ψ an MLP.

The following theorem shows that, with the appropriate choice of MLP sizes and number of iterations T, our proposed architecture is WL equivalent:

Theorem 3.2. (Proof in appendix D.2) *Consider the FSW-GNN architecture for input graphs in $\mathcal{G}_{\leq N}(\mathbb{R}^d)$, with $T = N$ iterations, where $\Phi^{(t)}, \Psi$ are just linear funtions, and all features (except for input features) are of dimension $m \geq 2Nd + 2$. Then for Lebesgue almost every choice of model parameters, the graph embedding defined by the architecture is WL equivalent.*

The proof of theorem 3.2 is based on the theory of σ-subanalytic functions and the *Finite Witness Theorem*, introduced in [5].

It is worth noting that the output dimension m required in practice is typically considerably lower than the one required in theorem 3.2. This can be explained intuitively by the following fact: if all input graphs originate from a subset of $\mathcal{G}_{\leq N}(\mathbb{R}^d)$ with intrinsic dimension D that is lower than the ambient dimension $n \cdot d$, then it can be shown that $m = 2D + 2$ suffices for WL-equivalence.

From separation to bi-Lipschitzness. In general, WL-equivalence does not imply bi-Lipschitzness. As mentioned above, sum-based MPNN can be injective but are never bi-Lipschitz. In contrast, we shall now prove that for FSW-GNN, WL-equivalence does imply bi-Lipschitzness, under the assumption that the feature domain Ω is compact:

Theorem 3.3. (Proof in appendix D.3) *Let $\Omega \subset \mathbb{R}^d$ be compact. Under the assumptions of theorem 3.2, the FSW-GNN is bi-Lipschitz with respect to ρ_{DS} on $\mathcal{G}_{\leq N}(\Omega)$. If, additionally, Ω does not contain $\mathbf{0}$, then the FSW-GNN is bi-Lipschitz with respect to TMD on $\mathcal{G}_{\leq N}(\Omega)$.*

We now give a high-level explanation of the proof idea. We rely on the following facts: (1) the output of FSW-GNN for an input graph $G = (V, E, X)$ is piecewise-linear with respect to the vertex-feature matrix X. This follows from properties of the FSW embedding functions $E^{(t)}_{\text{FSW}}$ and $E^{\text{Glob}}_{\text{FSW}}$ used in eqs. (11) and (12). (2) both metrics ρ_{DS} and TMD can be transformed, with bounded distortion, into metrics that are piecewise-linear, by choosing all the vector norms they employ to be the ℓ_1 norm. The claim then follows from these observations and the following lemma, which shows that two functions that are piecewise linear and have the same zero set, are bi-Lipschitz with respect to one another:

Lemma 3.4. (Proof in appendix D.3) *Let $f, g : M \to \mathbb{R}_{\geq 0}$ be nonnegative piecewise-linear functions defined on a compact polygon $M \subset \mathbb{R}^d$. Suppose that for all $x \in M$, $f(x) = 0$ if and only if $g(x) = 0$. Then there exist real constants $c, C > 0$ such that*

$$c \cdot g(x) \leq f(x) \leq C \cdot g(x), \quad \forall x \in M. \tag{13}$$

We note that the assumption of a compact domain is essential; see Remark 6 in [10]. While lemma 3.4 is rather intuitive, we are not aware of it appearing previously in the literature. This lemma easily implies[1] previous bi-Lipschitzness results in the literature, such as Theorem 1.4 in [9], Theorem 1 in [10], and Theorem 3.10 in [8]. We believe the lemma has the potential to serve as a valuable tool for proving bi-Lipschitzness results in additional domains in the future. To illustrate this, in Appendix F we show how the lemma can be used to construct a bi-Lipschitz *subgraph aggregation MPNN*.

To complement theorem 3.3, we now present an analogous result for node-level tasks: we show that the node features computed by FSW-GNN are bi-Lipschitz with respect to the *Tree Distance* discussed in eq. (4).

Theorem 3.5. (Proof in appendix D.3) *Suppose that $\Omega \subset \mathbb{R}^d$ is a compact set that does not contain zero. Under the assumptions of theorem 3.2, the node features computed by FSW-GNN are bi-Lipschitz with respect to the Tree Distance metric on $\mathcal{G}_{\leq N}(\Omega)$.*

4 Oversmoothing, Oversquashing, and Bi-Lipschitz MPNN

Training *deep* MPNNs is one of the core challenges in graph neural networks [51]. The difficulty is often attributed to *oversmoothing* [58] or *oversquashing* [3]. Both phenomena can be regarded as a

[1] In all these scenarios, the proposed embedding and metric are *homogeneous*, so that bi-Lipschitzness on a compact polygon implies global bi-Lipschitzness.

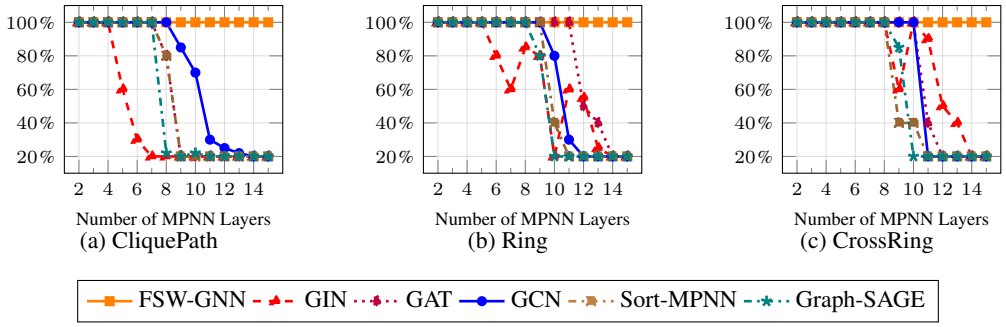

Figure 3: Performance comparison of MPNN models across the CliquePath, Ring, and CrossRing graph transfer tasks as presented in [27].

form of metric distortion induced by the MPNN as the number of iterations grows. Oversmoothing is the phenomenon where, for large t,

$$\left\| \boldsymbol{x}_v^{(t)} - \boldsymbol{x}_u^{(t)} \right\| \approx 0, \quad \forall v, u \in V. \tag{14}$$

Oversquashing, as defined in [27], refers the phenomenon where a non-negligible change in an input node feature $\boldsymbol{x}_v^{(0)}$ to a new value $\hat{\boldsymbol{x}}_v^{(0)}$ results in only a negligible change in features of far-away nodes, namely,

$$\left\| \boldsymbol{x}_u^{(t)} - \hat{\boldsymbol{x}}_u^{(t)} \right\| \ll \left\| \boldsymbol{x}_v^{(0)} - \hat{\boldsymbol{x}}_v^{(0)} \right\|, \text{ for } u \text{ far from } v. \tag{15}$$

In other words, information fails to propagate adequately between distant nodes through the message-passing process.

Accordingly, an MPNN that perfectly preserves distances between node features would provably not encounter oversmoothing or oversquashing. We note that FSW-GNN is not in this position: while it is bi-Lipschitz and thus has bounded distortion, we cannot rule out the possibility that its distortion grows with t. Nonetheless, we conjecture that FSW-GNN should perform much better than standard MPNN on long range tasks. This is because, as shown in [26], standard MPNNs are only lower Hölder in expectation, namely, the inequality

$$\|x_v^{(t)} - x_u^{(t)}\| \geq C_t \left[\mathrm{TD}(\mathbf{T}_v^{(t)}, \mathbf{T}_u^{(t)}) \right]^{\alpha_t}$$

holds only in expectation, with constants C_t and exponents α_t that grow with t. In contrast, for FSW-GNN we have $\alpha_t = 1$ for all t, and only C_t may grow with t. We provide empirical evidence for our conjecture in section 5, where we show that our bi-Lipschitz MPNN outperforms standard MPNNs on long-range tasks.

5 Numerical Experiments

We compare the performance of FSW-GNN with standard MPNNs and Sort-MPNN on both real-world benchmarks and synthetic long-range tasks[2].

Empirical distortion evaluation. First, to assess the metric distortion induced by different MPNNs, we compared the distances induced by each MPNN vs. the TMD and DS metric on a particularly challenging set of graph pairs; see appendix E for details. Empirical estimates $\hat{C}, \hat{c}$ of the constants C,c of eq. (5) were computed, and the distortion estimate was taken as the ratio $\hat{C}/\hat{c}$. The results appear in fig. 4. Similar results were obtained for the TMD; see fig. 6 in the appendix. As seen in the figures, our method yields considerably lower distortion than all competitors, which aligns with our theoretical guarantees.

[2]Our code will be available to the public upon paper acceptance.

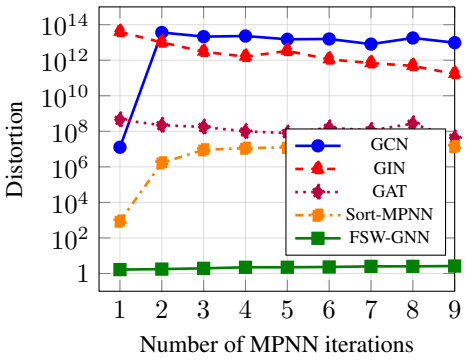

Figure 4: Empirical distortion evaluation with respect to the Doubly-Stochastic (DS) metric

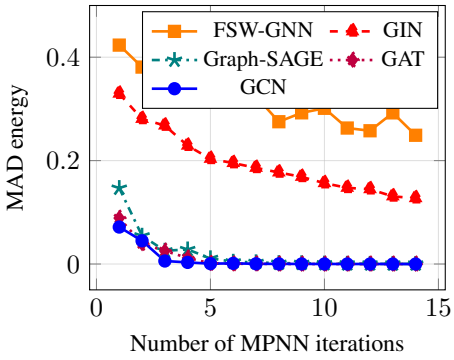

Figure 5: MAD Energy vs. Number of MPNN layers for various models on the Ring long-range task

Transductive Learning. Next, we compare FSW-GNN with GCN, GAT and Sort-MPNN for nine transductive learning tasks taken from [55]. As shown in table 1, FSW-GNN outperforms the competition in six out of nine tasks. SortMPNN is a clear winner in two of the other graphs, both of which have relatively large average degrees (see Table 4). In addition, we include the state-of-the-art results known to us for each dataset: For Cora [42]; for Citeseer [43]; for PubMed [44]; for Chameleon [57]; for Squirrel [46]; for Actor [40]; for Cornell [32]; for Texas [49]; for Wisconsin [41]. These results are typically achieved by models that are strictly more powerful and computationally expensive than any MPNN.

Graph classification and regression. In the appendix, we we include results on standard graph classication and regression tasks: the peptides-func and peptides-struct datasets of the LRGB benchmark [28], and he MUTAG and Protein datasets [52]. These results appear in table 2 and table 3.

Long range tasks. Next, we consider several synthetic long-range tasks proposed in the oversquashing literature [3, 27]. In these problems, one can control the problem radius r, which is the number of MPNN iterations needed to solve the problem.

We first consider the NeighborsMatch problem from [3]. In fig. 1, we compare the performance of FSW-GNN with standard MPNN on the NeigborsMatch problem with $2 \leq r \leq 8$. Our FSW-GNN achieves perfect accuracy for all values of r, while Sort-MPNN fails at $r = 8$ and the other competing methods falter for $r \geq 6$.

Next, we consider the 'graph transfer' tasks [27], in which a 'source' node feature is propagated to a 'target' node that is r steps away. All other nodes have 'blank' features. Here we used three graph topologies proposed by [27]: clique, ring, and crossring, with problem radii r varying from 2 to 15.

As shown in fig. 3, FSW-GNN is the only method to achieve 100% accuracy across all three graphs and radii. Other models start failing at much smaller r, which depends on the graph topology. We do find that our model's performance deteriorates when $r > 20$.

Finally, we evaluate the relation between our empirical results and oversmoothing on the ring experiment, using the Mean Average Distance (MAD) [58]. As seen in fig. 5, all methods except for FSW-GNN and GIN exhibit oversmoothing starting from some r.

We note that long-range issues can be alleviated by graph rewiring methods, which effectively reduce problem radii [37], or by adding global information using spectral filters [33] or graph transformers. A key distinction of FSW-GNN is that it performs well *without modifying* the original graph topology. Thus, this work directly addresses the problem of relaying long-range messages, rather than circumventing it by architectural modifications.

6 Conclusion

In this paper, we introduced FSW-GNN, the first bi-Lipschitz MPNN. Empirically, we found that FSW-GNN is highly effective, particularly for long-range problems, and appears to mitigate oversmoothing and oversquashing due to its inherent ability to preserve graph metrics.

A slight drawback of FSW-GNN is that its runtime is somewhat higher than that of standard MPNNs; see table 6. An interesting direction for future work is to obtain quantitative estimates of its bi-Lipschitz constants.

References

[1] Anders Aamand et al. "Exponentially Improving the Complexity of Simulating the Weisfeiler-Lehman Test with Graph Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. 2022. URL: https://openreview.net/forum?id=AyGJDpN2eR6. 1, 4

[2] Yonathan Aflalo, Alexander Bronstein, and Ron Kimmel. "On convex relaxation of graph isomorphism". In: *Proceedings of the National Academy of Sciences* 112.10 (2015), pp. 2942–2947. DOI: 10.1073/pnas.1401651112. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.1401651112. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1401651112. 5

[3] Uri Alon and Eran Yahav. "On the Bottleneck of Graph Neural Networks and its Practical Implications". In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2021. URL: https://openreview.net/forum?id=i80OPhOCVH2. 2, 7, 9, 23

[4] Tal Amir and Nadav Dym. *Fourier Sliced-Wasserstein Embedding for Multisets and Measures*. 2024. arXiv: 2405.16519 [cs.LG]. URL: https://arxiv.org/abs/2405.16519. 2, 6, 21

[5] Tal Amir et al. "Neural Injective Functions for Multisets, Measures and Graphs via a Finite Witness Theorem". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=TQlpqmCeMe. 1, 2, 4, 7, 21

[6] László Babai. "Graph isomorphism in quasipolynomial time". In: *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*. 2016, pp. 684–697. 3

[7] Radu Balan. "Stability theorems for Fourier frames and wavelet Riesz bases". In: *Journal of Fourier Analysis and applications* 3 (1997), pp. 499–504. 2

[8] Radu Balan, Naveed Haghani, and Maneesh Singh. "Permutation invariant representations with applications to graph deep learning". In: *arXiv preprint arXiv:2203.07546* (2022). 2, 6, 7

[9] Radu Balan and Efstratios Tsoukanis. *G-Invariant Representations using Coorbits: Bi-Lipschitz Properties*. 2023. arXiv: 2308.11784 [math.RT]. 7

[10] Radu Balan, Efstratios Tsoukanis, and Matthias Wellershoff. "Stability of sorting based embeddings". In: *arXiv preprint arXiv:2410.05446* (2024). 7

[11] Afonso S Bandeira et al. "Saving phase: Injectivity and stability for phase retrieval". In: *Applied and Computational Harmonic Analysis* 37.1 (2014), pp. 106–125. 2

[12] Jose Bento and Stratis Ioannidis. "A family of tractable graph distances". In: *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM. 2018, pp. 333–341. 5

[13] Florian Bernard, Christian Theobalt, and Michael Moeller. "DS*: Tighter Lifting-Free Convex Relaxations for Quadratic Matching Problems". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018. 5

[14] Beatrice Bevilacqua et al. "Equivariant subgraph aggregation networks". In: *arXiv preprint arXiv:2110.02910* (2021). 23

[15] Jan Böker. "Graph Similarity and Homomorphism Densities". In: *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik. 2021. 3, 5

[16] Jan Böker et al. "Fine-grained expressivity of graph neural networks". In: *Advances in Neural Information Processing Systems* 36 (2024). 3

[17] César Bravo, Alexander Kozachinskiy, and Cristóbal Rojas. "On dimensionality of feature vectors in MPNNs". In: *arXiv preprint arXiv:2402.03966* (2024). 4

[18] César Bravo, Alexander Kozachinskiy, and Cristobal Rojas. "On dimensionality of feature vectors in MPNNs". In: *Proceedings of the 41st International Conference on Machine Learning*. Ed. by Ruslan Salakhutdinov et al. Vol. 235. Proceedings of Machine Learning Research. PMLR, 21–27 Jul 2024, pp. 4472–4481. URL: https://proceedings.mlr.press/v235/bravo24a.html. 1

[19] Jameson Cahill, Andres Contreras, and Andres Contreras-Hip. "Complete set of translation invariant measurements with Lipschitz bounds". In: *Applied and Computational Harmonic Analysis* 49.2 (2020), pp. 521–539. 2

[20] Jameson Cahill, Joseph W. Iverson, and Dustin G. Mixon. *Towards a bilipschitz invariant theory*. 2024. arXiv: 2305.17241 [math.FA]. 2

[21] Samantha Chen et al. "Weisfeiler-Lehman Meets Gromov-Wasserstein". In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 17–23 Jul 2022, pp. 3371–3416. URL: https://proceedings.mlr.press/v162/chen22o.html. 3

[22] Zhengdao Chen et al. "On the equivalence between graph isomorphism testing and function approximation with gnns". In: *Advances in neural information processing systems* 32 (2019). 4

[23] Cheng Cheng et al. "Stable phase retrieval from locally stable and conditionally connected measurements". In: *Applied and Computational Harmonic Analysis* 55 (2021), pp. 440–465. 2

[24] Ching-Yao Chuang and Stefanie Jegelka. "Tree Mover's Distance: Bridging Graph Metrics and Stability of Graph Neural Networks". In: *Advances in Neural Information Processing Systems*. 2022. URL: https://openreview.net/forum?id=Qh89hwiP5ZR. 2–5, 16, 17

[25] Gabriele Corso et al. "Principal neighbourhood aggregation for graph nets". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 13260–13271. 2

[26] Yair Davidson and Nadav Dym. "On the H\"{o} lder Stability of Multiset and Graph Neural Networks". In: *arXiv preprint arXiv:2406.06984* (2024). 2, 4, 8

[27] Francesco Di Giovanni et al. "On over-squashing in message passing neural networks: The impact of width, depth, and topology". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 7865–7885. 8, 9, 23

[28] Vijay Prakash Dwivedi et al. "Long Range Graph Benchmark". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 22326–22340. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf. 9

[29] Nadav Dym. "Exact recovery with symmetries for the doubly stochastic relaxation". In: *SIAM Journal on Applied Algebra and Geometry* 2.3 (2018), pp. 462–488. 5

[30] Nadav Dym and Steven J. Gortler. *Low Dimensional Invariant Embeddings for Universal Geometric Learning*. 2023. arXiv: 2205.02956 [cs.LG]. 21

[31] Nadav Dym, Haggai Maron, and Yaron Lipman. "DS++ a flexible, scalable and provably tight relaxation for matching problems". In: *ACM Transactions on Graphics (TOG)* 36.6 (2017), pp. 1–14. 5

[32] Moshe Eliasof, Eldad Haber, and Eran Treister. "Graph Neural Reaction Diffusion Models". In: *SIAM Journal on Scientific Computing* 46.4 (2024), pp. C399–C420. 9

[33] Simon Geisler et al. "Spatio-Spectral Graph Neural Networks". In: *arXiv preprint arXiv:2405.19121* (2024). 9

[34] Justin Gilmer et al. "Neural Message Passing for Quantum Chemistry". In: *International Conference on Machine Learning*. 2017. URL: https://api.semanticscholar.org/CorpusID:9665943. 1

[35] Martin Grohe. "The logic of graph neural networks". In: *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE. 2021, pp. 1–17. 3, 5

[36] Martin Grohe. "word2vec, node2vec, graph2vec, X2vec: Towards a Theory of Vector Embeddings of Structured Data". In: *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. PODS'20. Portland, OR, USA: Association for Computing Machinery, 2020, pp. 1–16. ISBN: 9781450371087. DOI: 10.1145/3375395.3387641. URL: https://doi.org/10.1145/3375395.3387641. 2, 3, 5

[37] Benjamin Gutteridge et al. "Drew: Dynamically rewired message passing with delay". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 12252–12267. 9

[38] Will Hamilton, Zhitao Ying, and Jure Leskovec. "Inductive Representation Learning on Large Graphs". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf. 4

[39] Snir Hordan et al. "Complete Neural Networks for Complete Euclidean Graphs". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 11. 2024, pp. 12482–12490. 21

[40] Yiming Huang et al. "Higher-order graph convolutional network with flower-petals laplacians on simplicial complexes". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 11. 2024, pp. 12653–12661. 9

[41] Yiming Huang et al. "Higher-order graph convolutional network with flower-petals laplacians on simplicial complexes". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 11. 2024, pp. 12653–12661. 9

[42] Mohammad Rasool Izadi et al. "Optimization of graph neural networks with natural gradient descent". In: *2020 IEEE international conference on big data (big data)*. IEEE. 2020, pp. 171–179. 9

[43] Mohammad Rasool Izadi et al. "Optimization of graph neural networks with natural gradient descent". In: *2020 IEEE international conference on big data (big data)*. IEEE. 2020, pp. 171–179. 9

[44] Mohammad Rasool Izadi et al. "Optimization of graph neural networks with natural gradient descent". In: *2020 IEEE international conference on big data (big data)*. IEEE. 2020, pp. 171–179. 9

[45] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=SJU4ayYgl. 4

[46] Christian Koke and Daniel Cremers. "HoloNets: Spectral Convolutions do extend to Directed Graphs". In: *arXiv preprint arXiv:2310.02232* (2023). 9

[47] Soheil Kolouri et al. "Wasserstein Embedding for Graph Learning". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=AAes_3W-2z. 2

[48] Ron Levie. "A graphon-signal analysis of graph neural networks". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=J0RD92Tmfc. 2

[49] Sitao Luan et al. "Revisiting heterophily for graph neural networks". In: *Advances in neural information processing systems* 35 (2022), pp. 1362–1375. 9

[50] Vince Lyzinski et al. "Graph Matching: Relax at Your Own Risk". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.1 (2016), pp. 60–73. DOI: 10.1109/TPAMI.2015.2424894. 5

[51] Christopher Morris et al. *Future Directions in Foundations of Graph Machine Learning*. 2024. arXiv: 2402.02287 [cs.LG]. 2, 7

[52] Christopher Morris et al. "Tudataset: A collection of benchmark datasets for learning with graphs". In: *arXiv preprint arXiv:2007.08663* (2020). 9

[53] Christopher Morris et al. "Weisfeiler and leman go machine learning: The story so far". In: *The Journal of Machine Learning Research* 24.1 (2023), pp. 15865–15923. 3

[54] Christopher Morris et al. "Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks". In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019, pp. 4602–4609. DOI: 10.1609/AAAI.V33I01.33014602. URL: https://doi.org/10.1609/aaai.v33i01.33014602. 1, 4

[55] Hongbin Pei et al. "Geom-gcn: Geometric graph convolutional networks". In: *arXiv preprint arXiv:2002.05287* (2020). 9

[56] Chendi Qian et al. "Ordered subgraph aggregation networks". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 21030–21045. 23

[57] Emanuele Rossi et al. "Edge directionality improves learning on heterophilic graphs". In: *Learning on Graphs Conference*. PMLR. 2024, pp. 25–1. 9

[58] T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. "A Survey on Oversmoothing in Graph Neural Networks". In: *ArXiv* abs/2303.10993 (2023). URL: https://api.semanticscholar.org/CorpusID:257632346. 7, 9

[59] Edward R Scheinerman and Daniel H Ullman. *Fractional graph theory: a rational approach to the theory of graphs*. Courier Corporation, 2013. 5, 19, 20

[60] Petar Velickovic et al. "Graph Attention Networks". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL: https://openreview.net/forum?id=rJXMpikCZ. 4

[61] Keyulu Xu et al. "How Powerful are Graph Neural Networks?" In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=ryGs6iA5Km. 1, 4

[62] Keyulu Xu et al. "How powerful are graph neural networks?" In: *arXiv preprint arXiv:1810.00826* (2018). 15, 21

[63] Muhan Zhang et al. "An End-to-End Deep Learning Architecture for Graph Classification". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. Ed. by Sheila A. McIlraith and Kilian Q. Weinberger. AAAI Press, 2018, pp. 4438–4445. DOI: 10.1609/AAAI.V32I1.11782. URL: https://doi.org/10.1609/aaai.v32i1.11782. 2

[64] Markus Zopf. "1-wl expressiveness is (almost) all you need". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2022, pp. 1–8. 3

Table 2: Performance of different models on the LRGB results.

Dataset	peptides-func (AP↑)	peptides-struct (MAE↓)
GINE	0.6621±0.0067	0.2473±0.0017
GCN	0.6860±0.0050	**0.2460±0.0007**
GatedGCN	0.6765±0.0047	0.2477±0.0009
SortMPNN	**0.6914±0.0056**	0.2494±0.0021
FSW-GNN	0.6864±0.0048	0.2489±0.00155
SOTA	**0.73**	**0.242**

Table 3: Performance of different models on the MUTAG and Protein datasets.

Model	**MUTAG**	**Protein**
GIN [62]	89.4 ± 5.6	76.2 ± 2.8
GCN	85.6 ± 5.8	76.0 ± 3.2
GraphSage	85.1 ± 7.6	75.9 ± 3.2
SortMPNN	$\mathbf{90.99 \pm 6.2}$	76.46 ± 3.68
AdaptMPNN	90.41 ± 6.1	75.12 ± 3.64
FSW-GNN	90.55 ± 6.1	$\mathbf{76.93 \pm 7.64}$
SOTA	96.66 ± 1.23	$\mathbf{84.91 \pm 1.62}$

A Statistics on our benchmarks

We present some statistics of the learning problems considered in the main text. In Table 4, we present the statistics for each transductive dataset, and for the MUTAG dataset, including the number of nodes, edges, features, classes, average degree, and density, measuring the number of edges divided by the number of maximal edges. As we can see in the table, these datasets are very sparse.

Table 5 shows the same statistics for the Peptides tasks from the LRGB dataset.

Table 4: Graph statistics for transductive learning and MUTAG.

Dataset	**Cora**	**Cite.**	**Pubm.**	**Cham.**	**Squi.**	**Actor**	**Corn.**	**Texa.**	**Wisc.**	**MUTAG**
# Nodes	2708	3327	19717	2277	5201	7600	183	183	251	188
# Edges	5429	4732	44338	36101	217073	33544	295	309	499	744
# Features	1433	3703	500	2325	2089	931	1703	1703	1703	7
# Classes	7	6	3	5	5	5	5	5	5	2
Avg. Degree	4	2	4	31	83	8	3	3	4	8
Density	0.0007	0.0009	0.0001	0.0159	0.0161	0.0012	0.0177	0.0184	0.0041	0.0422

Table 5: Graph statistics for Peptides datasets.

Dataset	**Peptides-Func**	**Peptides-Struct**
# Graphs	15,535	15,535
# Nodes (Avg.)	150.94	150.94
# Edges (Avg.)	2.04	2.04
Avg. Degree	2.04	2.04
Density	1.74×10^{-6}	1.74×10^{-6}
# Classes	-	10

B Implantation details

For the Transductive learning and the LRGB dataset, we looked for each learning rate in the set $1e^{-3}, 5e^{-3}$ and weights decay in the set $0.0, 1e^{-5}$ and learning rate factor of 0.7. For each such configuration we stop according to the validation and take the best configuration of the validation set.

For all Over-squashing tasks we took a learning rate of 0.001 and weight decay 0, and learning rate factor of 0.7. In all datasets we took the original dataset splits to train, test and validation sets.

C Relation to Tree Mover's Distance

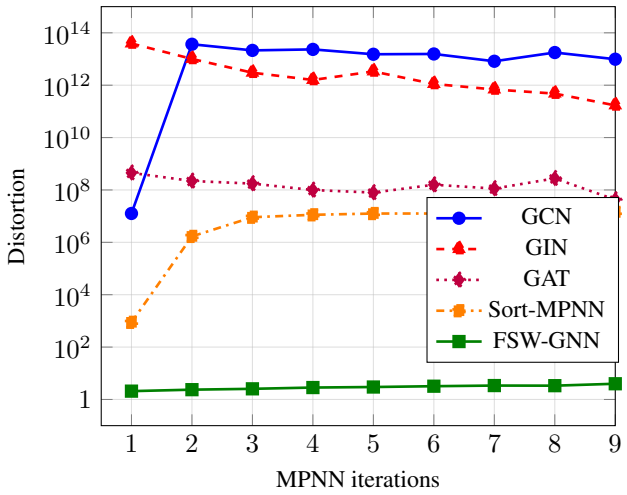

Figure 6: Empirical distortion evaluation with respect to the Tree Mover's Distance (TMD)

In this appendix we give a full description of the Tree Mover's Distance as defined in Chuang and Jegelka [24].

We first review Wasserstein distances. Recall that if (X, d) is a metric space, $\Omega \subseteq X$ is a subset, then the Wasserstein distance can be defined on the space of multisets consisting of n elements in Ω via

$$W_d(x_1, \ldots, x_n, y_1, \ldots, y_n) = \min_{\tau \in S_n} \sum_{j=1}^{n} d(x_j, y_{\tau(j)})$$

For multisets of different size, the authors of [24] used an augmentation map, which, for a fixed parameter n, augments multisets of size $r \leq n$ by padding with $n - r$ instances of an element z in $X \setminus \Omega$, namely

$$\Gamma^z(x_1, \ldots, x_r) = (x_1, \ldots, x_r, x_{r+1} = z, \ldots x_n = z)$$

and the augmented distance on multi-sets of size up to n is defined by

$$W_d(X, \hat{X}) = W_d^0\Big(\Gamma^z(X), \Gamma^z\Big(\hat{X}\Big)\Big).$$

We now return to define the TMD. We consider the space of graphs $\mathcal{G}_{\leq N}(\Omega)$, consisting of graphs with $\leq N$ nodes, with node features coming from a compact domain $\Omega \subseteq \mathbb{R}^d$ such that $0 \notin \Omega$. The TMD is defined using the notion of computation trees:

Definition C.1. (Computation Trees). Given a graph $\mathcal{G} = (V, E, X)$ with node features $\{x_v\}_{v \in V}$, let $\mathbf{T}_v^1$ be the rooted tree with a single node v, which is also the root of the tree, and node features x_v. For $t \in \mathbb{N}$ let $\mathbf{T}_v^t$ be the depth-t computation tree of node v constructed by connecting the neighbors of the leaf nodes of $\mathbf{T}_v^{t-1}$ to the tree. Each node is assigned the same node feature it had in the original graph $\mathcal{G}$. The multiset of depth-t computation trees defined by $\mathcal{G}$ is denoted by $\mathcal{T}_G^K := \{\mathbf{T}_v^t\}_{v \in V}$. Additionally, for a tree $\mathbf{T}$ with root r, we denote by $\mathcal{T}_r$ the multiset of subtrees that root at the descendants of r.

Definition C.2. (Blank Tree). A blank tree $\bar{\mathbf{T}}_0$ is a tree (graph) that contains a single node and no edge, where the node feature is the zero vector 0.

Recall that by assumption, all node features will come from the compact set Ω, and $0 \notin \Omega$. Therefore, the blank tree is not included in the space of trees with features in Ω, and can be used for augmentation.

We can now define the tree distance:

Definition C.3. (Tree Distance).[3] The distance between two trees $\mathbf{T}_a, \mathbf{T}_b$ with features from Ω and $0 \notin \Omega$, is defined recursively as

$$\mathrm{TD}(\mathbf{T}_a, \mathbf{T}_b) := \begin{cases} \|x_{r_a} - x_{r_b}\|_1 + W_{\mathrm{TD}}^{\bar{\mathbf{T}}_0}(\mathcal{T}_{r_a}, \mathcal{T}_{r_b}) & \text{if } t > 1 \\ \|x_{r_a} - x_{r_b}\|_1 & \text{otherwise} \end{cases}$$

where t denotes the maximal depth of the trees $\mathbf{T}_a$ and $\mathbf{T}_b$. Here $W_{\mathrm{TD}}^{\bar{\mathbf{T}}_0}$ denotes the Wasserstein metric obtained from the metric TD on the space of trees of smaller depth, with augmentation by blank trees $\bar{\mathbf{T}}_0$.

Definition C.4. (Tree Mover's Distance). Given two graphs, G_a, G_b and $w, t \geq 0$, the tree mover's distance is defined by

$$\mathrm{TMD}^t(G_a, G_b) = W_{\mathrm{TD}}^{\bar{\mathbf{T}}_0}(\mathcal{T}_{G_a}^t, \mathcal{T}_{G_b}^t),$$

where $\mathcal{T}_{G_a}^t$ and $\mathcal{T}_{G_b}^t$ denote the multiset of all depth t computational trees arising from the graphs G_a and G_b, respectively. Chuang and Jegelka [24] proved that $\mathrm{TMD}^t(G_a, G_b)$ is a pseudo-metric that fails to distinguish only graphs that cannot be separated by $t+1$ iterations of the WL test. Thus, assuming that $0 \notin \Omega$, $\mathrm{TMD}^t(G_a, G_b)$ is WL equivalent on $\mathcal{G}_{\leq N}(\Omega)$.

In addition, it is easy to see from the definition of TMD that it satisfies the following properties:

1. $\mathrm{TMD}^t((\mathbf{A}, \alpha \cdot \mathbf{X}), (\mathbf{B}, \alpha \cdot \mathbf{Y})) = \alpha \cdot \mathrm{TMD}^t(\mathbf{X}, \mathbf{Y})$ for any $\alpha \geq 0$.
2. For fixed $\mathbf{A}, \mathbf{B}$, the TMD metric is piecewise linear in $(\mathbf{X}, \mathbf{Y})$.

These properties will be used to show that under the above assumptions, the embedding computed by FSW-GNN is bi-Lipschitz with respect to TMD.

D Proofs

D.1 DS metric

Here we prove theorem 3.1

Theorem 3.1. *ρ_{DS} is a WL-equivalent metric on $\mathcal{G}_{\leq N}(\mathbb{R}^d)$.*

Proof. ρ_{DS} is symmetric, since if a pair's minimum is obtained at S then the exact value is obtained for the opposite pair with S^T and vice-versa.

To prove the triangle inequality, let Let $(V_1, \mathbf{A}, \mathbf{X}), (V_2, \mathbf{B}, \mathbf{Y}), (V_3, \mathbf{C}, \mathbf{Z})$ three arbitrary graphs with $|V_i| = n_i$, $i = 1, 2, 3$. Then the following holds:

$$\alpha := \rho_{\mathrm{DS}}(\mathbf{A}, \mathbf{B}) = |n_1 - n_2| + min_{S \in \Pi(n_1, n_2)} ||\mathbf{A}S - S\mathbf{B}||_1 + \sum_{i,j} S_{i,j} ||\mathbf{X}_i - \mathbf{Y}_j||_1$$

$$\beta := \rho_{\mathrm{DS}}(\mathbf{B}, \mathbf{C}) = |n_2 - n_3| + min_{S \in \Pi(n_2, n_3)} ||\mathbf{B}S - S\mathbf{C}||_1 + \sum_{i,j} S_{i,j} ||\mathbf{Y}_i - \mathbf{Z}_j||_1$$

$$\gamma := \rho_{\mathrm{DS}}(\mathbf{A}, \mathbf{C}) = |n_1 - n_3| + min_{S \in \Pi(n_1, n_3)} ||\mathbf{A}S - S\mathbf{C}||_1 + \sum_{i,j} S_{i,j} ||\mathbf{X}_i - \mathbf{Z}_j||_1$$

We want to prove $\gamma \leq \beta + \alpha$. Let S^1, S^2 be the minimizers of the first two equations. Define $S^3 = n_2 \cdot S^1 \cdot S^2$ and note that $S^3 \in \Pi(n_1, n_3)$ because, using the notation 1_d to the all one d-dimensional vector, we have

$$S^3 1_{n_3} = n_2 \cdot S^1 \cdot S^2 1_{n_3} = S^1 1_{n_2} = \frac{1}{n_1} 1_{n_1}$$

$$1_{n_1}^T S^3 = n_2 \cdot 1_{n_1}^T S_1 \cdot S_2 = 1_{n_2}^T S_2 = \frac{1}{n_3} 1_{n_3}.$$

Next, we prove the following lemma (as usual, norms in this lemma are Frobenius norms)

[3] This definition slightly varies from from the original definition in [24], due to our choice to set the depth weight to 1 and using the 1-Wasserstein which is equivalent to optimal transport.

Lemma D.1. *Let $S \in \Pi(n_1, n_2)$, then for all $B \in \mathbb{R}^{n_2 \times b}$ and $A \in \mathbb{R}^{a \times n_1}$*

$$\|SB\|_1 \leq \frac{1}{n_2}\|B\|_1, \quad \|AS\|_1 \leq \frac{1}{n_1}\|A\|_1$$

Proof.

$$\begin{aligned}
\|SB\|_1 &= \sum_{i=1}^{n_1}\sum_{j=1}^{b}|(SB)_{ij}| \\
&\leq \sum_{i=1}^{n_1}\sum_{j=1}^{b}\sum_{k=1}^{n_2} S_{ik}|B_{kj}| \\
&= \sum_{j=1}^{b}\sum_{k=1}^{n_2}|B_{kj}| \cdot \sum_{i=1}^{n_1} S_{ik} \\
&= \frac{1}{n_2}\|B\|_1
\end{aligned}$$

A symmetric argument can be used to show that $\|AS\|_1 \leq \frac{1}{n_1}\|A\|_1$ □

Given the lemma, we now have

$$\begin{aligned}
\frac{1}{n_2}||\mathbf{A}S^3 - S^3\mathbf{C}||_1 &= ||\mathbf{A}S^1S^2 - S^1S^2\mathbf{C}||_1 = ||\mathbf{A}S^1S^2 - S^1\mathbf{B}S^2 + S^1\mathbf{B}S^2 - S^1S^2\mathbf{C}||_1 \\
&\leq ||\mathbf{A}S^1S^2 - S^1\mathbf{B}S^2||_1 + ||S^1\mathbf{B}S^2 - S^1S^2\mathbf{C}||_1 = ||(\mathbf{A}S^1 - S^1\mathbf{B}) \cdot S^2||_1 \\
&+ ||S^1 \cdot (\mathbf{B}S^2 - S^2\mathbf{C})||_1 \leq \frac{1}{n_2}||\mathbf{A}S^1 - S^1\mathbf{B}||_1 + \frac{1}{n_2} \cdot ||\mathbf{B}S^2 - S^2\mathbf{C}||_1
\end{aligned}$$

Next, we show the second part is also smaller:

$$\begin{aligned}
\frac{1}{n_2}\sum_{i,j} S^3_{i,j} \cdot |\mathbf{X}_i - \mathbf{Y}_j|_1 &= \sum_{i,j}\sum_{k} S^1_{i,k} \cdot S^2_{k,j} \cdot ||\mathbf{X}_i - \mathbf{Z}_j||_1 \\
&\leq \sum_{i,j}\sum_{k} S^1_{i,k} \cdot S^2_{k,j} \cdot (||\mathbf{X}_i - \mathbf{Z}_k||_1 + ||\mathbf{Z}_k - \mathbf{Y}_j||_1) \\
&= \sum_{i,j}\sum_{k=1}^{n} S^1_{i,k} \cdot S^2_{k,j} \cdot ||\mathbf{X}_i - \mathbf{Z}_k||_1 + \sum_{i,j}\sum_{k=1}^{n} S^1_{i,k} \cdot S^2_{k,j} \cdot ||\mathbf{Y}_j - \mathbf{Z}_k||_1
\end{aligned}$$

We now open both sums:

$$\begin{aligned}
\sum_{i,j}\sum_{k} S^1_{i,k} \cdot S^2_{k,j} \cdot ||\mathbf{X}_i - \mathbf{Z}_k||_1 &= \sum_{i}\sum_{k} S^1_{i,k} \cdot ||\mathbf{X}_i - \mathbf{Z}_k||_1 \cdot \sum_{j} S^2_{k,j} \\
&= \frac{1}{n_2}\sum_{i}\sum_{k} S^1_{i,k} \cdot ||\mathbf{X}_i - \mathbf{Z}_k||_1
\end{aligned}$$

With the same argument, we obtain that

$$\sum_{i,j}\sum_{k} S^1_{i,k} \cdot S^2_{k,j} \cdot ||\mathbf{Y}_j - \mathbf{Z}_k||_1 = \frac{1}{n_2}\sum_{i,j} S^2_{i,j} \cdot ||\mathbf{Y}_i - \mathbf{Z}_j||_1$$

So overall we obtain

$$\begin{aligned}
\alpha \leq& |n_1 - n_3| + ||\mathbf{A}S^3 - S^3\mathbf{C}||_1 + \sum_{i,j} S^3_{i,j}||\mathbf{X}_i - \mathbf{Z}_j||_1 \\
\leq& |n_1 - n_2| + ||\mathbf{A}S^1 - S^1\mathbf{B}||_1 + \sum_{i,j} S^1_{i,j}||\mathbf{X}_i - \mathbf{Y}_j||_1 + |n_2 - n_3| \\
&+ ||\mathbf{B}S^2 - S^2\mathbf{C}||_1 + \sum_{i,j} S^3_{i,j}||\mathbf{Y}_i - \mathbf{Z}_j||_1 = \alpha + \beta
\end{aligned}$$

Now, we show that our metric ρ_{DS} is equivalent to WL. Clearly, any pair of graphs with different numbers of vertices are distinguished both by WL and by ρ_{DS}. Thus, in the following, we assume that the two graphs have the same number of vertices.

We begin first with some needed definitions.

Partitions. Most of our techniques are inspired by [59]. Given $\mathcal{G} = (\mathbf{A}, \mathbf{X})$, we define a stable partition $\mathcal{V} = \mathcal{P}_1 \cup \mathcal{P}_2 ... \cup \mathcal{P}_k$ if

$$\forall u, v \in \mathcal{P}_i, \mathbf{X}_u = \mathbf{X}_v \tag{16}$$

$$\forall l \in [k], i \in [k], u, v \in \mathcal{P}_i, |\mathcal{N}_u \cap \mathcal{P}_l| = |\mathcal{N}_v \cap \mathcal{P}_l| \tag{17}$$

In simple words, two nodes in the same partition must have the same feature and the same number of neighbors with the same feature. Note that the union of all singleton is a valid stable partition. We can characterize a partition as a k tuple, and in the $i'th$ place, we put a tuple of the common feature and a vector telling the number of neighbors in other partitions. We say that graphs have the same stable partition; if, up to permitting of the k tuple indices, we have the same k tuple.

Lemma D.2. *The number of colors in 1-WL can't increase at level n. In addition, all nodes with the same color at step n make a stable partition.*

Proof. The number of colors of 1-WL can only increase as the number of iterations increases. It is also known that, if at some step t the number of colors does not increase, then the process is 'stuck'. Accordingly, after at most n iterations, the number of different colors will stay the same. Denote by $\mathcal{P} = (\mathcal{P}_1, .., \mathcal{P}_s)$ a partition of the nodes with the same coloring at the n iteration, and be $c \in C$ some color, we claim that nodes in the same partition, have the same number of neighbors with color c. As otherwise, those nodes that now have the same coloring will have different coloring in the next $n + 1$ iteration. But as the number of colors doesn't decrease (because of the concatenation of the current color), we have at least one more color, a contradiction. So, we found a stable partition. □

Lemma D.3. *The following conditions are equivalent:*

- $\mathcal{G} \cong_{1-WL} \mathcal{H}$
- *Both graphs have a common stable partition. That is, there is a relabeling of G such that the same partition $\mathcal{P}_1, .., \mathcal{P}_s$ of V is a stable partition for both $\mathcal{G}$ and $\mathcal{H}$, and the features inside each partition are the same, in both $\mathcal{G}$ and $\mathcal{H}$.*

Proof. Assume we have the same common partition. Up to renaming the names of the vertices of the nodes in the first graphs, we assume we have the same stable partition with the same parameters. Assume that a common stable partition exists $\mathcal{P} = \mathcal{P}_1 \cup .. \cup P_k$. Then, by simple induction on the number of iterations, we will prove that all nodes in the same partition have the same color in both graphs.
Basis Nodes in the same partition have the same initial feature and, thus, have the same color (in both graphs).
Step By the induction hypothesis, all nodes in the same partition iteration T have the same color; and they both have the same number of neighbors with the same color, so the aggregation yields the same output. Thus, in iteration $T + 1$, those nodes also have the same color. Note that this argument is symmetric to both graphs; thus, the 1-WL test will not distinguish them. On the other hand, if $\mathcal{G}_1, \mathcal{G}_2$ have the same 1-WL embedding, then we partition the nodes to those classes with the same color at iteration n, denoted by $\mathcal{P} = \mathcal{P}_1 \cup .. \cup P_k$. We have to show this partition is valid. First, by definition, those nodes $u, v \in \mathcal{P}_i$ have the same node feature (by iteration 1); next, if u, v don't have the same number of neighbors in $\mathcal{P}_l$ for some $l \in [k]$, then their color won't be the same at iteration $n + 1$. But, as shown in lemma D.2, the number of distinct colors can't increase between n to $n + 1$. Thus, we found a common stable partition. □

Before proving the following lemma, we revise a definition from [59]. We define direct the direct sum of matrices S_1 and S_2 by:

$$S_1 \oplus S_2 = \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix}$$

Given $S \in \mathbb{R}^{n\times n}$, we say S is composable if there exists P, Q, S_1, S_2 such that

$$S = P \cdot (S_1 \oplus S_2) \cdot Q$$

Such that P, Q are permutation matrices. By simple induction, we can write M as a direct sum of an indecomposable

$$S = P \cdot (S_1 \oplus S_2 \oplus ... \oplus S_t) \cdot Q$$

Lemma D.4. *Let $S \in \Pi(n, n)$ and assume it's in the form of $S = S_1 \oplus ... \oplus S_k$ such that all blocks are indecomposable and $\sum_{i,j} S_{i,j} \cdot ||\boldsymbol{X}_i - \boldsymbol{Y}_j|| = 0$. Denote by $i_1, ..., i_{k+1}$ the indices define the start and the end of $S_1, ..., S_t$ and by $I_k := [i_k, i_{k+1}]$. Then $\boldsymbol{X}_i = \boldsymbol{Y}_j, \forall t \in [k], \forall i, j \in I_t$.*

Proof. Given $t \in [k]$, we build a bipartite graph from I_t to itself, such that two indexes i, j are connected if $S_{i,j} > 0$. Note that this graph is connected, as otherwise, S_t could be composed into its connected components, and thus S_t would be composable. Given a path $P = (i_1, ..., i_l)$ in the graph, note that by definition, as the metric vanishes, and $S_{i_j, i_{j+1}} > 0$, so $\mathbf{X}_{i_j} = \mathbf{Y}_{i_{j+1}}$, thus $\mathbf{X}_i = \mathbf{Y}_j, i \in I_k, j \in I_k$ and we are done. □

Theorem D.5. *Be $\mathcal{G}, \mathcal{H}$ two featured graphs, then*

$$d(\mathcal{G}, \mathcal{H}) = 0 \iff \mathcal{G} \overset{\text{WL}}{\sim} \mathcal{H}$$

We first prove that if the two graphs are 1-WL equivalent, then this metric vanishes. We may assume they have the same stable partition as we proved above. Take $\mathcal{P} = \mathcal{P}_1 \cup .. \cup \mathcal{P}_t$, rename the nodes such that they come in consecutive order and denote by $n_1, n_2, .., n_k$ the sizes of the partitions. Note that by definition, $\forall u, v \in \mathcal{P}_i$, both have the same feature and number of neighbors in all $\mathcal{P}_l$. As in the book [59], define $S := \frac{1}{n_1} J_{n_1} \oplus ... \oplus \frac{1}{n_j} J_{n_k}$ and from the book [59], we know that $\mathbf{A}S = S\mathbf{B}$. As all nodes in the same partition have the same feature, $\mathbf{X}_i = \mathbf{Y}_j, \forall i, j \in I_k = [n_t, n_{t+1}], \forall t \in [k]$, so as S is non-zero only on indexes in the same partition then also this metric vanishes, so also the sum vanishes on S.

For the next direction, be $\mathcal{G} = (\mathbf{A}, \mathbf{X}), \mathcal{H} = (\mathbf{B}, \mathbf{Y})$ two graphs and assume there exists a matrix of the form of $S = P(S^1 \oplus ... \oplus S_t)Q$ such that the metric vanishes and denote by $D = S^1 \oplus ... \oplus S_t$ We show we can choose S such that it's a block matrix.

$$\begin{aligned}
||PDQ \cdot \mathbf{A} - \mathbf{B} \cdot PDQ|| =& ||P \cdot (DQ\mathbf{A} \cdot Q^{-1} - P^{-1}\mathbf{B}P \cdot D) \cdot Q|| \\
\overset{(*)}{=}& ||DQ\mathbf{A} \cdot Q^{-1} - P^{-1}\mathbf{B}P \cdot D|| \\
=& \sum_{i,j} S_{i,j} \cdot |\mathbf{X}_i - \mathbf{Y}_j| \sum_{i,j} (PDQ)_{i,j} \cdot |\mathbf{X}_i - \mathbf{Y}_j| \\
=& \sum_{i,j} D_{\pi_1^{-1}(i), \pi_2(j)} \cdot |\mathbf{X}_i - \mathbf{Y}_j| \\
=& \sum_{i,j} D_{i,j} \cdot |\mathbf{X}_{\pi_1(i)} - \mathbf{Y}_{\pi_2^{-1}(j)}|,
\end{aligned}$$

where (*) follows from the fact that permutation matrices preserve the norm. We define two graphs $\hat{\mathcal{G}} = (Q\mathbf{A}Q^{-1}, Q\mathbf{X}) \cong \mathcal{G}, \hat{\mathcal{H}} = (P^{-1}\mathbf{A}P, P^{-1}\mathbf{Y}) \cong \mathcal{H}$. So, we can choose S to be a diagonal block matrix. Note that D defines a partition $\mathcal{P}_1, \ldots, \mathcal{P}_s$ of the nodes, and we will prove that it's a stable partition of both graphs. From $SA = BS$, we obtain, as in the book [59], that each of the two nodes $u, v \in \mathcal{P}_k$ has the same number of neighbors in $\mathcal{P}_l, \forall l \in [s]$. Next, by the lemma D.4, we know that $\mathbf{X}_i = \mathbf{Y}_j, \forall i, j \in \mathcal{P}_k$, so nodes in the same partition have the same feature. Thus, this partition is stable. So, $\hat{\mathcal{G}}, \hat{\mathcal{H}}$ have exactly the same partition. Then $\mathcal{G}, \mathcal{H}$ have the same partition up to isomorphism, and by lemma D.3, both graphs are 1-WL equivalent. □

D.2 FSW-GNN: Equivalence to WL

We now prove theorem 3.2 that says the FSW-GNN is equivalent to WL.

Theorem 3.2. *Consider the FSW-GNN architecture for input graphs in $\mathcal{G}_{\leq N}(\mathbb{R}^d)$, with $T = N$ iterations, where $\Phi^{(t)}, \Psi$ are just linear funtions, and all features (except for input features) are of dimension $m \geq 2Nd + 2$. Then for Lebesgue almost every choice of model parameters, the graph embedding defined by the architecture is WL equivalent.*

Proof. This proof employs the finite witness theorem [5], and the basic methodology to use it for graph neural network injectivity, first presented by Hordan et al. [39].

It is sufficient to show that the functions $\Phi^{(t)}$ and Ψ are injective vector-to-vector functions, and that the FSW ebmedding maps used in the FSW-GNN are injective multiset to vector functions. The injectivity of all these functions ensures that the MPNN is WL-equivalent [62].

By Theorem 4.1 in [4], if we take the FSW embedding with an output dimension of $2Nd+1$, defined on the space of multisets with at most N features in $\mathbb{R}^d$, then the embedding will be injective for Lebesgue almost every choice of its parameters. This injectivity is in the sense that two multisets will attain the same value if and only if they define the same distribution. This can happen when the multisets are not equal, but have different cardinalities. However, the last coordinae of the FSW embedding, which encodes the multiset cardinality, ensures that this will not be an issue. Accordingly the FSW embedding is injective on the multiset space.

We note that the output of the FSW embedding will now be node features of a higher dimension of $2Nd + 2$, rather than just d. However, the set of all possible features obtrained from the previous iteration will be a semialgebraic set of dimension $\leq Nd$. As long as this is the case, reapplying the FSW embedding with output dimension of $2Nd + 2$ will still be injective, for almost every choice of parameters.

The same argument can also be applied for the linear functions $\Phi^{(t)}$ and Ψ used in FSW-GNN: It is an easy consequence of the finite witness theorem [5, 30], that almost every $(2Nd + 1) \times m$ matrix, applied to a semialgebraic subset of $\mathbb{R}^m$ of dimension $\leq Nd$, will be injective. This concludes the proof. □

D.3 FSW-GNN: Bi-Lipschitzness

We first prove lemma 3.4.

Lemma 3.4. *Let $f, g : M \to \mathbb{R}_{\geq 0}$ be nonnegative piecewise-linear functions defined on a compact polygon $M \subset \mathbb{R}^d$. Suppose that for all $x \in M$, $f(x) = 0$ if and only if $g(x) = 0$. Then there exist real constants $c, C > 0$ such that*

$$c \cdot g(x) \leq f(x) \leq C \cdot g(x), \quad \forall x \in M. \tag{13}$$

Proof. By definition, there is a partition of M into a finite union of compact polygons, so that the restriction of f to each polygon is an affine function. g also has such a partition, and by taking the mutual refinement of the two partitions, we obtain a finite partition of M into compact polygons, such that the restriction of both f and g to each polygon is affine.

Accordingly, it is sufficient to prove the claim on a single compact polygon $P \subseteq M$ on which the restriction of both f and g is affine. A compact polygon has a finite number of extreme points $v_1, \ldots, v_k$, and every point in the polygon can be written as a convex combination of these points. Now let us denote

$$I = \{i \in [k] | \quad f(v_i) > 0\} = \{i \in [k] | \quad g(v_i) > 0\}.$$

We note that if I is empty, then f and g are identically zero on P, and there is nothing to prove. Otherwise, the following quantities are well defined and strictly positive:

$$m_f = \min_{i \in I} f(v_i), \quad M_f = \max_{i \in I} f(v_i)$$
$$m_g = \min_{i \in I} g(v_i), \quad M_g = \max_{i \in I} g(v_i)$$

Then for every $x \in P$, there exist non-negative $\theta_1, \ldots, \theta_k$ which sum to 1 such that $x = \sum_{i=1}^{k} \theta_i v_i$. It follows that

$$\begin{aligned} f(x) &= f(\sum_{i=1}^{k} \theta_i v_i) = \sum_{i=1}^{k} \theta_i f(v_i) = \sum_{i \in I} \theta_i f(v_i) \\ &\geq m_f \sum_{i \in I} \theta_i = \frac{m_f}{M_g} \cdot M_g \sum_{i \in I} \theta_i \geq \frac{m_f}{M_g} \cdot \sum_{i \in I} \theta_i g(v_i) \\ &= \frac{m_f}{M_g} \cdot g(\sum_{i=1}^{k} \theta_i v_i) = \frac{m_f}{M_g} \cdot g(x). \end{aligned}$$

By reversing the roles of f and g, we obtain that for every $x \in P$,

$$g(x) \geq \frac{m_g}{M_f} f(x).$$

This concludes the proof. □

We now turn to the full proof of theorem 3.3.

Theorem 3.3. *Let $\Omega \subset \mathbb{R}^d$ be compact. Under the assumptions of theorem 3.2, the FSW-GNN is bi-Lipschitz with respect to ρ_{DS} on $\mathcal{G}_{\leq N}(\Omega)$. If, additionally, Ω does not contain $\mathbf{0}$, then the FSW-GNN is bi-Lipschitz with respect to TMD on $\mathcal{G}_{\leq N}(\Omega)$.*

Proof. Following the reasoning in the main text, the rest of the proof is as follows: Let $G, \tilde{G} \in \mathcal{G}_{\leq N}(\Omega)$, $G = (V, E, X)$ and $\tilde{G} = \left(\tilde{V}, \tilde{E}, \tilde{X}\right)$, with n and $\tilde{n}$ vertices respectively. Since there is a finite number of choices of possible $n, \tilde{n} \leq N$ and edges E, $\tilde{E}$ It is enough to show that once these parameters are chosen, the bi-Lipschitz ratio is bounded on all choices of $X \in \Omega^n$, $\tilde{X} \in \Omega^{\tilde{n}}$, . Define the function $f\left(X, \tilde{X}\right) = \|\boldsymbol{h}_G - \boldsymbol{h}_{\tilde{G}}\|_1$, $g\left(X, \tilde{X}\right) = \rho\left(G, \tilde{G}\right)$, where ρ is either ρ_{DS} or TMD. By the comment above, both f and g are piecewise linear. Since both FSW-GNN and the metric ρ are WL-equivalent, f and g have the same zero set. Lastly, since Ω is compact and does not contain $\mathbf{0}$, the distance from Ω to $\mathbf{0}$ is strictly positive, and thus Ω can be extended to a compact polygon that does not contain $\mathbf{0}$ (e.g. by taking the set-difference between a compact box that contains Ω and a small ℓ_∞ ball around $\mathbf{0}$ that is disjoint with Ω). Thus, lemma 3.4 guarantees the existence of Lipschitz constants c, C.

Finally, note that by equivalence of all norms on a finite dimensional space, the same bi-Lipschitz equivalence holds if we replace the one-norm in the definition of f with any other norm. Q.E.D. □

Theorem D.6. *Suppose that $\Omega \subset \mathbb{R}^d$ is a compact set that does not contain zero. Under the assumptions of theorem 3.2, the node features computed by FSW-GNN are bi-Lipschitz with respect to the Tree Distance metric on $\mathcal{G}_{\leq N}(\Omega)$.*

Proof. Let $G = (V, A, \boldsymbol{X})$, $\tilde{G} = \left(\tilde{V}, \tilde{A}, \tilde{\boldsymbol{X}}\right)$ be two graphs in $\mathcal{G}_{\leq N}(\Omega)$. Let $t \in \mathbb{N}$ denote the number of MPNN/WL iterations iterations applied. Let $v \in V$, $\tilde{v} \in \tilde{V}$. Let $\boldsymbol{h}_v^{(t)}$ and $\boldsymbol{h}_{\tilde{v}}^{(t)}$ be the output features of vertices v and $\tilde{v}$ computed by FSW-GNN. Then $\text{TD}\left(\mathbf{T}_v^{(t)}, \mathbf{T}_{\tilde{v}}^{(t)}\right) = 0$ if and only if v and $\tilde{v}$ are assigned the same features by the WL test, which in turn takes place if and only if $\boldsymbol{h}_v^{(t)} = \boldsymbol{h}_{\tilde{v}}^{(t)}$. Note that $\|\boldsymbol{h}_v - \boldsymbol{h}_{\tilde{v}}\|_1$ is a piecewise-linear function of the input vertex features $\{\boldsymbol{x}_v\}_{v \in V}$ and $\{\tilde{\boldsymbol{x}}_{\tilde{v}}\}_{\tilde{v} \in \tilde{V}}$. Moreover, $\text{TD}\left(\mathbf{T}_v^{(t)}, \mathbf{T}_{\tilde{v}}^{(t)}\right) = 0$ can be transformed with bounded distortion to a piecewise-linear function of $\{\boldsymbol{x}_v\}_{v \in V}$ and $\{\tilde{\boldsymbol{x}}_{\tilde{v}}\}_{\tilde{v} \in \tilde{V}}$. Lastly, similarly to the proof of theorem 3.3, Ω can be extended to a compact polygon that does not contain $\mathbf{0}$. Thus, by lemma 3.4, there exist constants $0 < c, C < \infty$ such that

$$c \cdot \text{TD}\left(\mathbf{T}_v^{(t)}, \mathbf{T}_{\tilde{v}}^{(t)}\right) \leq \left\|\boldsymbol{h}_v^{(t)} - \boldsymbol{h}_{\tilde{v}}^{(t)}\right\|_1 \leq C \cdot \text{TD}\left(\mathbf{T}_v^{(t)}, \mathbf{T}_{\tilde{v}}^{(t)}\right).$$

□

E Experiment Details

We used the Adam optimizer for all experiments.

For the empirical distortion evaluation, we used pairs of graphs $G, \tilde{G}$, each of which consisting of four vertices and the edges $1-2-3-4-1$. Two random vectors $v_0, \Delta v \in \mathbb{R}^d$ were drawn i.i.d. Gaussian and normalized to unit length. In G all vertex features were set to v_0, whereas in $\tilde{G}$ they were set to $v_0 + \varepsilon\sigma\Delta v$, with $\sigma = 1$ for v_2, v_4 and -1 for v_1, v_3. We used $\varepsilon = \left\{1, 10^{-1}, 10^{-2}, \ldots, 10^{-6}\right\}$, and generated 100 pairs for each value of ε, and evaluated the constants C, c for the resulting pairs. This experiment was repeated 10 times and the average distortion was taken. To ensure accurate results, we used 64-bit floating-point arithmetic in this experiment.

For the NeighborsMatch problem from [3], we used the protocol developed in their paper: we used their implementation for the MPNNs we compared to, with a hidden dimension of 64 for all models, searched for each of its best hyper-parameters, and reported the training accuracy. For fair comparison with rival models, we repeated each sample 100 times, as was done in [3].

For the Ring dataset, we used the results from [27] and trained our models with a hidden dimension of 64.

For the LRGB dataset, we trained all models under the constraint of 500K parameters. In contrast, for the MolHIV dataset, there was no restriction, and we trained the models with 40K parameters.

For the transductive learning tasks, we used a hidden dimension of 128 across all models.

Timing. Here, we show the testing time for FSW-GNN, GIN, and GCN, GGNN, GAT, and GraphSAGE on the MUTAG graph classification task.

Table 6: Comparison of Average Time in seconds per testing in evaluation mode for different models on MUTAG.

Model	Avg Time per Testing
GIN	0.09
GCN	0.08
GAT	0.09
GGNN	0.08
GraphSage	0.07
FSW-GNN	0.11

F Ordered subgraph aggregation

In this section, we recall sub-graph aggregation networks and show that FSW-GNN could be used to create a Bi-lipshiz embedding with respect to the metric equivalent to sub-graph WL.

F.1 Ordered subgraph 1-WL

. The OSWL (Ordered sub-graph WL) is taken from [14, 56]. Given 1-WL procedure using T iterations denoted by WL^T, and deterministic graph sampling Π(could be node/edge deletion), we define the following three steps:

- Step 1: Deterministic graph sampling (e.g. node/edge deletion) $\Pi(\mathcal{G}) = \{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$.
- $y_i = WL^T(\mathcal{G}_i), \mathcal{G}_i \in \Pi(\mathcal{G})$
- $y_G = \phi(\{y_1, \ldots, y_m\})$

Where WL^T denotes the application of T iterations of the WL test, and ϕ is an injective multiset function.

Definition F.1. We say $\mathcal{G}_1, \mathcal{G}_2$ to be equivalent up to OSWL, if $OSWL^T(\mathcal{G}_1) = OSWL^T(\mathcal{G}_2), \forall T \in \mathbb{N}$, and denoted by $\mathcal{G}_1 \cong \mathcal{G}_2$. As for WL, checking equality for $T = |\mathcal{G}_1|$ is sufficient to ensure equality for all T.

Ordered subgraph MPNN. Ordered subgraph MPNNs are very similar to OSWL. Given an MPNN and a multi-set to vector embedding ϕ, we define its sub-graph version:

- Step 1: Deterministic graph sampling $\Pi(\mathcal{G}) = \{\mathcal{G}_1, \dots, \mathcal{G}_m\}$
- $y_i = MPNN(\mathcal{G}_i), \mathcal{G}_i \in \Pi(\mathcal{G})$
- $y_G = \phi(\{y_1, \dots, y_m\})$

OSWL equivalent metric. We define a metric OSWL equivalent similarly to 2.1:
Definition (WL metric)**.** A *OSWL metric on* $\mathcal{G}_{=N}(\Omega)$ is a function $\rho : \mathcal{G}_{=N}(\Omega) \times \mathcal{G}_{=N}(\Omega) \to \mathbb{R}_{\geq 0}$ that satisfies the following conditions for all $G_1, G_2, G_3 \in \mathcal{G}_{=N}(\Omega)$:

$$\rho(G_1, G_2) = \rho(G_2, G_1) \qquad \textit{Symmetry} \tag{18a}$$
$$\rho(G_1, G_3) \leq \rho(G_1, G_2) + \rho(G_2, G_3) \qquad \textit{Triangle inequality} \tag{18b}$$
$$\rho(G_1, G_2) = 0 \iff G_1 \cong G_2. \qquad \textit{OSWL equivalence} \tag{18c}$$

Theorem F.2. *Be a 1-WL equivalent metric d_{WL}, and sampling procedure, the following metric is OSWL equivalent:*

$$d_{sub}(\mathcal{G}, \mathcal{H}) = min_{\pi \in S_m} \sum_{i=1}^{m} d_{WL}(\mathcal{G}_i, \mathcal{H}_{\pi(i)})$$

Proof. We prove only the last part, which is not trivial. Assume first that the number of nodes is the same, then:

$$\begin{aligned} d_{sub}(\mathcal{G}, \mathcal{H}) = 0 \iff \exists \pi \in S_m : \forall i \in [m] : d_{WL}(\mathcal{G}_i, \mathcal{H}_{\pi(i)}) = 0 \iff \\ \iff \exists \pi \in S_m : \forall i \in [m], y_i^{\mathcal{G}} = y_{\pi(i)}^{\mathcal{H}} \\ \phi(\{y_1^{\mathcal{G}}, \dots, y_m^{\mathcal{G}}\}) = \phi(\{y_1^{\mathcal{H}}, \dots, y_m^{\mathcal{H}}\}) \iff OSWL(\mathcal{G}) = OSWL(\mathcal{H}) \end{aligned}$$

□

Theorem F.3. *Assume we use FSW-GNN as MPNN, FSW aggregation as ϕ, d_{DS}, as the 1-WL equivalent metric, then the overall OSMPNN is OSWL bilipshiz.*

Proof. Note that the OSMPNN is piece-wise linear as a composition of piece-wise linear functions. Also, d_{sub} is piece-wise linear a summation of piece-wise linear functions. In addition, both functions vanish on the same set, so by the 3.4, the OSMPNN is bilipshiz. □