

MMET: A Multi-Input and Multi-Scale Transformer for Efficient PDEs Solving

Yichen Luo¹, Jia Wang², Dapeng Lan³, Yu Liu³ and Zhibo Pang^{*,1,4}

¹Division of ISE, KTH Royal Institute of Technology, Stockholm, Sweden

²School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, China

³Techforgood AS, Oslo, Norway

⁴Department of Automation Technology, ABB Corporate Research Sweden, Vasteras, Sweden
{yichenlu, zhibo}@kth.se, Jia.Wang02@xjtlu.edu.cn, {dapengl, liu.yu}@ieee.org

Abstract

Partial Differential Equations (PDEs) are fundamental for modeling physical systems, yet solving them in a generic and efficient manner using machine learning-based approaches remains challenging due to limited multi-input and multi-scale generalization capabilities, as well as high computational costs. This paper proposes the Multi-input and Multi-scale Efficient Transformer (MMET), a novel framework designed to address the above challenges. MMET decouples mesh and query points as two sequences and feeds them into the encoder and decoder, respectively, and uses a Gated Condition Embedding (GCE) layer to embed input variables or functions with varying dimensions, enabling effective solutions for multi-scale and multi-input problems. Additionally, a Hilbert curve-based reserialization and patch embedding mechanism decrease the input length. This significantly reduces the computational cost when dealing with large-scale geometric models. These innovations enable efficient representations and support multi-scale resolution queries for large-scale and multi-input PDE problems. Experimental evaluations on diverse benchmarks spanning different physical fields demonstrate that MMET outperforms SOTA methods in both accuracy and computational efficiency. This work highlights the potential of MMET as a robust and scalable solution for real-time PDE solving in engineering and physics-based applications, paving the way for future explorations into pre-trained large-scale models in specific domains. This work is open-sourced at <https://github.com/YichenLuo-0/MMET>.

1 Introduction

The solution of complex partial differential equations (PDEs) has long been one of the great challenges in engineering. In the vast majority of cases involve elasticity, fluid dynamics or other physics fields, it is difficult to obtain analytical solutions for their governing PDEs. As an alternative, numerical methods such as the Finite Element Method (FEM) are usually used to approximate the result. However, these methods

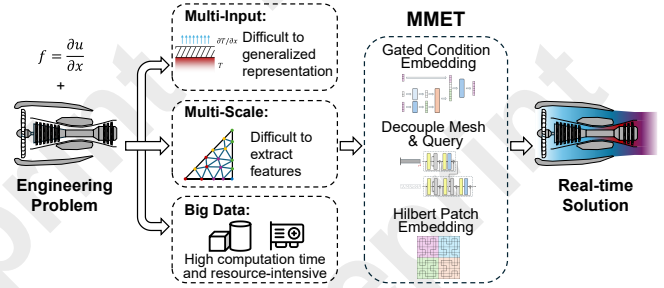


Figure 1: MMET mainly addresses three challenges in applying machine learning to large-scale PDEs solving, including multiple initial/boundary conditions input, multi-scale resolution, and high computational cost in large-scale geometric models.

are computationally expensive and challenging to accelerate using parallel hardware.

Recent advances in the field of Artificial Intelligence (AI) have explored the use of machine learning techniques to approximate physical laws directly from PDEs [Raissi *et al.*, 2019] or big data [Wang *et al.*, 2021], and have demonstrated promising accuracy in Computational Fluid Dynamics (CFD) [Raissi *et al.*, 2020; Mao *et al.*, 2020], magnetohydrodynamics [Rosofsky and Huerta, 2023], and other fields. Nonetheless, traditional fully connected neural networks can typically solve only a single instance in a set of PDEs. Any changes in geometric shape, initial/boundary conditions, or material properties require retraining of the neural network. This makes the speed advantage of AI-based approach over traditional FEM a pseudo proposition, as the cost of training a neural network for a single instance can be comparable to or even exceed that of FEM.

Recently, generative pre-trained large-scale models based on the Transformer architecture [Vaswani *et al.*, 2017] have shown strong zero-shot or few-shot generalization abilities in natural language processing [OpenAI, 2023], machine vision [Liu *et al.*, 2024], and more scenarios. By extending this paradigm to the field of PDE-driven physics simulation, it becomes feasible to develop universal models that do not require expensive retraining for each instance. Several research efforts have attempted to fit PDEs using the Transformer architecture, targeting various aspects such as enhancing operator learning [Li *et al.*, 2022; Shih *et al.*, 2025], gener-

alization [Santos *et al.*, 2023], multi-scale meshes [Hao *et al.*, 2023], simulation accuracy [Zhao *et al.*, 2024], geometric model segmentation [Wu *et al.*, 2024a], and improving the attention mechanism to enhance accuracy [Cao, 2021].

Despite these advancements, the above methods primarily address specific challenges within their respective domains. However, a universal paradigm that can systematically tackle all the key challenges within a unified framework remains absent. These challenges can be summarized as follows.

- **Multi-Input Representation:** A complete PDE solving problem can contain multiple inputs, such as differential operator, geometry domain, initial/boundary conditions, etc., which are in the form of variables or functions. The model must possess a robust and adaptable input encoding mechanism capable of efficiently encoding above diverse inputs.
- **Multi-Scale Resolution:** Different scenarios often require varying output resolutions, leading to inconsistencies in query inputs. However, Transformer-based models may alter their attention mechanisms as the input sequence changes, resulting in fluctuations in query accuracy under different resolutions.
- **Computationally Expensive:** Meshes in industrial scenarios often consist of tens to hundreds of thousands of node points, posing significant challenges to the cost and efficiency of attention mechanisms at such scales.

Based on these limitations, we propose the **Multi-input and Multi-scale Efficient Transformer (MMET)** architecture, which can be used to solve a class of general problems in real time for certain groups of PDEs. Our contributions can be summarized as follows:

- Proposing the Gated Condition Embedding (GCE) layer, embedding input functions and variables in units of spatio-temporal points, which can effectively encode complex input conditions while ensuring that the input sequence length does not increase significantly.
- Decoupling the query points and the geometry mesh into two sequences for the input of the encoder and decoder, respectively, enabling queries at any resolution.
- Reserializing and patching the mesh sequence using the Hilbert curve [Hilbert, 1935], significantly reducing the computational cost of the attention mechanism.

Experiments across various physical fields demonstrate that these innovative methods significantly enhance both the accuracy and computational efficiency of models in solving multi-input and multi-scale PDEs.

2 Related Work

2.1 Physics-Informed Neural Networks

Physics-Informed Neural Networks (PINNs) [Raissi *et al.*, 2019] are a class of numerical simulation approaches based on machine learning aiming to approximate the solution of PDEs by incorporating physical laws directly into the loss function of the artificial neural network. This methodology can actually be regarded as a form of self-supervised learning

rather than a network design, as the architecture of the neural network itself is highly flexible.

2.2 Operator Learning

The operator learning paradigm, originating from DeepONet [Wang *et al.*, 2021], has recently replaced traditional neural network architectures as a new generalized PDE solver. Among these methods, the FNO [Li *et al.*, 2021] stands out for its efficiency, leveraging linear projections in the Fourier domain to handle high-dimensional inputs. Extensions such as U-NO [Rahman *et al.*, 2022] and F-FNO [Tran *et al.*, 2023] improve multiscale representation or computational efficiency, further expanding the versatility of neural operators. On the other side, models like the GNO [Anandkumar *et al.*, 2020], MGNO [Li *et al.*, 2020], and Geo-FNO [Li *et al.*, 2023b] have been proposed to enhance the perception of irregular geometric models by employing graph-based learning and geometric Fourier transforms. Several recent studies, such as 3D-GeoCA [Deng *et al.*, 2024] and Geom-DeepONet [He *et al.*, 2024], further enhance this capability by introducing techniques from the field of point cloud processing.

While operator learning methods demonstrate generalization ability in the face of variations in differential operators, challenges arise when dealing with other complex local variables such as geometries or boundary conditions. Without an efficient attention mechanism, these architectures often struggle to encode those intricate local features. It is worth mentioning that 3D-GeoCA [Deng *et al.*, 2024] uses a backbone structure to encode the geometric features and boundary conditions of the problem domain. In their experiments, this method was also integrated with Point-BERT [Yu *et al.*, 2022] and other advanced Transformers. However, its main purpose is to enhance the capacity of the model to perceive complex geometric configurations, while the ability to generalize across multiple types of inputs remains unexplored.

2.3 Transformer for PDEs

Several past works have explored the possibility of applying the Transformer architecture to the field of AI for PDEs. In which, HT-NET [Ma *et al.*, 2022] takes advantage of the Swin Transformer [Liu *et al.*, 2021] and the multigrid method to enhance the ability of multi-scale spatial perception. OFormer [Li *et al.*, 2022] and IPOT [Lee and Oh, 2024] use Transformer to learn operators from meshes with varying resolutions. Recent studies, GNOT [Hao *et al.*, 2023] and UPT [Alkin *et al.*, 2024], attempted to enhance the multi-input generalization and multi-scale ability of the model. In addition, there are several studies devoted to improving the attention mechanism and increasing the accuracy of neural networks in fitting PDEs or operators [Cao, 2021; Li *et al.*, 2023a; Xiao *et al.*, 2024].

The vast majority of the above models are trained using labeled data, which are extremely expensive in real industrial scenarios. While PINNs have demonstrated considerable potential in reducing data dependency, these complex networks have more complex gradients and are harder to converge when using PDEs as part of the loss function [Rathore *et al.*, 2024]. In this regard, PIT [Santos *et al.*, 2023] and PinnsFormer [Zhao *et al.*, 2024] explored direct training through

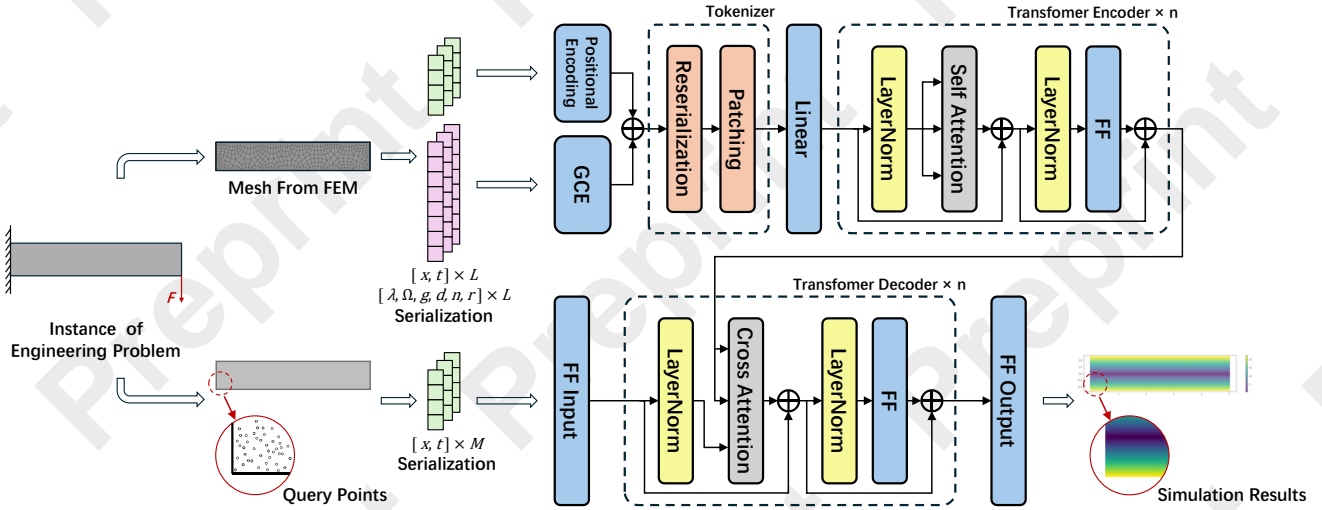


Figure 2: Overall architecture of MMET. Where L and M are the sequence lengths of the serialized mesh and query points, respectively. In this architecture, the encoder is responsible for extracting and encoding the feature information of the whole problem domain, which comes from the mesh pseudo-sequence; and the decoder is responsible for solving the value at the query points based on the encoded information.

physics-informed methods and have been successful in some small-scale cases. Nevertheless, they did not consider generalization to more complex input conditions.

In addition, most of these studies apply the attention mechanism to the whole domain and directly embed each sample point as a token. When the complexity of the geometric model increases, the number of sample points or meshes required may reach tens or even hundreds of thousands, making the sequence length unacceptable for the global attention mechanism. Transolver [Wu *et al.*, 2024a] proposed a new physics-attention mechanism for segmenting geometric meshes to solve this problem, making it possible to solve complex geometric models under a fixed sequence length. Nevertheless, it relies on generating a weight map for each slice through a feedforward subnetwork, which lack global receptive fields to capture diverse input conditions. Therefore, a well-trained slicing subnetwork can only be effective for dynamic inputs within limited ranges. To address these varying conditions in a more universal manner, a predefined, non-learned slicing strategy should be considered.

3 Problem Setup

We consider the following PDE in a general form:

$$\mathcal{N}(u; \lambda) = 0, \quad x \in \Omega, t \in [0, T], \quad (1)$$

where $u(x, t)$ is the expected value to be solved; $\mathcal{N}(u; \lambda)$ is the differential operator that describes the PDE, which is parameterized by physical parameters λ ; Ω is the spatial domain that depends on the geometry; T is the max time of the field.

For the complete definition of a problem instance, the PDE should also have the following initial conditions:

$$u(x, 0) = g(x), \quad x \in \Omega, \quad (2)$$

and different types of boundary conditions:

$$\begin{aligned} u &= d(x, t), & x \in \partial\Omega_d, t \in [0, T], \\ \partial u / \partial x &= n(x, t), & x \in \partial\Omega_n, t \in [0, T], \\ a(\partial u / \partial x) + bu &= r(x, t), & x \in \partial\Omega_r, t \in [0, T], \end{aligned} \quad (3)$$

where $\partial\Omega$ represents the boundary of the problem domain, and d, n, r are the Dirichlet, Neumann, and Robin boundary conditions, respectively.

To develop an ideal large-scale universal model for PDE solving, it is necessary to design a network characterized by the following mapping relationship:

$$f(x, t, \lambda, \Omega, g, d, n, r; \theta) \rightarrow u, \quad (4)$$

where θ are the parameters of the model, the domain Ω and initial/boundary conditions g, d, n, r are highly complex and locally varying. The mapping f is required to accurately solve a set of instances with the above variables, which represents a general class of problems in the engineering field.

4 Our Method

4.1 Overall Architecture

Given the complexity of geometric features, a common approach is to discretize them into point cloud or unstructured mesh for feature extraction. In MMET, we employ two discretization strategies for the instance under consideration, as shown in Figure 2. First, we adopt the same mesh generation technique as used in classical FEM [Okereke and Keates, 2018]. The generated mesh serves as the input pseudo-sequence of the encoder to extract the global features. For the region of interest, the domain is further discretized into points to be queried, and is subsequently fed into the decoder.

In the encoder part of Figure 2, the input mesh pseudo-sequence comprises not only the spatial-temporal coordinates (x, t) , but also the sampling values of the differential operator \mathcal{N} , initial/boundary conditions g, d, n, r , and other necessary variables at this point. The coordinates of each point are directly fed into the positional encoding layer as position indices, and this layer can adopt a learnable feedforward network or fixed positional encoding as shown in PIT [Santos *et al.*, 2023]. To handle complex input variables or functions,

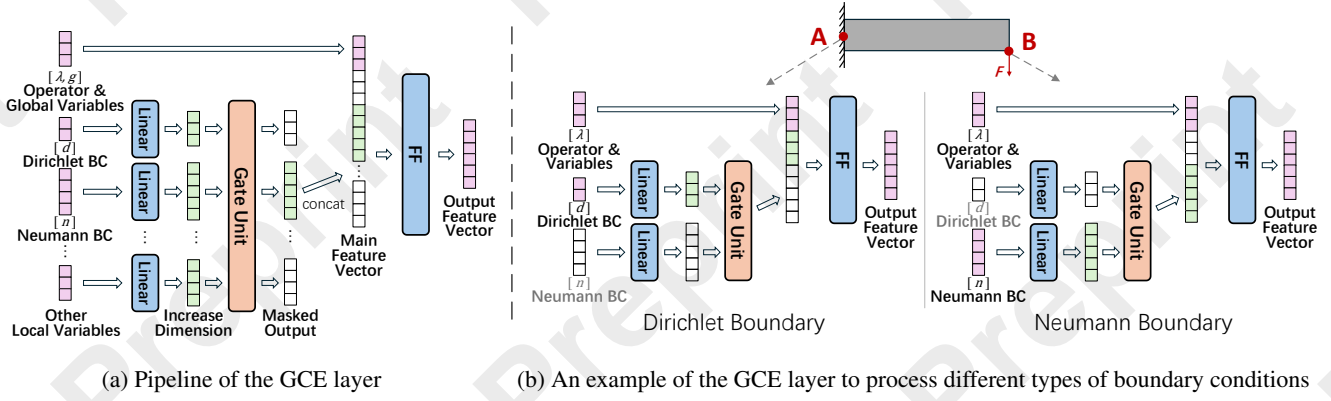


Figure 3: Pipeline of the GCE layer and an example. The white square in the vector indicates that the value at that position is missing. Through dimension expansion and gating, this module can effectively extract features from different dimensions of input variables.

the innovative GCE layer is employed for effective embedding, which will be described in detail in Section 4.2. Subsequently, the sequence is reserialized using a Hilbert curve for spatial coherence and divided into patches of a specified size. Each patch is transformed into a token via a linear layer, which will be described in Section 4.3. The resulting token sequence is fed into the encoder for feature encoding.

In the decoder part of Figure 2, to enable querying at arbitrary resolutions, each query point is directly treated as a token and fed into the decoder. The self-attention layer typically used in the decoder of the classical Transformer [Vaswani *et al.*, 2017] is removed to ensure that the output is not influenced by the construction of the query sequence itself. Therefore, the query point sequence only performs cross-attention with the output feature sequence of the encoder. This design offers significant flexibility, allowing query points to be sampled in any quantity and at any location within the domain Ω . Furthermore, if large number of points must be queried simultaneously or hardware resources are limited, the query sequence can be divided into batches and processed sequentially without affecting the final result.

In our model, all activation functions are replaced by the wavelet function proposed in PinnsFormer [Zhao *et al.*, 2024] for better accuracy. The attention layer can be configured as classical dot-product attention [Vaswani *et al.*, 2017], Galerkin attention [Cao, 2021], or linear attention [Hao *et al.*, 2023], depending on the specific requirements.

4.2 Gated Condition Embedding

The previous approach GNOT [Hao *et al.*, 2023] uses an architecture similar to Mixture-of-Experts (MoE) approach [Shazeer *et al.*, 2017; Fedus *et al.*, 2022] to encode different features, which may lead to extremely long input sequences caused by repeated encoding of different features at the same point. The GCE layer is a novel module we designed specifically to address this deficiency. As shown in Figure 3a, we embed different input functions in units of spatio-temporal points (x, t) rather than the sample values sequence of each function. Under this embedding approach, samples of different boundary conditions d , n , and r exhibit different dimensional properties depending on whether the boundary point

corresponds to Dirichlet, Neumann, or Robin types.

As an example, in Figure 3b, we consider two boundary points A and B with Dirichlet and Neumann boundary conditions $d_A(x_A, t_A)$ and $n_B(x_B, t_B)$, respectively. The input vector of point A only needs to contain its sampled value d_A , while point B may need to contain not only sampled value n_B , but also the normal vector \vec{n} of that boundary. However, neural networks require input vectors to have fixed dimensions. Naively masking missing features with zeros introduces ambiguity, as a fully connected network cannot distinguish between an input value that is genuinely zero and one representing missing data.

The GCE layer resolves this issue by mapping each type of input variable to a higher-dimensional space using a linear layer. After training, the original feature dimensions form a hyperplane in this higher-dimensional space that does not intersect the origin. If the input vectors of points A and B are expressed as vectors $\vec{d}_A \in \mathbb{R}^a$, $\vec{n}_B \in \mathbb{R}^b$, the outputs of the proposed linear layers are given as:

$$\begin{aligned} \vec{h}_A &= W_d \cdot \vec{d}_A + \vec{b}_d, & \vec{h}_A &\in \mathbb{R}^{a+1}, \\ \vec{h}_B &= W_n \cdot \vec{n}_B + \vec{b}_n, & \vec{h}_B &\in \mathbb{R}^{b+1}, \end{aligned} \quad (5)$$

where $W_d \in \mathbb{R}^{(a+1) \times a}$ and $W_n \in \mathbb{R}^{(b+1) \times b}$ are the weight matrices of the linear layers for Dirichlet and Neumann boundary condition inputs, respectively, and they are independent of each other; $\vec{b}_d \in \mathbb{R}^{a+1}$ and $\vec{b}_n \in \mathbb{R}^{b+1}$ are the bias vectors of these layers.

A gate unit is then used to control whether the mapped features \vec{h}_A and \vec{h}_B should be integrated into the embedding vector. If this feature is missing, the value at that position in the embedding vector will be set to zero, representing the origin in the high-dimensional space. Importantly, this gate unit is parameter-free and relies solely on the presence or absence of the corresponding variables for a given point. Therefore, the embedding vectors of points A and B can be expressed as:

$$\begin{aligned} \vec{e}_A &= \text{MLP}[\text{concat}(\lambda, g, \vec{h}_A, \vec{o}_A)], & \vec{e}_A &\in \mathbb{R}^c, \\ \vec{e}_B &= \text{MLP}[\text{concat}(\lambda, g, \vec{o}_B, \vec{h}_B)], & \vec{e}_B &\in \mathbb{R}^c, \end{aligned} \quad (6)$$

where $\vec{o}_A \in \mathbb{R}^{b+1}$ and $\vec{o}_B \in \mathbb{R}^{a+1}$ are zero vectors masked by the gate unit, which are used to pad missing inputs.

This design allows the model to reliably distinguish between missing inputs and inputs with a value of zero, ensuring a more robust and interpretable representation.

4.3 Reserialization and Patch Embedding

Patching the mesh sequence to a larger slice not only reduces its overall length but also enhances the model’s performance on complex instances, as directly applying attention mechanisms to a large number of mesh points often hinders the model’s ability to capture meaningful relationships [Wu *et al.*, 2022]. To ensure generalization to different input conditions, we avoid using any learnable patch method such as those proposed in Transolver [Wu *et al.*, 2024a].

Inspired by the work in point cloud processing, PTV3 [Wu *et al.*, 2024b], we first reserialize the mesh points using the Hilbert curve [Hilbert, 1935], which is a type of space-filling curve designed to map high-dimensional discrete spaces into one-dimensional sequences while preserving spatial coherence. Compared with alternatives like Z-order curves or random raster scans, Hilbert ordering more consistently maps nearby nodes in 2D/3D space to adjacent positions in the serialized sequence, making it well-suited for downstream patching. This improves the likelihood that spatially related features are grouped within the same patch, which is especially beneficial for attention-based models operating under fixed memory budgets.

As shown in Figure 4, we take a 2D mesh as an example. Consider a node (x, y) within this unstructured mesh, linearly mapped into a square where each dimension is discretized into 2^n units. The Hilbert curve recursively subdivides the square into four subregions, with the number of recursion determined by the order number n . Each subregion at a given order i is sequentially assigned a code $b_i \in [0, 3]$, which can be expressed as:

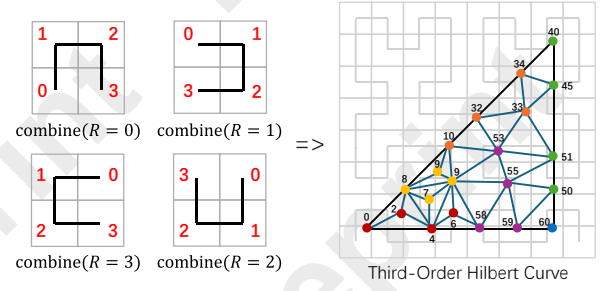
$$b_i = \text{index}(x, y, R_i), \quad i \in [0, n], \quad (7)$$

where R_i represents the rotation direction at order i , which depends on the starting orientation of the recursion. The $\text{index}()$ function is visualized in Figure 4a, dynamically adjusting the rotation direction R_i to ensure that spatially adjacent coordinates maintain locality in the serialized sequence. The Hilbert code e for each coordinate can be computed as:

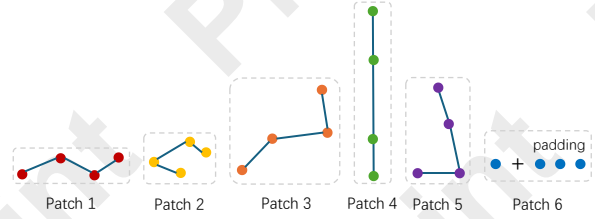
$$e = \sum_{i=0}^{n-1} b_i \times 2^{2i}. \quad (8)$$

Each node in the original mesh is sorted according to their Hilbert code e to form a new 1D sequence. This sequence is then patched according to a fixed size. The vectors within each patch are concatenated and passed through a linear layer to extract patch features, which serve as individual tokens for the encoder.

This approach further benefits from FEM’s adaptive meshing: regions with sharp gradients are a priori assigned denser meshes. By fixing the patch size, these regions generate more tokens, allowing the attention mechanism to prioritize critical areas. This design enhances MMET’s capability in multi-scale scenarios while preserving accuracy for complex geometries.



(a) Reserialize the mesh using Hilbert curve



(b) Patching the mesh nodes as 1D sequence

Figure 4: Reserialize and patching of the mesh. Points with same color indicate the same patch. Regions with denser nodes are assigned more patches, as patch 1 and patch 2.

5 Experiment and Result

5.1 Benchmark Comparisons

In this section, we conduct benchmark experiments and compare our model to demonstrate its advantages over previous methods.

Experimental Setting. We select six representative datasets spanning elasticity, fluid mechanics, and thermodynamics. Here, A indicates a dataset with multiple geometric shapes, B indicates datasets with various boundary conditions, and C indicates datasets with multi-scale query resolution. The following is a brief introduction.

- **Poisson:** A simple 2D Poisson problem that is mainly used to verify the effectiveness of each baseline under physics-informed training.
- **Darcy Flow** [Takamoto *et al.*, 2024]: The 2D Darcy flow problem from the PDEBench datasets, aims to learn the mapping from spatially varying diffusion coefficients to pressure distributions.
- **Shape-Net Car (A)** [Umetani and Bickel, 2018]: A multi-scale 3D aerodynamics problem used to predict the flow velocity v and surface pressure p of different cars. This dataset includes complex geometric features.
- **Heat2d (A, B)** [Hao *et al.*, 2023]: A multi-scale heat conduction problem used to predict the temperature field T from input functions.
- **Beam2d (B, C)** [Bai *et al.*, 2023]: This dataset solves the stress σ_x , σ_y , τ_{xy} , and displacement u , v of a rectangular cantilever beam when the end is subjected to a bending moment. We modify this case so that the bending moment varies dynamically within a certain range.

| Dataset | Property | | Relative L_2 Average Error | | | | |
|---------------|-----------|------------|------------------------------|------------|----------|---------------------|----------------|
| | Challenge | Variable | PinnsFormer | Transolver | GNOT | MMET (encoder-only) | MMET |
| Poisson | – | u | 4.90e-3 | 7.19e-2 | 2.73e-2 | 1.12e-2 | 5.65e-2 |
| Darcy Flow | – | p | 3.69e-2 | 7.87e-2 | 3.23e-1 | 3.83e-2 | 3.34e-2 |
| Shape-Net Car | A | v | 4.88e-2 | 3.29e-2* | 2.07e-2* | 1.93e-2 | 4.35e-2 |
| | | p | 1.21e-1 | 7.98e-2* | 7.45e-2* | 7.38e-2 | 1.11e-1 |
| Heat2d | A, B | T | - | - | 4.13e-2* | 3.42e-2 | 3.36e-2 |
| Beam2d | B, C | u | - | - | 3.37e-1 | - | 4.64e-2 |
| | | v | - | - | 8.71e-2 | - | 7.46e-3 |
| | | σ_v | - | - | 1.08e-2 | - | 1.16e-3 |
| HeatSink2d | A, B, C | T | - | - | 1.33e-1 | - | 1.18e-1 |

Table 1: Performance comparison on our selected datasets. Where “-” indicates that the model does not support solving such problems, and results of other models with “*” indicate that these results are from their original paper. A lower error value indicates that the model performs better. In the Beam2d dataset, σ_v represents the von Mises yield stress, which is calculated by normal stresses σ_x , σ_y , and shear stress τ_{xy} .

- HeatSink2d (A, B, C): A thermodynamic case designed to solve the internal temperature T of a 2D heat sink with variable height and surface temperature. This case relies solely on PDEs for physics-informed training, and without using labeled data.

Meanwhile, we selected another three recent Transformer-based SOTA methods as baselines, including PinnsFormer [Zhao *et al.*, 2024], Transolver [Wu *et al.*, 2024a], and GNOT [Hao *et al.*, 2023]. For MMET, we also tested the encoder-only configuration on some of the datasets that do not need for multi-scale queries.

For all models, we employ AdamW [Loshchilov and Hutter, 2019] or L-BFGS [Liu and Nocedal, 1989] optimizers for combinatorial optimization. The model is trained on two NVIDIA H20 GPUs. The relative L_2 error is used to evaluate all models, which has also been widely applied in previous works [Wu *et al.*, 2024a; Xiao *et al.*, 2024; Hao *et al.*, 2023]. The error is computed as follows:

$$e = \frac{1}{N} \sum_{i=1}^N \frac{\|u_i - u'_i\|_2}{\|u\|_2}, \quad (9)$$

where u_i is the prediction value output by the model, u'_i is the ground truth, and N is the number of data in the test set.

For the Poisson, Shape-Net Car and Darcy Flow dataset, since there are no multiple boundary condition inputs to embed, we therefore replace the GCE layer by a simple feed-forward network. For the HeatSink2d dataset, the error is expected to be larger than for others because training relies solely on PDEs, while the ground truth from Ansys Workbench (via FEM) also contains approximation errors.

Result and Discussion. The results in Table 1 demonstrate that our model consistently outperforms selected SOTA baselines in nearly all complex datasets, highlighting the effectiveness of MMET in addressing large-scale and dynamic physical fields. In contrast, PinnsFormer and Transolver lack effective encoding mechanisms for boundary conditions, rendering them incapable of handling problems with multiple input conditions. It is worth noting that on the Heat2d dataset designed by GNOT’s team (which is characterized by multi-input, multi-geometry, and multi-scale), our model shows higher accuracy than GNOT. This further validates the capability of MMET in tackling intricate and diverse scenarios.

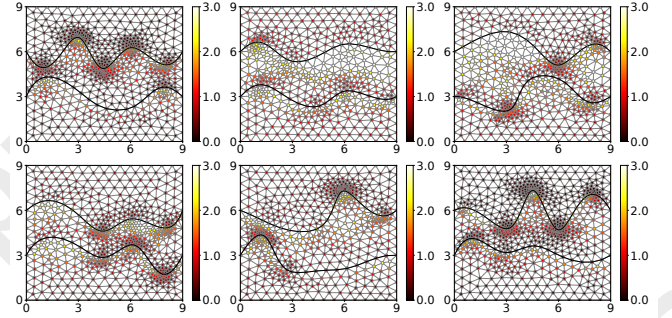


Figure 5: The attention heatmap of the decoder on the Heat2d dataset, under varying boundary conditions and geometries, adaptively focuses on critical regions such as geometric boundaries or gradients, confirming the spatial interpretability of MMET.

However, we also observe that the encoder-decoder structure in complete MMET converges more slowly and is sensitive to large learning rates. Therefore, for tasks that do not require arbitrary resolution queries, such as Shape-Net Car, an encoder-only variant is recommended to improve training speed, stability, and accuracy.

5.2 Multi-Scale Query Experiment

We designed a supplementary experiment to verify the zero-shot query ability of our model at different resolutions. For the Beam2d dataset, in the training stage, our model uses an unstructured mesh with 5404 nodes generated by Ansys Workbench. The query sequence is a regular point matrix with a resolution of $[50 \times 20]$. In the inference stage, we keep the mesh input the same as in the training stage and test the performance of the query points at four different resolutions, as shown in Figure 6.

Experimental results indicate that decoupling the mesh from the query sequence enables our model to maintain fully consistent error performance across varying query resolutions. This demonstrates the distinct fitting capability of MMET within the continuous function space, comparable to that of traditional fully connected networks. In contrast, other Transformer-based models suffer from accuracy fluctuations as the input sequence length changes.

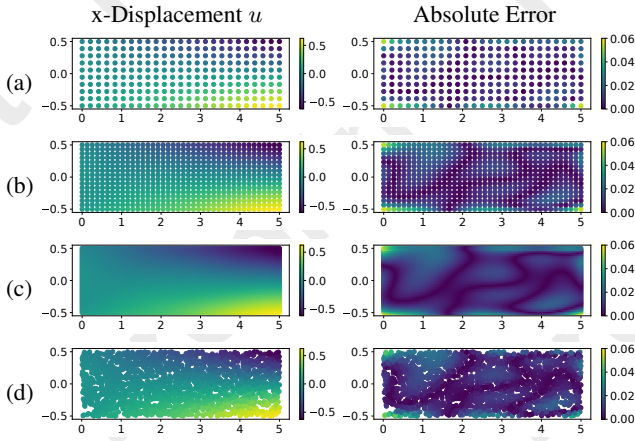


Figure 6: Results of x-axis displacement u queries under different resolutions. Where (a) is the result under $[25 \times 10]$ resolution; (b) is the result under $[40 \times 16]$ resolution; (c) is the result under $[100 \times 40]$ resolution; (d) is the result under 2000 completely random points.

| Dataset Part | MLP | MLP + Boundary Type | GCE |
|---------------|---------|------------------------|----------------|
| HeatSink2d -D | 3.90e-1 | 2.42e-2 | 1.52e-2 |
| HeatSink2d -N | 2.61e-1 | 2.11e-2 | 9.86e-3 |

Table 2: Relative L_2 error under different types of embedding layers. Where “-D” represents the part of the data in the dataset with temperature (Dirichlet) boundaries, and “-N” represents the part of the data with heat flux (Neumann) boundaries.

5.3 Ablation Experiment

In this section, we evaluate the impact of disabling the GCE layer and modifying the patch size in different datasets, and analyzing their effects on the performance of our model.

GCE Layer. To assess the significance of the GCE layer, we designed an experiment introducing ambiguity in boundary representation. This experiment is modified from HeatSink2d dataset, where a boundary is represented as a heat flux (Neumann) boundary in part of the dataset and as a zero-value temperature (Dirichlet) boundary in the remainder. If a zero pad is used to pad the missing inputs, then the sampled characteristics of the two parts on this boundary would be exactly the same. We evaluated the performance of the model using three embedding approaches: a simple MLP, an MLP with an additional input node to encode the boundary type, and our GCE layer. The results of this experiment are presented in Table 2.

It can be seen that using a simple MLP layer results in complete failure, with errors exceeding 20% in both parts of the dataset. This is because the exact same input makes the MLP unable to effectively distinguish between different types of boundary conditions. Introducing a boundary type indicator into the input indeed enables the model to learn effectively. Nonetheless, our GCE layer achieves significantly higher accuracy. This result shows the superior performance of our method in resolving input ambiguities and dynamically encodes boundary conditions with different dimensions.

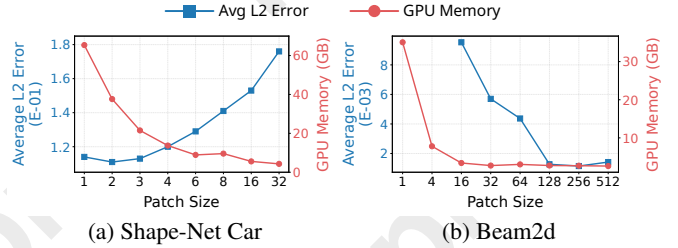


Figure 7: The lowest relative L_2 error and GPU memory consumption for each dataset. Specifically, the GPU memory consumption is tested in the inference phase with a batch size of 50.

Patch Embedding. In the Shape-Net Car and Beam2d datasets, we dynamically adjust the patch size in a certain range to evaluate the effectiveness of our patching method. Notably, a patch size of 1 is equivalent to disabling patch embedding. To ensure a fair comparison, the model scale and training configuration are maintained consistently in all settings. The best accuracy and GPU memory consumption are shown in Figure 7.

For the Shape-Net Car dataset, the experimental results indicate that our method surpasses the no-patch configuration when the patch size ranges from 2 and 3, while significantly reducing GPU memory consumption during training and inference. However, excessively large patch sizes lead to a noticeable decline in performance without obvious gains in efficiency. In contrast, for the Beam2d dataset, the model fails to converge with small patch sizes, indicating an inability to effectively extract information from a complex mesh. As the patch size increases, the convergence rate and final accuracy of the model improve progressively.

These findings highlight the effectiveness of our approach in handling complex meshes. The Shape-Net Car dataset, derived from 3D point clouds sampled uniformly across model surfaces, differs fundamentally from the Beam2d dataset, which consists of meshes generated by Ansys Workbench. The latter a priori incorporates physical information. This enables the model to prioritize regions with complex gradients more effectively. By employing fixed-size patches, our method implicitly assigns different weights to various regions of the geometry, thereby enhancing the efficiency and accuracy of the attention mechanism.

6 Conclusion

In this work, we proposed the MMET, a novel framework for real-time solving of generalized PDEs with multi-input and multi-scale resolution. The architecture incorporates key innovations, including the GCE layer and Hilbert curve-based reserialization and patching, enabling efficient handling of complex geometries and input conditions with scalability and accuracy. Experimental results across various benchmarks demonstrate the superior performance of MMET compared to SOTA methods, showcasing its potential for broad applications in complex physics-based simulations.

In the future, we plan to scale up the model, striving to establish it as a foundational pre-trained model for specific physics fields.

Acknowledgments

Yichen Luo is sponsored by the China Scholarship Council for PhD study at KTH Royal Institute of Technology, Sweden (No.202408320060). This work was partially supported by the National Natural Science Foundation of China (Grant No.72401232), the XJTLU External Research Fund under 2024-015, and LLL24ZZ-02-01 and LLL24ZZ-02-02 funded by the Liaoning Liaohe Laboratory.

References

- [Alkin *et al.*, 2024] Benedikt Alkin, Andreas Fürst, Simon Lucas Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers: A framework for efficiently scaling neural operators. In *Proceedings of the 38th International Conference on Neural Information Processing Systems (NeurIPS)*, 2024.
- [Anandkumar *et al.*, 2020] Anima Anandkumar, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Nikola Kovachki, Zongyi Li, Burigede Liu, and Andrew Stuart. Neural operator: Graph kernel network for partial differential equations. In *Proceedings of the ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020. Workshop paper.
- [Bai *et al.*, 2023] Jinshuai Bai, Timon Rabczuk, Ashish Gupta, Laith Alzubaidi, and Yuantong Gu. A physics-informed neural network technique based on a modified loss function for computational 2d and 3d solid mechanics. *Computational Mechanics*, 71(3):543–562, 2023.
- [Cao, 2021] Shuhao Cao. Choose a transformer: Fourier or galerkin. In *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*, 2021.
- [Deng *et al.*, 2024] Jingyang Deng, Xingjian Li, Haoyi Xiong, Xiaoguang Hu, and Jinwen Ma. Geometry-guided conditional adaptation for surrogate models of large-scale 3d pdes on arbitrary geometries. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI)*, 2024.
- [Fedus *et al.*, 2022] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(1), 01 2022.
- [Hao *et al.*, 2023] 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)*, 2023.
- [He *et al.*, 2024] Junyan He, Seid Koric, Diab Abueidda, Ali Najafi, and Iwona Jasiuk. Geom-deeponet: A point-cloud-based deep operator network for field predictions on 3d parameterized geometries. *Computer Methods in Applied Mechanics and Engineering*, 429:117130, 2024.
- [Hilbert, 1935] David Hilbert. Über die stetige abbildung einer linie auf ein flächenstück. *Dritter Band: Analysis-Grundlagen der Mathematik· Physik Verschiedenes: Nebst Einer Lebensgeschichte*, pages 1–2, 1935.
- [Lee and Oh, 2024] Seungjun Lee and Taeil Oh. Inducing point operator transformer: a flexible and scalable architecture for solving pdes. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence (AAAI)*, 2024.
- [Li *et al.*, 2020] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS)*, 2020.
- [Li *et al.*, 2021] 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 *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021.
- [Li *et al.*, 2022] Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations’ operator learning. *Transactions on Machine Learning Research*, 2022.
- [Li *et al.*, 2023a] Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for pde surrogate modeling. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NeurIPS)*, 2023.
- [Li *et al.*, 2023b] Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.
- [Liu and Nocedal, 1989] Dong C. Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1–3):503–528, 08 1989.
- [Liu *et al.*, 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [Liu *et al.*, 2024] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun. Sora: A review on background, technology, limitations, and opportunities of large vision models. *arXiv preprint arXiv:2402.17177*, 2024.
- [Loshchilov and Hutter, 2019] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.
- [Ma *et al.*, 2022] Mingjun Ma, Haiying Xia, Yumei Tan, Haisheng Li, and Shuxiang Song. Ht-net: hierarchical context-attention transformer network for medical ct image segmentation. *Applied Intelligence*, 52:10692 – 10705, 2022.

- [Mao *et al.*, 2020] Zhiping Mao, Ameya D. Jagtap, and George Em Karniadakis. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020.
- [Okereke and Keates, 2018] M. Okereke and S. Keates. *Finite Element Mesh Generation*, pages 165–186. Springer International Publishing, Cham, 2018.
- [OpenAI, 2023] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [Rahman *et al.*, 2022] Md Ashiqur Rahman, Zachary E. Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators. *Transactions on Machine Learning Research*, 2022.
- [Raissi *et al.*, 2019] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
- [Raissi *et al.*, 2020] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.
- [Rathore *et al.*, 2024] Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training pinns: A loss landscape perspective. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.
- [Rosofsky and Huerta, 2023] Shawn Rosofsky and Ema Huerta. Magnetohydrodynamics with physics informed neural operators. *Machine Learning: Science and Technology*, 4, 07 2023.
- [Santos *et al.*, 2023] Fabricio Dos Santos, Tara Akhound-Sadegh, and Siamak Ravanbakhsh. Physics-informed transformer networks. In *Proceedings of the NeurIPS 2023 Workshop on The Symbiosis of Deep Learning and Differential Equations III*, 2023. Workshop paper.
- [Shazeer *et al.*, 2017] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarczyk, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, 2017.
- [Shih *et al.*, 2025] Benjamin Shih, Ahmad Peyvan, Zhongqiang Zhang, and George Em Karniadakis. Transformers as neural operators for solutions of differential equations with finite regularity. *Computer Methods in Applied Mechanics and Engineering*, 434:117560, 2025.
- [Takamoto *et al.*, 2024] Makoto Takamoto, Timothy Pradipta, Raphael Leiteritz, Dan MacKinlay, Francesco Alessiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. In *36th Conference on Neural Information Processing Systems (NeurIPS)*, 2024.
- [Tran *et al.*, 2023] Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *Proceedings of the 11th International Conference on Learning Representations*, 2023.
- [Umetani and Bickel, 2018] Nobuyuki Umetani and Bernd Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Transactions on Graphics*, 37(4), 07 2018.
- [Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, 2017.
- [Wang *et al.*, 2021] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40):eabi8605, 2021.
- [Wu *et al.*, 2022] Haixu Wu, Jialong Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Flowformer: Linearizing transformers with conservation flows. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022.
- [Wu *et al.*, 2024a] Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.
- [Wu *et al.*, 2024b] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. In *Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [Xiao *et al.*, 2024] Zipeng Xiao, Zhongkai Hao, Bokai Lin, Zhijie Deng, and Hang Su. Improved operator learning by orthogonal attention. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.
- [Yu *et al.*, 2022] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [Zhao *et al.*, 2024] Zhiyuan Zhao, Xueying Ding, and B. Aditya Prakash. PINNsformer: A transformer-based framework for physics-informed neural networks. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, 2024.