Summary of changes to reviewer UzmR's concerns:

- Q 1.1&1.2: Reorganized the terminology as "neuron"→"pseudo node" [Section 3.2: Lines(R)210-(R)218] and "functional features/features"→"message" [Section 3.3: Lines(R)252-(R)264].
- Q 1.3: Added explicit reference to $h^{(0)}$, $h^{(1)}$, $h^{(2)}$. [Section 4.1: Line(R)310-(R)313]
- Q 1.4: Changed: "Homogeneous"→"homophilic", and "heterogeneous"→"heterophilic". [Abstract Line94]
- Q 2: Evaluated the necessity of distinguishing states and functional features. [same location as Q 1.1&1.2], [Appendix C.6]

Summary of changes to reviewer URo6's concerns:

- Q 1.1: Removed "expressive" except for related work section.
- Q 1.2: Changed: "expressive"→"graph". [Abstract Line66]
- Q 1.3(1): Changed: "dilemma"→"problem". [Abstract Line72]
- Q 1.3(2)&1.4: Removed "expressive". Added the related work. [Section 1 Lines(R)85-(R)89]
- Q 1.5-1.7: Removed heavily technical details. [Section 1 Lines112-133]
- Q 2.1: Added related works. [Section 2.1 Lines155-(R)113, Lines(R)140-(R)148]
- Q 2.2: Added "in practice" to distinguish from the theoretical case. [Section 2.1 Line(R)125]
- Q 2.3: Added reference for "the uniform connected ones". [Section 2.1 Lines199-201]
- Q 3.1&3.2: Removed discussion in Lines 136-141 and unused definition in Lines 160-161. (Lines refer to the original version)
- Q 3.3: Added background discussion between DyN/$\mathbf{N^2}$ and RNN/SSM. [Appendix A]
- Q 3.4: Improved the illustration. [Fig. 2]
- Q 3.5: Added more details for "Common State Space". [Section 3.2]
- Q 4: Added more evaluation results. [Section 5.3.1]

Summary of changes to reviewer 68jQ's concerns:

- Q 1: Added concise implementation description. [Section 4 Lines(R)282-(R)287]
- Q 2.1: Added ablation on pseudo nodes. [Appendix C.7]
- Q 2.2: Added intuitions for the implementation. [Appendix D]
- Q 3: Fixed minor typos and errors.
- Q 6: Unified the notations $\mathbf{X}$ and $\mathbf{S}$ as $\mathbf{M}$. [Section 4]

Summary of changes to reviewer fYCj's concerns:

- Q 1.1: Reorganized the terminology as "neuron"→"pseudo node" [Section 3.2 Lines(R)210-(R)218].
- Q 1.2: Added more motivation for the paper. [Section 3 Lines205-211, (R)172-(R)180]
- Q 1.3: Added background for Pei and Wang. [Section 3 Lines(R)172-(R)180, (R)191-(R)195]
- Q 2: Added citation for Differentiable Graph Modules (DGMs). [Section 2.1 Lines(R)144-(R)148]
- Q 3: Added extra evaluation on Long-Range Graph Benchmark. [Appendix C.1]

Towards Dynamic Message Passing on Graphs

Anonymous Authors[1]

Abstract

Message passing in graph neural networks (GNNs) plays a vital role in extracting graph features. However, the over-reliance on input topology diminishes the efficacy of message passing and restricts the representation ability of GNNs. Despite efforts made to mitigate the reliance, existing study encounters over-squashing or high computational expense problems, which invokes the demands for flexible message passing with low complexity. In this paper, we propose a novel dynamic message-passing mechanism for GNNs. It projects graph nodes and pseudo nodes into a common space with measurable spatial proximity between them. The measured proximity can be interpreted as pseudo edges, empowering global communication between graph nodes to proceed intermediately through pseudo nodes under linear complexity. As nodes move in the space, their spatial proximity changes accordingly, resulting in dynamic message-passing pathways. We further develop a series of GNNs named $\mathbf{N^2}$ based on our dynamic message-passing mechanism. $\mathbf{N^2}$ employs a single recurrent layer to recursively generate the displacements of pseudo and graph nodes, learning optimal dynamic pathways. Evaluation on eighteen benchmarks demonstrates the superior performance of $\mathbf{N^2}$ over popular GNNs. $\mathbf{N^2}$ successfully scales to large-scale benchmarks and achieves consistent performance on both homophilic and heterophilic graphs.

1. Introduction

The inherent irregular structure of graphs poses nontrivial challenges in graph representation learning (Zhou et al., 2020). To enable effective learning on graphs, Graph Neural Networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018) have been specifically designed for graph-structured data. In GNNs, message passing (Gilmer et al., 2017; Veličković, 2022) plays a vital role in extracting graph features. The vanilla message passing (Kipf & Welling, 2017; Wu et al., 2019) applies node-central aggregation and is constrained between the adjacent nodes. During node-central aggregation, the message-passing process accesses the neighbors in an isotropic manner (Huang et al., 2022) and aggregates multi-hop features iteratively. In consequence, the message passing relies heavily on the input graph structure, which leads to GNNs with over-smoothing (Oono & Suzuki, 2021) or over-squashing (Alon & Yahav, 2021; Di Giovanni et al., 2023a;b).

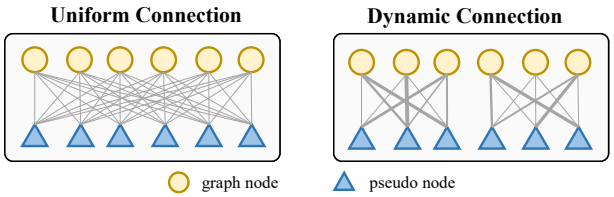


Figure 1. **Comparison between Different Connection Patterns for Pseudo Nodes.** Pseudo nodes are incorporated as alternative message-passing pathways on graphs, which follow certain patterns to connect with all the graph nodes.

To further improve the effectiveness of GNNs for graph representation learning, one direct solution is to decouple message-passing pathways from the input graph structure. Along this line, methods (Franceschi et al., 2019; Deac et al., 2022) have been proposed to extend message passing beyond input graph structure to enable global information exchange. Some methods directly model pairwise relation (Ying et al., 2021; Kreuzer et al., 2021; Mialon et al.), but the dense relation modeling causes high computational and space complexity. Other methods incorporate pseudo nodes connected with all the input nodes (Gilmer et al., 2017; Hwang et al., 2021; Liu et al., 2022; Shirzad et al., 2023). However, these models follow a uniform connection pattern, *i.e.*, each pseudo node employs identical edge weights to connect with all the graph nodes (as illustrated in Fig. 1). An overwhelming number of node features are squashed equally into pseudo nodes, leading to less discriminative representations for downstream tasks. As a result, pseudo nodes with uniform connection become the bottlenecks of message passing (Shirzad et al., 2023).

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

All these limitations in the graph space indicate the importance of a more flexible message-passing fashion with low complexity. In this paper, we propose a novel dynamic message-passing mechanism for GNNs. Our mechanism aims to dynamically model the specific relations between pairs of graph nodes and pseudo nodes (as illustrated in Fig. 1). To achieve this, both graph nodes and pseudo nodes are embedded in a common space with measurable spatial relation between them. Regarding the measured relation as pseudo edges, the message passing on the input graphs can be extended to pseudo nodes. As a result, graph nodes can communicate with each other intermediately through pseudo nodes. Based on our dynamic graph **N**odes-pseudo **N**odes mechanism for message passing, we further develop a series of GNNs named $\mathbf{N^2}$. $\mathbf{N^2}$ incorporates a recurrent layer to parameterize the displacements of graph nodes and pseudo nodes in the common space. With both graph nodes and pseudo nodes moving in the common space, their spatial relations change accordingly, different from the prior uniform connection pattern (Gilmer et al., 2017; Hwang et al., 2021; Liu et al., 2022; Shirzad et al., 2023). Notably, as graph nodes perform global message passing intermediately through pseudo nodes, our method requires lower complexity by avoiding fully connected computation (Ying et al., 2021; Kreuzer et al., 2021; Mialon et al.). Our contributions are summarized as:

- We propose a dynamic message-passing mechanism for GNNs. By measuring the specific spatial proximity between graph and pseudo nodes in common space to construct the dynamic pathways, the message passing is achieved flexibly with linear complexity.
- We develop a series of GNNs named $\mathbf{N^2}$, which employs a recurrent layer to parameterize the evolutionary displacements in the common space, empowering evolving dynamic message-passing pathways.
- We demonstrate the advantages of $\mathbf{N^2}$ on eighteen real-world benchmarks, where $\mathbf{N^2}$ achieves superior performance.

2. Related Work

2.1. Graph Neural Networks

The concept of GNN was first introduced by Scarselli et al. (Scarselli et al., 2009), where a recurrent layer recursively updated node features. According to Banach's fixed point theorem (Banach, 1922), implementing the recurrent layer as a contraction mapping guarantees the existence of a unique fixed point representation for any input graph, towards which the recursive updates converge. However, their contraction mapping formulation is topology-dependent, which can lead to over-smoothing as the number of recursive steps increases. In contrast, the recurrent layer in $\mathbf{N^2}$ decouples message passing from input topology, and thus empowering global communication between graph nodes.

Over the past decade, convolutional neural networks (CNNs) (Krizhevsky et al., 2017; He et al., 2016) have achieved notable success. In pursuit of reproducing such accomplishments for graph data, attempts have focused on designing expressive convolution operators (Bruna et al., 2014; Niepert et al., 2016). Based on graph signal processing theory, some models approximate diverse convolution filters with parameterized polynomials, such as ChebNet (Defferrard et al., 2017), GDC (Klicpera et al., 2019) and GPRGNN (Chien et al., 2022). Despite being theoretically expressive (Wang & Zhang, 2022), these methods struggle to achieve high expressiveness in practice, as increasing the polynomial order incurs prohibitive computational complexity. Consequently, constrained-order GNNs (Kipf & Welling, 2017) have been proposed to perform local aggregation on graphs, coupling message passing with input topology. The coupled process contains inherent limitations, including over-smoothing (Li et al., 2018; Oono & Suzuki, 2021) and over-squashing (Alon & Yahav, 2021; Topping et al., 2021).

To overcome these limitations, some works try to decouple message passing from input topology and introduce alternate pathways, including edge shortcuts, pseudo nodes, and graph pooling operations. By adding edge shortcuts, methods aim to improve the message-passing efficiency on graphs. These methods can relieve certain bottlenecks on the input graphs (Topping et al., 2021; Deac et al., 2022; Qian et al., 2023) and aggregate multi-hop information during message passing (Abu-El-Haija et al., 2019; Gutteridge et al., 2023). Notably, we categorize graph structure learning methods (Zhu et al., 2021; Kazi et al., 2023; Wang et al., 2023; Zhao et al., 2023) into the edge shortcut paradigm, which constructs message-passing pathways by modeling edge connectivity. While edge shortcuts refine local connections, pseudo nodes directly enable global message passing. Prior works (Gilmer et al., 2017; Liu et al., 2022; Shirzad et al., 2023) connect pseudo nodes with all the graph nodes uniformly. Consequently, each node can access the full graph in a maximum of two steps. However, the uniform connection induces pseudo nodes to become over-squashing bottlenecks (Shirzad et al., 2023), limiting the global message passing. Unlike prior work, $\mathbf{N^2}$ models dynamic interactions between nodes and pseudo nodes, with edge weights varying flexibly across them.

Another line of effort in decoupling message passing from input topology is hierarchical GNNs. These methods (Ranjan et al., 2020; Yuan & Ji, 2020) learn multi-scale graph features through iterative graph pooling. For instance, Diff-

Pool (Ying et al., 2018) and GMT (Baek et al., 2022) learn soft assignment matrices to aggregate nodes into coarser levels. Meanwhile, Graph U-Nets (Gao & Ji, 2019), SAGPool (Lee et al., 2019) rank and select salient subsets to prune less critical nodes. Different from graph pooling methods, $\mathbf{N^2}$ performs both local and global message passing in each recursive step, avoiding information loss in coarser graphs (Wu et al., 2022a).

2.2. Graph Transformers

In comparison with CNNs, Transformers have also achieved superior performance across various domains (Vaswani et al., 2017). Transformers employ pairwise attentions as core building blocks, which can be seen as message passing on fully connected graphs. Therefore, transformers can be readily generalized to graph-structured data by integrating graph inductive bias, such as structural encoding (Zhang et al., 2020; Dwivedi & Bresson, 2021), attention bias (Ying et al., 2021), and compositional message passing layers (Rong et al., 2020; Wu et al., 2021). Researchers distinguish this type of model as graph transformers from traditional GNNs. However, dense attention requires quadratic space complexity and cubic time complexity, which is intractable for large-scale graphs. In order to scale graph transformers to larger graphs, recent methods propose to approximate dense attention through expander graphs (Shirzad et al., 2023), kernel functions (Wu et al., 2022b), and diffusion (Wu et al., 2023a). One similar work (Cai et al., 2023) to ours proves that message-passing layers with a single pseudo node can approximate self-attention. In this paper, we follow a contrary thread and develop $\mathbf{N^2}$ with a single shared message passing layer and multiple pseudo nodes. Each pseudo node interacts dynamically with graph nodes, avoiding forming the over-squashing bottlenecks as the uniform connected pattern (Gilmer et al., 2017; Liu et al., 2022; Shirzad et al., 2023).

3. Towards Dynamic Message Passing

This section describes how to reduce the message-passing complexity through learnable pseudo nodes and achieve flexible message passing via dynamic pathways. To introduce pseudo nodes in message passing, we unify graph nodes and pseudo nodes in a common space. To construct dynamic pathways between nodes, we propose a spatial proximity measurement in the common space.

3.1. Preliminaries

Notations. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ denotes node set of size $|\mathcal{V}| = n$ and $\mathcal{E} = \{e_{i,j} | v_j \in \mathcal{N}(v_i)\}$ denotes edge set of size $|\mathcal{E}| = m$. $\mathcal{N}(\cdot)$ denotes the one-hop neighbor set of the given node. Each node $v \in \mathcal{V}$ corresponds to a feature vector $\mathbf{x}_v \in \mathbb{R}^d$ where d is the number of features. Let $\mathbf{X} = (\mathbf{x}_{v_1}, \mathbf{x}_{v_2}, \cdots, \mathbf{x}_{v_n})^\top \in \mathbb{R}^{n \times d}$ be the graph node feature matrix. Except for node features, each graph comes with inherent topology, denoted as adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ or $\mathbf{E} \in \mathbb{R}^{n \times n \times d_e}$ with d_e edge features. For the adjacency matrix, $\mathbf{A}_{i,j} = 1$ if there exists an edge $e_{i,j} \in \mathcal{E}$. $\mathbf{E}_{i,j,\cdot} = \mathbf{e}_{i,j} \in \mathbb{R}^{d_e}$.

Parameterize Relations as Shared Function. In this work, we aim to model the relations between graph nodes and learnable pseudo nodes with a tractable number of parameters. Uniform connection pattern (Gilmer et al., 2017; Liu et al., 2022; Shirzad et al., 2023) directly learns the weights between pairs of graph and pseudo nodes, which correlates the number of parameters to the number of graph nodes. To address this problem, we gain inspiration from DyN (Pei & Wang, 2023), a distinctive neural network model that directly models neurons, instead of learning connection weights as conventional approaches (Krizhevsky et al., 2017; Vaswani et al., 2017). DyN inherits the notion of state space from prior works (Hochreiter & Schmidhuber, 1997; Gu et al., 2021), embedding neurons within a high-dimensional space $\mathcal{S} \subseteq \mathbb{R}^q$, and characterizes neuron states through their spatial coordinates. Connection weights among neurons are determined by the corresponding path integral in this space. Thereby, neuron embeddings become the learnable parameters, as opposed to traditional connection weights. The path integral function is shared among neurons, saving from heavy parameters. For more discussion on the relationship among DyN and (Hochreiter & Schmidhuber, 1997; Gu et al., 2021), please refer to Appendix A.

3.2. Common State Space

In light of DyN, we propose to unify graph nodes and pseudo nodes in the common state space $\mathcal{S}$, which enables the further construction of dynamic message-passing pathways. Here, we borrow the concept of *state* from prior works, such as DyN and LSTM (Hochreiter & Schmidhuber, 1997), to denote the learned descriptive embedding of graph nodes and pseudo nodes. Examples of the encoded information in the descriptive embeddings include node types and local topology.

Embedded Node States. Given a pseudo node set $\mathcal{U} = \{u_1, u_2, \cdots, u_{n_p}\}$ of size $|\mathcal{U}| = n_p$, pseudo nodes can be embedded in the state space as learnable parameter $\mathbf{R} = (\mathbf{r}_{u_1}, \cdots, \mathbf{r}_{u_{n_p}})^\top \in \mathbb{R}^{n_p \times q}$. These learnable elements are named "pseudo nodes" to align with graph nodes from inputs. Pseudo nodes can be associated with input graphs with pseudo edges and participants in the message passing between graph nodes. We will elaborate on how to associate pseudo nodes to input graphs after introducing the spatial proximity measurement in the state space. Given the node set $\mathcal{V}$ with feature matrix $\mathbf{X}$, the node

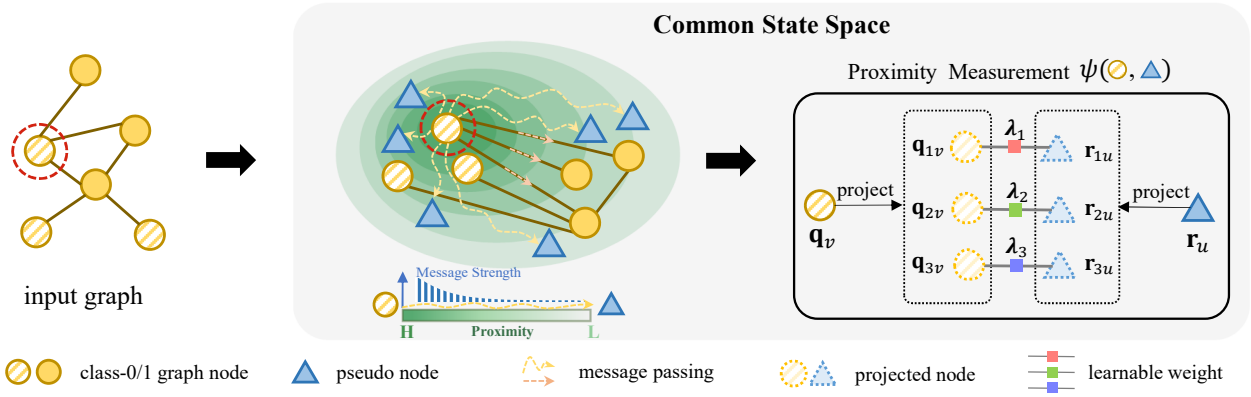


Figure 2. **Dynamic Message-Passing Pathway Construction in Common State Space.** Nodes and pseudo nodes interact actively in the common state space, constructing dynamic message-passing pathways through proximity measurement. In experimental model analysis, the pseudo nodes tend to be attracted toward a distinct node cluster.

states $\mathbf{Q} = (\mathbf{q}_1, \cdots, \mathbf{q}_n)^\top \in \mathbb{R}^{n\times q}$ can be formulated as $\mathbf{Q} = f(\mathbf{X})$, where $f(\cdot)$ is a permutation equivariant function, $\mathbf{q}_i \in \mathcal{S}$. Displacements of a node in the state space signify shifts in the learned descriptive embeddings, *i.e.*, the states. The objective of a GNN is to model the true distribution of input nodes in the state space, wherein nodes with similar features obtain proximal embeddings. The quantitative metric assessing proximity in the state space will be delineated in the subsequent context.

Proximity Measurement. To model complex relationships, we assume that the embedded nodes have a non-linear relationship in the state space $\mathcal{S}$. However, modeling non-linearity via operations such as path integral can be computationally complex. To address this problem, we approximate non-linear relationships with piece-wise weighted inner products. Taking proximity measurement between nodes as an example, each node is divided into k units. The proximity between two nodes v and v' in the state space can be formulated as:

$$\psi\left(\mathbf{q}_v, \mathbf{q}_{v'}\right) = \sum_{i=1}^{k} \lambda_i \mathbf{q}_{iv}^\top \mathbf{q}_{iv'}, \quad \mathbf{q}_{i\cdot} = g_i(\mathbf{q}_\cdot), \tag{1}$$

where λ_i is a learnable parameter, $g_i(\cdot)$ can be any non-linear function. $\psi(\cdot,\cdot)$ is termed as proximity instead of distance because it ranges in $\mathbb{R}$. The spatial proximity is weighted under the inner product similarity between k pairs of units to approximate complex relationships. We conduct ablation studies on the number of unit k in Appendix C.5. For proximity measurement between n nodes and n_p pseudo nodes in the state space, Eq. 1 entails a space complexity of merely $O(kn_pn)$. Sharing common state space gives rise to measurable proximity between pairs of pseudo nodes and graph nodes, thus motivating our interpretation of pseudo nodes as the message-passing pathway alternates.

3.3. Associating Pseudo Nodes to Input Graphs

Pseudo Edges. Based on the measurable spatial proximity between graph nodes and pseudo nodes, we now introduce how to obtain pseudo edges and thus associate pseudo nodes to input graphs. For each pseudo node $u \in \mathcal{U}$ embedded at $\mathbf{r}_u$ in the state space, it has pseudo edges with any pseudo node u' and node v, where pseudo edge weights $e_{u,u'}$ and $e_{u,v}$ can be formulated as proximity:

$$e_{u,u'} = \psi(\mathbf{r}_u, \mathbf{r}_{u'}), \quad e_{u,v} = \psi(\mathbf{r}_u, \mathbf{q}_v), \tag{2}$$

where $\psi(\cdot)$ is the spatial proximity defined in Eq. 1. With weighted pseudo edges $e_{u,\cdot}$, pseudo nodes can now interact with nodes.

Messages for Embedded Nodes. Following the common practice in GNNs (Veličković, 2022), we interpret the interaction between embedded nodes as message passing. To achieve this, both graph nodes v and pseudo nodes u learn their messages $\mathbf{m}_v, \mathbf{m}_u \in \mathbb{R}^d$ to be passed in the state space. As a result, graph nodes v and pseudo nodes u can be further described as:

$$\begin{aligned} u &\triangleq (\mathbf{r}_u, \mathbf{m}_u, e_{u,u'}, e_{u,v}),\ u' \in \mathcal{U}, v \in \mathcal{V}, \\ v &\triangleq (\mathbf{q}_v, \mathbf{m}_v, e_{v,v'}, e_{v,u}),\ v' \in \mathcal{V}, e_{v,v'} \in \mathcal{E}. \end{aligned} \tag{3}$$

The messages of graph nodes are initialized with node features. During the interaction between pseudo nodes and graph nodes, pseudo nodes emit their messages $\mathbf{m}_u$ and aggregate messages $\mathbf{m}_v$ from the graph nodes. Emitting messages initiates examinations of the input graphs, which enables pseudo nodes to probe the graph topology and extract target patterns. Based on the interaction, pseudo nodes optimize their positions in the state space, forming dynamic pseudo edges and serving flexible message passing on graphs. All these processes will be formulated in the next subsection.

3.4. Dynamic Message Passing in the State Space

Building upon the construction of common state space and pseudo nodes, graph nodes are now able to exchange messages in both local and global scope.

Global Message Passing. In the state space, nodes perform global message passing based on their embedded positions, where pseudo nodes serve as relays. Given node states $\mathbf{Q} \in \mathbb{R}^{n\times q}$, pseudo node states $\mathbf{R} \in \mathbb{R}^{n_p\times q}$ and node messages $\mathbf{M}^{\mathtt{n}} = (\mathbf{m}_{v_1}, \mathbf{m}_{v_2}, \cdots, \mathbf{m}_{v_n})^\top \in \mathbb{R}^{n\times d}$, the global message passing process can be formulated as:

$$\mathbf{G} = \mathbf{E}^{\mathtt{np}}\mathbf{M}^{\mathtt{n}}, \quad \mathbf{E}^{\mathtt{np}}_{ij} = e_{v_i,u_j} = \psi(\mathbf{Q}_{i,\cdot}, \mathbf{R}_{j,\cdot}), \tag{4}$$

$$\hat{\mathbf{G}} = \mathbf{E}^{\mathtt{pp}}\mathbf{G}, \quad \mathbf{E}^{\mathtt{pp}}_{ij} = e_{u_i,u_j} = \psi(\mathbf{R}_{i,\cdot}, \mathbf{R}_{j,\cdot}),$$

$$\Delta\mathbf{R} = h^{(0)}(\hat{\mathbf{G}}), \quad \mathbf{M}^{\mathtt{p}} = h^{(1)}(\hat{\mathbf{G}}), \tag{5}$$

$$\mathbf{M}^{\mathtt{glob}} = \mathbf{E}^{\mathtt{pn}}\mathbf{M}^{\mathtt{p}}, \ \mathbf{E}^{\mathtt{pn}}_{ij} = e_{u_i,v_j} = \psi([\mathbf{R}+\Delta\mathbf{R}]_{i,\cdot}, \mathbf{Q}_{j,\cdot}), \tag{6}$$

where $\mathbf{E}^{\mathtt{np}} \in \mathbb{R}^{n_p\times n}$ denotes the edge weight matrix from nodes to pseudo nodes, $\mathbf{E}^{\mathtt{pp}}$ and $\mathbf{E}^{\mathtt{pn}}$ follow the similar name rule. An example of $\mathbf{E}^{\mathtt{np}}_{ij}$ computation is illustrated in Fig. 2. Eq. 4 formulates the process that pseudo nodes aggregate messages from nodes. Eq. 5 formulates the global aggregation process among pseudo nodes, where $\Delta\mathbf{R} \in \mathbb{R}^{n_p\times q}$ denotes the learned displacements for pseudo nodes in the state space, $\mathbf{M}^{\mathtt{p}} \in \mathbb{R}^{n_p\times d}$ encodes the learned messages for pseudo nodes with the refined global information, $h^{(0)}(\cdot)$ and $h^{(1)}(\cdot)$ can be any non-linear function. Eq. 6 formulates the message diffusion process from pseudo nodes to graph nodes, thus finishing the global message passing.

For simplicity, we compile Eq. 4-6 as $\mathbf{M}^{\mathtt{glob}}, \Delta\mathbf{R} = \mathtt{GlobMP}(\mathbf{Q}, \mathbf{M}^{\mathtt{n}}, \mathbf{R})$. Note that the space complexity of our global message passing is $O(knn_p)$ with $k, n_p \ll n$, significantly lower than $O(n^2)$ in dense global message passing.

Local Message Passing. For local message passing, topology-coupled message passing is employed to encode the local structure. The local message-passing process can be formulated as:

$$\mathbf{m}_v^{\mathtt{local}} = \frac{1}{|\mathcal{N}(v)|+1}\left[\mathbf{m}_v + \sum_{v'\in\mathcal{N}(v)} h^{(2)}(\mathbf{m}_{v'}||\mathbf{e}_{v,v'})\right], \tag{7}$$

where $h^{(2)}(\cdot)$ can be any non-linear function, $||$ denotes the concatenate operation. Through local message passing, nodes aggregate messages from their adjacent nodes. Eq. 7 can be compiled as $\mathbf{M}^{\mathtt{local}} = \mathtt{LocalMP}(\mathbf{M}, \mathbf{E})$.

Similar to Eq. 5, nodes also learn their corresponding displacements through message passing, thereby interacting dynamically with pseudo nodes in the state space.

4. Implementation of $\mathbf{N^2}$

Based on the construction of common state space, we further develop a series of GNNs named $\mathbf{N^2}$ for graph and node classification tasks. $\mathbf{N^2}$ embeds graph nodes and pseudo nodes into the common state space, employing a recurrent layer to parameterize the displacements of embedded nodes. The recurrent layer includes pseudo-node adaptation and dynamic message passing. Pseudo-node adaptation employs `GlobMP` to generate query messages toward graph nodes. Dynamic message passing then extracts graph features through `LocalMP` and refines the extract features at the pseudo-node level with `GlobMP`. By feeding the states recursively into the recurrent layer, $\mathbf{N^2}$ can model the multi-step evolving dynamics of all the embedded nodes. We first outline the key process in the l-th recursive step and then describe different output fashion for specific downstream tasks.

4.1. Pseudo-node Adaptation

Pseudo nodes are initialized randomly in the state space. To adapt to specific input graphs, $\mathbf{N^2}$ first aggregates node messages to pseudo nodes, adjusting pseudo node states and corresponding messages accordingly. Given graph node messages $\mathbf{M}^{\mathtt{n}(l-1)}$ ($\mathbf{M}^{\mathtt{n}(0)} = \mathbf{X}$), node states $\mathbf{Q}^{(l-1)}$ and pseudo node states $\mathbf{R}^{(l-1)}$ at the l-th recursive step, the adaptation process can be formulated as:

$$\hat{\mathbf{M}}^{\mathtt{p}(l)}, \Delta\hat{\mathbf{R}}^{(l)} = \mathtt{GlobMP}\left(\mathbf{Q}^{(l-1)}, \mathbf{M}^{\mathtt{n}(l-1)}, \mathbf{R}^{(l-1)}\right), \tag{8}$$
$$\hat{\mathbf{R}}^{(l)} = \mathbf{R}^{(l-1)} + \Delta\hat{\mathbf{R}}^{(l)},$$

where $\hat{\mathbf{M}}^{\mathtt{p}(l)}$ denotes the messages of pseudo nodes after being received by nodes. These messages serve as probing signals, targeting different patterns on graphs. A linear transformation layer with $\mathtt{LeakyReLU}(\cdot)$ is adopted as the non-linear function for $h^{(0)}(\cdot)$, $h^{(1)}(\cdot)$ and $h^{(2)}(\cdot)$, denoted by $\mathtt{NL}(\cdot)$.

4.2. Dynamic Message Passing

$\mathbf{N^2}$ performs message passing at both the local level and global level. In local message passing, nodes exchange their own messages $\mathbf{M}^{\mathtt{n}(l-1)}$, received messages $\hat{\mathbf{M}}^{\mathtt{p}(l)}$ as well as node states $\mathbf{Q}^{(l-1)}$:

$$\hat{\mathbf{M}}^{\mathtt{n}(l)} = \mathtt{LocalMP}\left[\left(\mathbf{M}^{\mathtt{n}(l-1)}\|\hat{\mathbf{M}}^{\mathtt{p}(l)}\|\mathbf{Q}^{(l-1)}\right), \mathbf{E}\right], \tag{9}$$
$$\hat{\mathbf{Q}}^{(l)} = \mathbf{Q}^{(l-1)} + \mathtt{NL}(\hat{\mathbf{M}}^{\mathtt{n}(l)}).$$

Through local message passing, node messages are updated to $\hat{\mathbf{M}}^{\mathtt{n}(l)}$ in response to the probing messages $\hat{\mathbf{M}}^{\mathtt{p}(l)}$ and present node states $\mathbf{Q}^{(l-1)}$. $\mathbf{N^2}$ then sends the updated

Table 1. **Graph Classification Results on Small-scale Benchmarks (measured by accuracy: %).**

	PROTEIN	NCI1	IMDB-B	IMDB-M	COLLAB
#GRAPHS	1,113	4,110	1,000	1,500	5,000
#NODES	39.06	29.87	19.77	13.00	74.49
#EDGES	145.60	64.60	193.10	131.87	4914.4
#NODE FEATURES	3	37	0	0	0
PATCHY-SAN	75.00±2.51	78.60±1.90	71.00±2.29	45.23±2.84	72.60±2.15
GCN	73.24±0.73	76.29±1.79	73.26±0.46	50.39±0.41	80.59±0.27
PG	76.80±3.80	82.80±1.30	76.80±2.60	53.20±3.60	80.90±0.80
COCN	76.86±0.13	82.89±0.19	77.26±0.27	56.32±0.18	86.15±0.10
GIN	73.84±4.46	76.62±1.80	72.78±0.86	48.13±1.36	78.19±0.63
+PSEUDO NODE	74.11±4.12	77.08±1.49	-	-	-
GRAPHSAGE	73.48±5.66	73.82±2.17	68.80±4.50	47.60±3.50	73.90±1.70
+PSEUDO NODE	73.93±5.68	74.31±2.27	-	-	-
DIFFPOOL	75.62±5.17	76.62±1.93	73.14±0.70	51.31±0.72	82.13±0.43
+PSEUDO NODE	75.98±3.89	77.08±1.33	-	-	-
TOPKPOOL	70.48±1.01	67.02±2.25	71.58±0.95	48.59±0.72	77.58±0.85
SAGPOOL	71.56±1.49	67.45±1.11	72.55±1.28	50.23±0.44	78.03±0.31
STRUCTPOOL	75.16±0.86	78.64±1.53	72.06±0.64	50.23±0.53	77.27±0.51
SEP	76.42±0.39	79.35±0.33	74.12±0.56	51.53±0.65	81.28±0.15
GMT	75.09±0.59	76.35±2.62	73.48±0.76	50.66±0.82	80.74±0.54
$\mathbf{N^2}$ (OURS)	**77.53±1.78**	**83.52±3.75**	**79.95±2.46**	**57.31±2.19**	**86.72±1.62**

Table 2. **Graph Classification Results on Large-scale Benchmark OGB-molpcba (measured by average precision: %).**

#GRAPHS	#NODES	#EDGES
437,929	26	28.1
METHODS	AVERAGE PRECISION (%)	
GCN	20.20±0.24	
+PSEUDO NODE	24.24±0.34	
GIN	22.66±0.28	
+PSEUDO NODE	27.03±0.23	
MPNN+PSEUDO NODE	28.48±0.26	
GRAPHTRANS	27.61±0.29	
SAN	27.65±0.42	
GRAPHGPS	29.07±0.28	
GRAPHORMER	31.39±0.32	
$\mathbf{N^2}$ (OURS)	**33.90±0.73**	

messages to global message passing:

$$
\begin{aligned}
&\mathbf{M}^{\mathrm{p}(l)}, \Delta\mathbf{R}^{(l)} = \texttt{GlobMP}\left(\hat{\mathbf{Q}}^{(l)}, \hat{\mathbf{M}}^{\mathrm{n}(l)}, \hat{\mathbf{R}}^{(l)}\right), \\
&\mathbf{R}^{(l)} = \hat{\mathbf{R}}^{(l)} + \Delta\mathbf{R}^{(l)}, \\
&\mathbf{Q}^{(l)} = \hat{\mathbf{Q}}^{(l)} + \mathbf{M}^{\mathrm{p}(l)}, \\
&\mathbf{M}^{\mathrm{n}(l)} = \hat{\mathbf{M}}^{\mathrm{n}(l)} + \texttt{NL}(\mathbf{M}^{\mathrm{p}(l)}).
\end{aligned}
\tag{10}
$$

4.3. Output Module

$\mathbf{N^2}$ updates the states of nodes and pseudo nodes recursively through pseudo-node adaption and dynamic message passing. The associate parameters are shared across recursive steps. After L recursive steps, nodes and pseudo nodes now reach their final states $\mathbf{Q}^{(L)}$ and $\mathbf{R}^{(L)}$. $\mathbf{N^2}$ takes node states and pseudo node states as the learned representation for node-level and graph-level tasks respectively. For graph classification, $\mathbf{N^2}$ applies $\texttt{NL}(\cdot)$ to aggregate the pseudo nodes.

In the output module, $\mathbf{N^2}$ utilizes learnable parameter $\mathbf{C} \in \mathbb{R}^{n_c \times q}$ as the states of n_c class nodes in the state space. The proximity between class nodes and the final recursive output will be taken as the model prediction on input labels.

5. Experiment

In this section, we provide empirical evaluation results of $\mathbf{N^2}$ on real-world benchmarks. $\mathbf{N^2}$ is implemented with PyTorch (Paszke et al., 2019) and PyTorch-Geometric (Fey & Lenssen, 2019), and trained on a single Nvidia Geforce RTX 4090. The detailed experimental settings are presented in Appendix B.

5.1. Graph Classification

Experimental Setups. For graph classification, we adopt six benchmarks including three biochemical datasets OGB-molpcba (Hu et al., 2020), PROTEINS (Morris et al., 2020), NCI1 (Morris et al., 2020) and three social network datasets (Morris et al., 2020) COLLAB, IMDB-BINARY and IMDB-MULTI. The statistics of six benchmarks are summarized in Tab. 1 and Tab. 2. We choose convolutional GNNs, GNNs with a single pseudo node, hierarchical GNNs, and graph transformers as the baselines of graph classification. For detailed baseline setups, please refer to Appendix. B.1.2.

Performance. $\mathbf{N^2}$ and baselines are evaluated on both small-scale and large-scale benchmarks. The evaluation results in Tab. 1 and Tab. 2 showcase the ability of $\mathbf{N^2}$ to outperform various GNNs and graph transformers. Especially on the large-scale benchmark OGB-molpcba, $\mathbf{N^2}$ surpasses baseline models with only 500K parameters, while Graphormer reaches 31.39% average precision with 119.5M. Compared to GNNs with a single pseudo node, $\mathbf{N^2}$ gains significant improvements. This demonstrates the effectiveness of our common state space where pseudo nodes and nodes can interact dynamically with each other.

5.2. Node Classification

Experimental Setups. For node classification, we conduct experiments on (**1**) six middle-scale benchmarks: homophilic graphs (Shchur et al., 2019) (AmazonPhoto, AmazonComputers, CoauthorCS, and CoauthorPhysics), heterophilic graphs (Platonov et al., 2023) (Questions, Amazon-ratings, Tolokers, and Minesweeper); (**2**) four large-scale benchmarks: homophilic graphs (Hu et al., 2020) (OGB-arXiv, OGB-proteins), heterophilic graphs (Lim et al., 2021)

Table 3. **Node Classification Results on Small-scale Heterophilic Graphs (measured by ROC-AUC except accuracy for Amazon-ratings: %).** † denotes our reproduced results.

	QUESTIONS	AMAZON-RATINGS	TOLOKERS	MINESWEEPER
#NODES	48,921	24,492	11,758	10,000
#EDGES	153,540	93,050	519,000	39,402
SGC	75.91±0.96	50.66±0.48	80.70±0.97	70.88±0.90
GCN	76.09±1.27	48.70±0.63	83.64±0.67	89.75±0.52
GAT	77.43±1.20	49.09±0.63	83.70±0.47	92.01±0.68
GPRGNN	55.48±0.91	44.88±0.34	72.94±0.97	86.24±0.61
H2GCN	63.59±1.46	36.47±0.23	73.35±1.01	89.71±0.31
FAGCN	77.24±1.26	44.12±0.30	77.75±1.05	88.17±0.73
GLOGNN	65.74±1.19	36.89±0.14	73.39±1.17	51.08±1.23
GT	77.95±0.68	**51.17±0.66**	83.23±0.64	91.85±0.76
GRAPHORMER	OOM	OOM	OOM	OOM
GRAPHGPS	OOM	OOM	84.70±0.56	92.29±0.61
EXPHORMER †	73.86±0.58	49.36±0.36	84.20±0.22	90.42±0.10
N^2 (OURS)	**78.07±0.63**	50.25±0.53	**86.25±0.41**	**93.97±0.27**

Table 4. **Node Classification Results on Small-scale Homophilic Graphs (measured by accuracy: %).**

	COAUTHORCS	COAUTHORPHY	AMZPHOTO	AMZCOMPUTERS
#NODES	18,333	34,493	7,487	13,381
#EDGES	81,894	247,962	119,043	245,778
GCN	92.92±0.12	96.18±0.07	92.70±0.20	89.65±0.52
GAT	93.61±0.14	96.17±0.08	93.87±0.11	90.78±0.17
GPRGNN	95.13±0.09	96.85±0.08	94.49±0.14	89.32±0.29
APPNP	94.49±0.07	96.54±0.07	94.32±0.14	90.18±0.17
GT	94.64±0.13	97.05±0.05	94.74±0.13	91.18±0.17
GRAPHORMER	OOM	OOM	92.74±0.14	OOM
SAN	94.51±0.15	OOM	94.86±0.10	89.83±0.16
GRAPHGPS	93.93±0.12	OOM	95.06±0.13	OOM
NAGPHORMER	95.75±0.09	97.34±0.03	95.49±0.11	91.22±0.14
EXPHORMER	**95.77±0.15**	97.16±0.13	95.27±0.42	91.59±0.31
N^2 (OURS)	94.44±0.45	**97.56±0.28**	**95.75±0.34**	**92.51±0.13**

(arXiv-year, genius). The statistics of node classification benchmarks are summarized in Tab. 3-5. We choose related convolutional GNNs and graph transformers as the baselines of node classification. For detailed baseline setups, please refer to Appendix. B.2.2.

Performance. The evaluation results of $\mathbf{N^2}$ and baselines are presented in Tab. 3-5. $\mathbf{N^2}$ reaches superior or comparable performance against strong GNN baselines on both small-scale and large-scale benchmarks. Compared to graph transformers based on dense attention, including Graphormer, SAN, and GraphGPS, $\mathbf{N^2}$ surpasses the baselines on small-scale benchmarks and successfully scales to large-scale benchmarks. Note that Exphormer based on sparse attention also suffers from the out-of-memory problem in our experimental environment. For other sparse graph transformers, $\mathbf{N^2}$ can achieve comparable results and keep performance consistency between heterophilic and homophilic benchmarks.

5.3. Model Analysis

5.3.1. EFFECTIVENESS OF $\mathbf{N^2}$

Tackling Over-squashing. $\mathbf{N^2}$ employs pseudo nodes to enable graph nodes to communicate globally with linear complexity. To evaluate the effectiveness of $\mathbf{N^2}$ in tackling over-squashing, we conduct experiments on Trees

Table 5. **Node Classification Results on Large-scale Benchmarks (measured by ROC-AUC for Genius and OGB-proteins and accuracy for arXiv-year and OGB-arXiv: %).** † denotes our reproduced results.

	GENIUS	ARXIV-YEAR	OGB-ARXIV	OGB-PROTEINS
#NODES	421,961	169,343	169,343	132,534
#EDGES	984,979	1,166,243	1,166,243	39,561,252
SGC	82.36±0.37	32.83±0.13	67.79±0.27	70.31±0.23
GCN	87.42±0.37	46.02±0.26	71.74±0.29	72.51±0.35
GAT	55.80±0.87	46.05±0.51	67.63±0.23	74.63±1.24
GPRGNN	90.05±0.31	45.07±0.21	71.78±0.18	\
APPNP	85.36±0.62	38.15±0.26	\	\
MIXHOP	90.58±0.16	51.81±0.17	\	\
H2GCN	OOM	49.09±0.10	OOM	OOM
LINKX	**90.77±0.27**	56.00±1.34	\	\
GRAPHORMER				
SAN		OOM		
GRAPHGPS				
EXPHORMER	OOM	OOM	72.44±0.28	OOM
NODEFORMER	88.62±0.27 (†)	37.68±0.30 (†)	59.90±0.42	77.45±1.15
SGFORMER	83.91±2.60 (†)	44.34±0.07 (†)	**72.63±0.13**	79.53±0.38
N^2 (OURS)	89.32±0.26	**58.69±0.42**	70.01±0.65	**79.55±0.79**

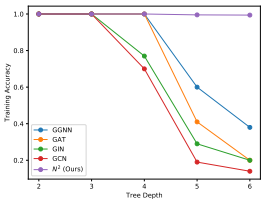


Figure 3. **Effectiveness Study on Over-squashing.**

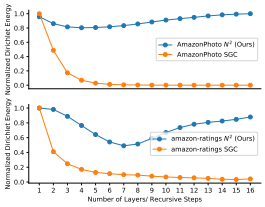
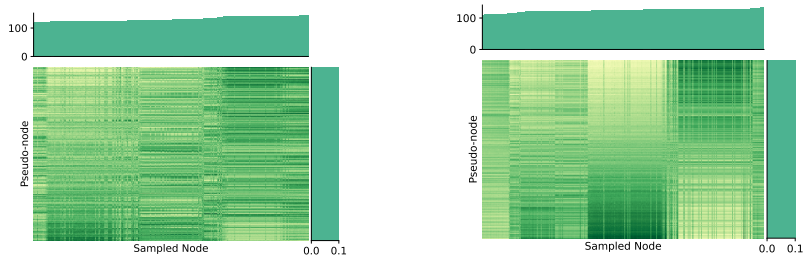


Figure 4. **Effectiveness Study on Over-smoothing.**

Match (Alon & Yahav, 2021). The training accuracies are presented in Fig. 3. We can see that $\mathbf{N^2}$ maintains the fitting ability across different tree depths, demonstrating its effectiveness to counter over-squashing.

Tackling Over-smoothing. The other issue encountered when message passing relies heavily on input topology is over-smoothing (Oono & Suzuki, 2021). In $\mathbf{N^2}$, pseudo nodes are adopted as message-passing pathway alternates, decoupling the message-passing process from input topology. To evaluate whether $\mathbf{N^2}$ can alleviate the over-smoothing issue, we explore the changes in the Dirichlet energy (Cai & Wang, 2020) as the number of recursive steps increases. Fig. 4 presents the comparison results between $\mathbf{N^2}$ and SGC (Wu et al., 2019) on AmazonPhoto and amazon-ratings. The reported values are normalized to [0, 1] by dividing the maximum energy values respectively. The results show that $\mathbf{N^2}$ maintains the Dirichlet energy as the recursive steps increase, alleviating over-smoothing.

Expressiveness over vanilla MPNN. We also compare our proposed dynamic message-passing mechanism with the vanilla MPNN, subgraph GNN, and local GNN on the subgraph counting task (Zhang et al., 2023). The results of baselines are from (Zhang et al., 2023). From the results in Tab. 6, we can see that $\mathbf{N^2}$ surpasses the expressiveness



(a) $\mathbf{E}^{\mathrm{np}}$ (b) $\mathbf{E}^{\mathrm{pn}\top}$

Figure 5. **Message Passing Analysis.** The proximities between sampled nodes and pseudo nodes are depicted in the center of each sub-figure, where darker green indicates higher proximity and brighter green indicates lower proximity. The distribution on the top/right of each sub-figure denotes the sum of proximity for each node/pseudo node. The sampled nodes are ranked based on the sum of proximity.

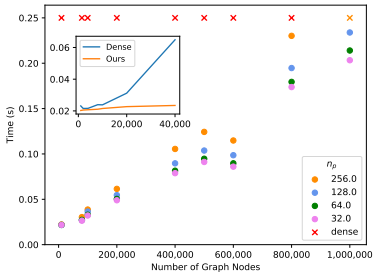


Figure 6. **Complexity Analysis.** The results under different numbers of graph nodes are compared. Marker `X` denotes out-of-memory. `Dense` denotes dense pairwise relation modeling.

Table 6. **Expressiveness Analysis on Subgraph Counting (measured by mean absolute error).**

	CYCLE5,V↓	CYCLE6,V↓	CHORDAL4,V↓	CHORDAL5,V↓
MPNN	0.30	0.21	0.32	0.24
SUBGRAPH GNN	0.09	0.08	0.05	0.07
LOCAL 2-GNN	0.01	0.02	0.00	0.02
LOCAL 2-FGNN	0.01	0.02	0.00	0.01
$\mathbf{N^2}$ (OURS)	0.03	0.02	0.07	0.05

of the vanilla MPNN and achieves competitive results with more expressive GNNs, indicating the potential of the proposed dynamic message-passing mechanism.

(a) 0 (b) 20 (c) 500

Figure 7. **Distribution of Embedded Nodes.** The t-sne (Van der Maaten & Hinton, 2008) results under different training epochs are compared. Input nodes with different labels are depicted as: **0**, **1**, **2**, **3**, **4**, **5**, 6. Pseudo nodes are depicted as △. Class nodes are depicted as ⋆.

5.3.2. DISTRIBUTION OF EMBEDDED NODES

$\mathbf{N^2}$ enables nodes and pseudo nodes to interact directly with each other in the common state space. In order to analyze the behavior of embedded nodes, we visualize the distribution of all the embedded nodes through training. The distributions of embedded nodes on AmazonPhoto at the start of recursive steps are depicted in Fig. 7. For more visualization through multiple recursive steps, please refer to Appendix C.2. From Fig. 7, we can see that as the graph nodes are clustered with different class nodes, pseudo nodes are also split into several groups. Each pseudo-node group is attracted toward a distinct node cluster. This indicates that the randomly initialized pseudo nodes will attend a particular node group, serving as the global message-passing pathways to other groups. As a result, each pseudo node assumes a balanced fraction of the global message load, mitigating the risks of inducing over-squashing bottlenecks.

To further analyze the message passing between nodes and pseudo nodes, we visualize their corresponding proximity on AmazonPhoto. The proximity matrix $\mathbf{E}^{\mathrm{pn}}$ and $\mathbf{E}^{\mathrm{np}}$ during pseudo node adaptation in Eq. 8 is depicted in Fig. 5. For more proximity visualization results, please refer to Appendix C.3. In Fig. 5, 1000 nodes are sampled randomly and ranked based on their intro-/outre-proximity summation. The intro-/outre-proximity summation indicates the message load a node takes or emits during the message passing. As depicted in Fig. 5, both nodes and pseudo nodes assume a balanced message load. Each pseudo node has various proximity toward individual graph nodes, constructing dynamic connections instead of uniform ones. All these properties empower efficient global message passing on graphs.

5.3.3. COMPLEXTY ANALYSIS

Our proposed dynamic message-passing mechanism enables graph nodes to access each other without dense pairwise modeling. As described in Sec. 3.4, the space complexity of our global message passing is $O(knn_p)$. We further conduct an empirical analysis of $\mathbf{N^2}$ on computational complexity. As illustrated in Fig. 6, the computational time exhibits linear scalability for the number of nodes n, while being insensitive to the number of pseudo nodes n_p. In comparison to dense pairwise modeling, we substitute the global message passing in $\mathbf{N^2}$ with a dense attention scheme. The resulting growth rate of computational time for $\mathbf{N^2}$ is lower than the dense method. Moreover, the dense method fails to extend to large-scale graphs, whereas $\mathbf{N^2}$ solely encounters out-of-memory exceptions given the extreme situation of $n_p = 256$ and $n = 1\mathtt{M}$.

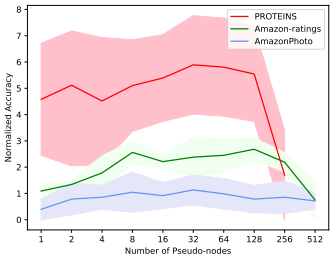


Figure 8. **Ablation Studies on the Number of Pseudo nodes (n_p).**

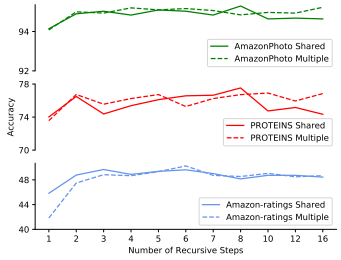


Figure 9. **Comparison between Single Shared Recurrent Layer and Multiple Recurrent Layers.**

Table 7. **Ablation Studies on the Modules of $\mathbf{N^2}$ (measured by accuracy: %).** PA. denotes pseudo-node adaptation, L. denotes local message passing, and G. denotes global message passing.

		Amz -ratings	Amz Photo	RPO -TEINS
w/o.	PA.	48.20	95.10	75.76
	L.	48.69	94.84	75.77
	G.	50.16	94.58	75.54
Full		**50.25**	**95.75**	**77.53**

5.3.4. Pseudo Node Ablation

We conduct ablation studies on the number of pseudo nodes n_p. Fig. 8 presents the classification accuracy under different numbers of pseudo nodes on PROTEINS, AmazonPhoto, and amazon-ratings. To present the results in a comparable form within the same figure, the accuracy values have been normalized by subtracting the minimum value. As n_p increases, $\mathbf{N^2}$ gets improvement in accuracy for all three datasets. However, the classification accuracy exhibits degradation on PROTEINS when n_p reaches 256. This is because n_p exceeds the optimization capacity of $\mathbf{N^2}$ on PROTEINS. A similar degradation also happens on amazon-ratings. In practice, the optimal value of n_p is around 16 to 32 for graph classification while reaching 128 to 300 for node classification. We also conduct ablation studies in Appendix C.7 to evaluate the effectiveness of pseudo nodes, where $\mathbf{N^2}$ with pseudo nodes outperforms dense message passing.

5.3.5. Recurrent Layer Ablation

By employing the same recurrent layer through recursive steps, $\mathbf{N^2}$ attains comparable performance to baseline models while requiring substantially fewer parameters. To analyze the efficacy of parameter-sharing, we conduct ablation studies on the number of recurrent layers, contrasting $\mathbf{N^2}$ employing shared parameters against $\mathbf{N^2}$ with multiple recurrent layers. For simplicity, we denote the number of recurrent layers as L_p, where $\mathbf{N^2}$ with multiple recurrent layers has L_p equals to the number of recursive steps L and $\mathbf{N^2}$ with shared parameters has $L_p = 1$.

As presented in Fig. 9, $\mathbf{N^2}$ with $L_p = 1$ and multiple recursive steps generally achieves matching performance with $L_p = L$. This indicates that shared parameters are sufficient in modeling convergent dynamics of embedded nodes in the state space (Fig. 7). However, on Amazon-ratings and AmazonPhoto, when L_p surpasses 8, $\mathbf{N^2}$ achieves better performance against shared parameters. In further empirical analysis, we find that embedded nodes in $\mathbf{N^2}$ with $L_p = 1$ tend to maintain current dynamics through recursive steps and thus become less effective in precise position optimization. Please refer to Appendix C.4 for detailed analysis. A step-dependent parameter may further improve the performance of $\mathbf{N^2}$ with $L_p = 1$ on a larger number of recursive steps L. We will keep exploring it in future work.

5.3.6. Module Ablation

Ablation studies are conducted on the modules of $\mathbf{N^2}$, including pseudo-node adaptation, local message passing, and global message passing. As shown in Tab. 7, $\mathbf{N^2}$ with three modules achieves superior performance across all the benchmarks. In comparison among the ablated $\mathbf{N^2}$, removing global message passing leads to a larger degradation in graph classification accuracy. Conversely, node classification exhibits greater sensitivity to the removal of local message passing. Moreover, pseudo-node adaptation is also required for $\mathbf{N^2}$ for all three benchmarks. This indicates that adapting the randomly initialized distribution of pseudo nodes enables better interactions with graph nodes.

6. Conclusion

In this paper, we presented a dynamic message passing method on graphs. Both graph nodes and pseudo nodes are embedded in a common state space, interacting and adjusting their position actively. The common state space enables measurable relations between all the embedded nodes. Therefore, we interpreted groups of pseudo nodes as pseudo nodes, constructing dynamic pathway alternates for message passing. Based on the proposed dynamic message passing, we further developed a series of GNNs named $\mathbf{N^2}$ for graph and node classification tasks. Experimental results show that $\mathbf{N^2}$ achieves superior performance over competitive baseline models. For limitations discussion, please refer to Appendix E.

Impact Statements

This paper proposed a novel method for constructing dynamic message-passing pathways by bridging graph nodes and pseudo nodes in a common state space. Our goal is to advance the field of graph representation learning. The proposed method remains independent of specific downstream applications. As graph data are ubiquitous in the real world, there are many potential applications of our work, including computational biology (Zaidi et al., 2023), intelligent transportation (Rahmani et al., 2023), and algorithmic reasoning (Diao & Loynd, 2023). Regarding ethical considerations, we do not presently foresee evident issues or potential for adverse societal impacts.

References

Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Steeg, G. V., and Galstyan, A. MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing. In *International Conference on Machine Learning*, pp. 21–29, Long Beach, USA, 2019. PMLR.

Alon, U. and Yahav, E. On the Bottleneck of Graph Neural Networks and its Practical Implications. In *International Conference on Learning Representations*, Virtual Only, 2021.

Baek, J., Kang, M., and Hwang, S. J. Accurate Learning of Graph Representations with Graph Multiset Pooling. In *International Conference on Learning Representations*, virtual, 2022.

Banach, S. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae*, 3(1):133–181, 1922.

Bo, D., Wang, X., Shi, C., and Shen, H. Beyond Low-frequency Information in Graph Convolutional Networks. In *AAAI Conference on Artificial Intelligence*, volume 35, pp. 3950–3957, virtual, 2021.

Bresson, X. and Laurent, T. Residual Gated Graph ConvNets, April 2018.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral Networks and Locally Connected Networks on Graphs. *arXiv:1312.6203 [cs]*, May 2014.

Cai, C. and Wang, Y. A Note on Over-Smoothing for Graph Neural Networks. *arXiv:2006.13318 [cs, stat]*, June 2020.

Cai, C., Hy, T. S., Yu, R., and Wang, Y. On the Connection Between MPNN and Graph Transformer. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 3408–3430. PMLR, July 2023.

Chen, J., Gao, K., Li, G., and He, K. NAGphormer: A Tokenized Graph Transformer for Node Classification in Large Graphs. In *International Conference for Learning Representations*, 2023.

Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. Simple and Deep Graph Convolutional Networks. In *International Conference on Machine Learning*, pp. 1725–1735, virtual, 2020. PMLR.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive Universal Generalized PageRank Graph Neural Network. In *International Conference on Learning Representations*, virtual, 2022.

Deac, A., Lackenby, M., and Veličković, P. Expander Graph Propagation. In *Proceedings of the First Learning on Graphs Conference*, pp. 38:1–38:18. PMLR, 2022.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *arXiv:1606.09375 [cs, stat]*, February 2017.

Di Giovanni, F., Giusti, L., Barbero, F., Luise, G., Lio', P., and Bronstein, M. On Over-Squashing in Message Passing Neural Networks: The Impact of Width, Depth, and Topology. In *International Conference on Machine Learning*, 2023a.

Di Giovanni, F., Rusch, T. K., Bronstein, M., Deac, A., Lackenby, M., Mishra, S., and Veličković, P. How does over-squashing affect the power of GNNs? *Transactions on Machine Learning Research*, 2023b.

Diao, C. and Loynd, R. Relational Attention: Generalizing Transformers for Graph-Structured Tasks. In *The Eleventh International Conference on Learning Representations*, February 2023.

Dwivedi, V. P. and Bresson, X. A Generalization of Transformer Networks to Graphs. In *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, January 2021.

Dwivedi, V. P., Rampášek, L., Galkin, M., Parviz, A., Wolf, G., Luu, A. T., and Beaini, D. Long Range Graph Benchmark. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22326–22340, December 2022.

Fey, M. and Lenssen, J. E. Fast Graph Representation Learning with PyTorch Geometric. In *International Conference on Learning Representations Workshop on Graphs and Manifolds*, 2019.

Franceschi, L., Niepert, M., Pontil, M., and He, X. Learning Discrete Structures for Graph Neural Networks. In *International Conference on Machine Learning*, pp. 1972–1982. PMLR, 2019.

Gao, H. and Ji, S. Graph U-Nets. In *International Conference on Machine Learning*, pp. 2083–2092, Long Beach, California, USA, 2019. PMLR.

Gasteiger, J., Bojchevski, A., and Günnemann, S. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2018.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural Message Passing for Quantum Chemistry. In *International Conference on Machine Learning*, volume 70, pp. 1263–1272, Sydney, Australia, 2017. PMLR.

Gu, A., Goel, K., and Re, C. Efficiently Modeling Long Sequences with Structured State Spaces. In *International Conference on Learning Representations*, October 2021.

Gutteridge, B., Dong, X., Bronstein, M. M., and Giovanni, F. D. DRew: Dynamically Rewired Message Passing with Delay. In *International Conference on Machine Learning*, pp. 12252–12267. PMLR, July 2023.

Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs. In *International Conference on Neural Information Processing Systems*, pp. 1025–1035, Red Hook, USA, 2017. Curran Associates Inc.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, 2016.

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. ISSN 0899-7667.

Hu*, W., Liu*, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., and Leskovec, J. Strategies for Pre-training Graph Neural Networks. In *International Conference on Learning Representations*, September 2019.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open Graph Benchmark: Datasets for Machine Learning on Graphs. In *Advances in Neural Information Processing Systems*, volume 33, pp. 22118–22133. Curran Associates, Inc., 2020.

Huang, Z., Wang, Y., Li, C., and He, H. Going Deeper into Permutation-Sensitive Graph Neural Networks. In *International Conference on Machine Learning*, pp. 9377–9409, Baltimore, Maryland, USA, 2022. PMLR.

Hwang, E., Thost, V., Dasgupta, S. S., and Ma, T. Revisiting Virtual Nodes in Graph Neural Networks for Link Prediction. 2021.

Kazi, A., Cosmo, L., Ahmadi, S.-A., Navab, N., and Bronstein, M. M. Differentiable Graph Module (DGM) for Graph Convolutional Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2): 1606–1617, February 2023. ISSN 1939-3539. doi: 10.1109/TPAMI.2022.3170249.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. In *International Conference for Learning Representations*, San Diego, USA, 2015. Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Kipf, T. N. and Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*, Toulon, France, 2017.

Klicpera, J., Weiß enberger, S., and Günnemann, S. Diffusion Improves Graph Learning. In *International Conference on Advances in Neural Information Processing Systems*, volume 32, pp. 13333–13345, Vancouver, Canada, 2019. Curran Associates, Inc.

Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., and Tossou, P. Rethinking Graph Transformers with Spectral Attention. In *Advances in Neural Information Processing Systems*, volume 34, pp. 21618–21629. Curran Associates, Inc., 2021.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Lee, J., Lee, I., and Kang, J. Self-Attention Graph Pooling. In *International Conference on Machine Learning*, pp. 3734–3743, Long Beach, California, USA, 2019. PMLR.

Leskovec, J. and Krevl, A. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, 2014.

Li, Q., Han, Z., and Wu, X.-M. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. In *AAAI Conference on Artificial Intelligence*, New Orleans, USA, 2018.

Li, X., Zhu, R., Cheng, Y., Shan, C., Luo, S., Li, D., and Qian, W. Finding Global Homophily in Graph Neural Networks When Meeting Heterophily. In *International Conference on Machine Learning*, volume 162, pp. 13242–13256, Baltimore, Maryland, USA, 2022. PMLR.

Lim, D., Hohne, F., Li, X., Huang, S. L., Gupta, V., Bhalerao, O., and Lim, S. N. Large Scale Learning on Non-Homophilous Graphs: New Benchmarks and Strong

Simple Methods. In *Advances in Neural Information Processing Systems*, volume 34, pp. 20887–20902. Curran Associates, Inc., 2021.

Liu, X., Cheng, J., Song, Y., and Jiang, X. Boosting Graph Structure Learning with Dummy Nodes. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 13704–13716. PMLR, 2022.

Mialon, G., Chen, D., Selosse, M., and Mairal, J. GraphiT: Encoding Graph Structure in Transformers. *arXiv preprint*, June .

Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. TUDataset: A collection of benchmark datasets for learning with graphs. In *International Conference on Machine Learning Workshop on Graph Representation Learning and Beyond*, Virtual Only, 2020. arXiv.

Niepert, M., Ahmed, M., and Kutzkov, K. Learning convolutional neural networks for graphs. In *International Conference on Machine Learning*, volume 48, pp. 2014–2023, New York, NY, USA, 2016. PMLR.

Oono, K. and Suzuki, T. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. In *International Conference for Learning Representations*, January 2021.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An imperative style, high-performance deep learning library. In *International Conference on Neural Information Processing Systems*, pp. 8026–8037, Red Hook, NY, USA, 2019. Curran Associates Inc.

Pei, Z. and Wang, S. Dynamics-inspired Neuromorphic Visual Representation Learning. In *International Conference on Machine Learning*, pp. 27521–27541. PMLR, July 2023.

Platonov, O., Kuznedelev, D., Diskin, M., Babenko, A., and Prokhorenkova, L. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, Kigali, Rwanda, February 2023.

Qian, C., Manolache, A., Ahmed, K., Zeng, Z., den Broeck, G. V., Niepert, M., and Morris, C. Probabilistically Rewired Message-Passing Neural Networks. In *The Twelfth International Conference on Learning Representations*, 2023.

Rahmani, S., Baghbani, A., Bouguila, N., and Patterson, Z. Graph Neural Networks for Intelligent Transportation Systems: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, 24(8):8846–8885, 2023. ISSN 1558-0016.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a General, Powerful, Scalable Graph Transformer. In *Advances in Neural Information Processing Systems*, volume 35, pp. 14501–14515, New Orleans, USA, December 2022.

Ranjan, E., Sanyal, S., and Talukdar, P. ASAP: Adaptive Structure Aware Pooling for Learning Hierarchical Graph Representations. In *AAAI Conference on Artificial Intelligence*, volume 34, pp. 5470–5477, New York, NY, USA, 2020.

Rong, Y., Bian, Y., Xu, T., Xie, W., WEI, Y., Huang, W., and Huang, J. Self-Supervised Graph Transformer on Large-Scale Molecular Data. In *Advances in Neural Information Processing Systems*, volume 33, pp. 12559–12571. Curran Associates, Inc., 2020.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

Shchur, O., Mumme, M., Bojchevski, A., and Günnemann, S. Pitfalls of Graph Neural Network Evaluation. *arXiv:1811.05868 [cs, stat]*, June 2019.

Shi, Y., Huang, Z., Feng, S., Zhong, H., Wang, W., and Sun, Y. Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification. In *International Joint Conference on Artificial Intelligence*, volume 2, pp. 1548–1554, Virtual Event / Montreal, Canada, 2021. ijcai.org.

Shirzad, H., Velingker, A., Venkatachalam, B., Sutherland, D. J., and Sinop, A. K. Exphormer: Sparse Transformers for Graphs. In *International Conference on Machine Learning*, volume 202, Honolulu, USA, June 2023. PMLR.

Sun, J., Wang, S., Han, X., Xue, Z., and Huang, Q. All in a row: Compressed convolution networks for graphs. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 33061–33076, Honolulu, USA, July 2023. PMLR.

Sun, Y., Deng, H., Yang, Y., Wang, C., Xu, J., Huang, R., Cao, L., Wang, Y., and Chen, L. Beyond Homophily: Structure-aware Path Aggregation Graph Neural Network. In *International Joint Conference on Arti-*

ficial Intelligence, volume 3, pp. 2233–2240, 2022. doi: 10.24963/ijcai.2022/310.

Topping, J., Di Giovanni, F., Chamberlain, B. P., Dong, X., and Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference for Learning Representations*, November 2021.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6000–6010, Red Hook, NY, USA, December 2017. Curran Associates Inc.

Veličković, P. Message passing all the way up. February 2022. doi: 10.48550/arXiv.2202.11097.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph Attention Networks. In *International Conference on Learning Representations*, Vancouver, Canada, February 2018.

Wang, H., Fu, Y., Yu, T., Hu, L., Jiang, W., and Pu, S. PROSE: Graph Structure Learning via Progressive Strategy. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, pp. 2337–2348, New York, NY, USA, 2023. Association for Computing Machinery.

Wang, X. and Zhang, M. How Powerful are Spectral Graph Neural Networks. In *International Conference on Machine Learning*, pp. 23341–23362, Baltimore, Maryland, USA, 2022. PMLR.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying Graph Convolutional Networks. In *International Conference on Machine Learning*, pp. 6861–6871, Long Beach, USA, 2019. PMLR.

Wu, J., Chen, X., Li, S., and Xu, K. Structural entropy guided graph hierarchical pooling. In *International Conference on Machine Learning*, volume 162, pp. 24017–24030, Baltimore, Maryland, USA, 2022a. PMLR.

Wu, Q., Zhao, W., Li, Z., Wipf, D., and Yan, J. NodeFormer: A Scalable Graph Structure Learning Transformer for Node Classification. In *Advances in Neural Information Processing Systems*, October 2022b.

Wu, Q., Yang, C., Zhao, W., He, Y., Wipf, D., and Yan, J. DIFFormer: Scalable (Graph) Transformers Induced by Energy Constrained Diffusion. In *The Eleventh International Conference on Learning Representations*, February 2023a.

Wu, Q., Zhao, W., Yang, C., Zhang, H., Nie, F., Jiang, H., Bian, Y., and Yan, J. SGFormer: Simplifying and Empowering Transformers for Large-Graph Representations. In *Thirty-Seventh Conference on Neural Information Processing Systems*, November 2023b.

Wu, Z., Jain, P., Wright, M., Mirhoseini, A., Gonzalez, J. E., and Stoica, I. Representing Long-Range Context for Graph Neural Networks with Global Attention. In *Advances in Neural Information Processing Systems*, volume 34, pp. 13266–13279. Curran Associates, Inc., 2021.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How Powerful are Graph Neural Networks? In *International Conference on Learning Representations*, New Orleans, LA, USA, 2019.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. Do Transformers Really Perform Bad for Graph Representation? In *Advances in Neural Information Processing Systems*, volume 34, pp. 28877–28888. Curran Associates, Inc., November 2021.

Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., and Leskovec, J. Hierarchical Graph Representation Learning with Differentiable Pooling. In *Advances in Neural Information Processing Systems*, volume 31, pp. 4805–4815, Montréal, Canada, 2018. Curran Associates, Inc.

Yuan, H. and Ji, S. StructPool: Structured Graph Pooling via Conditional Random Fields. In *International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

Zaidi, S., Schaarschmidt, M., Martens, J., Kim, H., Teh, Y. W., Sanchez-Gonzalez, A., Battaglia, P., Pascanu, R., and Godwin, J. Pre-training via Denoising for Molecular Property Prediction. In *The Eleventh International Conference on Learning Representations*, February 2023.

Zhang, B., Gai, J., Du, Y., Ye, Q., He, D., and Wang, L. Beyond Weisfeiler-Lehman: A Quantitative Framework for GNN Expressiveness. In *International Conference on Learning Representations*, October 2023.

Zhang, J., Zhang, H., Xia, C., and Sun, L. Graph-Bert: Only Attention is Needed for Learning Graph Representations. *arXiv:2001.05140 [cs, stat]*, January 2020.

Zhao, W., Wu, Q., Yang, C., and Yan, J. GraphGLOW: Universal and Generalizable Structure Learning for Graph Neural Networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, pp. 3525–3536, New York, NY, USA, 2023. Association for Computing Machinery.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, January 2020. ISSN 2666-6510. doi: 10/gjt96x.

Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., and Koutra, D. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pp. 7793–7804, Red Hook, NY, USA, 2020. Curran Associates Inc.

Zhu, Y., Xu, W., Zhang, J., Liu, Q., Wu, S., and Wang, L. Deep Graph Structure Learning for Robust Representations: A Survey. *arXiv:2103.03036 [cs]*, March 2021.

A. State Space

N^2 unifies graph nodes and pseudo nodes in a common state space. The concept of state space has been widely adopted in various machine learning methods. In reinforcement learning, the state space is typically modeled as a discrete space, encompassing all possible states an agent can be in. By taking discrete actions, agents transition between states. Through interactions with the environment, agents learn optimal policies that determine their actions at certain states, maximizing the cumulative reward over time.

In contrast to the discrete paradigm, recurrent models such as recurrent neural networks (RNNs), long short-term memory (LSTM), and state space models (SSMs) utilize continuous state representations to model sequential data. In this setting, input sequences are tokenized and consumed by the model in a successive manner. As a result, recurrent models learn to update the state embeddings recursively.

Building upon the continuous paradigm, DyN (Pei & Wang, 2023) further applies measurements to the state space, giving rise to measurable spatial relations between state embeddings. All the input tokens can now be modeled simultaneously in the state space.

However, modeling dense pairwise relations among a large number of nodes becomes intractable in the graph representation learning setting, due to the quadratic complexity. To reduce the complexity, we introduce pseudo nodes to serve as proxies for pairwise relations between nodes. Consequently, the relations between graph nodes are decomposed into two components: relations between source nodes and pseudo nodes, and relations between pseudo nodes and target nodes. Since the number of pseudo nodes is substantially smaller than the number of graph nodes, this decomposition effectively reduces the overall complexity. A similar decomposition strategy can be employed for the message-passing scheme.

B. Details on Experiments

B.1. Graph Classification

B.1.1. Benchmark Descriptions

We adopt six benchmarks for graph classification, including three biochemical datasets OGB-molpcba (Hu et al., 2020), PROTEINS (Morris et al., 2020), NCI1 (Morris et al., 2020) and three social network datasets (Morris et al., 2020) COLLAB, IMDB-BINARY and IMDB-MULTI.

OGB-molpcba is a large-scale molecular property prediction benchmark. Each graph is a discrete molecule, wherein nodes denote individual atoms and edges encode chemical bonds between atoms.

PROTEINS comprises 1,113 protein graphs with amino acids constituting the nodes. The associated binary classification task involves predicting protein category labels, specifically discriminating between enzymes versus non-enzymes.

NCI1 constitutes 4,110 graphs of chemical compounds assembled by the National Cancer Institute (NCI). Graph labels categorize compounds as exhibiting either positive or negative efficacy against cell lung cancer.

IMDB-BINARY and **IMDB-MULTI** constitute collaboration network graphs of movie actors and actresses, with graph labels indicating movie genres.

COLLAB constitutes scientific collaboration networks wherein each graph is an ego network for a researcher, encompassing their co-authors. Graph labels indicate the specific scientific interest corresponding to each researcher.

B.1.2. Experimental Setups

We choose (**1**) Convolutional GNNs: GCN (Kipf & Welling, 2017), PATCHY-SAN (Niepert et al., 2016), GraphSAGE (Hamilton et al., 2017), GIN (Xu et al., 2019), and CoCN (Sun et al., 2023); (**2**) GNNs with a single pseudo node from (Cai et al., 2023); (**3**) Hierarchical GNNs: DiffPool (Ying et al., 2018), Graph U-Net (Gao & Ji, 2019), SAGPool (Lee et al., 2019), StructPool (Yuan & Ji, 2020), SEP (Wu et al., 2022a), and GMT (Baek et al., 2022); (**4**) Graph transformers: Graphormer (Ying et al., 2021), SAN (Kreuzer et al., 2021), GraphGPS (Rampášek et al., 2022), and GraphTrans (Cai et al., 2023) as the baselines of graph classification.

Except for OGB-molpcba, we perform 10-fold cross-validation with LIB-SVM following Xu et al. and report average performance. Average precision is reported for OGB-molpcba while accuracy is for the others. Since COLLAB, IMDB-

BINARY and IMDB-MULTI have no graph node features, we use the one-hot encoding of node degrees as node features following Xu et al..

For experiments on all benchmarks, the learning rate is set to 1×10^{-3}. We adopt Adam (Kingma & Ba, 2015) as optimizer and set weight decay as 1×10^{-6}. Early stopping regularization is employed, where we stop the training if there is no further reduction in the validation loss during 300 epochs. The maximum epoch number is set to 1000. The batch size is set to 1024 on OGB-molpcba, 256 on PROTEINS, NCI1, IMDB-BINARY, IMDB-MULTI, and COLLAB. The detailed hyper-parameter settings on all benchmarks are reported in Tab. S8.

B.2. Node Classification

B.2.1. Benchmark Descriptions

For node classification, we conduct experiments on (**1**) six middle-scale benchmarks: homophilic graphs (Shchur et al., 2019) (AmazonPhoto, AmazonComputers, CoauthorCS, and CoauthorPhysics), heterophilic graphs (Platonov et al., 2023) (Questions, Amazon-ratings, Tolokers, and Minesweeper); (**2**) four large-scale benchmarks: homophilic graphs (Hu et al., 2020) (OGB-arXiv, OGB-proteins), heterophilic graphs (Lim et al., 2021) (arXiv-year, genius).

CoauthorCS and CoauthorPhysics originate from the Microsoft Academic Graph, comprising co-authorship graphs wherein nodes denote researchers and edges denote co-authorships between pairs. Node features encapsulate keyword frequencies extracted from an author's publications. Graph labels categorize the dominant research interest in computer science or physics for each author.

AmazonComputers and AmazonPhoto constitute Amazon co-purchase graphs. Nodes denote products that are available for purchase and edges denote co-purchase relation between pairs of items. Node features are encoded customer review texts corresponding to each product. Graph labels categorize the products.

Questions originates from the Yandex Q question-answering website, comprising user activity graphs over a one-year interval (September 2021 to August 2022). Nodes represent users with an interest in the "medicine". Edges denote one user answered another user's posted question. The associated binary graph classification task involves predicting which users remained active on the website without account deletion or blocking. Node features are derived by averaging FastText word embedding vectors corresponding to user profile descriptions.

Amazon-ratings utilizes the Amazon product co-purchasing network metadata sourced from the SNAP Datasets (Leskovec & Krevl, 2014). Nodes are products with edges encoding frequent co-purchase relations between item pairs. The associated task involves predicting the average reviewer rating for each product, which are grouped into five ordinal rating classes. Node features are derived by averaging FastText embedding representations corresponding to each product's description text.

Tolokers encapsulates crowdsourcing participation data sourced from the Toloka platform. Nodes denote contributors, referred to as "tolokers", involved in at least one out of 13 selected projects. Edges link toloker pairs that have completed the same tasks. The associated binary classification is to predict which tolokers have been banned from projects. Node features are profile attributes and task performance statistics of each toloker.

Minesweeper is a synthetic 100x100 grid network. Nodes denote grid cells. 20% of nodes are randomly designated as mines. The associated prediction task is to classify which nodes are mine or not. For all nodes, input features are initialized as one-hot vectors encoding counts of neighboring mines. The initialized features of 50% of randomly selected nodes are then reset to unknown values. These nodes are indicated by an additional binary indicator.

OGB-arXiv is a citation network of Computer Science papers on arXiv. Nodes represent individual articles whereas directed edges denote one paper citing another. Node features are derived by averaging 128-dimensional word embeddings corresponding to the title and abstract of each publication. The associated multi-class classification is to predict the primary category for arXiv articles across 40 classes.

OGB-proteins is an undirected, weighted graph of proteins. Nodes denote proteins while edges denote biologically meaningful associations between proteins. Edge features represent confidence scores for each association type. Node features are one-hot vectors denoting the species of each protein. The associated multi-label classification is to predict the presence of 112 potential protein functions, formulated as a binary prediction task for each label.

Genius is an online social network. Each node represents a user. The associate binary classification task is to predict whether

Table S8. **Hyper-parameter Setups for $\mathbf{N^2}$**

	#Recursive Steps (L)	Hidden Dim.	State Space Dim.	#Units (k)	#Pseudo Nodes (n_p)	Dropout
Genius	3	128	64	8	256	0.1
arXiv-year	6	128	64	8	300	0.1
OGB-arXiv	5	128	64	8	320	0.3
OGB-proteins	3	128	64	8	256	0.3
questions	6	128	64	8	256	0.1
amazon-ratings	8	64	64	8	256	0.3
tolokers	6	128	64	8	256	0.1
minesweeper	7	64	64	8	128	0.3
CoauthorCS	3	128	64	8	256	0.4
CoauthorPhysics	3	128	64	8	256	0.5
AmazonPhoto	5	128	64	8	256	0.1
AmazonComputers	3	128	64	8	256	0.3
PROTEINS	8	128	64	8	32	0.1
NCI1	6	128	128	8	32	0.1
COLLAB	6	128	64	8	32	0.1
IMDB-BINARY	6	128	64	8	32	0.1
IMDB-MULTI	6	128	64	8	8	0.2
OGB-molpcba	6	128	64	8	32	0.3

the accounts are marked or not.

arXiv-year is a citation network from arXiv. Each node represents a research publication. The associate classification task is to predict the publication time of each node.

B.2.2. Experimental Setups

For baseline models, we consider (**1**) convolutional GNNs: GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), APPNP (Gasteiger et al., 2018), MixHop (Abu-El-Haija et al., 2019), SGC (Wu et al., 2019), H2GCN (Zhu et al., 2020), FAGCN (Bo et al., 2021), LINKX (Lim et al., 2021), GPRGNN (Chien et al., 2022), and GloGNN (Li et al., 2022); (**2**) Graph transformers: GT (Shi et al., 2021), Graphormer (Ying et al., 2021), SAN (Kreuzer et al., 2021), GraphGPS (Rampášek et al., 2022), Nodeformer (Wu et al., 2022b), NAGphormer (Chen et al., 2023), SGFormer (Wu et al., 2023b) and Exphormer (Shirzad et al., 2023) based on pseudo node.

ROC-AUC is reported for Questions, Tolokers, and Minesweeper which accuracy is for the others. We apply 60%/20%/20% train/val/test random splits for Amazon and Coauthor benchmarks and follow the standard splits as the original papers for the rest benchmarks. We reproduce the results of Exphormer (Shirzad et al., 2023) on (Genius, arXiv-year, OGN-Proteins, Questions, Amazon-ratings, Tolokers, Minesweepr), Nodeformer (Wu et al., 2022b) and SGFormer (Wu et al., 2023b) on (Genius and arXiv-year) with their released code for a fair comparison.

For experiments on all benchmarks, the learning rate is set to 1×10^{-3}. We adopt Adam (Kingma & Ba, 2015) as optimizer and set weight decay as 1×10^{-6}. Early stopping regularization is employed, where we stop the training if there is no further reduction in the validation loss during 300 epochs. The maximum epoch number is set to 1000. The detailed hyper-parameter settings on all benchmarks are reported in Tab. S8.

C. Additional Experimental Results

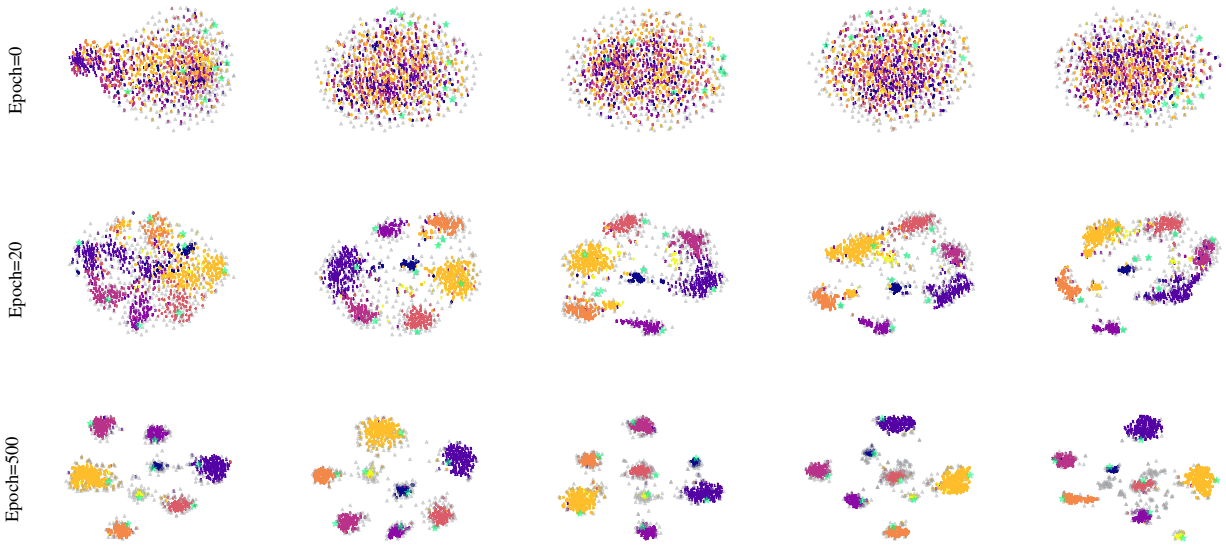
C.1. Effectiveness on Tackling Over-squashing

We further benchmark $\mathbf{N^2}$ on Peptides-struct (Dwivedi et al., 2022), a graph regression benchmark involving long-range interactions. Baselines include traditional GNNs: GCN (Kipf & Welling, 2017), GCNII (Chen et al., 2020), GINE (Xu et al., 2019; Hu* et al., 2019), and GatedGCN (Bresson & Laurent, 2018); methods aiming to capture long-range features: SAN (Kreuzer et al., 2021), PathNN (Sun et al., 2022), Drew-GCN (Gutteridge et al., 2023), and Exphormer (Shirzad et al.,

Table S9. **Effectiveness Study on Peptides-struct (measured by mean absolute error.**

	PEPTIDES-STRUCT ↓
GCNII	34.71
GCN	34.96
GINE	35.47
GATEDGCN	33.57
SAN	25.45
PATHNN	25.45
DREW-GCN	25.36
EXPHORMER	24.81
N^2 (OURS)	25.12

2023). As presented in Tab. S9, we can see that N^2 achieves competitive performance with Exphormer and surpasses the rest baselines. This indicates the effectiveness of N^2 to encounter over-squashing.



(a) 0 (b) 1 (c) 2 (d) 3 (e) 4

Figure S10. **Full Results on the Distribution of Embedded Nodes.** The t-sne (Van der Maaten & Hinton, 2008) results under different recursive steps across training are compared. Each row is from a single epoch. Each column depicts results from the same recursive step. Input nodes with different labels are depicted as: **0**, **1**, **2**, **3**, **4**, **5**, **6**. Pseudo nodes are depicted as △. Class nodes are depicted as ⋆.

C.2. Distribution of Embedded Nodes

The distribution of embedded nodes through training is visualized in Fig. S10. We can see that pseudo nodes and nodes adjust their relative position actively in the state space. For the same recursive step through training, pseudo nodes are split into several groups. Each group is attracted toward a distinct node cluster. For the same epoch, the relative positions between node clusters and attracted pseudo nodes also evolve through recursive steps. This indicates that N^2 optimizes the proximity between nodes and pseudo nodes recursively, constructing dynamic message-passing pathways.

C.3. Proximity for Message Passing

To further analyze the message passing between nodes and pseudo nodes, we visualize their corresponding proximity on AmazonPhoto. In Fig. S11, 1000 nodes are sampled randomly and ranked based on their intro-/outre-proximity summation. The intro-/outre-proximity summation indicates the message load a node takes or emits during the message passing. From

Table S10. **Ablation Studies on Messages.**

	W. MESSAGES	W/O. MESSAGES	W/O.-W.
QUESTIONS	**78.07**	76.01	-2.06
AMAZON-RATINGS	**50.25**	47.79	-2.46
TOLOKERS	**86.25**	85.05	-1.20
MINESWEEPER	**93.97**	92.38	-1.59
COAUTHORCS	**94.44**	92.39	-2.05
COAUTHORPHYSICS	**97.56**	96.27	-1.29
AMAZONPHOTO	**95.75**	93.80	-1.95
AMAZONCOMPUTERS	**92.51**	90.73	-1.78

Fig. S11, we can see that the intro-/outre-proximity is distributed evenly across different nodes and pseudo nodes. This indicates that both nodes and pseudo nodes assume a balanced message load. For each proximity matrix, the proximity value varies between different pairs of nodes and pseudo nodes, thus constructing dynamic connections instead of uniform connections.

C.4. Displacements Comparison between Single Shared Recurrent Layer and Multiple Recurrent Layers

To understand the difference in multi-step performance, where $\mathbf{N^2}$ with multiple recurrent layers ($L_p = L$) achieves better performance against $\mathbf{N^2}$ with shared parameters ($L_p = 1$), we further analyze the displacements of embedded nodes on AmazonPhoto and Amazon-ratings. Fig. S12 depicts the comparison results. Four displacement types are presented in the figure, including displacements of pseudo nodes in pseudo-node adaptation $\Delta\hat{\mathbf{R}}^{(l)}$ (Eq. 8) and global message passing $\Delta\mathbf{R}^{(l)}$ (Eq. 10), displacements of nodes in local communication $\texttt{NL}(\hat{\mathbf{X}}^{(l)})$ (Eq. 9) and global communication $\mathbf{S}^{(l)}$ (Eq. 10). We take the Frobenius norm of each displacement matrix and divide it by the larger results between $\mathbf{N^2}$ with $L_p = 1$ and $\mathbf{N^2}$ with $L_p = L$ across recursive steps, *e.g.* $\texttt{max}\left(\{\mathbf{S}^{(l)}_{\texttt{single}}, \mathbf{S}^{(l)}_{\texttt{deep}}, l \in [1, L]\}\right)$.

From Fig. S12, we can see the consistency in the shape of the displacement curve between Amazon-ratings and AmazonPhoto. The displacement curves of $\mathbf{N^2}$ with $L_p = 1$ are smooth, whereas $\mathbf{N^2}$ with $L_p = L$ demonstrates fluctuation. We attribute this difference to the distinct inertia characteristics exhibited by the embedded nodes. For embedded nodes in $\mathbf{N^2}$ with $L_p = 1$, they generally possess greater inertia, and thus tend to maintain current dynamics through recursive steps. In contrast, embedded nodes in $\mathbf{N^2}$ with $L_p = L$ have smaller inertia and vary their dynamics. Therefore, $\mathbf{N^2}$ with $L_p = L$ can adjust the displacements of embedded nodes more flexibly according to different situations and gain better performance in situations requiring precise distribution optimization. A step-dependent parameter may further improve the performance of $\mathbf{N^2}$ with $L_p = 1$ on a larger number of recursive steps. We leave this for future exploration.

C.5. Ablation on Proximity Measurement

To approximate non-linear relationships with low complexity, we employ piece-wise weighted inner products on the proximity measurement1. Each embedded node is divided into k units. Ablation studies are conducted on k with Amazon-ratings, AmazonPhoto, and PROTEINS. As depicted in Fig. S13, $\mathbf{N^2}$ with multiple units outperforms single unit on all three benchmarks and gains improvement with k increasing. The optimal number of units k is around 8.

C.6. Ablation on Messages

In $\mathbf{N^2}$, graph nodes and pseudo nodes perform message passing based on their current states in the common state space and pass on learned messages to each other. States and messages take different roles in our proposed method. Specifically, states are the descriptive embeddings of nodes corresponding to specific tasks, and messages are employed for information exchange. For the message passing from graph nodes to pseudo nodes, the messages contain the features of an input graph. From pseudo nodes to graph nodes, the messages may contain the query information towards input graphs. To evaluate the necessity of distinguishing states and messages, we conduct ablation studies on messages by directly passing node states in message passing. The results with or without functional features are presented in Tab. S10. We can see that applying both states and messages achieves the best performance.

Table S11. **Comparison between Dense and Pseudo Node-based Message Passing.**

	DENSE	$\mathbf{N^2}$
IMDB-B	77.94	**79.95**
IMDB-M	56.41	**57.31**
PROTEINS	76.94	**77.53**

C.7. Ablation on Pseudo Nodes

In addition to investigating the number of pseudo nodes, we conduct ablation studies to examine the impact of including pseudo nodes within $\mathbf{N^2}$. For cases without pseudo nodes, dense message passing is employed. However, as dense computation is not scalable, only small-scale benchmarks are utilized for this ablation study. The results are presented in Table S11. Surprisingly, we find that $\mathbf{N^2}$ with pseudo nodes outperforms dense computation on IMDB-B, IMDB-M, and PROTEINS. This can be attributed to pseudo nodes functioning as information bottlenecks in global message passing, facilitating the removal of redundant information while extracting discriminative features from inputs.

D. Intuitions for the $\mathbf{N^2}$ Implementation

We provide more intuitions for the implementation in Section 4. $\mathbf{N^2}$ embeds graph nodes and pseudo nodes into the common state space, employing a recurrent layer to parameterize the displacements of embedded nodes. The recurrent layer includes pseudo-node adaptation and dynamic message passing. Pseudo-node adaptation employs `GlobMP` to generate query messages toward graph nodes. Dynamic message passing then extracts graph features through `LocalMP` and refines the extract features at the pseudo-node level with `GlobMP`.

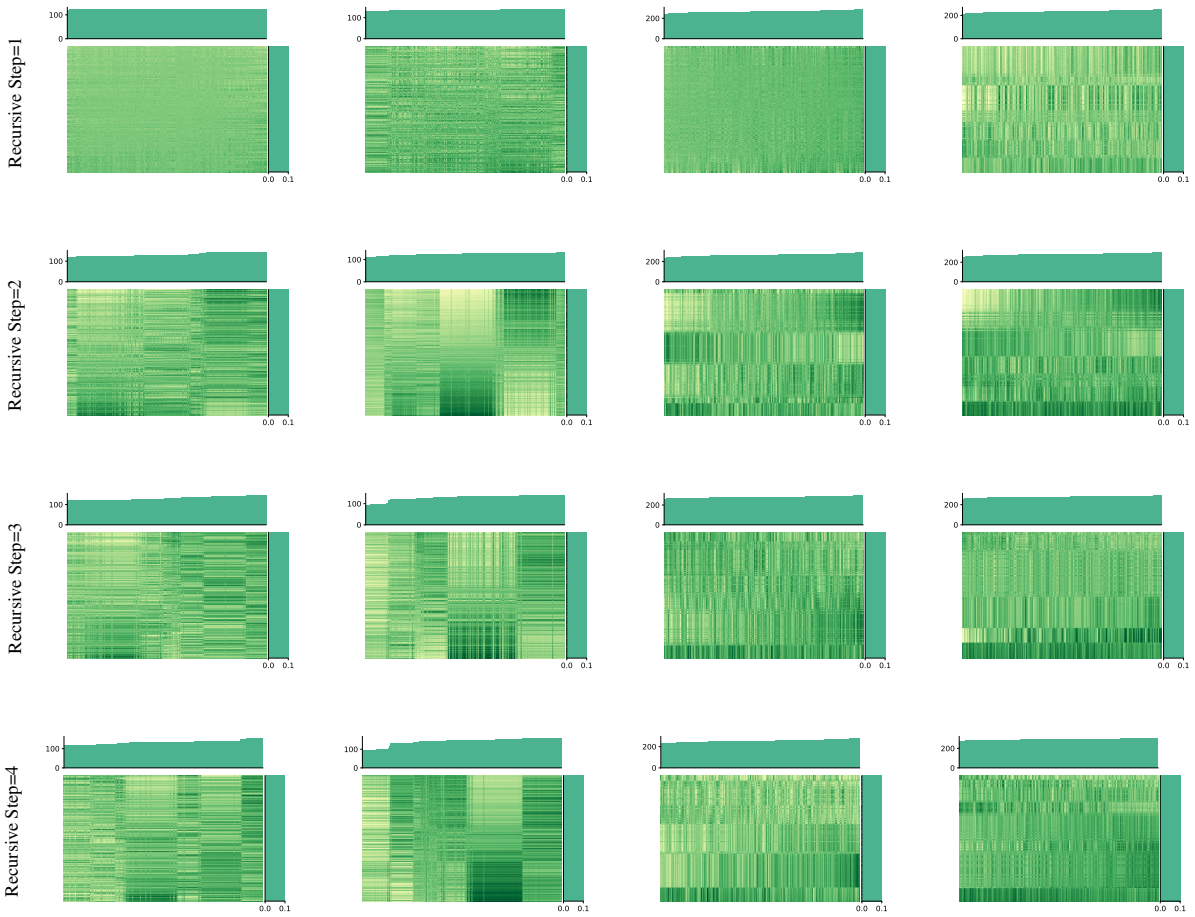
As described by the pseudo-node adaptation in Eq. 8, pseudo nodes first aggregate messages from graph nodes. Based on the information learned from these collected messages, pseudo nodes generate displacements $\Delta\mathbf{R}$ to adjust their own representations, enabling better interactions with graph nodes. Subsequently, pseudo nodes emit responding messages $\mathbf{S}$ back to the graph nodes, probing for specific information of the input graphs.

During the dynamic message passing, Eq. 9 presents the local message passing on input graphs. Graph nodes receive query messages emitted from pseudo nodes, and process them together with their own generated messages and states through the `LocalMP` function. As a result, graph nodes learn the features of the input graphs and generate feedback messages to pseudo nodes accordingly. The generated messages containing the features of input graphs can also be leveraged to determine the displacements of graph nodes.

Eq. 10 presents how graph nodes perform global message passing intermediately through pseudo nodes. Pseudo nodes receive the feedback messages and again generate their displacements. As feedback messages are aggregated at the pseudo-nodes level, more global information is incorporated, guiding the movements of graph nodes and the generation of their messages.

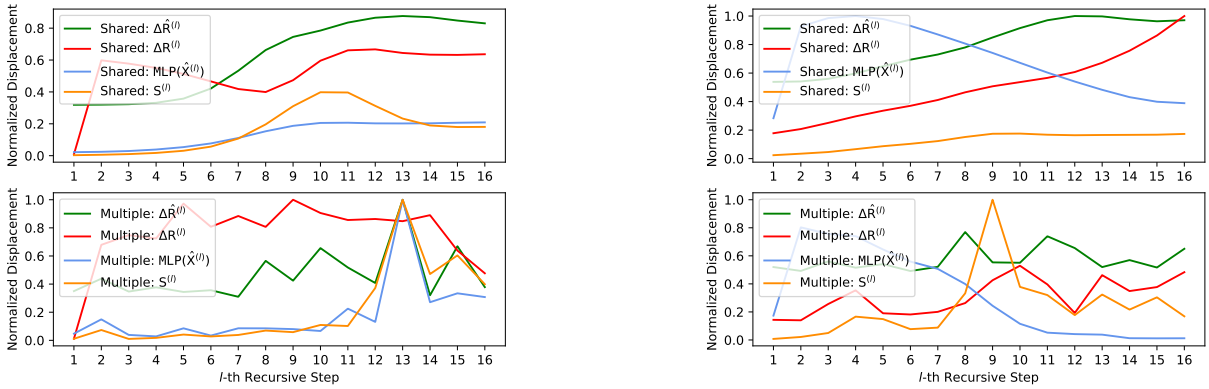
E. Limitations

$\mathbf{N^2}$ employs shared parameters to update the distribution of pseudo nodes and graph nodes recursively. We only studied $\mathbf{N^2}$ with a simple update mechanism. As discussed in the model analysis section, $\mathbf{N^2}$ obtains degradation in performance when the number of recursive steps increases. In addition, the learnable pseudo nodes can be regarded as parameters in GNNs, classified in line with neurons of GNNs. Under this interpretation of pseudo nodes, only a subset of neurons from GNNs are embedded in the common state space within our framework. Comprehensively bridging all neurons and graph nodes remains an open research direction.



(a) PNode. Adapt. $\mathbf{E}^{\mathtt{np}}$ (b) PNode Adapt. $\mathbf{E}^{\mathtt{pn}\top}$ (c) Dy. MP. $\mathbf{E}^{\mathtt{np}}$ (d) Dy. MP. $\mathbf{E}^{\mathtt{pn}\top}$

Figure S11. **Full Results on the Message Passing Analysis.** The results under different recursive steps across training are compared. Each row is from a single epoch. Each column depicts results from the same recursive step. PNode. Adpat. denotes the pseudo node adaptation module in $\mathbf{N^2}$. Dy. MP. denotes the dynamic message passing in $\mathbf{N^2}$. The proximities between sampled nodes and pseudo nodes are depicted in the center of each sub-figure, where darker green indicates higher proximity and brighter green indicates lower proximity. Each column of the proximity matrix associates with a node while each row associates a pseudo node. The distribution on the top of each sub-figure denotes the sum of proximity for each node while the distribution on the right is for each pseudo node. The sampled nodes are ranked based on the sum of proximity.



(a) Amazon-ratings (b) AmazonPhoto

Figure S12. **Displacement Comparison between $\mathbf{N^2}$ with a Shared Recurrent Layer and $\mathbf{N^2}$ with Multiple Recurrent Layers.**

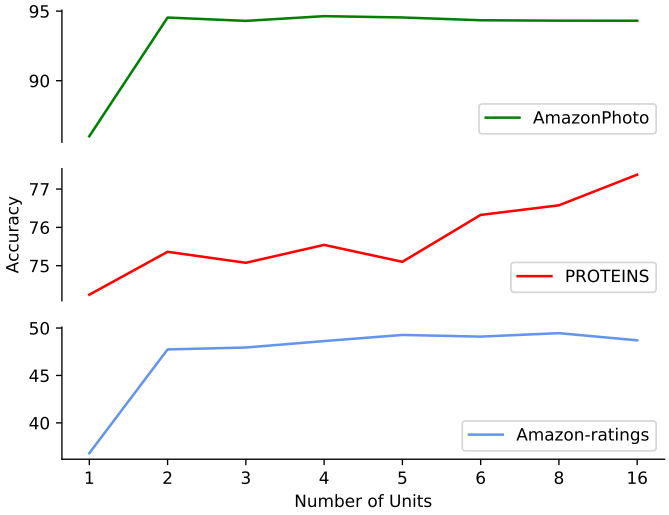


Figure S13. **Ablation Studies on the Number of Units k.**