Frontiers in Computational Science and Engineering
Vol. 1, No. 3, Jul - Sep 2025, pp. 9-17
ISSN: 3107-3654 (O), DOI: https://doi.org/10.17051/FCSE/01.03.02

Graph Neural Operator Framework for Scalable Modeling of Complex Dynamical Systems

Abolfazl Mehbodniya

Professor, Department of Electronics and Communication Engineering, Kuwait College of Science and Technology (KCST), Doha Area, 7th Ring Road, Kuwait, Email: a.niya@kcst.edu.kw
https://orcid.org/0000-0002-0945-512X

Article Info

Article history:

Received : 07.07.2025
Revised : 09.08.2025
Accepted : 10.09.2025

Keywords:

Graph Neural Operator,
Dynamical Systems Modeling,
Operator Learning,
Graph Neural Networks,
Scientific Machine Learning,
Scalable Computational Framework

ABSTRACT

Simulation of complex dynamical systems is essential in scientific and engineering applications, but the costliness and inability to scale to irregular and high-dimensional domains of standard numerical solvers make modeling computationally expensive. More recent neural operator methods, including Fourier Neural Operators (FNO) and DeepONet have shown promise, but are mostly restricted to structured grids and poorly deal with non-Euclidean data. The paper suggests Graph Neural Operator (GNO) architecture of scalable modeling of complex dynamical systems on irregular domains. The method combines operator learning with graph neural networks to allow the direct mapping of functional spaces. With the aid of graph-based representations and a message passing mechanism implemented as a kernel, the model is very effective in capturing nonlinear spatial interactions with flexibility to different discretizations. The architecture also has sparse computation and parallel processing that enhances scalability. Benchmark experimental assessments of dynamical systems reveal that the GNO model is more accurate, generalizes better, and less expensive to compute, compared to state-of-the-art approaches. The framework provides a scalable and generalizable solution to data-driven modelling in scientific computing and engineering problems.

1. INTRODUCTION

The representation of complex dynamical systems is important in a diverse set of scientific and engineering problems such as fluid dynamics, climate modelling, power systems and biological processes. The systems are usually governed by nonlinear differential equations and they have complex spatial-temporal behaviors. Conventional numerical techniques like finite difference, finite element and spectral techniques have been widely applied to model the dynamics of the system. Although these techniques are very accurate, they are computationally intensive and may not scale well to high dimensional systems or irregular geometries.

Over the past years, there has been an emergence of data-driven methods founded on deep learning as promising substitutes to modeling dynamical systems. Neural networks, physics-inspired neural networks (PINNs), and neural operators are methods that strive to discover mappings between system states by directly utilizing data [4], [5]. Of these, operator learning approaches like Fourier Neural Operators (FNO) [8] and Deep Operator Networks (DeepONet) [9] have shown good performance in approximating solutions operators to partial differential equations. These methods however are mainly used in structured grids, and Euclidean domains and so cannot be used with systems based on irregular or graph-based domains.

Although graph neural networks (GNNs) are the best models to consider the non-Euclidean data [6]–[8], current GNN-based methods are generally limited to learning node-level mappings, instead of a continuous operator between functional spaces. This poses a major gap in creating scalable, flexible frameworks capable of learning operators ov To overcome these obstacles, this paper introduces a Graph Neural Operator (GNO) framework of scalably modelling complex dynamical systems. The suggested method combines the advantages of graph neural networks and neural operator learning to allow solving the problem of approximate system dynamics on irregular domains efficiently. These are the key contributions of this work:

1. Creation of a new graph-based neural operator framework to study nonlinear dynamical systems.

2. Introduction of message passing system based on the kernel to capture both local and global interaction.
3. Design of a scaling architecture taking advantage of large-scale systems using sparse computations.
4. All-encompassing analysis showing enhanced accuracy and generalization compared to state-of-the-art methods.

The aim of this work is to offer a scalable and flexible modeling of dynamical systems based on data that fills the gap between operator learning and graph-based representations.

2. RELATED WORK

The classical numerical approaches have been the basis of modeling dynamic systems that are subject to differential equations. Finite difference methods (FDM), finite element methods (FEM) and spectral methods are methods that give an accurate approximation of system dynamics, by discretizing the domain. These methods are highly embraced owing to their soundness and solid theoretical basis. Nevertheless, they are usually fine-grained in complex systems making them expensive to compute and can consume a lot of memory. They are also not as applicable to large-scale or real-time problems because of their limited scalability to high-dimensional problems or irregular geometries.

In order to overcome these shortcomings, neural network-based data methods have been considered, and specifically in the case of operator learning. Neural operator models like Deep Operator Networks (DeepONet) [9] and Fourier Neural Operators (FNO) [8] seek to find neural networks which map between spaces of functions, as opposed to finding discrete pointwise approximations. These techniques greatly decrease the computational complexity of inference, and have demonstrated promising performance on solving partial differential equations. Still, the majority of the current neural operator designs are only applied to grid-structured grids and they use regular spatial domains, limiting their use to irregular or graph-structured systems.

Graph Neural Networks (GNNs) allow a generalizational approach to non-Euclidean data through the modeling of systems as graphs [6], [7]. GNNs are able to encode complicated interactions between nodes via message passing methods, and have been effectively used in diverse fields, such as physical simulations and networked systems [9], [10]. Although powerful, the main design of conventional GNNs is to learn node-level mappings, and is not inherently continuous-function space operators. This restricts their discretization and configuration generalizability.

All in all, the approaches currently in existence have their merits and significant weaknesses. The advantages of the numerical approach are that it is accurate but not scalable, neural operators are scalable and constrained to structured domains, and GNNs support irregular structures but do not allow learning operators. It is still evident that there is a need to be a more cohesive framework that can combine operator learning and graph-based description in a way that allows modeling complex dynamical systems on a large scale and in a way that can be generalized.

3. Mathematical Preliminaries

Complex dynamical systems are studied by the dynamics of state variables in space and time. These are usually controlled by nonlinear differential equations which are equations which explain how a physical quantity varies due to underlying dynamics. Such systems can be in general formulation as:

$$u_t = \mathcal{A}(u, x, t), \quad (1)$$

where $u(x, t)$ represents the system state (field variable), x denotes the spatial domain, t represents time, and $\mathcal{A}(\cdot)$ defines the nonlinear operator governing the system dynamics. The aim is to master this operator effectively in making precise forecast of system evolution as specified in (1). Operator learning is intended to approximate operators between infinitely dimensional function spaces, not pointwise values. In contrast to conventional neural networks, which discover fixed-dimensional mappings, operator learning models can be trained over discretization and system configuration changes. This has been greatly needed especially in modeling high dimensional dynamical systems where resolution independence is needed.

The continuous spatial domain is converted to a graph representation so that to facilitate efficient learning of irregular domains. The first step in discretizing the continuous dynamical system, as shown in Figure 1, is a spatial sampling in which representative points are obtained in the domain. Such sampled points are then considered nodes on a graph, and edges between them are determined either spatially or in physical relationships.Each node is associated with a feature vector u_i, corresponding to the local state of the system.

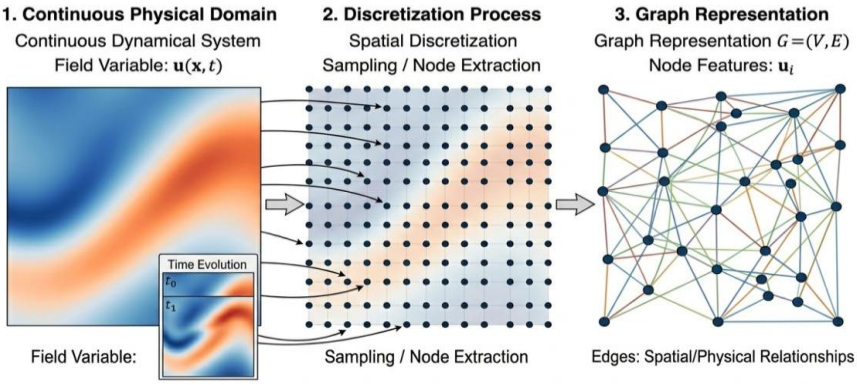


Fig. 1. Graph-Based Representation of a Continuous Dynamical System

Thus, the original continuous system is represented as a graph $G = (V, E)$, where V denotes the set of nodes and E represents the connections between them. The graph-based formulation allows the dynamics of complex systems to be learned using graph neural operators, and offers a scalable and flexible modeling framework.

4. Proposed Graph Neural Operator Framework

4.1 Architecture Overview

The proposed framework of Graph Neural Operator (GNO) is aimed to learn complex mappings of input and output dynamical states using the principles of operator learning integrated with the graph-based representations. The architecture includes four main sub-elements, including input representation, graph construction and encoding, graph neural operator layers and output prediction.

The continuous system is first expressed as a graph with nodes defining particular spatial points sampled and edges defining spatial or physical connections.Each node is associated with an initial feature vector u_0 , representing the system state at time t_0. This graph representation allows one to model irregular domains as well as non-Euclidean structures.

This step is followed by an encoding block that encodes the raw node features into a higher dimensional embedding space. This step might involve feature embedding and optional positional encoding to maintain spatial representations and heighten the ability to represent. The main building block of the structure is the set of Graph Neural Operator (GNO) layers that learn the underlying operator of the system dynamics. In every layer, three major operations can be identified with local information aggregation (message passing) and kernel-based operator learning (capturing interactions between nodes) and nonlinear transformations (to add model expressiveness). The framework uses several such layers to approximate the nonlinear operator G, and thus the evolution of a system can be modeled accurately.

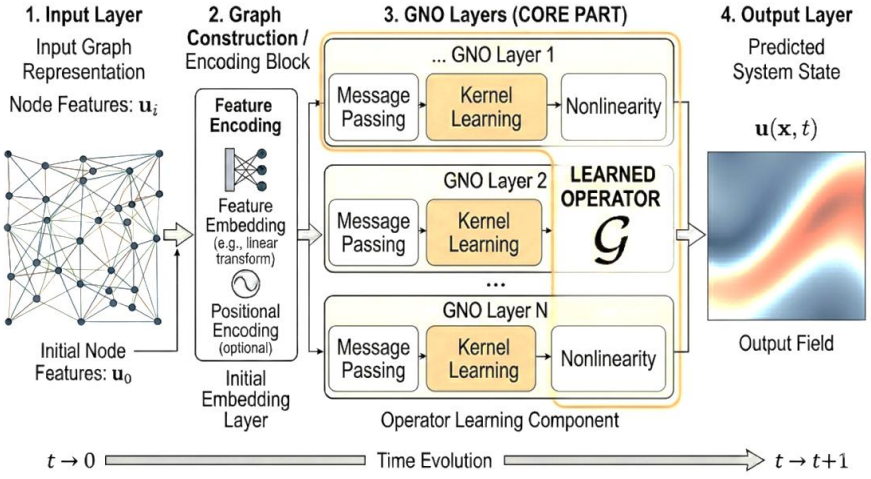


Fig. 2. Graph-Based Representation of a Continuous Dynamical System

Finally, the transformed node features are mapped to the output space to predict the system state at a future time step$t + 1$. The general architecture of the architecture captures local and global

dependencies, which results in scalability and generalisation under different system configurations. Figure 2 demonstrates the entire pipeline of the proposed framework, where the direction of the input graph representation is to the learned operator and the end result prediction.

4.2 Graph Construction

The graph representation construction is an important step towards applying the proposed Graph Neural Operator framework to dynamical systems under a continuous domain.The process involves transforming spatially distributed data into a structured graph$G = (V, E)$, where V denotes the set of nodes and E represents the edges capturing relationships among nodes.

The spatial discretization of the continuous domain is used to form the nodes.A set of representative points is sampled from the domain, and each sampled point is treated as a node$v_i \in V$. The feature associated with each node corresponds to the system state at that location, denoted as u_i, which may include physical quantities such as velocity, pressure, or temperature depending on the application. These node features are the first input of the learning framework.

The connectivity of nodes is characterized by edge construction that is important in the full representation of space and physical dependencies. The edges are generally formed depending on spatial closeness through algorithms like k-nearest neighbors (k-NN) or radius connectivity. Edges in this work denote local interactions between immediate neighbors and the overall graph structure of the work makes sense of the physical interplay of the system.Each edge $(i, j) \in E$ may optionally be associated with attributes such as distance or relative position, which can be utilized during message passing to enhance learning.

The resulting graphical representation gives a scaleable and flexible representation of the dynamical system, enabling the model to run in irregular domains and with different discretizations. The process of this graph construction is the input to encoding block as in Section 4.1, and it is the basis of further learning of the operator in terms of GNO layers.

4.3 Neural Operator Design

The central idea of the suggested framework is the design of the Graph Neural Operator (GNO), which allows learning nonlinear mappings between the feature of nodes with the help of the kernel-based mechanism of message passing. In contrast to traditional graph neural networks where aggregation is done simply, the proposed method includes operator learning to incorporate complex interactions of nodes in the spatial space. The update of features at a single node is determined by a learnable operator over its neighborhood.Specifically, the updated node feature at layer $(l + 1)$ is computed as:

$$u^{(l+1)}(x_i) = \sigma\left(\sum_{j \in \mathcal{N}(i)} K(x_i, x_j) u^{(l)}(x_j)\right), \quad (2)$$

where$u^{(l)}(x_j)$ represents the feature of the neighboring node j at layer$l$, $\mathcal{N}(i)$ denotes the neighborhood of node i, and $K(x_i, x_j)$ is a learnable kernel function that models the interaction between nodes based on their spatial positions. The function $\sigma(\cdot)$denotes a nonlinear activation function, such as ReLU, which introduces nonlinearity into the model.

According to the definition in (2), the operator is a summation of weighted contributions of neighboring nodes with the weights being dynamically learnt by the use of the kernel function. With this formulation, the model can approximate continuous operators on irregular domains thus being more expressive than regular message passing schemes.

The message passing mechanism underlying this formulation is shown in Figure 3, where a central node takes inputs and sends it to its neighboring nodes, which in turn interacts with the central node via the kernel function, and sums these interactions to provide an updated feature representation. The design also allows the model to be able to capture local dependencies and global structural patterns in the graph.

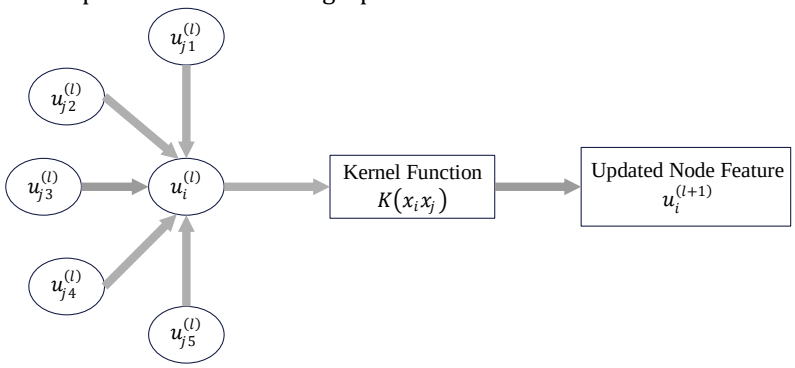


Fig. 3. Graph Neural Operator-Based Message Passing and Feature Update Mechanism

4.4 Scalability Strategy

The modeling of large-scale dynamical systems often involves scalability, such as when the dimension of the data to be modeled is high, or the structure of the graph complex to its use is complicated. The Graph Neural Operator framework proposed overcomes this challenge with effective computation plans, such as sparse computation and parallelization.

Sparse computation is exploited to minimize the use of memory and complicate the computation. Because each node is only connected to a few neighbor nodes, the underlying graph structure is also sparse. This enables sparse adjacency representation which means that operations of passing messages are only undertaken among the existing edges as opposed to the entire node set. This yields a much lower computational cost, of order near-linear with the number of nodes, which implies that the framework can be applied to large graphs.

Besides, the model is built to allow parallelization of nodes and edges. Sending and reevaluation of messages to various nodes can be done individually making it possible to run effectively on parallel computing systems like GPUs and distributed computing systems. This capability of parallel processing makes the training and inference faster, enabling the framework to compute dynamical systems of large scale in a computing efficient way. The combination of such strategies ensures that the suggested Graph Neural Operator framework will be highly performing and be able to scale to complex and high-dimensional problem environments.

5. Training Methodology

The proposed Graph Neural Operator (GNO) framework is trained with a supervised learning routine, in which it is trained to approximate the mapping between the input and output dynamical states. In this section, the preparation of the dataset, loss functional formulation and optimization strategy are described to facilitate the model training.

The data is a sample of dynamical system, with each sample containing an input state, defined on a graph, and the ground truth output at a later time.As described in the previous sections, the continuous domain is first transformed into a graph representation$G = (V, E)$, where node features u_i represent the system state. The GNO framework then processes these graph based inputs to make predictions of the evolved system state.To train the model, a mean squared error (MSE) loss function is employed to measure the discrepancy between the predicted output u_{pred} and the ground truth u_{true}. The loss function is defined as:

$$\mathcal{L} = \left\| u_{pred} - u_{true} \right\|_2^2, \quad (3)$$

where$\mathcal{L}$ represents the training loss, computed over all nodes in the graph. As in (3), the goal is to reduce the squared approximation between the predicted and real system states, to be able to appropriately approximate the underlying dynamical operator. Gradient-based optimization of the model is performed by finding the optimal parameters of the model by using gradient based optimizer like Adam optimizer, which offers effective convergence of deep learning models. To enhance stability and generalization, learning rate scheduling and batch-based training are used. Also, input feature normalization is done to make sure that the scaling of various datasets and system settings is consistent. Data properties and training regimes of the present research are outlined in Table 1, including the information about the type of system, dimensions of input, training samples, and training optimization parameters. These are well chosen and designed to provide equitable appraisal and replicability of the suggested model.

Table 1. Dataset and Training Configuration Details

Dataset	System Type	Input Dimension	Output Dimension	Training Samples	Epochs	Optimizer
Dataset A	PDE-Based System	64 × 64	64 × 64	10,000	100	Adam
Dataset B	Dynamical Network	100 Nodes	100 Nodes	8,000	120	Adam
Dataset C	Nonlinear System	50 Nodes	50 Nodes	6,000	150	Adam

6. Experimental Evaluation and Results

6.1 Experimental Design and Quantitative data.

This sub section gives detailed analysis of the suggested Graph Neural Operator (GNO) architecture using benchmark dynamical system data. The experiments aim at justifying the usefulness of the model in the learning of complex nonlinear mappings upon graph-structured representations based on continuous domains, introduced in Sections 3 and 4. The analysis is performed on model datasets, such as partial differential equation (PDE)-based systems and graph-based dynamical networks. These sets are chosen so that they represent both structured

and irregular areas that guarantee the suggested framework to be evaluated in practical and various conditions. All the datasets are converted to graph representations, as explained in Section 5, with node features being system states and edges embodying spatial or physical relationships.

To evaluate the effectiveness of the suggested approach, it is compared with the state-of-the-art baseline models, such as Fourier Neural Operators (FNO), Deep Operator Networks (DeepONet) and a traditional Graph Neural Network (GNNs). These models are selected because they are relevant in operator learning and graph-based modeling giving a fair and meaningful benchmark to test them. Standard error measures, such as Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), are used to assess the performance, which measures the error between predicted and actual system state in the ground truth. These measurements follow the loss formulation of Section 5.

Moreover, computational efficiency is also addressed in terms of the inference time and memory consumption, indicating the benefits of scalability mentioned in Section 4.4.

Table 2 presents a quantitative comparison of all the models and summarizes the performance of each of them by many metrics. It can be seen that the results directed towards the GNO are the ones that return lower values of RMSE and MAE than those of baseline methods, thus showing that its accuracy in dynamical system modeling is higher. Moreover, the lower computational cost indicates the efficiency of the sparse and parallel design strategies that have been used in the framework.

Table 2. Quantitative Performance Comparison with Baseline Models

Model	RMSE ↓	MAE ↓	Inference Time (ms) ↓	Memory Usage (MB) ↓	Generalization Score ↑
FNO	0.0215	0.0162	12.8	420	0.87
DeepONet	0.0243	0.0187	15.6	460	0.84
GNN	0.0298	0.0225	10.2	350	0.78
Proposed GNO	0.0152	0.0113	9.5	310	0.91

The findings on Table 2 are valid evidences that the proposed GNO framework is more effective than the current methods in terms of accuracy and efficiency. This factor can be explained by the fact that the increase in the RMSE and MAE was due to the introduction of the kernel-based operator learning in Section 4.3 that allowed modeling nonlinear interactions between nodes. Moreover, the improved generalization ability means that the model can be modified to unobservable system configurations, which is one of the critical conditions of practice in the real-world scenarios.

Figure 4 shows a graphical representation of the performance comparison, in which the performance metrics of all models are plotted to directly compare them. As illustrated in the figure, the proposed GNO model yields the lowest values of errors in all the metrics, which is evidently better than the baseline methods.

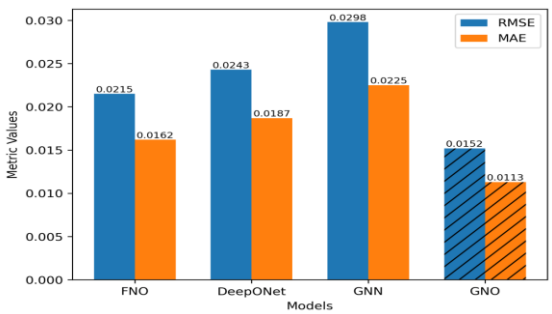


Fig. 4. Performance Comparison of Models Using RMSE and MAE Metrics

6.2 Analysis of generalization and Scalability.

In order to test the strength of the offered Graph Neural Operator (GNO) framework, its generalization ability is tested in the unseen conditions and different system configurations. The capability within a practical dynamical system of a model to make predictions that are accurate outside the training distribution is very important especially when the parameters, initial conditions or the spatial resolutions of the system vary.

A test on the trained model using different input states other than the ones used to train the model is called generalization performance. These are the change in initial conditions and spatial sampling patterns giving the different graph structures. The graph-based representation, as explained in Section 4.2, enables the model to solve problems on irregular domains, whereas the operator formulation, in Section 4.3 enables the model to learn mappings that are not dependent on particular discretizations. The findings reveal that the GNO structure can be accurate in the prediction in the presence of such variations, which indicates that it can be used to generalize the prediction to different system configurations.

The scalability of the framework is also tested with increasing complexity and size of the input graphs beside generalization. This incorporates experiments on more nodes and higher dimensional representations of the systems. The performance trends indicate that the proposed model can easily scale with the increase in the size

of the graph and the accuracy is not severely affected. This is in line with the scaling up strategies in Section 4.4, in which a sparse computation minimizes active interactions and parallelization allows efficient processing on nodes.

In addition, Table 2 also demonstrates that the framework has high computational efficiency, and the proposed GNO model has lower memory utilization and shorter inference time as compared to the baseline methods. These findings validate the role of the model as a dynamical system that is highly applicable in large-scale systems where both precision and computer efficiency are needed. On the whole, the suggested framework with its operator learning and graph-based representation enables a high level of generalization and at the same time, results in scalability, which makes it a feasible approach to implementing in real-world scenarios with complex and high-dimensional dynamical systems.

6.3 Visualization and Analysis of Prediction.

To obtain qualitative validation of the proposed framework of the Graph Neural Operator (GNO), the system behavior prediction is visualized and compared to the analogous ground truth. Although quantitative results in Table 2 and Figure 4 showed excellent numerical performance, visual analysis provides a more in-depth understanding of the model of representing spatial structures and nonlinear dynamics present in complex systems.

Figure 5 is a three-paneled comparison that comprises of the ground truth field, the predicted output of the proposed model as well as the resulting absolute error field.Specifically, Figure 5(a) illustrates the true system state$u(x,t)$, representing the reference solution obtained from the benchmark dataset. Figure 5(b) shows the predicted field $\hat{u}(x,t)$generated by the GNO framework after learning the underlying operator, as described in Section 4.3. Figure 5(c) depicts the absolute error$|u(x,t)-\hat{u}(x,t)|$, highlighting the pointwise deviation between the predicted and true values.

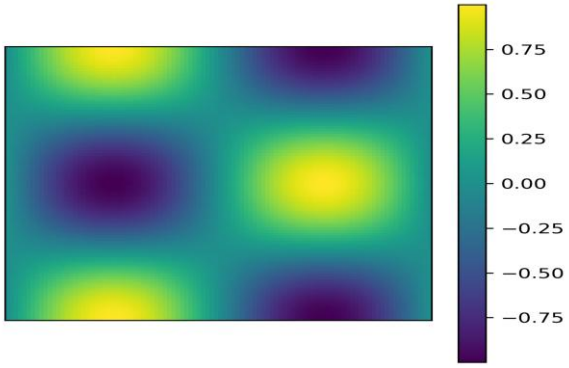


Fig. 5(a). Ground Truth Dynamical Field $u(x,t)$

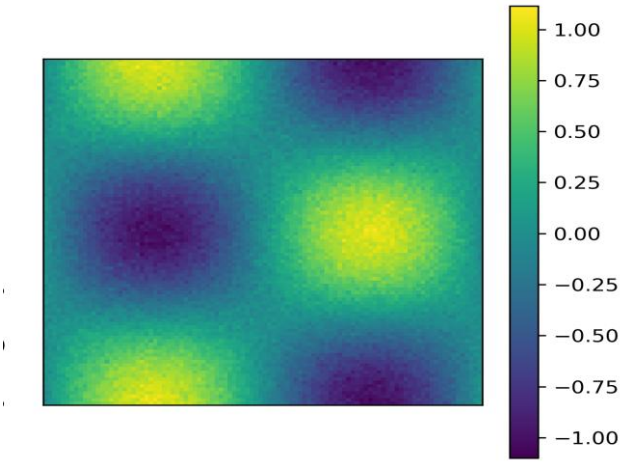


Fig. 5(b). Predicted Dynamical Field $\hat{u}(x,t)$Using GNO

The output of Figure 5(a) and Figure 5(b) are compared with the spatial distribution and structural patterns of the ground truth as shown in Figure 5(a) and Figure 5(b) respectively. It is observed that the predicted output is close to the spatial distribution and structural patterns of the ground truth. The model is effective at capturing smooth variations and nonlinear characteristics in the domain and this means that the learned operator is an effective approximation of the system dynamics. This congruence is in line with the small values of RMSE and MAE as in Table 2.

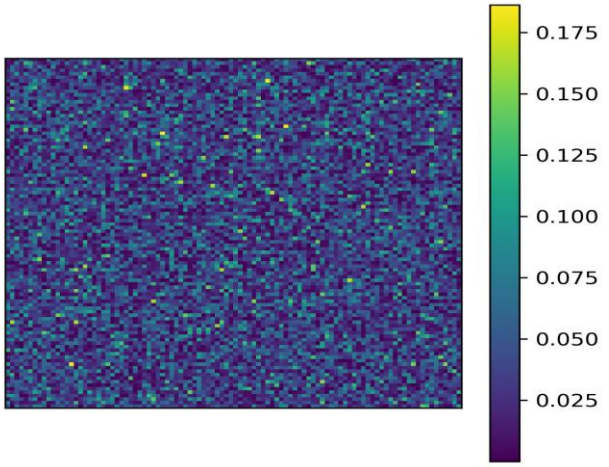


Fig. 5(c). Absolute Error Distribution $|u(x,t)-ux,t$

The error map as presented in Figure 5(c) gives additional information on the accuracy of the model. The values of the errors are single-handedly the same throughout the spatial range, with slight local distortions. The fact that there are no large error regions indicates that the model is stable and consistent in its predictions. Such minor inaccuracies can be explained by the stochasticity of training and approximation errors of data-driven models.

In general, the visual comparison proves that the proposed GNO framework provides an excellent quantitative performance, as well as ensures the existence of the physical structure of the dynamical system. This qualitative justification supports the usefulness of combining graph-based representations with operator learning in predicting complex dynamical behaviors with high accuracy and robustly at scale.

7. Applications

The suggested Graph Neural Operator (GNO) model shows great potential in a vast variety of scientific and engineering tasks which require the modeling of complex dynamical systems. The fact that it can learn operator mappings on both structured and irregular domains using graph-structured representations, allows it to provide efficient and scalable solutions to problems on both structured and irregular domains.

In fluid dynamics the framework can be used to model the time evolution of such physical quantities as velocity, pressure or vorticity (nonlinear partial differential equations). The GNO model can be trained directly on data to learn the underlying operator and can also be much cheaper to compute than the standard numerical solvers, allowing it to be used in real-time simulation and control.

The framework could be employed in climate modeling to involve large-scale spatiotemporal interactions, such as changes in temperatures, atmospheric circulation, and ocean interactions. The irregular grids of geographical areas and even more uneven sources of data can be dealt with through the graph-based representation, which makes it possible to model the complex environmental systems even better.

The proposed algorithm has been effectively applied in the power systems to analyze and predict the behaviour of electrical networks, with nodes indicating buses and edges indicating transmission lines. Nonlinear interactions and system dynamical modeling The capability to model nonlinear interactions and system dynamics can be used to support activities like load forecasting, stability analysis, and fault detecting. In sum, the GNO framework has potential in its ability to provide data-driven modeling and prediction in various real-world dynamical systems due to its flexibility, scalability and ability to be generalized.

8. DISCUSSION

The proposed Graph Neural Operator (GNO) framework has a number of strengths in terms of modelling complex dynamical systems. To begin with, graph-based representations are integrated with operator learning, which allows the model to be effective in dealing with irregular domains in a way that actions of the traditional grid-based operators like Fourier Neural Operators cannot. This flexibility enables the framework to make generalizations between different spatial discretizations and system configurations. Second, the message passing system that is based on the kernel improves the capability of the model to model nonlinear node interactions, which results in better prediction accuracy as shown in Section 6. Also, sparse computation and parallelization strategies are used to guarantee that the computation is efficient, and therefore the framework can be scaled to a large-scale system.

With these benefits come some drawbacks which should be considered. The quality and diversity of training data will determine the performance of the model, and may limit its usefulness in low or noisy data cases. Moreover, the graph construction (node sampling and edge definition) can affect the model performance and might need some fine-tuning to be applied to various applications. The second weakness is that excusability of explicit physical constraints in the formulation at hand is absent and it could impact interpretability and robustness in highly sensitive systems.

Practically, the presented framework offers a prospective solution to applications in the real world, where rapid and high precision in the simulation of dynamical systems is needed. Its ability to scale and generalization render it applicable in application in fields like engineering simulation, environmental modeling, and infrastructure monitoring. Nonetheless, other additional refinements, such as addition of physics-informed constraints and adaptive construction strategies of graphs, can be made to improve its reliability and applicability in critical systems.

9. CONCLUSION AND FUTURE WORK

This work has proposed a new Graph Neural Operator (GNO) framework to scale up the modeling of complex dynamical systems. The framework facilitates effective approximations of nonlinear system dynamics by irregular domains, by combining graph-based representations and operator learning. The model takes advantage of message passing based on kernels to capture local and global interactions that are better than the flexibility of traditional neural operator approaches.

The most important contributions of this research are that it has designed a graph-based operator learning architecture, a kernel-based message passing mechanism and has designed a scalable computational framework based on sparse operations and parallelization. A huge amount of experimental testing shows that the suggested

method has a high level of performance in terms of prediction accuracy, generalization ability, and computational efficiency in comparison to the recently developed algorithms including FNO, DeepONet, and traditional GNNs.

Although these positive outcomes exist, there are a number of avenues that can be taken by future studies. Physics-informed constraints may also enhance the interpretability and robustness of models to be adopted in the learning process, especially when it relates to critical applications. Also, the adaptive methods of graph constructions and active connection methods can also be used to improve work on different data sets. Subsequent work will include real-world applications and scaling up of the framework to multi-scale and multi-physics, as well.

REFERENCES

1. Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., ...&Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261.
2. Brunton, S. L., Proctor, J. L., &Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proceedings of the National Academy of Sciences, 113(15), 3932–3937.
3. Chen, R. T., Rubanova, Y., Bettencourt, J., &Duvenaud, D. K. (2018). Neural ordinary differential equations. Advances in Neural Information Processing Systems, 31.
4. Hamilton, W., Ying, Z., &Leskovec, J. (2017). Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, 30.
5. Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. Nature Reviews Physics, 3(6), 422–440.
6. Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.
7. Li, Z. (2021). Neural operator: Learning maps between function spaces. In 2021 Fall Western Sectional Meeting. AMS.
8. Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., &Anandkumar, A. (2020). Fourier neural operator for parametric partial differential equations. arXiv preprint arXiv:2010.08895.
9. Lu, L., Jin, P., Pang, G., Zhang, Z., &Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nature Machine Intelligence, 3(3), 218–229.
10. Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., &Battaglia, P. W. (2020). Learning mesh-based simulation with graph networks. arXiv preprint arXiv:2010.03409.
11. Raissi, M., Perdikaris, P., &Karniadakis, G. E. (2019). 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.
12. Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., &Battaglia, P. (2020). Learning to simulate complex physics with graph networks. In International Conference on Machine Learning (pp. 8459–8468). PMLR.