Permutation-equivariant and Proximity-aware Graph Neural Networks with Stochastic Message Passing

Ziwei Zhang, Chenhao Niu, Peng Cui, Jian Pei, *Fellow, IEEE,* Bo Zhang, and Wenwu Zhu, *Fellow, IEEE*

Abstract—Graph neural networks (GNNs) are emerging machine learning models on graphs. Permutation-equivariance and proximity-awareness are two important properties highly desirable for GNNs. Both properties are needed to tackle some challenging graph problems, such as finding communities and leaders. In this paper, we first analytically show that the existing GNNs, mostly based on the message-passing mechanism, cannot simultaneously preserve the two properties. Then, we propose Stochastic Message Passing (SMP) model, a general and simple GNN to maintain both proximity-awareness and permutation-equivariance. In order to preserve node proximities, we augment the existing GNNs with stochastic node representations. We theoretically prove that the mechanism can enable GNNs to preserve node proximities, and at the same time, maintain permutation-equivariance with certain parametrization. We report extensive experimental results on ten datasets and demonstrate the effectiveness and efficiency of SMP for various typical graph mining tasks, including graph reconstruction, node classification, and link prediction.

Index Terms—Graph Neural Network, Node Proximity, Permutation Equivariance, Message Passing.

1 Introduction

Graph neural networks (GNNs), as generalizations of neural networks for learning on graph data, have enjoyed successes in many applications, such as social recommendation [1], physical simulation [2], and protein interaction prediction [3]. The existing GNNs are mostly based on the message-passing mechanism [4].

A fundamental property well preserved by the message-passing GNNs is permutation-equivariance, i.e., if we randomly permutate the IDs of nodes while maintaining the graph structure unchanged, the representations of nodes in those GNNs are permutated accordingly. Mathematically, permutation-equivariance reflects one basic symmetric group of graph structures. Permutation-equivariance is highly useful for many graph mining tasks, such as node or graph classification [5], [6]. As another important property, pairwise proximities between nodes are crucial for some other graph mining tasks, such as link prediction and community detection [7], [8]. Some GNNs, such as Position-aware GNN (P-GNN) [8], have specifically designed mechanisms to ensure proximity-awareness.

In many applications of GNNs, both proximity-awareness and permutation-equivariance are indispensable. Consider mining communities and leaders in graphs. Figure 1 shows a toy example for illustration. Figure 1(a) shows the communities and Figure 1(b) shows the nodes categorized by the k-core number centrality. To discover the communities, proximity-awareness is essential since the nodes in the same community are tightly connected and have large proximities. Permutation-equivariance helps to measure the centrality because most centrality measurements are permutation-equivariant by definition [9].

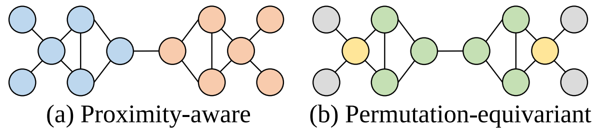


Fig. 1. A toy example of illustrating finding communities and leaders. Labels are shown in different colors. (a) The two communities discovered by spectral clustering, in which proximity-awareness is essential. (b) The node labels correspond to the k-core number, a type of node centrality. Permutation-equivariance is important for the task.

Do the existing GNNs, which are built on message-passing, honor both proximity-awareness and permutation-equivariance? Surprisingly and unfortunately, the answer is no. We show that proximity-awareness and permutation-equivariance are incompatible in the exiting GNNs (see Theorem 1). This deficiency in the existing GNNs is particularly irritating since, for the same task, different datasets may rely on the two properties to different extents. Taking link prediction as an example, we observe that permutation-equivariant GNNs such as GCN [10] or GAT [11] show better performance than P-GNN in coauthor graphs, but perform worse in biological graphs (see Section 5.3 for details). A work in drug repurposing for Covid-19 [12] shows a similar dilemma: proximity-aware methods and permutation-equivariant GNNs discover completely different drug candidates.

Can we develop a general GNN that is proximity-aware and also maintains permutation-equivariance? In this paper,

- *Z. Zhang, P. Cui, and W. Zhu are with the Department of Computer Science and Technology at Tsinghua University, Beijing 100084, China. Email: {zwzhang,cuip,wwzhu}@tsinghua.edu.cn*
- *C. Niu is with the School of Computer Science at Carnegie Mellon University, Pittsburgh, PA 15213, USA. Work was done during his undergraduate study at Tsinghua Unversity. E-mail: cniu@andrew.cmu.edu*
- *J. Pei is with the School of Computing Science at Simon Fraser University, Burnaby, BC V5A 1S6, Canada. E-mail: jpei@cs.sfu.ca*
- *B. Zhang is with Tencent. E-mail: nevinzhang@tencent.com.*
- *Z. Zhang and C. Niu contributed equally to this work.*

Manuscript received June 01, 2021; revised January 14, 2022.

we propose Stochastic Message Passing (SMP)[1], a general and simple GNN to preserve both proximity-awareness and permutation-equivariance properties. In order to preserve node proximities, we augment the existing GNNs with stochastic node representations. We theoretically prove that the mechanism can enable GNNs to preserve node proximities (see Theorems 2 and 3). At the same time, SMP is equivalent to a permutation-equivariant GNN with certain parametrization and thus is at least as powerful as those GNNs in permutation-equivariant tasks (see Remark 1). Therefore, SMP is general and flexible in handling both proximity-aware and permutation-equivariant tasks, which is also demonstrated by our extensive experimental results. Besides, owing to the simple structure, SMP is computationally efficient, with a running time roughly the same as the simplest GNNs, such as SGC [13], and is at least an order of magnitude faster than P-GNN on large graphs. Our ablation studies further show that a linear instantiation of SMP is expressive enough as adding extra non-linearities does not lift the performance of SMP on the majority of datasets. Our contributions are summarized as follows.

- We propose Stochastic Message Passing (SMP), a simple and general GNN to handle both proximity-aware and permutation-equivariant graph tasks.
- We prove that SMP has theoretical guarantees in preserving walk-based proximities and is as powerful as the existing GNNs in permutation-equivariant tasks.
- Extensive experimental results demonstrate the effectiveness and efficiency of SMP. We show that a linear SMP instantiation is expressive enough on the majority of datasets.

The rest of the paper is organized as follows. We review related work in Section 2. In Section 3, we show the incompatibility between walk-based proximity and permutation-equivariance in message-passing GNNS. We develop our proposed Stochastic Message Passing in Section 4. We report an extensive experimental study in Section 5, and conclude the paper in Section 6. We provide additional experiments, details for reproducibility, and proofs in the appendix.

2 Related Work

We briefly review GNNs, the permutation-equivariance property, and the proximity-awareness property. We refer readers to [14] for a comprehensive survey.

The earliest GNNs adopt a recursive definition of node states [15], [16] or a contextual realization [17]. GGS-NNs [18] replace the recursive definition with recurrent neural networks (RNNs). Spectral GCNs [19] define graph convolutions using graph signal processing [20] with ChebNet [21] and GCN [10] approximating the spectral filters using a low-order Chebyshev polynomial and the first-order polynomial, respectively. MPNNs [4], GraphSAGE [3], and MoNet [22] are proposed as general frameworks by characterizing GNNs with a message-passing function and an updating function. More advanced variants such as GAT [11], JK-Nets [23], GIN [24], and GraphNets [25] follow these frameworks.

Li *et al.* [26], Xu *et al.* [24], Morris *et al.* [27], and Maron *et al.* [28] show the connection between GNNs and the Weisfeiler-Lehman algorithm [29] of graph isomorphism tests, in which permutation-equivariance holds a key constraint. Further, Maron *et al.* [6] and Keriven *et al.* [5] analyze the permutation-equivariance property of GNNs theoretically. To date, most of the existing GNNs are permutation-equivariant and are not proximity-aware. One exception is P-GNN [8], which proposes to capture the positions of nodes using the relative distance between the target node and some randomly chosen anchor nodes. However, P-GNN cannot satisfy permutation-equivariance and is computationally expensive. Concurrent to our work, some studies propose to use position encodings to enhance GNNs in preserving graph structures [30], [31], [32], [33]. Most of these methods rely on eigenvectors of a graph matrix, which are computationally expensive. Our proposed stochastic node representations can also be regarded as a type of position encoding while being extremely simple yet efficient.

In order to enhance the expressive power of GNNs in graph isomorphism tests and also motivated by the literature on distributed computing [34], some studies suggest assigning unique node identifiers for GNNs [35], such as one-hot IDs [36] or random numbers [37], [38], [39]. For example, Sato *et al.* [38] show that random numbers can enhance GNNs in tackling two graph-based NP problems with a theoretical guarantee, namely the minimum dominating set and the maximum matching problem. Fey *et al.* [40] empirically show the effectiveness of random features in the graph matching problem. Concurrent to our work, RNI [41] shows that GNNs with random node features are universal in theory. Our work here, which also adopts stochastic node representations, differs in that we systematically study how to preserve permutation-equivariance and proximity-awareness simultaneously in a simple yet effective framework, a novel topic distinct from those existing studies. Besides, we theoretically prove that our proposed method can preserve walk-based proximities. We also demonstrate the effectiveness of our method on large-scale benchmarks for both node- and edge-level tasks, while no similar results are reported in the literature.

Another line of research is tackling the over-smoothing problem [26], [42] and developing deep GNNs [43]. Since these studies are orthogonal to our paper, we expect these strategies to also work for our proposed SMP.

The design of our method is also inspired by the literature on random projection for dimensionality reduction [44]. To the best of our knowledge, we are the first to study random projection in the scope of GNNs. More remotely, our definition of node proximities is inspired and inherited from graph kernels [45], [46], network embedding [47], [48], and the general studies of graphs [49].

3 Message-passing GNNs and Analyses

In this section, we first introduce preliminaries of message-passing GNNs, walk-based proximities, and permutation-equivariance. Then, we show the incompatibility between proximity-awareness and permutation-equivariance in the existing GNNs.

1. Code is available at https://github.com/NiuChH/SMP.

We consider a graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{F})$ where $\mathcal{V} = \{v_1, ..., v_N\}$ is a set of $N = |\mathcal{V}|$ nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of $M = |\mathcal{E}|$ edges, and $\mathbf{F} \in \mathbb{R}^{N \times d_0}$ is a matrix of d_0 node features. Denote by $\mathbf{A}$ the adjacency matrix, and by $\mathbf{A}_{i,:}$, $\mathbf{A}_{:,j}$, and $\mathbf{A}_{i,j}$, respectively, the i^{th} row, the j^{th} column and an element in the matrix. In this paper, we assume unweighted and undirected graphs. The neighborhood of node v_i is denoted by $\mathcal{N}_i$. Let $\tilde{\mathcal{N}_i} = \mathcal{N}_i \cup \{v_i\}$.

The existing GNNs usually follow a message-passing framework [4], where a neighborhood aggregation function $\text{AGG}(\cdot)$ and an updating function $\text{UPDATE}(\cdot)$ are adopted in the l^{th} layer:

$$\begin{aligned} \mathbf{m}_i^{(l)} &= \text{AGG}\left(\left\{\mathbf{h}_j^{(l)}, \mathbf{h}_i^{(l)}, \mathbf{e}_{i,j}, \forall j \in \tilde{\mathcal{N}_i}\right\},\right) \\ \mathbf{h}_i^{(l+1)} &= \text{UPDATE}\left(\left[\mathbf{h}_i^{(l)}, \mathbf{m}_i^{(l)}\right]\right), \end{aligned} \tag{1}$$

where $\mathbf{h}_i^{(l)} \in \mathbb{R}^{d_l}$ is the representation of node v_i at the l^{th} layer, d_l is the dimensionality, $\mathbf{e}_{i,j}$ is the edge feature when available, and $\mathbf{m}_i^{(l)}$ are the messages. We also denote by $\mathbf{H}^{(l)} = [\mathbf{h}_1^{(l)}, ..., \mathbf{h}_N^{(l)}]$ and $[\cdot, \cdot]$ the concatenation operation. The node representations are initialized as node features, i.e., $\mathbf{H}^{(0)} = \mathbf{F}$. We represent a GNN following Eq. (1) with L layers by a parameterized function as follows[2]:

$$\mathbf{H}^{(L)} = \mathcal{F}_{\text{GNN}}(\mathbf{A}, \mathbf{F}; \mathbf{W}), \tag{2}$$

where $\mathbf{H}^{(L)}$ is node representation learned by the GNN and $\mathbf{W}$ represents all the parameters.

A key property of GNNs is permutation-equivariance.

Definition 1 (Permutation-equivariance)**.** *Consider a graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{F})$ and any permutation $\mathcal{P} : \mathcal{V} \to \mathcal{V}$ so that $G' = (\mathcal{V}, \mathcal{E}', \mathbf{F}')$ has an adjacency matrix $\mathbf{A}' = \mathbf{P}\mathbf{A}\mathbf{P}^T$ and a feature matrix $\mathbf{F}' = \mathbf{P}\mathbf{F}$, where $\mathbf{P} \in \{0,1\}^{N \times N}$ is the permutation matrix, i.e., $\mathbf{P}_{i,j} = 1$ iff $\mathcal{P}(v_i) = v_j$. A GNN satisfies permutation-equivariance if the node representations for G and G′ are equivariant with respect to $\mathbf{P}$, i.e.,*

$$\mathbf{P}\mathcal{F}_{GNN}(\mathbf{A}, \mathbf{F}; \mathbf{W}) = \mathcal{F}_{GNN}(\mathbf{P}\mathbf{A}\mathbf{P}^T, \mathbf{P}\mathbf{F}; \mathbf{W}). \tag{3}$$

It is well-known that GNNs following Eq. (1) satisfy permutation-equivariance [6].

Definition 2 (Automorphism)**.** *A graph G is said to have (non-trivial) automorphism if there exists a non-identity permutation matrix $\mathbf{P} \neq \mathbf{I}_N$ so that $\mathbf{A} = \mathbf{P}\mathbf{A}\mathbf{P}^T$ and $\mathbf{F} = \mathbf{P}\mathbf{F}$, i.e., the graph has a non-trivial isomorphism to itself. We denote by $\mathcal{C}_G = \bigcup_{\mathbf{P} \neq \mathbf{I}_N} \{(i,j) | \mathbf{P}_{i,j} \neq 0, i \neq j\}$ the corresponding automorphic node pairs.*

Using Definition 1 and 2, we immediately have the following corollary.

Corollary 1. *If a graph has a non-trivial automorphism, a permutation-equivariant GNN produces identical node representations for automorphic node pairs*

$$\mathbf{h}_i^{(L)} = \mathbf{h}_j^{(L)}, \forall (i,j) \in \mathcal{C}_G. \tag{4}$$

Since node representations are used for downstream tasks, Corollary 1 shows that permutation-equivariant GNNs cannot differentiate automorphic node pairs. A direct consequence is that permutation-equivariant GNNs cannot preserve walk-based proximities between pairs of nodes.

Definition 3 (Walk-based Proximities)**.** *For a given graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{F})$, denote by matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$ the walk-based proximities between pairs of nodes, defined by*

$$\mathbf{S}_{i,j} = \mathcal{S}\left(\{v_i \rightsquigarrow v_j\}\right), \tag{5}$$

where $\{v_i \rightsquigarrow v_j\}$ represents the set of walks from node v_i to v_j and $\mathcal{S}(\cdot)$ is a real-valued function. The length of a walk-based proximity is the maximum length of all the walks of the proximity.

Typical examples of walk-based proximities include the shortest distance [8], the high-order proximities (a sum of walks weighted by their lengths) [50], and random walk probabilities [51].

Definition 4. *For a given walk-based proximity, a GNN is said to preserve the proximity if there exists a decoder function $\mathcal{F}_{de}(\cdot)$ such that for any graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{F})$, there exist parameters $\mathbf{W}_G$ so that $\forall \epsilon > 0$:*

$$\left|\mathbf{S}_{i,j} - \mathcal{F}_{de}\left(\mathbf{H}_{i,:}^{(L)}, \mathbf{H}_{j,:}^{(L)}; \mathcal{S}(\cdot)\right)\right| < \epsilon, \tag{6}$$

where

$$\mathbf{H}^{(L)} = \mathcal{F}_{GNN}(\mathbf{A}, \mathbf{F}; \mathbf{W}_G). \tag{7}$$

For notation convenience, we also write the decoder function as $\mathcal{F}_{de}\left(\mathbf{H}_{i,:}^{(L)}, \mathbf{H}_{j,:}^{(L)}\right)$ when there is no ambiguity regarding $\mathcal{F}(\cdot)$.

The definition applies to any GNN architecture as long as it fits Eq. (1). Moreover, in the definition, we only constrain the inputs of the decoder function to be node representations $\mathbf{H}$ and the proximity function $\mathcal{S}(\cdot)$, but we do not constrain the form of the decoder function. In other words, the decoder function can be arbitrarily sophisticated, e.g., deep neural networks with a sufficient number of layers and hidden units. Now we are ready to present the incompatibility.

Theorem 1. *For any walk-based proximity function $\mathcal{S}(\cdot)$ satisfying Definition 3, a permutation-equivariant GNN cannot preserve $\mathcal{S}(\cdot)$, except for the trivial situation where all node pairs have the same proximity, i.e., $\forall i, j$, $\mathbf{S}_{i,j} = c$, and c is a constant.*[3]

Proof. We prove by contradiction. Assume there exists a non-trivial $\mathcal{S}(\cdot)$ that can be preserved by a permutation-equivariant GNN. Consider any graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{F})$. We will construct a graph G' with automorphism from G so that any GNN cannot preserve $\mathcal{S}(\cdot)$ on G'. Specifically, let $N = |\mathcal{V}|$. We create $G' = (\mathcal{V}', \mathcal{E}', \mathbf{F}')$, $|\mathcal{V}'| = 2N$, such that

$$\mathcal{E}'_{i,j} = \begin{cases} \mathcal{E}_{i,j} & \text{if } i \leq N, j \leq N \\ \mathcal{E}_{i-N,j-N} & \text{if } i > N, j > N \\ 0 & \text{else} \end{cases}$$

$$\mathbf{F}'_{i,:} = \begin{cases} \mathbf{F}_{i,:} & \text{if } i \leq N \\ \mathbf{F}_{i-N,:} & \text{if } i > N \end{cases}.$$

Basically, we generate two "copies" of the original graph, one indexing from 1 to N, and the other one from $N+1$ to $2N$. By assumption, there exists a permutation-equivariant

2. Since the final layer of GNNs is task-specific, e.g., a softmax layer for node classification or a readout layer for graph classification, we only consider the GNN architecture to its last hidden layer.

3. Proposition 1 in P-GNN [8] can be regarded as a special case of Theorem 1 using the shortest distance proximity.

GNN that can preserve $\mathcal{S}(\cdot)$ in G'. Let the node representations for such a GNN as $\mathbf{H}'^{(L)} = \mathcal{F}_{\text{GNN}}(\mathbf{A}', \mathbf{F}'; \mathbf{W}_{G'})$. It is easy to see that node v'_i and v'_{i+N} in G' form an automorphic node pair. According to Corollary 1, their representations are identical, i.e.,

$$\mathbf{H}'^{(L)}_{i,:} = \mathbf{H}'^{(L)}_{i+N,:}, \forall i \leq N. \tag{8}$$

Note that there exists no walk from the two copies, i.e. $\left\{v'_i \rightsquigarrow v'_j\right\} = \left\{v'_j \rightsquigarrow v'_i\right\} = \emptyset, \forall i \leq N, j > N$. As a result, for $\forall i \leq N, j \leq N, \forall \epsilon > 0$, we have:

$$\begin{aligned}|\mathbf{S}_{i,j} - \mathcal{S}(\emptyset)| &\leq \left|\mathbf{S}_{i,j} - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j,:}\right)\right| + \left|\mathcal{S}(\emptyset) - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j,:}\right)\right| \\ &= \left|\mathbf{S}_{i,j} - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j,:}\right)\right| + \left|\mathbf{S}_{i,j+N} - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j+N,:}\right)\right| < 2\epsilon.\end{aligned}$$

We can prove the same for $\forall i > N, j > N$. The equation naturally holds if $i \leq N, j > N$ or $i > N, j \leq N$, since $\left\{v'_i \rightsquigarrow v'_j\right\} = \emptyset$. Combining the results, we have $\forall \epsilon > 0, \forall i, j$, $|\mathbf{S}_{i,j} - \mathcal{S}(\emptyset)| < 2\epsilon$. Since ϵ can be arbitrarily small, the equation shows that all node pairs have the same proximity $c = \mathcal{S}(\emptyset)$. In other words, $\mathcal{S}(\cdot)$ is a trivial situation. A contradiction. □

An alternative proof by constructing connected graphs as contradictions is provided in Appendix C.2.

Since walk-based proximities are rather general and widely used in many graph mining tasks such as link prediction, Theorem 1 shows that the existing permutation-equivariant GNNs cannot handle these tasks well.

4 STOCHASTIC MESSAGE PASSING

In this section, we develop our stochastic message passing model. We first describe our framework, and then explore a linear implementation and non-linear extensions.

4.1 Stochastic Message Passing Framework

Theorem 1 indicates that a major shortcoming of permutation-equivariant GNNs is that they cannot differentiate automorphic node pairs. To solve that problem, we need to introduce some mechanism as "symmetry breaking", i.e., to enable GNNs to distinguish symmetric nodes. To achieve this goal, we sample a stochastic matrix $\mathbf{E} \in \mathbb{R}^{N \times d}$, where each element follows an i.i.d. normal distribution $\mathcal{N}(0, 1)$. We leave exploring other possible stochastic signals besides Gaussian distributions as future works. The stochastic matrix can provide signals to distinguish the nodes because they are randomly sampled without being affected by the graph automorphism. In fact, we can easily calculate that the Euclidean distance between two stochastic signals divided by a constant $\sqrt{2}$ follows a chi distribution χ_d, that is,

$$\frac{1}{\sqrt{2}} |\mathbf{E}_{i,:} - \mathbf{E}_{j,:}| \sim \chi_d, \forall i, j. \tag{9}$$

When d is reasonably large, e.g., $d > 20$, the probability of two signals being close is very low. Then, inspired by the message-passing framework, we apply a GNN on the stochastic matrix so that the nodes can exchange information of the stochastic signals,

$$\tilde{\mathbf{E}} = \mathcal{F}_{\text{GNN}}\left(\mathbf{A}, \mathbf{E}; \mathbf{W}\right). \tag{10}$$

We call $\tilde{\mathbf{E}}$ the stochastic representation of nodes. By the message-passing on the stochastic signals, $\tilde{\mathbf{E}}$ can be used to preserve node proximities (will be shown in Theorem 2 and Theorem 3 in a moment). To still allow our model to utilize node features, we concatenate $\tilde{\mathbf{E}}$ with node representations from another GNN with node features as inputs. That is,

$$\begin{gathered}\mathbf{H} = \mathcal{F}_{\text{output}}([\tilde{\mathbf{E}}, \mathbf{H}^{(L)}]) \\ \tilde{\mathbf{E}} = \mathcal{F}_{\text{GNN}}\left(\mathbf{A}, \mathbf{E}; \mathbf{W}\right), \mathbf{H}^{(L)} = \mathcal{F}_{\text{GNN}'}(\mathbf{A}, \mathbf{F}; \mathbf{W}'),\end{gathered} \tag{11}$$

where $\mathcal{F}_{\text{output}}(\cdot)$ is an aggregation function, such as a linear function or simply the identity mapping. In a nutshell, our proposed method augments the existing GNNs with a stochastic representation learned by message-passings to differentiate different nodes and preserve node proximities.

There is also a delicate choice worthy mentioning, i.e., whether the stochastic matrix $\mathbf{E}$ is fixed or resampled in each epoch. On the one hand, by fixing $\mathbf{E}$, the model can learn to memorize the stochastic representation and distinguish different nodes, but with the cost of being unable to handle nodes not seen during training. On the other hand, by resampling $\mathbf{E}$ in each epoch, the model can have a better generalization ability since the model cannot simply remember one specific stochastic matrix. However, the node representations are not fixed (but pairwise proximities are preserved; see Theorem 2). In these cases, $\tilde{\mathbf{E}}$ is more capable of handling pairwise tasks such as link prediction or pairwise node classification.

In this paper, we fix $\mathbf{E}$ for transductive datasets and resample $\mathbf{E}$ for inductive datasets (see Section 5.1 for the experimental settings and Section 5.7 for an ablation study of this design).

Time Complexity From Eq.(11), the time complexity of our framework mainly depends on the two GNNs in learning the stochastic and permutation-equivariant node representations. In this paper, we instantiate these two GNNs using simple message-passing GNNs, such as GCN [10] and SGC [13] (see Section 4.2 and Section 4.3). Thus, the time complexity of our method is the same as those models employed, which is $O(M)$, i.e., linear with respect to the number of edges. We also empirically compare the running time of different models in Section 5.8. Besides, GNN acceleration schemes such as sampling [52], [53], [54] or partitioning the graph [55] can be directly applied to our framework.

4.2 A Linear Instantiation

Based on the general framework in Eq. (11), let us explore its minimum model instantiation, i.e., a linear model.

Specifically, inspired by Simplified Graph Convolution (SGC) [13], we adopt a linear message-passing for both GNNs, i.e.,

$$\mathbf{H} = \mathcal{F}_{\text{output}}([\tilde{\mathbf{E}}, \mathbf{H}^{(L)}]) = \mathcal{F}_{\text{output}}([\tilde{\mathbf{A}}^K \mathbf{E}, \tilde{\mathbf{A}}^K \mathbf{F}]), \tag{12}$$

where $\tilde{\mathbf{A}} = (\mathbf{D} + \mathbf{I})^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})(\mathbf{D} + \mathbf{I})^{-\frac{1}{2}}$ is the normalized graph adjacency matrix with self-loops proposed in GCN [10], $\mathbf{I}$ is the identity matrix, and K is the number of propagation steps. We also set $\mathcal{F}_{\text{output}}(\cdot)$ in Eq. (12) to a linear mapping or identity mapping.

Elegantly, this simple SMP instantiation has a theoretical guarantee on preserving walk-based proximities.

Theorem 2. *An SMP in Eq.* (12) *can preserve the walk-based proximity* $\tilde{\mathbf{A}}^K(\tilde{\mathbf{A}}^K)^T$ *with high probability if the dimensionality of the stochastic matrix* d *is sufficiently large, i.e.,* $\forall \epsilon > 0$ *and* $\delta > 0$, $\exists\, d_0$ *so that for any* $d > d_0$,

$$P\left(\left|\mathbf{S}_{i,j} - \mathcal{F}_{de}\left(\mathbf{H}_{i,:}, \mathbf{H}_{j,:}\right)\right| < \epsilon\right) > 1 - \delta, \tag{13}$$

where $\mathbf{H}$ *is the node representation obtained from SMP in Eq.* (12). *The result holds for any stochastic matrix, no matter whether* $\mathbf{E}$ *is fixed or resampled during each epoch.*

Proof. Our proof is mostly based on the random projection theory. First, since we show in Theorem 1 that the permutation-equivariant representations cannot preserve any walk-based proximity, here we develop our proof assuming $\mathbf{H} = \tilde{\mathbf{E}}$. This can be easily achieved in the model by ignoring $\mathbf{H}^{(L)}$ in $\mathcal{F}_{\text{output}}([\tilde{\mathbf{E}}, \mathbf{H}^{(L)}])$. For example, if we set $\mathcal{F}_{\text{output}}(\cdot)$ as a linear function, the model can learn to set the corresponding weights for $\mathbf{H}^{(L)}$ as all-zeros and weights for $\tilde{\mathbf{E}}$ as an identity matrix.

We set the decoder function as a normalized inner product

$$\mathcal{F}_{\text{de}}\left(\mathbf{H}_{i,:}, \mathbf{H}_{j,:}\right) = \frac{1}{d}\mathbf{H}_{i,:}\mathbf{H}_{j,:}^T = \frac{1}{d}\tilde{\mathbf{E}}_{i,:}\tilde{\mathbf{E}}_{j,:}^T. \tag{14}$$

Let $\mathbf{a}_i = \tilde{\mathbf{A}}^K_{i,:}$. Recall $\tilde{\mathbf{E}} = \tilde{\mathbf{A}}^K\mathbf{E}$. Then, we have

$$\left|\mathbf{S}_{i,j} - \mathcal{F}_{\text{de}}\left(\mathbf{H}_{i,:}, \mathbf{H}_{j,:}\right)\right| = |\mathbf{a}_i\mathbf{a}_j^T - \frac{1}{d}\tilde{\mathbf{E}}_{i,:}\tilde{\mathbf{E}}_{j,:}^T| = |\mathbf{a}_i\mathbf{a}_j^T - \frac{1}{d}\mathbf{a}_i\mathbf{E}\mathbf{E}^T\mathbf{a}_j^T|.$$

Since $\mathbf{E}$ is a Gaussian random matrix, using the Johnson-Lindenstrauss lemma [44] (in the inner product preservation form, e.g., see Corollary 2.1 and its proof in [56]), $\forall 0 < \epsilon' < \frac{1}{2}$, we have

$$P\left(|\mathbf{a}_i\mathbf{a}_j^T - \frac{1}{d}\mathbf{a}_i\mathbf{E}\mathbf{E}^T\mathbf{a}_j^T| \le \frac{\epsilon'}{2}(\|\mathbf{a}_i\| + \|\mathbf{a}_j\|)\right) > 1 - 4e^{-\frac{(\epsilon'^2 - \epsilon'^3)d}{4}}.$$

By setting $\epsilon' = \frac{\epsilon}{\max_i \|\mathbf{a}_i\|}$, we have $\epsilon > \frac{\epsilon'}{2}(\|\mathbf{a}_i\| + \|\mathbf{a}_j\|)$ and

$$P\left(\left|\mathbf{S}_{i,j} - \mathcal{F}_{\text{de}}\left(\mathbf{H}_{i,:}, \mathbf{H}_{j,:}\right)\right| < \epsilon\right) > 1 - 4e^{-\frac{\left(\frac{\epsilon}{\max_i \|\mathbf{a}_i\|}^2 - \frac{\epsilon}{\max_i \|\mathbf{a}_i\|}^3\right)d}{4}},$$

which leads to the theorem by solving and setting d_0 as follows.

$$4e^{-\frac{\left(\frac{\epsilon}{\max_i \|\mathbf{a}_i\|}^2 - \frac{\epsilon}{\max_i \|\mathbf{a}_i\|}^3\right)d_0}{4}} = \delta \Rightarrow d_0 = \frac{4\log\frac{4}{\delta}\left(\max_i \|\mathbf{a}_i\|\right)^3}{\epsilon^2\max_i \|\mathbf{a}_i\| - \epsilon^3}.$$

□

Next, we show that SMP is equivalent to a permutation-equivariant GNN with certain parametrization.

Remark 1. *Suppose we adopt* $\mathcal{F}_{output}(\cdot)$ *as a linear function with the output dimensionality same as* $\mathcal{F}_{GNN'}$. *Then, Eq.* (11) *is equivalent to the permutation-equivariant* $\mathcal{F}_{GNN'}(\mathbf{A}, \mathbf{F}; \mathbf{W}')$ *if the parameters in* $\mathcal{F}_{output}(\cdot)$ *are all-zeros for* $\tilde{E}$ *and an identity matrix for* $\mathbf{H}^{(L)}$.

The result is straightforward from the definition.

Corollary 2. *For any task, Eq.* (11) *with a linear* $\mathcal{F}_{output}(\cdot)$ *in Remark 1 is at least as powerful as the permutation-equivariant* $\mathcal{F}_{GNN'}(\mathbf{A}, \mathbf{F}; \mathbf{W}')$, *i.e., the minimum training loss of using* $\mathbf{H}$ *in Eq.* (11) *is equal to or smaller than that using* $\mathbf{H}^{(L)} = \mathcal{F}_{GNN'}(\mathbf{A}, \mathbf{F}; \mathbf{W}')$.

In other words, SMP will not hinder the performance[4] even if the tasks are strictly permutation-equivariant, since the stochastic representations are concatenated with the permutation-equivariant GNNs followed by a linear mapping. In these cases, the linear SMP is equivalent to SGC [13].

Combining Theorem 2 and Corollary 2, the linear SMP instantiation in Eq. (12) is capable of handling both proximity-aware and permutation-equivariant tasks.

4.3 Non-linear Extensions

One may be curious whether a more sophisticated variant of Eq. (11) can further improve the expressiveness of SMP. There are three adjustable components in Eq. (11): two GNNs in propagating the stochastic matrix and node features, respectively, and an output function. In theory, adopting non-linear models as either component is able to enhance the expressiveness of SMP. Indeed, if we use a sufficiently expressive GNN in learning $\tilde{\mathbf{E}}$ instead of linear propagations, we can prove a more general version of Theorem 2.

Theorem 3. *For any length-L walk-based proximity, i.e.,*

$$\mathbf{S}_{i,j} = \mathcal{S}\left(\{v_i \rightsquigarrow v_j\}\right) = \mathcal{S}\left(\{v_i \rightsquigarrow v_j | len(v_i \rightsquigarrow v_j) \le L\}\right),$$

where $len(\cdot)$ *is the length of a walk, there exists an SMP variant in Eq.* (11) *with* $\mathcal{F}_{GNN}(\mathbf{A}, \mathbf{E}; \mathbf{W})$ *containing* $L + 1$ *layers (including the input layer) to preserve that proximity if the following conditions hold: (1) The stochastic matrix* $\mathbf{E}$ *contains identifiable unique signals for different nodes, i.e.* $\mathbf{E}_{i,:} \neq \mathbf{E}_{j,:}, \forall i \neq j$. *Here we assume that the Gaussian random vectors* $\mathbf{E}$ *are rounded to machine precision so that* $\mathbf{E}$ *is drawn from a countable subspace of* $\mathbb{R}$. *(2) The message-passing and updating functions in learning* $\tilde{\mathbf{E}}$ *are bijective. (3) The decoder function* $\mathcal{F}_{de}(\cdot)$ *also takes* $\mathbf{E}$ *as inputs and is universal approximation.*

Proof. The proof of the theorem is given in Appendix C.1. □

We can also adopt more advanced methods for $\mathcal{F}_{\text{output}}(\cdot)$, such as attentions or even another GNN, so that the two GNNs are more properly integrated.

Although non-linear extensions of SMP can, in theory, increase the model expressiveness, they also take a higher risk of over-fitting due to model complexity, not to mention that the computational cost also increases. In practice, we find from our ablation studies that the linear SMP instantiation in Eq. (12) works reasonably well on most of the datasets (please refer to Section 5.7 for details).

5 Experiments

In this section we report our extensive experimental studies. We first describe the experiment settings, benchmarks and baselines. Then, we use a synthetic dataset to demonstrate the simultaneous needs of both permutation-equivariance and proximity-awareness in applications, illustrating the deficiencies of the existing GNNs in preserving the two properties and the capability of our SMP method. We use

4. Similar to previous analyses such as [3], [24], we only consider the minimum training loss because the optimization landscapes and generalization gaps of deep neural networks are difficult to analyze analytically. We leave such explorations as future works.

TABLE 1
The statistics of the datasets. For Email and PPI, #Nodes and #Edges are summed over all the graphs and the experiments are conducted in an inductive setting.

Dataset	#Graphs	#Nodes	#Edges	#Features	#Classes
Grid	1	400	760	-	-
Comm	1	400	3,800	-	20
Email	7	1,005	25,571	-	42
CS	1	18,333	81,894	6,805	15
Physics	1	34,493	247,962	8,415	5
PPI	24	56,944	818,716	50	-
PPA	1	576,289	30,326,273	58	-
Cora	1	2,708	5,429	1,433	7
CiteSeer	1	3,327	4,732	3,703	6
PubMed	1	19,717	44,338	500	3

a series of benchmark tasks, including link prediction, node classification, and pairwise node classification, to comprehensively examine the capability of our SMP method against the strong baselines. Next, we conduct ablation studies of SMP. Last, we evaluate the efficiency of our method.

5.1 Experimental Setup

Except for the proof-of-concept experiment in Section 5.2, we use the following setup.

5.1.1 Datasets

We conduct experiments on the following **ten** datasets: two simulation datasets, **Grid** and **Communities** (**Comm** in abbreviation) [8], a communication dataset **Email** [8], two coauthor networks, **CS** and **Physics** [57], two protein interaction networks, **PPI** [3] and **PPA** [7], and three benchmarks, **Cora**, **CiteSeer**, and **PubMed** [58]. We summarize the statistics of datasets in Table 1 and provide datasets details in Appendix B.1.

These datasets cover a wide spectrum of application domains, various sizes, and with or without node features. Since Email and PPI contain more than one graph, we conduct experiments in an *inductive setting*, i.e., the training, validation, and testing set are split with respect to different graphs. We repeat each experiment 5 times for all datasets except for PPA (3 times for each experiment on PPA), and report the averaged results and the standard deviations after the plus-minus signs.

5.1.2 Baselines

We adopt two sets of baselines. The first set is permutation-equivariant GNNs including GCN [10], GAT [11], and SGC [13]. They are widely adopted GNN architectures. The second set contains P-GNN [8], a representative proximity-aware GNN.

In comparing with the baselines, we mainly evaluate two variants of SMP with different $\mathcal{F}_{\text{output}}(\cdot)$: SMP-Identity, i.e., $\mathcal{F}_{\text{output}}(\cdot)$ as an identity mapping, and SMP-Linear, i.e., $\mathcal{F}_{\text{output}}(\cdot)$ as a linear function. Note that both variants adopt linear message-passing functions as SGC. We conduct ablation studies with more SMP variants in Section 5.7.

For fair comparisons, we adopt the same architecture and hyper-parameters for all the methods (please refer to Appendix B.2 for details). For datasets without node features, we adopt a constant vector as the node features.

TABLE 2
The results of node classification measured in accuracy (%) on the proof-of-concept synthetic dataset. The best result and the second-best result for each task, respectively, are in bold and underlined.

Model	Node Labels		
	Community	Social Status	Both
Random	10.0	50.0	5.0
SGC	10.0±0.0	51.0±1.4	5.0±0.0
GCN	8.7 ±1.1	$\underline{91.6\pm1.8}$	$\underline{8.9\pm0.9}$
GAT	10.0±0.0	73.4±12.9	5.0±0.0
P-GNN	$\underline{64.1\pm4.8}$	54.9±9.8	5.6±1.2
SMP-Linear	**98.8±0.6**	**93.9±0.9**	**93.8±1.6**

5.2 A Proof-of-concept Experiment

We first conduct a proof-of-concept experiment to demonstrate the importance of preserving both permutation-equivariance and proximity-awareness. We generate a synthetic dataset similar to the intuition behind the example in Figure 1 as follows. First, the nodes are randomly partitioned into a set of communities. The nodes within the same community have a higher probability of forming edges than the nodes in different communities, i.e., the well-known stochastic block model [49]. Then, within each community, we generate a social status for each node with two possible choices. If a node is *active*, it has a high probability of forming edges with other nodes in the same community. Otherwise, the node has a low probability of forming edges with others, i.e., *inactive*. From the above generating process, we can see that proximity-awareness is essential to predict which community a node belongs to, since nodes within the same community have large proximities. To predict whether a node is active or inactive, permutation-equivariance is helpful, since the social status serves as a type of centrality measurements. Please refer to Appendix B.1 for further details of the synthetic dataset.

We conduct experiments on the synthetic dataset for the node classification task, i.e., predicting the node labels. We consider the following three cases. (1) **Community**: The node label is the community that the node belongs to. (2) **Social Status**: The node label is the social status of the node. (3) **Both**: The node label is the Cartesian product of (1) and (2), i.e., every community and social status pair is a distinct label. We use a softmax layer on the learned node representations as the classifier, and use accuracy, i.e., the percentage of nodes correctly classified, as the evaluation metric. We omit the results of SMP-Identity since the node representations in SMP-Identity have a fixed dimensionality that does not match the number of classes.

Table 2 shows the results, which are consistent with our analyses. The permutation-equivariant GNNs perform reasonably well on predicting the social status labels but cannot discover communities, since node proximities are not preserved in those methods. P-GNN manages to handle community labels well, but performs poorly for social status labels. None of them can handle the most challenging setting where both properties are needed to predict the node labels of community and status.

SMP performs consistently well in all three cases. The results clearly show that SMP can simultaneously preserve

TABLE 3
The results of link prediction tasks measured in AUC (%). The best result and the second-best result for each dataset, respectively, are in bold and underlined.

Model	Grid	Comm	Email	CS	Physics	PPI
SGC	57.6±3.8	51.9±1.6	68.5±7.0	96.5±0.1	**96.6±0.1**	80.5±0.4
GCN	61.8±3.6	50.3±2.5	67.4±6.9	93.4±0.3	93.8±0.2	78.0±0.4
GAT	61.0±5.5	51.1±1.6	53.5±6.3	93.7±0.9	94.1±0.4	79.3±0.5
P-GNN	73.4±6.0	97.8±0.6	70.9±6.4	82.2±0.5	Out of memory	80.8±0.4
SMP-Identity	55.1±4.8	**98.0±0.7**	72.9±5.1	96.5±0.1	96.5±0.1	81.0±0.2
SMP-Linear	**73.6±6.2**	97.7±0.5	**75.7±5.0**	**96.7±0.1**	96.1±0.1	**81.9±0.3**

TABLE 4
The results of link prediction on the PPA dataset.

Model	Hits@100
SGC	0.1187±0.0012
GCN	0.1867±0.0132
GraphSAGE	0.1655±0.0240
P-GNN	Out of Memory
Node2vec	0.2226±0.0083
Matrix Factorization	0.3229±0.0094
SMP-Identity	0.2018±0.0148
SMP-Linear	**0.3582±0.0070**

permutation-equivariance and proximity-awareness when needed and retain highly competitive performance for each property. In fact, for the community labels, SMP significantly outperforms P-GNN, demonstrating that SMP can better preserve proximities between nodes.

Next, we report experimental results on benchmark datasets.

5.3 Link Prediction

Link prediction predicts missing links in a graph. We randomly split the edges into three exclusive parts of relative sizes 80%, 10% and 10%, and use them for training, validation, and testing, respectively. Besides these positive samples, we obtain negative samples by randomly sampling an equal number of node pairs that do not have edges for training/validation/testing. For all the methods, we set a simple classifier: $\mathrm{Sigmoid}(\mathbf{H}_i^T \mathbf{H}_j)$, i.e., use the inner product to predict whether a node pair (v_i, v_j) forms a link, and use AUC (area under the ROC curve) as the evaluation metric. One exception to this setting is that on the PPA dataset, we follow the splits and evaluation metric (i.e., Hits@100) provided by the dataset [7]. Limited by space, the results for three benchmarks (Cora, CiteSeer, and PubMed) are shown in Appendix A.2.

The results except PPA are shown in Table 3. SMP achieves the best results on five out of the six datasets and is highly competitive (the second-best result) on the other (Physics). The results demonstrate the effectiveness of SMP in link prediction tasks. We attribute the strong performance of SMP to its capability of maintaining both proximity-awareness and permutation-equivariance properties.

On Grid, Communities, Email, and PPI, both SMP and P-GNN outperform the permutation-equivariant GNNs, confirming the importance of preserving node proximities. Although SMP is simpler and more efficient than P-GNN, SMP reports even better results.

When node features are available (CS, Physics, and PPI), SGC outperforms GCN and GAT. The results re-validate the findings in SGC [13] and LightGCN [59] that the non-linearity in GNNs is not necessarily indispensable. Some plausible reasons include that the additional model complexity brought by non-linear operators makes the models tend to overfit or difficult to be trained. On those datasets, SMP retains comparable performance on two coauthor graphs and shows better performance on PPI, possibly because the node features on the protein graphs are less informative than the node features on coauthor graphs for predicting links. Thus, preserving graph structure is more beneficial on PPI. As we experiment on Email and PPI in the inductive setting, the results show that SMP also can handle inductive tasks well.

The results on PPA are shown in Table 4. SMP outperforms all the baselines, showing that it can handle large-scale graphs with millions of nodes and edges. PPA is part of a recently released Open Graph Benchmark (OGB) [7]. The superior performance on PPA further demonstrates the effectiveness of SMP in link prediction.

TABLE 5
The results of node classification tasks measured by accuracy (%). The best results and the second-best results for each dataset, respectively, are in bold and underlined. OOM represents out of memory.

Model	Comm	CS	Physics	Cora	CiteSeer	PubMed
SGC	7.1±2.1	67.2±12.8	92.3±1.6	76.9±0.2	63.6±0.0	74.2±0.1
GCN	7.5±1.2	91.1±0.7	93.1±0.8	81.4±0.5	**71.3±0.5**	**79.3±0.4**
GAT	5.0±0.0	90.5±0.5	**93.1±0.4**	**82.9±0.5**	71.2±0.6	77.9±0.5
P-GNN	5.2±0.5	77.6±7.6	OOM	59.2±1.5	55.7±0.9	OOM
SMP-Linear	**99.9±0.3**	**91.5±0.8**	93.1±0.8	80.9±0.8	68.2±1.0	76.5±0.8

5.4 Node Classification

In the task of node classification, we need ground-truths in the evaluation. Thus, we only adopt datasets with node labels. Specifically, for CS and Physics, we adopt 20/30 labeled nodes per class for training/validation and the rest for testing [57]. For Comm, we adjust the number as 5/5/10 labeled nodes per class for training/validation/testing. For Cora, CiteSeer, and PubMed, we use the default splits that come with the datasets. We do not adopt Email because some graphs in the dataset are too small to show stable results. We also exclude PPI since it is a multi-label dataset. Other settings are the same as Section 5.2.

The results are shown in Table 5. SMP reports nearly perfect results on Comm. Since the node labels are generated by graph structures on Comm and there are no node features, a model has to be proximity-aware to handle Comm well. P-GNN, which shows promising results in the link prediction task, fails miserably here.

On the other five graphs, SMP reports highly competitive performance. These graphs are commonly-used benchmarks for GNNs. P-GNN, which completely ignores permutation-equivariance, performs poorly as expected.

TABLE 6
The results of pairwise node classification tasks measured in AUC (%). The best result and the second-best result for each dataset, respectively, are in bold and underlined.

Model	Comm	Email	CS	Physics	Cora	CiteSeer	PubMed
SGC	67.4±2.4	56.3±5.4	**99.8±0.0**	99.6±0.0	99.2±0.3	**95.5±0.7**	92.3±0.3
GCN	64.9±2.3	55.0±5.7	96.8±0.7	**99.7±0.1**	97.7±0.6	92.9±1.2	**94.8±0.4**
GAT	52.5±1.3	47.7±2.7	95.2±0.6	96.3±0.2	91.6±0.7	73.6±2.7	87.1±0.2
P-GNN	98.6±0.5	63.3±5.5	90.0±0.5	Out of memory	85.5±1.2	49.8±1.8	Out of memory
SMP-Identity	**98.8±0.5**	56.9±4.1	99.7±0.0	99.6±0.0	99.2±0.2	95.2±1.1	91.9±0.3
SMP-Linear	**98.8±0.5**	**74.5±4.1**	**99.8±0.0**	99.6±0.0	**99.3±0.3**	95.3±0.4	93.4±0.2

TABLE 7
The results of graph reconstruction measured in AUC (%). The best and the second-best results for each dataset, respectively, are in bold and underlined.

Model	Grid	Comm	Email	CS	Physics	PPI	Cora	CiteSeer	PubMed
SGC	74.8±0.4	65.4±1.6	71.6±0.3	66.7±0.1	66.2±0.0	76.3±0.2	56.7±9.7	58.5±0.1	71.9±0.1
GCN	73.0±0.3	63.7±1.2	72.5±0.4	75.5±0.4	76.8±0.4	79.2±0.4	68.2±3.9	69.8±8.0	77.2±2.1
GAT	59.6±1.2	52.9±1.1	56.9±1.9	57.0±1.4	59.1±0.7	61.1±1.9	57.8±1.0	63.2±1.5	58.8±0.8
P-GNN	**99.4±0.1**	97.7±0.1	85.6±0.8	**97.2±0.6**	Out of memory	85.2±0.6	**98.1±0.6**	**99.7±0.1**	Out of memory
SMP-Identity	99.2±0.1	97.5±0.1	80.0±0.3	77.1±2.3	73.7±0.3	79.5±0.2	89.7±5.7	97.1±0.8	77.0±0.1
SMP-Linear	99.1±0.1	**97.8±0.1**	**86.7±0.2**	96.3±0.2	**95.5±0.2**	**85.5±0.1**	96.3±0.1	98.2±0.1	**95.8±0.2**

In contrast, SMP manages to be competitive with the permutation-equivariant GNNs, as endorsed by Remark 1. In fact, SMP even shows better results than its counterpart, SGC, indicating that preserving proximities is also helpful.

5.5 Pairwise Node Classification

We follow P-GNN [8] and experiment on pairwise node classification, i.e., predicting whether two nodes have the same label. Compared with node classification in Section 5.4, pairwise node classification focuses more on the relation between nodes and thus more likely to require a model to be proximity-aware.

Similar to link prediction, we split the positive samples (i.e., node pairs with the same label) into an 80%-10%-10% training-validation-testing set with an equal number of randomly sampled negative pairs. For large graphs, since the possible positive samples are intractable (i.e. $O(N^2)$), we use a random subset. As we also need node labels as the ground-truth, we only conduct pairwise node classification on the datasets when node labels are available. We also exclude the results on PPI since the dataset is multi-labeled and cannot be used in a pairwise setting [8]. Similar to the link prediction task in Section 5.3, we adopt a simple inner product classifier and use AUC as the evaluation metric.

The results are shown in Table 6. We observe consistent results as the link prediction task, i.e., SMP reports the best results on four datasets and the second-best results on the other three datasets. These results again verify that SMP can effectively preserve and utilize node proximities while retaining comparable performance when the tasks are more permutation-equivariant like, e.g., on CS, Physics, and the three benchmarks (Cora, CiteSeer, and PubMed).

5.6 Graph Reconstruction

To examine whether SMP can indeed preserve node proximities, we conduct experiments on graph reconstruction [60], i.e., using the node representations learned by GNNs to reconstruct the edges of the graph. Graph reconstruction corresponds to the first-order proximity between nodes, i.e., whether two nodes directly have a connection, which is the most straightforward node proximity [61]. Specifically, following link prediction and pairwise node classification, we adopt the inner product classifier $\mathrm{Sigmoid}(\mathbf{H}_i^T \mathbf{H}_j)$ and use AUC as the evaluation metric. To control the impact of node features (i.e., many graphs exhibit assortative mixing [49], thus even models only using node features can reconstruct the edges to a certain extent), we do not use node features for all the models.

The results are reported in Table 7. The results show that SMP greatly outperforms permutation-equivariant GNNs such as GCN and GAT for the graph reconstruction task, clearly demonstrating that SMP can better preserve node proximities. P-GNN shows highly competitive results as SMP. However, similar to the other tasks, the intensive memory usage makes P-GNN unable to handle medium-scale graphs such as Physics and PubMed.

5.7 Ablation Studies

We conduct ablation studies by comparing different SMP variants, including SMP-Identity, SMP-Linear, and additional three variants.

- In SMP-MLP, we set $\mathcal{F}_{\text{output}}(\cdot)$ to a fully-connected network with one hidden layer.
- In SMP-Linear-GCN$_{\text{feat}}$, we set $\mathcal{F}_{\text{GNN}'}(\mathbf{A}, \mathbf{F}; \mathbf{W}')$ in Eq. (11) to a GCN [10], i.e., induce non-linearity in the message-passing for features. $\mathcal{F}_{\text{GNN}}(\mathbf{A}, \mathbf{E}; \mathbf{W})$ and $\mathcal{F}_{\text{output}}(\cdot)$ are still linear.
- In SMP-Linear-GCN$_{\text{both}}$, we set both GNNs in Eq. (11), i.e., $\mathcal{F}_{\text{GNN}}(\mathbf{A}, \mathbf{E}; \mathbf{W})$ and $\mathcal{F}_{\text{GNN}'}(\mathbf{A}, \mathbf{F}; \mathbf{W}')$, to a GCN [10], i.e., induce non-linearity in message-passing for both features and stochastic representations. $\mathcal{F}_{\text{output}}(\cdot)$ is linear.

TABLE 8
The ablation study of SMP variants for the link prediction task. Datasets except PPA are measured by AUC (%) and PPA is measured by Hits@100. The best result and the second-best result for each dataset are in bold and underlined, respectively.

Model	Grid	Comm	Email	CS	Physics	PPI	PPA
SMP-Identity	55.1±4.8	**98.0±0.7**	72.9±5.1	$\underline{96.5\pm0.1}$	**96.5±0.1**	81.0±0.2	0.2018±0.0148
SMP-Linear	$\underline{73.6\pm6.2}$	97.7±0.5	**75.7±5.0**	**96.7±0.1**	$\underline{96.1\pm0.1}$	$\underline{81.9\pm0.3}$	$\underline{0.3582\pm0.0070}$
SMP-MLP	72.1±4.3	$\underline{97.8\pm0.6}$	62.7±8.1	88.9±0.8	89.2±0.4	80.1±0.3	0.2035±0.0038
SMP-Linear-GCN$_{\text{feat}}$	72.8±4.2	**98.0±0.4**	$\underline{74.2\pm3.9}$	92.9±0.6	94.3±0.2	**82.3±1.0**	**0.4090±0.0087**
SMP-Linear-GCN$_{\text{both}}$	**80.5±3.9**	97.3±0.7	73.4±5.5	89.8±2.0	91.7±0.2	79.7±0.3	0.2125±0.0232

TABLE 9
The results of comparing whether the stochastic signals $\mathbf{E}$ are fixed or not during different training epochs for the link prediction task. The better of the two results are in bold.

Model	$\mathbf{E}$	Grid	Comm	CS	Physics	Email	PPI
SMP-Identity	Fixed	55.1±4.8	**98.0±0.7**	**96.5±0.1**	96.5±0.1	**75.9±3.9**	80.4±0.4
	Not Fixed	**55.2±4.1**	97.6±0.7	96.4±0.1	96.5±0.1	72.9±5.1	**81.0±0.2**
SMP-Linear	Fixed	**73.6±6.2**	**97.7±0.5**	**96.7±0.1**	96.1±0.1	71.3±3.9	71.5±0.7
	Not Fixed	64.4±2.9	97.4±0.1	96.2±0.1	96.1±0.1	**75.7±5.0**	**81.9±0.3**

TABLE 10
The average running time (milliseconds) for each epoch (including both training and testing), on the link prediction task. OOM represents out of memory.

Model	Grid	Comm	Email	CS	Physics	PPI
SGC	25	28	58	210	651	704
GCN	25	35	75	214	612	784
GAT	36	43	140	258	801	919
P-GNN	81	84	206	19,340	OOM	6,521
SMP-Identity	26	37	96	284	751	840
SMP-Linear	28	26	84	212	616	832
SMP-MLP	23	28	83	237	614	831
SMP-Linear-GCN$_{\text{feat}}$	23	29	90	231	636	855
SMP-Linear-GCN$_{\text{both}}$	34	40	95	228	626	895

We show the results of link prediction in Table 8. The results for node classification and pairwise node classification, which show similar conclusions, are provided in Appendix A.3.

In general, SMP-Linear shows impressive performance, achieving the best or second-best results on six datasets and highly competitive on the other (Comm). SMP-Identity, which does not have learnable parameters in the output function, performs slightly worse. The results demonstrate the importance of adopting a learnable linear layer in the output function, which is consistent with Remark 1. SMP-MLP does not lift the performance in general, showing that adding extra complexities in $\mathcal{F}_{\text{output}}(\cdot)$ brings no gain in those datasets. SMP-Linear-GCN$_{\text{feat}}$ reports the best results on Communities, PPI, and PPA, indicating that adding extra non-linearities in propagating node features is helpful for some graphs. SMP-Linear-GCN$_{\text{both}}$ reports the best results on Gird with a considerable margin. Recall that Grid has no node features. The results indicate that inducing non-linearities can help the stochastic representations to better capture proximities for some graphs.

We also assess the effects of whether the stochastic signals $\mathbf{E}$ are fixed or not during different training epochs for our proposed SMP. For brevity, we only report the results of link prediction in Table 9. The results show that fixing $\mathbf{E}$ usually leads to better results on transductive datasets (recall that datasets except Email and PPI are transductive) and resampling $\mathbf{E}$ leads to better results on inductive datasets. The results are consistent with our analysis in Section 4.1.

5.8 Efficiency

To compare the efficiency of different methods quantitatively, we report the running time of different methods in Table 10. The results are averaged over 3,000 epochs on an NVIDIA TESLA M40 GPU with 12 GB of memory.

The results show that SMP is computationally efficient, i.e., only marginally slower than SGC and comparable to GCN. P-GNN is at least an order of magnitude slower except for the extremely small graphs such as Grid, Communities, or Email, which have no more than a thousand nodes. In addition, the expensive memory cost makes P-GNN unable to work on large-scale graphs.

6 CONCLUSION

In this paper, we propose SMP, a general and simple GNN to preserve both proximity-awareness and permutation-equivariance. We augment the existing GNNs with stochastic node representations. We prove that SMP can enable GNNs to preserve node proximities and is equivariant to a permutation-equivariant GNN with certain parametrization. Our experimental results demonstrate the effectiveness and efficiency of SMP.

ACKNOWLEDGMENTS

This work was supported in part by National Key Research and Development Program of China (No. 2020AAA0106300), National Natural Science Foundation of China (No. U1936219, No. 62141607, No. 62050110), and Beijing Academy of Artificial Intelligence (BAAI). All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies. Peng Cui and Wenwu Zhu are corresponding authors.

REFERENCES

[1] J. Ma, C. Zhou, P. Cui, H. Yang, and W. Zhu, "Learning disentangled representations for recommendation," in *Advances in Neural Information Processing Systems 32*, 2019, pp. 5712–5723.

[2] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, "Neural relational inference for interacting systems," in *International Conference on Machine Learning*, 2018, pp. 2688–2697.

[3] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[4] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning*, 2017, pp. 1263–1272.

[5] N. Keriven and G. Peyré, "Universal invariant and equivariant graph neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 7090–7099.

[6] H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman, "Invariant and equivariant graph networks," in *International Conference on Learning Representations*, 2019.

[7] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," *Neural Information Processing Systems (NeurIPS)*, 2020.

[8] J. You, R. Ying, and J. Leskovec, "Position-aware graph neural networks," in *International Conference on Machine Learning*, 2019, pp. 7134–7143.

[9] S. P. Borgatti, "Centrality and network flow," *Social networks*, vol. 27, no. 1, pp. 55–71, 2005.

[10] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the 6th International Conference on Learning Representations*, 2017.

[11] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proceedings of the 7th International Conference on Learning Representations*, 2018.

[12] D. M. Gysi, Í. Do Valle, M. Zitnik, A. Ameli, X. Gan, O. Varol, S. D. Ghiassian, J. Patten, R. A. Davey, J. Loscalzo *et al.*, "Network medicine framework for identifying drug-repurposing opportunities for covid-19," *Proceedings of the National Academy of Sciences*, vol. 118, no. 19, 2021.

[13] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International Conference on Machine Learning*, 2019, pp. 6861–6871.

[14] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[15] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2008.

[16] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2. IEEE, 2005, pp. 729–734.

[17] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 498–511, 2009.

[18] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," in *International Conference on Learning Representations*, 2016.

[19] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," in *International Conference on Learning Representations*, 2014.

[20] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE signal processing magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[21] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, 2016, pp. 3844–3852.

[22] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5115–5124.

[23] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *International Conference on Machine Learning*, 2018, pp. 5453–5462.

[24] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations*, 2019.

[25] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner *et al.*, "Relational inductive biases, deep learning, and graph networks," *arXiv:1806.01261*, 2018.

[26] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[27] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, "Weisfeiler and leman go neural: Higher-order graph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

[28] H. Maron, H. Ben-Hamu, H. Serviansky, and Y. Lipman, "Provably powerful graph networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 2156–2167.

[29] N. Shervashidze, P. Schweitzer, E. J. v. Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels," *Journal of Machine Learning Research*, vol. 12, no. Sep, pp. 2539–2561, 2011.

[30] H. Cui, et al., "On positional and structural node features for graph neural networks on non-attributed graphs," *The Sixth International Workshop on Deep Learning on Graphs (DLG-KDD'21)*, 2021.

[31] V. P. Dwivedi, C. K. Joshi, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *arXiv preprint arXiv:2003.00982*, 2020.

[32] D. Beani, S. Passaro, V. Létourneau, W. Hamilton, G. Corso, and P. Liò, "Directional graph networks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 748–758.

[33] Z. Zhang, P. Cui, J. Pei, X. Wang, and W. Zhu, "Eigen-gnn: A graph structure preserving plug-in for gnns," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[34] D. Angluin, "Local and global properties in networks of processors," in *Proceedings of the twelfth annual ACM symposium on Theory of computing*, 1980, pp. 82–93.

[35] A. Loukas, "What graph neural networks cannot learn: depth vs width," in *International Conference on Learning Representations*, 2020.

[36] R. Murphy, B. Srinivasan, V. Rao, and B. Ribeiro, "Relational pooling for graph representations," in *International Conference on Machine Learning*, 2019.

[37] G. Dasoulas, L. Dos Santos, K. Scaman, and A. Virmaux, "Coloring graph neural networks for node disambiguation," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 7 2020, pp. 2126–2132.

[38] R. Sato, M. Yamada, and H. Kashima, "Random features strengthen graph neural networks," in *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*. SIAM, 2021, pp. 333–341.

[39] G. Corso, L. Cavalleri, D. Beaini, P. Liò, and P. Veličković, "Principal neighbourhood aggregation for graph nets," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[40] M. Fey, J. E. Lenssen, C. Morris, J. Masci, and N. M. Kriege, "Deep graph matching consensus," in *International Conference on Learning Representations*, 2020.

[41] R. Abboud, I. I. Ceylan, M. Grohe, and T. Lukasiewicz, "The surprising power of graph neural networks with random node initialization," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 8 2021, pp. 2112–2118.

[42] K. Oono and T. Suzuki, "Graph neural networks exponentially lose expressive power for node classification," in *International Conference on Learning Representations*, 2020.

[43] G. Li, M. Muller, A. Thabet, and B. Ghanem, "Deepgcns: Can gcns go as deep as cnns?" in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9267–9276.

[44] S. S. Vempala, *The random projection method*. American Mathematical Soc., 2005, vol. 65.

[45] T. Gärtner, P. Flach, and S. Wrobel, "On graph kernels: Hardness results and efficient alternatives," in *Learning theory and kernel machines*. Springer, 2003, pp. 129–143.

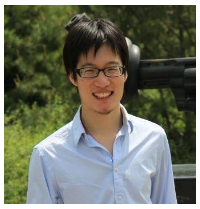
[46] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *Fifth IEEE international conference on data mining (ICDM'05)*. IEEE, 2005, pp. 8–pp.

[47] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.

[48] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.
[49] M. Newman, *Networks*. Oxford university press, 2018.
[50] Z. Zhang, P. Cui, X. Wang, J. Pei, X. Yao, and W. Zhu, "Arbitrary-order proximity preserved network embedding," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2778–2786.
[51] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *International Conference on Learning Representations*, 2019.
[52] J. Chen, J. Zhu, and L. Song, "Stochastic training of graph convolutional networks with variance reduction," in *International Conference on Machine Learning*, 2018, pp. 942–950.
[53] J. Chen, T. Ma, and C. Xiao, "Fastgcn: Fast learning with graph convolutional networks via importance sampling," in *International Conference on Learning Representations*, 2018.
[54] W. Huang, T. Zhang, Y. Rong, and J. Huang, "Adaptive sampling towards fast graph representation learning," in *Advances in neural information processing systems*, 2018.
[55] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 257–266.
[56] K. Sham and S. Greg, "Random projections," CMSC 35900 (Spring 2009) Large Scale Learning, 2020, [Online; accessed 4-September-2020]. [Online]. Available: https://ttic.uchicago.edu/~gregory/courses/LargeScaleLearning/lectures/jl.pdf
[57] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," *Relational Representation Learning Workshop, NeurIPS 2018*, 2018.
[58] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.
[59] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 639–648.
[60] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 1225–1234.
[61] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 1067–1077.
[62] D. J. Watts, "Networks, dynamics, and the small-world phenomenon," *American Journal of sociology*, vol. 105, no. 2, pp. 493–527, 1999.

Ziwei Zhang received his Ph.D. from the Department of Computer Science and Technology, Tsinghua University, in 2021. He is currently a post-doc researcher in the Department of Computer Science and Technology at Tsinghua University. His research interests focus on machine learning on graphs, including graph neural network (GNN) and network representation learning. He has published over a dozen papers in prestigious conferences and journals, including KDD, AAAI, IJCAI, NeurIPS, and TKDE.

Chenhao Niu received his B.E. degree in Computer Science and Technology from Tsinghua University, 2020. He is currently a M.Sc. student in Machine Learning at Carnegie Mellon University. His research interests are mainly in machine learning with graph neural networks and generative models, and he has related publications on KDD, AISTATS and TKDE.



Peng Cui is an Associate Professor with tenure in Tsinghua University. He got his PhD degree from Tsinghua University in 2010. His research interests include causally-regularized machine learning, network representation learning, and social dynamics modeling. He has published more than 100 papers in prestigious conferences and journals in data mining and multimedia. His recent research won the IEEE Multimedia Best Department Paper Award, SIGKDD 2016 Best Paper Finalist, ICDM 2015 Best Student Paper Award, SIGKDD 2014 Best Paper Finalist, IEEE ICME 2014 Best Paper Award, ACM MM12 Grand Challenge Multimodal Award, and MMM13 Best Paper Award. He is PC co-chair of CIKM2019 and MMM2020, SPC or area chair of WWW, ACM Multimedia, IJCAI, AAAI, etc., and Associate Editors of IEEE TKDE, IEEE TBD, ACM TIST, and ACM TOMM etc. He received ACM China Rising Star Award in 2015, and CCF-IEEE CS Young Scientist Award in 2018. He is now a Distinguished Member of ACM and CCF, and a Senior Member of IEEE.

Jian Pei is a Professor in the School of Computing Science at Simon Fraser University. He is a renown leading researcher in the general areas of data science, big data, data mining, and database systems. He is recognized as a Fellow of the Royal Society of Canada (Canada's national academy), the Canadian Academy of Engineering, ACM and IEEE. He is one of the most cited authors in data mining, database systems, and information retrieval. Since 2000, he has published one textbook, two monographs and over 300 research papers in refereed journals and conferences, which have been cited extensively by others. His research has generated remarkable impact substantially beyond academia. For example, his algorithms have been adopted by industry in production and popular open source software suites. Jian Pei also demonstrated outstanding professional leadership in many academic organizations and activities. He was the editor-in-chief of the IEEE Transactions of Knowledge and Data Engineering (TKDE) in 2013-16, the chair of ACM SIGKDD in 2017-2021, and a general co-chair or program committee co-chair of many premier conferences. He maintains a wide spectrum of industry relations with both global and local industry partners. He is an active consultant and coach for industry. He received many prestigious awards, including the 2017 ACM SIGKDD Innovation Award, the 2015 ACM SIGKDD Service Award, the 2014 IEEE ICDM Research Contributions Award, the British Columbia Innovation Council 2005 Young Innovator Award, an NSERC 2008 Discovery Accelerator Supplements Award, an IBM Faculty Award (2006), a KDD Best Application Paper Award (2008), an ICDE Influential Paper Award (2018), a PAKDD Best Paper Award (2014), and a PAKDD Most Influential Paper Award (2009).

Bo Zhang is a researcher of the Search Product Center, WeChat Recommend Product Department, Tencent Inc., China. He got his BEng degree in 2009 from School of Computer Science, Xidian University, China, and got his Master Degree in 2012 from School of Computer Science and Technology, Zhejiang University, China.

Wenwu Zhu is currently a Professor of the Computer Science Department of Tsinghua University. Prior to his current post, he was a Senior Researcher and Research Manager at Microsoft Research Asia. He was the Chief Scientist and Director at Intel Research China from 2004 to 2008. He worked at Bell Labs New Jersey as a Member of Technical Staff during 1996-1999. He received his Ph.D. degree from New York University in 1996.

He served as the Editor-in-Chief for the IEEE Transactions on Multimedia (T-MM) from January 1, 2017, to December 31, 2019. He has been serving as Vice EiC for IEEE Transactions on Circuits and Systems for Video Technology (TCSVT) and the chair of the steering committee for IEEE T-MM since January 1, 2020. His current research interests are in the areas of multimedia computing and networking, and big data. He has published over 350 papers in the referred journals and received nine Best Paper Awards including IEEE TCSVT in 2001 and 2019, and ACM Multimedia 2012. He is an IEEE Fellow, AAAS Fellow, SPIE Fellow and a member of the European Academy of Sciences (Academia Europaea).

Appendix A
Additional Experimental Results

A.1 Comparison with Using Node IDs

We compare SMP with augmenting GNNs using a one-hot encoding of node IDs, i.e., the identity matrix. Intuitively, since the IDs of nodes are unique, such a method does not suffer from the automorphism problem and should also enable GNNs to preserve node proximities. However, using such a one-hot encoding has two major problems. First, the dimensionality of the identity matrix is $N \times N$. Thus, the number of parameters in the first message-passing layer is also $O(N)$. Therefore, the method is inevitably computationally expensive and may not be scalable to large-scale graphs. Having a large number of parameters may also cause overfitting. Second, the node IDs are not transferable across different graphs, i.e., the node with ID v_1 in one graph and the node with ID v_1 in another graph do not necessarily share any similarity. Since the parameters in the message-passing depend on the node IDs as input features, using one-hot encoding cannot handle inductive tasks well.

We empirically compare such a method with SMP and report the results in Table 11. The results show that SMP-Linear outperforms GCN$_{\text{onehot}}$ in most cases. Besides, GCN$_{\text{onehot}}$ fails to handle Physics, which is only a medium-scale graph, due to the heavy memory usage. One surprising result is that GCN$_{\text{onehot}}$ outperforms SMP-Linear on Grid, the simulated graph where nodes are placed on a 20×20 grid. A plausible reason is that since the edges in Grid follow a specific rule, using a one-hot encoding gives GCN$_{\text{onehot}}$ enough flexibility to learn the rules, and the model does not overfit because the graph has a rather small scale.

One may wonder whether SMP is transferable across different graphs, since the stochastic features are independently drawn. Empirically, we find that SMP reports reasonably good performance on inductive datasets, such as Email and PPI. One plausible reason is that, since the proximities of nodes are preserved even the random features per se are different (see Theorem 2 and Theorem 3) , all subsequent parameters based on proximities can be transferred.

Besides, only using the stochastic representation of SMP in Eq. (10) can be regarded as combining node IDs while fixing the first-layer parameters during training (assuming the first message-passing layer takes the form of matrix multiplication between node features and the weight matrix, as in SGC, GCN, and GAT). By fixing the parameters, both the computational bottlenecks and transferability problems are resolved.

A.2 Additional Link Prediction Results

We further report the link prediction results on three GNN benchmarks: Cora, CiteSeer, and PubMed. The results in Table 12 show similar trends as other datasets presented in Section 5.3, i.e., SMP reports comparable results to the other permutation-equivariant GNNs while P-GNN cannot handle the task well.

A.3 Additional Ablation Studies

We report ablation study results for the node classification task and pairwise node classification task in Table 13 and Table 14, respectively. The results again show that SMP-Linear generally achieves good-enough results on the majority of the datasets, and adding non-linearity does not consistently lift the performance.

A.4 Comparison with Position Encoding

To compare the effectiveness of our proposed method with position encodings of GNNs, we adopt one recent method EigenGNN [33], which uses the eigenvectors of a graph structure matrix to enhance the existing GNNs in preserving graph structures. Specifically, we use GCN as the base architecture for EigenGNN. All hyper-parameters are kept consistently to ensure a fair comparison. The results for the link prediction task are shown in Table 15. Notice that we omit the results on PPA since calculating eigenvectors for extremely large-scale graphs is infeasible.

From the table, we can observe that SMP-Linear outperforms or is comparable to EigenGNN on eight of nine datasets, demonstrating the effectiveness of SMP. One exception is on Grid, where EigenGNN shows remarkably better results. We attribute this to the fact that Grid is an especially regular simulated graph (recall that nodes are placed on a 20×20 grid) and therefore EigenGNN can better capture this pattern. Besides, both EigenGNN and SMP greatly outperform GCN in most cases, demonstrating the importance of preserving proximities for GNNs.

Appendix B
Details for Reproducibility

B.1 Datasets

- Proof-of-concept synthetic dataset: the dataset is generated using an add-on version of the stochastic block model [49]. The graph has 400 nodes in 10 communities. Each node has an independent equal chance to be active or inactive. The probability of forming edges is (1) 0.8; (2) 0.5; (3) 0.2; and (4) 0.005, respectively, between (1) two active nodes in the same community; (2) an active node and an inactive one in the same community; (3) two inactive nodes in the same community; and (4) two nodes in different communities. There is no node feature.
- **Grid** [8]: A simulated 2D grid graph with size 20×20.
- **Communities** [8]: A simulated caveman graph [62] composed of 20 communities, each community containing 20 nodes. The graph is perturbed by rewiring 1% edges randomly. It has no node feature. The label of each node indicates the community that the node belongs to.
- **Email**[5] [8]: Seven real-world email communication graphs. Each graph has six communities and each node has an integer label indicating the community that the node belongs to.
- **Coauthor Networks**[6] [57]: Two networks from Microsoft academic graph in CS and Physics with nodes representing authors and edges representing co-authorships. The node features are embeddings of the paper keywords of the authors.
- **PPI**[5] [3]: 24 protein-protein interaction networks. Each node has a 50-dimensional feature vector.

5. https://github.com/JiaxuanYou/P-GNN/tree/master/data
6. https://github.com/shchur/gnn-benchmark/tree/master/data/npz/

TABLE 11
The results of comparing SMP with using one-hot node IDs. OOM indicates out of memory. "—" indicates that we do not adopt the dataset for the task because the dataset does not have ground-truth labels.

Task	Model	Grid	Comm	Email	CS	Physics	PPI	Cora	CiteSeer	PubMed
Link Prediction	GCN_onehot	91.5±2.1	98.3±0.7	71.2±3.5	93.1±1.3	OOM	78.6±0.3	86.8±1.5	81.7±1.1	89.4±0.5
	SMP-Linear	73.6±6.2	97.7±0.5	75.7±5.0	96.7±0.1	96.1±0.1	81.9±0.3	92.7±0.7	92.6±1.0	95.4±0.2
Pairwise Node Classification	GCN_onehot	—	98.9±0.5	67.3±5.6	97.6±0.2	OOM	—	98.2±0.3	94.4±1.2	98.9±0.1
	SMP-Linear	—	98.8±0.5	74.5±4.1	99.8±0.0	99.6±0.0	—	99.3±0.3	95.3±0.4	93.4±0.2
Node Classification	GCN_onehot	—	99.6±1.0	—	86.9±1.5	OOM	—	77.6±1.1	57.7±5.8	74.9±0.6
	SMP-Linear	—	99.9±0.3	—	91.5±0.8	93.1±0.8	—	80.9±0.8	68.2±1.0	76.5±0.8

TABLE 12
The results of link prediction measured in AUC (%) on three benchmarks. The best result and the second-best result for each dataset, respectively, is in bold and underlined.

Model	Cora	CiteSeer	PubMed
SGC	**93.6±0.6**	**94.7±0.8**	**95.8±0.2**
GCN	90.6±1.0	78.2±1.7	92.4±0.9
GAT	88.5±1.2	87.8±1.0	89.2±0.8
P-GNN	75.4±2.3	70.6±1.1	Out of memory
SMP-Identity	93.0±0.6	92.9±0.5	94.5±0.3
SMP-Linear	92.7±0.7	92.6±1.0	95.4±0.2
SMP-MLP	82.8±0.9	80.7±1.1	88.0±0.6
$\text{SMP-Linear-GCN}_\text{feat}$	86.7±1.4	81.1±1.4	90.5±0.6
$\text{SMP-Linear-GCN}_\text{both}$	80.1±2.5	80.0±2.0	81.1±2.0

- **PPA**[7] [7]: A network representing biological associations between proteins from 58 species. The node features are one-hot vectors of the species that the proteins are taken from.
- **Cora**, **CiteSeer**, **PubMed**[8] [58]: Three citation graphs where the nodes correspond to papers and the edges correspond to citations between papers. The node features are bag-of-words and the node labels are the ground truth topics of the papers.

B.2 Hyper-parameters

- All datasets except PPA: we uniformly set the number of hidden layers to 2 for all methods, i.e., two message-passing steps, and set the dimensionality of hidden layers to 32, i.e., $\mathbf{H}^{(l)} \in \mathbb{R}^{N\times 32}$, for $1 \le l \le L$ (for GAT, we use 4 heads with each head containing 8 units). We use the Adam optimizer with an initial learning rate of 0.01 and decay the learning rate by 0.1 at epoch 200. The weight decay is 5e-4. We train the model for 1,000 epochs and evaluate the model every 5 epochs. We report the testing performance at the epoch which achieves the best validation performance. For SMP, the dimensionality of the stochastic matrix is $d = 32$. For P-GNN, we use the P-GNN-F version, which uses the truncated 2-hop shortest path distance instead of the exact shortest distance for scalability.
- PPA: as suggested in the original paper [7], we set the number of GNN layers to 3 with each layer containing 256 hidden units, and add a three-layer MLP after taking the Hadamard product between pair-wise node embeddings as the predictor, i.e., $\text{MLP}(\mathbf{H}_i \odot \mathbf{H}_j)$. We use the Adam optimizer with an initial learning rate of 0.01. We set the number of epochs for training to 40, evaluate the results on validation sets every epoch, and report the testing results using the model with the best validation performance. We find that the dataset has issues with exploding gradients, and thus adopt a gradient clipping strategy by limiting the maximum p2-norm of gradients to 1.0. The dimensionality of the stochastic matrix in SMP is $d = 64$.

B.3 Hardware and Software Configurations

All experiments are conducted on a server with the following configurations.

- Operating System: Ubuntu 18.04.1 LTS
- CPU: Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz
- GPU: NVIDIA TESLA M40 with 12 GB of memory
- Software: Python 3.6.8, PyTorch 1.4.0, PyTorch Geometric 1.4.3, NumPy 1.18.1, Cuda 10.1

APPENDIX C PROOF

C.1 Proof of Theorem 3

Proof. Similar to the proof of Theorem 2, we only utilize $\tilde{\mathbf{E}}$ in our proof. Denote by $\mathbf{e}_i^{(l)}, 0 \le l \le L$, the representations of node v_i in the l^{th} layer of $\mathcal{F}_\text{GNN}(\mathbf{A}, \mathbf{E}; \mathbf{W})$, i.e., $\mathbf{e}_i^{(0)} = \mathbf{E}_{i,:}$ and $\mathbf{e}_i^{(L)} = \tilde{\mathbf{E}}_{i,:}$. Our proof strategy is to show that the stochastic node representations can remember all the information about the walks, which can be decoded by the decoder $\mathcal{F}_\text{de}(\cdot)$.

First, as the message-passing and the updating function are bijective by assumption, we can recover from the node representations in each layer all their neighborhood representations in the previous layer. Specifically, there exists $\mathcal{F}^{(l)}(\cdot)$, $1 \le l \le L$, such that

$$\mathcal{F}^{(l)}\left(\mathbf{e}_i^{(l)}\right) = \left[\mathbf{e}_i^{(l-1)}, \left\{\mathbf{e}_j^{(l-1)}, j \in \mathcal{N}_i\right\}\right]^9. \tag{15}$$

To keep our notations clear, we split the function into two parts, one for the node itself and the other one for its neighbors

$$\mathcal{F}_\text{self}^{(l)}\left(\mathbf{e}_i^{(l)}\right) = \mathbf{e}_i^{(l-1)}, \mathcal{F}_\text{neighbor}^{(l)}\left(\mathbf{e}_i^{(l)}\right) = \left\{\mathbf{e}_j^{(l-1)}, j \in \mathcal{N}_i\right\}. \tag{16}$$

7. https://snap.stanford.edu/ogb/data/linkproppred/ppassoc.zip
8. https://github.com/kimiyoung/planetoid/tree/master/data

9. To let $\mathcal{F}^{(l)}(\cdot)$ output a set with arbitrary lengths, we can adopt sequence-based models such an LSTM.

TABLE 13
The ablation study of different SMP variants for the node classification task. The best results and the second-best results are in bold and underlined, respectively.

Model	Comm	CS	Physics	Cora	CiteSeer	PubMed
SMP-Linear	99.9±0.3	**91.5±0.8**	**93.1±0.8**	**80.9±0.8**	**68.2±1.0**	76.5±0.8
SMP-MLP	**100.0±0.2**	90.1±0.5	92.3±0.8	79.3±0.8	67.0±1.5	76.8±0.9
SMP-Linear-GCN$_{\text{feat}}$	**100.0±0.0**	89.8±0.7	92.9±0.8	78.9±1.2	67.8±0.6	**77.3±0.6**
SMP-Linear-GCN$_{\text{both}}$	**100.0±0.2**	77.4±4.2	87.1±3.5	69.2±2.5	49.8±3.1	68.1±4.1

TABLE 14
The ablation study of different SMP variants for the pairwise node classification task. The best results and the second-best results are in bold and underlined, respectively.

Model	Comm	Email	CS	Physics	Cora	CiteSeer	PubMed
SMP-Identity	98.8±0.5	56.9±4.1	99.7±0.0	**99.6±0.0**	99.2±0.2	95.2±1.1	91.9±0.3
SMP-Linear	98.8±0.5	**74.5±4.1**	**99.8±0.0**	**99.6±0.0**	**99.3±0.3**	**95.3±0.4**	93.4±0.2
SMP-MLP	98.7±0.3	65.4±6.3	94.3±0.6	97.6±0.4	90.3±3.0	67.7±13.7	93.4±0.4
SMP-Linear-GCN$_{\text{feat}}$	**99.0±0.4**	60.2±9.3	95.6±0.7	98.3±0.7	96.1±0.7	88.8±1.6	**94.8±0.2**
SMP-Linear-GCN$_{\text{both}}$	98.8±0.4	61.6±6.0	95.2±0.7	97.8±0.8	94.3±1.9	83.5±3.9	94.1±0.7

TABLE 15
The results of comparing SMP with EigenGNN for the link prediction task measured by AUC. The best result for each dataset is in bold.

Model	Grid	Comm	Email	CS	Physics	PPI	Cora	CiteSeer	PubMed
GCN	61.8±3.6	50.3±2.5	67.4±6.9	93.4±0.3	93.8±0.2	78.0±0.4	90.6±1.0	78.2±1.7	92.4±0.9
EigenGNN	**92.0±0.7**	**97.7±0.3**	74.7h±3.1	93.2±0.3	93.4±0.8	81.7±0.4	88.7±0.9	83.0±1.3	91.0±0.3
SMP-Linear	73.6±6.2	**97.7±0.5**	**75.7±5.0**	**96.7±0.1**	**96.1±0.1**	**81.9±0.3**	**92.7±0.7**	**92.6±1.0**	**95.4±0.2**

For the first function, if we successively apply such functions from the l^{th} layer to the input layer, we can recover the input features of the GNN, i.e., $\mathbf{E}$. Since the stochastic matrix $\mathbf{E}$ contains an identifiable unique signal for each node, we can decode the node ID from $\mathbf{e}_i^{(0)}$, i.e., there exists $\mathcal{F}_{\text{self}}^{(0)}\left(\mathbf{e}_i^{(0)};\mathbf{E}\right)=i$. We denote the process of applying such $l+1$ functions to get the node ID as

$$\mathcal{F}_{\text{self}}^{(0:l)}\left(\mathbf{e}_i^{(l)}\right)=\mathcal{F}_{\text{self}}^{(0)}\left(\mathcal{F}_{\text{self}}^{(1)}\left(\ldots\left(\mathcal{F}_{\text{self}}^{(l)}\left(\mathbf{e}_i^{(l)}\right)\right)\right);\mathbf{E}\right)=i. \tag{17}$$

For the second function, we can apply $\mathcal{F}_{\text{neighbor}}^{(l-1)}(\cdot)$ to the decoded vector set so that we can recover their neighborhood representations in the $(l-2)^{th}$ layer. Similar procedures can be performed successively until we reach the first layer.

Next, we show that for $\mathbf{e}_j^{(l)}$, there exists a length-l walk $v_i \rightsquigarrow v_j = (v_{a_1}, v_{a_2}, ..., v_{a_{l+1}})$, where $v_{a_1}=v_i$, $v_{a_{l+1}}=v_j$, if and only if $\mathcal{F}_{\text{self}}^{(0:l)}\left(\mathbf{e}_j^{(l)}\right)=a_l=j$ and there exist $\mathbf{e}^{(l-1)}, ..., \mathbf{e}^{(0)}$ such that:

$$\begin{aligned}
\mathbf{e}^{(l-1)} &\in \mathcal{F}_{\text{neighbor}}^{(l)}\left(\mathbf{e}_j^{(l)}\right), \mathcal{F}_{\text{self}}^{(0:l-1)}\left(\mathbf{e}^{(l-1)}\right)=a_l,\\
\mathbf{e}^{(l-2)} &\in \mathcal{F}_{\text{neighbor}}^{(l-1)}\left(\mathbf{e}^{(l-1)}\right), \mathcal{F}_{\text{self}}^{(0:l-2)}\left(\mathbf{e}^{(l-2)}\right)=a_{l-1},\\
&\ldots\\
\mathbf{e}^{(0)} &\in \mathcal{F}_{\text{neighbor}}^{(1)}\left(\mathbf{e}^{(1)}\right), \mathcal{F}_{\text{self}}^{(0:0)}\left(\mathbf{e}^{(0)}\right)=a_1=i.
\end{aligned} \tag{18}$$

This result is easy to obtain as follows.

$$\begin{aligned}
&(v_{a_1}, v_{a_2}, ..., v_{a_{l+1}}) \text{ is a walk}\\
\Leftrightarrow &\mathcal{E}_{a_i,a_{i+1}}=\mathcal{E}_{a_{i+1},a_i}=1, \forall 1\le i\le l \Leftrightarrow a_i\in\mathcal{N}_{a_{i+1}}, \forall 1\le i\le l\\
\Leftrightarrow &\exists \mathbf{e}^{(i-1)}\in\mathcal{F}_{\text{neighbor}}^{(i)}\left(\mathbf{e}^{(i)}\right), \mathcal{F}_{\text{self}}^{(0:i-1)}\left(\mathbf{e}^{(i-1)}\right)=a_i, \forall 1\le i\le l.
\end{aligned}$$

Note that all the information is encoded in $\tilde{\mathbf{E}}$, i.e., we can decode $\{v_i \rightsquigarrow v_j | \text{len}(v_i \rightsquigarrow v_j) \le L\}$ from $\mathbf{e}_j^{(L)}$ by successively applying $\mathcal{F}_{\text{self}}^{(l)}(\cdot)$ and $\mathcal{F}_{\text{neighbor}}^{(l)}(\cdot)$. We can also apply $\mathcal{F}_{\text{self}}^{(0:L)}$ to $\mathbf{e}_i^{(L)}$ to get the start node ID i. Putting all together, we have

$$\mathcal{F}\left(\mathbf{e}_j^{(L)}, \mathbf{e}_i^{(L)}\right)=\{v_i \rightsquigarrow v_j | \text{len}(v_i \rightsquigarrow v_j)\le L\}, \tag{19}$$

where $\mathcal{F}(\cdot)$ is composed of $\mathcal{F}_{\text{self}}^{(l)}(\cdot)$, $0\le l\le L$, and $\mathcal{F}_{\text{neighbor}}^{(l)}(\cdot)$, $1\le l\le L$. Applying the proximity function $\mathcal{S}(\cdot)$, we have:

$$\mathcal{S}\left(\mathcal{F}\left(\mathbf{e}_j^{(L)}, \mathbf{e}_i^{(L)}\right)\right)=\mathbf{S}_{i,j}. \tag{20}$$

We finish the proof by setting the real decoder function $\mathcal{F}_{\text{de}}(\cdot)$ to arbitrarily approximate this desired function $\mathcal{S}\left(\mathcal{F}(\cdot,\cdot)\right)$ under the universal approximation assumption. □

C.2 An Alternative Proof of Theorem 1

Some may find that our proof of Theorem 1 in the main paper leads to multiple connected components in the constructed graph G'. Next, we give an alternative proof maintaining one connected component in G' assuming the original graph G is connected) under an additional assumption that the walk-based proximity is of finite length.

Proof. Similar to the previous proof, we will prove by contraction. We assume there exists a non-trivial $\mathcal{S}(\cdot)$ that a permutation-equivariant GNN can preserve. Besides, we assume the length of $\mathcal{S}(\cdot)$ is upper bounded by $l_{\max}$, where $l_{\max}$ is any finite number, i.e., $\forall i, j$,

$$\mathbf{S}_{i,j} = \mathcal{S}\left(\{v_i \rightsquigarrow v_j\}\right) = \mathcal{S}\left(\{v_i \rightsquigarrow v_j | \text{len}(v_i \rightsquigarrow v_j) \leq l_{\max}\}\right), \tag{21}$$

where $\text{len}(v_i \rightsquigarrow v_j)$ is the length, i.e., number of nodes minus 1, of the walk $v_i \rightsquigarrow v_j$.

For a connected graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{F})$, we create $G' = (\mathcal{V}', \mathcal{E}', \mathbf{F}')$ that has automorphism. Then we will show that any GNN cannot preserve $\mathcal{S}(\cdot)$ on G'. Specifically, denoting $\tilde{N} = N + l_{\max}$, we let G' have $3\tilde{N}$ nodes so that:

$$\mathcal{E}'_{i,j} = \begin{cases} \mathcal{E}_{i,j} & \text{if } i, j \leq N \\ 1 & \text{if } N \leq i, j \leq \tilde{N}+1, |j-i| = 1 \\ \mathcal{E}_{i-\tilde{N}, j-\tilde{N}} & \text{if } \tilde{N} < i, j \leq \tilde{N}+N \\ 1 & \text{if } \tilde{N}+N \leq i, j \leq 2\tilde{N}+1, |j-i| = 1 \\ \mathcal{E}_{i-2\tilde{N}, j-2\tilde{N}} & \text{if } 2\tilde{N} < i, j \leq 2\tilde{N}+N \\ 1 & \text{if } 2\tilde{N}+N \leq i, j, |j-i| = 1 \\ 1 & \text{if } i = 3\tilde{N}, j = 1 \text{ or } j = 3\tilde{N}, i = 1 \\ 0 & \text{else} \end{cases},$$

$$\mathbf{F}'_{i,:} = \begin{cases} \mathbf{F}_{i,:} & \text{if } i \leq N \\ 0 & \text{if } N < i \leq \tilde{N} \\ \mathbf{F}_{i-\tilde{N},:} & \text{if } \tilde{N} < i \leq \tilde{N}+N \\ 0 & \text{if } \tilde{N}+N < i \leq 2\tilde{N} \\ \mathbf{F}_{i-2\tilde{N},:} & \text{if } 2\tilde{N} < i \leq 2\tilde{N}+N \\ 0 & \text{if } 2\tilde{N}+N < i \end{cases}. \tag{22}$$

Intuitively, we create three "copies" of G and three "bridges" to connect the copies and thus make G' also connected. It is also easy to see that nodes v'_i, $v'_{i+\tilde{N}}$, and $v'_{i+2\tilde{N}}$ all form automorphic node pairs. Therefore, we have:

$$\mathbf{H}'^{(L)}_{i,:} = \mathbf{H}'^{(L)}_{i+\tilde{N},:} = \mathbf{H}'^{(L)}_{i+2\tilde{N},:}, \forall i \leq \tilde{N}. \tag{23}$$

Next, we see that the nodes in G' are divided into six parts (three copies and three bridges), which we denote as $\mathcal{V}'_1 = \{v_1 ... v_N\}$, $\mathcal{V}'_2 = \{v_{N+1} ... v_{\tilde{N}}\}$, $\mathcal{V}'_3 = \left\{v_{\tilde{N}+1} ... v_{\tilde{N}+N}\right\}$, $\mathcal{V}'_4 = \left\{v_{\tilde{N}+N+1}, ..., v_{2\tilde{N}}\right\}$, $\mathcal{V}'_5 = \left\{v_{2\tilde{N}+1}, ..., v_{2\tilde{N}+N}\right\}$, and $\mathcal{V}'_6 = \left\{v_{2\tilde{N}+N+1}, ..., v_{3\tilde{N}}\right\}$. Since $\mathcal{V}'_2$, $\mathcal{V}'_4$, $\mathcal{V}'_6$ are bridges with a length $l_{\max}$, any walk crosses these bridges will have a length large than $l_{\max}$. For example, let us focus on $v_i \in \mathcal{V}'_1$, i.e., $1 \leq i \leq N$. If v_j is in $\mathcal{V}'_3$, $\mathcal{V}'_4$, or $\mathcal{V}'_5$ (i.e., $\tilde{N} < j \leq 2\tilde{N}+N$), any walk $v_i \rightsquigarrow v_j$ will either cross the bridge $\mathcal{V}'_2$ or $\mathcal{V}'_6$ and thus has a length larger than $l_{\max}$. As a result, we have:

$$\mathbf{S}_{i,j} = \mathcal{S}\left(\{v_i \rightsquigarrow v_j\}\right) = \mathcal{S}\left(\{v_i \rightsquigarrow v_j | \text{len}(v_i \rightsquigarrow v_j) \leq l_{\max}\}\right) = \mathcal{S}\left(\emptyset\right). \tag{24}$$

If $v_j \in \mathcal{V}'_1$ or $v_j \in \mathcal{V}'_2$, i.e., $j \leq \tilde{N}$, we can use the fact that v_j and $v_{j+\tilde{N}}$ forms an automorphic node pair, i.e., $\forall \epsilon > 0$, we have

$$\begin{aligned} |\mathbf{S}_{i,j} - \mathcal{S}(\emptyset)| &\leq \left|\mathbf{S}_{i,j} - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j,:}\right)\right| + \left|\mathcal{S}(\emptyset) - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j,:}\right)\right| \\ &= \left|\mathbf{S}_{i,j} - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j,:}\right)\right| + \left|\mathbf{S}_{i,j+\tilde{N}} - \mathcal{F}_{\text{de}}\left(\mathbf{H}'^{(L)}_{i,:}, \mathbf{H}'^{(L)}_{j+\tilde{N},:}\right)\right| < 2\epsilon. \end{aligned} \tag{25}$$

Similarly, if $v_j \in \mathcal{V}'_6$, i.e., $2\tilde{N}+N < j$, we can use the fact that v_j and $v_{j-\tilde{N}}$ forms an automorphic node pair to prove the same inequality. Thus, we prove that if $i \leq N, \forall \epsilon > 0, \forall j$, $|\mathbf{S}_{i,j} - \mathcal{S}(\emptyset)| < 2\epsilon$. The same proof strategy can be applied to $i > N$. Since ϵ can be arbitrarily small, the results show that all node pairs have the same proximity $\mathcal{S}(\emptyset)$, which leads to a contraction and finishes our proof. □