Exploration of Positional Encodings in the LSPE Framework

Ashley Leung
ho.k.leung@mail.mcgill.ca
School of Computer Science, McGill University

Michael Montemurri
michael.montemurri@mcgill.ca
Mila - Quebec AI Institute
Department of Mathematics and Statistics, McGill University

Gabriel Woloz
gabriel.woloz@mail.mcgill.ca
School of Computer Science, McGill University

Abstract

Positional encodings (PEs) play a critical role in enhancing the performance of graph neural networks (GNNs) and transformer-based architectures on molecular property prediction tasks. Building on the Learnable Structural and Positional Encoding (LSPE) framework proposed by Dwivedi et al., this work investigates the impact of alternative initialization strategies for node positional encodings. While the original LSPE framework focuses on Laplacian and random walk-based embeddings, we explore four additional methods: PageRank-based distance encodings, random anchor-based encodings, structure-preserving embeddings (SPE), and centrality-based measures. We evaluate these encodings across multiple GNN and transformer models on three benchmark datasets: ZINC, OGBG-MOLTOX21, and OGBG-MOLPCBA. Our findings indicate that, although most alternatives did not surpass the baseline Random Walk Positional encoding (RWPE), centrality-based encodings—particularly closeness centrality—showed promising improvements. Conversely, approaches such as anchor-based and SPE initializations suffered from either overfitting or scalability limitations. These results suggest that carefully aligned positional signals can improve model performance and highlight the potential for further research into the design and learning of effective PEs within the LSPE framework.

CCS Concepts

• **Computing methodologies → Machine learning approaches**.

Keywords

Graph neural networks, positional encodings, molecular property prediction, LSPE framework

1 Introduction and Motivation

Graph Neural Networks (GNNs) have become the dominant learning paradigm for graph-structured data. In particular, GNNs have shown strong results in molecular property prediction tasks, where atoms are represented as nodes and bonds as edges [15]. A widely used GNN variant, the Message Passing Neural Network (MPNN), learns node embeddings through the iterative aggregation of local neighbourhood information [7]. This local aggregation means that standard GNNs are limited in their ability to capture global structure [17]. This is particularly problematic in molecular graphs, where two atoms can have identical local structure but play very different roles depending on their global position in the molecule.

Graph transformer approaches such as Graphormers and GPS++ have been proposed to alleviate the shortcomings of MPNNs [11, 16].

Code available at https://github.com/gabesw/gnn-lspe.

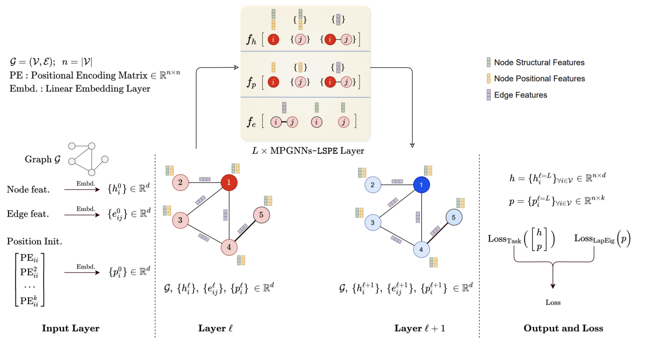
$\mathcal{G} = (\mathcal{V}, \mathcal{E});\ n = |\mathcal{V}|$
PE : Positional Encoding Matrix $\in \mathbb{R}^{n \times n}$
Embd. : Linear Embedding Layer
f_h
f_p
f_e
$L \times$ MPGNNs-LSPE Layer
Node Structural Features
Node Positional Features
Edge Features
Graph $\mathcal{G}$
Node feat.
Embd.
$\{h_i^0\} \in \mathbb{R}^d$
Edge feat.
$\{e_{ij}^0\} \in \mathbb{R}^d$
Position Init.
PE_{ii}, PE_{ii}^2, …, PE_{ii}^k
$\{p_i^0\} \in \mathbb{R}^d$
$\mathcal{G}, \{h_i^\ell\}, \{e_{ij}^\ell\}, \{p_i^\ell\} \in \mathbb{R}^d$
$\mathcal{G}, \{h_i^{\ell+1}\}, \{e_{ij}^{\ell+1}\}, \{p_i^{\ell+1}\} \in \mathbb{R}^d$
$h = \{h_i^{\ell=L}\}_{\forall i \in \mathcal{V}} \in \mathbb{R}^{n \times d}$
$p = \{p_i^{\ell=L}\}_{\forall i \in \mathcal{V}} \in \mathbb{R}^{n \times k}$
$\mathrm{Loss}_{\mathrm{Task}}\left(\begin{bmatrix} h \\ p \end{bmatrix}\right)$
$\mathrm{Loss}_{\mathrm{LapEig}}(p)$
Loss
Input Layer
Layer ℓ
Layer $\ell + 1$
Output and Loss

Figure 1: Diagram of MPGNNs-LSPE architecture as defined in Dwivedi et al. [6]

They introduce positional encodings (PEs), such as centrality encoding, distance encoding, and more, to pass on structural information and provide nodes with information about their position in the graph. These architectures have demonstrated that incorporating structural priors into their attention mechanisms yields promising results for molecular property prediction.

Dwivedi et al. [6] propose the Learnable Structural and Positional Encodings (LSPE) framework, illustrated in Figure 1, which is a general framework that decouples the structural and positional embeddings of a node and learns both iteratively through training. The positional embedding represents the node's position in the graph and the structural embedding represents the node features and local neighbourhood information. Their approach initializes positional embeddings using either Laplacian eigenvectors (LapPE) or Random Walk Positional embeddings (RWPE), then iteratively refines them layer by layer according to a specified loss. They emphasize the importance of the initial PEs, but they leave many potential candidates unexplored. We describe the LSPE framework in more depth in Section 4.1.

Our Approach. In this project, we investigate whether alternative positional encoding initialization strategies can yield empirical improvements within the LSPE framework for molecular property prediction. Specifically, we explore four methods that were not tested by the original authors that we hypothesize may improve performance: (1) PageRank-based encodings, described by Li et al. [9], (2) Random anchor-based encodings [17], (3) Structure Preserving Embeddings (SPE) [14], and (4) Centrality-based Encodings. We describe each of these in more detail in Section 3.

2 Related Work

Graph Transformers and Positional Bias. Transformer-based graph models such as Graphormer [16], GPS [13], and its molecule-optimized variant GPS++ [11], incorporate structural priors like shortest-path distances and centrality measures as attention biases to capture global structure. These models have achieved state-of-the-art performance on molecular property benchmarks, highlighting the importance of positional awareness in graph learning.

Distance Encoding. GNN architectures that rely only on local message passing are theoretically bounded in expressivity by the 1-hop Weisfeiler Leman graph isomorphism (1-WL) test. We refer to such GNNs as WLGNNs. [9, Li et al.] address these shortfalls by integrating a Distance Encoding (DE) into the GNN architectures in two ways: (1) by using a DE as an additional node feature and (2) by using a DE to control message aggregation of WLGNNs (DEA-GNNs). The DE of a node can be implemented via multiple measures including, but not limited to, shortest-path distance (SPD) and generalized PageRank scores. Their results show DEA-GNNs outperform GNNs without DE by up to 15% in accuracy and greatly outperform other top models specifically designed for their tested tasks [9].

Spectral and Diffusion-Based Positional Encodings. Laplacian Eigenvector Positional Encodings (LapPE), based on the spectral decomposition of the graph Laplacian [1] have been widely used as PEs in GNNs and graph transformers [5], though they suffer from sign and basis ambiguity as the Laplacian eigenvectors are defined only up to a sign flip (± 1). Random Walk Positional Encodings (RWPE), introduced by Li et al. [9] and further developed by Dwivedi et al. [6], offer an alternative by encoding a diffusion-based notion of proximity for each node. This encoding method yielded the best results within the LSPE framework and will serve as the baseline initial positional encoding in our work. We provide further details in Section 4.2.

Learnable and Invariant Positional Encodings. The LSPE framework [6] moves beyond traditional static encodings by *learning* positional features throughout model training. This proposed framework decouples the structural and positional embeddings and updates both iteratively, using RWPE or LapPE as initializations. As this framework serves as the basis for our work, we expand on the details in Section 4.1. Position-aware GNNs (P-GNN) [17] use distances to randomly sampled anchor sets as position indicators. Inspired by P-GNNs, we explore anchor-based PEs as a simplified, non-trainable initialization method to evaluate whether global context can be captured effectively without embedding anchors directly into the architecture. We provide more details on our proposed approach in Section 4.4.

Structure-Preserving Embeddings. Other related approaches focus on embedding node positions such that the topology of the graph is preserved. Structure Preserving Embeddings (SPE) [14] learn a kernel matrix such that a simple graph reconstruction algorithm (e.g. k-NN) run on the embedding will recover the original graph. SPE offers a theoretically principled way to preserve graph topology in Euclidean space. We assess its potential as an initial positional embedding in LSPE to understand whether such geometric structure enhances model performance.

3 Problem Definition

In the LSPE framework proposed by Dwivedi et al. [6], the authors initialize the *learnable* positional embeddings of the nodes using either LapPE or RWPE before refining them iteratively through layer-wise updates. The authors emphasize the importance of the initial embedding to the model's performance. In this project, we investigate alternative initial positional encodings that have not yet been explored within the LSPE framework. Specifically, we evaluate: (1) a PageRank-based distance encoding developed by Li et al. [9], (2) a random anchor-based positional encoding inspired by Position-aware GNNs [17], (3) a Structure-Preserving Embedding (SPE), developed by Shaw et al. [14], and (4) centrality-based encodings such as node degree. We hypothesize that these alternatives may lead to empirical improvements by providing more informative initial positional representations within the LSPE framework. While each of these strategies offers theoretical benefits, they also introduce trade-offs. For example, SPE requires solving a semidefinite program, which may not scale well to large graphs. Anchor-based methods rely on stochastic sampling, which may lead to unstable performance. PageRank encodings can be dense and memory-intensive, and degree-based encodings may not capture sufficient global structure. We aim to assess not just their performance, but also their practical utility within the LSPE framework.

4 Methodology

In this section, we provide a more detailed overview of the LSPE framework and introduce our proposed methods for alternative initializations of node positional encodings.

4.1 The LSPE Framework

The Learnable Structural and Positional Encoding (LSPE) framework, introduced by Dwivedi et al. [6], and illustrated in Figure 1, is a general framework that augments either sparse MPNNs or fully-connected transformer-based GNNs by explicitly modeling both structural and positional representations within each layer of the network. This is achieved by maintaining two parallel channels of information: node feature representations $\mathbf{h}_v^{(l)}$ and positional encodings $\mathbf{p}_v^{(l)}$ at each layer l for every node v.

At layer $l = 0$, $\mathbf{h}_v^{(0)}$ corresponds to the input node features, while $\mathbf{p}_v^{(0)}$ is initialized using a positional encoding method such as Laplacian eigenvectors or random walk-based features. Note that in the LSPE framework, $\mathbf{p}_v^{(0)}$ is *not fixed*: it is propagated and updated layer by layer. This contrasts with the typical message-passing GNNs with positional encodings (MP-GNNs-PE) formulation (see Equation (2) in [6]), where PE is incorporated only once at initialization:

$$\mathbf{h}_v^{(0)} = \mathbf{x}_v + \mathbf{p}_v,$$

and only $\mathbf{h}_v^{(l)}$ is updated in subsequent layers.

In LSPE, both $\mathbf{h}_v^{(l)}$ and $\mathbf{p}_v^{(l)}$ evolve according to separate but interacting update functions, as defined in Equations (3)–(9) of the original paper. The general update scheme is as follows:

$$\mathbf{h}_v^{(l+1)} = \text{Update}_h \left(\mathbf{h}_v^{(l)}, \mathbf{p}_v^{(l)}, \left\{\mathbf{h}_u^{(l)}, \mathbf{p}_u^{(l)}, \mathbf{e}_{uv} \mid u \in \mathcal{N}(v)\right\}\right), \quad (1)$$

$$\mathbf{p}_v^{(l+1)} = \text{Update}_p \left(\mathbf{p}_v^{(l)}, \left\{\mathbf{p}_u^{(l)}, \mathbf{e}_{uv} \mid u \in \mathcal{N}(v)\right\}\right), \quad (2)$$

where $\mathcal{N}(v)$ denotes the neighbours of node v, and $\mathbf{e}_{uv}$ represents edge features. The Update_h function is typically a message-passing or attention-based block that takes into account both structural and positional embeddings. The Update_p function updates position encodings independently using a separate attention layer.

By decoupling structural message passing (over $\mathbf{h}_v$) and learnable position propagation (over $\mathbf{p}_v$), LSPE enables the model to better capture both local relational patterns and global spatial information across the graph. At the final layer L, the model outputs both $\mathbf{h}_v^{(L)}$ and $\mathbf{p}_v^{(L)}$, which can be pooled or combined for downstream tasks such as graph classification or regression.

This dual-channel propagation scheme has been shown to improve performance on benchmark datasets such as ZINC and OGBG-MOLTOX21, particularly when initialized with strong positional priors like Random Walk Positional encodings (RWPE). Since the LSPE framework maintains and updates positional encodings $\mathbf{p}_v$ independently of the structural representations $\mathbf{h}_v$, the quality of the *initial* positional encoding plays a pivotal role in the model's ability to learn meaningful global representations. In fact, Dwivedi et al. demonstrate that omitting positional encodings significantly decreases performance, while initializing with RWPE leads to better results than Laplacian eigenvectors (LapPE). They attribute this difference to the inherent sign and basis ambiguity of the LapPE. This highlights the key insight that we explore: the choice of initialization is a critical design decision within the LSPE framework.

4.2 Random Walk Positional Encodings (RWPE)

Before introducing our proposed alternatives, we detail the RWPE initialization used as the default in LSPE, as it serves as our primary baseline. RWPEs, first introduced by Li et al. [9] and extended in the LSPE framework by Dwivedi et al. [6], provide a diffusion-based method for capturing local and global structural context in graphs. RWPE is based on the idea that the transition probabilities of a random walk over the graph reflect a node's proximity to other nodes and its structural role.

Let $G = (V, E)$ be an undirected graph with $n = |V|$ nodes. Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be the adjacency matrix of the graph and $\mathbf{D} \in \mathbb{R}^{n\times n}$ be the diagonal degree matrix, where $D_{ii} = \sum_j A_{ij}$. The random walk transition matrix is given by:

$$\mathbf{P} = \mathbf{A}\mathbf{D}^{-1}$$

This defines a Markov chain over the graph, where P_{ij} represents the probability of transitioning from node i to node j in one step.

To obtain the RWPE, we compute the powers of the transition matrix to capture the behavior of multi-step random walks. The RWPE for node $v \in V$ for a predefined walk length K, is given by:

$$\mathbf{p}_v^{\text{RWPE}} = \left[\mathbf{P}_{vv}^1, \mathbf{P}_{vv}^2, \ldots, \mathbf{P}_{vv}^K\right] \in \mathbb{R}^K$$

Here, $\mathbf{P}_{vv}^k$ denotes the probability that a random walk starting at node v returns to v after k steps. Intuitively, this sequence captures how "central" or "self-connected" a node is over increasingly long horizons. High return probabilities at multiple steps indicate strong local connectivity or a tendency to remain within a local region.

In practice, we compute the diagonal entries of $\mathbf{P}^k$ for $k = 1, \ldots, K$ and assign them as the positional embedding of each node:

$$\mathbf{P}^k = \underbrace{\mathbf{P}\cdot\mathbf{P}\cdots\mathbf{P}}_{k\text{ times}} \quad \Rightarrow \quad \mathbf{p}_v^{\text{RWPE}}[k] = \left(\mathbf{P}^k\right)_{vv}$$

The resulting positional encoding matrix $\mathbf{P}^{\text{RWPE}} \in \mathbb{R}^{n\times K}$ can be normalized (e.g., using layer norm or min-max scaling) and passed as the initial positional encoding $\mathbf{p}_v^{(0)}$ in the LSPE framework. As shown in prior work, RWPE is especially effective in preserving diffusion and connectivity patterns relevant to downstream tasks such as molecular property prediction.

4.3 PageRank-Based Encodings

We will first investigate the introduction of Distance Encoding (DE) to the LSPE GNN architecture (See related work 2). Li et al. note that DE- and DEA-GNNs prove better expressivity than WL-GNNs, but limitations on regular graphs and high memory requirements may hinder its usage for larger graphs, and they chose not to further investigate the method. We seek to determine the experimental efficacy of this approach on both small and large graphs across a suite of molecular datasets. Li et al. suggest using SPD and generalized PageRank scores as DE measures. As the SPD DE is very similar to the random anchor-based positional encoding, we will focus on generalized PageRank in this paper.

Li et al. define distance encoding as a function

$$\zeta\left(\cdot \mid S, A\right) : V \to \mathbb{R}^k$$

Which is **permutation invariant**, given a target set of nodes $S \in 2^V \setminus \emptyset$ of G with adjacency matrix A. For simplicity, they chose DE as a set aggregation (e.g. sum-pooling) of DEs between nodes u, v where $v \in S$:

$$\zeta\left(u \mid S\right) = \text{AGG}\left(\{\zeta\left(u \mid v\right) \mid v \in S\}\right)$$

We note that the RWPE previously outlined can be classified as a distance-based encoding based on this definition. Multiple proper $\zeta\left(u \mid v\right)$ can be chosen such as ζ_{spd} for shortest-path-distance or ζ_{gpr} for generalized PageRank, which can be represented as:

$$\zeta_{gpr}\left(u \mid v\right) = \sum_{k\geq 1} \gamma_k \left(P^k\right)_{uv} = \left(\sum_{k\geq 1} \gamma_k P^k\right)_{uv}, \quad \gamma_k \in \mathbb{R},\ \forall k \in \mathbb{N}$$

where $P = AD^{-1}$ is the random walk matrix and γ_k is some geometric decay rate for a random walk of length k.

4.4 Random Anchor-Based Positional Encodings

The random anchor-based positional encoding strategy is inspired by Position-aware Graph Neural Networks (P-GNNs) [17]. While P-GNNs integrate anchor distances into a trainable message-passing framework, we instead extract static positional encodings based on the same concept of random anchor distances, which are then used as initial input features in our models. This modification simplifies the learning pipeline while preserving much of the expressive power that comes from anchoring nodes in a global coordinate system.

Let $G = (V, E)$ denote an undirected graph with $n = |V|$ nodes. For each node $v \in V$, we aim to compute a vector-valued positional encoding $\mathbf{p}_v \in \mathbb{R}^d$ that encodes its relative position with respect to a collection of reference sets sampled from the graph. These sets are referred to as *anchor sets*, and can be thought of as stochastic landmarks within the global structure of G.

4.4.1 Anchor Set Sampling: Bourgain-Based Design. The anchor sets $\{S_1, S_2, \ldots, S_k\} \subseteq 2^V$ are constructed following the Bourgain embedding scheme [2, 10], which offers a theoretical guarantee on distortion in mapping arbitrary finite metric spaces into Euclidean space. While the details are outlined in the original paper and in You et al. [17], the key idea is to sample anchor sets of varying cardinalities, providing a set of reference points with varying granularity.

Specifically, we sample $k = c \log^2 n$ anchor sets, where c is a constant hyperparameter. Each anchor set $S_{i,j} \subset V$ is constructed by independently including each node $u \in V$ with probability 2^{-i}, for $i = 1, \ldots, \log n$ and $j = 1, \ldots, c \log n$. The resulting anchor sets are exponentially distributed in size, allowing a balance between spatial resolution and coverage.

Theoretical guarantees from Bourgain's theorem ensure that with this sampling scheme, the distances from any node v to each anchor set $S_{i,j}$ yield a low-distortion embedding of the graph metric space into $\mathbb{R}^d$, with distortion bounded by $O(\log n)$ and dimensionality $d = O(\log^2 n)$.

4.4.2 Encoding Node Positions via Anchor Distances. For each node $v \in V$, we define its positional encoding $\mathbf{p}_v \in \mathbb{R}^d$ as:

$$\mathbf{p}_v = \left[f(d(v, S_{1,1})), f(d(v, S_{1,2})), \ldots, f(d(v, S_{\log n, c \log n})) \right]$$

where $d(v, S_{i,j}) = \min_{u \in S_{i,j}} \mathrm{SP}(v, u)$ denotes the shortest-path distance from v to the closest node in anchor set $S_{i,j}$, and $f : \mathbb{N} \to \mathbb{R}$ is a transformation function. We typically use:

$$f(d) = \frac{1}{d+1}$$

to normalize distances and avoid singularities when $d = 0$. In our experiments, all pairwise distances $d(v, u)$ are computed using NetworkX's built-in shortest-path function.

4.5 Structure Preserving Embedding-Based Encoding

Structure Preserving Embedding (SPE) [14] is a graph embedding technique that aims to preserve the topological structure of a graph in a low-dimensional Euclidean space. Given an undirected graph $G = (V, E)$ with adjacency matrix $A \in \{0, 1\}^{n \times n}$, SPE seeks to learn a kernel matrix $K \in \mathbb{R}^{n \times n}$, where $K \succeq 0$, such that the graph's connectivity can be recovered from the embedded coordinates derived from K using a simple connectivity algorithm like k-nearest neighbours (k-NN). We illustrate this in Figure 2, where we compare the reconstruction of the graph of a sample molecule from a Laplacian eigenvector embedding and our implemented SPE embedding.

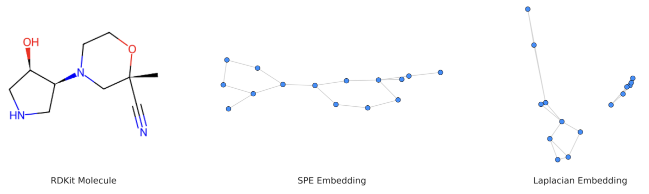

Figure 2: Visualization of molecular graphs reconstructed from different node embeddings. A molecule rendered using RDKit alongside its corresponding 2D layout derived from (1) Structure-Preserving Embeddings (SPE) and (2) Laplacian Eigenmaps (LapPE). SPE better preserves the original molecular topology in low-dimensional space.

4.5.1 Formulation. SPE constructs node embeddings by solving a semidefinite program (SDP) that enforces two primary constraints:

(1) **Low-rank objective:** The optimization favors embeddings that are low-dimensional and close to the spectral embedding of the graph. Specifically, the objective maximizes $\mathrm{tr}(KA)$, where $\mathrm{tr}(\cdot)$ denotes the matrix trace and A is the adjacency matrix. To avoid unbounded growth, a trace constraint $\mathrm{tr}(K) \leq 1$ is imposed.
(2) **Structure preservation:** The optimization includes linear inequalities on K that guarantee graph reconstruction via a chosen algorithm G, typically k-NN. These constraints ensure that for each node i, the distances to connected neighbours $j \in N(i)$ are strictly smaller than to any disconnected node $j' \notin N(i)$.

Let the kernel-defined distance between nodes i and j be defined as:

$$D_{ij} = K_{ii} + K_{jj} - 2K_{ij}$$

Then, the structure preserving constraint for k-NN graphs can be written as:

$$D_{ij} > (1 - A_{ij}) \max_{m \in N(i)} D_{im} \quad \forall i, j$$

This ensures that edges in the original graph correspond to smaller distances in the embedding space.

To solve for K, the following convex optimization problem is defined:

$$\max_{K \succeq 0,\, \mathrm{tr}(K) \leq 1,\, \sum_{i,j} K_{ij} = 0} \mathrm{tr}(KA) - C\xi$$

subject to the structure preserving constraints described above, where $\xi \geq 0$ is a slack variable allowing for soft violations, and C is a penalty coefficient that controls tolerance to these violations.

Once K is optimized, the node embeddings $X \in \mathbb{R}^{n \times d}$ are extracted from the top d eigenvectors of K. These embeddings can then be used as initial PEs in the LSPE framework.

4.5.2 Discussion. In our work, we compute these embeddings once at initialization and assign the resulting coordinate vectors as the node-level positional encodings.

Although solving an SDP is computationally intensive ($O(n^3)$ in general), this cost is incurred only once during pre-processing. Although we recognize this method is likely infeasible for large molecules, we are particularly interested in evaluating if providing such a rich positional embedding can improve performance on small molecules.

4.6 Centrality Encodings

Centrality measures assign a ranking to a node based on their importance in a given network. Standard centrality encodings such as node degree, closeness, and betweenness centrality each offer their own definition of importance. Degree centrality interprets higher number of edges and neighbours as more important. Specifically, we encoded the node indegrees.

Closeness centrality of node i is defined as

$$x_i = \frac{1}{n-1}\sum_j \frac{1}{s_{ij}}$$

where s_{ij} is the shortest-path distance between nodes i and j. Specifically, we encoded the mean distance from a node to other nodes based on shortest-paths.

Betweenness centrality of node i is defined as

$$x_i = \frac{1}{n^2}\sum_{st} \frac{n^i_{st}}{t_{st}}$$

where n^i_{st} is the number of shortest-paths from s to t that pass through i, and t_{st} is the total number of shortest-paths from s to t. Specifically, we encoded the average rate at which shortest-paths pass through a node.

Previous implementations of degree-based centrality encoding showed promising performance boost in transformer-based architectures [16]. Unlike in Ying et al. (2021), they will be added as initial positional encoding methods instead of node features.

5 Experimental Setup

The original LSPE work evaluated four GNN architectures—GatedGCN, PNA, SAN, and GraphiT across three datasets (ZINC, OGBG-MOLTOX21, and OGBG-MOLPCBA), under different combinations of positional encoding strategies (None, RWPE, LapPE), LSPE augmentation (enabled or not), and positional loss (enabled or not).

In our study, we reproduce the results with LSPE augmentation enabled with no positional loss and RWPE as the initial PE. We further experiment with a variety of unexplored PEs, assessing how these initializations affect performance across the same set of models, datasets, and experimental conditions.

5.1 Models

We use the four GNN architectures from the LSPE paper:

- GatedGCN [3]
- PNA (Principal Neighbourhood Aggregation) [4]
- SAN (Self-Attention Network) [8]
- GraphiT (Graph Transformer with Structural Priors) [12]

Each model is tested within the LSPE framework under different choices of positional encoding initialization.

5.2 Datasets

We conduct experiments on the following benchmark datasets:

- **ZINC:** A molecular property regression benchmark consisting of real-world molecular graphs. ZINC is widely used for predicting properties like constrained solubility. Following Dwivedi et al. [6], we use a 12,000-graph subset split into 10,000 training, 1,000 validation, and 1,000 test samples.
- **OGBG-MOLTOX21:** A binary classification dataset for predicting 12 toxicity measurements of molecules. It consists of 7,831 scaffold-split graphs, where the scaffold split is based on 2D molecular structure to ensure realistic generalization performance.
- **OGBG-MOLPCBA:** A large-scale multi-label binary classification dataset for bioactivity prediction. It includes 437,929 scaffold-split molecular graphs and 128 binary targets indicating active or inactive assay responses.

All three datasets were originally benchmarked in the LSPE paper. To ensure the integrity of our experimental setup, we first reproduce the baseline results reported in the original paper under the same training conditions. This step verifies that our pipeline is correctly aligned before introducing alternative initialization strategies for the positional encodings. It is important to note that, while the original LSPE paper reports metrics averaged over four independent runs, our results are based on a single run per configuration due to computational constraints.

5.3 Evaluation Metrics

- **ZINC:** Mean Absolute Error (MAE) on the test set. Lower values indicate better performance on this regression task.
- **MOLTOX21:** Average ROC-AUC score across the 12 binary classification targets, as computed using the OGB evaluator. Higher is better.
- **MOLPCBA:** Mean Average Precision (AP) score across the 128 binary classification targets, as evaluated by OGB tools. This metric is used due to significant class imbalance.

5.4 Training Protocol

While specific hyperparameters may vary across experiments with different positional encodings, the majority of our results are obtained using the following setup:

- **Optimizer:** Adam
- **Initial learning rate:** 5×10^{-4}, with `ReduceLROnPlateau` (factor = 0.8, patience = 10)
- **Batch size:** 256
- **Dropout:** 0.4 in message-passing layers; 0.1 in readout layers
- **Epochs:** Up to 1000, with early stopping based on validation performance
- **Loss function:** Binary cross-entropy with logits
- **Readout function:** Sum pooling over node embeddings

5.5 Infrastructure

- **Hardware:** Compute Canada Beluga cluster with NVIDIA Tesla V100-SXM2-32GB/64GB GPUs, Local machine with an NVIDIA RTX 2080 8GB GDDR6 GPU, Mimi server with NVIDIA RTX A2000 12GB
- **Software:** Python 3.10, PyTorch, DGL, OGB, NumPy, SciPy, CVXPY (for SPE).
- **Reproducibility:** Experiments are conducted with fixed random seeds (41), consistent batch sizing, and checkpointing enabled.

6 Results

We successfully reproduced the baseline LSPE results using both MPNN- and Transformer-based models on the ZINC and MOLTOX21 datasets. While we were unable to fully reproduce results on the MOLPCBA dataset due to computational constraints, preliminary findings suggest similar trends across all three datasets. For succinctness, we omit the partial MOLPCBA results. As shown in Table 1, our reproduced results closely match the published values, suggesting that our training and evaluation pipelines are consistent with the original implementation and can reliably isolate the effects of different positional encodings. The outcomes of our various experimental initial PEs can also be found in Table 1.

7 Discussion of Results

The majority of our explored positional encodings did not outperform the baseline RWPE. However, centrality-based encodings showed promising improvements in several evaluation settings. In this section, we provide a detailed discussion of results for each PE method individually, highlighting observed trends, failure modes, and possible interpretations.

7.1 Anchor Based PEs

We observe that the random anchor-based PE tends to induce overfitting in the MPNN models. We see this in particular on the ZINC and MOLTOX21 datasets, where higher train performance did not translate into improved test metrics. One possible explanation is that these encodings introduce high-dimensional signals without enough regularization, causing the model to fit noise in the training data. To test this hypothesis, we increased the dropout rate and reduced the dimension of the PEs in an attempt to reduce the overfitting. We did not see any improvement in the results. We hypothesize that additionally, in smaller graphs like those in ZINC, global position may not be as discriminative as local structure, reducing the benefit of complex PEs. In other words, in the molecular setting, while global awareness is indeed important, it is the local structures that have larger influence on the molecular properties. This idea is supported in the existing work [13] by the underperformance of global attention based methods on molecular tasks. We attempted to implement a more deterministic sampling scheme in an effort to reduce the complexity introduced by the Bourgain sampling (while reducing the theoretical benefits). For this, we used the top $log(n)$ nodes in terms of degree as anchors. This did not yield improved results, likely because in molecules this introduces a bias toward specific subgraph regions.

On a more general note, it is also possible that the initial encodings lack alignment with task-relevant inductive biases (e.g., chemical substructure), which RWPE appears to capture more effectively. In other words, the positional signals provided by the anchor based encoding may not highlight the kinds of structural patterns that are most informative for the prediction task, whereas RWPE implicitly emphasizes these relevant substructures.

7.2 Distance Based PEs (GPR)

A similar issue of overfitting is observed for the generalized PageRank PE. This is especially clear on the ZINC dataset and likely stems from noise due to the small graphs and generalized PageRank's importance on global relative positioning which is less useful for the small molecules in the dataset. Recall that a target set $S \in 2^V \setminus \emptyset$ must be chosen to satisfy the definition of a Distance Encoding. Our results have shown that the choice and size of S significantly affects the performance of the DE on the model. Our initial tests used a fixed size with random sampling to choose nodes. By simply setting the size of S dynamically, we observed a 5.82% increase in performance from standard GPR. To take this further, we used an element-based approach to sampling along with the dynamic-sized set. To do this, we used the node features provided by the ZINC dataset to prefer certain nodes when sampling based on their element. The anchors are chosen in order of the most frequent atom in the molecule. The first is almost always Carbon which enables PageRank to focus more on the less frequent elements. This brought forth a 16.4% improvement over the basic GPR DE. While GPR performed poorly compared to other encodings, this method of Element-Aware Sampling could be applied to other encodings to potentially improve their results. Additionally, the increase in train MAE and decrease in test MAE for Element-Aware Sampling GPR suggests a reduction in the aforementioned overfitting issue.

7.3 Centrality Based PEs

Our experiment with centrality-based positional encodings yielded promising results that consistently outperformed the RWPE baseline for the MOLTOX21 dataset. Among the three centrality encodings, closeness centrality achieved the best performance with marginal improvements across all four models. It recorded an increase of 0.94% (0.7813 vs 0.774), 2.2% (0.7685 vs 0.752), 0.87% (0.7515 vs 0.745), and 0.53% (0.753 vs 0.749) for the GatedGCN, PNA, SAN, and GraphiT models respectively. This aligns with the observation made in the original paper by comparing sparse GNNs (GatedGCN, PNA) against Transformer GNNs (SAN, GraphiT), where sparse GNNs saw a greater performance despite Transformer GNNs being theoretically more powerful[6]. Its ability to identify nodes that are centrally positioned in the graph likely helped highlight key substructures in the proximity. Degree-based encoding performed comparably to the baseline and even demonstrated the highest TestAUC for the SAN model with an increase of 1.5% (0.7559 vs 0.745). Our results further support the notion that information on local structures is essential for molecular graphs. In contrast, our results for betweenness centrality did not yield any improvement upon the baseline and was therefore omitted from Table 1 due to space constraints. We hypothesize that there may not be strong correlation between molecular properties and the bridge nodes in between groups.

For the ZINC dataset, our centrality-based encodings were unable to surpass the RWPE baseline. Among the three centrality encodings, betweenness centrality achieved results most similar to the baseline, followed by closeness, then degree. Interestingly, we noticed over-fitting in betweenness only. Like our other encodings, the improved training metrics did not correlate to better test results. However, although closeness and degree yielded lower TestMAE than other types of encodings, they also had higher TrainMAE. The results suggest that the models were less effective at fitting the data using closeness and degree encodings overall. A possible explanation is that the models were simply able to memorize the

Table 1: Comparison of published results (*), our reproduced baselines (†), and new positional encodings across datasets.

	Model	Init PE	L	#Param	TestMAE±	TrainMAE±	Epochs	Time (epoch/total)
ZINC	GatedGCN	RWPE *	16	522870	0.093 ± 0.003	0.014 ± 0.003	440.75	15.17s / 1.99hr
	GatedGCN	RWPE †	16	522870	0.095	0.015	445.00	15.20s / 1.98hr
	GatedGCN	GPR	16	522870	0.3575	0.0114	387.00	16.17s / 1.78hr
	GatedGCN	GPR - Dynamic Anchors	16	522870	0.3367	0.0129	388.00	17.61s / 1.92hr
	GatedGCN	GPR - Element-Aware	16	522870	0.2989	0.0219	318.00	17.62s / 1.57hr
	GatedGCN	Betweenness	16	522870	0.1226	0.0087	480.00	8.71s / 1.20hr
	GatedGCN	Closeness	16	522870	0.1301	0.0175	395.00	8.69s / 0.97hr
	PNA	RWPE *	16	503061	0.095 ± 0.002	0.022 ± 0.002	462.25	127.69s / 16.61hr
	PNA	Betweenness	16	503061	0.1145	0.0148	489.00	77.92s / 10.65hr
	PNA	Closeness	16	503061	0.1390	0.0230	493.00	79.87s / 11.00hr
	GraphiT	RWPE *	10	588065	0.106 ± 0.002	0.028 ± 0.002	420.50	125.39s / 14.84hr
	GraphiT	GPR	10	588065	0.3421	0.0308	370.00	133.14s / 13.85hr
	Model	**Init PE**	**L**	**#Param**	**TestAUC±**	**TrainAUC±**	**Epochs**	**Time (epoch/total)**
MOLTOX21	GatedGCN	RWPE *	8	1063821	0.775 ± 0.003	0.906 ± 0.003	246.50	5.99s / 0.63hr
	GatedGCN	RWPE †	8	1063939	0.774	0.8972	238.00	4.10s / 0.29hr
	GatedGCN	Anchor	8	1063939	0.7507	0.9264	256.00	4.03s / 0.31hr
	GatedGCN	Degree	8	1063939	0.7748	0.9138	285.00	78.63s / 6.26hrs
	GatedGCN	Closeness	8	1063939	0.7813	0.8987	311.00	19.66s / 1.72hrs
	PNA	RWPE *	8	5357393	0.761 ± 0.007	0.871±0.009	215.50	7.61s/0.56hr
	PNA	RWPE †	8	5357393	0.752	0.8510	216.00	6.76s / 0.43hr
	PNA	Anchor	8	5352481	0.7359	0.8571	206.00	6.70s/ 0.44hr
	PNA	Degree	8	5357533	0.7636	0.8739	233.00	44.11s / 2.89hrs
	PNA	Closeness	8	5357533	0.7685	0.8938	216.00	44.13s / 2.68hrs
	SAN	RWPE *	10	1051017	0.744 ± 0.007	0.918 ± 0.018	281.75	30.82s / 2.84hr
	SAN	RWPE †	10	1051081	0.745	0.899	294.00	27.11s / 2.28hr
	SAN	Anchor	10	1051081	0.736	0.889	269.00	25.73s / 2.03hr
	SAN	Degree	10	1051081	0.7559	0.8729	273.00	17.42s / 1.35hr
	SAN	Closeness	10	1051081	0.7515	0.8823	276.00	137.36s / 10.59hr
	GraphiT	RWPE *	10	1051782	0.746 ± 0.010	0.934 ± 0.016	279.75	27.92s / 2.57hr
	GraphiT	RWPE †	10	1051782	0.749	0.867	283.00	24.74s / 2.05hr
	GraphiT	Anchor	10	1051782	0.708	0.877	274.00	23.41s / 1.90hr
	GraphiT	Degree	10	1051852	0.7502	0.8900	287.00	16.00s / 1.31hr
	GraphiT	Closeness	10	1051852	0.7530	0.9046	228.00	180.51s / 11.52hr

Note: Published results (*) are averaged over 4 runs as reported in Dwivedi et al. [6]. All reproduced (†) and original results reflect a single run due to computational constraints. Due to the use of different machines with varying computational resources, the reported time may show an unfair comparison of each PE's compute time.
Init PE legend: Anchor = Random Anchor-Based Positional Encoding; **GPR** = Generalized PageRank Encoding; **Degree** = Degree-Based Encoding; **Betweenness** = Betweenness Centrality Encoding; **Closeness** = Closeness Centrality Encoding.

training examples more effectively for betweenness without learning to generalize to unseen data, and the encodings offered less meaningful signals to molecular properties than RWPE.

Overall, we found the effectiveness of positional encodings to be highly task-dependent. While centrality-based encodings can meaningfully enhance performance in molecular classification, they may be less suitable for regression tasks without careful alignment to the prediction objective.

7.4 Structure Preserving Embeddings

While structure-preserving embeddings (SPE) offer a principled approach to graph representation, particularly for smaller graphs, they proved far too computationally intensive to scale effectively. In practice, the reliance on convex optimization solvers such as CVXPY resulted in inconsistent performance, with many graphs failing to yield a solution within the allotted time. For the sake of space, we omit the limited results that were successfully computed from Table 1.

8 Conclusion

In this project, we explored various alternative initial PE methods that had not been explored within the LSPE framework. We accurately reproduced the results of the original paper from Dwivedi et al. and implemented several PE methods from various papers across the literature. As in the original paper, we tested our implemented methods on 3 popular molecular datasets with various GNN and graph transformer based models. We found that initializing the PE with anchor based methods yielded reasonable results but did not surpass the RWPE baseline reported in the original paper.

While Distance Encodings like GPR did not perform well compared to baseline in our experiments, the results of the Element-Aware Sampling method display future research potential. By applying this sampling method to other encodings, their performance may be increased for similar tasks. Additional research is needed to confirm this.

The structure preserving embedding technique was more computationally intensive than originally thought, leading to subpar performance due to extremely limited scaling. Finally, we produce results that marginally surpassed the reported baseline results using centrality encodings for classification tasks, specifically the closeness centrality. This result offers promising reassurance that exploring additional PEs within this framework could indeed be a fruitful task.

8.1 Future Work

While we explored a wide range of positional encoding (PE) methods in this work, many avenues remain open for further investigation. We plan to continue extending this work by exploring additional hyperparameters, including dropout rates, PE dimensionalities, and learning rate schedules. As previously mentioned, we hypothesize that tuning these parameters may help reduce the overfitting observed in the anchor-based and PageRank-based PEs.

Improvements to the PageRank algorithm were observed when using Element-Aware Sampling. This may be applied to current implementations of PageRank on molecular datasets, or to other Positional Encodings that use anchor/sample sets.

Although we aim to provide initial conclusions on the practicality and robustness of the PEs studied, we believe that further experimentation and evaluation are necessary to assess their generalizability across different tasks and architectures fully.

Finally, a particularly interesting direction for future work involves tracking and interpreting the evolution of PEs across layers within the LSPE framework. Understanding how these encodings develop during training could offer valuable insight into what the model is focusing on at each stage of the learning process.

References

[1] Mikhail Belkin and Partha Niyogi. 2003. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation* 15, 6 (2003), 1373–1396.

[2] J. Bourgain. 1985. On Lipschitz Embedding of Finite Metric Spaces in Hilbert Space. *Israel Journal of Mathematics* 52 (1985), 46–52. doi:10.1007/BF02776078

[3] Xavier Bresson and Thomas Laurent. 2017. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553* (2017).

[4] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. 2020. Principal neighbourhood aggregation for graph nets. *Advances in neural information processing systems* 33 (2020), 13260–13271.

[5] Vijay Prakash Dwivedi, Chaitanya K. Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2020. Benchmarking Graph Neural Networks. *CoRR* abs/2003.00982 (2020). https://arxiv.org/abs/2003.00982

[6] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2021. Graph Neural Networks with Learnable Structural and Positional Representations. *CoRR* abs/2110.07875 (2021). arXiv:2110.07875 https://arxiv.org/abs/2110.07875

[7] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1263–1272. https://proceedings.mlr.press/v70/gilmer17a.html

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

[9] Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. 2020. Distance Encoding: Design Provably More Powerful Neural Networks for Graph Representation Learning. arXiv:2009.00142 [cs.LG] https://arxiv.org/abs/2009.00142

[10] Nati Linial, Eran London, and Yuri Rabinovich. 1995. The Geometry of Graphs and Some of Its Algorithmic Applications. *Combinatorica* 15, 2 (1995), 215–245. doi:10.1007/BF01200757

[11] Dominic Masters, Josef Dean, Kerstin Klaser, Zhiyi Li, Sam Maddrell-Mander, Adam Sanders, Hatem Helal, Deniz Beker, Ladislav Rampášek, and Dominique Beaini. 2022. GPS++: An Optimised Hybrid MPNN/Transformer for Molecular Property Prediction. *arXiv e-prints*, Article arXiv:2212.02229 (Nov. 2022), arXiv:2212.02229 pages. doi:10.48550/arXiv.2212.02229 arXiv:2212.02229 [q-bio.QM]

[12] Grégoire Mialon, Dexiong Chen, Margot Selosse, and Julien Mairal. 2021. Graphit: Encoding graph structure in transformers. *arXiv preprint arXiv:2106.05667* (2021).

[13] Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. 2022. Recipe for a General, Powerful, Scalable Graph Transformer. *arXiv e-prints*, Article arXiv:2205.12454 (May 2022), arXiv:2205.12454 pages. doi:10.48550/arXiv.2205.12454 arXiv:2205.12454 [cs.LG]

[14] Blake Shaw and Tony Jebara. 2009. Structure preserving embedding. In *Proceedings of the 26th Annual International Conference on Machine Learning* (Montreal, Quebec, Canada) *(ICML '09)*. Association for Computing Machinery, New York, NY, USA, 937–944. doi:10.1145/1553374.1553494

[15] Oliver Wieder, Stefan Kohlbacher, Mélaine Kuenemann, Arthur Garon, Pierre Ducrot, Thomas Seidel, and Thierry Langer. 2020. A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies* 37 (2020), 1–12.

[16] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do Transformers Really Perform Bad for Graph Representation? *CoRR* abs/2106.05234 (2021). arXiv:2106.05234 https://arxiv.org/abs/2106.05234

[17] Jiaxuan You, Rex Ying, and Jure Leskovec. 2019. Position-aware Graph Neural Networks. *CoRR* abs/1906.04817 (2019). arXiv:1906.04817 http://arxiv.org/abs/1906.04817