How do positional encodings generalise to edges?

1090505

Abstract

Recently, graph neural networks have augmented node features with structural and positional encodings to improve generalisation, convergence and even expressive power. Edges, however, have not received this treatment and edge features are often ignored. This miniproject generalises positional encodings to edges, derives an implementation, provides a theoretical discussion and concludes with an empirical investigation. I make three key contributions. Firstly, I propose a generalisation of positional encodings to edges, by concatenating the Laplacian eigenvectors of equivalent graphs onto edge features. Secondly, I provide a theoretical treatment, which demonstrates that these edge positional encodings increase the expressive power of some networks. Thirdly, I empirically investigate the effect of edge positional encodings on convergence and generalisation, and find evidence that they may act as a useful inductive bias.

1 Introduction

Transformers motivated positional encodings Following the success of transformers in other domains, such as text [Vaswani et al., 2017] or vision [Dosovitskiy et al., 2021], there was great interest in generalising the transformer architecture to graphs. This involved generalising positional encodings to graphs. The most popular graph positional encodings augment node features with the eigenvectors of the graph Laplacian [Dwivedi and Bresson, 2021; Kreuzer et al., 2021]. After the success of graph positional encodings on graph transformers, it was shown that positional encodings can also improve the expressive power and generalisation of message-passing neural networks [Dwivedi et al., 2023], which led to widespread adoption. Despite the success of positional encodings for nodes, there has been no investigation into positional encodings for edges. This opens an interesting research question: *How do positional encodings generalise to edges? Do they have any effect on the convergence, generalisation or expressivity of networks?*

Augmenting edges with positional encodings has two primary advantages:

Imposing a useful inductive bias Many popular architectures, such as GATv2 [Brody et al., 2022], use edge features only to influence the weight with which they attend to each neighbour. Placing positional information in edge features imposes an inductive bias that nodes should use positional information to determine how much to attend to their neighbours. Whilst positional information can be represented in node features, node features are universal and agnostic of the relationship between adjacent nodes. As such, encoding positional information into edges (potentially in addition to nodes) could act as a useful inductive bias which encourages more dynamic attention and may lead to better generalisability and faster convergence in some settings.

Increasing expressivity Some architectures use edge features to influence the value of nodes [Gong and Cheng, 2019; Kim et al., 2019]. Using edge positional encodings in addition to node encodings increases the expressive power of such architectures, allowing them to distinguish some isospectral graphs. Other architectures, such as ESA [Buterez et al., 2024] can achieve impressive performance

without any positional encodings by attending to edge representations: for a graph $G = (V, E)$, interleave GAT layers and full attention layers on $\{\mathbf{x}_u \parallel \mathbf{x}_v \parallel \mathbf{e}_{uv} \mid (u, v) \in V\}$. ESA becomes significantly more theoretically expressive by incorporating edge positional encodings.

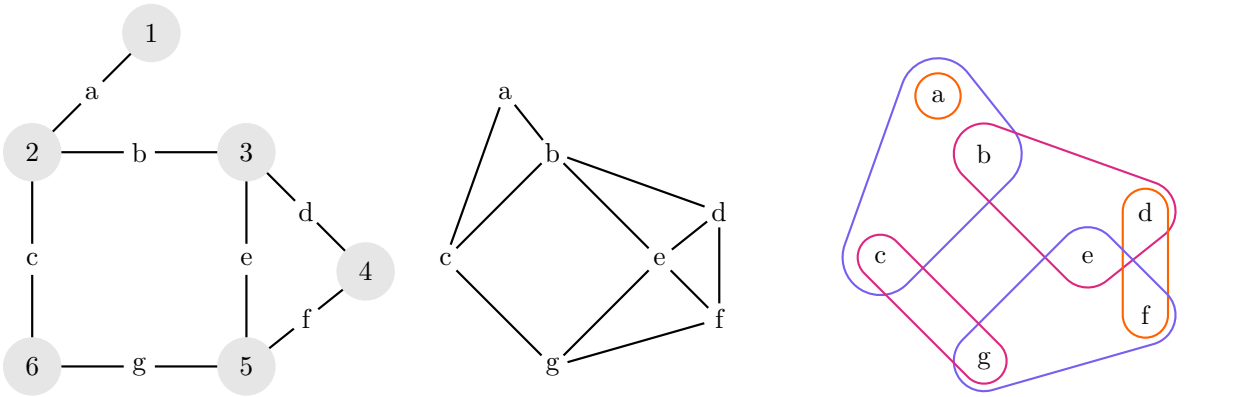
2 Method

This section introduces, and provides mathematical justification for, a method of generalising positional encodings to edges. I motivate necessary conditions to generalise Laplacian positional encodings to edges, and show that the adjoint graph and dual hypergraph satisfy these conditions.

Positional encodings use the graph Laplacian Most state-of-the-art positional encodings are learnable functions of the eigenvectors of the graph Laplacian. Given that node encodings based on the graph Laplacian have been so successful, I ask whether there is an analagous Laplacian for edges which we could use to construct positional encodings. We require a Laplacian which is easy to compute, has real-valued eigenvectors and real-valued eigenvalues. The simplest method to ensure a Laplacian satisfies these conditions is for it to be the graph Laplacian of some graph whose nodes correspond to the edges in the original graph. These graphs are visualised in Figure 1.

The adjoint graph Given a graph $G = (V, E)$, the adjoint graph is defined as the graph $L(G) = (E, \{((u, v), (v, w)) \mid (u, v), (v, w) \in E\})$, *i.e.* the graph formed by taking the edges of G to be nodes, and connecting edges if, and only if, they are adjacent in G. The structure of all connected components can be recovered from $L(G)$ [Whitney, 1992], meaning that the adjoint graph contains all structural information about G. This means that its Laplacian contains all structural information about G, and so its Laplacian eigenvectors are a candidate edge positional encoding. The Laplacian of the adjoint graph is known as the Edge Laplacian [Chauhan and Reddy, 2023].

The dual hypergraph of G Given an undirected graph $G = (V, E)$, it's dual hypergraph is defined as $D(G) = (E, V)$, *i.e.* the hypergraph $D(G)$ where all vertices correspond to edges in G; and all edges correspond to vertices in G. D is perfectly invertible, so $D(G)$ contains all structural information about G. The dual hypergraph is simple and finite, meaning its Laplacian can easily be computed using the equation given by Rodriguez-Velazquez.



(a) Original graph, $G = (V, E)$ **(b)** Adjoint graph, $L(G)$ **(c)** Dual hypergraph, $D(G) = (E, V)$

Figure 1: Comparison of the original graph G to its adjoint graph $L(G)$ and dual hypergraph $D(G)$. I construct positional edge encodings from the eigenvectors of the Laplacians of $L(G)$ and $D(G)$. Notice how both $L(G)$ and $D(G)$ retain structural information about G. This means that encoding structural information about $L(G)$ or $D(G)$ into edges will also encode structural information about G.

3 Results

I begin with a theoretical investigation into how using edge positional encodings affects the expressivity of networks. This finds that edge positional encodings can increase expressivity, even when used in conjunction with node positional encodings. The section concludes with an empirical investigation which indicates that edge positional encodings may improve generalisation.

3.1 Theoretical results

For a positional encoding to affect the expressivity of an architecture, it must encode information which is not freely accessible to the architecture. Since the expressivity of many GNN architectures is upper-bounded by the distinguishability of the 1-WL graph isomorphism test [Xu et al., 2019], I demonstrate that edge positional encodings can be used to differentiate between many 1-WL indistinguishable graphs in Figure 2. This shows that edge positional encodings can increase the expressivity of networks which do not use node positional encodings.

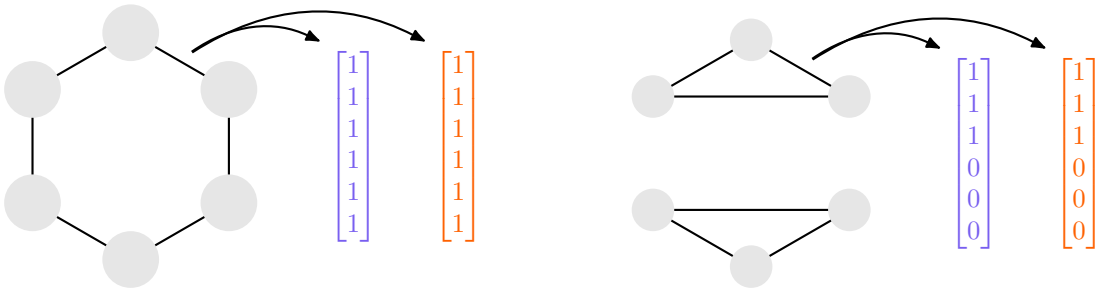


Figure 2: Two 1-WL indistinguishable graphs which have different positional encodings for a particular edge. This demonstrates that edge positional encodings can increase the expressivity of graph neural networks. The positional encodings computed from the adjoint are blue, while the positional encodings computed from the dual hypergraph are orange. The adjoint and dual positional encodings coincide in this example since all nodes have the same degree: they do not coincide in general. The least pair of 1-WL indistinguishable graphs for which the adjoint graph Laplacian eigenvectors and dual hypergraph Laplacian eigenvectors do not coincide has 12-nodes.

Edge encodings can distinguish graphs which node encodings cannot Node positional encodings using Laplacian eigenvectors allow networks to differentiate more graphs and thus increase their expressive power. Isospectral pairs, however, are pairs of graphs G_1 and G_2 whose graph Laplacians have the same multiset of eigenvectors. Isospectral pairs cannot be differentiated by Laplacian node positional encodings. I investigate isospectral pairs and find that if two graphs G_1 and G_2 are isospectral, then their adjoint graphs and dual hypergraphs are not necessarily isospectral. This means there are graphs which Laplacian node positional encodings cannot differentiate, but Laplacian edge positional encodings can differentiate. I provide an example of an isospectral pair of graphs which are distinguishable by edge positional encodings in Figure 3.

Edge encodings using the adjoint graph cannot distinguish graphs with isospectral adjoints Edge positional encodings which use the Laplacian of the adjoint graph are unable to distinguish graphs with isospectral adjoints. By enumeration, I find that there is no pair of isospectral graphs with fewer than 8 nodes such that both graphs are adjoints: this means that there are no pairs of graphs with fewer than 8 edges each whose adjoints are isospectral. Therefore, edge positional encodings using the adjoint graph can be used to differentiate every graph with fewer than 8 edges, indicating that edge positional encodings have high discriminative power.

Figure 3: An example isospectral pair: each graph has the same multiset of eigenvectors, so cannot be differentiated by conventional Laplacian node positional encodings. These graphs can be differentiated by edge positional encodings since their adjoint graphs and dual hypergraphs are not isospectral. Example isospectral pair taken from [Pistol, 2024].

3.2 Empirical results

To investigate whether edge positional encodings may be of practical use, I perform an empirical evaluation on the CSL and ZINC datasets [Murphy et al., 2019; Irwin and Shoichet, 2005], and determine what the effect of edge positional encodings is on expressivity, rate of convergence and generalisation. The CSL dataset is designed to test the ability of graph neural networks to distinguish 1-WL indistinguishable graphs. ZINC is a standard graph dataset: if edge encodings are performant, this would indicate that they may be of use in real-world problems.

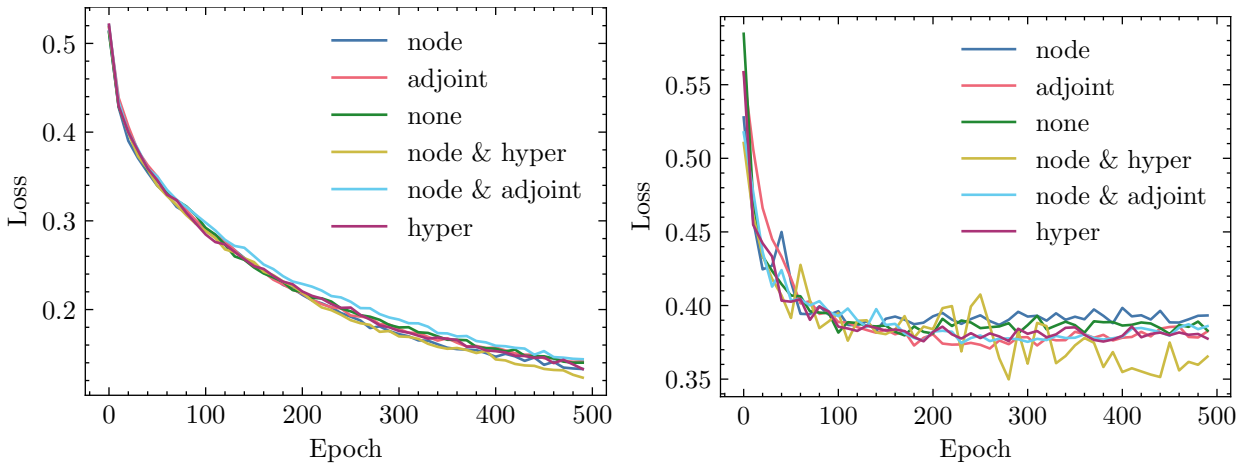
I use a modified version of the experimental setup and parameters used by Dwivedi et al., which was originally used to compare node positional encodings. I use the GATv2 architecture [Brody et al., 2022] since it natively allows edge features to influence attention coefficients. Since GATv2 only uses edge positional encodings to influence attention weights, I do not expect edge encodings to increase expressivity on CSL.

Edge encodings show promise Table 1 shows the performance of GATv2 on the CSL and ZINC test datasets. Each model is trained under the same setting: same model, same epochs, same data. The only difference is the type of positional encodings which were used. We see in Table 1 that edge encodings had strong performance on the ZINC dataset, even performing superior to node encodings. This provides weak evidence for my hypothesis, indicating that edge positional encodings may pose a useful inductive bias, even in the presence of node positional encodings. This experiment, however, only has one run of one model on one task.

	Positional Encoding					
Dataset	None	Node	Adjoint	Hyper	Node & Adjoint	Node & Hyper
CSL (↑)	0.1 ± 0	1 ± 0	0.1 ± 0	0.1 ± 0	1 ± 0	1 ± 0
ZINC (↓)	0.407	0.405	**0.395**	0.402	0.406	0.409

Table 1: A comparison of the performance of a 4-layer GATv2 model when trained on the CSL and ZINC datasets with different positional encodings. Performance on CSL is accuracy; performance on ZINC is ℓ_1-distance. This provides weak evidence that edge positional encodings may impose a useful inductive bias, thus improving generalisation. GATv2 only uses edge features to affect attention coefficients, in CSL without node encodings all values are the same, so attention coefficients do not affect the outcome. Uncertainties for CSL are calculated using 5-fold cross-validation.

The training curves for GATv2 on ZINC are shown in Figure 4. Notably, all models had similar training behaviour, and converged to similar solutions. This suggests that positional encodings are unlikely to have a significant effect on the rate of convergence.



(a) Training loss **(b)** Validation loss

Figure 4: A comparison of the training and validation curves of GATv2 on the ZINC dataset. Edge positional encodings seem to improve the generalisation of GATv2, indicating that edge positional encodings may be viable. Edge positional encodings, however, do not appear to significantly affect the rate of convergence.

4 Conclusions

Section 3 indicates that edge positional encodings may be worth further investigation, and may even be viable. These experiments, however, were carried out on one task; without confidence invervals; and the differences in performance were marginal. Drawing strong conclusions would require experiments with confidence intervals across a wide range of datasets and models.

4.1 Advantages

Increased expressivity Edge positional encodings can increase the expressivity of networks, even when they already use node positional encodings. Edge positional encodings allow graph neural networks to distinguish many 1-WL indistinguishable graphs; and even many isospectral graphs.

Inductive bias Edge positional encodings impose an inductive bias on many models, encouraging them to attend to positional information, possibly leading to better generalisation on some tasks.

4.2 Limitations

Compute cost Positional encodings, especially those based on Laplacian eigenvectors, require significant compute during the pre-processing stage [Buterez et al., 2024]. Almost every graph which we are interested in learning on will have more edges than nodes, meaning that edge positional encodings require even more compute than node positional encodings. This could be partially mitigated by using computationally cheaper edge positional encodings, *e.g.* based on random walks.

Lack of support for edge features Many graph neural networks do not support edge features, limiting possible adoption of edge positional encodings. Many networks which do use edge features, use them only to adjust attention coefficients; and not to directly influence node features. Only allowing edge positional encodings to affect weighted sums does not increase expressivity.

The code for the project is available at `https://github.com/Hjel2/edge-PE`.

References

S. Brody, U. Alon, and E. Yahav. How Attentive are Graph Attention Networks? In *ICLR*, 2022. URL https://openreview.net/forum?id=F72ximsx7C1.

D. Buterez, J. P. Janet, D. Oglic, and P. Lio. An end-to-end attention-based approach for learning on graphs, 2024. URL https://arxiv.org/abs/2402.10793.

S. Chauhan and A. S. Reddy. Spectral Properties of Edge Laplacian Matrix, 2023. URL https://arxiv.org/abs/2309.15841.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

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

V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson. Benchmarking Graph Neural Networks. *JMLR*, 2023. URL http://jmlr.org/papers/v24/22-0567.html.

L. Gong and Q. Cheng. Exploiting Edge Features for Graph Neural Networks. In *CVPR*, 2019. URL https://ieeexplore.ieee.org/document/8954414.

J. Irwin and B. Shoichet. ZINC - A Free Database of Commercially Available Compounds for Virtual Screening. *Journal of chemical information and modeling*, 2005. URL https://pubmed.ncbi.nlm.nih.gov/15667143/.

J. Kim, T. Kim, S. Kim, and C. D. Yoo. Edge-Labeling Graph Neural Network for Few-Shot Learning. In *CVPR*, 2019. URL https://ieeexplore.ieee.org/document/8954106.

D. Kreuzer, D. Beaini, W. L. Hamilton, V. Létourneau, and P. Tossou. Rethinking Graph Transformers with Spectral Attention. In *NeurIPS*, 2021. URL https://openreview.net/forum?id=huAdB-Tj4yG.

R. Murphy, B. Srinivasan, V. Rao, and B. Ribeiro. Relational Pooling for Graph Representations. In *ICML*, 2019. URL https://proceedings.mlr.press/v97/murphy19a.html.

M.-E. Pistol. Generating Isospectral but not Isomorphic Quantum Graphs, 2024. URL https://arxiv.org/abs/2104.12885.

J. A. Rodriguez-Velazquez. On the Laplacian Eigenvalues and Metric Parameters of Hypergraphs. *Linear and Multilinear Algebra*, 50:1–14, 2002.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is All you Need. In *NeurIPS*, 2017. URL https://papers.nips.cc/paper˙files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

H. Whitney. *Congruent Graphs and the Connectivity of Graphs*, pages 61–79. Birkhäuser Boston, Boston, MA, 1992. ISBN 978-1-4612-2972-8. doi: 10.1007/978-1-4612-2972-8_4. URL https://doi.org/10.1007/978-1-4612-2972-8˙4.

K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How Powerful are Graph Neural Networks? In *ICLR*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.