Appendix

A Anonymous Code

Our source code can be found `https://anonymous.4open.science/r/BENO-3D53`, which is an anonymous link for reproducibility.

B Derivation of the Green's function method.

We first review the definition of the Green's function, which is $G : \Omega \times \Omega \to \mathbb{R}$, which is the solution of the corresponding equation as follows:

$$\begin{cases} \nabla^2 G = \delta(x - x_0)\delta(y - y_0) \\ G|_{\partial\Omega} = 0 \end{cases} \tag{10}$$

According to Green's identities,

$$\iint_\Omega (u\nabla^2 G) d\sigma = \int_{\partial\Omega} u \frac{\partial G}{\partial n} dl - \iint_\Omega (\nabla u \cdot \nabla G) d\sigma \tag{11}$$

Since u and G are arbitrary, we can change the position to obtain that,

$$\iint_\Omega (G\nabla^2 u) d\sigma = \int_{\partial\Omega} G \frac{\partial u}{\partial n} dl - \iint_\Omega (\nabla u \cdot \nabla G) d\sigma \tag{12}$$

Subtract Eq. 12 from Eq. 11, we have,

$$\iint_\Omega (u\nabla^2 G - G\nabla^2 u) d\sigma = \int_{\partial\Omega} \left(u\frac{\partial G}{\partial n} - G\frac{\partial u}{\partial n} \right) dl \tag{13}$$

Substitute Eq. 13 into Eq. 10, we can have that,

$$\begin{aligned} \int_{\partial\Omega} \left(u\frac{\partial G}{\partial n} - G\frac{\partial u}{\partial n} \right) dl &= \iint_\Omega (u \cdot \nabla^2 G - G \cdot \nabla^2 u) d\sigma \\ &= \iint_\Omega (-u\delta(x - x_0)\delta(y - y_0) - G\nabla^2 u) d\sigma \\ &= -u(x_0, y_0) - \iint_\Omega G\nabla^2 u d\sigma \\ &= -u(x_0, y_0) + \iint_\Omega G f(x, y) d\sigma \end{aligned} \tag{14}$$

Namely, we have that,

$$\begin{aligned} u(x, y) = &\iint_\Omega G(x, y, x_0, y_0) f(x_0, y_0) d\sigma_0 \\ &+ \int_{\partial\Omega} \left[G(x, y, x_0, y_0) \frac{\partial u(x_0, y_0)}{\partial n_0} - u(x_0, y_0) \frac{\partial G(x, y, x_0, y_0)}{\partial n_0} \right] dl_0 \end{aligned} \tag{15}$$

When considering the Dirichlet boundary conditions, we can simplify the solution in the following form:

$$u(x, y) = \iint_\Omega G(x, y, x_0, y_0) f(x_0, y_0) d\sigma_0 - \int_{\partial\Omega} g(x_0, y_0) \frac{\partial G(x, y, x_0, y_0)}{\partial n_0} dl_0 \tag{16}$$

C Numerical solution of the elliptic PDE

The strong solution to equation 1 can be expressed in terms of the Green's function (see Section 3.1 and Appendix B for discussion). However, obtaining a closed form expression using the Green's function is typically not possible, except for some limited canonical domain shapes. In the present paper, we obtain the solution to equation 1 in arbitrary two dimensional domains Ω using the finite volume method (Hirsch, 2007). This numerical approach relies on discretizing the domain Ω using *cells*. The surfaces of these cells at the boundary, which are called *cell interfaces*, are used to specify the given boundary condition. The solution of equation 1 is then numerically approximated over N (e.g., for 32×32 cells, $N = 1024$) computational cells by solving,

$$\mathbf{P}\hat{\mathbf{u}} = \mathbf{f}, \tag{17}$$

where $\mathbf{P} \in \mathbb{R}^{N\times N}$ is an $N \times N$ matrix which denotes a second-order discretization of the ∇^2 operator incorporating the boundary conditions, $\hat{\mathbf{u}} \in \mathbb{R}^{N\times 1}$ is a vector of values at the cell centers, and $\mathbf{f} \in \mathbb{R}^{N\times 1}$ is a vector with values $f(\cdot,\cdot)$ at cell centers used to discretize the domain Ω. The matrix $\mathbf{P}$ resulting from this approach is positive definite and diagonally dominant, making it convenient to solve Equation 17 with a matrix-free iterative approach such as the Gauss-Seidel method (Saad, 2003).

D Details of Datasets

In this paper, we have established a comprehensive dataset for solving elliptic PDEs to facilitate various research endeavors. The elliptic PDEs solver is performed as follows. (1) A square domain is set with N_c number of cells in both x and y directions (note $N = N_c^2$). The number of corners is set, however, the size of the corners is chosen randomly. (2) The source term $f(x, y)$ is assigned assuming a variety of basis functions, including sinusoidal, exponential, logarithmic, and polynomial distributions. (3) The values of the boundary conditions $g(x, y)$ are set using continuous periodic functions with a uniformly distributed wavelength $\in [1, 5]$. (4) The Gauss-Seidel method (Saad, 2003) is used to iteratively obtain the solution $u(x, y)$. Each Poisson run generates two files: one for the interior cells with discrete values of x, y, f, and u and the other for the boundary interfaces with discrete values of x, y, and g. The simulations are performed on the Sherlock cluster at Stanford University.

E More Implementation Details

Our normalization process is performed using the z-score method (Patro & Sahu, 2015), where the mean and standard deviation are calculated from the training set. This ensures that all features are normalized based on the mean and variance of the training data. We also apply the CosineAnnealingWarmRestarts scheduler (Loshchilov & Hutter, 2016) during the training. Each experiment is trained for 1000 epochs, and validation is performed after each epoch. For the final evaluation, we select the model parameters from the epoch with the lowest validation loss. Consistency is maintained across all experiments by utilizing the same random seed.

All our experiments are evaluated on relative L2 error, abbreviated as L2, and mean absolute error (MAE), which are two commonly used metrics for evaluating the performance of models or algorithms. The relative L2 error, also known as the normalized L2 error, measures the difference between the predicted values and the ground truth values, normalized by the magnitude of the ground truth values. It is typically calculated as the L2 norm of the difference between the predicted and ground truth values, divided by the L2 norm of the ground truth values. On the other hand, MAE measures the average absolute difference between the predicted values and the ground truth values. It is calculated by taking the mean of the absolute differences between each predicted value and its corresponding ground truth value.

F Details of Baselines

Our proposed BENO is compared with a range of competing baselines as follows:

- **GKN** (Li et al., 2020b) develops an approximation method for mapping in infinite-dimensional spaces by combining non-linear activation functions with a set of integral operators. The integration of kernels is achieved through message passing on graph networks. For fair comparison, we re-implement it by adding the boundary nodes to the graph. To better distinguish between nodes belonging to the interior and those belonging to the boundary, we have also added an additional column of one-hot encoding to the nodes for differentiation.

- **FNO** (Li et al., 2020a) introduces a novel approach that directly learns the mapping from functional parametric dependencies to the solution. The method implements a series of layers computing global convolution operators with the fast Fourier transform (FFT) followed by mixing weights in the frequency domain and inverse Fourier transform, enabling an architecture that is both expressive and computationally efficient. For fair comparison, we re-implement it by fixing the value of out-domain nodes with a large number, and then implement the global operation.
- **GNN-PDE** (Lötzsch et al., 2022) represents the pioneering effort in training neural networks on simulated data generated by a finite element solver, encompassing various boundary shapes. It evaluates the generalization capability of the trained operator across previously unobserved scenarios by designing a versatile solution operator using spectral graph convolutions. For fair comparison, we re-implement it by adding the boundary nodes to the graph. To better distinguish between nodes belonging to the interior and those belonging to the boundary, we have also added an additional column of one-hot encoding to the nodes for differentiation.
- **MP-PDE** (Brandstetter et al., 2022) presents a groundbreaking solver that utilizes neural message passing for all its components. This approach replaces traditionally heuristic-designed elements in the computation graph with neural function approximators that are optimized through backpropagation. For fair comparison, we re-implement it by adding the boundary nodes to the graph. To better distinguish between nodes belonging to the interior and those belonging to the boundary, we have also added an additional column of one-hot encoding to the nodes for differentiation.

G Differences with other Neural Operators

In this section, we compare our method, BENO, with existing approaches in terms of several key aspects according to Table 1.

- PDE-agnostic prediction on new initial conditions: GKN, FNO, GNN-PDE, MP-PDE, and BENO are all capable of predicting new initial conditions.
- Train/Test space grid independence: GKN, GNN-PDE, and BENO exhibit independence between the training and testing spaces, while FNO and MP-PDE lack this independence.
- Evaluation at unobserved spatial locations: GKN, FNO, and BENO are capable of evaluating the PDE at locations that are not observed during training, while GNN-PDE and MP-PDE do not possess this capability.
- Free-form spatial domain for boundary shape: Only GNN-PDE and BENO are capable of dealing with arbitrary boundary shapes, while GKN and MP-PDE are limited in this aspect.
- Inhomogeneous boundary condition value: Only our method, BENO, has the ability to handle inhomogeneous boundary conditions, while GKN, FNO, GNN-PDE, and MP-PDE are unable to handle them.

In summary, compared to the existing methods, our method, BENO, possesses several distinct advantages. It can predict new initial conditions regardless of the specific PDE, maintains grid independence between training and testing spaces, allows evaluation at unobserved spatial locations, handles free-form spatial domains for boundary shapes, and accommodates inhomogeneous boundary condition values. These capabilities make BENO a versatile and powerful approach for solving time-independent elliptic PDEs.

H Algorithm

The whole learning algorithm of BENO is summarized in Algorithm 1.

I More Experimental Results

I.1 Sensitivity Analysis

In this section, we discuss the process of determining the optimal values for the number of MLP layers (M) and the number of Transformer layers (L) using grid search, a systematic approach for hyper-parameter tuning.

Algorithm 1 Learning Algorithm of the proposed BENO

Input: The forcing term f, the inhomogeneous boundary condition g on $\partial\Omega$.
Output: The solution prediction $\hat{u}$ of the elliptic PDEs.

```
 1: Construct the graph $\mathcal{G} = \{(\mathcal{V}, \mathcal{E})\}$ following Section 3.2;
 2: Initialize the parameters in our model;
    # Training procedure
 3: while not convergence do
 4:     for each training input do
 5:         Set the boundary value of one branch to zero following Eq. 16;
 6:         Set the source term of interior in the other branch to zero;
 7:         Feed the node/edge attributes to encoder following Section 3.3.1.;
 8:         Feed the boundary to the Transformer for boundary features $\mathcal{B}$;
 9:         Add $\mathcal{B}$ to the message passing processor following Eq. 8;
10:         Feed output features into a decoder to get the predictions $\hat{u}$;
11:         Calculate the loss using MSE;
12:         Update the parameters in BENO using back propagation;
13:     end for
14: end while
```

Grid search involves defining a parameter grid consisting of different combinations of M and L values. We specified M in the range of [2, 3, 4] and L in the range of [1, 2, 3] to explore a diverse set of configurations. We build multiple models, each with a different combination of M and L values, and train them on 4-Corners training dataset. The models are then evaluated using appropriate evaluation metrics on a separate validation set. The evaluation results allowed us to compare the performance of models across different parameter combinations.

After evaluating the models, we select the combination of M and L that yield the best performance according to our chosen evaluation metric. This combination became our final choice for M and L, representing the optimal configuration for our model. To ensure the reliability of our chosen parameters, we validate them on an independent validation set. This step confirmed that the model's performance remained consistent and reliable.

The grid search process provided a systematic and effective approach to determine the optimal values for $M = 3$ and $L = 1$, allowing us to fine-tune our model and achieve improved performance.

I.2 More Experimental Results

We have successfully validated our method's performance on the 4-Corners and mixed corners datasets during training and testing on other shape datasets, yielding favorable results. In this section, we will further supplement the evaluation by training on the No Corner dataset and testing on other shape datasets. Since the No Corner dataset does not include any corner scenarios, the remaining datasets present completely unseen scenarios for it, thereby providing a stronger test of the model's generalization performance.

Table 7 summarizes the results of training on 900 No-Corner samples and tested on all datasets. We can infer similar conclusions to those in the experimental section above. Our BENO performs well in learning on No-Corner cases, yielding more accurate solutions. Additionally, our method demonstrates stronger generalization ability, as it can obtain good solutions even in cases where corners of any shape have not been encountered.

I.3 Convergence Analysis

We draw the training curve of the train L2 norm and test L2 norm of three models trained on the 4-Corners dataset with inhomogeneous boundary value in Figure 5. It is obviously that although two baselines also contains the boundary information, they fail to learn the elliptic PDEs with non-decreasing convergence curves. However, our proposed BENO is capable of successfully learning complex boundary conditions with the use of the CosineAnnealingWarmRestarts scheduler, converges to a satisfactory result.

Table 7: Performances of our proposed BENO and the compared baselines, which are trained on 900 No-Corner samples and tested on 5 datasets under relative L2 Norm and MAE separately. The unit of the MAE metric is 1×10^{-3}.

Train on No-Corner dataset										
Test set	4-Corners		3-Corners		2-Corners		1-Corner		No-Corner	
Metric	L2	MAE	L2	MAE	L2	MAE	L2	MAE	L2	MAE
GKN	1.0147± 0.1128	3.3790± 0.8922	1.0179± 0.1212	3.5419± 0.8643	1.0047± 0.1166	3.4530± 0.9514	1.0072± 0.1098	3.2295± 0.8520	1.0028± 0.1060	3.6899± 0.8987
FNO	0.9714± 0.0128	3.3210± 0.6546	0.9745± 0.0175	3.3187± 0.6639	0.9733± 0.0137	3.3319± 0.6298	0.9789± 0.0210	3.3511± 0.6109	0.9755± 0.0121	3.3427± 0.6981
GNN-PDE	0.9988± 0.0051	3.1182± 0.8543	0.9997± 0.0054	3.2748± 0.8902	0.9994± 0.0054	3.3475± 0.8533	1.0002± 0.0056	3.1447± 0.8559	0.9998± 0.0056	3.7518± 1.0314
MP-PDE	1.0029± 0.0808	3.1005± 0.8158	1.0049± 0.0891	3.2488± 0.7941	0.9986± 0.0822	3.2902± 0.8651	0.9855± 0.0769	3.0356± 0.8133	0.9917± 0.07670	3.6648± 0.8949
BENO (ours)	**0.6870± 0.2038**	**1.8830± 0.6083**	**0.6036± 0.1940**	**1.7293± 0.5844**	**0.5760± 0.1998**	**1.6703± 0.6605**	**0.6192± 0.2259**	**1.6749± 0.5773**	**0.4093± 0.1873**	**1.2505± 0.5752**

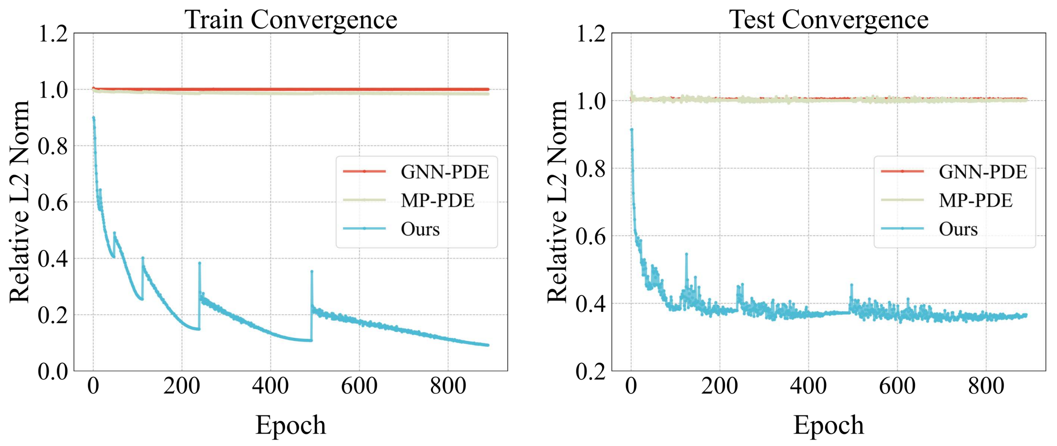

Figure 5: Visualization of the convergence curve of our BENO and two baselines.

J Limitations & Broader Impacts

Limitations. Although this paper primarily focuses on Dirichlet boundary conditions, it is essential to acknowledge that there are other types of boundary treatments, including Neumann and Robin boundary conditions. While the framework presented in this study may not directly address these alternative boundary conditions, it still retains its usefulness. Future research should explore the extension of the developed framework to incorporate these different boundary treatments, allowing for a more comprehensive and versatile solution for a broader range of practical problems.

Broader Impact. The development of a fast, efficient, and accurate neural network for solving PDEs holds significant potential for numerous physics and engineering disciplines. The impact of such advancements cannot be understated. By providing a more streamlined and computationally efficient approach, this research can revolutionize fields such as computational fluid dynamics, solid mechanics, electromagnetics, and many others. The ability to solve PDEs more efficiently opens up new possibilities for modeling and simulating complex physical systems, leading to improved designs, optimizations, and decision-making processes. The resulting advancements can have far-reaching implications, including the development of more efficient and sustainable technologies, enhanced understanding of natural phenomena, and improved safety and reliability in engineering applications. It is crucial to continue exploring and refining these neural network-based approaches to maximize their potential impact across a wide range of scientific and engineering disciplines.

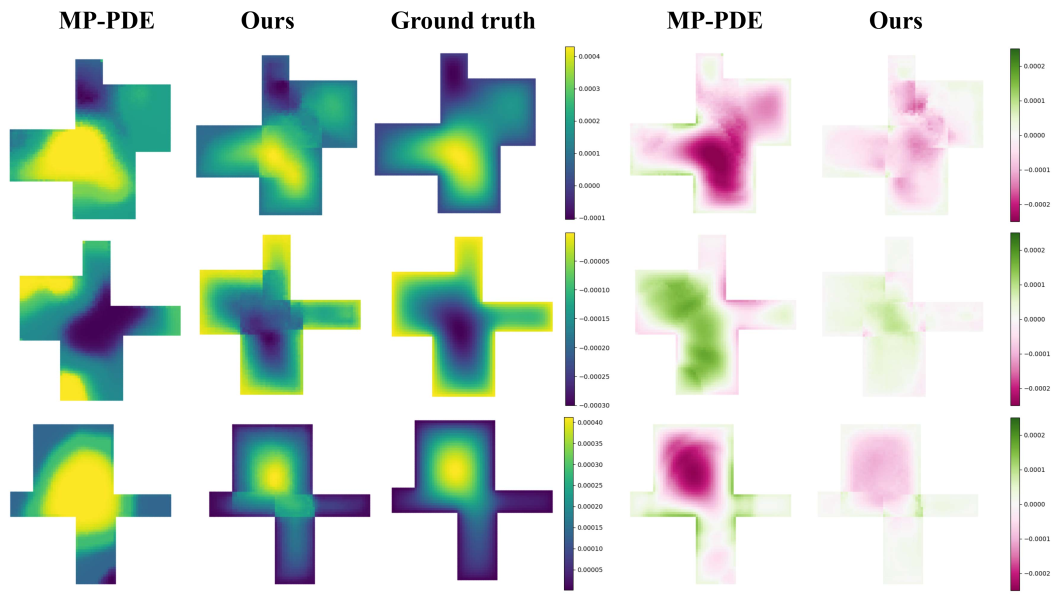

Figure 6: Visualization of prediction and prediction error on 64 $\times$ 64 grid resolution.

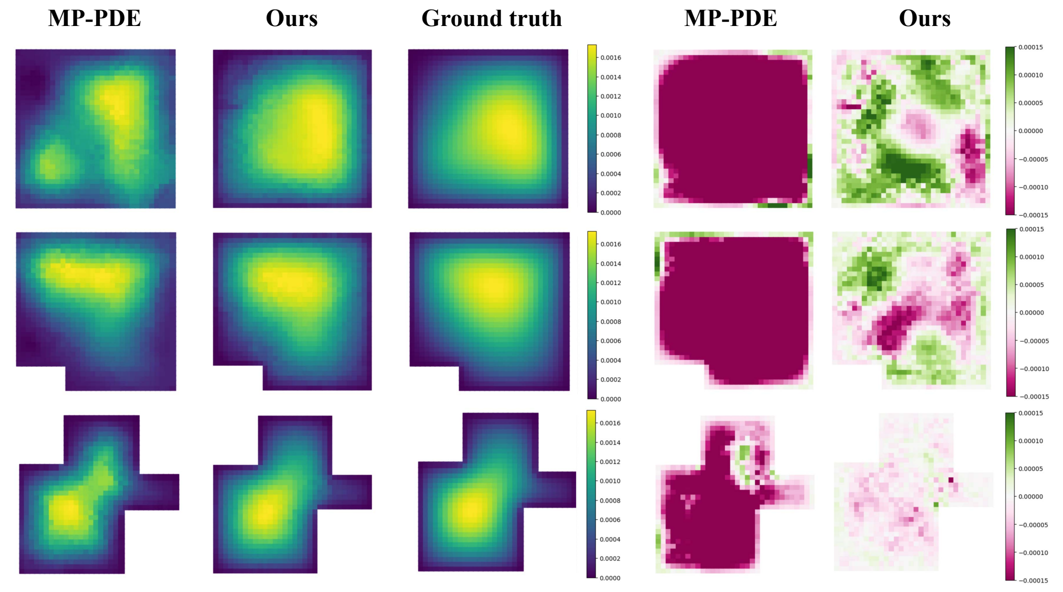

Figure 7: Visualization of prediction and prediction error on data with zero boundary value.

K More Visualization Analysis

In this section, we visualize the experimental results on a broader range of experiments. Figure 6 presents the comparison of solution prediction on 64 $\times$ 64 grid resolution. Figure 7 presents the comparison of solution prediction on data with zero boundary value. Figure 8 presents the qualitative results of training on the 4-Corners dataset and testing on data with various other shapes.

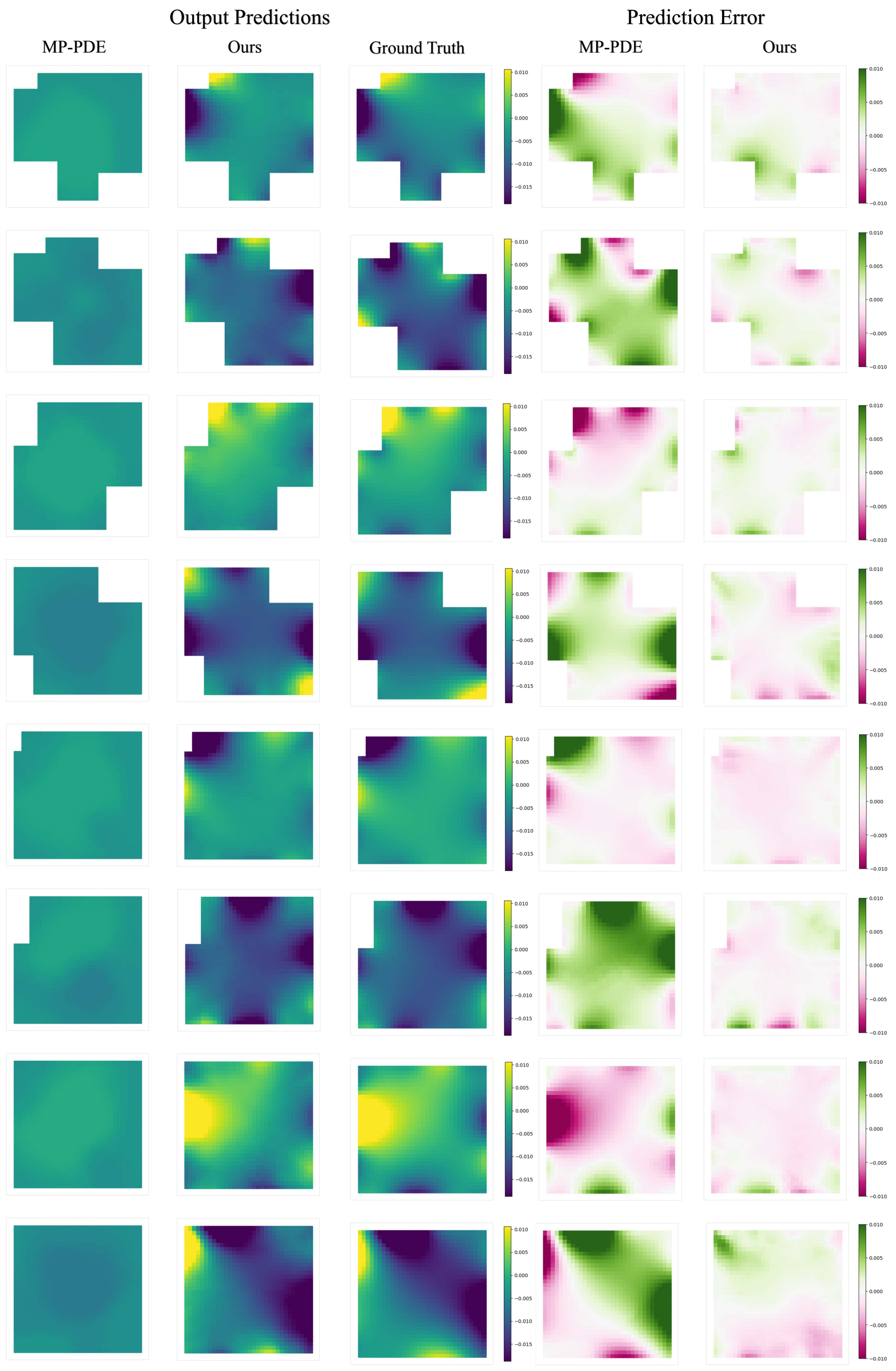

Figure 8: Visualization of prediction and prediction error from 3/2/1/No-Corner dataset, and each has two samples. We render the solution u of the baseline MP-PDE, our BENO and the ground truth in Ω.

Table 8: Performances of our proposed BENO and the compared baselines under Neumann boundary condition, which are trained on 900 4-corners samples and tested on 5 datasets under relative L2 norm and MAE separately. The unit of the MAE metric is 1×10^{-3}. Bold fonts indicate the best.

Train on 4-Corners dataset										
Test set	4-Corners		3-Corners		2-Corners		1-Corner		No-Corner	
Metric	L2	MAE	L2	MAE	L2	MAE	L2	MAE	L2	MAE
GKN	1.0118±0.1031	3.7105±1.0699	1.0046±0.1284	3.6013±1.2937	1.0301±0.1417	3.3880±1.0127	1.0025±0.1326	3.3675±0.9112	0.9827±0.1270	3.5691±1.2194
FNO	1.0547±0.1643	4.0316±0.8953	1.0587±0.1761	4.0219±0.8210	1.0519±0.1822	4.0308±0.8369	1.0533±0.1782	4.0276±0.8554	1.0549±0.1842	4.0417±0.8063
GNN-PDE	1.0105±0.0898	2.3685±0.6933	0.9907±0.1054	2.5474±0.8863	1.0132±0.1208	2.7348±0.8461	0.9821±0.1225	2.9824±0.8106	0.9711±0.1071	3.4930±1.1110
MP-PDE	1.0070±0.0813	2.3595±0.6941	0.9895±0.0973	2.5480±0.8955	1.0134±0.1120	2.7345±0.8393	0.9782±0.1240	2.9679±0.7958	0.9670±0.1164	3.4807±1.1143
BENO (ours)	**0.3568±0.0988**	**0.8311±0.2864**	**0.4201±0.1170**	**1.0814±0.3938**	**0.5020±0.1648**	**1.3918±0.5454**	**0.5074±0.1422**	**1.5676±0.4815**	**0.5221±0.1474**	**1.8649±0.5472**

Table 9: Performances of our proposed BENO and the compared baselines under Neumann boundary condition, which are trained on 900 mixed samples (180 samples each from 5 datasets) and tested on 5 datasets under relative L2 norm and MAE separately. The unit of the MAE metric is 1×10^{-3}. Bold fonts indicate the best.

Train on 4-Corners dataset										
Test set	4-Corners		3-Corners		2-Corners		1-Corner		No-Corner	
Metric	L2	MAE	L2	MAE	L2	MAE	L2	MAE	L2	MAE
GKN	1.0578±0.1859	3.8992±1.2048	1.0399±0.2116	3.6554±1.2172	1.0975±0.1912	3.5980±0.9631	1.0649±0.3096	3.6030±0.9310	1.0373±0.2210	3.7000±1.0831
FNO	1.0426±0.0917	3.5817±0.8212	1.0408±0.0925	3.6187±0.8338	1.0548±0.1239	3.6338±0.8042	1.0592±0.1065	3.6531±0.8448	1.0575±0.1019	3.6498±0.8239
GNN-PDE	0.9999±0.0008	2.3648±0.7703	1.0000±0.0010	2.6404±0.9250	0.9999±0.0010	2.7425±0.9808	1.0000±0.0010	3.1458±0.9672	1.0001±0.0010	3.7167±1.3370
MP-PDE	1.0245±0.1048	2.3973±0.7015	0.9989±0.1277	2.5510±0.8717	1.0277±0.1399	2.7722±0.8091	0.9940±0.1543	2.9998±0.7781	0.9731±0.1414	3.4930±1.0867
BENO (ours)	**0.4237±0.1237**	**1.0114±0.4165**	**0.3970±0.1277**	**1.0378±0.4221**	**0.3931±0.1347**	**1.0881±0.3993**	**0.3387±0.1279**	**1.0520±0.4253**	**0.3344±0.1171**	**1.2261±0.4467**

L EXPERIMENTS ON NEUMANN BOUNDARY

In this section, we consider to solve the Poisson equation with Neumann boundary conditions using our proposed BENO. In the context of Neumann boundary conditions, the equation takes the form:

$$\begin{aligned} \nabla^2 u(x,y) &= f(x,y), \quad \forall (x,y) \in \Omega, \\ \frac{\partial u(x,y)}{\partial n} &= g(x,y), \quad \forall (x,y) \in \partial\Omega, \end{aligned} \tag{18}$$

where f represents the source term, n typically represents the unit normal vector perpendicular to the boundary surface, and g specifies the prescribed rate of change normal to the boundary $\partial\Omega$. The challenge in solving Poisson's equation with Neumann boundary conditions lies in the proper treatment of the boundary derivative term, which requires sophisticated numerical schemes to approximate accurately.

Specifically, the model is trained exclusively on a dataset consisting of 900 4-corners samples. The robustness and generalizability of our approach were then evaluated on 5 different test datasets, which represent various boundary configurations encountered in practical applications. Each dataset is constructed to challenge the model with different boundary complexities.

The results are shown in Table 8 and Table 9. Our proposed BENO still demonstrates superior performance across all test datasets in comparison to the baselines, including GNN-PDE, and MP-PDE models. Particularly, BENO achieves the lowest MAE and relative L2 norm scores in the majority of the scenarios. In Table 8, when tested on the 4-Corners dataset, BENO exhibits an

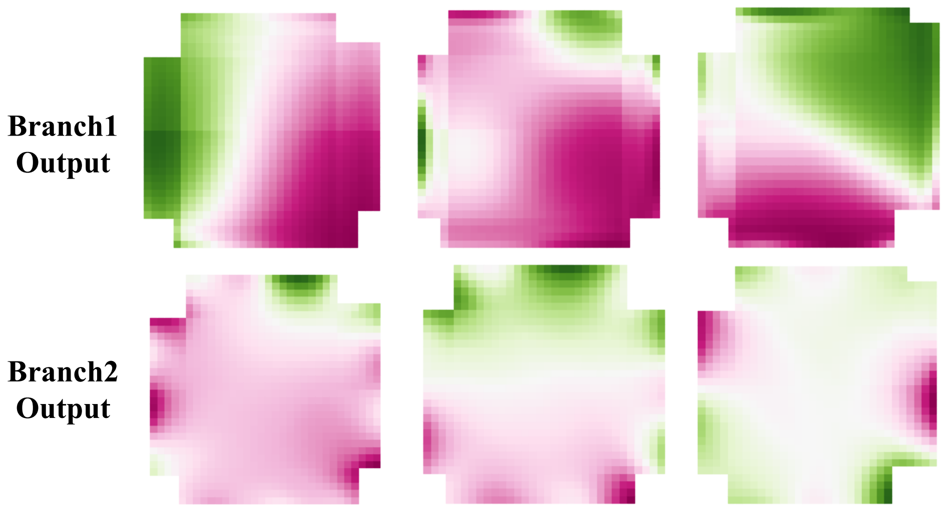

Figure 9: Visualization of the output from 2 GNN branches.

Table 10: Performances of our proposed BENO and the compared baselines on Darcy flow, which are trained on 900 4-corners samples and tested on 5 datasets under relative L2 norm and MAE separately. The unit of the MAE metric is 1×10^{-3}. Bold fonts indicate the best.

Train on 4-Corners dataset										
Test set	4-Corners		3-Corners		2-Corners		1-Corner		No-Corner	
Metric	L2	MAE	L2	MAE	L2	MAE	L2	MAE	L2	MAE
MP-PDE	0.5802± 0.1840	0.3269± 0.2085	0.5332± 0.1742	0.4652± 0.2999	0.6197± 0.1709	0.6307± 0.3282	0.6906± 0.1432	0.8469± 0.4087	0.7406± 0.1271	1.0906± 0.3949
BENO (ours)	**0.2431± 0.0895**	**0.1664± 0.0773**	**0.2542± 0.1252**	**0.2150± 0.1270**	**0.2672± 0.1497**	**0.2585± 0.1313**	**0.2466± 0.1405**	**0.3091± 0.2350**	**0.2366± 0.1104**	**0.3591± 0.2116**

L2 norm of 0.1091 and an MAE of 0.2599, outperforming all other methods and showcasing the effectiveness of our approach under strict 4-corners conditions.

When trained on mixed boundary conditions in Table 9, BENO still maintains the highest accuracy, yielding an relative L2 norm of 0.1023 and an MAE of 0.2373 on the 4-Corners test set, confirming its robustness to varied training conditions. Notably, the improvement is significant in the more challenging No-Corner test set, where BENO's MAE is 0.1517, a remarkable enhancement over the baseline methods. The bolded figures in the tables highlight the instances where BENO outperforms all other models, underscoring the impact of our boundary-embedded techniques.

The consistency of BENO's performance under different boundary conditions underscores its potential for applications in computational physics where such scenarios are prevalent. Besides, the experimental outcomes affirm the efficacy of BENO in handling complex boundary problems in the context of PDEs. It is also worth noting that the BENO model not only improves the prediction accuracy but also exhibits a significant reduction in error across different test cases, which is critical for high-stakes applications such as numerical simulation in engineering and physical sciences.

M Experiments on Darcy Flow

In this section, we consider the solution of the Darcy flow using our proposed BENO approach. The 2-d Darcy flow is a second-order linear elliptic equation of the form

$$\begin{aligned} \nabla \cdot (\kappa(x,y)\nabla u(x,y)) &= f(x,y), \quad \forall (x,y) \in \Omega, \\ \partial u(x,y) &= g(x,y), \quad \forall (x,y) \in \partial\Omega, \end{aligned} \tag{19}$$

where the coefficients κ is generated by taking a linear combination of smooth basis function in the solution domain. The coefficients of the linear combination of these basis functions is taken from uniform distribution of random numbers. Dirichlet boundary condition is imposed along the boundary $\partial\Omega$ using the function g which is sufficiently smooth. The objective of BENO is to map from the coefficient κ to solution u of the PDE in Equation 19.

The model was exclusively trained on a dataset comprised of 900 samples, each featuring 4-corner configurations. To assess the robustness and adaptability of our method, we conduct evaluations on five distinct test datasets. These datasets are deliberately chosen to represent a variety of boundary conditions commonly encountered in real-world applications, with each one designed to present the model with different levels of boundary complexity. The outcomes of these evaluations are detailed in Table 10. Our proposed BENO model consistently outperforms the best baseline across all test datasets. Notably, BENO achieves the lowest Mean MAE and relative L2 norm in the majority of these scenarios. This performance underscores the effectiveness of our approach, particularly under the stringent conditions of 4-corner boundaries.

N Visualization of Two Branches

In this section, the visualized outputs of two distinct branches offer a deeper insight into our model's functionality. Branch1, with the boundary input set to zero, is posited to approximate the impact emanating from the interior, while Branch2, nullifying the interior inputs, is conjectured to capture the boundary's influence on the interior. The observations from Figure 9 lend credence to our hypothesis, indicating a discernible delineation of roles between the two branches.

Extending this analysis, we further postulate that the interplay between Branch1 and Branch2 is critical for accurately modeling the PDE solution landscape. The synergy of these branches, as evidenced in our results, showcases a composite model that effectively balances the intricate boundary-interior dynamics. This balance is crucial in situations where boundary conditions significantly dictate the behavior of the system, further emphasizing the robustness and adaptability of our model. The innovative dual-branch strategy presents a promising avenue for enhancing the interpretability and precision of PDE solutions in complex domains.

O Hyper-parameter List

Table 11: Hyper-parameters Configuration

Hyper-parameter Name	**Hyper-parameter Value**
Boundary Dimension	128
Node Dimension	128
Edge Dimension	128
Epochs	1000
Learning Rate	5e-05
MLP Layers in Eq. 7	3
Nearest Node Number K	8
Message Passing Steps T	5
Transformer Layers	1
Number of Attention Head	2
Number of iterations for the first restart	16
Scheduler	CosineAnnealingWarmRestarts
Activation Function	Sigmoid Linear Unit (SiLU)
GPU Device	Nvidia A100 GPU