Learning Dynamical Systems Requires Rethinking Generalization

Rui Wang
UC San Diego
ruw020@ucsd.edu

Danielle Maddix
Amazon Research
dmmaddix@amazon.com

Christos Faloutsos
Amazon and CMU
faloutso@amazon.com

Yuyang Wang
Amazon Research
yuyawang@amazon.com

Rose Yu
UC San Diego
roseyu@eng.ucsd.edu

Abstract

The ability to generalize to unseen data is at the core of machine learning. A traditional view of generalization refers to unseen data from the same distribution. Dynamical systems challenge the conventional wisdom of generalization in learning systems due to distribution shifts from non-stationarity and chaos. In this paper, we investigate the generalization ability of dynamical systems in the forecasting setting. Through systematic experiments, we show deep learning models fail to generalize to shifted distributions in the data and parameter domains of dynamical systems. We find a sharp contrast between the performance of deep learning models on interpolation (same distribution) and extrapolation (shifted distribution). Our findings can help explain the inferior performance of deep learning models compared to physics-based models on the COVID-19 forecasting task.

1 Introduction

Conventional wisdom on generalization refers to the model's ability to adapt to unseen data. The underlying assumption is that the data is drawn independently and identically distributed (i.i.d) from the same distribution. Learning in dynamical systems violates such an assumption given its temporal dependency. Another challenge is the distribution shift: if the dynamics are non-stationary or chaotic, the distribution is constantly changing. Therefore, learning dynamical systems provides a natural venue for us to study generalization.

Dynamical systems [Day, 1994, Strogatz, 2018] are used to describe the evolution of phenomena occurring in nature, in which an evolution equation $d\boldsymbol{y}/dt = \boldsymbol{f_\theta}(\boldsymbol{y}, t)$ models the time dependence of the state $\boldsymbol{y}$, where $\boldsymbol{f_\theta}$ is a non-linear operator parameterized by a set of parameters $\boldsymbol{\theta}$. We consider the temporal dynamics forecasting problem of predicting an sequence of future states $\boldsymbol{y}_{t+1}, ..., \boldsymbol{y}_{t+q} \in \mathbb{R}^d$ given an sequence of historic states $\boldsymbol{y}_{t-k}, ..., \boldsymbol{y}_t \in \mathbb{R}^d$, where d is the feature dimension. We aim to learn a function $h \in \mathcal{H}$ that $h(\boldsymbol{y}_{t-k}, ..., \boldsymbol{y}_t) = \boldsymbol{y}_{t+1}, ..., \boldsymbol{y}_{t+q}$. Two distribution shift scenarios occur: non-stationary dynamics and dynamics changing with different parameters.

A plethora of work is devoted to learning dynamical systems. When $\boldsymbol{f_\theta}$ is known, numerical methods are most commonly used for estimating $\boldsymbol{\theta}$ [Houska et al., 2012]. When $\boldsymbol{f_\theta}$ is unknown, data-driven methods, such as deep sequence learning models [Flunkert et al., 2017, Rangapuram et al., 2018, Benidis et al., 2020, Sezer et al., 2019], including sequence to sequence models and Transformer [Vaswani et al., 2017, Wu et al., 2020, Li et al., 2020], have demonstrated success learning dynamical systems. Fully connected (FC) neural networks can also be used autoregressively to produce multiple time-step forecasts. Physics-informed models [Raissi and Karniadakis, 2018, Al-Aradi et al., 2018, Sirignano and Spiliopoulos, 2018] directly learn the solution of differential equations with neural networks given coordinates and time as input, which cannot be used for fore-

1st NeurIPS workshop on Interpretable Inductive Biases and Physically Structured Learning (2020), virtual.

casting since the future time would always lie outside of the training domain and neural networks are unreliable on unseen domain. [Chen et al., 2018, Wang et al., 2020, Ayed et al., 2019] have developed deep learning models integrated with differential equations, while making the strong assumption that the training and test data have the same domain.

Deep neural networks often struggle with distributional shifts [Kouw and Loog, 2018, Amodei et al., 2019] that naturally occur in learning dynamical systems. In forecasting, the data in the future lies outside the training domain, and requires methods to extrapolate to the unseen domain. This is in contrast to classical machine learning theory, where generalization refers to model adapting to unseen data drawn from the same distribution [Hastie et al., 2009, Poggio et al., 2012]. Learning dynamic systems requires the model to generalize to unseen data with shifted distributions in both the data and parameter domains.

In this work, we experimentally explore the two cases, distribution shift in the data and parameter domains, where four widely-used deep sequence learning models fail to learn and predict the correct dynamics. We show in a synthetic experiment that these models cannot handle a small vertical distribution shift when forecasting stationary *Sine* waves. We also study the task of forecasting three other non-linear dynamics: the *Lotka-Volterra*, *FitzHugh–Nagumo* and *SEIR* equations, and show that these models have poor generalization to the unseen parameter domain of dynamical systems.

2 Generalization in Learning Dynamical Systems

2.1 Dynamical Systems

Lotka-Volterra (*LV*) system of equations (2.1) describe the dynamics of biological systems in which predators and preys interact, where d denotes the number of species interacting and p_i denotes the population size of species i at time step t. The unknown parameters $r_i \geq 0$, $k_i \geq 0$ and A_{ij} denote the intrinsic growth rate of species i, the carrying capacity of species i when the other species are absent, and the interspecies competition between two different species, respectively.

FitzHugh–Nagumo (*FHN*) [FitzHugh, 1961] and, independently, [Nagumo et al., 1962] derived the equations (2.2) to qualitatively describe the behaviour of spike potentials in the giant axon of squid neurons. The system describes the reciprocal dependencies of the voltage x across an axon membrane and a recovery variable y summarizing outward currents. The unknown parameters a, b, and c are dimensionless and positive, and c determines how fast y changes relative to x.

SEIR system of equations (2.3) models the spread of infectious diseases Tillett1992Dynamics. It has four compartments: Susceptible (S) denotes those who potentially have the disease, Exposed (E) models the incubation period, Infected (I) denotes the infectious who currently have the disease, and Removed/Recovered (R) denotes those who have recovered from the disease or have died. The total population N is assumed to be constant and the sum of these four states. The unknown parameters β, σ and γ denote the transmission, incubation, and recovery rates, respectively.

$$\frac{dp_i}{dt} = r_i p_i \Big(1 - \frac{\sum_{j=1}^{d} A_{ij} p_j}{k_i}\Big), \quad i = 1, 2, \ldots, d. \tag{2.1}$$

$$\begin{cases} \dfrac{dx}{dt} = c(x + y - \dfrac{x^3}{3}), \\ \dfrac{dy}{dt} = -\dfrac{1}{c}(x + by - a). \end{cases} \tag{2.2}$$

$$\begin{cases} dS/dt = -\beta SI/N, \\ dE/dt = \beta SI/N - \sigma E, \\ dI/dt = \sigma E - \gamma I, \\ dR/dt = \gamma I, \\ N = S + E + I + R. \end{cases} \tag{2.3}$$

2.2 Interpolation vs. Extrapolation

Suppose $p_{\mathcal{S}}$ is the training data distribution and $p_{\mathcal{T}}$ is the test data distribution. Let $\mathcal{H}$ be a hypothesis class, and we aim to learn a function $h \in \mathcal{H}$ that $h(\boldsymbol{y}_{t-k}, ..., \boldsymbol{y}_t) = \boldsymbol{y}_{t+1}, ..., \boldsymbol{y}_{t+q}$, where $\boldsymbol{y}_i \in \mathbb{R}^d$. Let $l : (\mathbb{R}^{k \times d} \times \mathbb{R}^{q \times d}) \times \mathcal{H}$ be a loss function. The empirical risk is $\hat{L}(h) = \frac{1}{n} \sum_{i=1}^{n} l((\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}), h)$, where $(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}) \sim p_{\mathcal{S}}$ is the i^{th} of n training samples. The test error is given as $L(h) = \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{z}) \sim p_{\mathcal{T}}}[l((\boldsymbol{x}, \boldsymbol{z}), h)]$. Both $\boldsymbol{x}^{(i)}$ and $\boldsymbol{z}^{(i)}$ are sequences of states in our setting. Small $\hat{L}(h) - L(h)$ usually indicates good generalization. Apart from $p_{\mathcal{S}}$ and $p_{\mathcal{T}}$,

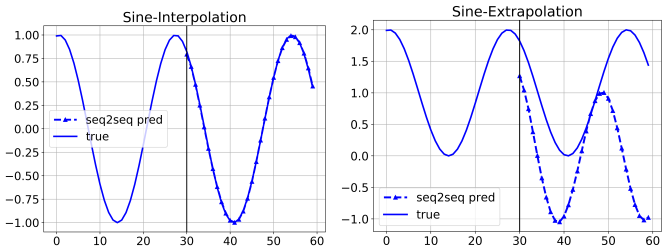

Figure 1: `Seq2Seq` predictions on an interpolation (left) and an extrapolation (right) test samples of Sine dynamics, the vertical black line in the plots separates the input and forecasting period.

RMSE	`Inter`	`Extra`
`Seq2Seq`	0.012	1.242
`Auto-FC`	0.009	1.554
`Transformer`	0.016	1.088
`NeuralODE`	0.012	1.214

Table 1: RMSEs of the interpolation and extrapolation tasks of Sine dynamics.

we also define the *parameter* distributions of training and test samples as $\theta_{\mathcal{S}}$ and $\theta_{\mathcal{T}}$, where the *parameter* here refers to the parameters and the initial values of dynamical systems.

We define two types of interpolation and extrapolation tasks. Regarding the data domain, we define a task as an interpolation task when the data domain of the test data is a subset of the domain of the training data, i.e., $\text{Dom}(p_{\mathcal{T}}) \subseteq \text{Dom}(p_{\mathcal{S}})$, and then extrapolation is occurs $\text{Dom}(p_{\mathcal{T}}) \not\subseteq \text{Dom}(p_{\mathcal{S}})$. Regarding the system parameter domain, an interpolation task indicates that $\text{Dom}(\theta_{\mathcal{T}}) \subseteq \text{Dom}(\theta_{\mathcal{S}})$, and an extrapolation task indicates that $\text{Dom}(\theta_{\mathcal{T}}) \not\subseteq \text{Dom}(\theta_{\mathcal{S}})$.

2.3 Generalization in dynamical systems: unseen data in the different data domain

Through a simple experiment on learning the *Sine* curves, we show deep sequence models have poor generalization on extrapolation tasks regarding the data domain, i.e. $\text{Dom}(p_{\mathcal{T}}) \not\subseteq \text{Dom}(p_{\mathcal{S}})$. We generate 2k *Sine* samples of length 60 with different frequencies and phases, and randomly split them into training, validation and interpolation-test sets. The extrapolation-test set is the interpolation-test set shifted up by 1. We investigate four models, including `Seq2Seq` (sequence to sequence with LSTMs), `Transformer`, `FC` (autoregressive fully connected neural nets) and `NeuralODE`. All models are trained to make 30 steps ahead prediction given the previous 30 steps. See the *Sine* subsection of Appendix A for details.

Table 1 shows that all models have substantially larger errors on the extrapolation test set. Figure 1 shows `Seq2Seq` predictions on an interpolation (left) and an extrapolation (right) test samples. We can see that `Seq2Seq` makes accurate predictions on the interpolation-test sample, while it fails to generalize when the same sample is shifted up only by 1.

2.4 Generalization in dynamical systems: unseen data with different system parameters

Even when $\text{Dom}(p_{\mathcal{T}}) \subseteq \text{Dom}(p_{\mathcal{S}})$, deep sequence models may fail to predict correct dynamics if there is a distributional shift in the parameter domain, i.e., $\text{Dom}(\theta_{\mathcal{T}}) \not\subseteq \text{Dom}(\theta_{\mathcal{S}})$. For each of the three dynamics in section 2.1, we generate 6k synthetic time series samples with different system parameters and initial values. The training/validation/interpolation-test sets for each dataset have the same range of system parameters while the extrapolation-test set contains samples from a different range. Table 2 shows the *parameter* distribution of test sets. For each dynamics, we perform two experiments to evaluate the models' extrapolation generalization ability on the initial values and the system parameters. All samples are normalized so that $\text{Dom}(p_{\mathcal{T}}) = \text{Dom}(p_{\mathcal{S}})$. See Appendix A for more details.

Table 2: The initial values and system parameters ranges of interpolation and extrapolation test sets.

	System Parameters		Initial Values	
	Interpolation	Extrapolation	Interpolation	Extrapolation
LV	$\boldsymbol{k} \sim U(0, 250)^4$	$\boldsymbol{k} \sim U(250, 300)^4$	$\boldsymbol{p_0} \sim U(30, 200)^4$	$\boldsymbol{p_0} \sim U(0, 30)^4$
FHN	$c \sim U(1.5, 5)$	$c \sim U(0.5, 1.5)$	$x_0 \sim U(2, 10)$	$x_0 \sim U(0, 2)$
SEIR	$\beta \sim U(0.45, 0.9)$	$\beta \sim U(0.3, 0.45)$	$I_0 \sim U(30, 100)$	$I_0 \sim U(10, 30)$

Table 3: RMSEs on initial values and system parameter interpolation and extrapolation test sets.

RMSE	*LV*				*FHN*				*SEIR*			
	$\boldsymbol{k}$		$\boldsymbol{p_0}$		c		x_0		β		I_0	
	Int	Ext	Int	Ext	Int	Ext	Int	Ext	Int	Ext	Int	Ext
Seq2Seq	0.050	0.215	0.028	0.119	0.093	0.738	0.079	0.152	1.12	4.14	2.58	7.89
FC	0.078	0.227	0.044	0.131	0.057	0.402	0.057	0.120	1.04	3.20	1.82	5.85
Transformer	0.074	0.231	0.067	0.142	0.102	0.548	0.111	0.208	1.09	4.23	2.01	6.13
NeuralODE	0.091	0.196	0.050	0.127	0.163	0.689	0.124	0.371	1.25	3.27	2.01	5.82

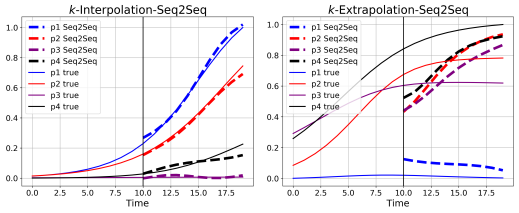

Figure 2: Seq2Seq predictions on a $\boldsymbol{k}$-interpolation (left) and a $\boldsymbol{k}$-extrapolation (right) test samples of *LV* dynamics, the vertical black line separates the input and forecasting period.

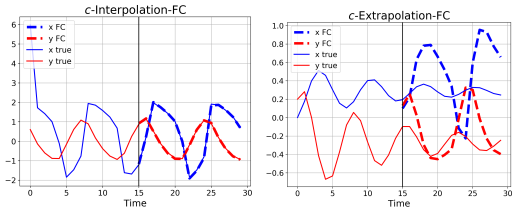

Figure 3: FC predictions on a c-interpolation (left) and a c-extrapolation (right) test samples of *FHN* dynamics, the vertical black line in the plots separates the input and forecasting period.

Table 3 shows the prediction RMSEs of the models on initial values and system parameter interpolation and extrapolation test sets. We observe that the models' prediction errors on extrapolation test sets are much larger than the error on interpolation test sets. Figures 2-3 show that Seq2Seq and FC fail to make accurate prediction when tested outside of the parameter distribution of the training data even though they make accurate predictions for parameter interpolation test samples. All experiments were run on Amazon Sagemaker [Liberty et al., 2020].

2.5 Case study: COVID-19 forecasting

The COVID-19 trajectories of the numbers of infected (I), removed (R) and death (D) cases can be considered as a dynamical system that is governed by complex ODEs. We perform a benchmark study by comparing the various deep learning models and ODE-based models on the task of 7-day ahead COVID-19 trajectories prediction. All details can be found in Appendix B. We observe that ODEs-based methods overall outperform the deep learning methods, especially for week July 13. One potential reason is that the number of cases in most states increase dramatically in July, and the test data is outside of the training data range. Neural networks are unreliable in this case as we show in section 2.3. Another potential reason is that we are still in the early or middle stage of the COVID-19 pandemic, which can affect the distribution of the unknown parameters. For instance, the contact rate β changes with government regulations, and the recovery rate γ may increase as we gain more treatment experience. Thus, there is a high chance that test samples are outside of the parameter domain of training data. In that case, the deep learning models would not make accurate predictions for COVID-19 as we show in section 2.4. See Appendix B for details.

3 Conclusion

We experimentally show that four deep sequence learning models fail to generalize to unseen data with shifted distributions in both the data and dynamical system parameter domains, even though these models are rich enough to memorize the training data, and perform well on interpolation tasks. This poses a challenge on learning real world dynamics with deep learning models. To achieve accurate prediction of dynamics, this work shows that we need to ensure that both the data and parameter domains of the training set are sufficient enough to cover the domains of the test set.

References

[Al-Aradi et al., 2018] Al-Aradi, A., Correia, A., Naiff, D., Jardim, G., and Saporito, Y. (2018). Solving nonlinear and high-dimensional partial differential equations via deep learning. *arXiv preprint arXiv:1811.08782*.

[Amodei et al., 2019] Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. (2019). Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*.

[Ayed et al., 2019] Ayed, I., Bézenac, E. D., Pajot, A., and Gallinari, P. (2019). Learning partially observed PDE dynamics with neural networks.

[Benidis et al., 2020] Benidis, K., Rangapuram, S. S., Flunkert, V., Wang, B., Maddix, D. C., Türkmen, A., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., Callot, L., and Januschowski, T. (2020). Neural forecasting: Introduction and literature overview. *ArXiv*, abs/2004.10240.

[Chen et al., 2018] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018). Neural ordinary differential equations. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 6571–6583. Curran Associates, Inc.

[Day, 1994] Day, R. H. (1994). Complex economic dynamics-vol. 1: An introduction to dynamical systems and market mechanisms. *MIT Press Books*, 1.

[Dong et al., 2020] Dong, E., Du, H., and Gardner, L. (2020). An interactive web-based dashboard to track covid-19 in real time. *Lancet Inf Dis.*, 20(5):533–534. doi:10.1016/S1473–3099(20)30120–1.

[FitzHugh, 1961] FitzHugh, R. (1961). Impulses and physiological states in theoretical models of nerve membrane. *Biophyiscal Journal.*, 1:445–466.

[Flunkert et al., 2017] Flunkert, V., Salinas, D., and Gasthaus, J. (2017). Deepar: Probabilistic forecasting with autoregressive recurrent networks. *ArXiv*, abs/1704.04110.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). Springer.

[Houska et al., 2012] Houska, B., Logist, F., Diehl, M., and Impe, J. V. (2012). A tutorial on numerical methods for state and parameter estimation in nonlinear dynamic systems. In Alberer, D., Hjalmarsson, H., and Re, L. D., editors, *Identification for Automotive Systems, Volume 418, Lecture Notes in Control and Information Sciences*, page 67–88. Springer.

[Kouw and Loog, 2018] Kouw, W. M. and Loog, M. (2018). An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*.

[Li et al., 2020] Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. (2020). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *arXiv preprint arXiv:1907.00235*.

[Liberty et al., 2020] Liberty, E., Karnin, Z., Xiang, B., Rouesnel, L., Coskun, B., Nallapati, R., Delgado, J., Sadoughi, A., Astashonok, Y., Das, P., Balioglu, C., Chakravarty, S., Jha, M., Gautier, P., Arpin, D., Januschowski, T., Flunkert, V., Wang, Y., Gasthaus, J., Stella, L., Rangapuram, S., Salinas, D., Schelter, S., and Smola, A. (2020). Elastic machine learning algorithms in amazon sagemaker. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, SIGMOD '20, page 731–737, New York, NY, USA. Association for Computing Machinery.

[Nagumo et al., 1962] Nagumo, J., Arimoto, S., and Yoshizawa, S. (1962). An active pulse transmission line simulating nerve axon. *Proceedings of the IRE*, 50(10):2061–2070.

[Poggio et al., 2012] Poggio, T., Rosasco, L., Frogner, C., and Canas, G. D. (2012). Statistical learning theory and applications.

[Raissi and Karniadakis, 2018] Raissi, M. and Karniadakis, G. E. (2018). Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141.

[Rangapuram et al., 2018] Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., and Januschowski, T. (2018). Deep state space models for time series forecasting. In *NeurIPS*.

[Sezer et al., 2019] Sezer, O. B., Gudelek, M. U., and Ozbayoglu, A. M. (2019). Financial time series forecasting with deep learning : A systematic literature review: 2005-2019. *arXiv preprint arXiv:1911.13288*.

[Sirignano and Spiliopoulos, 2018] Sirignano, J. and Spiliopoulos, K. (2018). Dgm: A deep learning algorithm for solving partial differential equations. *arXiv preprint arXiv:1708.07469*.

[Strogatz, 2018] Strogatz, S. H. (2018). *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. CRC press.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *ArXiv*.

[Wang et al., 2020] Wang, R., Kashinath, K., Mustafa, M., Albert, A., and Yu, R. (2020). Towards physics-informed deep learning for turbulent flow prediction. *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery and data mining*.

[Wu et al., 2020] Wu, N., Green, B., Ben, X., and O'Banion, S. (2020). Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv preprint arXiv:2001.08317*.

[Zou et al., 2020] Zou, D., Wang, L., Xu, P., Chen, J., Zhang, W., and Gu, Q. (2020). Epidemic model guided machine learning for covid-19 forecasts in the united states. *medRXiv preprint https://doi.org/10.1101/2020.05.24.20111989*.

A Additional Experiments Details

We use L2 loss for training and all hyperparamters, including number of layers, hidden dimension and learning rate, are tuned exhaustively on the validation set.

Sine We generate 2000 samples of length 60 from $\sin(wt + b)$. We set step size as 0.2, frequency $w \sim U(0.5, 1.5)$ and phase $b \sim U(0, 5)$. We shuffle and split these samples into 1200 training samples, 400 validation samples and 400 interpolation test samples.

SEIR We generate 6000 synthetic *SEIR* time series of length 60 based on Equ 2.3 with *scipy.integrate.odeint* with various parameters β, σ, γ and initial value I_0. First, we split all samples into a training set, a validation set, an interpolation test set and extrapolation test set based on the range of β. The training/validation/interpolation-test sets have the same range of $\beta \sim U(0.45, 0.9)$. The extrapolation-test set contains time series with $\beta \sim U(0.3, 0.45)$. The DL models are trained to make 40-step ahead predictions given the first 20 steps as input. We remove the trend of the trajectories of four variables by differencing. Then we investigate if the DL models can extrapolate to different initial I, so we also try training the models on times series with $I \sim U(30, 100)$, and test them on an I_0-interpolation test set where $I \sim U(30, 100)$ and an I_0-extrapolation test set where $I \sim U(1, 30)$.

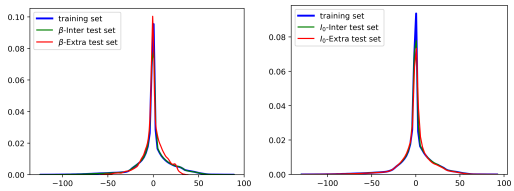

Figure 4: The data distribution of the training, $\beta(I_0)$-interpolation and $\beta(I_0)$-extrapolation test sets

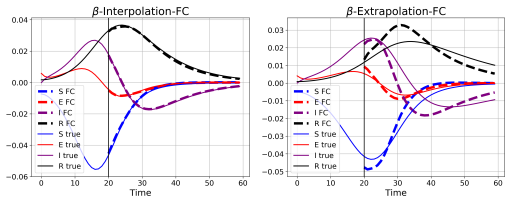

Figure 5: `FC` predictions on a β-interpolation (left) and a β-extrapolation (right) test samples of *SEIR* dynamics, the vertical black line in the plots separates the input and forecasting period.

LV We generate 6000 synthetic 4D *LV* time series of length 20. We normalize each sample so that all values are within the range of 0 and 1. The training/validation/interpolation-test sets have the same range of $\boldsymbol{k} \sim U(0, 250)^4$, the and extrapolation-test set contains time series with $\boldsymbol{k} \sim U(250, 300)^4$. We also investigate if the DL models can extrapolate to different initial values $\boldsymbol{p_0}$. We also train the models on samples with $\boldsymbol{p_0} \sim U(30, 200)^4$ and test them on $\boldsymbol{p_0} \sim U(0, 30)^4$ with same experimental setup.

FNH We generate 6000 synthetic *FNH* time series of length 50. Same as before, we test if the DL models can generalize to different range of parameters and initial values. The models are trained to make 25-step ahead predictions given the first 25 steps as input. c-interpolation test set contains sample with $c \sim U(1.5, 5)$ and c-extrapolation test set contains samples with $c \sim U(0.5, 1.5)$. x_0-interpolation test set contains sample with $x_0 \sim U(2, 10)$ and x_0-extrapolation test set contains sample with $x_0 \sim U(0, 2)$.

B Case study: COVID-19 forecasting

B.1 Proposed Method: `AutoODE`

We present our proposed `AutoODE` model that given an ODE in Eqn. (B.1) learns the unknown parameters with automatic differentiation using gradient-based methods. Unlike with neural networks, `AutoODE` is data-efficient, and the model only needs to be fit on the days before the prediction week. We apply this physics-based method to COVID-19 forecasting, using the ODEs in Eqn. (B.1) improved upon from the `SuEIR` model [Zou et al., 2020], where we estimate the unknown parameters

β_i, σ_i, μ_i, and γ_i, which denote the transmission, incubation, discovery, and recovery rates, respectively.

The total population $N_i = S_i + E_i + U_i + I_i + R_i$ is assumed to be constant for each of the U.S. states $i = 1, \ldots, n$.

$$\begin{cases} \dfrac{dS_i}{dt} = -\dfrac{\sum_{j=1}^{n}[\beta_i(t)A_{ij}(I_j + E_j)S_i]}{N_i}, \\ \dfrac{dE_i}{dt} = \dfrac{\sum_{j=1}^{n}[\beta_i(t)A_{ij}(I_j + E_j)S_i]}{N_i} - \sigma_i E_i, \\ \dfrac{dU_i}{dt} = (1-\mu_i)\sigma_i E_i, \\ \dfrac{dI_i}{dt} = \mu_i\sigma_i E_i - \gamma_i I_i, \\ \dfrac{dR_i}{dt} = \gamma_i I_i, \\ \dfrac{dD_i}{dt} = r_i(t)\dfrac{dR_i}{dt}. \end{cases} \tag{B.1}$$

Low Rank Approximation to the Transmission Matrix: A_{ij} We introduce a transmission matrix A to model the transmission rate among the 50 U.S. states. Each entry of A is the element-wise product of the sparse U.S. states adjacency matrix M and the correlation matrix C that is learned from data, that is, $A = C \odot M \in \mathbb{R}^{n\times n}$. We omit the transmission between the states that are not adjacent to each other to avoid overfitting. To reduce the number of parameters and improve the computational efficiency to $\mathcal{O}(kn)$, we use a low rank approximation to generate the correlation matrix $C = B^T D$, where $B, D \in \mathbb{R}^{k\times n}$ for $k << n$.

Piecewise Linear Transmission Rate: $\beta_i(t)$ Most compartmental models assume the transmission rate β_i is constant, which does not hold for COVID-19. The transmission rate of COVID-19 changes over time due to government regulations, such as school closures and social distancing. Even though we do short-term forecasting (7 days ahead), it is possible that the transmission rate may change during the training period. Instead of a constant approximation to β_i, we use a piecewise linear function over time $\beta_i(t)$, and set the breakpoints, slopes and biases as trainable parameters.

Death Rate Modeling: $r_i(t)$ Figure 6 shows the trajectories of the number of accumulated removed and death cases in four different states. We can see a relationship between the numbers of accumulated removed and death cases can be close to linear, exponential or concave. Since we do short-term forecasting, the death rate $r_i(t)$ can be assumed as a linear function $a_i t + b_i$ to cover both the convex and concave functions, where a_i and b_i are set as learnable parameters.

Numerical Integration To solve the coupled ordinary differential equations, we use the 4-th order Runge-Kutta Method (RK4) numericalbook. In the Neural ODE method chen19, the authors use the adjoint method to have the neural networks bypass the numerical solver, and be applicable to higher dimensional problems. In our case, since our method uses low dimension ordinary differential equations, RK4 is sufficient to generate accurate predictions. We directly implement RK4 in Pytorch, and allow backpropagation through it with a fixed time-step Δt.

Weighted Loss Function We set the unknown parameters in Eqn. (B.1) as trainable, and apply a gradient-based optimizer to minimize the following weighted loss function:

$$L(\boldsymbol{A}, \boldsymbol{\beta}, \boldsymbol{\sigma}, \boldsymbol{\mu}, \boldsymbol{\gamma}, \boldsymbol{r}) = \frac{1}{T}\sum_{t=1}^{T} w(t)\Big[l(\hat{I}_t, I_t) + \alpha_1 l(\hat{R}_t, R_t) + \alpha_2 l(\hat{D}_t, D_t)\Big],$$

with weights α_1, α_2 and loss function $l(\cdot,\cdot)$ to find the optimal parameters. We utilize these weights to balance the loss of the three states due to scaling differences, and also reweigh the loss at different time steps. We give larger weights to more recent data points by setting $w(t) = \sqrt{t}$. The constants, α_1, α_2 and T are tuned on the validation set. We set $l(\cdot,\cdot)$ to be the quantile loss MQCNN2018Wen for both `AutoODE` and the DL models.

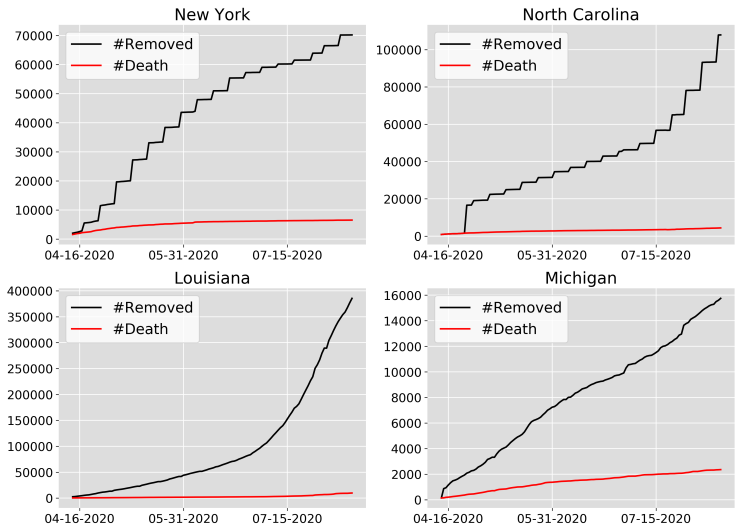

Figure 6: The trajectories of number of accumulated removed and death cases at New York, North Carolina, Louisiana and Michigan.

B.2 Experimental Results

Dataset We use the COVID-19 data from Apr 14 to Sept 12 provided by Johns Hopkins University [Dong et al., 2020]. It contains the number of cumulative number infected (I), recovered (R) and death (D) cases. Figure 7 shows the rolling averages and standard deviation intervals of daily increase time series in New York, Pennsylvania, Maryland and Virginia.

Experimental Setup We investigate the following six DL models on forecasting COVID-19 trajectories, sequence to sequence with LSTMs (`Seq2Seq`), `Transformer`, autoregressive fully connected neural nets (`FC`), `Neural ODE`, graph convolution networks (`GCN`) and graph attention networks (`GAN`). To train these DL models, we standardize I, R and D time series of each state individually to avoid one set of features dominating another. We use sliding windows to generate samples of sequences before the week that we want to predict and split them into training and validation sets. To train ODEs-based models, we rescale the trajectories of the number of cumulative cases of each state by the population of that state. We perform exhaustive search of the hyperparameters, including the learning rate, hidden dimensions and number of layers, for every DL model on the validation set. All these DL models are trained to predict the number of daily new cases instead of the number of accumulated cases because we want to detread the time series, and put the training and test samples in the same approximate range. For graphical models, we view each state as a node, and then the adjacency matrix is the US states adjacency matrix.

Results Table 4 shows the 7-day ahead prediction mean absolute errors of three features I, R and D for the weeks of July 13, Aug 23 and week Sept 6. We can see that `AutoODE` overall performs better than `SuEIR` and all the DL models. `FC` and `Seq2Seq` have better prediction accuracy of death counts. All DL models have much bigger errors on the prediction of week July 13, which may be due to insufficient training data. Another reason is that the number of cases in most states increase dramatically in July, and the test data is outside of the training data range, and neural networks are known to not be reliable in these cases Kouw2018domain, Amodei2016Safety. Figure 8 shows the 7-day ahead COVID-19 predictions of I, R and D in Massachusetts by `AutoODE` and the best DL model (`FC`). The prediction by `AutoODE` is closer to the target and has smaller confidence intervals. This demonstrates the effectiveness of our model, and the benefits of the combination of machine learning techniques with compartmental models.

Table 4: Proposed AutoODE wins in predicting I and R: 7-day ahead prediction MAEs on COVID-19 trajectories of accumulated number of infectious, removed and death cases.

MAE	$07/13 \sim 07/19$			$08/23 \sim 08/29$			$09/06 \sim 09/12$		
	I	R	D	I	R	D	I	R	D
FC	8379	5330	257	559	701	**30**	775	654	**33**
Seq2Seq	5172	2790	**99**	781	700	40	728	787	35
Transformer	8225	2937	2546	1282	1308	46	1301	1253	41
NeuralODE	7283	5371	173	682	661	43	858	791	35
GCN	6843	3107	266	1066	923	55	1605	984	44
GAN	4155	2067	153	1003	898	51	1065	833	40
SuEIR	1746	1984	136	639	778	39	888	637	47
AutoODE	**818**	**1079**	109	**514**	**538**	41	**600**	**599**	39

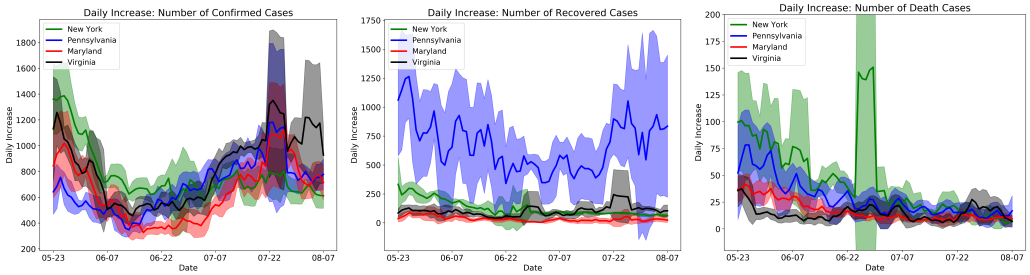

Figure 7: The rolling averages and standard deviation intervals of daily increase time series of four US states. Left: the number of confirmed cases; Middle: the number of recovered cases; Right: the number of death cases

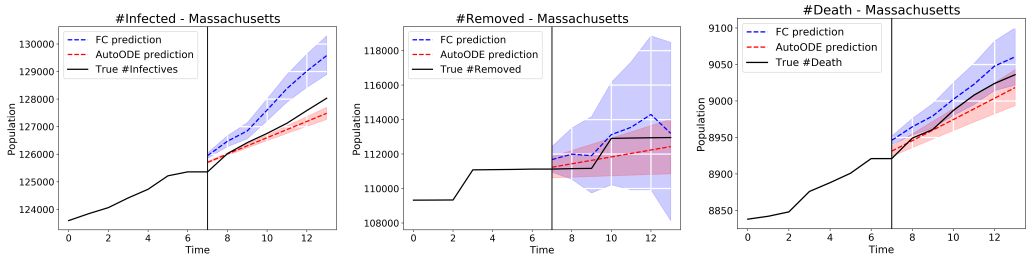

Figure 8: Proposed AutoODE wins: I, R and D predictions for week $08/23 \sim 08/29$ in Massachusetts by our proposed AutoODE model and the best performing DL model FC.