GIST: GAUGE-INVARIANT SPECTRAL TRANSFORMERS FOR SCALABLE GRAPH NEURAL OPERATORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adapting transformers to meshes and graph-structured data presents significant computational challenges, particularly when leveraging spectral methods that require eigendecomposition of the graph Laplacian, a process incurring cubic complexity for dense matrices or quadratic complexity for sparse graphs, a cost further compounded by the quadratic complexity of standard self-attention mechanism. Conventional approximate spectral methods compromise the gauge symmetry inherent in spectral basis selection, risking the introduction of spurious features tied to the gauge choice that could undermine generalization. In this paper, we propose a transformer architecture that is able to preserve gauge symmetry through distance-based operations on approximate randomly projected spectral embeddings, achieving linear complexity while maintaining gauge invariance. By integrating this design within a linear transformer framework, we obtain end-to-end memory and computational costs that scale linearly with the number of nodes in the graph. Unlike approximate methods that sacrifice gauge symmetry for computational efficiency, our approach maintains both scalability and the principled inductive biases necessary for effective generalization to unseen graph structures in inductive graph learning tasks. We demonstrate our method's flexibility by benchmarking on standard transductive and inductive node classification tasks, achieving results matching the state-of-the-art on multiple datasets. Furthermore, we demonstrate scalability by deploying our architecture as a discretization-free Neural Operator for large-scale computational fluid dynamics mesh regression, surpassing state-of-the-art performance on aerodynamic coefficient prediction reformulated as a graph node regression task.

1 INTRODUCTION

Following their incredible success for processing sequential data in Natural Language Processing, Transformers (Vaswani et al., 2017) have been demonstrating a remarkable capacity for handling data of increasing structural complexity. Lee et al. (2019a) have proposed a variant of the transformer block to permutation invariant data with their Set Transformer architecture; Dosovitskiy et al. (2021) have adapted the self-attention mechanism to 2D images with the very influential Vision Transformer architecture; and Bertasius et al. (2021) have extended transformers to video analysis with their Video Vision Transformer (ViViT), demonstrating how attention mechanisms can capture both spatial and temporal dependencies across video frames. This progression from sequential text to increasingly structured data indicates a trajectory suggesting that Transformers are poised to tackle even more complex data structure, including irregular meshes and graphs.

Indeed, recent developments in adapting Transformers to graphs have shown promising results in capturing long-range dependencies that traditional Graph Neural Networks (GNNs) struggle with due to their reliance on localized message passing (Dwivedi et al., 2022; Zhu et al., 2023). Unlike GNNs that aggregate information from nearest neighboring nodes by iterating through layers, Transformers can directly capture global relationships across the whole graph through self-attention, enabling them to reason about distant node interactions in a single layer.

However, adapting transformers to graphs introduces significant computational and theoretical challenges that must be carefully addressed to realize their full potential on large-scale graphs.

The main problem addressed in this work is introducing meaningful "graph positional encoding": embeddings that, analogous to positional encodings for sequential data, bias attention toward nearby nodes within the graph. Spectral embeddings derived from the graph Laplacian are natural candidates for capturing graph proximity structure due to their ability to reflect global and local geometric relationships. However, exact computation of spectral embeddings via eigendecomposition is prohibitive for large graphs, scaling cubically with the number of nodes, and at best, quadratically for sparse graphs, making these approaches impractical in real-world settings.

Resorting to more computationally feasible approximate methods introduces its own challenges, as such approximations can inadvertently break the intrinsic symmetries of the Laplacian eigenspaces. This often results in embeddings sensitive to arbitrary basis choices and alignment, leading to spurious inductive biases that compromise the generalization power and robustness of the model.

Our approach seeks to overcome this barrier by designing graph position encodings that are both computationally efficient and fundamentally gauge-invariant, ensuring that graph structure is encoded without introducing undesirable biases due to arbitrary choices in spectral representations.

2 Related Works

Graph Transformers. Graphormer (Ying et al., 2021) introduces the idea of integrating structural encodings such as shortest path distances and centrality in Transformers. Similarly, the paper by Dwivedi et al. (2022) proposes LSPE (Learnable Structural and Positional Encodings), an architecture that decouples structural and positional representations. Kreuzer et al. (2021) propose Spectral Attention Network (SAN), which introduce learned positional encodings from the full Laplacian spectrum. Park et al. (2022) develop Graph Relative Positional Encoding (GRPE), which extends relative positional encoding to graphs by considering features representing node-topology and node-edge interactions. Hierarchical Graph Transformer (Zhu et al., 2023) addresses scalability to million-node graphs through graph hierarchies and coarsening techniques.

Scalable Attention Architectures. Recent advances have tried to tackle the quadratic scaling of self-attention through various approaches, including cross-attention bottlenecks that map inputs to fixed-size latent representations or concepts (Jaegle et al., 2021b; Rigotti et al., 2022), kernel-based attention mechanisms using random feature approximations (Choromanski et al., 2020), feature map decomposition methods that linearize the attention computation (Katharopoulos et al., 2020), and memory-efficient variants with sub-linear complexity (Likhosherstov et al., 2021). As noted by Dao & Gu (2024), many such linear transformer models are directly related to linear recurrent models such as state-space-models (Gu et al., 2021; 2022; Gu & Dao, 2023; Chennuru Vankadara et al., 2024)

Neural Operators. Further addressing the scalability of these graph-based methods is essential for applying them to complex domains such as geometry meshes and point clouds. In these settings, graphs are induced by the connectivity of an underlying continuous object whose discretization is not unique: it can be sampled at arbitrarily many densities and resolutions. High-density discretizations can render the graph prohibitively large, undermining both efficiency and scalability in existing methods. As a result, efficient mesh downsampling and/or re-discretization onto regular lattices (e.g., via SDF-based volumetric grids), and task-aware coarsening learned by GNNs, were commonly required to make these problems tractable.

In recent years, neural operators have shown success in learning maps between continuous function spaces rather than fixed-dimensional vectors. Two properties are crucial here: (i) *discretization invariance*, i.e., a single set of parameters applies across discretizations (meshes, resolutions, and sampling locations) of the same underlying continuum problem; and (ii) *global integration*, i.e., the ability to represent nonlocal interactions via learned integral kernels, rather than being limited to finite-receptive-fields. Formally, a neural operator composes learned integral operators with pointwise nonlinearities, yielding universal approximation results for continuous nonlinear operators and implementations that share weights across resolutions. Our approach preserves these neural operator properties and improves scalability, allowing it to be applied to these cases (Kovachki et al., 2023).

Foundational operator families. The Fourier Neural Operator (FNO) parameterizes kernels in the spectral domain and evaluates them with FFT-based spectral convolutions, sharing weights across

resolutions and enabling efficient nonlocal interactions on grids (Li et al., 2021). The Graph Neural Operator (GNO) realizes the kernel via message passing, supporting irregular meshes and geometry variation while keeping the learned map discretization-agnostic (Li et al., 2020). Convolutional Neural Operators (CNOs) define continuous convolutions with learnable kernels and interpolation, specifying the operator in the continuum and discretizing only at runtime (Raonić et al., 2023).

Hybrid designs pair geometry-aware encoders with operator layers to handle complex shapes. GINO couples a graph encoder/decoder with a latent FNO on a proxy grid from SDF or point-cloud inputs and shows convergence across large 3D, multi-geometry problems (Li et al., 2023). Encoder–decoder operator learners, such as DeepONet, use a branch network for inputs and a trunk network for coordinate queries, directly supporting heterogeneous sampling (Lu et al., 2021); U-NO adds a multi-resolution U-shaped backbone for multiscale effects (Rahman et al., 2022).

Transformers as neural operators. Self-attention behaves as a learned, data-dependent kernel integral, and with suitable positional features can approximate continuous maps on variable-length sets for discretization-invariant operator learning; cross-attention evaluates outputs at arbitrary coordinates (Tsai et al., 2019; Yun et al., 2020; Lee et al., 2019b; Jaegle et al., 2021a). *Transolver* casts PDE operator learning as attention from query coordinates to context tokens built from input fields, yielding resolution-agnostic inference and strong generalization across meshes (Wu et al., 2024a). Recent operator-oriented transformers, e.g., GNOT, add geometric normalization and gating to stabilize training on irregular meshes and multi-condition PDEs (Hao et al., 2023).

3 Approach

3.1 Preliminaries

Self-attention, permutation invariance and positional encoding. The core of Transformers, the celebrated *attention mechanism*, is a powerful algorithm to condition the processing of each input token contextually to all other tokens. To describe that more formally, given tokens $x_i \in \mathbb{R}^d$ for $i = 1, \dots, N$, one builds corresponding query, key and value representations q_i, k_i, v_i, typically by applying linear or affine operations f_q, f_k, f_v on the tokens themselves:

$$q_i = f_q(x_i), \quad k_i = f_k(x_i), \quad v_i = f_v(x_i). \tag{1}$$

Scaled dot-product attention then consists in computing:

$$o_i = \sum_{j=1}^{N} \alpha_{ij} v_j, \quad \text{where } \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{N} \exp(e_{ik})} \text{ with } e_{ij} = \frac{q_i^\top k_j}{\sqrt{d}} \quad \text{are the } \textit{attention weights}. \tag{2}$$

This operation is manifestly *permutation invariant*, as arbitrarily permuting indices j results in the same output o_i.

In order to break this invariance and enable the architecture to pay more or less attention to tokens depending on their absolute or relative position as opposed to all inputs uniformly, already the original Transformer paper introduced *positional encoding*, fixed or learned embeddings that encode a token's coordinates in the sequence (Vaswani et al., 2017). These would modify equation 1 by adding positional embeddings PE_i to the tokens: $x_i \leftarrow x_i + \mathrm{PE}_i$.

Another more generalizable approach by Shaw et al. (2018) implements *relative positional embeddings* by modifying equation 2 with $e_{ij} = \frac{q_i^\top k_j}{\sqrt{d}} + b_{ij}$, i.e. adding bias terms that shift attention in favor of neighboring tokens by decreasing its contribution as $|i - j|$ increases.

This reveals to be a very promising approach, since it can be naturally extended to other structures than tokens on a linear sequence, as long as one can provide a bias matrix b_{ij} that reflects distances on the data structure of interest.

Graph Laplacian and spectral methods in graph transformers. The *graph Laplacian* is a fundamental operator for representing graph structure and has been widely proposed as a basis for representing graph structures in connection to and as a generalization of CNNs (see e.g. Bruna et al. (2014), end Defferrard et al. (2016)).

Its connection with Transformers can be intuitively motivated starting from the original sine and cosine positional encodings originally proposed by Vaswani et al. (2017), and in particular by observing that sine and cosine are eigenfunctions of the Laplace operator (the diffusion operator on a linear 1D domain). By analogy, the eigenvectors of the graph Laplacian (the generalization of the Laplacian on graphs) should serve as natural basis to encode positional information on graphs, extending the idea of positional encoding from sequences to graphs.

More formally, given the adjacency matrix $A \in \mathbb{R}^{N\times N}$ of an undirected graph, the (normalized) *graph Laplacian* is defined as $\mathcal{L} = \mathbb{1} - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, where D is the diagonal degree matrix. This operator is related to one-step transition probabilities of a random walk on the graph where edges represent possible transitions weighted by their connectivity (Chung, 1996). The graph Laplacian induces a distance between given nodes i and j on the graph known as the *resistance distance*, which is defined as $\Omega(i,j) = (e_i - e_j)^\top \mathcal{L}^\dagger (e_i - e_j)$, where $e_i \in \mathbb{R}^N$ is the ith canonical basis element indicizing node i, and $\mathcal{L}^\dagger$ denotes the Moore-Penrose pseudoinverse of $\mathcal{L}$ (Klein & Randić, 1993).

Indeed, since the graph Laplacian is symmetric and positive semidefinite, $\Omega(i,j)$ satisfies the *metric axioms*: *non-negativity* (i.e., $\Omega(i,j) \geq 0$ for all i,j), *identity* (i.e., $\Omega(i,j) = 0$ iff $i = j$), *symmetry* (i.e., $\Omega(i,j) = \Omega(j,i)$), and the *triangle inequality* (i.e., $\Omega(i,j) \leq \Omega(i,k) + \Omega(k,j)$ for all i,j,k).

Using the spectral decomposition of $\mathcal{L}$, the resistance distance can now be written as follows: $\Omega(i,j) = (e_i - e_j)^\top \sum_k \frac{1}{\lambda_k} u_k u_k^\top (e_i - e_j) = \sum_k \frac{1}{\lambda_k}((u_k)_i - (u_k)_j)^2$, where λ_k are the nonzero eigenvalues and u_k the corresponding eigenvectors. This can be rewritten as:

$$\Omega(i,j) = ||\phi_i - \phi_j||^2 \text{ for the } \textbf{Laplacian eigenmaps } \phi_i \text{ with components } (\phi_i)_k = \frac{1}{\sqrt{\lambda_k}}(u_k)_i. \tag{3}$$

To summarize, the spectrum and eigenvectors of the graph Laplacian can be used to construct vectors, the Laplacian eigenmaps ϕ_i, which are node embeddings whose distance reflects a bonafide metric on the graph, making them natural candidates for positional encoding on graph structures (Dwivedi & Bresson, 2021). Unfortunately, this idea incurs a major computational bottleneck due to the eigendecomposition having $\mathcal{O}(N^3)$ time complexity for dense graphs and at best $\mathcal{O}(N^2)$ for sparse graphs with specialized solvers. This complexity severely limits the scalability of transformer approaches using spectral graph embeddings to large graphs.

Spectral Embedding Approximation. Because direct spectral decomposition scales poorly with graph size, we propose to resort to standard iterative approximate decomposition methods such as truncated series approximation that can be implemented efficiently for sparse matrices like graph adjacencies (Saad, 2003). We are particularly interested in methods that scale linearly with the number of nodes.

We briefly summarize the idea behind the use of a von Neumann series for approximating the resistance distance, as it provides useful intuition on our method, but we will refer to the standard literature for more details.

We first start by writing the graph Laplacian as $\mathcal{L} = \mathbb{1} - P$, where P is a *transition matrix on the graph* (defined as $P = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ if we are using the normalized graph Laplacian as before, or $P = D^{-1}A$ if instead we are using the so-called random walk graph Laplacian). Methods such the von Neumann series are based on the fact that for P with spectral radius < 1 (as in our case): $\mathcal{L}\cdot\sum_k P^k = (\mathbb{1}-P)(\sum_k P^k) = \mathbb{1}$, meaning that $\sum_k P^k$ equals the (pseudo-) inverse of $\mathcal{L}$ defining the resistance distance Ω. This iteration can be truncated at a large enough iteration step K.

These informal considerations reveal 2 interesting points: 1) since P is a transition matrix that can be though of as an operator diffusion "probability mass" out on the graph a P^k will in some sense connect neighbors k-hops away on the graph; 2) because P is typically sparse (due to real-world graphs having connectivity $\ll$ number of nodes), we can take advantage of the celebrated Johnson-Lindenstrauss Lemma and use random projections to accurately approximate the iteration P^k using a matrix multiplication which is exponentially smaller than the size of P and number of nodes N (Dasgupta & Gupta, 2003). In short, the idea here is to generate a random projection $R \in \mathbb{R}^{N\times r}$ with $r = \mathcal{O}(\log(N)/\epsilon^2)$ for a small error tolerance $\epsilon > 0$, and exploit the fact that computing PR is fast and so is $P \cdot (P^{k-1}R)$ given $P^{k-1}R$, which in turn gives an iterative way of computing (P^kR) that reduces the time complexity from $\mathcal{O}(N^3k)$ to $\mathcal{O}(N \cdot r \cdot k)$, realizing our goal to achieve

a scalable method liner in the number of nodes N. This method can be further improved upon by exploiting sparsity as detailed in Chen et al. (2019), which develop FastRP, the specific random projection-based truncated approximation method that we will use in practice.

At this point, it is important to note that approximate methods like FastRP will allow us to recover Laplacian eigenmaps ϕ_i in equation 3 up to a an arbitrary projection R. This then translates to "rotated eigenmaps" $\tilde{\phi}_i = R^\top \phi_i \in \mathbb{R}^r$. Fortunately, a consequence of Johnson-Lindenstrauss is that for appropriate random projection ensembles, $R^\top R$ is concentrated around the $N \times N$ unit matrix: $||R^\top R - \mathbb{1}||^2 = \mathcal{O}(\epsilon)$ (Dasgupta & Gupta, 2003)), meaning that the rotated eigenmaps $\tilde{\phi}_i$ closely preserve the resistance distance: $\Omega(i,j) = ||\phi_i - \phi_j||^2 = ||\tilde{\phi}_i - \tilde{\phi}_j||^2 + \mathcal{O}(\epsilon)$.

Invariance Breaking Challenges. The rotated eigenmaps $\tilde{\phi}_i$ preserve the resistance distance structure, making them good, efficient, and low-dimensional graph positional embeddings for a given graph. However, the specific choice of matrix R corresponds to selecting a particular basis in the eigenspace of the graph Laplacian, thereby breaking the intrinsic *gauge invariance* associated with arbitrary rotations, sign flips, and eigenvector multiplicities that naturally occur in the eigendecomposition (Bronstein et al., 2017; Dwivedi et al., 2022). Breaking gauge invariance (i.e., choosing an arbitrary basis) introduces spurious inductive biases that could link downstream model outputs to irrelevant coordinate system artifacts rather than meaningful graph structure. More concretely, a neural network that learns to associate an output to some particular feature spuriously arising from the arbitrary basis and rotation choices, won't be able to generalize on inputs that no longer present the spurious feature.

This motivates our approach, which consists in using approximate eigenmaps $\tilde{\phi}_i$ as Transformer "positional" encoding on graphs, but by making sure that the Transformer will only be able to operate on them through mechanisms that preserve gauge invariance, i.e. do not depend on R and the corresponding graph Laplacian eigendecomposition.

3.2 Our Approach: GIST.

Similarity-Based Encodings and Operations. Approximate spectral decomposition methods like FastRP (Chen et al., 2019) allow us to efficiently (in time complexity $\mathcal{O}(N)$ in the number of nodes in the graph) obtain well-defined graph positional embeddings as projected Laplacian eigenmaps. However, the basis choice and rotation implicit in the projection break gauge invariance, which, as discussed, introduces spurious features that could negatively bias our architecture. This could be particularly pernicious in an *inductive task*, if our model were to be trained for instance on a graph whose node have graph positional embeddings $\{R\phi_i\}_i$, but tested on a different graph with node embeddings $\{R'\phi'_j\}_j$ obtained from a different projection matrix R' and/or eigenmaps based on a different arbitrary way to span eigenspaces with higher multiplicity (which could happen simply due to numerical instability, as Bronstein et al. (2017) note).

Our main contribution is to propose a Transformer architecture that recovers gauge invariance from invariance-breaking spectral embeddings, which can then still take advantage of the efficiency of approximate spectral decomposition methods while side-stepping the challenges that they introduce.

Our strategy is simple and based on the observation that, within a gauge choice, similarities between spectral embeddings are (approximately) preserved: $(R\phi_i)^\top (R\phi_j) = \phi_i^\top (R^\top R)\phi_j \approx \phi_i^\top \phi_j$. If, therefore, we are to adapt our Transformer architecture so that graph positional encoding only affects operations as a function of relative similarities, then gauge invariance is recovered (since changing the gauge does not alter any computation).

Gauge-Invariant Spectral Self-Attention. We are now ready to introduce our main contribution which is **Gauge-Invariant Spectral Transformer (GIST)**. The first ingredient of GIST is *Gauge-Invariant Spectral Self-Attention* which, for each node $i = 1, \ldots, N$ in a graph with N nodes, uses graph positional embeddings $\tilde{\phi}_i = R\phi_i \in \mathbb{R}^r$ obtained by an arbitrary random projection $R \in \mathbb{R}^{r \times N}$ and gauge choice ϕ_i (where the gauge choice, i.e. the eigenbasis u_k, affect the eigenmaps ϕ_i through equation 3). Gauge-Invariant Spectral Self-Attention then modifies equation 1 as follows:

$$q_i = \tilde{\phi}_i, \quad k_i = \tilde{\phi}_i, \quad v_i = f_v(x_i).$$

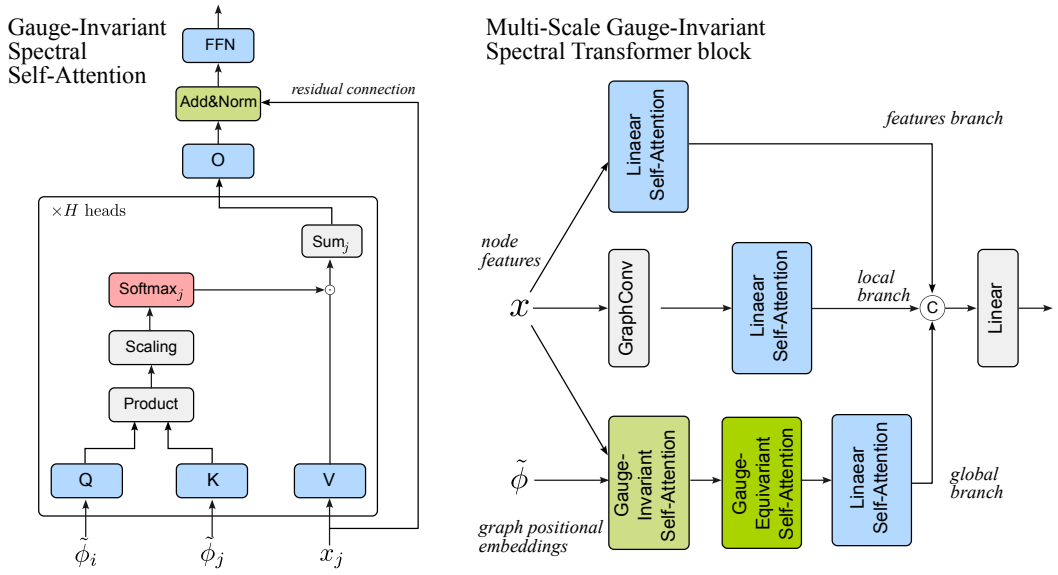

Figure 1: Gauge-Invariant Spectral Transformer. *Left*: Gauge-Invariant Spectral Self-Attention operates on graph positional embeddings $\tilde{\phi}$ as queries and keys, and node features x as values. The output of the self-attention operation is then combined to x thorough as residual connection. Limiting $\tilde{\phi}$ to queries and keys preserves gauge invariance across the self-attention block. *Right*: Gauge-Invariant Self-Attention is embedded in a Multi-Scale Gauge-Invariant Spectral Transformer Block which comprises 3 parallel branches inspired by EfficientViT.

What this achieves is that the attention weights α_{ij} in equation 2 "unbreak" gauge invariance since $e_{ij} = \frac{q_i^\top k_j}{\sqrt{d}} = \frac{\tilde{\phi}_i^\top \tilde{\phi}_j}{\sqrt{d}} = \frac{\phi_i^\top R^\top R \phi_j}{\sqrt{d}} = \frac{\phi_i^\top \phi_j}{\sqrt{d}}$, which are gauge invariant (see Fig. 1, left). Notice that R can also incorporate the gauge choice inherent in choosing eigenvectors signs and eigenspace rotations, meaning that also the corresponding invariances are restored. Algorithm 1 in Section A.1 explains how we compute graph spectral positional embeddings and Algorithm 2 details how the implementation of Gauge-Invariance Spectral Self-Attention relates to regular Self-Attention.

Gauge-Equivariant Spectral Self-Attention. The Gauge-Invariant Spectral Self-Attention operation thus preserves gauge invariance, but at the cost of giving up a lot of the flexibility of regular self-attention. In particular, there is no mechanism that allows for a modification of the vectors $\tilde{\phi}_i$ through learning. In fact, applying even just a linear operation on $\tilde{\phi}_i$ would break again gauge invariance. However, notice that rescaling each $\tilde{\phi}_i$ by a scalar possibly depending on the node features $s(x_i) \in \mathbb{R}$ would modify the similarity between graph positional embeddings in the same way across gauge choices, since scalars commute with orthogonal projections, meaning that it is an *equivariant* operation across gauges: $(s(x_i)\tilde{\phi}_i)^\top (s(x_j)\tilde{\phi}_j) = s(x_i)s(x_j)(\tilde{\phi}_i^\top \tilde{\phi}_j) = s(x_i)s(x_j)(\phi_i^\top \phi_j)$.

Remarkably, such a gauge-equivariant operation can also be straight-forwardly implemented via a modification of self-attention by modifying equation 2 as follows:

$$q_i = f_q(x_i), \quad k_i = f_k(x_i), \quad v_i = \tilde{\phi}_i,$$

and the output in equation 2 such that it is constrained to operate on $\tilde{\phi}_i$, i.e. $\tilde{\phi}_i^{l+1} = \sum_{j=1}^{N} \alpha_{ij} v_j$, where $\tilde{\phi}_i^{l+1}$ indicates the graph positional encoding that will be used in the next layer $l+1$. Algorithm 3 in Section A.1 details how the implementation of Gauge-Invariance Spectral Self-Attention relates to regular Self-Attention.

Linear Self-Attention, and Multi-Scale Architecture. Gauge-Invariant Spectral Self-Attention ensures that we can compute reliable graph positional encoding with linear time complexity in the number of nodes in the graph N. In order to maintain that linear scaling end-to-end, the very

last component of our architecture aims to address the quadratic scaling of Transformers by implementing a linear version of self-attention. In particular, we implement the *linear transformer* by Katharopoulos et al. (2020), and in order to fully exploit its capabilities and mitigate its drawbacks like the reported lack of sharp attention scores compared to softmax attention, we design a parallel architecture inspired from EfficientViT by Cai et al. (2024) who proposed a multi-scale linear attention architecture. Just like EfficientViT our *Multi-Scale Gauge-Invariant Spectral Transformer Block* has 3 parallel branches: a *feature branch* consisting in a linear transformer block acting on node features x alone, a *local branch* consisting in a graph-convolution layer also acting on x followed by a linear transformer block, and a *global branch* consisting in our Gauge-Invariant Spectral Self-Attention layer followed by a Gauge-Equivariant Spectral Self-Attention layer (which as explained act on both node features x and graph positional embeddings $\tilde{\phi}$) then followed by a linear transformer. In keeping with the analogy with EfficientViT, the role of the graph-convolution layer (which simply averages node features across adjacent nodes) is to emphasize local information, which would be otherwise diffused by linear attention. Conversely, Gauge-Invariant Spectral attention has the role of integrating global information across the graph. This block is represented in the right panel of Fig. 1 and represents a unit layer that is sequentially repeated multiple times in our models.

4 RESULTS

4.1 NODE CLASSIFICATION TASKS

To demonstrate the key advantages of GIST, we evaluate it on both transductive and inductive node classification datasets. Transductive tasks are a common graph neural networks paradigm, and consists in training and evaluating the model on the same graph, with the goal of predicting at test-time node labels that where not provided at training (infilling). Inductive tasks on the other hand, operate on a disjoint set of graphs and aim to predict properties on an entirely new graph.

Experiment Setup We evaluate our method on transductive graph benchmarks using the official training, validation, and test splits and evaluation protocols. For each method, we select optimal hyperparameters by optimizing over the validation split of each dataset. To obtain the final result, we conduct a training run on the combined training and validation set and evaluate the model on the corresponding test set. We train across multiple random seeds and report the mean ± standard deviation of the relevant metric.

4.1.1 TRANSDUCTIVE TASKS

We evaluate our method on the three standard Planetoid citation benchmarks for the transductive setting where the whole graph is observed at train time: Cora (2,708 nodes, 5,429 edges, 1,433 bag-of-words features, seven classes), CiteSeer (3,327 nodes, 4,732 edges, 3,703 features, six classes), and PubMed (19,717 nodes, 44,338 edges, 500 features, three classes). Train-val-test sets follow the Planetoid public split, and we report node-classification accuracy (Sen et al., 2008; Yang et al., 2016; Kipf & Welling, 2017)

Across these benchmarks, GIST is competitive with strong graph convolutional and transformer-style baselines (see Table 1). On Pubmed, GIST attains the best mean accuracy among the reported methods ($81.20\% \pm 0.41$), narrowly surpassing enhanced GCN variants (e.g., $81.12\% \pm 0.52$) and outperforming GAT/GraphSAGE families. On Cora and Citeseer, GIST achieves results comparable to the top results (within $\sim$1–2 points of GCNII/SGFormer and the enhanced GCN), landing at $84.00\% \pm 0.60$ and $71.31\% \pm 0.50$, respectively.

4.1.2 INDUCTIVE TASKS

We evaluate our method on two inductive benchmarks: PPI, a collection of 24 disjoint tissue-specific protein–protein interaction graphs where nodes (proteins) have 50 features and 121 non–mutually-exclusive GO labels (We use the standard split of 20 graphs for training, 2 for validation, and 2 for testing, and report micro-averaged F1 on the unseen test graphs), and Elliptic, a time-evolving directed Bitcoin transaction graph with 203,769 transactions (nodes), 234,355 payment-flow edges, and 166 features across 49 snapshots, labeled licit/illicit with many nodes unlabeled due to class

Table 1: Transductive node classification on the Planetoid benchmarks (Cora, Citeseer, Pubmed). We report test accuracy (%) as mean±std across random seeds using the standard public split (higher is better). Benchmark results are taken from the following references: (Kipf & Welling, 2017; Hu et al., 2021; Luo et al., 2024; Veličković et al., 2018; Chiang et al., 2019; OGB, 2025; Zeng et al., 2020; Chen et al., 2020; Brody et al., 2022; Choi, 2022; Wu et al., 2024b) .

Model	Cora (Accuracy ↑)	Citeseer (Accuracy ↑)	Pubmed (Accuracy ↑)
GCN (baseline)	81.60 ± 0.40	71.80 ± 0.01	79.50 ± 0.30
GraphSAGE	71.49 ± 0.27	71.93 ± 0.85	79.41 ± 0.53
GIN	77.60 ± 1.10	–	–
GAT	83.00 ± 0.70	69.30 ± 0.80	78.40 ± 0.90
GCNII	**85.50 ± 0.50**	72.80 ± 0.60	79.80 ± 0.30
GATv2	82.90 ±	71.60 ±	78.70 ±
SGFormer	84.82 ± 0.85	72.60 ± 0.20	80.30 ± 0.60
GCN (enhanced)	85.10 ± 0.67	**73.14 ± 0.67**	81.12 ± 0.52
GIST (Ours)	84.00 ± 0.60	71.31 ± 0.50	**81.20 ± 0.41**

Table 2: Inductive node classification on PPI and Elliptic. Results are reported as micro-F1 (higher is better). Benchmark results are taken from the following references: (Chen et al., 2020; Weber et al., 2019; Chiang et al., 2019; Veličković et al., 2018; Zhang et al., 2018; Zeng et al., 2020; Chen et al., 2025b; Brody et al., 2022)

Model	PPI (micro-F1 ↑)	Elliptic Bitcoin (micro-F1 ↑)
GCN (baseline)	51.50 ± 0.60	96.10 ±
GraphSAGE	61.20 ±	**97.70** ±
GAT	97.30 ± 0.02	96.90 ±
GaAN	98.70 ±	–
Cluster-GCN	99.36 ±	–
GraphSAINT	99.50 ±	–
GCNII	**99.53 ± 0.01**	–
GCNIII	**99.50 ± 0.03**	–
GATv2	96.30 ±	–
GIST (Ours)	**99.50 ± 0.03**	94.70 ± 0.03

imbalance (we train on the first 29 time steps, validate on the next 5, and test on latter 14, reporting micro-F1.

On PPI, GIST matches the best large-scale sampling methods and deep residual GCNs (see Table 2), reaching $99.50\% \pm 0.03$ micro-F1, on par with GCNIII and within noise of the strongest GCNII setting (99.53%). On the temporally inductive Elliptic dataset, GIST attains $94.70\% \pm 0.03$ micro-F1. While this trails the strongest GraphSAGE configuration, GIST maintains stable performance across future time steps. These findings collectively demonstrate GIST's effectiveness as a competitive graph learning approach, validating the successful trade-off between computational overhead and representational power.

4.2 Neural Operators

GIST as Neural Operator. As discussed, GIST fulfills the core properties of neural operators. It defines an operator $\mathcal{G} : \mathcal{X} \to \mathcal{Y}$ that maps functions sampled on a mesh to functions on the same (or another) mesh while remaining agnostic to discretization. Positional information enters only through inner products of Laplacian eigenmaps, which are invariant to spectral gauge (sign flips/rotations within eigenspaces) and stable across mesh refinements/coarsenings. In the refinement limit, these similarities recover the continuum Green's-function kernel. Self-attention therefore realizes a nonlocal, geometry-aware kernel integral $o_i = \sum_j \alpha_{ij} v_j$ with $\alpha_{ij} = \mathrm{softmax}(\tilde{\phi}_i^\top \tilde{\phi}_j / \sqrt{d})$, providing *global integration*.

Table 3: Surface pressure prediction accuracy on **DrivAerNet**. Reported metrics: mean squared error (MSE) and relative ℓ_2 error (Rel L2). Lower is better for both.

Year	Model	MSE ($\times 10^{-2}$)	Rel L2 (%)
2024	RegDGCNN (Elrefaie et al., 2024)	9.01	28.49
2024	Transolver (Wu et al., 2024a)	5.37	22.52
2025	FigConvNet (Choy et al., 2025)	4.38	20.98
2025	TripNet (Chen et al., 2025a)	4.23	20.35
2025	**GIST (ours)**	**4.16**	**20.10**

Mesh Based Inductive Task. To illustrate these properties and the scalability of GIST, we apply it on a real-world continuous mesh based problem: the DrivAerNet dataset. DrivAerNet is a high-fidelity CFD dataset of parametric car geometries comprising 4,000 designs; with each design providing a watertight surface mesh with approximately 0.5M surface vertices per car and accompanying aerodynamic fields (pressure, velocity, wall-shear) plus global coefficients (e.g., C_d). We model each car as a graph whose nodes are surface vertices and edges follow mesh connectivity. Our task is *node-level* regression of the surface pressure field on previously unseen cars. Inductive generalization is enforced by holding out entire designs for validation and testing as per the published split. We report per-mesh R^2 and RMSE for surface pressure prediction, averaged across test meshes. Among the datasets considered here, DrivAerNet is the largest: both in total data volume and in per-graph node count (Elrefaie et al., 2024).

As illustrated in Table 3, GIST outperforms existing methods on this task. Relevant baseline method MSE and L2 values are pulled from their respective papers. For GIST, 3 layers were used with a hidden dimension of 384 and a node dropout of 0.7. The spectral embedding was computed per vertex with a degree of 96, and the 256 embedding dimensions were appended with the euclidean vertex coordinates and normal vectors. Unlike existing methods on this task, no lossy down-sampling to a lattice grid or arbitrary latent space is required due to the inherent scalability of GIST.

We hypothesize that DrivAerNet's low-valence surface meshes intensify oversquashing in message-passing GNNs, since each vertex communicates only with close neighbors. In contrast, GIST's global attention and geometry-aware embeddings (e.g., spectral/shape features) provide nonlocal coupling and coordinate-stable cues that better capture long-range flow interactions critical for surface field predictions.

5 Conclusions

We presented a gauge-invariant spectral transformer architecture that addresses the fundamental computational and theoretical challenges of applying Transformers to graph-structured data. Our method achieves linear complexity in both memory and computation while preserving, and unlike existing approximate methods that sacrifice gauge invariance for efficiency, our approach obtains scalability, while avoiding pernicious inductive biases that would come about with breaking invariance and hinder generalization. Experimental validation demonstrates state-of-the-art performance on standard transductive and inductive node classification benchmarks, confirming the balance between computational efficiency and representational quality. The scalability of our approach is further validated by state-of-the-art performance on a large-scale computational fluid dynamics task, a benchmark that is usually tackled with discretization-invariant neural operators acting on lossy grid discretization schemes that our approach can circumvent by virtue of its intrinsic scalability.

6 Reproducibility and Ethics Statement

To ensure reproducibility of our results we will release our complete source code, including preprocessing scripts, model implementations, and evaluation pipelines, upon publication.

The authors would also like to disclose that a Large Language Model (LLM) was used to minimally aid in the writing of the paper by paraphrasing specific sentences for brevity, clarity, and to avoid stylistic flaws such as repetition. In addition, once a first Related Works section had been compiled by us, and LLM was used to help retrieve and discover possible relevant papers that we had missed. All references provided by the LLM were carefully checked against the literature by the authors.

References

Leaderboards for Node Property Prediction. https://snap-stanford.github.io/ogb-web/docs/leader_nodeprop/, May 2025.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is Space-Time Attention All You Need for Video Understanding?, June 2021.

Shaked Brody, Uri Alon, and Eran Yahav. How Attentive are Graph Attention Networks?, January 2022.

Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4): 18–42, 2017.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral Networks and Locally Connected Networks on Graphs, May 2014.

Han Cai, Junyan Li, Muyan Hu, Chuang Gan, and Song Han. EfficientViT: Multi-Scale Linear Attention for High-Resolution Dense Prediction, February 2024.

Haochen Chen, Syed Fahad Sultan, Yingtao Tian, Muhao Chen, and Steven Skiena. Fast and Accurate Network Embeddings via Very Sparse Random Projection. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, pp. 399–408, New York, NY, USA, November 2019. Association for Computing Machinery. ISBN 978-1-4503-6976-3. doi: 10.1145/3357384.3357879.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and Deep Graph Convolutional Networks, July 2020.

Xiaoyu Chen et al. Tripnet: Learning large-scale high-fidelity 3d car aerodynamics with triplane networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

Yancheng Chen, Wenguo Yang, and Zhipeng Jiang. Wide & Deep Learning for Node Classification, May 2025b.

Leena Chennuru Vankadara, Jin Xu, Moritz Haas, and Volkan Cevher. On Feature Learning in Structured State Space Models. *Advances in Neural Information Processing Systems*, 37:86145–86179, 2024.

Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 257–266, July 2019. doi: 10.1145/3292500.3330925.

Julie Choi. Personalized PageRank Graph Attention Networks, August 2022.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking Attention with Performers. *arXiv:2009.14794 [cs, stat]*, September 2020.

Junhyuk Choy et al. Figconvnet: Factorized implicit global convolution for automotive aerodynamic surrogate modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Fan RK Chung. Lectures on spectral graph theory. *CBMS Lectures, Fresno*, 6(92):17–21, 1996.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality, May 2024.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, January 2003. ISSN 1042-9832, 1098-2418. doi: 10.1002/rsa.10073.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021.

Vijay Prakash Dwivedi and Xavier Bresson. A Generalization of Transformer Networks to Graphs. *arXiv:2012.09699 [cs]*, January 2021.

Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph Neural Networks with Learnable Structural and Positional Representations, February 2022.

Mohamed Elrefaie, Florin Morar, Angela Dai, and Faez Ahmed. DrivAerNet++: A large-scale multimodal car dataset with computational fluid dynamics simulations and deep learning benchmarks. *arXiv preprint arXiv:2406.09624*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34:572–585, 2021.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently Modeling Long Sequences with Structured State Spaces, August 2022.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*. PMLR, 2023. URL https://proceedings.mlr.press/v202/hao23c.html.

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, February 2021.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Cătălin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver io: A general architecture for structured inputs & outputs, 2021a. URL https://arxiv.org/abs/2107.14795. ICLR 2022 version.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, pp. 4651–4664. PMLR, 2021b.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. *arXiv:2006.16236 [cs, stat]*, June 2020.

Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks, February 2017.

D. J. Klein and M. Randić. Resistance distance. *Journal of Mathematical Chemistry*, 12(1):81–95, December 1993. ISSN 1572-8897. doi: 10.1007/BF01164627.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24:1–97, 2023. URL http://jmlr.org/papers/v24/21-1524.html.

Devin Kreuzer, Dominique Beaini, William L. Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking Graph Transformers with Spectral Attention, October 2021.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks. In *International Conference on Machine Learning*, pp. 3744–3753. PMLR, May 2019a.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 2019b. URL https://proceedings.mlr.press/v97/lee19d.html.

Ping Li, Trevor J. Hastie, and Kenneth W. Church. Very sparse random projections. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 287–296, Philadelphia PA USA, August 2006. ACM. ISBN 978-1-59593-339-3. doi: 10.1145/1150402.1150436.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations, 2020. URL https://arxiv.org/abs/2003.03485.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Zongyi Li, Nikola Borislavov Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d PDEs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023. URL https://arxiv.org/abs/2309.00583. See also arXiv:2309.00583.

Valerii Likhosherstov, Krzysztof M. Choromanski, Jared Quincy Davis, Xingyou Song, and Adrian Weller. Sub-linear memory: How to make performers slim. *Advances in Neural Information Processing Systems*, 34:6707–6719, 2021.

Lu Lu, Pengzhan Jin, Guofei Pang, Zongyi Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021. doi: 10.1038/s42256-021-00302-5.

Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic GNNs are Strong Baselines: Reassessing GNNs for Node Classification, October 2024.

Wonpyo Park, Woonggi Chang, Donggeon Lee, Juntae Kim, and Seung-won Hwang. GRPE: Relative Positional Encoding for Graph Transformer, October 2022.

Md Ashiqur Rahman, Zachary E. Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators, 2022. URL https://arxiv.org/abs/2204.11127.

Bogdan Raonić, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of PDEs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023. URL https://proceedings.neurips.cc/paper/2023/hash/f3c1951b34f7f55ffaecada7fde6bd5a-Abstract-Conference.html.

M. Rigotti, C. Miksovic, I. Giurgiu, T. Gschwind, and P. Scotton. Attention-based Interpretability with Concept Transformers. In *International Conference on Learning Representations (ICLR)*, 2022.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, January 2003. ISBN 978-0-89871-534-7 978-0-89871-800-3. doi: 10.1137/1.9780898718003.

Prithviraj Sen, Gal Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-Attention with Relative Position Representations, April 2018.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. In *Proceedings of EMNLP-IJCNLP*, pp. 4344–4353, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1443.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *arXiv:1706.03762 [cs]*, June 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, February 2018.

Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I. Weidele, Claudio Bellei, Tom Robinson, and Charles E. Leiserson. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. In *KDD '19 Workshop on Anomaly Detection in Finance*, 2019.

Haixu Wu et al. Transolver: A fast transformer solver for pdes on general geometries. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024a.

Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. SGFormer: Simplifying and Empowering Transformers for Large-Graph Representations, August 2024b.

Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pp. 40–48, 2016.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do Transformers Really Perform Bad for Graph Representation?, November 2021.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1912.10077.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. GraphSAINT: Graph Sampling Based Inductive Learning Method, February 2020.

Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. GaAN: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs, March 2018.

Wenhao Zhu, Tianyu Wen, Guojie Song, Xiaojun Ma, and Liang Wang. Hierarchical Transformer for Scalable Graph Learning, May 2023.

A APPENDIX

A.1 PSEUDO-CODE

Note that, contrary to the main text, in the pseudocode we use matrix notation where vectors are row vectors, following the standard computer science convention. Thus, $\mathbf{\Phi} \in \mathbb{R}^{N \times r}$ has nodes as rows and embedding dimensions as columns, consistent with machine learning frameworks where data samples are stored row-wise.

Algorithm 1 Broken Gauge-Invariance Spectral Embeddings (based on FastRP (Chen et al., 2019))

Require: Graph adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, embedding dimensionality r, maximum power k
Ensure: Matrix of N node graph positional embeddings $\mathbf{\Phi} \in \mathbb{R}^{N \times r}$
1: Produce very sparse random projection $\mathbf{R} \in \mathbb{R}^{N \times r}$ according to Li et al. (2006)
2: $\mathbf{P} \leftarrow \mathbf{A} \cdot \mathbf{D}^{-1}$ the random walk transition matrix, where $\mathbf{D}$ is the degree matrix
3: $\mathbf{\Phi_1} \leftarrow \mathbf{P} \cdot \mathbf{R}$
4: **for** $i = 2$ to k **do**
5: $\quad \mathbf{\Phi}_i \leftarrow \mathbf{P} \cdot \mathbf{\Phi}_{i-1}$
6: **end for**
7: $\mathbf{\Phi} = \mathbf{\Phi}_1 + \mathbf{\Phi}_2 + \cdots + \mathbf{\Phi}_k$
8: **return** $\mathbf{\Phi}$

Below we provide pseudo-code for the core computations of GIST, the *Gauge-Invariant Spectral Self-Attention* block and the *Gauge-Equivariant Spectral Self-Attention* block. For illustration purposes, we compare the algorithms to a stripped down implementation of self-attention. We then point out the modifications that our algorithms apply to that basic functionality by indicating in red red any addition to vanilla self-attention and in strike-through text anything that has to be removed.

Algorithm 2 Gauge-Invariant Spectral Self-Attention (softmax version)

Require: Node feature tokens $\mathbf{X} \in \mathbb{R}^{N \times d}$, graph positional embeddings $\mathbf{\Phi} \in \mathbb{R}^{N \times r}$
Ensure: Output sequence $\mathbf{O} \in \mathbb{R}^{N \times d}$ to be applied to features $\mathbf{X}$
1: **// Compute attention matrices**
2: ~~$\mathbf{Q} \leftarrow \mathbf{X} \cdot \mathbf{W}_Q$ where $\mathbf{W}_Q \in \mathbb{R}^{d \times d}$~~ $\mathbf{Q} \leftarrow \mathbf{\Phi}$
3: ~~$\mathbf{K} \leftarrow \mathbf{X} \cdot \mathbf{W}_K$ where $\mathbf{W}_K \in \mathbb{R}^{d \times d}$~~ $\mathbf{K} \leftarrow \mathbf{\Phi}$
4: $\mathbf{V} \leftarrow \mathbf{X} \cdot \mathbf{W}_V$ where $\mathbf{W}_V \in \mathbb{R}^{d \times d}$
5: **// Compute attention weights and output**
6: $\mathbf{A} \leftarrow \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{r}}\right)$
7: $\mathbf{O} \leftarrow \mathbf{A}\mathbf{V}$
8: **return** $\mathbf{O}$

Algorithm 3 Gauge-Equivariant Spectral Self-Attention (softmax version)

Require: Node feature tokens $\mathbf{X} \in \mathbb{R}^{N \times d}$, graph positional embeddings $\mathbf{\Phi} \in \mathbb{R}^{N \times r}$
Ensure: Output sequence $\mathbf{O} \in \mathbb{R}^{N \times d}$ to be applied to graph positional embeddings $\mathbf{\Phi}$
1: **// Compute attention matrices**
2: $\mathbf{Q} \leftarrow \mathbf{X} \cdot \mathbf{W}_Q$ where $\mathbf{W}_Q \in \mathbb{R}^{d \times d}$
3: $\mathbf{K} \leftarrow \mathbf{X} \cdot \mathbf{W}_K$ where $\mathbf{W}_K \in \mathbb{R}^{d \times d}$
4: ~~$\mathbf{V} \leftarrow \mathbf{X} \cdot \mathbf{W}_V$ where $\mathbf{W}_V \in \mathbb{R}^{d \times d}$~~ $\mathbf{V} \leftarrow \mathbf{\Phi}$
5: **// Compute attention weights and output**
6: $\mathbf{A} \leftarrow \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{r}}\right)$
7: $\mathbf{O} \leftarrow \mathbf{A}\mathbf{V}$
8: **return** $\mathbf{O}$