Cayley Graph Propagation

JJ Wilson JOSEPHJWILSON74@GMAIL.COM
Independent Researcher

Petar Veličković PETARV@GOOGLE.COM
Google DeepMind

Editor: Sophia Sanborn, Christian Shewmake, Simone Azeglio, Nina Miolane

Abstract

In spite of the plethora of success stories with graph neural networks (GNNs) on modelling graph-structured data, they are notoriously vulnerable to tasks which necessitate mixing of information between distant pairs of nodes, especially in the presence of bottlenecks in the graph. For this reason, a significant body of research has dedicated itself to discovering or pre-computing graph structures which ameliorate such bottlenecks. Bottleneck-free graphs are well-known in the mathematical community as *expander graphs*, with prior work—Expander Graph Propagation (EGP)—proposing the use of a well-known expander graph family—the Cayley graphs of the $\mathrm{SL}(2, \mathbb{Z}_n)$ special linear group—as a computational template for GNNs. However, despite its solid theoretical grounding, the actual computational graphs used by EGP are *truncated* Cayley graphs, which causes them to lose expansion properties. In this work, we propose to use the full Cayley graph within EGP, recovering significant improvements on datasets from the Open Graph Benchmark (OGB). Our empirical evidence suggests that the retention of the nodes in the expander graph can provide benefit for graph representation learning, which may provide valuable insight for future models.

Keywords: graph neural networks, graph representation learning, graph machine learning, oversquashing, bottlenecks, expander graphs, cayley graphs

1. Introduction

Graph neural networks (GNNs) depend on propagating information between neighbouring nodes in the graph (Bronstein et al., 2021) where the *message passing* (Gilmer et al., 2017a) paradigm serves as an architecture for facilitating this information. This paradigm involves iterative exchange of messages, with nodes aggregating and updating their representations based on received information from neighbours. However, a phenomenon known as *oversquashing* (Alon and Yahav, 2020) can limit the effectiveness of this message passing process. Oversquashing occurs when a large volume of messages are aggregated into fixed-size vectors, which hinder the *expressive power* of GNNs, especially when dealing with long-range node interactions (Di Giovanni et al., 2023).

The Expander Graph Propagation (EGP) paper (Deac et al., 2022) identified four desirable criteria to mitigate oversquashing and effectively handle global context in graph representation learning: *global information propagation*, *no bottlenecks*, *subquadratic time and space complexity* and *no dedicated preprocessing*. The authors surveyed prior approaches, including traditional GNNs, master-node methods (Gilmer et al., 2017b; Battaglia et al., 2018) and fully connected graphs (Alon and Yahav, 2020), ultimately recognising the efficacy of *expander graphs* for bottleneck-free information propagation.

© 2023 J. Wilson & P. Veličković.

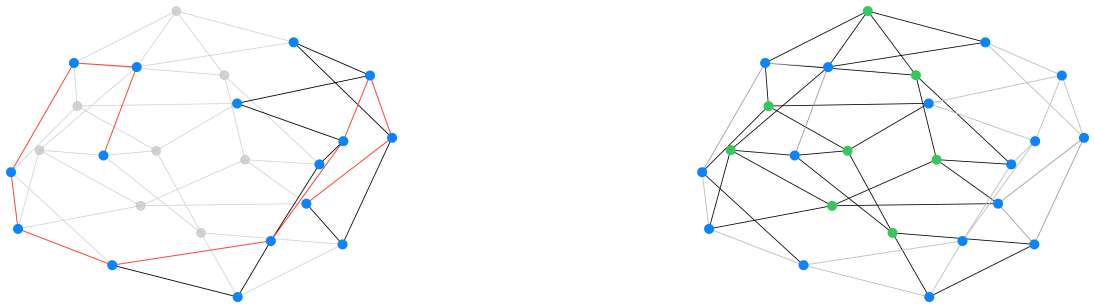

Figure 1: Both Cayley graphs represent $\mathrm{SL}(2, \mathbb{Z}_3)$ with $|V| = 24$ using the same construction. **Left** A truncated Cayley graph (*spectral gap*: 0.0751, *diameter*: 10) aligned to a given input graph. **Right**: The *full* Cayley graph (*spectral gap*: 1.2679, *diameter*: 4) structure indicating the additional *virtual nodes* (in green).

In the EGP paper, a family of expander graphs have been constructed leveraging the well-known theoretical results of *special linear groups*, $\mathrm{SL}(2, \mathbb{Z}_n)$, for which a family of corresponding Cayley graphs, $\mathrm{Cay}(\mathrm{SL}(2, \mathbb{Z}_n); S_n)$, can be derived. Here, S_n (Deac et al. (2022), Definition 8) denotes a particular generating set for $\mathrm{SL}(2, \mathbb{Z}_n)$. For appropriate choices of S_n, the corresponding Cayley graphs are guaranteed to have expansion properties.

Importantly, although Cayley graphs are scalable, achieving a specific number of nodes is not always feasible; for instance, the node count of $\mathrm{Cay}(\mathrm{SL}(2, \mathbb{Z}_n); S_n)$ is known to be in $O(n^3)$. The challenge of determining the right order n of the set $\mathbb{Z}_n$ such that $|V(\mathrm{Cay}(\mathrm{SL}(2, \mathbb{Z}_n); S_n))|$ takes a particular value, $|V|$, has been recognised in the EGP paper. This consideration is addressed within the paper by identifying the smallest n for which $|V(\mathrm{Cay}(\mathrm{SL}(2, \mathbb{Z}_n); S_n))| \geq |V|$, and then *truncating* the Cayley graph to use only the first $|V|$ nodes in breadth-first order. However, while Deac et al. (2022) showcased the utility for this particular construction, there is no guarantee that this is the optimal way to offer a Cayley graph as a computational template for a GNN. In Figure 1 (left) we demonstrate how a truncated Cayley graph of $\mathrm{SL}(2, \mathbb{Z}_n)$ may lose its coveted expansion properties by drastically increasing its diameter and decreasing its spectral gap; see Appendix A for a more detailed analysis. This reintroduces long-range interactions and leaves the GNN over such a graph vulnerable to oversquashing.

In this paper, we instead decide to embrace the *full* Cayley graph, along with any additional nodes it offers which cannot be aligned to the input graph—for which we perform a separate feature initialisation. The full Cayley graph does more than (trivially) preserving beneficial expansion properties: the additional nodes (and pathways connecting them to the original graph) may be interpreted as *virtual nodes* (Pham et al., 2017) which offer increased "coverage" of the graph compared to the truncated version. It is generally well recognised that *virtual nodes* provide shortcuts for message passing between nodes along the graph edges, as supported by the empirical evidence of Hwang et al. (2022). Additionally, this work aligns with the principles of JK networks (Xu et al., 2018b), where the full Cayley

graph is interwoven with the input graph such that varying neighbourhood ranges are employed to facilitate improved structure-aware representations.

We refer to our EGP variant as *Cayley graph propagation* (**CGP**), to emphasise the fact we're using the *full Cayley graph* structure.

2. Cayley Graph Propagation

The setup for CGP closely aligns with that of EGP in most aspects. We continue to consider the input to a GNN as a node feature matrix $\mathbf{X} \in \mathbb{R}^{|V| \times d}$ and an adjacency matrix $\mathbf{A} \in \mathbb{R}^{|V| \times |V|}$, which can be fed in an edge-list manner.

Additionally, the construction of the Cayley graph $\mathrm{Cay}(\mathrm{SL}(2, \mathbb{Z}_n); S_n)$ is still done by choosing the smallest n such that $|V(\mathrm{Cay}(\mathrm{SL}(2, \mathbb{Z}_n); S_n))| \geq |V|$. However, we no longer truncate the Cayley graph such that a subgraph $\mathbf{A}^{\mathrm{Cay}(n)}_{1:|V|,1:|V|}$ is extracted – instead, we opt for a different approach of retaining all of the nodes of the Cayley graph, and its corresponding adjacency matrix $\mathbf{A}^{\mathrm{Cay}(n)}$.

This construction requires us to add new nodes into the graph; hence, we need to modify the feature matrix into an extended version, $\mathbf{X}^{\mathrm{Cay}(n)} \in \mathbb{R}^{|V(\mathrm{Cay}(n))| \times d}$. To construct this, we featurise the first $|V|$ nodes using the data from $\mathbf{X}$, and treat any additional nodes as *virtual nodes*, initialised in some pre-defined way. Specifically:

$$\mathbf{X}^{\mathrm{Cay}(n)}_{1:|V|} = \mathbf{X} \qquad \mathbf{X}^{\mathrm{Cay}(n)}_{|V|+1:|V(\mathrm{Cay}(n)|} \sim \mathrm{InitVirt} \tag{1}$$

where InitVirt is any sampling procedure for initialising d-dimensional feature vectors; for example, we may choose to sample random features from $\mathcal{N}(0, 1)$, or initialise them to zeros. See Table 2 for a comparison of various feature initialisation approaches.

Because EGP makes advantage of both the input graph (specified by $\mathbf{A}$) and the generated Cayley graph (specified by $\mathbf{A}^{\mathrm{Cay}(n)}$), we also need to appropriately extend the original adjacency matrix, $\mathbf{A}$, to incorporate the new nodes. Since the input graph layers are intended to preserve the input graph topology as much as possible, we construct such a matrix $\widetilde{\mathbf{A}} \in \mathbb{R}^{|V(\mathrm{Cay}(n)| \times |V(\mathrm{Cay}(n)|}$ by adding *self-edges* to the virtual nodes only:

$$\widetilde{\mathbf{A}}_{1:|V|,1:|V|} = \mathbf{A} \qquad \widetilde{\mathbf{A}}_{|V|+1:|V(\mathrm{Cay}(n)|,|V|+1:|V(\mathrm{Cay}(n)|} = \mathbf{I} \tag{2}$$

$$\widetilde{\mathbf{A}}_{1:|V|,|V|+1:|V(\mathrm{Cay}(n)|} = \widetilde{\mathbf{A}}_{|V|+1:|V(\mathrm{Cay}(n)|,1:|V|} = \mathbf{0} \tag{3}$$

CGP now proceeds in the same manner as EGP: alternating GNN layers, such that every odd layer operates over the input graph—to preserve the topological information therein—and every even layer operates over the generated Cayley graph—to support bottleneck-free global communication. For a two-layer CGP model, this can be depicted as:

$$\mathbf{H} = \mathrm{GNN}(\mathrm{GNN}(\mathbf{X}^{\mathrm{Cay}(n)}, \widetilde{\mathbf{A}}; \theta_1), \mathbf{A}^{\mathrm{Cay}(n)}; \theta_2) \tag{4}$$

where θ_1 and θ_2 are the parameters of the first and second GNN layer, respectively. This implementation works with any choice of base GNN; here we make advantage of the graph isomorphism network (Xu et al. (2018a), GIN):

$$\boldsymbol{h}_u = \phi\left((1+\epsilon)\,\boldsymbol{x}_u + \sum_{v \in \mathcal{N}_u} \boldsymbol{x}_v\right) \tag{5}$$

Table 1: Comparative performance evaluation on the two studied datasets.

Model	`ogbg-molhiv`	`ogbg-molpcba`
GIN	0.7542 ± 0.0084	0.2250 ± 0.0031
GIN + EGP	0.7681 ± 0.0128	0.2305 ± 0.0023
GIN + CGP	$\mathbf{0.7903} \pm 0.0172$	$\mathbf{0.2333} \pm 0.0016$

where $\boldsymbol{x}_u \in \mathbb{R}^d$ are the features of node u, ϵ is a learnable scalar, and ϕ is a MLP.

The final node embeddings $\mathbf{H} \in \mathbb{R}^{|V(\mathrm{Cay}(n)|\times k}$ may then be used for downstream node, graph or graph-level tasks. To avoid direct influence of virtual nodes in these predictions, we use only the embeddings corresponding to the original graph's nodes, that is, $\mathbf{H}_{1:|V|}$, in downstream tasks.

The CGP model upholds the requirements of the four criteria set by Deac et al. (2022)—arguably, in a more theoretically grounded way than EGP; Figure 1 and Appendix A provide empirical evidence of this. Specifically, the lower diameter of the graph used in CGP enhances its ability to eliminate oversquashing and bottlenecks, which is further supported by having a higher spectral gap. Furthermore, the CGP model may be able to make up for one of the limitations of the Cayley graph construction: the inability to find the best way to align it to a given input graph, mitigating the potential for *stochastic* effects in the process. The additional virtual nodes act as "bridges" between poorly connected communities in the Cayley graph, ameliorating any poorly-connected regions caused by misalignment.

3. Empirical Evidence

Our work extends the foundation of the EGP model, advocating the claim that a *full Cayley graph* structure better alleviates oversquashing and bottlenecks with empirical evidence.

OGB Datasets The `ogbg-molhiv` and `ogbg-molpcba` were chosen to conduct experimentation, which provides emulation for real-world analysis. They are among the largest molecule property prediction datasets within the scope of the MoleculeNet benchmark (Wu et al., 2018), and also suitable for our hardware. Significantly, the `ogbg-molhiv` yielded the most improvements for the EGP model compared to the baseline GIN (Xu et al., 2018a).

Models In line with the EGP model it uses the baseline model of GIN (Xu et al., 2018a), using the open-source implementation of OGB (Hu et al., 2020) and their given hyperparameters. The only modification to the model is using the *full Cayley graph* as part of the architecture and how to initialise the virtual nodes. For CGP, *zero* initialisation yields the best performance. For fair comparison, the models have been trained using a NVIDIA V100 with 16 GB of memory where EGP and CGP achieved training speeds akin to each other. In Appendix B we support the effectiveness of using the *full Cayley graph* structure by evaluating other node initialisation strategies.

Results The evaluation results are showcased in Table 1. It is evident that propagating over a *full Cayley graph* structure can lead to enhanced performance compared to the EGP model. EGP is empirically sensitive to the choice of alignment and truncation of Cayley graph, and for our choice of alignments, the performance on both `ogbg-molhiv` and `ogbg-molpcba` is lower than what is reported in Deac et al. (2022). Conversely CGP does

not suffer from such issues, this is because it does not truncate the graph; see Appendix C where we support this claim by exploring different alignment strategies. The results of the datasets provide empirical evidence that leveraging the *full Cayley graph* effectively preserves their expansion properties, thereby improving the alleviation of bottlenecks.

Acknowledgments

We would like to thank Petar Veličković's co-authors of the Expander Graph Propagation paper, including Andreea Deac and Mark Lackenby. Their original work of leveraging the use of expander graphs within the context of graph neural networks to alleviate bottlenecks and counter oversquashing provided the basis for our work. Furthermore, we would like to thank Alex Vitvitskyi and Csaba Szepesvári for their valuable insights prior to submission, as well as the anonymous reviewers for their helpful feedback.

Many thanks to the developers of Open Graph Benchmark and their community, whose open-source implementations provided a strong foundation to build this work upon and carry out experiments.

References

Ralph Abboud, Ismail Ilkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization. *arXiv preprint arXiv:2010.01179*, 2020.

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

Andreea Deac, Marc Lackenby, and Petar Veličković. Expander graph propagation. In *Learning on Graphs Conference*, pages 38–1. PMLR, 2022.

Francesco Di Giovanni, T Konstantin Rusch, Michael M Bronstein, Andreea Deac, Marc Lackenby, Siddhartha Mishra, and Petar Veličković. How does over-squashing affect the power of gnns? *arXiv preprint arXiv:2306.03589*, 2023.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. *CoRR*, abs/1704.01212, 2017a. URL http://arxiv.org/abs/1704.01212.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017b.

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.

EunJeong Hwang, Veronika Thost, Shib Sankar Dasgupta, and Tengfei Ma. An analysis of virtual nodes in graph neural networks for link prediction (extended abstract). In *The First Learning on Graphs Conference*, 2022. URL https://openreview.net/forum?id=dI6KBKNRp7.

Trang Pham, Truyen Tran, Hoa Dam, and Svetha Venkatesh. Graph classification via deep learning with virtual nodes. *arXiv preprint arXiv:1708.04357*, 2017.

Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018a.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, 2018b.

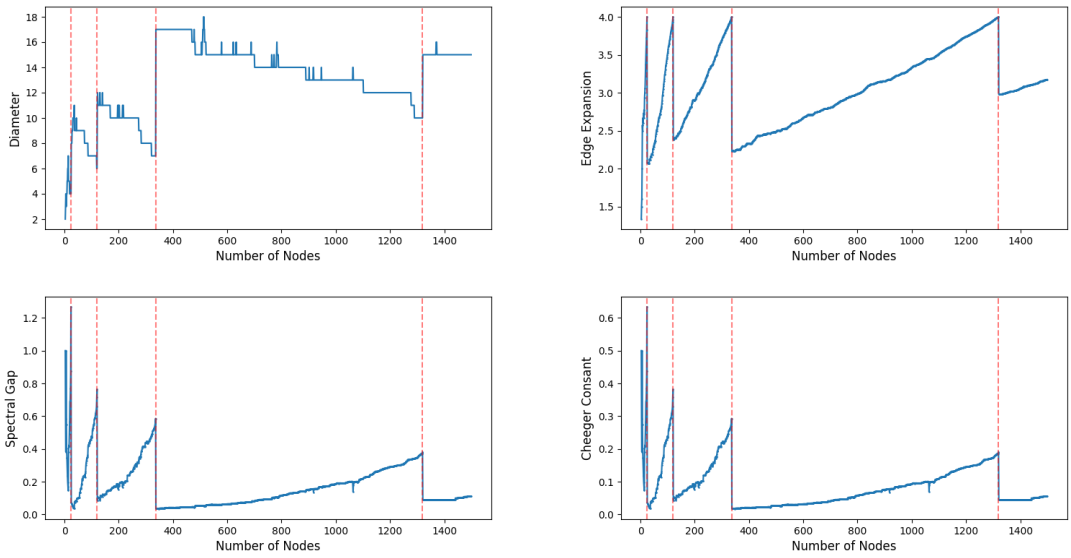

Figure 2: Illustrates how truncating the Cayley graph structures influences expansion properties, with the full Cayley graph node intervals indicated by the dotted red-line.

Appendix A. Expansion properties of truncated Cayley graphs

Figure 2 show the relationship between truncating a Cayley graph and its desired expansion properties. We provide analysis of the first four graphs from the family of Cayley graphs, $\mathrm{Cay}(\mathrm{SL}(2,\mathbb{Z}_n);S_n)$. Notably, the number of nodes is determined by the following expression:

$$|V(\mathrm{Cay}(\mathrm{SL}(2,\mathbb{Z}_n);S_n))| = n^3 \prod_{\text{prime } p|n} \left(1-\frac{1}{p^2}\right) \tag{6}$$

To preserve the EGP (Deac et al., 2022) construction strategy for an input graph's alignment to a Cayley graph where only the first $|V|$ nodes are used in breadth-first order; we truncate in a way so as not to disconnect the graph. For all given graphs you can see a trend where the more desirable properties are at the given intervals of a *full Cayley graph* number of nodes.

Diameter The diameter of a graph influences the effectiveness of traversal between nodes. A lower diameter facilitates a more efficient graph structure, enabling nodes to reach each other in a shorter number of hops.

Edge Expansion The edge expansion quantifies a graph's connectivity by evaluating the ratio of the edge count to the number of nodes, therefore a high edge expansion suggests a well-connected graph.

Spectral Gap The spectral gap, a fundamental concept in graph theory, provides valuable insight into the graph's connectivity and expansion properties. A desirable high spectral gap signifies strong connectivity and expansion.

Cheeger Constant The Cheeger constant offers measurement of the narrowest bottleneck in a graph; a higher Cheeger constant provides evidence that a graph is globally lacking bottlenecks. The exact Cheeger constant $h(G)$ is known to be a computationally challenging problem. To address this difficulty, we use the Cheeger inequality as per Chung (1997), given by:

$$\frac{\lambda}{2} \leq h(G) \leq \sqrt{2\lambda} \tag{7}$$

The inequality establishes a relationship with respect to λ, defining a range for the Cheeger constant. Given the computational complexity of determining the precise Cheeger constant, we utilise the lower bound to visualise this information on the graph.

The analysis of truncating a Cayley graph to align with a given input graph supports our claim that it adversely affects the deseriable expansion properties. It is observed that the most unfavourable scenario occurs just beyond the range of the proceeding the interval of a Cayley graph number of nodes.

Appendix B. Virtual node initialisation strategies

The results in Table 2 further support the claim that leveraging a *full Cayley Graph* is highly beneficial, influencing both local interactions within their neighbourhood and the global context of the graph. The pre-defined strategies have all performed better than EGP (for the provided dataset `ogbg-molhiv`).

Mean The mean initialisation strategy entails setting all *virtual nodes* to the average derived from the collective node features from the input graph.

Random A random node initialisation (RNI) strategy was chosen, due to research in (Abboud et al., 2020), which provides evidence that supports expressive power of GNNs with RNI.

Zeros The initialisation of the *virtual nodes* to zeros. Interestingly, this strategy yields the best mean performance out of all the tested approaches. However, this is not to state this is the best node initialisation strategy for CGP. Rather, it serves as a solid foundation for more refined techniques in the future.

Appendix C. Cayley graph alignment sensitivity

The analysis conducted in Table 3 provides empirical evidence to further support our claim—CGP is more resilient compared to EGP for the alignment between the input graph and the expander graph. For each result, the values fall within the range of one standard deviation as observed in the base CGP model. Figure 1 provides an example of a truncated Cayley graph that has poor alignment with the input graph. We examine the sensitivity between the input graph and Cayley graph by implementing a strategy whereby we explore different alignments. The Cayley graph being constructed in a breadth-first manner means we can align the input graph in an alternative approach by offsetting the starting node.

In practice, this simple approach results in different edges being aligned with the Cayley graph, therefore we have to handle the additional nodes accordingly. In accordance with Cayley graph propagation (CGP) they are still handled as *virtual nodes* and initialised in some pre-defined way, thereby alleviating the sensitivity of aligning to the input graph.

Table 2: Comparative analysis of different strategies to initialise the virtual nodes for full Cayley graph. The baseline models are GIN (Xu et al., 2018a) and EGP (Deac et al., 2022) both of which are using a similar implementation of OGB (Hu et al., 2020). To establish a fair basis, the results have been trained using a NVIDIA V100 with 16 GB of memory and same number of parameters, which resulted in a training time within five hours for both the EGP and CGP models.

Model	`ogbg-molhiv`
GIN	0.7542 ± 0.0084
GIN + EGP	0.7681 ± 0.0128
GIN + CGP_{mean}	0.7806 ± 0.0123
GIN + $\text{CGP}_{\text{random}}$	0.7728 ± 0.0160
GIN + $\text{CGP}_{\text{zeros}}$	$\mathbf{0.7903} \pm 0.0172$

Table 3: Comparative performance evaluation between the baseline CGP model and different Cayley graph alignment strategies—all of which are still using a similar implementation of OGB (Hu et al., 2020). The offset method used as per Appendix C is indicated by the subscript number, i.e. for CGP_1 this is offset by 1 compared to the baseline CGP model.

Model	`ogbg-molhiv`
GIN + CGP	$\mathbf{0.7903} \pm 0.0172$
GIN + CGP_1	0.7854 ± 0.0183
GIN + CGP_3	0.7737 ± 0.0129
GIN + CGP_5	0.7797 ± 0.0068
GIN + CGP_7	0.7888 ± 0.0094