EC-Gate: Expansion Contribution-Aware Gating for Structure-Guided Message Passing in GNNs

Yutong Zheng
Imperial College London
yutong.zheng120@imperial.ac.uk

Xiangyu Zhao
Imperial College London
x.zhao22@imperial.ac.uk

Danilo Mandic
Imperial College London
d.mandic@imperial.ac.uk

Abstract

The message passing framework has largely driven the success of GNNs, yet it faces potential limitations: over-squashing, over-smoothing and expressiveness constraints. A promising solution is structure-guided message passing, which leverages the graph structure to guide information flow and better capture long-range dependencies. We present EC-Gate, a lightweight plug-in that leverages Expansion Contribution (EC)—a layer-wise measure of how edges expand the receptive field—to drive group-wise gates that regulate message propagation. By concentrating capacity on structurally critical edges, EC-Gate can improve the sensitivity bound in large hidden dimensions, while limiting overfitting. EC-Gate delivers significant improvements across synthetic and molecular benchmarks. Remarkably, when implemented on a standard GCN backbone, it achieves state-of-the-art performance on PCBA and competitive results on Lipo and AqSol, showing that EC serves as a strong structural prior. Furthermore, the empirical analysis of gate activations reveals how EC-Gate modulates message passing in an anisotropic manner.

1 Introduction

Graph Neural Networks (GNNs) have emerged as a dominant approach for learning tasks on graph-structured data. Their success largely stems from the message-passing framework, which forms the backbone of most GNN architectures. Message Passing Neural Networks (MPNNs) operate by iteratively updating node representations based on messages aggregated from neighbors. The framework adapts to diverse graphs and ensures consistency across isomorphic graphs. However, this paradigm exhibits several weaknesses—including limited expressivity [1, 2], over-smoothing [3, 4], where node representations become overly similar across the graph, and over-squashing [5, 6], where information from exponentially expanding neighborhoods is compressed into fixed-size embeddings. These limitations hinder the ability of GNNs to capture long-range dependencies, which are crucial for modeling complex systems and physical, chemical, and biological phenomena [7].

To improve the modeling of long-range dependencies, a widely explored solution is graph rewiring [6–11], which enhances connectivity by modifying the graph structure to facilitate information flow across distant nodes. However, rewiring presents limitations in many practical settings. Creating shortcuts between distant nodes may disrupt the original topology of the graph, undermine the iterative diffusion process, and create ambiguity in assigning features to synthetic edges that lack clear semantic grounding. Empirical studies across multiple benchmarks [12–15] further report that standard GNNs often remain competitive with, or even outperform, Graph Transformers [16, 17], as an extreme form of rewiring, highlighting the importance of preserving the native graph structure over excessive connectivity modifications.

Instead of rewiring, an alternative is to refine the information flow within the original graph through anisotropic message passing (also called anisotropic aggregation), such as GAT [18, 19] and GatedGCN [20], adaptively weight neighbour contributions based on both source and target node features. Structure-based methods incorporate topological cues, either by directly modulating edge weights

Zheng et al., EC-Gate: Expansion Contribution-Aware Gating for Structure-Guided Message Passing in GNNs. *Proceedings of the Fourth Learning on Graphs Conference (LoG 2025)*, PMLR 338, Hybrid Event, December 10–12, 2025.

(e.g., based on degrees [21], substructure counts [22, 23], or Laplacian eigenvectors [24]) or by enriching nodes with positional encodings [16, 25]. In addition to purely topological cues, Euclidean space graphs such as meshes in physical simulations provide natural directional information through relative positions, closely aligned with how physical quantities evolve across the system [26–28]. Previous work in molecular dynamics shows that the incorporation of such geometric signals is critical to accurate and stable predictions [29–31]. However, such geometric signals are not always available or sufficient, motivating further exploration of how structural properties can guide anisotropic message passing.

Motivated by the idea of leveraging structural signals for anisotropic message passing, we propose Expansion Contribution-Aware Gating (EC-Gate), a lightweight plug-in that modulates message flow according to an edge's Expansion Contribution—that is, the number of previously unreachable nodes it introduces into a target's receptive field at each layer. High-Expansion Contribution edges directly channel long-range signals and often emerge as bottlenecks connecting distant regions, while zero-contribution edges largely trace indirect routes that reinforce local refinement. Additionally, Expansion Contribution varies across layers, aligning with the iterative diffusion characteristic of message passing frameworks.

In this paper, we make the following contributions:

- We introduce Expansion Contribution, a novel edge-level label, and develop EC-Gate, a lightweight and group-wise gating mechanism that integrates seamlessly with existing message-passing GNNs.
- We theoretically show that EC-Gate can enhance expressiveness and improve the sensitivity bound by widening structurally critical edges, thereby mitigating over-squashing while avoiding overfitting.
- Empirical results show that EC-Gate significantly and consistently improves the performance of message-passing architectures across synthetic and molecular benchmarks, achieving state-of-the-art results in the non-pretrained setting on PCBA and remaining competitive on Lipo and AqSol. Empirical analysis of gate activations reveals that EC-Gate modulates message passing anisotropically.

2 Related Work

2.1 Selection Mechanism in GNNs

A selection mechanism regulates how information propagates, appearing in various GNN forms such as the channel-wise gating in GatedGCN [20] and attention-based neighbor weighting in GAT [18, 19]. Despite their conceptual simplicity, these mechanisms continue to achieve top performance in recent benchmarks of classic GNNs [14, 15] and have been adopted in competitive application models [32]. The graph gradient has been used to construct a gate that adaptively determines whether a node should update its representation with the aggregated messages [33]. MTGCN [34] introduces semantic tracks for message routing and filters propagation based on learned category-aware affiliations. Other recent approaches shift selection from individual neighbors to higher-level units, using state space models to process compressed node sets or sequential subgraph embeddings [35–37]. Nevertheless, most of these methods depend heavily on node features and often overlook the structural richness of the graph.

2.2 Selective Width Expansion GNNs

One way to alleviate over-squashing is through width expansion, but it must be applied carefully to avoid overfitting. To the best of our knowledge, only PANDA selectively expands the hidden dimensions for nodes identified as potential bottlenecks, where bottlenecks are determined based on pre-selected node centrality metrics [38]. However, this mechanism has two key limitations: (1) Bottlenecks in a graph are dynamic, evolving with information flow during message passing. For example, a node connected to a high-degree neighbor can still become a bottleneck by receiving a large amount of information from that neighbor. Simply expanding the hidden state of the high-degree node itself does not address this issue completely. (2) Not all tasks prioritize long-range dependencies. In some cases, distant nodes introduce noise rather than useful information, suggesting the need for a gating mechanism.

3 Preliminaries

3.1 Notations.

A graph is represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges defining connectivity. The adjacency matrix $\mathbf{A} \in \{0,1\}^{N\times N}$ encodes an unweighted graph, where $N = |\mathcal{V}|$ is the total number of nodes. Each node $v \in \mathcal{V}$ and edge $e \in \mathcal{E}$ is associated with features $\mathbf{h}_v \in \mathbb{R}^D$ and $\mathbf{h}_e \in \mathbb{R}^C$, respectively. For $v \in \mathcal{V}$, let $\mathcal{N}(v)$ denote the 1-hop neighbors of v, and $\mathcal{R}^l(v)$ represent its receptive field at layer l. Formally, $\mathcal{R}^l(v) = \{u \in \mathcal{V} : d(v,u) \le l\}$, where $d(v,u)$ is the shortest-path distance between v and u. This receptive field includes all nodes whose features contribute to the embedding of v at layer l.

3.2 Message Passing GNNs.

The Message Passing Neural Network (MPNN) framework defines node updates using a two-stage process: message aggregation followed by feature update [39]. Specifically, node representations are computed by stacking L layers of the form:

$$\mathbf{m}_v^l = \mathrm{agg}\left(\left\{f^l\left(\mathbf{h}_u^{l-1}, \mathbf{h}_v^{l-1}, \mathbf{h}_{e_{uv}}\right) : e_{uv} \in \mathcal{E}\right\}\right) \tag{1}$$

$$\mathbf{h}_v^l = \mathrm{com}^l\left(\mathbf{h}_v^{l-1}, \mathbf{m}_v^l\right) \tag{2}$$

for $l = 1, \ldots, L$, where $f^l(\cdot)$ computes messages from neighbors, $\mathrm{agg}()$ is a permutation-invariant aggregator (e.g., $\sum$, $\max$), and com^l updates the node state. After l layers, $\mathbf{h}_v^l$ contains information from the l-hop neighborhood. A graph-level representation is then obtained via pooling (e.g., mean, sum). Various GNN models emerge as specific cases of the MPNN framework [40]. A common form of aggregation uses predefined scalar weights to combine messages from neighboring nodes, as in $\mathrm{agg}^l\left(\left\{f^l(\cdot)\right\}\right) = \sum_{u\in\mathcal{N}(v)} f^l\left(\mathbf{h}_u^{l-1}\right)$. This can be equivalently written using a Graph Shift Operator (GSO) as $\sum_{u=1}^{N} \mathbf{A}_{vu} f^l\left(\mathbf{h}_u^{l-1}\right)$, where $\mathbf{A}_{vu} \neq 0$ if and only if $(u,v) \in \mathcal{E}$.

3.3 Sensitivity Bound.

The phenomenon of over-squashing can be understood in terms of the sensitivity of $\mathbf{h}_v^l$ to an input feature $\mathbf{h}_u^0$ [6], where node u is at a distance l from node v. Building on this, Di Giovanni et al. [41] analyzed the role of the network width in mitigating over-squashing through the Jacobian sensitivity bound:

$$\left\| \frac{\partial \mathbf{h}_v^{(l)}}{\partial \mathbf{h}_u^{(0)}} \right\|_{L_1} \le \underbrace{(\kappa_\sigma w p)^l}_{\text{model}} \underbrace{(\mathbf{S}_{r,a}^l)_{vu}}_{\text{topology}} \tag{3}$$

where κ_σ is the Lipschitz constant of the nonlinearity, w is the largest entry across all weight matrices, p is the hidden dimension width, and l is the network depth. $\mathbf{S}$ is a graph shift operator, defined as $\mathbf{S}_{r,a} = \alpha_r \mathbf{I} + \alpha_a \mathbf{A}$, where the c_r and c_a represent the contributions of the residual and the aggregation term. The bound is derived under degree-normalized adjacency. In a graph $\mathcal{G}$, where $\mathbf{S}_{r,a}^l$ decays exponentially with l, increasing the width p allows the MPNN to counteract this decay and mitigate over-squashing. However, this comes with the risk of overfitting and poorer generalization [41].

4 Proposed Method

We will first introduce the edge labeling process, followed by the framework details, and conclude with a theoretical explanation of how our method mitigates over-squashing by selectively expanding the message-passing width along key edges. Our implementation is available at GitHub.

4.1 Edge Labeling and Expansion Contribution

At layer l, an edge e_{uv} either enlarges the receptive field of v, bringing in nodes that were unreachable at layer $l-1$, or routes information already accessible via other paths. We formalize this difference with an edge-wise discrete label, the *Expansion Contribution (EC)*, defined as follows.

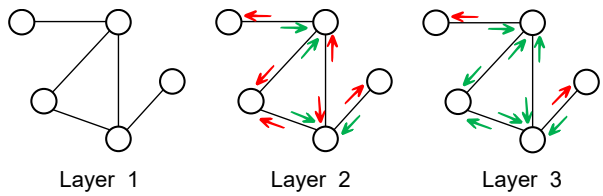


Figure 1: Illustrative example of layer-wise expansion behavior (with edge labels: for EC > 0, for EC $= 0$). At Layer 1, message passing is in its initialization stage, where all nodes carry new information and EC is indistinguishable. At higher layers, once the receptive field surpasses the graph diameter, no new expansions occur; in this regime, accumulated historical EC statistics can be used as guidance.

Algorithm 1 Expansion Contribution (EC) via Boolean Adjacency

Require: Adjacency $\mathbf{A} \in \{0,1\}^{N \times N}$, max depth L
Ensure: Edge-wise EC $\{c_{uv}^{(\ell)}\}$
▷ Boolean operations: OR/AND/NOT as $\vee, \wedge, \neg$;
▷ $\mathbf{H}^{(\ell)}$: reachable at exactly ℓ hops; $\mathbf{C}^{(\ell)}$: receptive field (reachable within $\leq \ell$)
$\mathbf{H}^{(1)} \leftarrow \mathbf{A}, \quad \mathbf{C}^{(0)} \leftarrow \mathbf{I}$ ▷ init
for $\ell = 2$ **to** L **do**
  $\mathbf{H}^{(\ell)} \leftarrow \mathbf{H}^{(\ell-1)}\mathbf{A}$ ▷ exactly-ℓ reachability
  $\mathbf{C}^{(\ell)} \leftarrow \mathbf{C}^{(\ell-1)} \vee \mathbf{H}^{(\ell)}$ ▷ cumulative reachability
  $\Delta^{(\ell)} \leftarrow \mathbf{H}^{(\ell)} \wedge \neg\mathbf{C}^{(\ell-1)}$ ▷ new-at-ℓ frontier
  for all (u, v) with $\mathbf{A}[u, v] = 1$ **do**
    $c_{uv}^{(\ell)} \leftarrow \left\| \mathbf{C}^{(\ell-1)}[u,:] \ \wedge \ \Delta^{(\ell)}[v,:] \right\|_1$ ▷ expansion contribution
  end for
end for
return $\{c_{uv}^{(\ell)}\}$

Definition 1 (Expansion Contribution)**.** *Let $\mathcal{R}^l(v)$ denote the receptive field of node v at layer l. For an edge e_{uv} (message $u \to v$), its Expansion Contribution at layer l is*

$$c_{uv}^l = \left|\mathcal{R}^{l-1}(u) \ \cap \ \left(\mathcal{R}^l(v) \setminus \mathcal{R}^{l-1}(v)\right)\right| \ \in \ \mathbb{N}.$$

This scalar quantifies how many *previously unreachable nodes* are brought into the receptive field of v via e_{uv} at layer l. Intuitively, edges with larger c_l lie on shortest routes with few intermediate nodes, so they deliver long-range information with minimal dilution. In contrast, edges with $c_l = 0$ mainly carry information that has already arrived via other paths, often after passing through more intermediates, or return previously propagated signals as feedback. A visual example of how Expansion Contribution evolves across layers is illustrated in Figure 1. In practice, this value modulates message propagation via a gating mechanism (see Section 4.2).

The expansion contribution of each edge does not depend on learnable model parameters and can thus be predetermined. For implementation details, see Algorithm 1. The algorithm computes adjacency powers $\mathbf{A}^\ell$ to capture ℓ-hop connectivity and identifies new connections at each layer by comparing with previous powers $\mathbf{A}^{l-1}$. These operations consist of matrix multiplications and logical operations over Boolean matrices, keeping the per-layer cost relatively lightweight. For large graphs, more scalable alternatives such as breadth-first search can be employed.

Definition 1 implies the following coverage property.

Proposition 1 (Shortest-Path Coverage)**.** *Let $w, v \in \mathcal{V}$ with graph distance $d(w, v) = m$, and let $\pi = \{e_{x_i x_{i+1}}\}_{i=0}^{m-1}$ be any shortest path from $w = x_0$ to $v = x_m$. Then every edge on π has nonzero expansion contribution at the corresponding layer:*

$$c_{x_i x_{i+1}}^{i+1} \ \geq \ 1, \qquad i = 0, 1, \ldots, m-1.$$

Proof. Fix $i \in \{0, \ldots, m-1\}$ and take $z = w = x_0$ as a witness node. Along a shortest path, $d(x_{i+1}, z) = i+1$, hence $z \in \mathcal{R}^{i+1}(x_{i+1}) \setminus \mathcal{R}^{i}(x_{i+1})$. Also $d(x_i, z) = i$, so $z \in \mathcal{R}^{i}(x_i)$. By Definition 1, this gives $c^{i+1}_{x_i x_{i+1}} \geq 1$. □

4.1.1 Comparison with Weisfeiler-Lehman (WL) Test

The Weisfeiler–Lehman (WL) test is a widely used, efficient tool est for graph isomorphism, but it cannot distinguish certain non-isomorphic graph pairs. This limitation motivates the search for methods with stronger discriminative power. Figure 2 illustrates a representative pair of graphs that the 1-WL test fails on, but which can be separated by our method.

Proposition 2. *Augmenting a GNN that is as expressive as the 1-WL test with expansion contribution yields strictly stronger than the 1-WL test in distinguishing non-isomorphic graphs.*

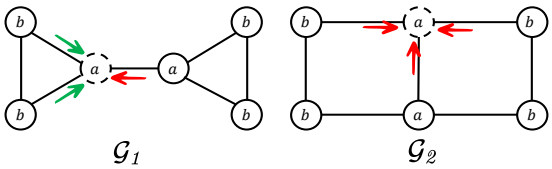


Figure 2: Two graphs indistinguishable under the 1-WL test but separated by expansion contribution (EC) values at the second layer.

4.2 EC-Gate: Expansion Contribution-Aware Gating

At each layer we modulate incoming messages with a *multiplicative gate* that depends only on the edge-wise expansion contribution c^{l}_{uv}. To keep the parameter overhead low, we adopt a *grouped* design: every hidden vector $\mathbf{h}^{l-1}_u \in \mathbb{R}^D$ is split into G disjoint channel groups, and a single scalar gate is shared within each group. The aggregated message is then computed as

$$\mathbf{m}^{l}_{v} = \sum_{u \in \mathcal{N}(v)} \bigoplus_{k=1}^{G} g_k(c^{l}_{uv})\, f^{l}\left(\mathbf{h}^{l-1}_{u}\right)_{[k]}, \tag{1}$$

where $g_k : \mathbb{N}_{\geq 0} \to [0,1]$ is a lightweight MLP applied to a compressed version of EC (e.g., $\log(1 + \mathrm{EC})$), followed by a sigmoid activation, $f^{l}\left(\mathbf{h}^{l-1}_{u}\right)_{[k]} \in \mathbb{R}^{D/G}$ is the k-th group, and $\bigoplus$ concatenates the gated groups. This compression stabilizes training and reflects our focus on whether an edge contributes new information (EC>0) rather than its exact expansion magnitude. Channel-wise gating is a special case of our grouped design with $G = D$. Since c^{l}_{uv} depends only on graph structure, the overall procedure is isomorphism-invariant. This design can be seamlessly integrated into a wide range of message-passing GNNs. In practice, if EC-Gate is applied immediately after nonlinear activations or in models that already employ gating operations (e.g., GatedGCN), it may lead to overly sparse signals and unstable training. In such cases, we precede EC-Gate with a linear projection. Otherwise, EC-Gate can be used directly.

4.3 Improving Sensitivity Bound via EC-Gate

We analyze how EC-Gate mitigates over-squashing through the sensitivity bound in Eq. 3. The exponential decay of information from distant nodes poses a challenge; although a uniform increase in hidden width p alleviates this, it simultaneously increases the risk of overfitting. EC-Gate, instead, can concentrate capacity along structurally critical routes without increasing the global width.

Proposition 3. *A GNN equipped with EC-Gate improves the sensitivity bound by selectively increasing hidden dimension along structurally critical edges, without expanding width uniformly across the graph.*

Proof. Fix the global hidden width p in a degree-normalized GNN with EC gates. Let u, v be nodes with shortest-path distance $l = d(u, v)$, and let $\mathcal{P}^{(l)}_{u \to v}$ denote the set of all $u \to v$ shortest paths of length l. Write $\mathbf{S}_{r,a} = c_r \mathbf{I} + c_a \mathbf{A}$, and let a_e denote the corresponding entry of $\mathbf{S}_{r,a}$ on edge e

From Eq. 3, when $l = d(u, v)$ only length-l terms contribute to $(\mathbf{S}^l_{r,a})_{vu}$, hence

$$\left\| \frac{\partial \mathbf{h}_v^{(l)}}{\partial \mathbf{h}_u^{(0)}} \right\|_{L_1} \leq (\kappa_\sigma w p)^l \sum_{\pi \in \mathcal{P}^{(l)}_{u \to v}} \prod_{e \in \pi} a_e. \tag{4}$$

We now incorporate EC-Gate via the *effective width* on edges. Split the hidden dimension into G equal groups; let $g_k(c_e) \in [0, 1]$ be the group gate on edge e. Define *effective width* as,

$$p_{\text{eff}}(e) = \rho_e \, p, \qquad \rho_e = \tfrac{1}{G} \sum_{k=1}^{G} g_k(c_e) \in [0, 1].$$

Therefore each width term in (4) is reweighted by ρ_e, yielding

$$\left\| \frac{\partial \mathbf{h}_v^{(l)}}{\partial \mathbf{h}_u^{(0)}} \right\|_{L_1} \leq (\kappa_\sigma w)^l \sum_{\pi \in \mathcal{P}^{(l)}_{u \to v}} \prod_{e \in \pi} \big(a_e \, p_{\text{eff}}(e)\big). \tag{5}$$

The bound in (5) remains below or equal to that in (4). Equality holds by setting $\rho_e = 1$ on the shortest path edges, whereas non-participating edges at this layer may remain below 1. Finally, by Proposition 1, shortest-path edges have positive EC at their corresponding layer; As the gates g_k are capable of learning nondecreasing responses to EC, shortest-path edges naturally receive larger ρ_e than indirect-path edges, thereby enabling EC-Gate to selectively increase effective width along structurally critical edges without uniform widening.

To connect EC-Gate with structural bottlenecks, we relate expansion contribution (EC) to edge betweenness centrality (EBC) (see Appendix A.1.1). EBC counts how often an edge lies on shortest paths over all source–target pairs, whereas EC is a local, target-conditioned measure: at layer l, $c^l_{uv} > 0$ precisely when $u \to v$ is a shortest-path step toward v. Summing over layers up to the graph diameter, $\sum_{l=1}^{\text{diam}(\mathcal{G})} c^l_{uv}$, yields a target-restricted analogue of $EBC(e_{uv})$. Edges with consistently large EC values thus coincide with high-EBC "bridge" edges, i.e., structural bottlenecks. Thus, EC-Gate can naturally emphasize these edges when long-range signals transmitted through them contribute positively to the task.

□

5 Experiments

In this section, we first evaluate EC-Gate on controlled synthetic benchmarks to examine its ability to capture long-range dependencies, and then on real-world molecular datasets. Additionally, Appendix A.3.1 provides a CliquePath case study illustrating effective gate evolution and node sensitivity.

5.1 Synthetic Datasets

Setup. We construct a synthetic benchmark to evaluate GNN performance on tasks requiring long-range information, following the task setup as [42], which includes single-source shortest-path (SSSP) length, node eccentricity and graph diameter prediction. These tasks require reasoning over global graph structure rather than relying solely on local neighborhoods. Each node is assigned a random feature drawn from a standard normal distribution, and for SSSP an additional binary feature marks the source node. The dataset contains approximately 10,000 graphs with 50–70 nodes (split 10:1:2 into training, validation, and test), generated from diverse graph models including Erdős–Rényi, Barabási–Albert, stochastic block models, k-Nearest Neighbors, and power-law cluster graphs. As summarized in Table 4 (Appendix) and illustrated in Figure 3, the synthetic graphs are generated using the listed parameters and exhibit diameter distributions mostly between 3 and 10, avoiding trivial cases such as fully connected graphs. For backbone models, we integrate

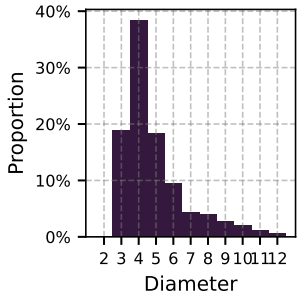


Figure 3: Graph Diameter Distribution in Synthetic Graphs (excluding extremely rare large diameters).

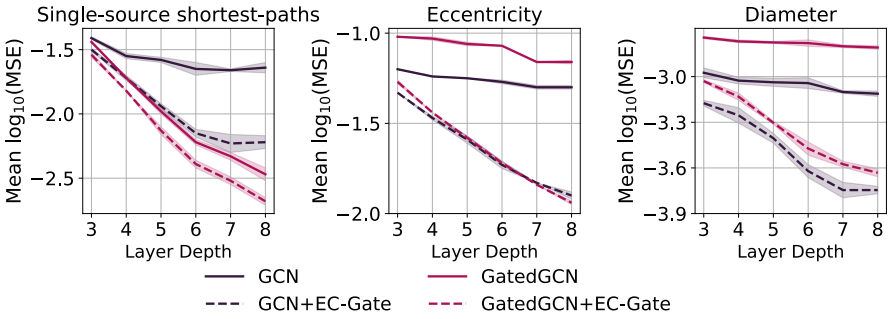


Figure 4: Performance Across Depths for Baseline and EC-Gated Models on Synthetic Tasks.

EC-Gate into GCN and GatedGCN, representing isotropic and gated (anisotropic) message-passing architectures. For depths beyond 5, EC values are approximated by reusing averages from earlier layers to reduce overhead and to examine whether historical expansion statistics may capture deeper edge contributions.

Results. Figure 4 reports test performance across increasing depths for GCN and GatedGCN backbones, with and without EC-Gate. At 8 layers, where under-reaching is no longer a major limitation, EC-Gate achieves substantial error reductions on all tasks, by 73.7% and 38.3% for SSSP, 74.8% and 83.4% for Eccentricity, and 76.7% and 85.0% for Diameter, using GCN and GatedGCN backbones, respectively. Although GatedGCN incorporates feature-based gating, it does not consistently outperform GCN—performing worse on the Eccentricity and Diameter tasks—whereas EC-Gate improves both backbones. This difference arises because in SSSP, the source node is explicitly marked by features, making feature-based gating effective; in contrast, for Eccentricity and Diameter, where node features are random, only structural cues are informative, allowing EC-Gate to excel. Moreover, for depths beyond 5, where EC values are approximated by averaging those from earlier layers, the upward performance trend persists, suggesting that historical expansion statistics capture deeper edge contributions.

5.2 Chemical Datasets

Setup. The second task set consists of real-world molecular graph benchmarks. We use PCBA [43], a multi-task binary classification dataset with 437.9K molecular graphs and 128 bioactivity labels, evaluated by average precision (AP). We also include Lipo [43], a regression benchmark containing experimental octanol–water partition coefficients (lipophilicity) for 4.2K compounds, and AqSol [44], also a regression dataset of 9.8K molecular graphs with experimentally measured aqueous solubility. These molecular properties are considered essential for drug discovery.

PEPTIDES tasks from LRGB [45] are excluded from our evaluation, as recent analysis [46], suggests that these tasks do not substantially involve long-range dependencies and their receptive fields remain largely constrained to one hop throughout training. They can be solved to near state-of-the-art performance with shallow GCNs, such as a 3-layer model [15]. In contrast, PCBA tends to perform better with deeper GNNs (e.g., ≥ 10 layers), as observed from recent hyperparameter sweeps [15]. Additionally, PCBA, Lipo and AqSol present strong overfitting tendencies. These characteristics make them suitable benchmarks to evaluate EC-Gate's ability to enhance long-range dependency modeling while controlling overfitting. Our model does not incorporate any positional or structural encodings, but utilizes edge features. Hyperparameter settings are provided in Table 3.

Results. Tables 1, and 2 present results on PCBA, Lipo, and AqSol, and compare EC-Gated GNNs with previously reported models. On PCBA, EC-Gated GCN achieves an AP of 30.2%, matching the current state of the art and improving by 6.9% over our plain GCN baseline. Notably, our baselines exceed the performance of GCN+ and GatedGCN+, underscoring the strength of our reference point. On Lipo, we compare against models that exploit 3D geometry, including those pretrained on large molecular datasets. Here, the 3D conformations are generated using cheminformatics toolkits, which provide approximate but not fully realistic geometries. EC-Gated GCN and GatedGCN remain competitive, ranking just below Uni-Mol, even though we do not have 3D coordinates. This suggests that expansion contribution provides a structural prior comparable to 3D geometric information,

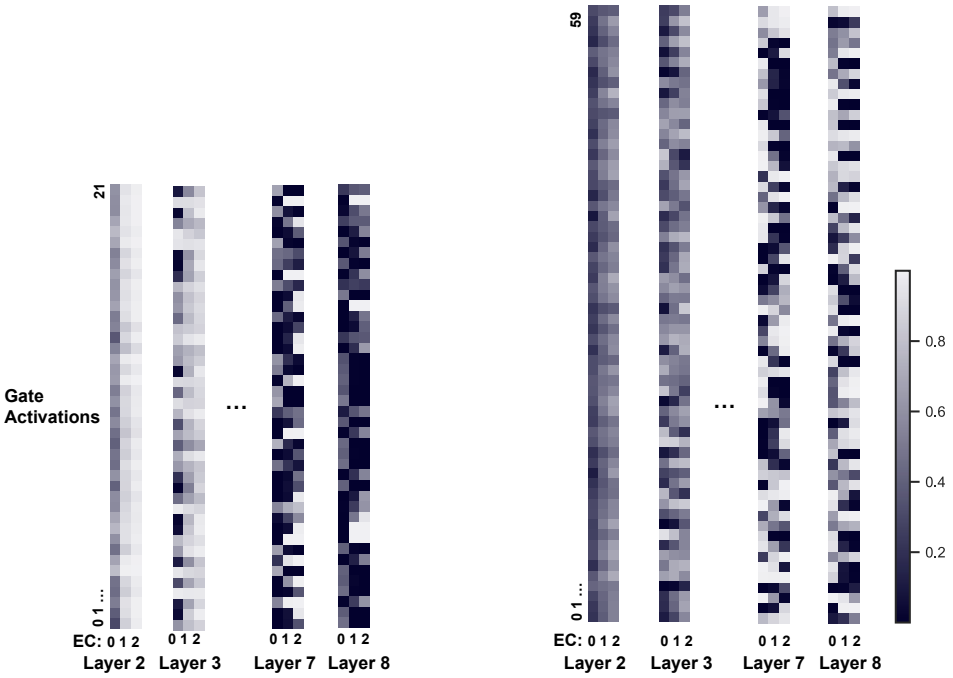
Table 1: Results on PCBA. Top two: **Best**, Second. Following [15].

	Model	PCBA (AP% ↑)
	GraphTrans	$27.61_{\pm 0.29}$
	SAN	$27.65_{\pm 0.42}$
	EGT $_{\text{pre-trained}}$	$28.61_{\pm 0.24}$
	GraphGPS	$29.07_{\pm 0.28}$
	Specformer	$29.72_{\pm 0.23}$
	Exphormer	$28.49_{\pm 0.25}$
	GECO	$29.61_{\pm 0.08}$
	GCN^{+} $_{\text{w/o PE}}$	$26.67_{\pm 0.34}$
	GCN^{+} $_{\text{w PE}}$	$27.21_{\pm 0.46}$
	GatedGCN^{+} $_{\text{w/o PE}}$	$28.13_{\pm 0.22}$
	GatedGCN^{+} $_{\text{w PE}}$	$29.81_{\pm 0.24}$
Ours	GCN	$28.24_{\pm 0.20}$
Ours	GCN $_{\text{EC-Gate}}$	$\mathbf{30.18}_{\pm \mathbf{0.17}}$
Ours	GatedGCN	$28.89_{\pm 0.28}$
Ours	GatedGCN $_{\text{EC-Gate}}$	$29.99_{\pm 0.21}$

Table 2: Performance on Lipo and AqSol. Top two results in each column are highlighted: **Best**, Second. † reported by [43]; ‡ by [44].

	Model	Lipo (RMSE ↓)	AqSol (MAE ↓)
	GCN	$0.797_{\pm 0.02}$ †	$1.372_{\pm 0.02}$ ‡
	GIN	$0.757_{\pm 0.01}$ †	$1.894_{\pm 0.02}$ ‡
	GatedGCN $_{\text{w/ PE}}$	–	$0.996_{\pm 0.01}$ †
	GROVER $_{\text{pre-trained}}$ [47]	$0.823_{\pm 0.01}$	–
	GeoGNN $_{\text{w/ 3D geo.}}$ [48]	$0.666_{\pm 0.03}$	–
	GEM $_{\text{pre-trained, w/ 3D geo.}}$ [48]	$0.660_{\pm 0.01}$	–
	Uni-Mol $_{\text{pre-trained, w/ 3D geo.}}$ [49]	$\mathbf{0.603}_{\pm \mathbf{0.01}}$	–
	GraphTransformer [16]	–	$1.110_{\pm 0.01}$
	Graph-UNets [50]	$0.716_{\pm 0.01}$	$1.063_{\pm 0.02}$
	MeGraph [51]	$0.688_{\pm 0.01}$	$1.002_{\pm 0.02}$
Ours	GCN	$0.715_{\pm 0.016}$	$1.244_{\pm 0.029}$
Ours	GCN $_{\text{EC-Gate}}$	$0.659_{\pm 0.005}$	$0.966_{\pm 0.017}$
Ours	GatedGCN	$0.714_{\pm 0.008}$	$1.228_{\pm 0.024}$
Ours	GatedGCN $_{\text{EC-Gate}}$	$0.645_{\pm 0.009}$	$\mathbf{0.942}_{\pm \mathbf{0.010}}$

and the two can potentially complement each other when combined. On Lipo, our method remains competitive and achieves up to 22.4% improvement over the baseline.



(a) GatedGCN **(b)** GCN

Figure 5: Comparison of learned gate activations across different GNN backbones (GCN vs. GatedGCN) on PCBA. Both backbones exhibit similar trends: at early layers, gates tend to amplify high-EC edges; at deeper layers, they partly suppress high-EC edges.

5.3 Empirical Analysis of EC-Gate Behaviour

Activation Patterns of EC-Gate. We analyze EC-Gate activations across different depths and edge expansion classes, with an example on PCBA shown in Figure 5. EC-Gate partitions each hidden vector into channel groups, with scalar gates that assign distinct weights to each group. At shallow depths (e.g., Layer 2), gate activations remain relatively smooth: most gates respond moderately across edge classes, typically lowering weights on zero-EC edges while raising them on expansion-contributing edges. At deeper layers (e.g., Layer 7–8), the activations become slightly more contrasted and some gates tend to suppress high-EC edges, whereas others show the opposite tendency. This shows that EC-Gate does not indiscriminately amplify distant signals; instead, it selectively filters and strengthens long-range information when beneficial. The pattern that activations vary with EC values is also observed in Lipo and AqSol (see Appendix A.3.2). As a counterexample, models showing nearly indistinguishable activations for inputs with different EC values fail to capture such selective gating behavior.

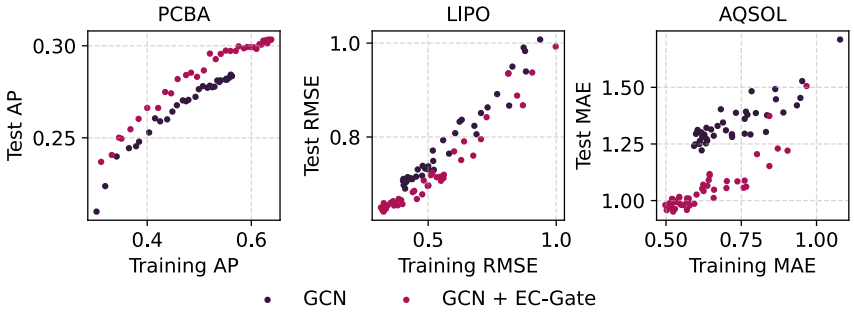


Figure 6: Overfitting Comparison on PCBA, Lipo, and AqSol

Over-smoothing Analysis. Figure 7 compares the Dirichlet energies of hidden representations across layers for GCN and its EC-Gate variant. EC-Gate maintains higher energy across layers, even with residual connections in the baseline, due to its structure-aware selective information reception, which helps mitigate oversmoothing.

Overfitting Analysis. Figure 6 plots training–test performance on three molecular tasks. We observe that, at comparable training levels, EC-Gate consistently achieves better test performance. This observations align with our theoretical analysis in Section 4.3, providing empirical evidence that EC-Gate mitigates overfitting even with large hidden dimensions.

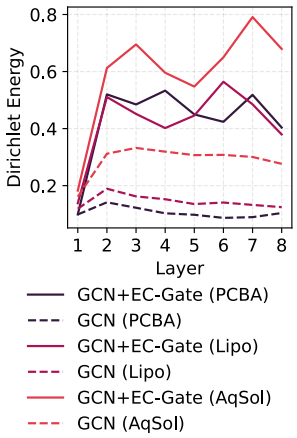


Figure 7: Layerwise DE

6 Conclusion

We introduce a structure-guided message passing approach that selectively captures long-range signals and shows promise in alleviating GNN limitations such as restricted expressivity and over-smoothing. Additionally, we acknowledge that the EC value, as a dynamic edge attribute, can be integrated into general GNN frameworks in various ways. The gating mechanism employed here is designed to enhance theoretical clarity and interpretability.

Limitations and Future Work. EC-Gate performs well across our reported benchmarks, but its benefits decrease on graphs lacking structural information, such as fully connected or regular grid graphs, where nodes share almost identical neighborhood structures. This limitation, shared by other structure-driven methods, means that in such cases, feature- or geometry-based mechanisms can serve as alternatives for anisotropic aggregation. Molecules are inherently three-dimensional, with atomic arrangements in 3D space playing a crucial role in determining their properties. However, widely used benchmarks, such as OGB-MOL, omit 3D geometric information, resulting in evaluations that are primarily 2D. A natural next step is to extend EC-Gate to 3D molecular graphs to quantify the additional value of structural cues when combined with geometry, thus providing a more realistic assessment of its potential impact in practical applications.

References

[1] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv [cs.LG]*, October 2018. 1

[2] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. *Proc. Conf. AAAI Artif. Intell.*, 33(01):4602–4609, July 2019. 1

[3] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. DropEdge: Towards deep graph convolutional networks on node classification. *arXiv [cs.LG]*, July 2019. 1

[4] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. *Proc. Conf. AAAI Artif. Intell.*, 32(1), April 2018. 1

[5] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv [cs.LG]*, June 2020. 1

[6] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv [stat.ML]*, November 2021. 1, 3

[7] Federico Errica, Henrik Christiansen, Viktor Zaverkin, Takashi Maruyama, Mathias Niepert, and Francesco Alesiani. Adaptive message passing: A general framework to mitigate oversmoothing, oversquashing, and underreaching. *arXiv [cs.LG]*, December 2023. 1

[8] Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding oversquashing in GNNs through the lens of effective resistance. *arXiv [cs.LG]*, February 2023.

[9] Kedar Karhadkar, Pradeep Kr Banerjee, and Guido Montúfar. FoSR: First-order spectral rewiring for addressing oversquashing in GNNs. *arXiv [cs.LG]*, October 2022.

[10] Ben Finkelshtein, Xingyue Huang, Michael Bronstein, and İsmail İlkan Ceylan. Cooperative graph neural networks. *arXiv [cs.LG]*, October 2023.

[11] Chendi Qian, Andrei Manolache, Kareem Ahmed, Zhe Zeng, Guy Van den Broeck, Mathias Niepert, and Christopher Morris. Probabilistically rewired message-passing neural networks. *arXiv [cs.LG]*, October 2023. 1

[12] Jan Tönshoff, Martin Ritzert, Eran Rosenbluth, and Martin Grohe. Where did the gap go? reassessing the long-range graph benchmark. *arXiv [cs.LG]*, September 2023. 1

[13] Florian Grötschla, Jiaqing Xie, and Roger Wattenhofer. Benchmarking positional encodings for GNNs and graph transformers. *arXiv [cs.LG]*, November 2024.

[14] Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic GNNs are strong baselines: Reassessing GNNs for node classification. *arXiv [cs.LG]*, June 2024. 2

[15] Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Can classic GNNs be strong baselines for graph-level tasks? simple architectures meet excellence. *arXiv [cs.LG]*, February 2025. 1, 2, 7, 8

[16] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv [cs.LG]*, December 2020. 1, 2, 8

[17] Chen Cai, Truong Son Hy, Rose Yu, and Yusu Wang. On the connection between MPNN and graph transformer. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 3408–3430. PMLR, 2023. 1

[18] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *arXiv [stat.ML]*, October 2017. 1, 2

[19] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv [cs.LG]*, May 2021. 1, 2

[20] Xavier Bresson and Thomas Laurent. Residual gated graph ConvNets. *arXiv [cs.LG]*, November 2017. 1, 2

[21] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv [cs.LG]*, September 2016. 2

[22] John Boaz Lee, Ryan A Rossi, Xiangnan Kong, Sungchul Kim, Eunyee Koh, and Anup Rao. Graph convolutional networks with motif-based attention. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, New York, NY, USA, November 2019. ACM. 2

[23] Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(1):657–668, January 2023. 2

[24] Dominique Beaini, Saro Passaro, Vincent Létourneau, Will Hamilton, Gabriele Corso, and Pietro Liò. Directional graph networks. In *International Conference on Machine Learning*, pages 748–758, 2021. 2

[25] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021. 2

[26] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*. 2

[27] Meire Fortunato, Tobias Pfaff, Peter Wirnsberger, Alexander Pritzel, and Peter Battaglia. MultiScale MeshGraphNets. In *ICML 2022 2nd AI for Science Workshop*.

[28] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, and Others. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023. 2

[29] Kristof T Schütt, Pieter-Jan Kindermans, Huziel E Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. *arXiv [stat.ML]*, June 2017. 2

[30] Johannes Gasteiger, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. *arXiv [cs.LG]*, March 2020.

[31] Johannes Gasteiger, Florian Becker, and Stephan Günnemann. GemNet: Universal directional graph neural networks for molecules. *arXiv [physics.comp-ph]*, June 2021. 2

[32] Yuan Gao, Hao Wu, Ruiqi Shu, Huanshuo Dong, Fan Xu, Rui Ray Chen, Yibo Yan, Qingsong Wen, Xuming Hu, Kun Wang, Jiahao Wu, Qing Li, Hui Xiong, and Xiaomeng Huang. OneForecast: A universal framework for global and regional weather forecasting. *arXiv [cs.LG]*, February 2025. 2

[33] T Konstantin Rusch, Benjamin P Chamberlain, Michael W Mahoney, Michael M Bronstein, and Siddhartha Mishra. Gradient gating for deep multi-rate learning on graphs. *arXiv [cs.LG]*, October 2022. 2

[34] Hongbin Pei, Yu Li, Huiqi Deng, Jingxin Hai, Pinghui Wang, Jie Ma, Jing Tao, Yuheng Xiong, and Xiaohong Guan. Multi-track message passing: Tackling oversmoothing and oversquashing in graph learning via preventing heterophily mixing. In *Forty-first International Conference on Machine Learning*, 2024. 2

[35] Ali Behrouz and Farnoosh Hashemi. Graph mamba: Towards learning on graphs with state space models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, volume 24, pages 119–130, New York, NY, USA, August 2024. ACM. 2

[36] Chloe Wang, Oleksii Tsepa, Jun Ma, and Bo Wang. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *arXiv [cs.LG]*, February 2024.

[37] Yinan Huang, Siqi Miao, and Pan Li. What can we learn from state space models for machine learning on graphs? *arXiv [cs.LG]*, June 2024. 2

[38] Jeongwhan Choi, Sumin Park, Hyowon Wi, Sung-Bae Cho, and Noseong Park. PANDA: Expanded width-aware message passing beyond rewiring. *arXiv [cs.LG]*, June 2024. 2

[39] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. *arXiv [cs.LG]*, April 2017. 3

[40] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv [cs.LG]*, April 2021. 3

[41] Francesco Di Giovanni, Lorenzo Giusti, Federico Barbero, Giulia Luise, Pietro Lio', and Michael M Bronstein. On over-squashing in message passing neural networks: The impact of width, depth, and topology. *ICML*, abs/2302.02941:7865–7885, February 2023. 3, 15

[42] Gabriele Corso, Luca Cavalleri, D Beaini, P Lio', and Petar Velickovic. Principal neighbourhood aggregation for graph nets. *Neural Inf Process Syst*, abs/2004.05718:13260–13271, April 2020. 6

[43] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020. 7, 8

[44] Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv [cs.LG]*, March 2020. 7, 8

[45] Vijay Prakash Dwivedi, Ladislav Rampášek, Mikhail Galkin, Alipanah Parviz, Guy Wolf, A Luu, and D Beaini. Long range graph benchmark. *Neural Inf Process Syst*, abs/2206.08164: 22326–22340, June 2022. 7

[46] Jacob Bamberger, Benjamin Gutteridge, Scott le Roux, Michael M Bronstein, and Xiaowen Dong. On measuring long-range interactions in graph neural networks. *arXiv [cs.LG]*, June 2025. 7

[47] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *arXiv [q-bio.BM]*, June 2020. 8

[48] Xiaomin Fang, Lihang Liu, Jieqiong Lei, Donglong He, Shanzhuo Zhang, Jingbo Zhou, Fan Wang, Hua Wu, and Haifeng Wang. Geometry-enhanced molecular representation learning for property prediction. *Nat. Mach. Intell.*, 4(2):127–134, February 2022. 8

[49] Gengmo Zhou, Zhifeng Gao, Qiankun Ding, Hang Zheng, Hongteng Xu, Zhewei Wei, Linfeng Zhang, and Guolin Ke. Uni-mol: A universal 3D molecular representation learning framework. *ChemRxiv*, March 2023. 8

[50] Hongyang Gao and Shuiwang Ji. Graph U-nets. *arXiv [cs.LG]*, May 2019. 8

[51] Honghua Dong, Jiawei Xu, Yu Yang, Rui Zhao, Shiwen Wu, Chun Yuan, Xiu Li, Chris J Maddison, and Lei Han. MeGraph: Capturing long-range interactions by alternating local and hierarchical aggregation on multi-scaled graph hierarchy. *Neural Inf Process Syst*, 36: 63609–63641, 2023. 8

[52] Lingxiao Zhao and Leman Akoglu. PairNorm: Tackling oversmoothing in GNNs. *arXiv [cs.LG]*, September 2019. 13

A Appendix

A.1 Supplementary Definitions

A.1.1 Edge betweenness centrality

Edge betweenness centrality (EBC) quantifies the importance of an edge in a graph by measuring how frequently it appears in shortest paths between node pairs. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the betweenness centrality of an edge $e_{u,v} \in \mathcal{E}$ is defined as:

$$B(e_{u,v}) = \sum_{s \neq t \in \mathcal{V}} \frac{\sigma_{st}(e_{u,v})}{\sigma_{st}}, \tag{6}$$

where σ_{st} is the total number of shortest paths between nodes s and t, and $\sigma_{st}(e_{u,v})$ is the number of those paths that pass through edge $e_{u,v}$.

A.1.2 Dirichlet Energy.

Over-smoothing is often assessed by measuring node feature similarity, with Dirichlet energy [52] commonly used to quantify variation across neighbors.

$$\mathcal{E}_{Dir}^{l} = \frac{1}{|\mathcal{E}|} \sum_{v \in \mathcal{V}} \sum_{u \in \mathcal{N}(v)} \|\mathbf{h}_v^{(l)} - \mathbf{h}_u^{(l)}\|_2^2. \tag{7}$$

As GNNs go deeper, excessive smoothing causes $\mathcal{E}_{Dir}$ to decay exponentially, leading to node embeddings collapsing into nearly identical values. As discriminative power declines, expressiveness is constrained, creating a key obstacle in GNN development.

A.2 Experimental Details

A.2.1 Model Architectures and Hyperparameters

All experiments were conducted in the following software and hardware environments: Python 3.11, PyTorch 2.5.1, PyTorch Geometric 2.7.0, CUDA 12.4.

Table 3: Hyperparameter settings of EC-Gated GCN for all datasets. Parenthesized values correspond to the settings for EC-Gated GatedGCN, which only differ slightly from GCN; separate tables are omitted for brevity. * "Group Size" refers to the number of disjoint channel groups in EC-Gate, each modulated by an independent scalar gate.

Hyperparameter	PCBA	Lipo	AqSol
# GNN Layers	8	8	8
With Edge Feature	True	True	True
Normalization	BN	BN	BN
Dropout	0.4	0.2	0.2
Residual Connections	True	True	True
With PE	False	False	False
Hidden Dim	600 (420)	180 (120)	120 (100)
Graph Pooling	GlobalAttn	GlobalAttn	GlobalAttn
Gate Groups*	10	10	20
Batch Size	512	32	32
Learning Rate	0.002	0.002	0.002
# Epochs	50	50	50
# Warmup Epochs	5	5	5
Learning Rate scheduler	CosAnnealing	CosAnnealing	CosAnnealing
Weight Decay	0	2e-5	2e-5
# Parameters	7.0M (8.2M)	0.8M (1.1M)	0.3M (0.5M)

The input node features are initialized using the OGB-provided atom and bond encoders. Each layer performs propagation, normalization, activation, and dropout sequentially, with an optional residual

connection. As listed in Table 3, we adopt *Global Attention Pooling* for graph-level readout. Given node embeddings $\{\mathbf{h}_i\}_{i=1}^n$, the pooled representation is computed as:

$$\mathbf{h}_{\text{graph}} = \sum_{i=1}^{n} \alpha_i \, \mathbf{h}_i, \quad \text{where} \quad \alpha_i = \frac{\exp\big(g_\theta(\mathbf{h}_i)\big)}{\sum_{j=1}^{n} \exp\big(g_\theta(\mathbf{h}_j)\big)}, \tag{8}$$

where $g_\theta(\cdot)$ is a learnable gate network implemented as a 3-layer MLP with Batch Normalization.

A.2.2 Synthetic Graphs Construction

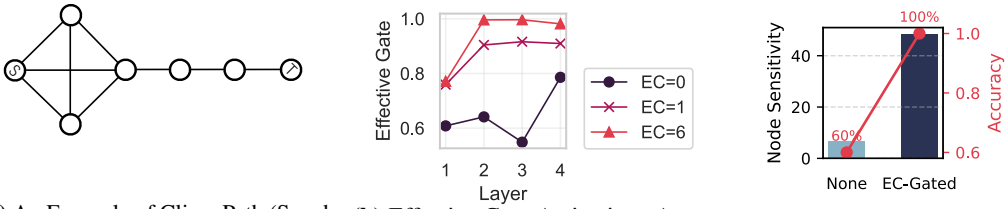
Table 4: Synthetic graphs with the number of nodes $N \in [50, 70]$, generated from diverse random graph models under parameter ranges chosen to ensure moderate sparsity.

Generator	Parameter Range in Code	Structural Characteristics / Role of Parameters	Weight
Watts–Strogatz (WS)	$k \in \{5, \ldots, 15\}$, $\beta \sim \mathcal{U}(0.02, 0.1)$	Small-world graph; k sets node degree, β controls rewiring and shortcut density.	0.15
Stochastic Block Model (SBM)	$B \in \{4, \ldots, 8\}$, $p_{\text{in}} \sim \mathcal{U}(0.4, 0.8)$, $p_{\text{out}} \sim \mathcal{U}(0.02, 0.1)$	Community-structured graph; p_{in} and p_{out} control intra/inter-block density. Block sizes follow a Dirichlet distribution.	0.15
Random Geometric Graph (kNN)	$k \in \{5, \ldots, 15\}$ in 2D unit square	Spatial graph; each node connects to k nearest neighbors, larger k gives denser local regions.	0.15
Barabási–Albert (BA)	$m \in \{1, \ldots, 4\}$	Scale-free network via preferential attachment; m controls edges formed by each new node.	0.15
Power-Law Cluster	$m \in \{1, \ldots, 4\}$, $p_\triangle \sim \mathcal{U}(0.1, 0.4)$	BA model with triadic closure; m sets new edges, $p_\triangle$ adds triangles for higher clustering.	0.15
Erdős–Rényi (ER)	$p \sim \mathcal{U}(0.05, 0.2)$	Random graph; edges appear independently with probability p, controlling density and diameter.	0.25

A.3 Supplementary Experimental Results

A.3.1 Graph Transfer

We provide an small but intuitive case study on a CliquePath transfer task (inspired by [41]), where the GNN must transfer the feature from a distant source node to the target node, thereby relying solely on long-range information to achieve successful prediction. Figure 8b shows how the *effective width*, as defined in Section 4.3, evolves across layers.



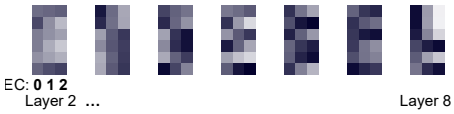
(a) An Example of CliquePath (S and T denote the source and target nodes.)

(b) Effective Gate Activations Across Layers

(c) Comparison of Node Sensitivity and Accuracy

Figure 8: (a) In this graph transfer task, a source node in a clique with a one-hot feature must be recovered by a distant target node; all other nodes hold constant features. (b) Gates exhibit monotonic growth with EC, highlighting selective amplification of high-EC edges that transmit distant information. (c) In this setting, the EC-Gated GCN achieves high accuracy, with a marked increase in node sensitivity.

A.3.2 Activation Patterns



(a) Lipo

(b) AqSol

Figure 9: Layer-wise EC-Gate activation patterns with GCN backbone. As EC changes, the EC-Gate values differ at each layer, serving as evidence of the method's effectiveness. Some gates vary only slightly with EC, which is reasonable since some features' passage is independent of EC.