GRAPH-BASED OPERATOR LEARNING FROM LIMITED DATA ON IRREGULAR DOMAINS

Anonymous authors
Paper under double-blind review

ABSTRACT

Operator learning seeks to approximate mappings from input functions to output solutions, particularly in the context of partial differential equations (PDEs). While recent advances such as DeepONet and Fourier Neural Operator (FNO) have demonstrated strong performance, they often rely on regular grid discretizations, limiting their applicability to complex or irregular domains. In this work, we propose a **G**raph-based **O**perator **L**earning with **A**ttention (GOLA) framework that addresses this limitation by constructing graphs from irregularly sampled spatial points and leveraging attention-enhanced Graph Neural Netwoks (GNNs) to model spatial dependencies with global information. To improve the expressive capacity, we introduce a Fourier-based encoder that projects input functions into a frequency space using learnable complex coefficients, allowing for flexible embeddings even with sparse or nonuniform samples. We evaluated our approach across a range of 2D PDEs, including Darcy Flow, Advection, Eikonal, and Nonlinear Diffusion, under varying sampling densities. Our method consistently outperforms baselines, particularly in data-scarce regimes, demonstrating strong generalization and efficiency on irregular domains.

1 INTRODUCTION

Learning mappings between function spaces is a fundamental task in computational physics and scientific machine learning, especially for approximating solution operators of partial differential equations (PDEs). Operator learning offers a paradigm shift by learning the solution operator directly from data, enabling fast, mesh-free predictions across varying input conditions. Despite their success, existing operator learning models such as DeepONet (Lu et al., 2019) and Fourier Neural Operator (FNO) (Li et al., 2020a) exhibit notable limitations that restrict their applicability in more general settings. A key shortcoming lies in their reliance on regular, uniform grid discretizations. FNO, for instance, requires inputs to be defined on fixed Cartesian grids to leverage fast Fourier transforms efficiently. This assumption limits their flexibility and generalization ability when applied to problems defined on complex geometries, irregular meshes, or unstructured domains, which are common in real-world physical systems. Furthermore, these models often struggle with sparse or non-uniformly sampled data, leading to degraded performance and increased computational cost when adapting to more realistic, heterogeneous scenarios.

To address these limitations, we propose a **G**raph-based **O**perator **L**earning with **A**ttention (GOLA) framework that leverages Graph Neural Networks (GNNs) to learn PDE solution operators over irregular spatial domains. By constructing graphs from sampled spatial coordinates and encoding local geometric and functional dependencies through message passing, the model naturally adapts to non-Euclidean geometries. To enhance global expressivity, we further incorporate attention-based mechanisms that can capture long-range dependencies more effectively and a Fourier-based encoder that projects input functions into a frequency domain using learnable complex-valued bases. Our model exhibits superior data efficiency and generalization, achieving smaller prediction errors with fewer training samples and demonstrating robustness under domain shifts.

The main contributions of this work are as follows:

- We introduce GOLA, a unified architecture combining spectral encoding and attention-enhanced GNNs for operator learning on irregular domains.

- We propose a learnable Fourier encoder that projects input functions into a frequency domain tailored for spatial graphs.
- Through extensive experiments, we demonstrate that GOLA generalizes across PDE types, sample densities, and resolution shifts, achieving state-of-the-art performance in challenging data-scarce regimes.

2 Related Work

There are many latest research about graph and attention methods in scientific machine learning (Xiao et al., 2024), (Kissas et al., 2022), (Boullé and Townsend, 2024), (Xu et al., 2024), (Jin and Gu, 2023), (Cuomo et al., 2022) (Kovachki et al., 2024), (Nelsen and Stuart, 2024), (Batlle et al., 2023).

Graph neural networks for scientific machine learning. (Battaglia et al., 2018) applies shared functions over nodes and edges, captures relational inductive biases and generalizes across different physical scenarios. (Bar-Sinai et al., 2019) learns data-driven discretization schemes for solving PDEs by training a neural network to predict spatial derivatives directly from local stencils. By replacing hand-crafted finite difference rules with learned operators, it adapts discretizations to the underlying data for improved accuracy and generalization. (Sanchez-Gonzalez et al., 2020) predicts future physical states by performing message passing over the mesh graph, capturing both local and global dynamics without relying on explicit numerical solvers. Graph Kernel Networks (GKNs) (Li et al., 2020b) directly approximates continuous mappings between infinite-dimensional function spaces by utilizing graph kernel convolution layers. PDE-GCN (Wang et al., 2022) represents partial differential equations on arbitrary graphs by combining spectral graph convolution with PDE-specific inductive biases. It learns to predict physical dynamics directly on graph-structured domains, enabling generalization across varying geometries and discretizations. The Message Passing Neural PDE Solver (Brandstetter et al., 2022) formulates spatiotemporal PDE dynamics by applying learned message passing updates on graph representations of the solution domain. Physics-Informed Transformer (PIT) (Dos Santos et al., 2023) embeds physical priors into the Transformer architecture to model PDE surrogate solutions. It leverages self-attention to capture long-range dependencies and integrates PDE residuals as soft constraints during training to improve generalization. GraphCast (Lam et al., 2024) learns the Earth's atmosphere as a spatiotemporal graph and uses a graph neural network to iteratively forecast future weather states based on past observations. It performs message passing over the graph to capture spatial correlations and temporal dynamics, enabling accurate medium-range forecasts.

Attention-based methods for scientific machine learning. U-Netformer (Liu et al., 2022) proposes a hybrid neural architecture that combines the U-Net's hierarchical encoder-decoder structure with transformer-based attention modules to capture both local and global dependencies in PDE solution spaces. Tokenformer (Zhou et al., 2023) reformulates PDE solving as a token mixing problem by representing input fields as tokens and applying self-attention across them to model spatial correlations. Adaptive Fourier Neural Operators (AFNO) (Guibas et al., 2021) are an efficient token-mixing mechanism for vision transformers that perform resolution-independent global convolution in the Fourier domain—enhanced by block-diagonal channel mixing, adaptive weight sharing, and frequency sparsification—to deliver quasi-linear complexity and superior performance over traditional self-attention on high-resolution image tasks. Our proposed GOLA combines the local relational strengths of attention-enhanced GNNs and the global spectral capabilities of Fourier-based encoding. This hybrid approach has shown notable improvements in generalization and data efficiency, particularly under challenging data-scarce conditions on irregular domains.

3 Methodology

3.1 Problem Formulation

Consider the general form of a PDE

$$\mathcal{N}[u](\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega \times [0, \infty) \tag{1}$$

where $\mathbf{x}$ denotes a compact representation of the spatial and temporal coordinates, Ω is the spatial domain, and $[0, \infty)$ is the temporal domain. $\mathcal{N}$ is a differential operator, $u(\mathbf{x})$ is the unknown solution,

and $f(\mathbf{x})$ is a given source term. The objective is to learn the solution operator $\mathcal{G} : \mathcal{F} \to \mathcal{U}$, where $\mathcal{F}$ and $\mathcal{U}$ are Banach spaces. We assume access to a training dataset $\mathcal{D} = \{(f_n, u_n)\}_{n=1}^{N}$, consisting of multiple input-output function pairs, where each $f_n(\cdot)$ and $u_n(\cdot)$ is represented by discrete samples over a finite set of points.

While existing approaches such as DeepONet and FNO have demonstrated strong performance, they typically rely on structured, grid-based discretizations of the domain. This assumption limits their applicability to unstructured meshes, complex geometries, and adaptively sampled domains. To overcome this limitation, we employ GNNs for operator learning by representing the domain as a graph. This allows for modeling on arbitrary domains and sampling patterns. Once trained, the operator learning model can efficiently predict the solution u for a new instance of the input f at random locations.

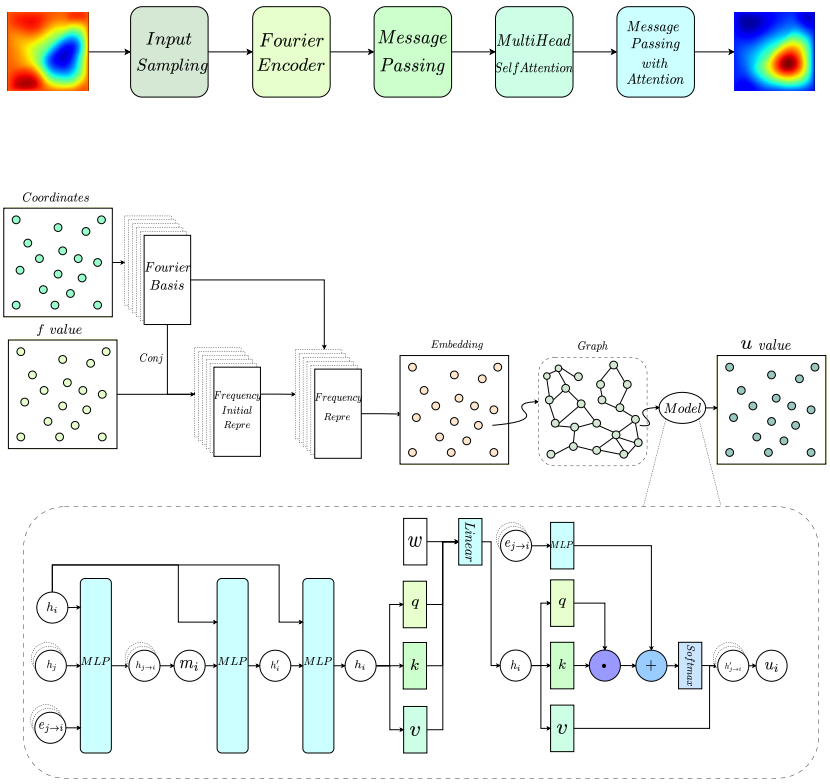

Figure 1: *GOLA: Graph-based Operator Learning with Attention.* The model first encodes input function values sampled on irregular spatial coordinates using a learnable Fourier encoder to obtain spectral node features. A graph is constructed based on spatial proximity, enabling message passing and multi-head self-attention to capture local and global dependencies. A final attention-based message passing layer refines the representation to predict the output solution values. GOLA effectively handles irregular domains and sparse samples, achieving strong generalization for PDE operator learning.

3.2 Graph Construction

To represent PDE solutions over irregular domains, we begin by randomly sampling a subset of points $\{x_i\}_{i=1}^{N}$ from a uniform grid in 2D space. We then construct a graph $G = (V, E)$ with nodes $V = \{x_i\}$ and edges E determined by a radius r. Edges are created based on spatial proximity. Two nodes are connected if the Euclidean distance between them is less than a threshold r such that $(i, j) \in E$ if and only if $\|x_i - x_j\|_2 \leq r$. Each edge (i, j) carries edge attributes e_{ij} that encode both geometric and feature-based information, such as the relative coordinates and function values at nodes i and j such that $e_{ij} = \|(x_i, x_j, f(x_i), f(x_j))$, where $\|$ is the concatenation operation. This graph-based representation allows us to model unstructured spatial domains and enables message passing among nonuniform samples.

3.3 FOURIER ENCODER

We define a set of learnable frequencies $\{\omega_m \in \mathbb{R}^2 \mid m = 1, \ldots, M\}$.

For any coordinate $x \in \mathbb{R}^2$, the m-th basis function is given by the complex exponential

$$\varphi_m(x) = e^{2\pi i \langle \omega_m, x \rangle} \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the standard Euclidean inner product, and i is the imaginary unit.

At the discrete level, for a batch of B samples and N points per sample, the basis matrix is defined as

$$\Phi \in \mathbb{C}^{B \times N \times M}, \quad \Phi_{b,i,m} = e^{2\pi i \langle \omega_m, x_i^{(b)} \rangle} \tag{3}$$

where $x_i^{(b)}$ denotes the i-th coordinate point in the b-th batch sample.

Given the input $f \in \mathbb{R}^{B \times C_{\text{in}} \times N}$ sampled at points $\{x_i\}$, we first project onto the Fourier basis. We compute the Fourier coefficients by

$$\hat{u}_{b,c,m} = \frac{1}{N} \sum_{i=1}^{N} f_{b,c,i} \, \overline{\varphi_m\left(x_i^{(b)}\right)} \tag{4}$$

where $\overline{(\cdot)}$ denotes complex conjugation.

We introduce a learnable set of complex Fourier coefficients $W \in \mathbb{C}^{C_{\text{in}} \times C_{\text{out}} \times M}$. The spectral filtering operation is

$$\hat{v}_{b,o,m} = \sum_{c=1}^{C_{\text{in}}} \hat{u}_{b,c,m} \, W_{c,o,m} \tag{5}$$

We reconstruct the output in the physical domain by applying the inverse transform

$$v_{b,o,i} = \sum_{m=1}^{M} \hat{v}_{b,o,m} \, \varphi_m\left(x_i^{(b)}\right) \tag{6}$$

Since v is complex-valued, we only take its real part for the output as $h = \text{Re}(v) \in \mathbb{R}^{B \times C_{\text{out}} \times N}$. The output h serves as the input node features for the downstream GNN model.

3.4 MESSAGE PASSING

Given a node $i \in V$ and its set of neighbors $\mathcal{N}(i)$, the pre-processed messages $\{m_{ij}\}_{j \in \mathcal{N}(i)}$ are first computed using a learnable neural network g_Θ as

$$m_{ij} = g_\Theta(h_i, h_j, e_{ij}) \tag{7}$$

where h_i and h_j are node features, and e_{ij} denotes edge attributes.

Then we aggregate message from neighbors such that

$$\hat{m} = \|\left(\frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} m_{ij}, \max_{j \in \mathcal{N}(i)} m_{ij}, \min_{j \in \mathcal{N}(i)} m_{ij}, \sqrt{\frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} (m_{ij} - \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} m_{ij})^2}\right) \tag{8}$$

This concatenated feature vector is processed by a post-aggregation neural network γ_Θ to produce the updated node representation by

$$h_i' = \gamma_\Theta\left(h_i, \hat{m}\right) \tag{9}$$

The updated node representation is passed through additional MLP layers with residual connections to enhance expressiveness.

3.5 MULTI-HEAD SELF-ATTENTION

We employ H independent attention heads. For each head h, the query, key and value functions are computed as linear projections

$$q^{(h)}(x) = W_q h'(x), \quad k^{(h)}(y) = W_k h'(y), \quad v^{(h)}(y) = W_v h'(y) \tag{10}$$

where $W_q, W_k, W_v \in \mathbb{R}^{d_h \times C_{\text{out}}}$, $q^{(h)}(x), k^{(h)}(y), v^{(h)}(y) \in \mathbb{R}^{d_h}$ are learned head-specific features, and d_h is the dimension per attention head.

Before computing attention, the keys and values are normalized

$$\tilde{k}^{(h)}(y) = \text{Norm}(k^{(h)}(y)), \quad \tilde{v}^{(h)}(y) = \text{Norm}(v^{(h)}(y)) \tag{11}$$

where $\text{Norm}(\cdot)$ denotes instance normalization.

We compute

$$G_h = \sum_{j=1}^{N} \tilde{k}^{(h)}(y_j)^\top \tilde{v}^{(h)}(y_j) w(y_j), \quad (\mathcal{K}_h h')(x_i) = q^{(h)}(x_i) G_h \tag{12}$$

The outputs are concatenated and projected to the output space by

$$(\mathcal{K}h')(x_i) = \Vert \left((\mathcal{K}_1 h')(x_i), \ldots, (\mathcal{K}_H h')(x_i) \right), \quad \hat{h}(x_i) = W_{\text{out}} (\mathcal{K}h')(x_i) \tag{13}$$

where where $G_h \in \mathbb{R}^{d_h \times d_h}$, w is calculated by the number of points, $W_{\text{out}} \in \mathbb{R}^{C_{\text{out}} \times (C_{\text{out}} \cdot H)}$.

The result is then passed through a linear projection layer to update the node features.

3.6 MESSAGE PASSING WITH ATTENTION

We update node features and add a skip connection by

$$\hat{h}_i' = W_1 \hat{h}_i + \sum_{j \in \mathcal{N}(i)} \alpha_{ij} \left(W_2 \hat{h}_j + W_3 e_{ij} \right), \quad \hat{h}_i' = \hat{h}_i' + W_s \hat{h}_i \tag{14}$$

The attention weights α_{ij} are computed using a scaled dot-product attention mechanism by

$$\alpha_{ij} = \text{softmax}_j \left(\frac{\left(W_4 \hat{h}_i \right)^\top \left(W_5 \hat{h}_j + W_3 e_{ij} \right)}{\sqrt{d}} \right) \tag{15}$$

where d is the dimensionality of the head, and the softmax is applied over the set of neighbors $j \in \mathcal{N}(i)$. Then we add a linear projection to produce the predicted solution $\hat{u}$.

3.7 TRAINING

The model is trained to minimize the relative L_2 error between predicted and true solutions by

$$\mathcal{L}_2(\theta) = \frac{\Vert u - \mathcal{G}_\theta(f) \Vert_{L^2(\Omega)}}{\Vert u \Vert_{L^2(\Omega)}} \tag{16}$$

4 THEORETICAL ANALYSIS

Following the universal approximation theorem for operators (Lu et al., 2019), neural operator architectures can approximate any continuous operator $\mathcal{G}$ between Banach spaces when provided with sufficient capacity.

Proposition. Let $\mathcal{G} : \mathcal{F} \to \mathcal{U}$ be a continuous nonlinear operator between separable Banach spaces. Then, under sufficient model capacity, the GOLA architecture $\mathcal{G}_\theta$ can approximate $\mathcal{G}$ arbitrarily well in the $L^2(\Omega)$ norm over a compact domain Ω, i.e., $\sup_{f \in \mathcal{F}_\delta} \|\mathcal{G}(f) - \mathcal{G}_\theta(f)\|_{L^2(\Omega)} < \epsilon$, for any $\epsilon > 0$ and compact subset $\mathcal{F}_\delta \subset \mathcal{F}$.

Proof. Given a function $f \in \mathcal{F} \subset L^2(\Omega)$, we sample it at N spatial locations $\{x_i\}_{i=1}^N \subset \Omega$ to obtain a discrete representation $f_N = (f(x_1), \ldots, f(x_N)) \in \mathbb{R}^N$. Since Ω is compact, by increasing N the point cloud $\{x_i\}$ becomes dense in Ω. Thus, f_N can approximate f arbitrarily well in $L^2(\Omega)$ norm via interpolation over the sampling set.

Define a set of complex Fourier basis functions $\{\phi_m(x) = e^{2\pi i \langle \omega_m, x \rangle}\}_{m=1}^M$. The Fourier basis is complete in $L^2(\Omega)$, so for any $f \in \mathcal{F}$ and $\delta > 0$, there exists M such that

$$\left\| f(x) - \sum_{m=1}^{M} \hat{f}_m \phi_m(x) \right\|_{L^2(\Omega)} < \delta.$$

This guarantees that the learnable Fourier encoder in GOLA can approximate the functional input f to arbitrary precision.

Construct a graph $G = (V, E)$ with node set $V = \{x_i\}_{i=1}^N$, where edges encode local spatial relationships. According to universal approximation results for GNNs (Xu et al., 2019), (Morris et al., 2019), for any continuous function defined on graphs, a GNN with sufficient depth and width can approximate it arbitrarily well. Thus, the GNN decoder can approximate the mapping from input features to solution values

$$(f(x_1), \ldots, f(x_N)) \mapsto (\mathcal{G}(f)(x_1), \ldots, \mathcal{G}(f)(x_N))$$

Let $\mathcal{T}_N$ denote the sampling operator, $\mathcal{F}_\theta$ the Fourier encoder, and $\mathcal{D}_\theta$ the GNN decoder. Then the GOLA operator can be written as

$$\mathcal{G}_\theta = \mathcal{D}_\theta \circ \mathcal{F}_\theta \circ \mathcal{T}_N$$

Each component is continuous and approximates its target arbitrarily well. Since composition of continuous approximations preserves continuity, and $\mathcal{F}_\delta$ is compact, the total approximation error can be made less than any $\varepsilon > 0$ by choosing N, M, and model capacity large enough such that

$$\sup_{f \in \mathcal{F}_\delta} \|\mathcal{G}(f) - \mathcal{G}_\theta(f)\|_{L^2(\Omega)} < \varepsilon$$

5 Experiments

We evaluate the proposed model GOLA on four 2D PDE benchmarks including Darcy Flow, Nonlinear Diffusion, Eikonal, and Advection. For each dataset, we simulate training data with 5, 10, 20, 30, 40, 50, 80, 100 samples and use 100 examples for testing. To construct graphs, we randomly sample 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000 points from a uniform 128×128 grid over the domain $[0, 1] \times [0, 1]$. The sampled points define the nodes of the graph. Our model learns to approximate the solution operator from these irregularly sampled inputs. We aim to test generalization under both limited data and resolution changes. We compare against the following baselines including DeepONet (Lu et al., 2019), AFNO (Guibas et al., 2021) and Graph Kernel Network (GKN) (Li et al., 2020b).

Comparisons with baselines. Table 1 reports the averaged test errors over 5 runs with different seeds across four PDE benchmarks—Darcy Flow, Advection, Eikonal, and Nonlinear Diffusion—in the low-data regime of 100 training samples with sample density $= 1000$ randomly selected from a uniform 128×128 grid over the domain $[0, 1] \times [0, 1]$. The proposed GOLA method consistently achieves the lowest error across all datasets. For Darcy Flow, GOLA attains an error of 0.1088 ± 0.0027, representing a 40.8% relative improvement over the best baseline, GKN (0.1840 ± 0.0040). In Advection, GOLA achieves 0.2227 ± 0.0185, reducing the error by 26.7% compared to GKN and by over 77% relative to AFNO and DeepONet. For Eikonal, GOLA obtains 0.0657 ± 0.0011, a 45.7% improvement over GKN, while Nonlinear Diffusion exhibits the largest relative gain—0.0430 ± 0.0005, which is 59.2% lower than GKN. Moreover, GOLA maintains standard deviations

on par with or below those of the best-performing baselines, indicating both superior accuracy and stable convergence.

Table 1: *Test errors for different models in irregular sampling points trained on 100 training data samples with sample density=1000 across various PDE benchmarks. The results are averaged over 5 runs in this paper.*

Dataset	AFNO	DeepONet	GKN	Ours(GOLA)
Darcy Flow	0.4310 ± 0.0040	0.5897 ± 0.0026	0.1840 ± 0.0040	$\mathbf{0.1088 \pm 0.0027}$
Advection	0.9845 ± 0.0007	0.9979 ± 0.0001	0.3043 ± 0.0041	$\mathbf{0.2227 \pm 0.0185}$
Eikonal	0.1828 ± 0.0017	0.1918 ± 0.0004	0.1210 ± 0.0043	$\mathbf{0.0657 \pm 0.0011}$
Nonlinear Diffusion	0.1686 ± 0.0016	0.2781 ± 0.0005	0.1052 ± 0.0038	$\mathbf{0.0430 \pm 0.0005}$

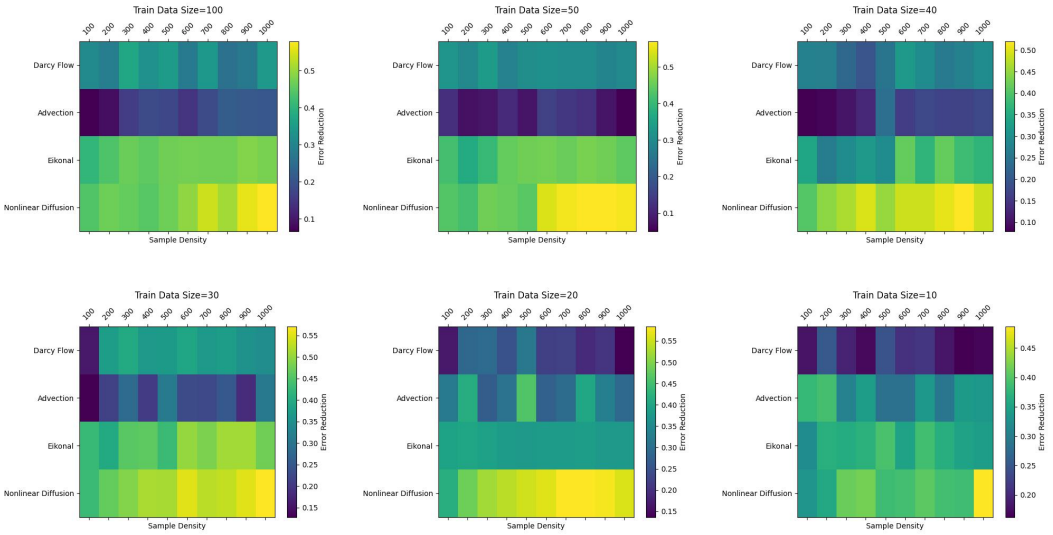

Figure 2: *Error reduction heatmaps across training data sizes and sample densities for PDE Benchmarks. Nonlinear Diffusion consistently shows the highest error reduction across all training sizes and densities and it becomes more prominent at high sample densities even under very small training size 10.*

Generalization across sample densities. From Table 2, we use 100 training data, and choose three types of sampling densities 20, 500, 1000 which represent small, medium and high sample densities. We observe a consistent trend that increasing sample density leads to significant performance improvements across all PDEs. The results highlight that higher sampling density substantially improves generalization, particularly for PDEs with more complex solution manifolds such as Darcy flow and nonlinear diffusion, and that even moderate densities 500 are sufficient to close much of the performance gap for Eikonal equations.

Table 2: *Test errors for small, medium, and high sampling densities with training data size=100.*

Sample Density	20	500	1000
Darcy flow	0.4422 ± 0.0213	0.1298 ± 0.0043	0.1088 ± 0.0027
Advection	0.4374 ± 0.0177	0.2654 ± 0.0163	0.2227 ± 0.0185
Eikonal	0.1267 ± 0.0019	0.0675 ± 0.0020	0.0657 ± 0.0011
Nonlinear diffusion	0.1901 ± 0.0060	0.0542 ± 0.0015	0.0430 ± 0.0005

Resolution generalization. From Table 3 and Figure 3, we use 100 training data and sample 1000 training sample points, then we test the relative L_2 error in different test sample densities 100, 500, 1000, 2000, 4000. We observe that higher test sample densities consistently reduce the error for all PDE families, reflecting improved approximation accuracy with denser test points.

Table 3: *Test errors for different test sampling densities with training sample density=1000.*

Test Sample Density	100	500	1000	2000	4000
Darcy flow	0.2475 ± 0.0041	0.1304 ± 0.0020	0.1088 ± 0.0027	0.0971 ± 0.0033	0.0895 ± 0.0035
Advection	0.3641 ± 0.0117	0.2505 ± 0.0149	0.2227 ± 0.0185	0.2218 ± 0.0202	0.2182 ± 0.0141
Eikonal	0.0790 ± 0.0031	0.0672 ± 0.0020	0.0657 ± 0.0011	0.0654 ± 0.0024	0.0654 ± 0.0019
Nonlinear diffusion	0.0893 ± 0.0020	0.0511 ± 0.0015	0.0430 ± 0.0005	0.0386 ± 0.0012	0.0368 ± 0.0015

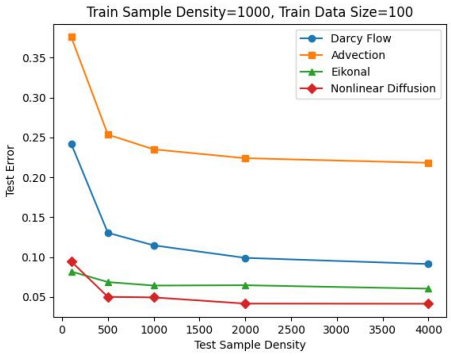

Figure 3: *Test error trend with test sample density*

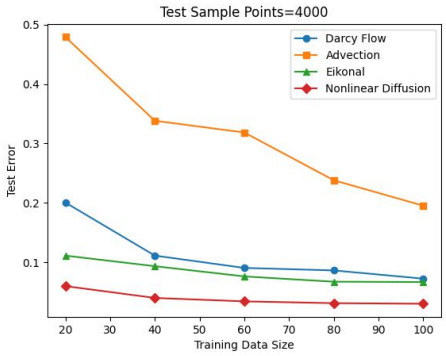

Figure 4: *Test error trend with train data size*

Data Efficiency. From Table 4, we use 2000 sample points and change different training data size to test the performance. From Figure 4, we report the results for 4000 sample points with different training data size. In Figure 5, we report the results for test error trend with respect to training data size in test sample points $\in \{200, 300, 400, 500, 600, 700, 800, 900\}$. Across all PDEs, we observe a clear trend of decreasing test error with increasing training data size, indicating effective data scaling behavior.

Table 4: *Test errors under varying numbers of training data size with sample density=2000.*

Training data size	20	40	60	80	100
Darcy flow	0.2027 ± 0.0161	0.1372 ± 0.0095	0.1071 ± 0.0073	0.0983 ± 0.0057	0.0913 ± 0.0029
Advection	0.5253 ± 0.0273	0.4026 ± 0.0182	0.3192 ± 0.0388	0.2709 ± 0.0243	0.2228 ± 0.0172
Eikonal	0.1029 ± 0.0047	0.0763 ± 0.0033	0.0678 ± 0.0028	0.0648 ± 0.0023	0.0647 ± 0.0021
Nonlinear diffusion	0.0815 ± 0.0139	0.0538 ± 0.0023	0.0429 ± 0.0036	0.0394 ± 0.0033	0.0360 ± 0.0013

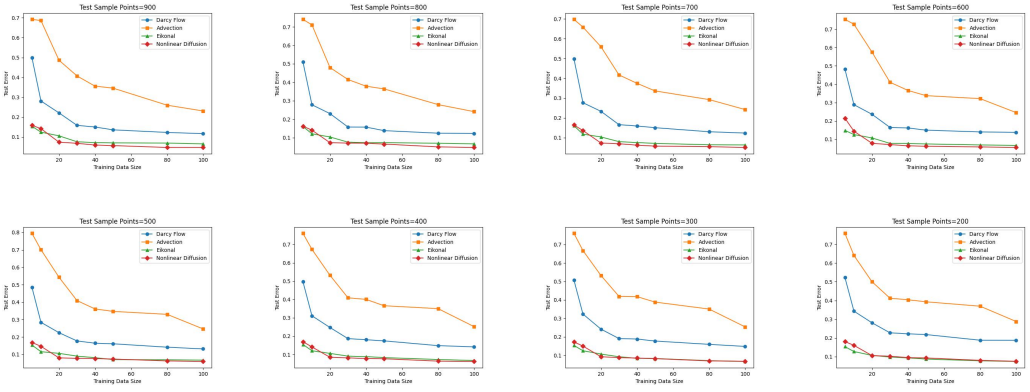

Figure 5: *Test error trends across varying sample densities for PDE benchmarks.*

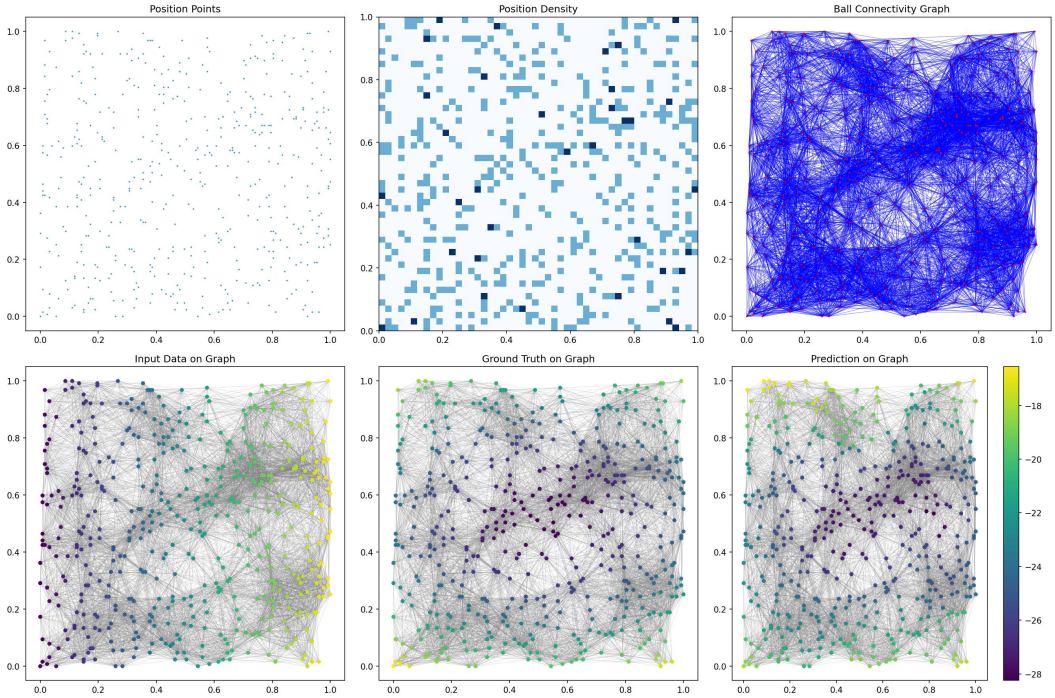

Figure 6: *Visualizations for graph with 1000 sample points on Advection.*

Graph Visualizations. We visualize graph construction in Figure 6. We randomly sample 1000 node positions in the unit square and use ball connectivity with a fixed radius 0.2 to construct graph. These results are shown on the top row. Then in this graph, we visualize input function values on the graph, ground-truth solution, and model prediction on the bottom row. Figure 6 demonstrates that (i) the graph construction preserves locality and global connectivity; (ii) the learned model generalizes well to unseen node configurations and accurately reconstructs the solution field; (iii) visual comparison between ground truth and predictions reveals minimal discrepancy, supporting the effectiveness of our proposed model GOLA.

Time Complexity and Memory Cost. We analyze the computational complexity of the GOLA architecture in terms of the number of spatial points N, Fourier modes M, feature channels C, and edges $E \sim \mathcal{O}(Nk)$, where k is the average number of neighbors in the sparse spatial graph. The time complexity for GOLA is $\mathcal{O}(MNC) + \mathcal{O}(NkC^2) + \mathcal{O}(NkC)$. The count of parameters for GOLA is 2,900,249.

6 Conclusion

In this work, We introduce **G**raph-based **O**perator **L**earning with **A**ttention (GOLA) framework, which combines a learnable Fourier encoder with attention-enhanced message passing to solve PDEs over irregular domains. By representing the spatial domain as a proximity graph and embedding inputs into a learnable spectral basis, GOLA effectively captures both local and global dependencies, enabling accurate operator approximation even under sparse sampling and complex geometries. Through comprehensive experiments across diverse PDE benchmarks including Darcy Flow, Advection, Eikonal, and Nonlinear Diffusion, GOLA consistently outperforms baselines including AFNO, DeepONet, GKN particularly in data-scarce regimes. We demonstrate GOLA's superior generalization, resolution scalability, and robustness to sparse sampling. These results highlight the potential of combining spectral encoding and localized message passing with attention to build continuous, data-efficient operator approximators that adapt naturally to non-Euclidean geometries. This study demonstrates that graph-based representations provide a powerful and flexible foundation for advancing operator learning in real-world physical systems with irregular data.

REFERENCES

Bar-Sinai, Y. et al. (2019). Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349.

Batlle, P., Darcy, M., Hosseini, B., and Owhadi, H. (2023). Kernel methods are competitive for operator learning. *arXiv preprint arXiv:2304.13202*.

Battaglia, P. W. et al. (2018). Relational inductive biases, deep learning, and graph networks. *Nature*, 557(7707):528–536.

Boullé, N. and Townsend, A. (2024). A mathematical guide to operator learning. *Handbook of Numerical Analysis*, 25:83–125.

Brandstetter, J. et al. (2022). Message passing neural pde solver. In *International Conference on Learning Representations (ICLR)*.

Cuomo, S., Di Cola, V., Giampaolo, F., Rozza, G., Raissi, M., and Piccialli, F. (2022). Physics-informed machine learning: A survey on problems, methods, and applications. *ACM Computing Surveys (CSUR)*, 55(1):1–38.

Dos Santos, F., Akhound-Sadegh, T., and Ravanbakhsh, S. (2023). Physics-informed transformer networks. In *The Symbiosis of Deep Learning and Differential Equations III*.

Guibas, J., Mardani, M., Li, Z., Tao, A., Anandkumar, A., and Catanzaro, B. (2021). Adaptive fourier neural operators: Efficient token mixers for transformers. *arXiv preprint arXiv:2111.13587*.

Jin, Z. and Gu, G. X. (2023). Leveraging graph neural networks and neural operator techniques for high-fidelity mesh-based physics simulations. *AIP Advances*, 13(4):046109.

Kissas, G., Seidman, J., Guilhoto, L. F., Preciado, V. M., Pappas, G. J., and Perdikaris, P. (2022). Learning operators with coupled attention. *Journal of Machine Learning Research*, 23(152):1–63.

Kovachki, N. B., Lanthaler, S., and Stuart, A. M. (2024). Operator learning: Algorithms and analysis. *arXiv preprint arXiv:2402.15715*.

Lam, R. et al. (2024). Graphcast: Learning skillful medium-range global weather forecasting. *Science*, 383(6674):346–351.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. (2020a). Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. (2020b). Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*.

Liu, Y., Lütjens, B., Azizzadenesheli, K., and Anandkumar, A. (2022). U-netformer: A u-net style transformer for solving pdes. *arXiv preprint arXiv:2206.11832*.

Lu, L., Jin, P., and Karniadakis, G. E. (2019). Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. (2019). Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609.

Nelsen, N. H. and Stuart, A. M. (2024). Operator learning using random features: A tool for scientific computing. *arXiv preprint arXiv:2408.06526*.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. (2020). Learning mesh-based simulation with graph networks. In *International Conference on Machine Learning (ICML)*.

Wang, L. et al. (2022). Pde-gcn: Learning pdes on graphs. In *International Conference on Machine Learning (ICML)*.

Xiao, Z., Hao, Z., Lin, B., Deng, Z., and Su, H. (2024). Improved operator learning by orthogonal attention. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F., editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 54288–54299. PMLR.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019). How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*.

Xu, M., Han, J., Lou, A., Kossaifi, J., Ramanathan, A., Azizzadenesheli, K., Leskovec, J., Ermon, S., and Anandkumar, A. (2024). Equivariant graph neural operator for modeling 3d dynamics. *arXiv preprint arXiv:2401.11037*.

Zhou, Q., Sun, L., and Lin, Z. (2023). Tokenformer: A token mixing architecture for solving pdes. *arXiv preprint arXiv:2310.02271*.