Prompting In-Context Operator Learning with Sensor Data, Equations, and Natural Language

Liu Yang, Tingwei Meng, Siting Liu, Stanley J. Osher*

Department of Mathematics, UCLA, Los Angeles, CA 90095, USA

In the growing domain of scientific machine learning, in-context operator learning [1] has demonstrated notable potential in learning operators from prompted data during inference stage without weight updates. However, the current model's overdependence on sensor data, may inadvertently overlook the invaluable human insight into the operator. To address this, we present a transformation of in-context operator learning into a multi-modal paradigm. We propose the use of "captions" to integrate human knowledge about the operator, expressed through natural language descriptions and equations. We illustrate how this method not only broadens the flexibility and generality of physics-informed learning, but also significantly boosts learning performance and reduces data needs. Furthermore, we introduce a more efficient neural network architecture for multi-modal in-context operator learning, referred to as "ICON-LM", based on a language-model-like architecture. We demonstrate the viability of "ICON-LM" for scientific machine learning tasks, which creates a new path for the application of language models.

1 Introduction

In [1], the authors introduced in-context operator learning as a new paradigm for operator learning. As in classic operator learning tasks, an operator maps a single input function or a tuple of input functions, referred to as the "condition", to an output function, referred to as the "quantity of interest (QoI)". In practice, we usually have no access to the analytical expression of these functions, but instead can collect sensor data in the form of key-value pairs, where the keys are discrete function inputs and the values are the corresponding function outputs.

A wide variety of scientific machine learning tasks can be conceptualized as operator learning problems. Consider the task of solving partial differential equations (PDEs) for instance, where the coefficient function serves as the condition, and the solution is the QoI. Conversely, for inverse problems, these roles are swapped. When dealing with problems that involve temporal evolution, the condition can be the initial state, while the QoI represents the state at a

*Corresponding Author: sjo@math.ucla.edu.

subsequent point in time. For control problems, the condition could correspond to the cost function and the initial state, while the QoI embodies the control signal. It's evident that the relationship between the condition and the QoI is highly contingent on the operator, which diverges based on the task at hand and the particular system in question.

In-context operator learning aims to train the neural network to acquire the ability to learn the operator from condition-QoI pairs, referred to as "examples", and apply the learned operator to the new condition to predict the corresponding QoI. This approach differs from classic operator learning approaches [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12] where the neural networks are limited to approximate specific operators, and thus need to be trained every time a new operator is encountered. In contrast, in-context operator learning trains the neural network as an operator learner, which aims to conduct the above learning process in the inference stage without weight update. This advancement offering a "train-once-apply-multiple" paradigm for a broad array of tasks, paves the way for large-scale foundation models [13], and potentially even artificial general intelligence, to be employed in scientific machine learning tasks.

The study by [1] showcases the successful implementation of in-context operator learning, which relies solely on sensor data. However, a crucial aspect of scientific machine learning is overlooked in this approach, namely, the human knowledge of the operator, which can span from nebulous natural language explanations to explicit differential equations. There's a strong case for incorporating such knowledge into the learning system alongside sensor data, as this could potentially enhance learning performance. If the human understanding of the operator is sufficiently detailed, the system might require fewer examples to learn the operator. Theoretically, it might even enable zero-shot learning, where the operator could be learned and utilized without the need for any examples.

Past research on the topic of scientific machine learning typically integrates human knowledge into the learning system by designing special loss functions or neural network architectures based on the differential equations or symmetry/conservation laws that govern the system. While these approaches have witnessed significant success, they are not without limitations. Firstly, it may not always be practical to design special loss functions or architectures, as the system might not be fully understood by humans, or the operator might be too complicated to be described by equations. Secondly, these bespoke loss functions or architectures are tailored for specific systems or tasks. When confronted with a new system, there is a requirement not only to design new loss functions or architectures but also typically to retrain the neural network.

In this paper, we explore an entirely different approach to infuse human knowledge into the learning system. Inspired by the recent success of large language models, we introduce a new component to in-context operator learning: the "caption". A caption is a string serving as a descriptor of the operator, and can take various forms such as equations written in LaTeX form, natural language descriptions, or a combination of both. Rather than crafting special loss functions or architectures, we simply feed the caption into the neural network as input alongside the examples. We thus evolve the in-context operator learn-

ing to be multi-modal, meaning that the neural network can learn the operator from sensor data, captions, or a combination of both. The diagram in Figure 1 illustrates the multi-modal in-context operator learning.

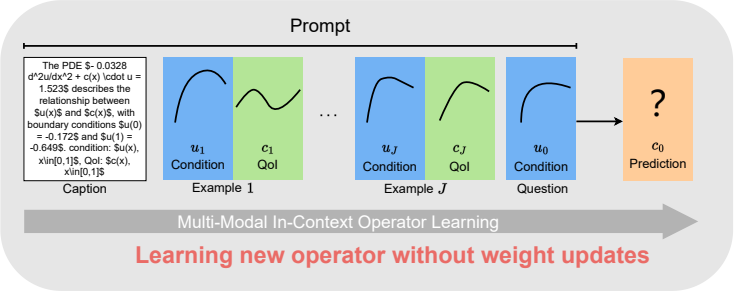


Figure 1: Diagram for Multi-modal in-context operator learning.

This method is substantially more flexible and can accommodate a broader range of systems under the same framework. Moreover, it retains the principle of in-context operator learning, where human knowledge is integrated into the learning system without any weight update.

In addition to evolving in-context operator learning into a multi-modal framework, we have also enhanced the neural network architecture delineated in [1]. The original architecture consists of two transformers [14]: an encoder and a decoder. In each iteration during the training stage, the neural network is solely trained to perform in-context operator learning with a certain number of examples, although this number can vary between different steps. We identified this architecture and training method as inefficient. In this paper, we propose a revamped architecture that merges the encoder and decoder into a single transformer, training the neural network to execute in-context operator learning with varying numbers of examples concurrently in each training step, ranging from zero (with a caption) or one (without a caption) up to the maximum capacity.

Interestingly and intentionally, the revised architecture and training scheme bears a stronger resemblance to generative language models like GPT-3 [15]. We merely utilize a standard transformer, appending a linear layer both before and after the transformer to map the input and output to the required dimensions, much akin to existing language models. The training scheme also parallels that of language models, where the neural network is trained to make predictions of QoI in each example based on preceding examples. The main deviation (and also the key challenge) is the necessity to design the input sequence and formulate a specialized mask to accommodate in-context operator learning tasks. Following the name "In-Context Operator Networks (ICON)" introduced in [1], we refer to the revised architecture and training scheme as "ICON-LM", where "LM" stands for "language model".

The adoption of a neural network architecture synonymous with language models is crucial for two reasons. First, it enables us to utilize existing techniques and tools developed for language models. Second, it paves the way for

integrating in-context operator learning with current language models. Contemporary language models are largely designed for natural language processing tasks, with some extending to image-related tasks. It remains unverified whether they can be employed for scientific machine learning tasks. In this paper, we present promising results demonstrating the viability of language model architecture for scientific machine learning, which indicates a new path for the application of language models.

Our contributions are summarized as follows:

1. We transform the in-context operator learning into a multi-modal framework by introducing "captions" as a means to incorporate human knowledge about the operator, in the form of natural language descriptions and equations. Compared with existing approaches like crafting special loss functions or neural network architectures, we demonstrate that this approach is more flexible and general, yet effective in enhancing learning performance and reducing data requirement, even enabling zero-shot learning.

2. We propose a language-model-like architecture for multi-modal in-context operator learning, namely "ICON-LM". Working with a novel input sequence and mask, we demonstrate that this architecture is superior to the original "ICON" architecture introduced in [1] in terms of both learning efficiency and prediction accuracy.

3. The demonstrated viability of language-model-like architecture for scientific machine learning tasks indicates a new path for the application of language models.

2 Related Work

2.1 Operator Learning and In-Context Operator Learning

Numerous neural network methods have been proposed for approximating operators, i.e., mappings that take functions as input and output. The early works of [2, 3] employed shallow neural networks for the approximation of nonlinear operators. A deep neural network approach to tackle parametric PDE challenges was suggested in [4]. PDE-Net, as presented in [6] enables forward predictions of PDE solutions using the inferred forward map. The study in [5] presented a Bayesian method to address uncertainty quantification in stochastic PDE scenarios. The Deep Operator Network (DeepONet), referenced in [7], introduces a neural network design that approximates the solution operator, mapping parameters or initial/boundary conditions to their corresponding solutions. The Fourier Neural Operator (FNO) from [9, 10] leverages the Fourier space's integral kernel to approximate the solution operator. Drawing inspiration from neural networks and model reduction, the paper [11] estimates input-output maps between infinite-dimensional spaces for parametric PDEs. Additional contributions can be found in [16, 17, 18, 19, 20].

Recently, a different operator learning paradigm, namely in-context operator learning, is proposed in [1]. Instead of approximating specific operators, in-context operator learning trains the neural network as an operator learner, which can learn and apply the operator in the inference stage without weight update.

2.2 Physics-Informed Machine Learning

In the literature, two approaches are commonly employed to incorporate physical knowledge in neural networks: hard constraints and soft constraints. We refer readers to the survey paper [21] on this topic. Hard constraints involve designing neural network architectures in a way that ensures any solution generated by the network meets the specified constraints (see, for example, [22, 23, 24, 25, 26, 27, 28]). While solutions with specifically designed architectures are guaranteed to be compliant to the physical constraints, creating such architectures demands extensive domain knowledge and may not be easily adaptable to other problems. Additionally, the expressivity and training complexity could be limited in these cases. Soft constraints are implemented by incorporating physics-informed terms into the loss function. For example, [29, 30, 31, 32, 33, 34, 35, 12, 8]. While more flexible in terms of neural network architecture design, this approach still requires precise knowledge of physics in the form of differential equations, variational problems, etc., which are not always available, especially when the system is not fully understood by humans.

In-context operator learning excels at addressing a broad spectrum of physical problems using a single neural network. The limited flexibility and generalizability of the previously mentioned approaches hinder their application to in-context operator learning. This limitation motivates our exploration in this paper, where we introduce a new method to incorporate physical knowledge: through "captions".

2.3 Multi-Modal Models

Unimodal language models solely rely on text data for training, limiting their ability to comprehend the visual world. In contrast, multimodal language models are trained on data in multiple forms, including texts and images, enabling them to understand the visual world. We refer readers to the survey [36] on this topic.

To fuse different modal data, one approach involves combining the extracted features or embeddings from different modal data and then feeding these embeddings into the same model [37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47]. Another approach converts other modal data into language data and uses these language representations as inputs for language models [48]. Some studies combine both techniques, utilizing both extracted features and converted language data as inputs to language models [49, 50].

During the training phase, due to the constraints of computational complexity, many models (such as [45, 46, 38]) freeze the parameters of the language

models and only train the parameters of the other components, like data embeddings or bridging projections. The performance of this training strategy is compared to end-to-end fine-tuning in [37].

In this paper, we integrate embeddings of sensor data and language captions into the same model, instead of converting one form of data into another. Given the model's relatively small size, we train it end-to-end from scratch.

3 GPT-assisted Caption Generation

In this study, we utilize sensor data obtained from [1] to train the ICON-LM model. This data set comprises 19 distinct problem types, including ordinary differential equations (ODEs), partial differential equations (PDEs) and mean-field control problems, each with 1,000 operators, giving us a total of 19,000 operators. Each operator has 100 associated condition-QoI pairs, i.e. "examples".

To perform multi-modal learning, for each operator, we additionally design 20 captions for training and 2 captions for testing. These captions fall into two categories: vague captions and precise captions.

We generate these captions in the following way. First, we employed ChatGPT to generate two sets of captions for each problem type. These AI-generated captions were largely appropriate, though a few necessitated slight manual adjustments. For the vague group, we instruct ChatGPT to use natural language or tell the form of the equation with parameters, but do not tell the actual value of the parameters. For the precise group, we instruct ChatGPT to disclose all parameter values, replacing the actual parameters with unique tokens. These tokens were then substituted with the real parameter values when formulating the input sequence for the ICON-LM model. Note that the parameters are functions for mean-field control problems, we thus discretize the functions to represent them.

For both the vague and precise groups, we generated 11 captions. In each group, the initial 10 captions were used for training, and the last one was used for testing.

To provide a clearer understanding of the captions used, we present some examples in the Appendix.

4 The ICON-LM Model

4.1 Input and Output Sequence

The input of the ICON-LM model consists of the caption as well as multiple pairs of conditions and QoIs.

We first use a pre-trained language model to convert the caption into a sequence of embeddings. We don't pool over the embeddings, as we care not only the general semantics of the caption, but also the details, e.g., the numbers in the equations. In this paper, we use a pre-trained math-aware language model

provided by [51] and published in HuggingFace[1]. This model was initialized from ALBERT-base-v2 model and further pre-trained on Math StackExchange. We chose this model since it added more LaTeX tokens to enable a better tokenization of mathematical formulas. While in principle we can fine-tune the language model during the training of ICON-LM, we found that the fixed pre-trained model already performs well in our experiments. We thus use the pre-trained model without fine-tuning to save computational resources.

Each condition and QoI function is represented by a set of key-value pairs. We found the technique in [1] to be effective in converting the key-value pairs into a sequence of vectors. Specifically, each vector is a concatenation of the key, value, and index column vector which distinguishes conditions and QoIs from different examples. In [1], the index vector is a one-hot vector. Here we design a more compact index vector by utilizing the binary representation of the index. For instance, the index vector for the condition is $[0, 0, 0, 1]^T$ in the first example, $[0, 0, 1, 0]^T$ in the second example, and $[0, 0, 1, 1]^T$ in the third example, so on and so forth. The index vector for the QoI is the negative of the index vector for the condition. By doing so, we can represent $2^N - 1$ examples with an index vector of size N, as opposed to a linear increase using one-hot vectors.

One key difference of the training scheme in this paper compared with [1] is that the learning is performed with varying numbers of examples concurrently in each training step. This process involves the creation of "queries", or vectors representing the keys of the QoI, in addition to the condition vectors and QoI vectors. Such queries are created for each example, instead of only for one example as in [48]. It's important to note that "queries" in this context are distinct from the queries associated with the attention mechanism of the transformer model. The index vectors for queries are the same as those for QoIs, but use different entries to make a distinction.

In table 1, we show the matrix representation of the j-th example for the one-dimensional forward ODE problem, where the condition consists of the control $c\colon [0, T] \to \mathbb{R}$ and the initial condition $u(0)$; the QoI is the state $u\colon [0, T] \to \mathbb{R}$.

Every caption embedding vector is subjected to a transformation via a linear layer, just as every condition, QoI, and query vector undergoes another linear layer transformation. These transformed vectors are subsequently concatenated, and the resulting combined matrix is supplied to the transformer. The transformer's output sequence undergoes a linear layer to align its dimensions with those of the QoI values. Note that in the output sequence, we only keep the ones corresponding to the query vectors, which aim to predict the QoI values for each query. The input/output sequence and the neural network architecture is depicted in Figure 2a.

[1] `https://huggingface.co/AnReu/math_albert`

		condition	QoI	query
key	term	$\begin{pmatrix} 0 & 0 & \dots & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \dots & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \dots & 0 \end{pmatrix}$
	time	$\begin{pmatrix} t_1 & t_2 & \dots & t_{n_j-1} & 0 \end{pmatrix}$	$\begin{pmatrix} \tau_1 & \tau_2 & \dots & \tau_{m_j} \end{pmatrix}$	$\begin{pmatrix} \tau_1 & \tau_2 & \dots & \tau_{m_j} \end{pmatrix}$
	space	$\begin{pmatrix} 0 & 0 & \dots & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \dots & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \dots & 0 \end{pmatrix}$
value		$\begin{pmatrix} c(t_1) & c(t_2) & \dots & c(t_{n_j-1}) & u(0) \end{pmatrix}$	$\begin{pmatrix} u(\tau_1) & u(\tau_2) & \dots & u(\tau_{m_j}) \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \dots & 0 \end{pmatrix}$
index		$\begin{pmatrix} \mathbf{e}_j & \mathbf{e}_j & \dots & \mathbf{e}_j & \mathbf{e}_j \end{pmatrix}$	$\begin{pmatrix} -\mathbf{e}_j & -\mathbf{e}_j & \dots & -\mathbf{e}_j \end{pmatrix}$	$\begin{pmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \end{pmatrix}$
		$\begin{pmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} -\mathbf{e}_j & -\mathbf{e}_j & \dots & -\mathbf{e}_j \end{pmatrix}$

Table 1: The matrix representation for the j-th example for the one-dimensional forward ODE problem. We use $n_j - 1$ key-value pairs to represent c, one key-value pair for $u(0)$, and m_j key-value pairs for u. Note that in the first row, we use the indicator 0 and 1 to distinguish different terms in the condition, i.e., c and $u(0)$. The third row is populated with zeros since there are no spatial coordinates in this problem. The keys for queries are the same as those for QoIs, but the values are populated with zero. $\mathbf{e}_j$ is the column index vector, $\mathbf{0}$ is a zero vector of the same size as $\mathbf{e}_j$.

4.2 Transformer Mask

The neural network is trained to predict the respective QoI value for each query, taking into account the captions, all the conditions and QoIs from previous examples, as well as the condition from the current example. The construction of the transformer mask is guided by the following principles:

1. The query vector is expected to attend to the caption embeddings, all preceding example condition and QoI vectors, all the condition vectors in the current example, and itself.

2. It's crucial that the query does not attend to any QoI vector in the current example, as the QoI values are the target of the prediction.

3. Also, the query should not attend to other queries in the same example, as the predictions should be independent of each other.

4. Owing to the first two constraints, the condition vectors can only attend to the caption embeddings, all the preceding example condition and QoI vectors, and all the condition vectors in the current example. They cannot attend to the QoIs in the current example, to prevent inadvertent leakage of the current example's QoIs to the query.

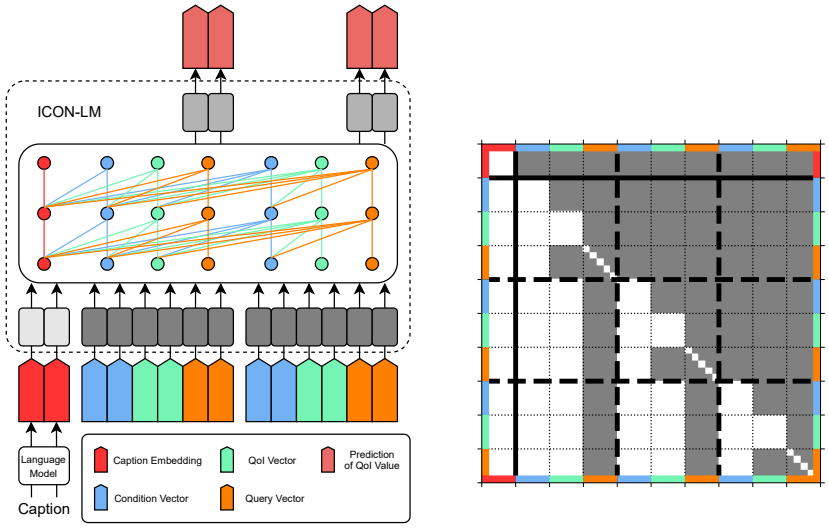
5. The QoI vectors can attend to the caption embeddings, all the condition and QoI vectors in preceding examples, and all the conditions and QoIs in the current example.

6. The caption embeddings can attend to all the caption embeddings, since we don't need to predict the next token as in generative language models.

Consequently, the resultant transformer mask is illustrated in Figure 2b. Verification of the mask, denoted by M, can be carried out by confirming

$MM = M$, which ensures that there is no indirect attention causing unintentional information leakage, like vector c's information being leaked to vector a through a attending to b, and b attending to c.

Lastly, it's worth noting that during batching, we may have to include placeholder vectors in the input sequence to maintain consistent length across a batch. These placeholder vectors need to be ignored in the attention mechanism. We accomplish this by carrying out an element-wise multiplication of the previously constructed mask with another mask specifically designed to indicate these placeholder vectors, in which the columns corresponding to the placeholder vectors are set to zero, and the rest are set to one.



(a) (b)

Figure 2: (a) Depiction of the input/output sequence and neural network structure of ICON-LM. For clarity, we present only two examples. The model of ICON-LM is highlighted by the black dashed enclosure. Grey rectangles with varying colors signify distinct linear layers. The transformer module's attention mechanism is demonstrated by connections between different component blocks in the sequence. (b) The transformer mask for ICON-LM. As illustrated, the mask for three examples can be divided into 10×10 blocks. In this matrix, white represents one, and grey represents zero. Pay attention to the diagonal blocks, indicating that each query attends to itself but not to other queries. Black solid lines separate blocks for captions and examples, while black dashed lines separate individual examples. Along the boundary, varying colors indicate different components – red for caption embeddings, blue for condition vectors, green for QoI vectors, and orange for query vectors, consistent with (a).

4.3 Training and Inference

We train the ICON-LM model to execute in-context operator learning, with the option of including or excluding captions. The loss function generally consists of the mean squared error between the predicted QoI values and the actual data. For training inclusive of captions, the loss function is calculated from the first example prediction up to the last, with the first example prediction being a zero-shot – a prediction solely based on the caption and condition, excluding any other examples. When training without captions, we exclude the caption from the input sequence and calculate the loss function from the second example's predictions to the last, bypassing zero-shot learning as it is not meaningful to predict the QoI value without any example or caption. The total loss comprises the losses from both scenarios.

During the inference stage, the new question condition vectors and query vectors are considered "the last example" and appended to the input sequence. The QoI vector for the new question is not required. The ICON-LM model's output corresponding to these query vectors gives the prediction for the corresponding QoI value.

Note that some components of the input sequence can, or should, be excluded in certain scenarios. This affects the corresponding rows and columns of the masks as well. In this paper, we applied the following four variations:

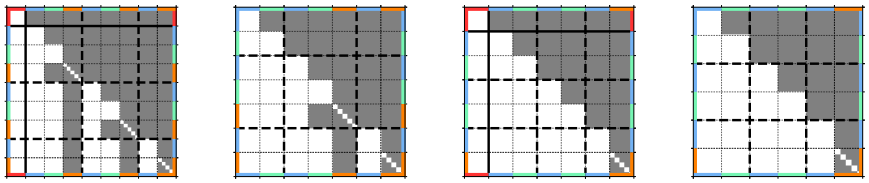
1. **Training with caption**: Here, predictions are needed for all queries. While the full mask described in section 4.2 is applicable, we find that the QoI vectors in the last example are never utilized. Hence, they are omitted from the input sequence to minimize the computational load.

2. **Training without caption**: In this scenario, predictions are required for all queries, except those for the first example. We exclude the caption embeddings, the first example's queries, and the last example's QoI vectors from the input sequence.

3. **Inference with caption**: Here, the focus lies only on the prediction for "the last example", i.e., the new question condition and QoI key. As such, the queries from all preceding examples can be omitted from the input sequence, along with the last example's QoI vectors, which are anyway absent during the inference stage.

4. **Inference without caption**: This case parallels the previous "inference with caption" scenario, except for the omission of the caption embeddings from the input sequence.

These four mask variations are depicted in Figure 3.



(a) (b) (c) (d)

Figure 3: Four transformer mask variants for ICON-LM. (a) Training with caption. (b) Training without caption. (c) Inference with caption. (d) Inference without caption. Varying colors indicate different components along the boundary, which are consistent with Figure 2b.

5 Experiments

5.1 ICON-LM v.s. Encoder-Decoder ICON

This section serves to draw a comparison between the ICON-LM model, the focus of this study, and the original encoder-decoder ICON model introduced in [1]. Note that here we train and test both models without captions.

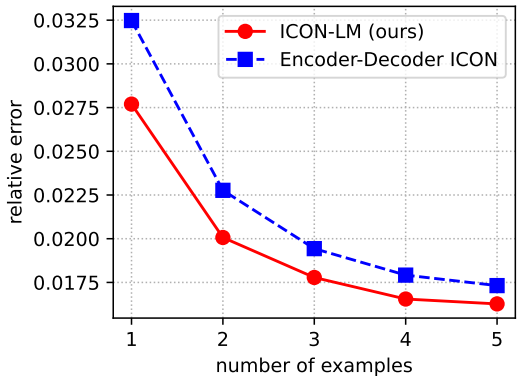


Figure 4: Relative testing errors for cases from one-shot to five-shot learning, without captions. Our ICON-LM outperforms encoder-decoder ICON in all cases.

The dataset used for training and testing, as well as all the training configurations for the encoder-decoder ICON are inherited from [1]. Specifically, the encoder-decoder ICON is trained with n examples and one question, with n randomly chosen from one to five in each step. Meanwhile, ICON-LM is trained with six examples, which means one-shot to five-shot learning concurrently in each step. The encoder-decoder ICON encompasses approximately 31.6 mil-

lion parameters, whereas the ICON-LM operates with nearly half that number, at around 15.8 million parameters. This substantial reduction can be credited to the ICON-LM's simplified architecture, which employs a single transformer roughly equivalent in size to either the encoder or decoder in the encoder-decoder ICON. Both models are trained for 1 million steps, with the same setups for optimizer and learning rate schedule. We put details in the appendix.

The batch size is 32 for encoder-decoder ICON, and 24 for ICON-LM. With such setups, both models take about 19GB GPU memory, and can fit in one NVIDIA GeForce RTX 4090 GPU with 24 GB memory. As for the time consumption, the training takes about 40.5 hours for the encoder-decoder ICON, and about 37 hours for ICON-LM.

We compare the relative testing error averaged over all 19 types of problems for cases from one-shot to five-shot learning. The results are shown in Figure 4. It's clear that ICON-LM outperforms encoder-decoder ICON in all cases.

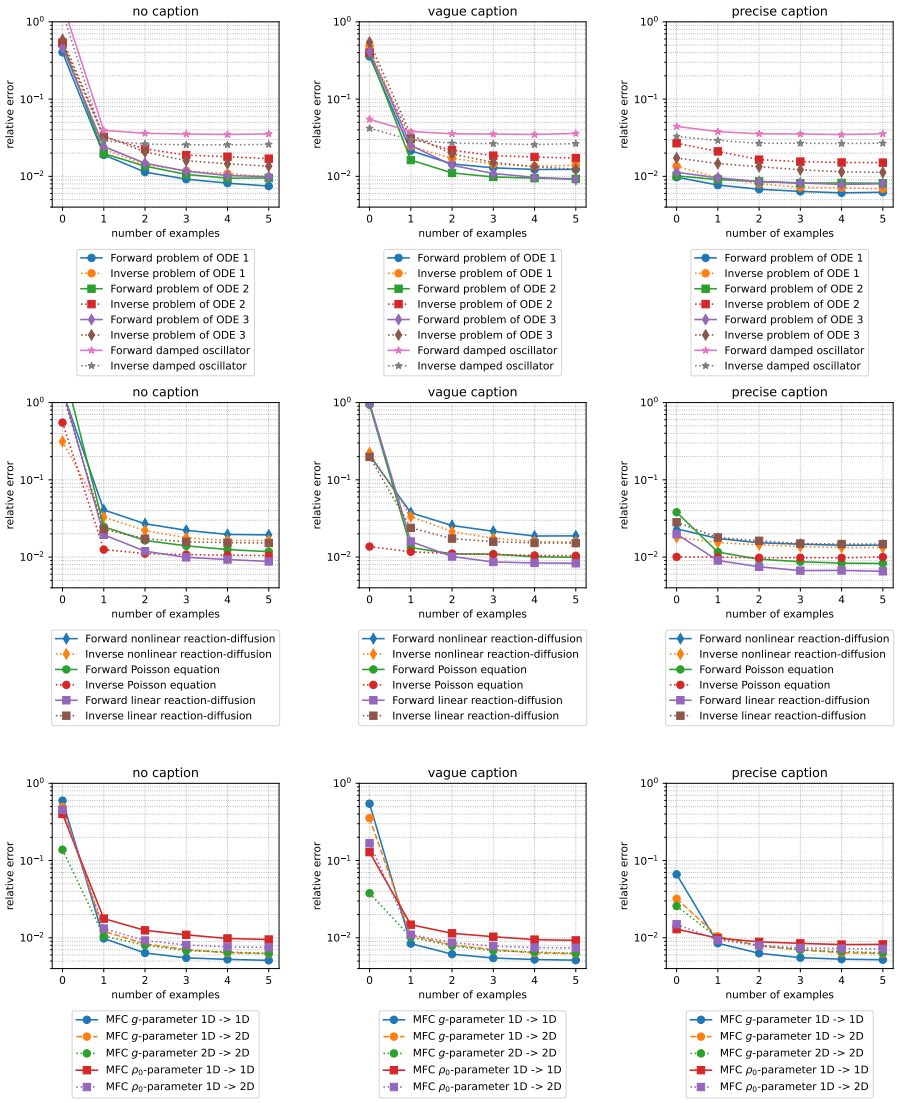
5.2 Multi-Modal In-Context Operator Learning

In this section, we demonstrate the effectiveness of multi-modal in-context operator learning. We train the ICON-LM model with the same setup as in Section 5.1, except that the training loss comprises the losses with and without captions, and that the batch size is set as 32.

For each of the 19 types of problems, we show the relative testing error without captions, with the testing vague captions, and with the testing precise captions in Figure 5. We can see that when the number of examples is small, e.g., zero-shot and one-shot learning, the testing error shows a clear trend of decreasing as we move from no caption to vague caption to precise caption. This indicates that the caption can help the neural network to learn the operator, and the more precise the caption is, the better the learning performance. When the number of examples is large, e.g., four-shot and five-shot learning, the captions don't seem to help much. This is because the neural network has already learned the operator from sufficient examples, and the caption is not needed.

It is worth noting that zero-shot learning with vague captions performs very well for some problems, almost as good as zero-shot learning with precise captions. These include the forward and inverse damped oscillator series problems, the inverse Poisson equation problem, and the mean-field control problem with g parameters, mapping from 2D density field to 2D density field. How can ICON-LM learn the operator without any knowledge of its parameters or examples?

However, upon reflection, these unexpected results become quite rational. For these types of problems, once the problem type is identified, the parameters can indeed be deduced solely based on the given question condition. Specifically, the decay rate of the damped oscillator can be computed from a segment of the time series, the boundary conditions of the Poisson equation are encapsulated within the conditions of $u(x)$, and the terminal cost in the mean-field control problem can be derived from the density field during the interval $t \in [0, 0.5]$. The success of ICON-LM in such scenarios indeed showcases its remarkable capability.



(a) (b) (c)

Figure 5: Relative testing error for cases from zero-shot to five-shot learning, for each type of problem. (a) Testing without captions. (b) Testing with vague captions. (c) Testing with precise captions.

6 Summary

This paper presents a novel approach to scientific machine learning with in-context operator learning, transforming it into a multi-modal framework by introducing "captions". These captions incorporate human knowledge about the operator, in the form of natural language descriptions and equations. We also introduce a more efficient neural network architecture for multi-modal in-context operator learning called "ICON-LM". This architecture closely aligns with language models, with the exception of novel input sequences and transformer mask designs.

In the experiments, we compared the ICON-LM model with the original encoder-decoder ICON model, both without captions. The proposed ICON-LM model, comprising approximately half parameters, surpasses the performance of the encoder-decoder ICON model with less training time. This can be attributed that the encoder-decoder ICON model is trained with a fixed number of examples in each step, while the ICON-LM model is trained with varying numbers of examples concurrently in each training step.

Moreover, the utility of multi-modal in-context operator learning was explored using the ICON-LM model. We found that captions' presence, especially precise ones that disclose the parameters in the operators, significantly improved learning performance when the number of examples was limited. It's noteworthy that for certain problem types, zero-shot learning with vague captions yielded comparable results to zero-shot learning with precise captions. In these problems, the operators can, in principle, be deduced solely based on the given question condition, once the problem type is identified. The success of ICON-LM in such scenarios indeed showcases its remarkable learning capability.

In this paper, the usage of "caption" is limited to the model input. In the future, we wish to explore the generation of captions with in-context operator learning. Specifically, we're interested in producing operator descriptions as equations, in natural language, or a blend of both, based on examples from sensory data. Such advancements could pave the way for automated scientific modeling and data analysis within the framework of in-context learning.

Acknowledgement

This work is partially funded by AFOSR MURI FA9550-18-502, ONR N00014-18-1-2527, N00014-20-1-2093 and N00014-20-1-2787. We would like to express our gratitude to ChatGPT for enhancing the wording during the paper writing phase. We also want to thank Dr. Hedi Xia for the helpful discussions.

References

[1] Liu Yang, Siting Liu, Tingwei Meng, and Stanley J Osher. In-context operator learning for differential equation problems. *arXiv preprint arXiv:2304.07993*, 2023.

[2] Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE transactions on neural networks*, 6(4):911–917, 1995.

[3] Tianping Chen and Hong Chen. Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks. *IEEE Transactions on Neural Networks*, 6(4):904–910, 1995.

[4] Yuehaw Khoo, Jianfeng Lu, and Lexing Ying. Solving parametric PDE problems with artificial neural networks. *European Journal of Applied Mathematics*, 32(3):421–435, 2021.

[5] Yinhao Zhu and Nicholas Zabaras. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. *Journal of Computational Physics*, 366:415–447, 2018.

[6] Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. PDE-Net: Learning PDEs from data. In *International conference on machine learning*, pages 3208–3216. PMLR, 2018.

[7] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

[8] 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.

[9] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

[10] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[11] Kaushik Bhattacharya, Bamdad Hosseini, Nikola B Kovachki, and Andrew M Stuart. Model reduction and neural networks for parametric PDEs. *The SMAI journal of computational mathematics*, 7:121–157, 2021.

[12] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *arXiv preprint arXiv:2111.03794*, 2021.

[13] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[15] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[16] Dmitrii Kochkov, Jamie A Smith, Ayya Alieva, Qing Wang, Michael P Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021.

[17] Georgios Kissas, Jacob H Seidman, Leonardo Ferreira Guilhoto, Victor M Preciado, George J Pappas, and Paris Perdikaris. Learning operators with coupled attention. *Journal of Machine Learning Research*, 23(215):1–63, 2022.

[18] Somdatta Goswami, Katiana Kontolati, Michael D Shields, and George Em Karniadakis. Deep transfer operator learning for partial differential equations under conditional shift. *Nature Machine Intelligence*, pages 1–10, 2022.

[19] Min Zhu, Handi Zhang, Anran Jiao, George Em Karniadakis, and Lu Lu. Reliable extrapolation of deep neural operators informed by physics or sparse observations. *Computer Methods in Applied Mechanics and Engineering*, 412:116064, 2023.

[20] Adam Subel, Yifei Guan, Ashesh Chattopadhyay, and Pedram Hassanzadeh. Explaining the physics of transfer learning in data-driven turbulence modeling. *PNAS nexus*, 2(3):pgad015, 2023.

[21] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[22] Linfeng Zhang, Jiequn Han, Han Wang, Roberto Car, and EJPRL Weinan. Deep potential molecular dynamics: a scalable model with the accuracy of quantum mechanics. *Physical review letters*, 120(14):143001, 2018.

[23] David Pfau, James S Spencer, Alexander GDG Matthews, and W Matthew C Foulkes. Ab initio solution of the many-electron schrödinger equation with deep neural networks. *Physical Review Research*, 2(3):033429, 2020.

[24] GP Purja Pun, R Batra, R Ramprasad, and Y Mishin. Physically informed artificial neural networks for atomistic modeling of materials. *Nature communications*, 10(1):2339, 2019.

[25] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016.

[26] Pengzhan Jin, Zhen Zhang, Aiqing Zhu, Yifa Tang, and George Em Karniadakis. SympNets: Intrinsic structure-preserving symplectic networks for identifying Hamiltonian systems. *Neural Networks*, 132:166–179, 2020.

[27] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

[28] Marios Mattheakis, Pavlos Protopapas, David Sondak, Marco Di Giovanni, and Efthimios Kaxiras. Physical symmetries embedded in neural networks. *arXiv preprint arXiv:1904.08991*, 2019.

[29] Weinan E, Jiequn Han, and Arnulf Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in mathematics and statistics*, 5(4):349–380, 2017.

[30] Jiequn Han, Arnulf Jentzen, and Weinan E. Solving high-dimensional partial differential equations using deep learning. *Proc. Natl. Acad. Sci. USA*, 115(34):8505–8510, 2018.

[31] Justin Sirignano and Konstantinos Spiliopoulos. DGM: a deep learning algorithm for solving partial differential equations. *J. Comput. Phys.*, 375:1339–1364, 2018.

[32] Weinan E and Bing Yu. The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. *Commun. Math. Stat.*, 6(1):1–12, 2018.

[33] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.*, 378:686–707, 2019.

[34] Yaohua Zang, Gang Bao, Xiaojing Ye, and Haomin Zhou. Weak adversarial networks for high-dimensional partial differential equations. *J. Comput. Phys.*, 411:109409, 14, 2020.

[35] Lars Ruthotto, Stanley J. Osher, Wuchen Li, Levon Nurbekyan, and Samy Wu Fung. A machine learning framework for solving high-dimensional mean field game and mean field control problems. *Proc. Natl. Acad. Sci. USA*, 117(17):9183–9193, 2020.

[36] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*, 2023.

[37] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. *arXiv:2303.03378*, 2023.

[38] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

[39] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[40] Xiaoman Zhang, Chaoyi Wu, Ziheng Zhao, Weixiong Lin, Ya Zhang, Yanfeng Wang, and Weidi Xie. PMC-VQA: Visual instruction tuning for medical visual question answering. *arXiv preprint arXiv:2305.10415*, 2023.

[41] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv:2304.08485*, 2023.

[42] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. LLaMA-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.

[43] Renjie Pi, Jiahui Gao, Shizhe Diao, Rui Pan, Hanze Dong, Jipeng Zhang, Lewei Yao, Jianhua Han, Hang Xu, and Lingpeng Kong Tong Zhang. DetGPT: Detect what you need via reasoning. *arXiv preprint arXiv:2305.14167*, 2023.

[44] Hang Zhang, Xin Li, and Lidong Bing. Video-LLaMA: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023.

[45] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

[46] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[47] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alex Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-language-action models transfer web knowledge to robotic control. 2023.

[48] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. An empirical study of GPT-3 for few-shot knowledge-based VQA. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3081–3089, 2022.

[49] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. VideoChat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*, 2023.

[50] Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. LLaMA-adapter V2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023.

[51] Anja Reusch, Maik Thiele, and Wolfgang Lehner. Transformer-encoder and decoder models for questions on math. 2022.

Appendix

Neural Network and Training Configurations

The transformer used in this paper has the configuration in Table 2. We utilize the AdamW optimizer with a warmup-cosine-decay schedule, employing the configuration in Table 3.

Table 2: Transformer Configuration

Layers	6
Heads in Multi-Head Attention	8
Input/Output Dimension of Each Layer	256
Dimension of Query/Key/Value in Attention Function	256
Hidden Dimension of Feedforward Networks	1024

Table 3: Configuration of Optimizer and Learning Rate Schedule

Initial Learning Rate	0.0
Peak Learning Rate	1e-4
End Learning Rate	0.0
Warmup Steps	First 10% of Total Steps
Cosine Annealing Steps	Remaining Steps
Global Norm Clip	1.0
Adam β_1	0.9
Adam β_2	0.999
Adam Weight Decay	1e-4

Caption Examples

In this section, we show several examples of training and testing captions for three characteristic problem types: ODE 3 forward problem, PDE 3 forward problem, and MFC g-parameter 1D $\rightarrow$ 1D. For each type, we show four captions for training, where the first two are in the vague group, and the last two are in the precise group. We also show the two captions for testing.

1. Caption examples for ODE 3 forward problem.

```
----Train----
An ODE with a state variable $u$ and control variable $c$.
    condition: $u(0)$ and $c(t), t\in[0,1]$, QoI: $u(t), t\in[0,1]$
function $u$ change over time with the rate of $a_1 \cdot u(t) +
    a_2 \cdot c(t) + a_3$. condition: $u(0)$ and $c(t), t\in[0,1]$,
     QoI: $u(t), t\in[0,1]$
```

```
The rate of change of $u(t)$ with respect to time is described by
    the equation $du(t)/dt = 0.37 \cdot u(t) + 1.46 \cdot c(t) +
    -0.241$. condition: $u(0)$ and $c(t), t\in[0,1]$, QoI: $u(t), t
    \in[0,1]$
The time derivative of the state variable $u(t)$ is equal to $0.37
    \cdot u(t) + 1.46 \cdot c(t) + -0.241$. condition: $u(0)$ and
    $c(t), t\in[0,1]$, QoI: $u(t), t\in[0,1]$

----Test----
The rate of change of $u(t)$ over time is given by the equation $du
    (t)/dt = a_1 \cdot u(t) + a_2 \cdot c(t) + a_3$. condition: $u
    (0)$ and $c(t), t\in[0,1]$, QoI: $u(t), t\in[0,1]$
The relationship between $u(t)$ and $c(t)$ is governed by the
    equation $du(t)/dt = 0.48 \cdot u(t) + 1.06 \cdot c(t) + 0.691$
    . condition: $u(0)$ and $c(t), t\in[0,1]$, QoI: $u(t), t\in
    [0,1]$
```

2. Caption examples for PDE 3 forward problem.

```
----Train----
A nonlinear partial differential equation with variables $u(x)$ and
     $c(x)$, where $c(x)$ is a part of the source term. condition:
    $u(x), x\in[0,1]$, QoI: $c(x), x\in[0,1]$
Nonlinear reaction-diffusion PDE expressed as $-\lambda \frac{d^2u(
    x)}{dx^2} + a \cdot u^3 = c(x)$. condition: $u(x), x\in[0,1]$,
    QoI: $c(x), x\in[0,1]$
$- 0.0712 d^2u(x)/dx^2 + 0.717 \cdot u(x)^3 = c(x)$, with $u(x=0) =
     -0.353$ and $u(x=1) = -0.132$. condition: $u(x), x\in[0,1]$,
    QoI: $c(x), x\in[0,1]$
Equation: $- 0.0712 \frac{d^2u(x)}{dx^2} + 0.717 \cdot u(x)^3 = c(x
    )$. Boundary condition: $u(x=0) = -0.353$, $u(x=1) = -0.132$.
    condition: $u(x), x\in[0,1]$, QoI: $c(x), x\in[0,1]$

----Test----
The nonlinear reaction-diffusion PDE $-\lambda \frac{d^2u(x)}{dx^2}
     + a \cdot u^3(x) = c(x)$ captures the dynamics of the system.
    condition: $c(x), x\in[0,1]$, QoI: $u(x), x\in[0,1]$
In this nonlinear reaction-diffusion PDE, $- 0.0944 \cdot \frac{d^2
    u(x)}{dx^2} + 1 \cdot u^3(x) = c(x)$, subject to the conditions
     $u(0) = -0.967$ and $u(1) = 0.522$. condition: $c(x), x\in
    [0,1]$, QoI: $u(x), x\in[0,1]$
```

3. Caption examples for MFC g-parameter 1D $\to$ 1D.

```
----Train----
A mean field control problem with density denoted as $\rho$,
    subject to a terminal cost integral of form $\int g(x)\rho(1,x)
     dx$ with an unspecified function $g$. condition: $\rho(0,x), x
    \in[0,1]$, QoI: $\rho(1,x), x\in[0,1]$
```

```
Consider a mean field control problem that minimizes $\inf_{\rho, m
    }\iint \frac{10m^2}{\rho} dx dt + \int g(x)\rho(1,x) dx$
    subject to $\partial_t \rho(t,x) + \nabla_x m(t,x) = 0.02 \
    Delta_x \rho(t,x)$ and $\rho(0,x)=\rho_0(x)$, with an undefined
     function $g$. condition: $\rho(0,x), x\in[0,1]$, QoI: $\rho(1,
    x), x\in[0,1]$
Addressing a mean field control problem where $\inf_{\rho, m}\iint
    \frac{10m^2}{\rho} dx dt + \int g(x)\rho(1,x) dx$ is minimized
    under the condition $\partial_t \rho(t,x) + \nabla_x m(t,x) =
    0.02 \Delta_x \rho(t,x)$ for $t \in [0,1], x \in [0,1]$, with
    periodic spatial boundary condition, and the function $g$
    satisfies $g(0), g(0.1), ..., g(0.9)$ = 0.315, 0.473, 0.0224,
    -0.459, -0.0797, 0.259, -0.192, -0.656, -0.0642, 0.314.
    condition: $\rho(0,x), x\in[0,1]$, QoI: $\rho(1,x), x\in[0,1]$
Consider a mean field control problem optimizing $\inf_{\rho, m}\
    iint \frac{10m^2}{\rho} dx dt + \int g(x)\rho(1,x) dx$ where
    the constraints $\partial_t \rho(t,x) + \nabla_x m(t,x) = 0.02
    \Delta_x \rho(t,x)$ are met for $t \in [0,1], x \in [0,1]$,
    with periodic spatial boundary condition, and the function $g$
    satisfies $g(0), g(0.1), ..., g(0.9)$ = 0.315, 0.473, 0.0224,
    -0.459, -0.0797, 0.259, -0.192, -0.656, -0.0642, 0.314.
    condition: $\rho(0,x), x\in[0,1]$, QoI: $\rho(1,x), x\in[0,1]$


----Test----
Examine a mean field control problem that involves a density
    variable $\rho$ and an unknown function $g$ in the terminal
    cost integral. condition: $\rho(0,x), x\in[0,1]$, QoI: $\rho(1,
    x), x\in[0,1]$
The mean field control problem is defined by $\inf_{\rho, m}\int \
    frac{10m^2}{\rho} dx dt + \int g(x)\rho(1,x) dx$, with
    constraint $\partial_t \rho(t,x) + \nabla_x m(t,x) = 0.02 \
    Delta_x \rho(t,x)$ for $t \in [0,1], x \in [0,1]$, with
    periodic spatial boundary condition, where $g$ is specified as
    $g(0), g(0.1), ..., g(0.9)$ = -0.612, -1.48, -1.52, -0.834,
    0.165, 0.705, 1.23, 1.34, 0.785, -0.513.  condition: $\rho(0,x)
    , x\in[0,1]$, QoI: $\rho(1,x), x\in[0,1]$
```