TRANSFORMERS FOR MODELING PHYSICAL SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformers are widely used in neural language processing due to their ability to model longer-term dependencies in text. Although these models achieve state-of-the-art performance for many language related tasks, their applicability outside of the neural language processing field has been minimal. In this work, we propose the use of transformer models for the prediction of dynamical systems representative of physical phenomena. The use of Koopman based embeddings provide a unique and powerful method for projecting any dynamical system into a vector representation which can then be predicted by a transformer model. The proposed model is able to accurately predict various dynamical systems and outperform classical methods that are commonly used in the scientific machine learning literature.[1]

1 INTRODUCTION

The transformer model (Vaswani et al., 2017), built on self-attention, has largely become the state-of-the-art approach for a large set of neural language processing (NLP) tasks including language modeling, text classification, question answering, etc. Although more recent transformer work is focused on unsupervised pre-training of extremely large models (Devlin et al., 2018; Radford et al., 2019; Dai et al., 2019; Liu et al., 2019), the original transformer model garnered attention due to its ability to out-perform other state-of-the-art methods by learning longer-term dependencies without recurrent connections. Given that the transformer model was originally developed for NLP, nearly all related work has been rightfully confined within this field with only a few exceptions. Here, we focus on the development of transformers to model dynamical systems that can replace otherwise expensive numerical solvers. In other words, we are interested in using transformers to learn the language of physics.

The surrogate modeling of physical systems is a research field that has existed for several decades and is a large on going effort in scientific machine learning. A surrogate model is defined as an approximate model of a physical phenomenon that is designed to replace an expensive computational solver that would otherwise be needed to resolve the system of interest. The key characteristic of surrogate models is their ability to model a distribution of initial or boundary conditions rather than learning just one solution. This is arguably essential for the justification of training a deep learning model versus using a standard numerical solver. The most tangible applications of surrogates are for optimization, design and inverse problems where many repeated simulations are typically needed. With the growing interest in deep learning, deep neural networks have been used for surrogate modeling a large range of physical systems in recent literature.

Standard deep neural network architectures such as auto-regressive, recurrent and LSTM based models have been largely demonstrated to be effective at modeling various physical dynamics (Mo et al., 2019; Geneva & Zabaras, 2020a; Tang et al., 2020; Maulik et al., 2020). However, the current literature is focused on systems of limited complexity and domain size. Thus much work is needed to scale such deep learning models to larger physical systems that are of scientific and industrial interest. This work deviates from this pre-existing literature by investigating the use of transformers for the prediction of physical systems, relying entirely on self-attention to model dynamics. To the authors best knowledge, this is the first work to explore transformer NLP architectures for the prediction of physics.

[1]Code available at: [URL available after review].

2 METHODS

When discussing dynamical systems, we are interested in systems that can be described through a dynamical ordinary or partial differential equation:

$$\begin{gathered}\boldsymbol{\phi}_t = F\left(\boldsymbol{x}, \boldsymbol{\phi}(t, \boldsymbol{x}, \boldsymbol{\eta}), \nabla_{\boldsymbol{x}}\boldsymbol{\phi}, \nabla^2_{\boldsymbol{x}}\boldsymbol{\phi}, \boldsymbol{\phi}\cdot\nabla_{\boldsymbol{x}}\boldsymbol{\phi}, \ldots\right), \quad F : \mathbb{R}\times\mathbb{R}^n \to \mathbb{R}^n, \\ t \in \mathcal{T} \subset \mathbb{R}, \quad \boldsymbol{x} \in \Omega \subset \mathbb{R}^m,\end{gathered} \tag{1}$$

in which $\boldsymbol{\phi} \in \mathbb{R}^n$ is the solution of this differential equation with parameters $\boldsymbol{\eta}$, in the time interval $\mathcal{T}$ and spatial domain Ω with a boundary $\Gamma \subset \Omega$. This general form can embody a vast spectrum of physical phenomena including fluid flow and transport processes, mechanics and materials physics, and molecular dynamics. In this work, we are interested in learning the set of solutions for a distribution of initial conditions $\boldsymbol{\phi}_0 \sim p(\boldsymbol{\phi}_0)$, boundary conditions $\mathcal{B}(\boldsymbol{\phi}) \sim p(\mathcal{B})\, \forall \boldsymbol{x} \in \Gamma$ or equation parameters $\boldsymbol{\eta} \sim p(\boldsymbol{\eta})$. This accounts for modeling initial value, boundary value and stochastic problems. We emphasize that this is fundamentally different, more difficult and of greater interest for most scientific applications compared to learning a single solution.

To make this problem applicable to the use of transformer models, the continuous solution is discretized in both the spatial and temporal domains such that the solution of the differential equation is $\Phi = \{\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \ldots \boldsymbol{\phi}_T\}\,; \boldsymbol{\phi}_i \in \mathbb{R}^{n\times d}$ for which $\boldsymbol{\phi}_i$ has been discretized by d points in Ω. We assume an initial state $\boldsymbol{\phi}_0$ and that the time interval $\mathcal{T}$ is discretized by T time-steps with a time-step size Δt. Hence, we pose the problem of modeling a dynamical system as a time-series problem. The machine learning methodology has two core components: the transformer for modeling dynamics and the embedding network for projecting physical states into a vector representation.

2.1 TRANSFORMER

The transformer model was originally designed with NLP as the sole application with word vector embeddings of a passage of text being the primary input (Vaswani et al., 2017). However, recent works have explored using attention mechanisms for different machine learning tasks (Veličković et al., 2017; Zhang et al., 2019; Fu et al., 2019) and a few investigate the use of transformers for applications outside of the NLP field (Chen et al., 2020). This suggests that self-attention and in particular transformer models may work well for any problem that can be posed as a sequence of vectors. In this work, the primary input to the transformer will be an embedded dynamical system, $\Xi = \{\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots \boldsymbol{\xi}_T\}$, where the embedded state at time-step i is denoted as $\boldsymbol{\xi}_i \in \mathbb{R}^e$. Given that we are interested in the prediction of a physical time series, this motivates the usage of a language modeling architecture that is designed for the sequential prediction of words in a body of text. We select the transformer decoder architecture used in the GPT models (Radford et al., 2018; 2019).

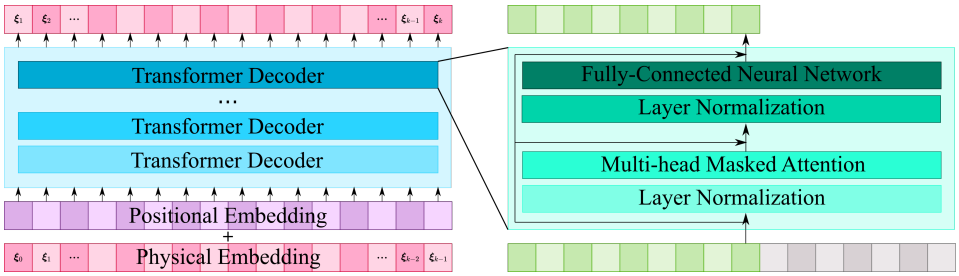


Figure 1: The transformer decoder model used for the prediction of physical dynamics.

Our model follows the GPT-2 architecture based on the implementation in the Hugging Face transformer repository (Wolf et al., 2019), but is significantly smaller in size than these modern NLP transformers. This model consists of a stack of transformer decoder layers that use masked attention, as depicted in Fig. 1. The inputs to the transformer are the embedded representation of the physical system with the sinusoidal positional encoding proposed in the original transformer (Vaswani et al., 2017). To train the model, consider a data set of D embedded i.i.d. time-series $\mathcal{D} = \left\{\Xi^i\right\}_{i=1}^{D}$ for which we can use the standard time-series Markov model (language modeling) log-likelihood:

$$L_{\mathcal{D}} = \sum_i^D \sum_j^T \log p\left(\boldsymbol{\xi}^i_j | \boldsymbol{\xi}^i_{j-k}, \ldots, \boldsymbol{\xi}^i_{j-1}, \boldsymbol{\theta}\right). \tag{2}$$

k is the context window and $\boldsymbol{\theta}$ are the model's parameters. Contrary to the standard NLP approach which poses the likelihood as a softmax over a dictionary of tokens, the likelihood here is taken as a standard Gaussian between the transformer's prediction and the target embedded value resulting in a L_2 loss. This is due to the fact that the solution to most physical systems cannot be condensed to a discrete finite set making tokenization into a finite dictionary not possible rendering a softmax approach not applicable. Training is the standard auto-regressive method used in GPT (Radford et al., 2018), as opposed to the word masking (Devlin et al., 2018), constrained to the embedded space. The physical states, $\boldsymbol{\phi}_i$, have the potential to be very high dimensional thus training the transformer in the lower-dimensional embedded space can significantly lower training costs.

2.2 Embedding Model

The second major component of the machine learning methodology is the embedding model responsible for projecting the discretized physical state space into a 1D vector representation. In NLP, the standard approach is to tokenize then embed a finite vocabulary of words, syllables or characters using methods such as n-gram models, Byte Pair Encoding (Gage, 1994), Word2Vec (Mikolov et al., 2013a;b), GloVe (Pennington et al., 2014), etc. These methods allow language to be represented by a series of 1D vectors that serve as the input to the transformer. Clearly a finite tokenization and such NLP embeddings are not directly applicable to a physical system, thus we propose our own embedding method designed specifically for dynamical systems. Consider learning the generalized mapping between the systems state space and embedded space: $\mathcal{F} : \mathbb{R}^{n\times d} \rightarrow \mathbb{R}^e$ and $\mathcal{G} : \mathbb{R}^e \rightarrow \mathbb{R}^{n\times d}$. Naturally, multiple approaches can be used especially if the dimensionality of the embedded space is less than that of the state-space but this is not always the case.

The primary approach that we will propose is a Koopman observable embedding which is a technique that can be applied universally to all dynamical systems. Considering the discrete time form of the dynamical system in Eq. 1, the evolution of the state variables can be abstracted by $\boldsymbol{\phi}_{i+1} = \mathbb{F}\left(\boldsymbol{\phi}_i\right)$ for which $\mathbb{F}$ is the dynamic map from one time-step to the next. The foundation of Koopman theory is that for such a dynamical system, there exists an infinite set of state observables, $g\left(\boldsymbol{\phi}_i\right)$, that evolve linearly in time such that:

$$\mathcal{K}g\left(\boldsymbol{\phi}_i\right) \triangleq g \circ \mathbb{F}\left(\boldsymbol{\phi}_i\right), \tag{3}$$

where $\mathcal{K}$ is known as the Koopman operator (Koopman, 1931) which is time-invariant. Modeling the dynamics of a system through the linear Koopman space can be attractive due to its simplification of the dynamics but also the potential physical insights it brings along with it. Spectral analysis of the Koopman operator can reveal fundamental dynamical modes that drive the system's evolution in time. However, these observables, $g\left(\boldsymbol{\phi}_i\right)$, are infinite dimensional and typically unknown. In recent years, various machine learning based approaches have been proposed to learn both the Koopman operator and observables approximated in a finite space for modeling, control and dynamical mode analysis (Takeishi et al., 2017; Li et al., 2017; Lusch et al., 2018; Otto & Rowley, 2019). While deep learning methods have enabled greater success with discovering Koopman observables and operators, applications have still been limited to fairly simple systems and do not work for long time prediction. This is likely due to the approximation of the finite-dimensional Koopman observables, limited data and complete dependence on the discovered Koopman operator $\mathcal{K}$. Suggesting the prediction of a system through a single linear transform clearly has significant limitations and is fundamentally a naive approach from a machine learning perspective.

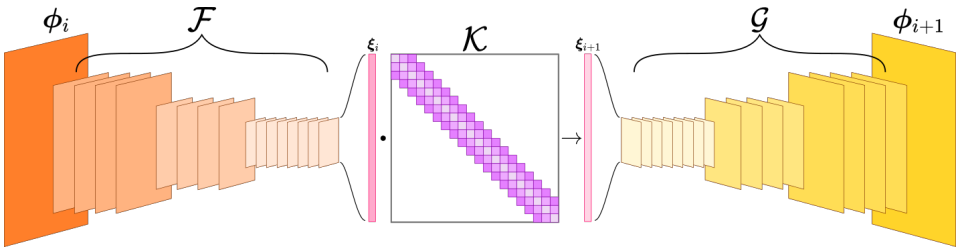


Figure 2: Example of a Koopman embedding for a two-dimensional system using a convolution encoder-decoder model.

In this work, we propose using approximate Koopman observations as a methodology to develop embeddings for the transformer model such that $\mathcal{F}(\boldsymbol{\phi}_i) \triangleq g(\boldsymbol{\phi}_i)$. As seen in Fig. 2, the embedding model follows a standard encoder-decoder model with the middle latent variables being the Koopman observables. Embedding model architectures for each numerical example are provided in Appendix A. We introduce a learnable Koopman operator which takes the form of a banded matrix that is symmetrical about the diagonal to encourage the discovery of dominate dynamical modes rather than high-frequency oscillations (Lusch et al., 2018; Otto & Rowley, 2019). This learned Koopman operator is disposed of once training of the embedding model is complete. Given the data set of physical state time-series, $\mathcal{D}_\Phi = \left\{\Phi^i\right\}_i^D$, the Koopman embedding model is trained with the following loss:

$$\mathcal{L}_{\mathcal{D}_\Phi} = \sum_{i=1}^{D}\sum_{j=0}^{T} \lambda_0 \underbrace{MSE\left(\boldsymbol{\phi}_j^i, \mathcal{G}\circ\mathcal{F}\left(\boldsymbol{\phi}_j^i\right)\right)}_{Reconstruction} + \lambda_1 \underbrace{MSE\left(\boldsymbol{\phi}_j^i, \mathcal{G}\circ\mathcal{K}^j\mathcal{F}\left(\boldsymbol{\phi}_0^i\right)\right)}_{Dynamics} + \lambda_2 \underbrace{\|\mathcal{K}\|_2^2}_{Decay}. \tag{4}$$

This loss function consists of three components: the first is a reconstruction loss which ensures a consistent mapping to and from the embedded representation. The second is the Koopman dynamics loss which pushes $\boldsymbol{\xi}_j$ to follow linear dynamics resulting in time-steps of similar dynamics to have similar embeddings. The last term decays the Koopman operator's parameters to help force the model to discover lower-dimensional dynamical modes. In reference to traditional NLP embeddings, we believe our Koopman observable embedding has a motivation similar to Word2Vec (Mikolov et al., 2013a) as well as more recent embedding methods such as context2vec (Melamud et al., 2016), ELMo (Peters et al., 2018), etc. Namely, these methods are based on the principle of context: words that are contextually related to each other have a similar embedding. The Koopman embedding has a similar objective encouraging physical realizations containing similar *time-invariant dynamical modes* to also have similar embeddings.

3 Chaotic Dynamics

As a foundational numerical example to rigorously compare the proposed model to other classical machine learning techniques, we will first look at surrogate modeling of the Lorenz system governed by:

$$\frac{dx}{dt} = \sigma\left(y - x\right), \quad \frac{dy}{dt} = x\left(\rho - z\right) - y, \quad \frac{dz}{dt} = xy - \beta z. \tag{5}$$

We use the classical parameters of $\rho = 28, \sigma = 10, \beta = 8/3$. For this numerical example, we wish to develop a surrogate model for predicting the Lorenz system given a random initial state $x_0 \sim \mathcal{U}(-20, 20)$, $y_0 \sim \mathcal{U}(-20, 20)$ and $z_0 \sim \mathcal{U}(10, 40)$. In other words, we wish to surrogate model various initial value problems for this system of ODEs. The Lorenz system is used here because of its well known chaotic dynamics which make it extremely sensitive to numerical perturbations and thus an excellent benchmark for assessing a machine learning model's accuracy.

A total of four alternative machine learning models are implemented: a fully-connected auto-regressive model, a fully-connected LSTM model, a deep neural network Koopman model (Lusch et al., 2018; Otto & Rowley, 2019) and lastly an echo-state model (Chattopadhyay et al., 2019; Lukoševičius, 2012). All of these models have been proposed in past literature for predicting many dynamical systems. Each are provided the same training, validation and testing data sets containing 2048, 64 and 256 time-series solved using a numerical solver, respectively. The training data set contains time-series of 256 time-steps while the validation and testing data sets have 1024 time-steps. Each model is allowed to train for 500 epochs if applicable. The proposed transformer and embedding model are trained for 200 and 300 epochs, respectively, with an embedding dimension of 32. The embedding model is a simple fully-connected encoder-decoder model, $\mathcal{F}: \mathbb{R}^3 \rightarrow \mathbb{R}^{32}; \mathcal{G}: \mathbb{R}^{32} \rightarrow \mathbb{R}^3$, which is trained first. The transformer is then trained with a context length of 64 with 4 transformer decoder layers.

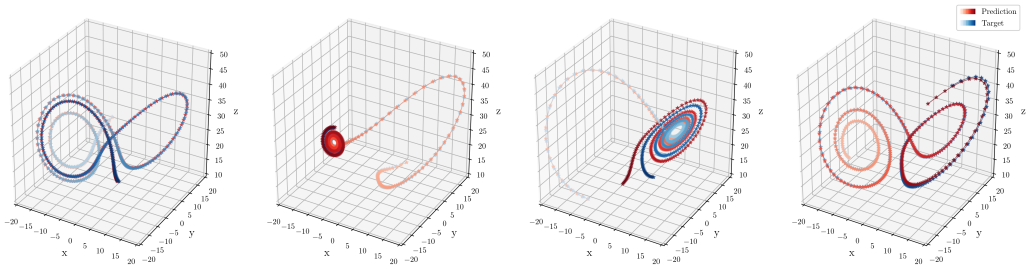
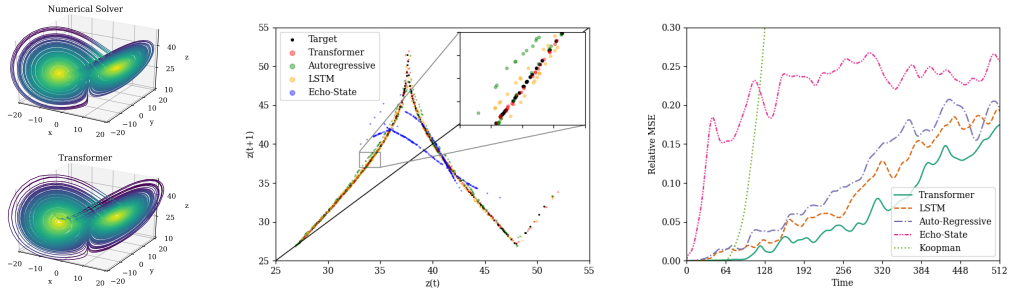


Figure 3: Four test case predictions using the transformer model for 320 time-steps.

We plot four separate test cases in Fig. 3 for which only the initial state is provided and the transformer model predicts 320 time-steps. Several test predictions for alternative models are provided in Appendix B. In general, we can see the transformer model is able to yield extremely accurate predictions even beyond its context length. Additionally, we plot the Lorenz solution for 25k time-steps from a numerical solver and predicted from transformer model in Fig. 4a. Note both have the same structure, which qualitatively indicates that the transformer indeed maintains physical dynamics.



(a) (b) (c)

Figure 4: (a) Lorenz solution of 25k time-steps. (b) The Lorenz map produced by each model. (c) The test relative mean-squared-error (MSE) with respect to time.

Table 1: Test set relative mean-squared-error (MSE) for surrogate modeling the Lorenz system at several time-step intervals.

		Relative MSE		
Model	Parameters	[0-64)	[64-128)	[128-192)
Transformer	36k/54k†	0.0003	**0.0060**	**0.0221**
LSTM	103k	0.0041	0.0175	0.0369
Autoregressive	92k	0.0057	0.0253	0.0485
Echo State	7.5k/6.3m‡	0.1026	0.1917	0.2209
Koopman	108k	**0.0001**	0.0962	2.0315

† Learnable parameters for the embedding/ transformer model.
‡ Learnable output parameters/ fixed input and reservoir parameters.

The proposed transformer and alternative models' relative mean squared errors for the test set are plotted in Fig. 4c as a function of time and listed in Table 1. We can see that the transformer model is able to achieve the best performance using the least number of parameters with the exception of the Koopman only model for early times. However, the Koopman model is unstable for long time predictions rendering it useless for surrogate predictions. To quantify accuracy of the chaotic dynamics of each model, the Lorenz map is plotted in Fig. 4b which is a well-defined relation between successive z local maximas despite the Lorenz's chaotic nature. Calculated using 25k time-step predictions from each model, again we can see that the transformer model agrees the best

with the numerical solver indicating that it has learned the best physical dynamics of all the tested models.

4 2D FLUID DYNAMICS

The next dynamical system we will test is transient 2D fluid flow governed by the Navier–Stokes equations:

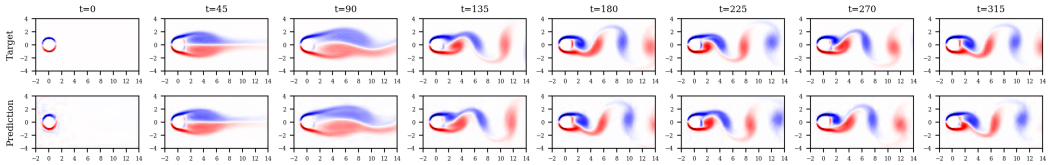
$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial u_i}{\partial x_j^2}, \tag{6}$$

in which u_i and p are the velocity and pressure, respectively. ν is the viscosity of the fluid. We consider modeling the classical problem of flow around a cylinder at various Reynolds number defined by $Re = u_{in} d/\nu$ in which $u_{in} = 1$ and $d = 2$ are the inlet velocity and cylinder diameter, respectively. In this work, we choose to develop a surrogate model to predict the solution at any Reynolds number between $Re \sim \mathcal{U}\,(100, 750)$. This problem is a fairly classical flow to investigate in scientific machine learning with various levels of difficulty (Lee & You, 2017; Morton et al., 2018; Lusch et al., 2018; Han et al., 2019; Xu & Duraisamy, 2020; Geneva & Zabaras, 2020b). Here we choose one of the more difficult forms: model the flow starting at a steady state flow field at $t = 0$. Meaning the model is provided zero information on the structure of the cylinder wake during testing.

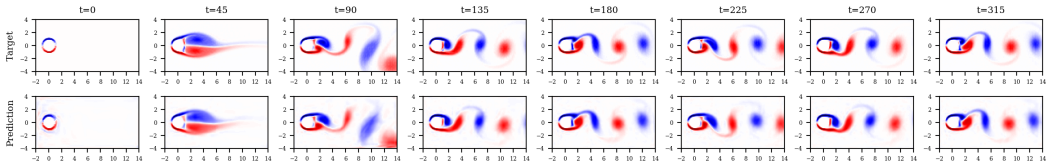
Training, validation and testing data is obtained from using the OpenFOAM simulator (Jasak et al., 2007), from which a rectangular structured sub-domain is sampled centered around the cylinder's wake. Given that the flow is two-dimensional, our model will predict the x-velocity, y-velocity and pressure fields, $\{\boldsymbol{u}_x, \boldsymbol{u}_y, \boldsymbol{p}\} \in \mathbb{R}^{3\times128\times64}$. The training, validation and test data sets consist of 27, 6 and 7 fluid flow simulations respectively with 400 time-steps each at a physical time-step size of $\Delta t = 0.5$. As a base line model, we train a convolutional encoder-decoder model with a stack of three convolutional LSTMs (Xingjian et al., 2015) in the center. We note that convolutional LSTMs have been used extensively in recent scientific machine learning literature for modeling various physical systems (Han et al., 2019; Wiewel et al., 2019; Tang et al., 2020; Geneva & Zabaras, 2020b), thus can be considered a state-of-the-art approach. The convolutional LSTM model is trained for 500 epochs.

Additionally, three different embedding methods are implemented: the first is the proposed Koopman observable embedding using a convolutional auto–encoder. This model encodes the fluid x-velocity, y-velocity, pressure and viscosity fields to an embedded dimension of 128, $\mathcal{F} : \mathbb{R}^{4\times128\times64} \rightarrow \mathbb{R}^{128}; \mathcal{G} : \mathbb{R}^{128} \rightarrow \mathbb{R}^{3\times128\times64}$. The second embedding method is using the same convolutional auto–encoder model but without the enforcement of Koopman dynamics on the embedded variables. The last embedding method tested was principal component analysis (PCA), a classical dimensionality reduction benchmark method for all fields of science. For each embedding method an identical transformer model with a context length of 128 with 4 transformer decoder layers is trained. Similar to the previous example, the embedding models are trained for 300 epochs when applicable and the transformer is trained for 200.

Each trained model is tested on the test set by providing the initial laminar state at $t = 0$ with the fluid viscosity and allowing the model to predict 400 time-steps into the future. Two test predictions using the proposed transformer model with Koopman embeddings are plotted in Fig. 5 in which the predicted vorticity fields are in good agreement with the true solution. The test set relative mean square error for each output field for each model is plotted in Fig. 6. The errors of each field over the entire time-series are listed in Table 2. Additional results are provided in Appendix C. For all alternative models, a rapid error increase can be seen between $t = [50, 100]$ which is due to the transition from the laminar flow into vortex shedding. It is clear that only the proposed transformer model with Koopman embeddings can correctly predict these dynamics demonstrating the benefits of enforcing Koopman dynamics on the embedded representations. Compared to the widely used ConvLSTM model, the proposed transformer offers far more reliable predictions with less learnable parameters.



(a) $Re = 233$



(b) $Re = 633$

Figure 5: Vorticity, $\boldsymbol{\omega} = \nabla_x \boldsymbol{u}_y - \nabla_y \boldsymbol{u}_x$, of two test case predictions using the proposed transformer with Koopman embeddings at Reynolds numbers 233 and 633.

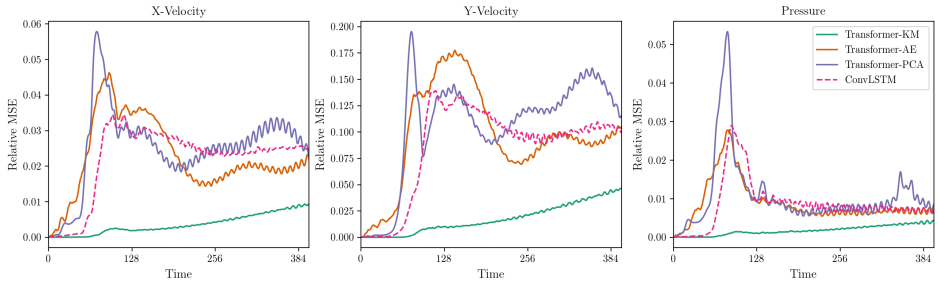


Figure 6: Test set relative mean-squared-error (MSE) of the base-line convolutional LSTM model (ConvLSTM) and transformers with Koopman (KM), auto-encoder (AE) and PCA embedding methods over time.

Table 2: Test set relative mean-squared-error (MSE) of each output field for surrogate modeling 2D fluid flow past a cylinder. Models listed include the convolutional LSTM model (ConvLSTM) and transformers with Koopman (KM), auto-encoder (AE) and PCA embedding methods.

		Relative MSE $[0-400]$		
Model	Parameters	$\boldsymbol{u}_x$	$\boldsymbol{u}_y$	$\boldsymbol{p}$
Transformer-KM	224k/628k†	**0.0036**	**0.0175**	**0.0018**
Transformer-AE	224k/628k†	0.0226	0.09606	0.0088
Transformer-PCA	3.1m/628k†	0.0251	0.1062	0.0108
ConvLSTM	934k	0.0220	0.0862	0.0086

† Learnable parameters for the embedding/ transformer model.

5 3D Reaction–Diffusion Dynamics

The final numerical example to demonstrate the proposed transformer model is a 3D reaction–diffusion system governed by the Gray–Scott model:

$$\frac{\partial u}{\partial t} = r_u \frac{\partial u}{\partial x_i^2} - uv^2 + f\left(1-u\right), \quad \frac{\partial v}{\partial t} = r_v \frac{\partial v}{\partial x_i^2} + vu^2 - \left(f+k\right)v, \tag{7}$$

in which u and v are the concentration of two species, r_u and r_v are their respective diffusion rates, k is the kill rate and f is the feed rate. This is a classical system of particular application to chemical processes as it models the following chemical reaction: $U + 2V \rightarrow 3V$; $V \rightarrow P$. For a set region of feed and kill rates, this seemingly simple system can yield a wide range of complex dynamics (Pearson, 1993; Lee et al., 1993). Hence, under the right settings, this system is

an excellent case study to push the proposed methodology to its predictive limits. In this work, we will use the parameters: $r_u = 0.2$, $r_v = 0.1$, $k = 0.055$ and $f = 0.025$ which results in a complex dynamical reaction seen in Fig. 7. Akin to the first numerical example, the initial condition of this system is stochastic such that the system is seeded with 3 randomly placed perturbations within the periodic domain.

Training, validation and testing data are obtained from a Runge-Kutta finite difference simulation on a structured grid, $\{\boldsymbol{u}, \boldsymbol{v}\} \in \mathbb{R}^{2\times32\times32\times32}$. The training, validation and test data sets consist of 512, 16 and 56 time-series, respectively, with 200 time-steps each at a physical time-step size of $\Delta t = 5$. A 3D convolutional encoder-decoder is used to embed the two species into a 512 embedding dimension, $\mathcal{F} : \mathbb{R}^{2\times32\times32\times32} \rightarrow \mathbb{R}^{512}; \mathcal{G} : \mathbb{R}^{512} \rightarrow \mathbb{R}^{2\times32\times32\times32}$. Transformer models with varying depth are trained all with a context length of 128. All other model and training parameters for the transformer models are consistent. A test prediction using the transformer model is shown in Fig. 7 and the errors for each trained transformer are listed in Table 3. Despite this system having complex dynamics in 3D space, the transformer is able to produce acceptable predictions with very similar structures as the numerical solver. To increase the model's predictive accuracy we believe the limitation here is not in the transformer, but rather the number of training data and the inaccuracies of the embedding model. This is supported by the fact that increasing the transformer's depth does not yield considerable improvements for the test errors in Table 3. Additional, results are provided in Appendix D.

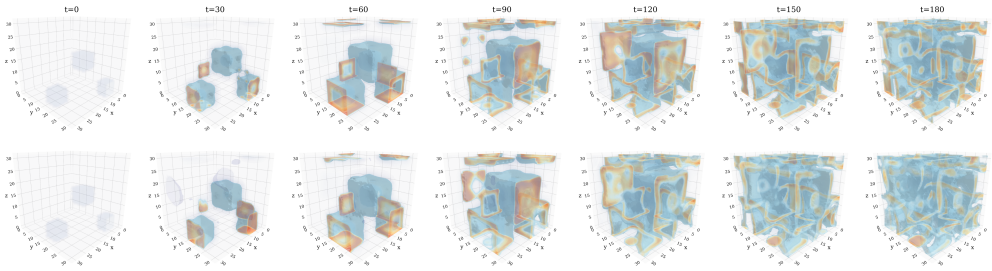


Figure 7: $\boldsymbol{u}$ volume plots of the target (top) and transformer prediction (bottom) for a test case of the Gray–Scott system. Isosurfaces displayed span the range $\boldsymbol{u} = [0.3, 0.5]$ to show the inner structure.

Table 3: Test set relative mean-squared-error (MSE) for surrogate modeling 3D Gray-Scott system.

			Relative MSE $[0-200]$	
Model	Layers	Parameters	$\boldsymbol{u}$	$\boldsymbol{v}$
Transformer	2	6.2m/6.6m†	0.0159	0.0120
Transformer	4	6.2m/12.9m†	0.0154	0.0130
Transformer	8	6.2m/25.5m†	**0.0125**	**0.0101**

† Learnable parameters for the embedding/ transformer model.

6 Conclusion

While transformers and self-attention models have been established as a powerful framework for NLP tasks, the adoption of such methodologies has yet to fully permute other fields. In this work, we have demonstrated the potential transformers have for modeling dynamics of physical phenomena. The challenge is identifying appropriate embeddings to represent the physical state of the system, for which we propose the use of Koopman dynamics to enforce dynamical context. Using the proposed methods, our transformer model can outperform alternative models widely seen in recent scientific machine learning literature. The investigation of unsupervised pre-training of such transformer models as well as gaining a better understanding of what attention mechanisms imply for physical systems will be the subjects of works in the near future.

Acknowledgments

The authors acknowledge computing resources provided by the AFOSR Office of Scientific Research through the DURIP program.

REFERENCES

Ashesh Chattopadhyay, Pedram Hassanzadeh, Devika Subramanian, and Krishna Palem. Data-driven prediction of a multi-scale Lorenz 96 chaotic system using a hierarchy of deep learning methods: Reservoir computing, ANN, and RNN-LSTM. 2019.

Mark Chen, Alec Radford, Rewon Child, Jeff Wu, Heewoo Jun, Prafulla Dhariwal, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3146–3154, 2019.

Philip Gage. A new algorithm for data compression. *C Users Journal*, 12(2):23–38, 1994.

Nicholas Geneva and Nicholas Zabaras. Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks. *Journal of Computational Physics*, 403:109056, 2020a.

Nicholas Geneva and Nicholas Zabaras. Multi-fidelity generative deep learning turbulent flows. *arXiv preprint arXiv:2006.04731*, 2020b.

Renkun Han, Yixing Wang, Yang Zhang, and Gang Chen. A novel spatial-temporal prediction method for unsteady wake flows based on hybrid deep neural network. *Physics of Fluids*, 31(12): 127101, 2019.

Hrvoje Jasak, Aleksandar Jemcov, Zeljko Tukovic, et al. Openfoam: A c++ library for complex physics simulations. In *International workshop on coupled methods in numerical dynamics*, volume 1000, pp. 1–20. IUC Dubrovnik Croatia, 2007.

Bernard O Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the national academy of sciences of the united states of america*, 17(5):315, 1931.

Kyoung J Lee, WD McCormick, Qi Ouyang, and Harry L Swinney. Pattern formation by interacting chemical fronts. *Science*, 261(5118):192–194, 1993.

Sangseung Lee and Donghyun You. Prediction of laminar vortex shedding over a cylinder using deep learning. *arXiv preprint arXiv:1712.07854*, 2017.

Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10):103111, 2017.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Mantas Lukoševičius. A practical guide to applying echo state networks. In *Neural networks: Tricks of the trade*, pp. 659–686. Springer, 2012.

Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):1–10, 2018.

Romit Maulik, Romain Egele, Bethany Lusch, and Prasanna Balaprakash. Recurrent neural network architecture search for geophysical emulation. *arXiv preprint arXiv:2004.10928*, 2020.

Oren Melamud, Jacob Goldberger, and Ido Dagan. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pp. 51–61, 2016.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013b.

Shaoxing Mo, Yinhao Zhu, Nicholas Zabaras, Xiaoqing Shi, and Jichun Wu. Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media. *Water Resources Research*, 55(1):703–728, 2019.

Jeremy Morton, Antony Jameson, Mykel J Kochenderfer, and Freddie Witherden. Deep dynamical modeling and control of unsteady fluid flows. In *Advances in Neural Information Processing Systems*, pp. 9258–9268, 2018.

Samuel E Otto and Clarence W Rowley. Linearly recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1):558–593, 2019.

John E Pearson. Complex patterns in a simple system. *Science*, 261(5118):189–192, 1993.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI Blog*, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning koopman invariant subspaces for dynamic mode decomposition. In *Advances in Neural Information Processing Systems*, pp. 1130–1140, 2017.

Meng Tang, Yimin Liu, and Louis J Durlofsky. A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems. *Journal of Computational Physics*, pp. 109456, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Steffen Wiewel, Moritz Becher, and Nils Thuerey. Latent space physics: Towards learning the temporal evolution of fluid flow. In *Computer Graphics Forum*, volume 38, pp. 71–82. Wiley Online Library, 2019.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pp. 802–810, 2015.

Jiayang Xu and Karthik Duraisamy. Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics. *Computer Methods in Applied Mechanics and Engineering*, 372:113379, 2020.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pp. 7354–7363. PMLR, 2019.

A KOOPMAN EMBEDDING MODELS

The architectures for the Koopman embedding models are provided. In general, each is a classical auto-encoder architecture using standard machine learning methodologies. For complete details we refer the reader to the code.

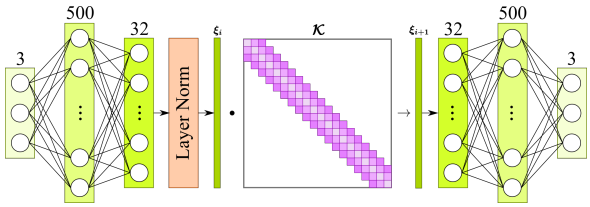


Figure 8: Fully-connected embedding network with ReLU activation functions for the Lorenz system.

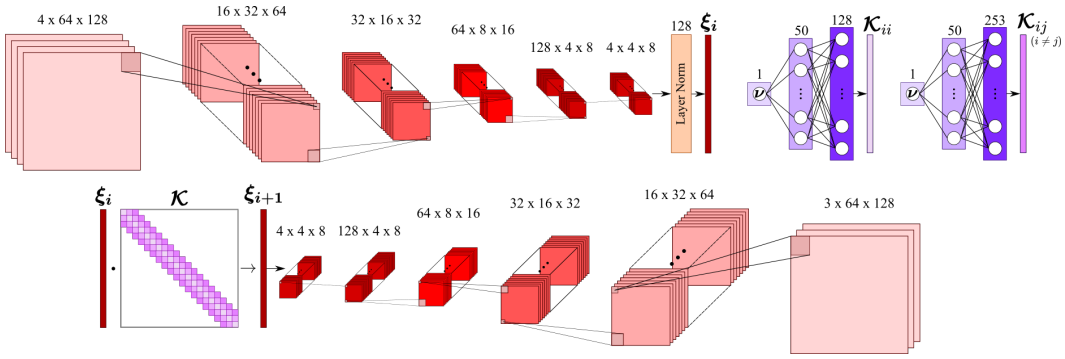


Figure 9: 2D convolutional embedding network with ReLU activation functions for the flow around a cylinder system consisting of 5 convolutional encoding/decoding layers. Each convolutional operator has a kernel size of $(3, 3)$. In the decoder, the feature maps are up-sampled before applying a standard convolution. Additionally, two auxiliary fully-connected networks are used to predict the diagonal and off-diagonal elements of the Koopman operator for each viscosity $\boldsymbol{\nu}$.

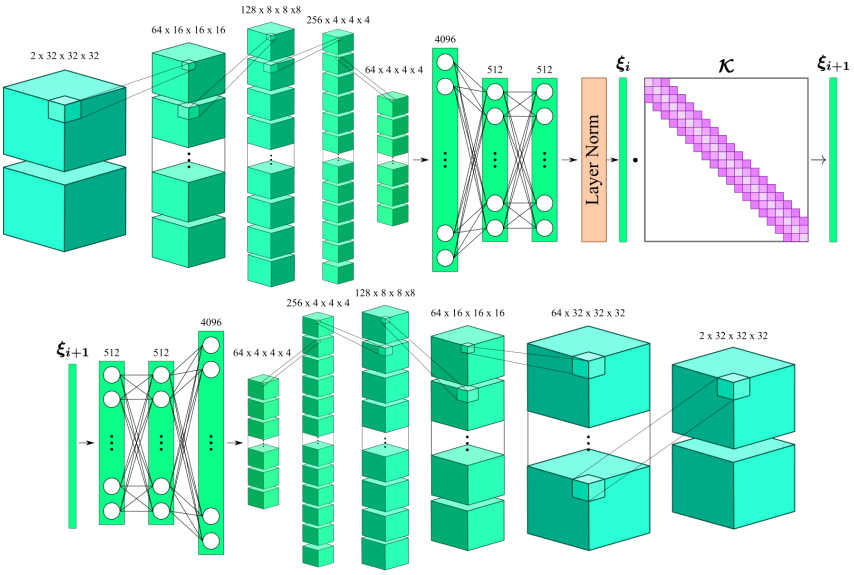


Figure 10: 3D convolutional embedding network with leaky ReLU activation functions for the Gray-Scott system. Batch-normalization used between each of the convolutional layers. In the decoder, the feature maps are up-sampled before applying a standard 3D convolution.

B LORENZ SUPPLEMENTARY RESULTS

Predictions for all of the tested models for three test cases are plotted in Fig. 11 in which we can see the transformer model is consistently accurate within the plotted time frame. Additional test predictions for the Lorenz system using the transformer model are shown in Fig. 12.

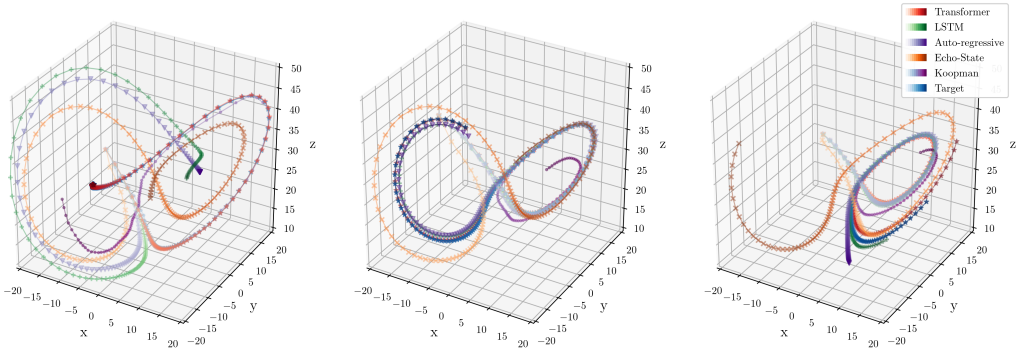


Figure 11: Three Lorenz test case predictions using each tested model for 128 time-steps.

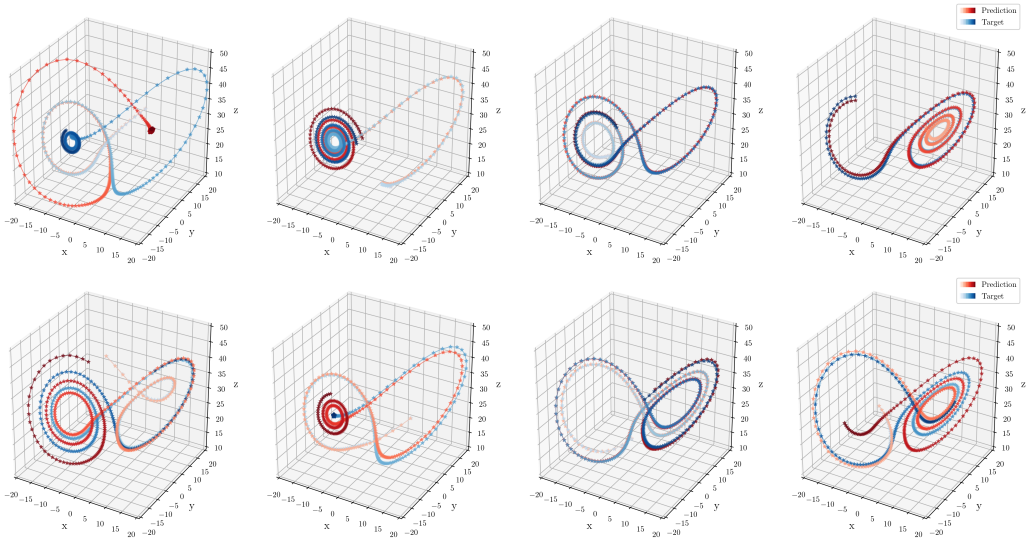


Figure 12: Lorenz test case predictions using the transformer model for 320 time-steps.

C CYLINDER SUPPLEMENTARY RESULTS

Prediction fields of all the tested models for a single test case are plotted in Figs. 13 & 14. While all models are able to perform adequately with qualitatively good results, the transformer model with Koopman embeddings (Transformer-KM) outperforms alternatives.

The evolution of the flow field projected onto the dominate eigenvectors of the learned Koopman operator for two Reynolds numbers are plotted in Fig. 15. This reflects the dynamical modes that were learned by the embedding model to impose physical "context" on the embeddings. Given that the eigenvectors are complex, we plot both the magnitude, $|\psi|$, and angle, $\angle\psi$. For both Reynolds numbers it is easy to see the initial transition region, $t < 100$, before the system enters the periodic vortex shedding. Once in the periodic region, we can see that the higher Reynolds number has a higher frequency which reflects the increased vortex shedding speed.

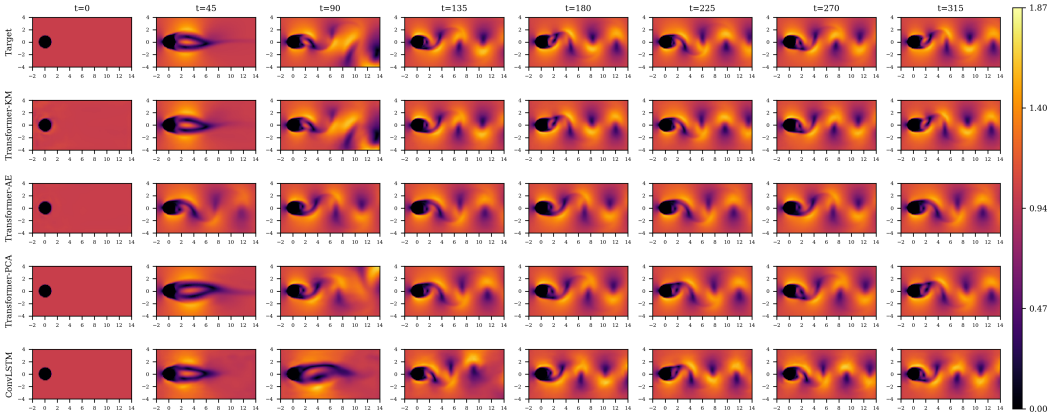


Figure 13: Velocity magnitude predictions of a test case at $Re = 633$ using each tested model.

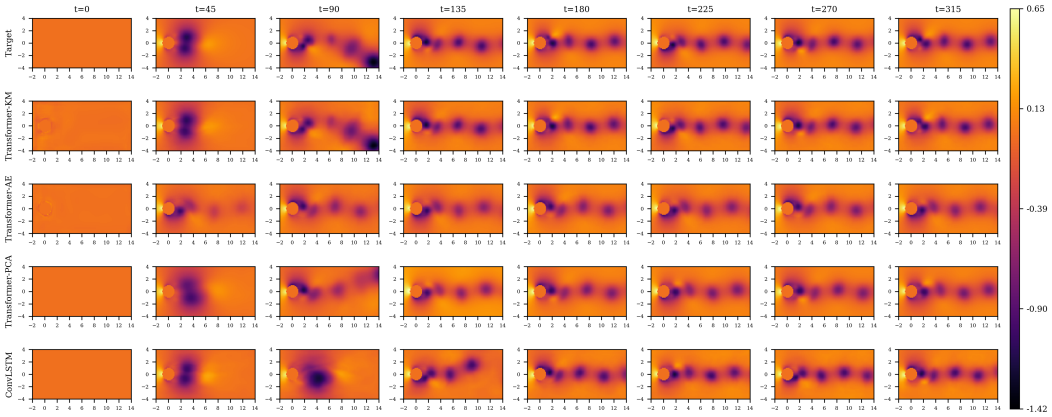
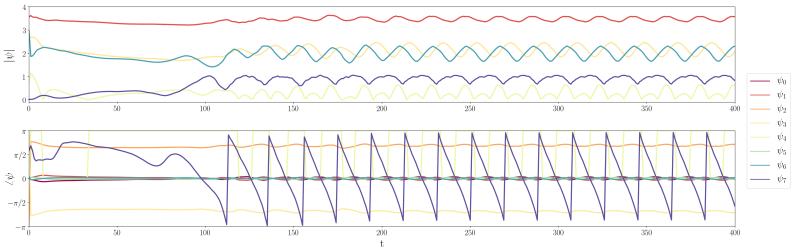
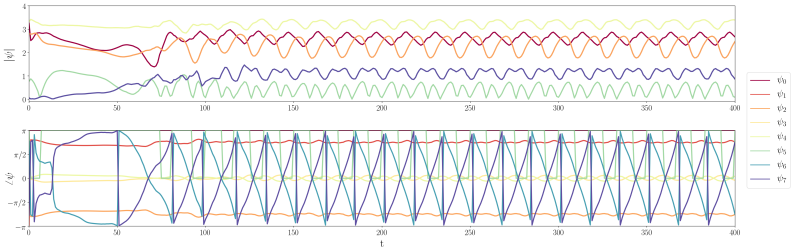


Figure 14: Pressure predictions of a test case at $Re = 633$ using each tested model.



(a) $Re = 233$

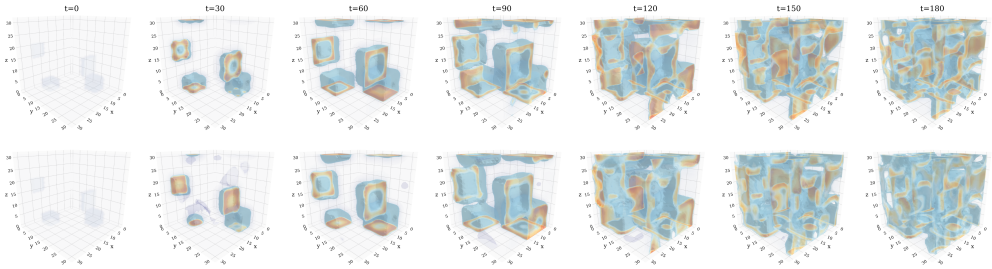


(b) $Re = 633$

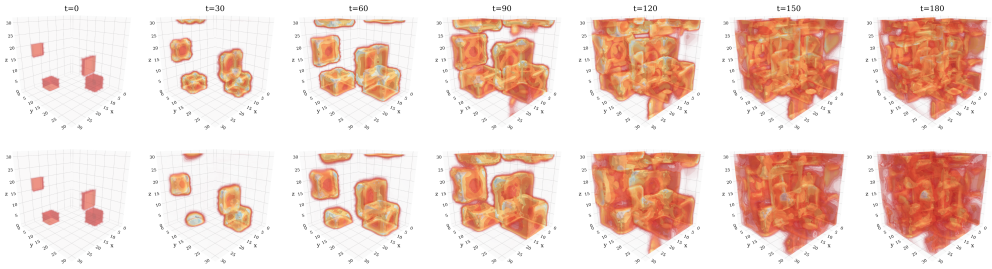
Figure 15: The dynamics of the fluid flow around a cylinder projected onto the 8 most dominate eigenvectors of the learned Koopman operator, $\mathcal{K}$, in the embedding model.

D GRAY-SCOTT SUPPLEMENTARY RESULTS

Volume plots of two test cases for both species are provided in Figs. 16 & 17. Additionally, to better visualize the accuracy of the trained transformer model, contour plots for three test cases are also provided in Fig. 18. Both the volume and contour plots illustrate the complex and unique structures that arise in this system for each initial condition.

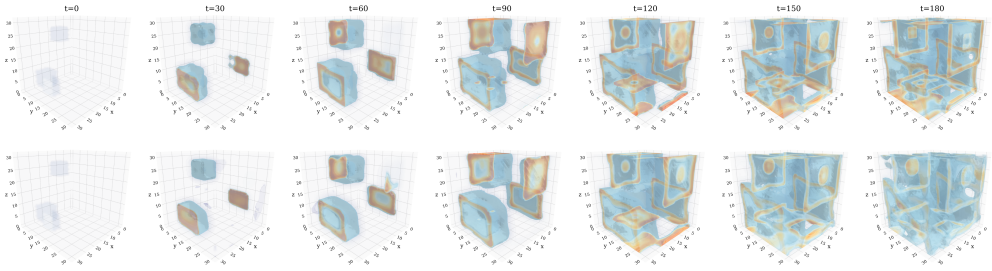


(a) $\boldsymbol{u}$ target (top) and transformer prediction (bottom)

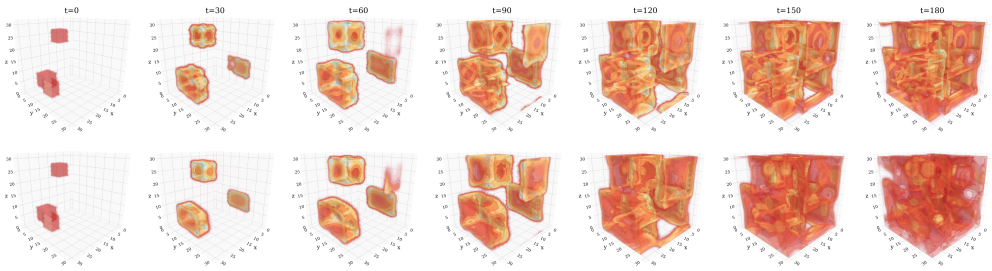


(b) $\boldsymbol{v}$ target (top) and transformer prediction (bottom)

Figure 16: Test case volume plots for the Gray–Scott system. Isosurfaces displayed span the range $\boldsymbol{u}, \boldsymbol{v} = [0.3, 0.5]$ to show the inner structure.



(a) $\boldsymbol{u}$ target (top) and transformer prediction (bottom)



(b) $\boldsymbol{v}$ target (top) and transformer prediction (bottom)

Figure 17: Test case volume plots for the Gray–Scott system. Isosurfaces displayed span the range $\boldsymbol{u}, \boldsymbol{v} = [0.3, 0.5]$ to show the inner structure.

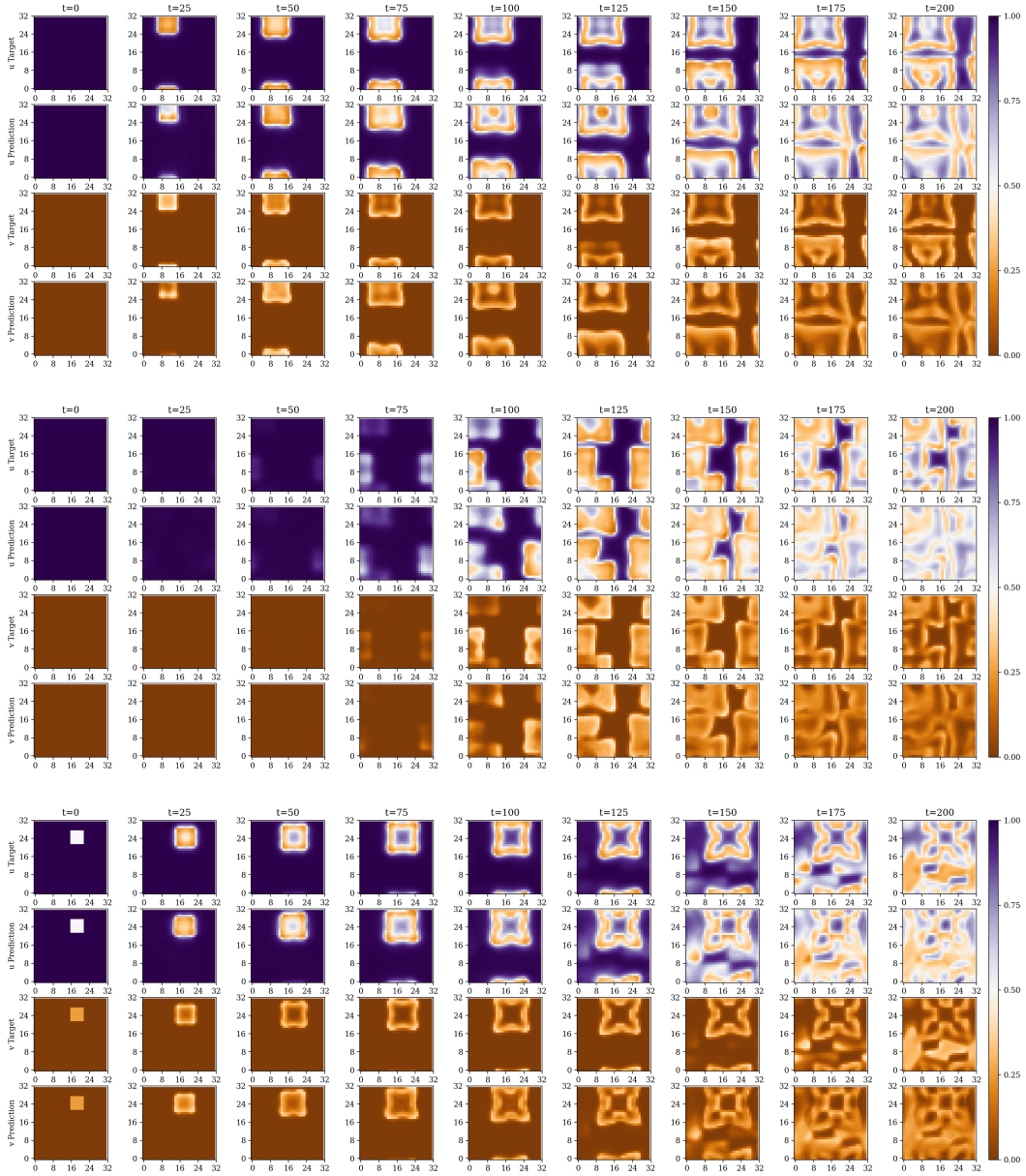


Figure 18: $x - y$ plane contour plots of three Gray–Scott test cases sliced at $z = 16$.