Using machine learning to predict extreme events in complex systems

Di Qi[a,b,1,2] and Andrew J. Majda[a,b,1,2]

[a]Department of Mathematics, Courant Institute of Mathematical Sciences, New York University, New York, NY 10012; and [b]Center for Atmosphere and Ocean Science, Courant Institute of Mathematical Sciences, New York University, New York, NY 10012

Contributed by Andrew J. Majda, November 8, 2019 (sent for review October 4, 2019; reviewed by Weinan E, J. Nathan Kutz, and Xiaoming Wang)

Extreme events and the related anomalous statistics are ubiquitously observed in many natural systems, and the development of efficient methods to understand and accurately predict such representative features remains a grand challenge. Here, we investigate the skill of deep learning strategies in the prediction of extreme events in complex turbulent dynamical systems. Deep neural networks have been successfully applied to many imaging processing problems involving big data, and have recently shown potential for the study of dynamical systems. We propose to use a densely connected mixed-scale network model to capture the extreme events appearing in a truncated Korteweg–de Vries (tKdV) statistical framework, which creates anomalous skewed distributions consistent with recent laboratory experiments for shallow water waves across an abrupt depth change, where a remarkable statistical phase transition is generated by varying the inverse temperature parameter in the corresponding Gibbs invariant measures. The neural network is trained using data without knowing the explicit model dynamics, and the training data are only drawn from the near-Gaussian regime of the tKdV model solutions without the occurrence of large extreme values. A relative entropy loss function, together with empirical partition functions, is proposed for measuring the accuracy of the network output where the dominant structures in the turbulent field are emphasized. The optimized network is shown to gain uniformly high skill in accurately predicting the solutions in a wide variety of statistical regimes, including highly skewed extreme events. The technique is promising to be further applied to other complicated high-dimensional systems.

anomalous extreme events | convolutional neural networks | turbulent dynamical systems

Extreme events and their anomalous statistics are ubiquitous in various complex turbulent systems such as the climate, material science, and neuroscience, as well as engineering design (1–4). Understanding and accurate prediction of such phenomena remain a grand challenge, and have become an active contemporary topic in applied mathematics (5–8). Extreme events can be isolated rare events (2, 9, 10), or they can often be intermittent and even frequent in space and time (6, 8, 11, 12). The curse of dimension forms one important obstacle for the accurate prediction of extreme events in large complex systems (3, 4, 6, 13), where both novel models and efficient numerical algorithms are required. A typical example can be found in recent laboratory experiments for turbulent surface water waves going through an abrupt depth change revealing a remarkable transition to anomalous extreme events from near-Gaussian incoming flows (1).

A statistical dynamical model is then proposed in refs. 14 and 15 that successfully predicts the anomalous extreme behaviors observed in the shallow water wave experiments. The truncated Korteweg–de Vries (tKdV) equation is proposed as the governing equation modeling the flow surface displacement. Gibbs invariant measures are induced based on the Hamiltonian form of the tKdV equation to describe the probability distributions at equilibrium. A statistical transition from symmetric near-Gaussian statistics to a highly skewed probability density function (PDF) is achieved by simply controlling the "inverse temperature" parameter in the Gibbs measure (15).

In recent years, machine learning strategies, particularly the deep neural networks, have been extensively applied to a wide variety of problems involving big data, such as image classification and identification (16–19). On the other hand, the construction of proper deep learning strategies for the study of complex turbulent flows still remains an actively growing topic. The deep neural network tools developed for imaging processing have been suggested to be applied for data-driven predictions of chaotic dynamical systems (20–22), climate and weather forecasts (23, 24), and parameterization of unresolved processes (25–27). In the statistical prediction of extreme events, the available data for training are often restricted in limited regimes. A successful neural network is required to maintain adaptive skill in wider statistical regimes with vastly distinct statistics away from the training dataset. Besides, a working prediction model for turbulent systems would also require a prediction time scale longer than the decorrelation time that characterizes the mixing rate of the state variables.

In this paper, we investigate the extent of skill of the deep neural networks in predicting statistical solutions of complex turbulent systems, especially involving highly skewed PDFs. The statistical tKdV equations serve as a difficult first test model for extreme

Significance

Understanding and predicting extreme events as well as the related anomalous statistics is a grand challenge in complex natural systems. Deep convolutional neural networks provide a useful tool to learn the essential model dynamics directly from data. A deep learning strategy is proposed to predict the extreme events that appear in turbulent dynamical systems. A truncated KdV model displaying distinctive statistics from near-Gaussian to highly skewed distributions is used as the test model. The neural network is trained using data only from the near-Gaussian regime without the occurrence of large extreme values. The optimized network demonstrates uniformly high skill in successfully capturing the solution structures in a wide variety of statistical regimes, including the highly skewed extreme events.

Author contributions: D.Q. and A.J.M. designed research, performed research, and wrote the paper.

Reviewers: W.E, Princeton University; J.N.K., University of Washington; and X.W., Florida State University.

The authors declare no competing interest.

Data deposition: The code created for this paper is made available on GitHub (https://github.com/qidigit/CNN_tKdV.git).

[1]D.Q. and A.J.M. contributed equally to this work.

[2]To whom correspondence may be addressed. Email: qidi@cims.nyu.edu or jonjon@cims.nyu.edu.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.1917285117/-/DCSupplemental.

First published December 23, 2019.

event prediction with simple trackable dynamics but a rich variety of statistical regimes from near-Gaussian to highly skewed PDFs showing extreme events. The important questions to ask are whether the deep networks can be trained to learn the complex hidden structures in the highly nonlinear dynamics purely from restricted data, and what are the essential structures required in the network to gain the ability to capture extreme events.

Our major goal here is to get accurate statistical prediction for the extreme events in time intervals significantly longer than the decorrelation time of the complex turbulent system. To achieve this, a convolutional neural network (mixed-scale dense neural network [MS-D Net]) which exploits multiscale connections and densely connected structures (28) is adopted to provide the basic network architecture to be trained using the model data from the tKdV equation solutions. This network enjoys the benefits of simpler model implementation and a smaller number of tunable training hyperparameters. Thus it becomes much easier to train, requiring less computational cost and technical tuning of the hyperparameters.

The key structures for the neural network to successfully capture extreme events include: 1) the use of a relative entropy (Kullback–Leibler divergence) loss function to calibrate the closeness between the target and the network output as distribution functions, so that the crucial shape of the model solutions is captured instead of a pointwise fitting in the turbulent output field values; and 2) calibrating the output data under a combination of empirical partition functions emphasizing the large positive and negative values in the model prediction so that the main features in the solutions are further emphasized. This convolutional neural network model enjoys the following major advantages for the prediction of extreme events: 1) The simple basic network architecture makes the model easier to train and efficient to predict the solutions among different statistical scenarios. 2) The network structure approximates the original model dynamics with the designed model loss function and treatment of output data, so it gives a better approximation to the complex structure of the system dynamics. 3) The temporal and spatial correlations in different scales are modeled automatically from the design of the network with convolution kernels representing different scales in different layers. 4) The method shows robust performance with different model hyperparameters, and can be generalized for the prediction of more complicated turbulent systems.

Direct numerical tests show high skill of the neural network in successfully capturing the extreme values in the solutions with the model parameters only learned from the near-Gaussian regime of vastly different statistics. The model also displays accurate prediction in much longer time beyond the decorrelation time scale of the state, proving the robustness of the methods. The successful prediction in the tKdV equation implies the potential of future applications of the network to more complicated high-dimensional systems.

Background for Extreme Events and Neural Network Structures

The tKdV Equations with Extreme Events. The tKdV model provides a desirable set of equations capable of capturing many complex features in surface water wave turbulence with simple trackable dynamics. Through a high wavenumber cutoff at Λ (with $J=2\Lambda+1$ grid points), the Galerkin projected state $u=\sum_{1\leq|k|\leq\Lambda}\hat{u}_k(t)e^{ikx}$ induces stronger turbulent dynamics than the original continuous one (15). The tKdV equation has been adapted to describe the sudden phase transition in statistics (14) where highly skewed extreme events are generated from near-Gaussian statistics for waves propagating across an abrupt depth change. The tKdV model is formulated on a periodic domain $x\in[-\pi,\pi]$ as

$$u_t+E_0^{1/2}L_0^{-3/2}D_0^{-3/2}uu_x+L_0^{-3}D_0^{1/2}u_{xxx}=0, \qquad [1]$$

where the state variable $u(x,t)$ represents the surface wave displacement to be learned directly using the deep neural network. The model is nondimensionalized using the characteristic scales E_0 as the total energy, L_0 as the length scale, and D_0 as the water depth. The steady-state distribution of the tKdV solution can be described by the invariant Gibbs measure derived from the equilibrium statistical mechanics (29),

$$\mathcal{G}_\theta(u)=C_\theta^{-1}\exp\left\{-\theta\left[E_0^{1/2}L_0^{-3/2}D_0^{-3/2}H_3(u)-L_0^{-3}D_0^{1/2}H_2(u)\right]\right\}\delta(E(u)-1), \qquad [2]$$

with a competition between the cubic term $H_3=1/6\int u^3$ and the quadratic term $H_2=1/2\int u_x^2$ from the Hamiltonian. The last term in Eq. **2** is the delta function constraining the total energy conservation, $E(u)=\frac{1}{2}\int u^2=1$. The only parameter $\theta<0$ as the inverse temperature determines the skewness of the PDF of u (14). The Gibbs measures [**2**] with different values of θ can be used to provide initial samples for the direct simulations of the model [**1**]. Different final equilibrium statistics (with various skewness) can be obtained based on the initial configuration of the ensemble set (that is, from picking different inverse temperature values θ). A detailed description of the statistical tKdV model together with the simulation setup is provided in *SI Appendix*, section A.

Training and prediction data from the same model with distinct statistics. The basic idea in training the deep neural network is to use a training dataset with solutions sampled from Eq. **2** using near-Gaussian statistics. The tKdV model dynamics can be learned from the training set without explicitly knowing the model dynamics. Then the question is, What is the range of skill in the trained neural network to predict the highly skewed non-Gaussian distributions among different sets of data? The training and prediction datasets are proposed from the ensemble solutions of the tKdV model [**1**] based on the following strategy:

1) In the training dataset, we generate solutions from an ensemble simulation starting from a near-Gaussian PDF using a small absolute value of the inverse temperature θ_0 (as shown in the first row and the near-Gaussian PDF in Fig. 1). On one hand, the model dynamics is represented by the group of solutions $\{u_{\theta_0}\}$ to be learned through the deep neural network. On the other hand, only near-Gaussian statistics is obtained in this training dataset, so the neural network cannot know about the skewed rare events appearing in other statistical regimes directly from the training process.
2) For the prediction dataset, we test the model skill using the data $\{u_\theta\}$ generated from various different initial inverse temperatures θ (as shown in the second and third rows as well as the skewed PDFs in Fig. 1). It provides an interesting test bed to check the scope of skill in the optimized neural network for capturing the distinctive statistics and extreme events.

The choices of the training and prediction datasets are illustrated in Fig. 1, which first shows, in the first three rows, realizations of the tKdV model solutions from different inverse temperatures θ. A smaller amplitude of θ gives near-Gaussian statistics in the model state u, while larger amplitudes of θ give trajectories with skewed PDFs. The corresponding equilibrium PDFs of the state u from direct ensemble dynamical solutions are compared next to illustrate explicitly the transition in statistics. Turbulent dynamics with multiscale structures are observed in all of the tKdV solutions. The autocorrelation function $\mathcal{R}_u(t)=\langle u(t)u(0)\rangle$ and the decorrelation time $T_{\mathrm{decorr}}=\int\mathcal{R}_u(t)$ are plotted in the last row of Fig. 1, confirming the rapid mixing in the solutions.

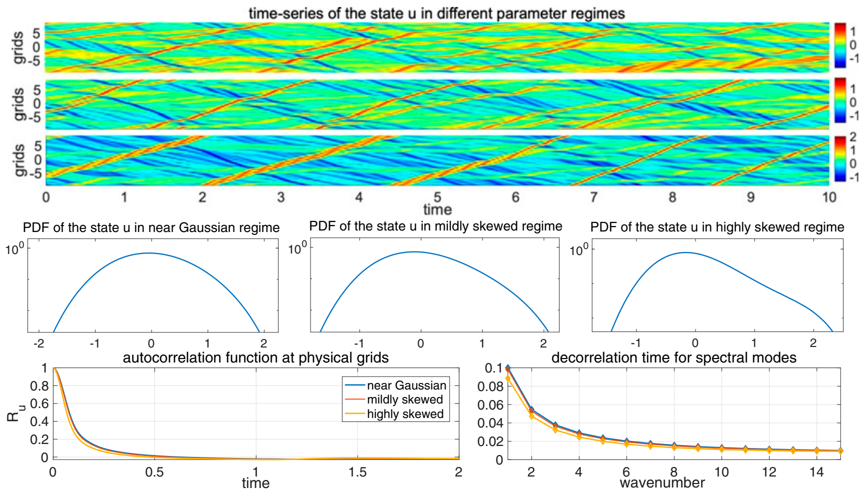

Fig. 1. Solutions and statistics of the tKdV equation in three typical parameter regimes with different statistics. The first three rows plot solution trajectories in the three regimes with near-Gaussian (first row), mildly skewed (second row), and highly skewed (third row) statistics. The corresponding equilibrium PDFs of the three cases are shown in the fourth row. The autocorrelation functions and decorrelation time of each Fourier mode of the model state *u* are compared in the last row.

Data structures for the deep neural networks. The deep convolutional networks can be viewed as a function $\mathbf{y}=f_M\left(\mathbf{x}\right)$, mapping the input signal $\mathbf{x}\in\mathbb{R}^{m\times n\times c}$ with m rows, n columns, and c channels to the output data $\mathbf{y}\in\mathbb{R}^{\mathrm{m}'\times\mathrm{n}'\times\mathrm{c}'}$ with m' rows, n' columns, and c' channels. We consider an ensemble simulation of M trajectories of the tKdV equation evaluated at the J grid points and measured in a time interval $[t_0,t_{N-1}]$. Thus the input data for the network comes from the ensemble solutions at the first N time measurements at $t_j=j\Delta t, j=0,\dots,N-1$,

$$\mathbf{x}^{(l)}=\left[u_0^{(l)},u_1^{(l)},\dots,u_{J-1}^{(l)}\right]^{\mathrm{T}}(t_0,\dots,t_{N-1})\in\mathbb{R}^{J\times N},$$

as a tensor with $m=J$ rows for the spatial grid points, $n=N$ columns for the discretized time evaluations, and only one channel $c=1$ for each of the input samples $l=1,\dots,M$. An ensemble of total M independent solutions from the Monte Carlo simulation is divided into minibatches to feed in the network in the training process. For simplicity, the output data are designed in the same shape as the predicted states evaluated at a later time $t=T+t_0$ starting from the previous initial data,

$$\mathbf{y}^{(l)}=\left[u_0^{(l)},u_1^{(l)},\dots,u_{J-1}^{(l)}\right]^{\mathrm{T}}(T+t_0,\dots,T+t_{N-1})\in\mathbb{R}^{J\times N}.$$

The forwarding time T controls how long we would like the network to push forward the states u in one time update. For one effective neural network for the complex system, the time scale T is expected to be longer than the decorrelation time, $T>T_{\mathrm{decor}}$. The above construction is supposed to feed both the time and spatial correlations of the original dynamical model into the neural network to be learned in the approximation map $\mathbf{y}=f_M(\mathbf{x})$.

Deep Convolutional Neural Network Architecture.

The basic structures of the convolutional neural network include the operators in each single convolution layer, and the connections between multiple layers. We would like to first keep the neural network in its simplest standard setup, so that we are able to concentrate on the improvement in key structures without risking getting lost in manipulating the various complicated ad hoc hyperparameters. More detailed convolutional network construction is described in *SI Appendix,* section B, following the general neural network architecture as in refs. 19 and 28.

Basic convolutional neural network unit. In each single convolution layer, the input data from the previous layer output are updated in the general form

$$\mathbf{y}=\sigma\left(g_h\left(\mathbf{x}\right)+b\right).$$

A convolutional operator g_h is first applied on the input data $\mathbf{x}$ in a small symmetric window with size $w\times w$, where the first dimension controls the correlation in the spatial direction and the second one controls the temporal correlation. A bias b is added to the convolved data before applying a final nonlinear operator σ using the common choice of rectified linear unit function. The convolution kernel starts with a small size, 3×3 (that is, using only the two nearest neighbor points in space and time), which enables fast computation and easy control. Naturally, periodic boundary condition is applied on the spatial dimension, and replicate boundary is added in time before t_0 and after t_{N-1} for the boundary padding. No additional structures are implemented in the convolution layer unit, to keep the basic standard architecture used in imaging processing (19).

A densely connected and mixed-scale structure. Next, we need to propose the connection between different layers. The common feedforward deep neutral network feeds the input data in the ith layer only to the next $(i+1)$th layer. The feedforward network often requires a larger number of layers to work; thus it is expensive to train and difficult to handle. Proper downscaling and upscaling steps going through the layers may also be required, while these downscaling and upscaling operations may not be a feasible approach for simulating the dynamical model time integration steps.

As an alternative approach, an MS-D Net is introduced, in ref. 28, by mixing different scales within each layer using a dilated convolution, and densely connecting all of the feature maps among all of the layers. First, to learn the multiscale structures, the convolution kernels in different layers are dilated differently by adding s zeros between the values in the original kernel $w\times w$. The dilated convolutions become especially appealing for capturing the multiscale structures in the turbulent dynamics. Different spatial and temporal scales are included adaptively with different convolution length scales. Second, the dense network connection includes all of the previous layer information to update the output data in the next layer. In the implementation, all of the previous layer outputs are piled together as input channels for the next layer. Together with the multiscale convolution kernels used in different layers, the output in the next layer combines the information in different scales and produces a balanced update in the next step.

The MS-D Net requires fewer feature maps and trainable parameters, so it is easier to handle compared with the direct feedforward network. It provides a desirable setup for the prediction in dynamical systems by feeding in all of the data in previous layers decomposed into different-scale structures. Then dynamical structures at different scales communicate with each other through the dense network connection. Intuitively, this is a reasonable structure for the turbulent solutions, since all of the history information is useful for the prediction in the next steps. The densely connected network structure is also comparable to the time integration scheme, where all of the history information is used to update the state at the next time step without using any downscaling and upscaling steps.

A Learning Strategy for Extreme Event Prediction

In this section, we construct the specific network structures designed for learning turbulent system dynamics and then the prediction of extreme events. In this case, with data from the turbulent models, small fluctuations in the solutions may introduce large errors in optimizing the loss function. We aim at capturing the dominant emerging features such as extreme events and are more interested in the statistical prediction than in the exact locations of the extreme values.

Loss Function Calibrating the Main Dynamical Features. In the training process, the model parameters are achieved through the optimization for the proper loss function (or cost) proposed depending on the target to be captured using the network. The two popular standard choices for the loss functions are using the L_1 error and the mean-square L_2 error: The L_1 error loss criterion measures the mean absolute error between each element in the output data $\mathbf{y}$ and target $\mathbf{z}$ through the L_1 distance,

$$l_1(\mathbf{y},\mathbf{z}) = \frac{1}{M}\sum_{m=1}^{M}\left\|\mathbf{y}^{(m)}-\mathbf{z}^{(m)}\right\|, \qquad [3]$$

where M is the training data size in one cycle, and $\left(\mathbf{y}^{(m)},\mathbf{z}^{(m)}\right)$ is one member of output data and target data from the training minibatch. L_2 error loss criterion measures the mean-squared error between each element in the output $\mathbf{y}$ and target $\mathbf{z}$ through the mean-square L_2 norm,

$$l_2(\mathbf{y},\mathbf{z}) = \frac{1}{M}\sum_{m=1}^{M}\left\|\mathbf{y}^{(m)}-\mathbf{z}^{(m)}\right\|^2, \qquad [4]$$

among all of the numbers of training samples M.

The above two loss functions offer pointwise measurements of the errors in space-time for each predicted sample from the network. This may cause problems, especially when the system for prediction is highly turbulent, with internal instability, and is fast mixing. Small error perturbation in the input data may lead to vastly different solutions shortly after the mixing decorrelation time. The pointwise measurements focus on the accuracy at each value of the solutions; thus the small fluctuation errors might be accumulated and amplified under such metrics, and unnecessary large weights are added to correct errors in the moderate-amplitude fluctuation parts.

On the other hand, we are most interested in the prediction of statistical features in the extreme events rather than the exact trajectory solutions of the system. The small shifts in the extreme value locations should be tolerated in the loss function. Therefore, a more useful choice could be the relative entropy loss function that measures the Kullback–Leibler divergence in the predicted density functions: The relative entropy loss function computes the distance between two distribution functions,

$$l_{\mathrm{KL}}(\tilde{\mathbf{y}},\tilde{\mathbf{z}}) = \frac{1}{M}\sum_{m=1}^{M}\sum_{i}\tilde{z}_i^{(m)}\log\frac{\tilde{z}_i^{(m)}}{\tilde{y}_i^{(m)}}, \qquad [5]$$

where the superscript m represents the minibatch members to be measured in the relative entropy metric, and the subscript i goes through all of the dimensions of the normalized variables $(\tilde{\mathbf{y}},\tilde{\mathbf{z}})$ to be described next.

Under the relative entropy loss function in Eq. **5**, the input data $\mathbf{y}$ and $\mathbf{z}$ are treated as distribution functions. The "shapes" between the output data and the target are compared, rather than the pointwise details, so that the loss function guides the network to focus on the main model dynamical features instead of the turbulent fluctuations that are impossible to fit accurately. The additional difficulty in training the network using the relative entropy loss function is the constraint on the form of the output data to measure. The input of the relative entropy is required to be in the form of a density distribution function.

Scaling the Output Data with Empirical Partition Functions. In this case of using the relative entropy loss function, we need to propose proper preprocessing of the output and target data to fit the

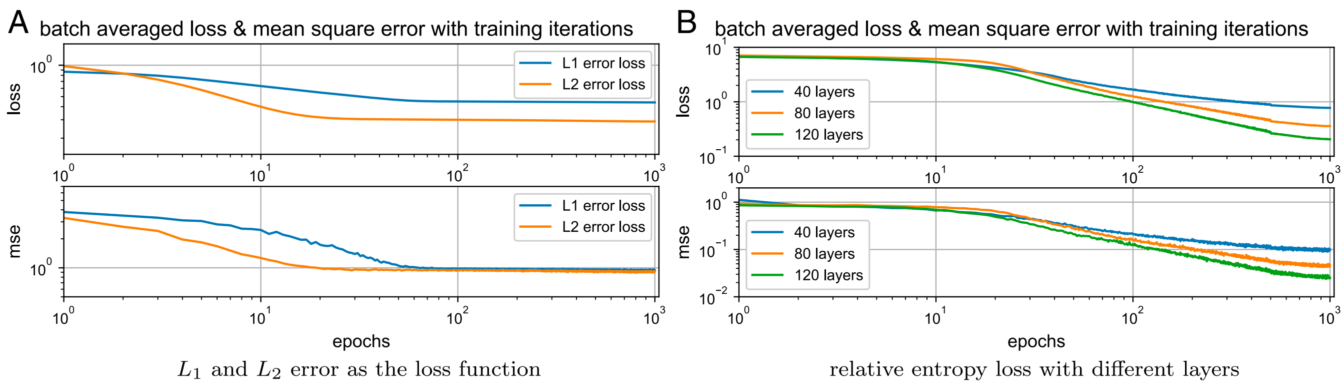

Fig. 2. Training loss function and the mean relative square error in the data (*A*) using L_1 and L_2 error loss functions and (*B*) using the relative entropy loss function with rescaled output data during the training iterations. Both networks with L_1 and L_2 loss are set to have $L = 80$ densely connected layers; the networks with the relative entropy loss are compared using $L = 40, 80, 120$ layers.

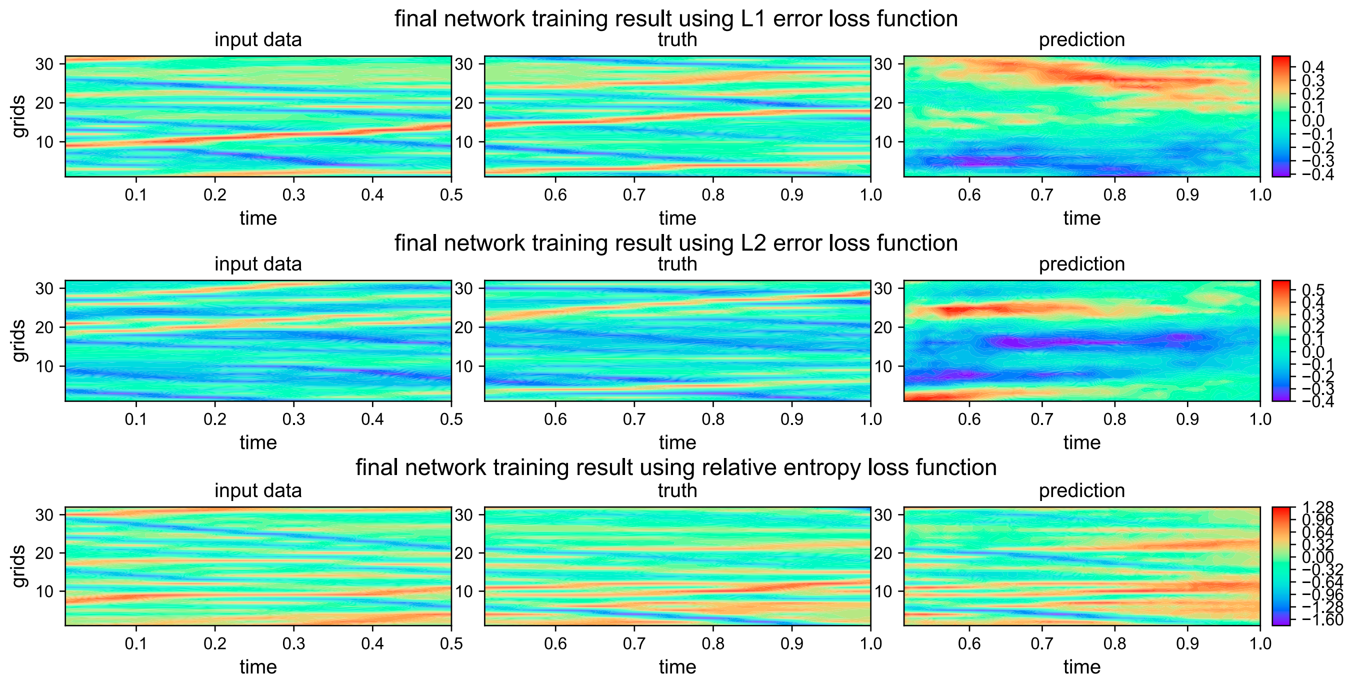

Fig. 3. One snapshot of the final training results with three different loss functions. (*Left*) the input data, (*Middle*) the true target to fit, and (*Right*) the output results from the trained networks. All networks contain $L = 80$ layers in the tests.

required structure as a probability distribution. One direct way to do so is by taking the softmax function for the output data from the neural network normalized by a partition function,

$$\tilde{y}_i = \frac{\exp(y_i)}{\sum_i \exp(y_i)}, \qquad [6]$$

before measuring the error in the loss function. In this way, the data to put into the relative entropy loss function are normalized inside the range $[0, 1]$ with summation 1. This agrees with the definition in the relative entropy inputs. More importantly, this normalization emphasizes the large positive values of the data. Thus it offers a better calibration for the occurrence of positive major flow structures to be captured in the solutions.

Furthermore, a better choice for balancing both the positive and negative dominant values in the training data is to introduce scales with "temperatures." We use the following two empirical partition functions with both positive and negative coefficients to rescale the output data:

$$\tilde{y}_i^+ = \frac{\exp(y_i/T_+)}{\sum_i \exp(y_i/T_+)}, \quad \tilde{y}_i^- = \frac{\exp(-y_i/T_-)}{\sum_i \exp(-y_i/T_-)}, \qquad [7]$$

where $T_+ > 0$, $T_- > 0$ are the positive and negative temperatures weighing the importance of dominant large-amplitude features in the scaled measure. Accordingly, the loss function to minimize under the relative entropy metric becomes a combination with the two empirical partition functions,

$$l_{\mathrm{EPF}}(\mathbf{y}, \mathbf{z}) = l_{\mathrm{KL}}(\tilde{\mathbf{y}}^+, \tilde{\mathbf{z}}^+) + \alpha l_{\mathrm{KL}}(\tilde{\mathbf{y}}^-, \tilde{\mathbf{z}}^-), \qquad [8]$$

where we use $\alpha > 0$ as a further balance between the positive and negative temperature components. In this combined empirical partition function metric using Eqs. **7** and **8**, the major flow structures in the turbulent field represented by the dominant extreme values are better characterized from both the positive and negative sides in the statistics of the model. In the following computational experiments, we always pick $T_+ = T_- = 1$ and $\alpha = 1$ for simplicity. More discussion of role of the balance weight α is provided in *SI Appendix*, section B.

Model Performance in Training and Prediction Stages

Using the previous model construction, we show the training and prediction performance using the MS-D Net combined with the relative entropy loss function and rescaled output data using empirical partition functions applied on the tKdV equations. In the numerical tests, we first consider the optimal prediction skill using the deep neural network within one updating cycle. Especially, we are more interested in capturing the non-Gaussian statistics from the network rather than the exact recovery of the single time series which should give large difference with small perturbations due to its turbulent nature. The basic strategy is to train the model using data from the near-Gaussian solutions with a small inverse temperature θ_0 in the Gibbs measure [**2**]; then the optimized neural network is used to predict the more skewed model statistics for regimes with larger absolute values of θ.

For simplicity, we set the input and output data of the network in the same shape. Then the model output from one single iteration of the network is compared with the target data from the true model solution. In the structure of the neural network, the standard setup is adopted for the tKdV solutions. In each layer of

Table 1. Mean and variance of the relative square errors among a test with 500 samples for the state *u* and the scaled state exp (*u*) using the same trained network for different statistical regimes

Error	Near-Gaussian	Mildly skewed	Highly skewed
u			
Mean	0.2682	0.2556	0.2690
Variance	0.0039	0.0048	0.0087
exp (*u*)			
Mean	0.0733	0.0764	0.0985
Variance	0.0005	0.0011	0.0060

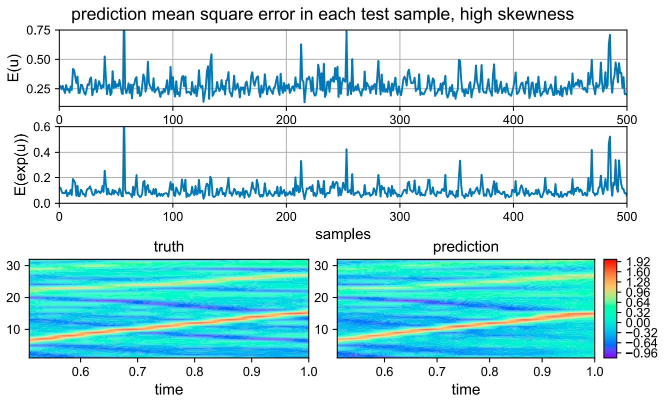

Fig. 4. Prediction in the regime with highly skewed statistics using the trained neural network with $L = 80$ layers. (*Top* and *Middle*) The relative square errors for the state u and the scaled state $\exp(u)$ among the 500 tested samples. (*Bottom*) One typical snapshot of the prediction compared with the truth.

the network, a kernel with minimum size $w = 3$ is taken for the convolution update. The dilation size for mixed scales changes from $s = 0$ to $s = 5$ repeatedly as the network grows in depth. Different network depths of layers L are tested for the model performance, while it is found that a moderate choice $L = 80$ is enough to produce desirable training and prediction results.

In calibrating the errors from model predictions, we propose the normalized square error between the network output $\mathbf{y}$ and the true target $\mathbf{z}$,

$$E(\mathbf{y}, \mathbf{z}) = \frac{\sum_{i=1}^{JN} |f(y_i) - f(z_i)|^2}{\sum_{i=1}^{JN} |f(z_i)|^2}, \qquad [9]$$

where the subscript i represents the ith component in the prediction/target set $\mathbf{y}, \mathbf{z} \in \mathbb{R}^{J \times N}$. The function $f(y)$ acting on each component of $\mathbf{y}$ can be used to extract the useful features to be calibrated. In the following tests, we use $f(y) = y$ to compare the original output of the data, and use the exponential scaling $f(y) = \exp(y)$ to check the prediction in positive extreme values.

Training the Neural Network Using Near-Gaussian Data. In the training process for turbulent system statistics, we include the temporal and spatial correlations together in the input data by considering a short time series of the solution. The training data are drawn from the model solutions of Eq. **1** only among the near-Gaussian regime statistics. In summary, we use the training dataset in the following structure:

1) The input data are from the ensemble solutions $u_j^{(m)}(t_n)$ of the tKdV equation. It is organized in a tensor of the shape (M, J, N), where $M = 10{,}000$ is the total ensemble size, $J = 32$ is the spatial discretization size, and $N = 50$ is the sampled time instants with the time step $\Delta t = 0.01$. Thus the initial data for training are given as the tKdV solutions in the time window $[0, 0.5]$.
2) The target data for the training result are the solutions $u_j^{(m)}(T + t_n)$ of the tKdV solution. The data are organized into the tensor with the same size (M, J, N) as the input data. We can consider different starting times for the target data by changing the time length T. The prediction time scale here is taken as $T = 0.5$, that is, to consider the prediction in time window $[0.5, 1]$.
3) The input data (with total ensemble size 10,000) are divided into 100 minibatches with size 100 for each training group in one epoch. In total, we use 1,000 epochs in the entire training process. The minibatches are randomly selected, with the batch indices resampled from random numbers in each step.

Notice that the above prediction time length $T = 0.5$ used in the experiments is much longer than the decorrelation time of the tKdV states. From the direct numerical results in Fig. 1, the autocorrelation function decays to near zero at $t = 0.5$, and the longest decorrelation time among the spectral modes is below 0.1.

First, we compare the training performance using the standard L_1 and L_2 loss functions in Eqs. **3** and **4** with the same MS-D Net structure. Fig. 2*A* shows the evolution of training loss functions and the mean relative square errors **[9]** among the training samples during the stochastic gradient descent iterations. According to the loss functions under the L_1 and L_2 distances, the training appears to be effective, and the error quickly dropped to smaller values in the first few steps. However, if we compare more carefully regarding the relative errors in the results, both cases get saturated quickly at a high error level near 1. The errors then become difficult to improve by training with a larger number of iterations and applying deeper layers. This is because, under both metrics, the model tries hard to fit the small-scale turbulent fluctuations in small values while missing the most important large-scale events in the solutions. It can be seen more clearly in Fig. 3 for typical training output snapshots compared with the truth. No desirable prediction can be reached.

In contrast, significant improvement is achieved by switching to the relative entropy loss function and adopting empirical partition functions to normalize the output data. In this case as illustrated in Fig. 2*B*, both the relative entropy loss function and the relative error drop to very small values in the final steps of the training iterations, implying high skill of the network in producing accurate predictions in the prediction time range (although it appears that the accuracy could be further improved by applying more iteration steps in the training, the results are already good enough after about 200 iterations). Note that we use logarithmic coordinates so the small values are emphasized. According to the last row of Fig. 3, for one typical training result snapshot, both the extreme values and the small-amplitude structures are captured in the model.

As a final remark, we check the proper depth needed for accurate predictions in the neural network. Fig. 2*B* also compares the same network under relative entropy loss but using different numbers of layers. A deeper network clearly can

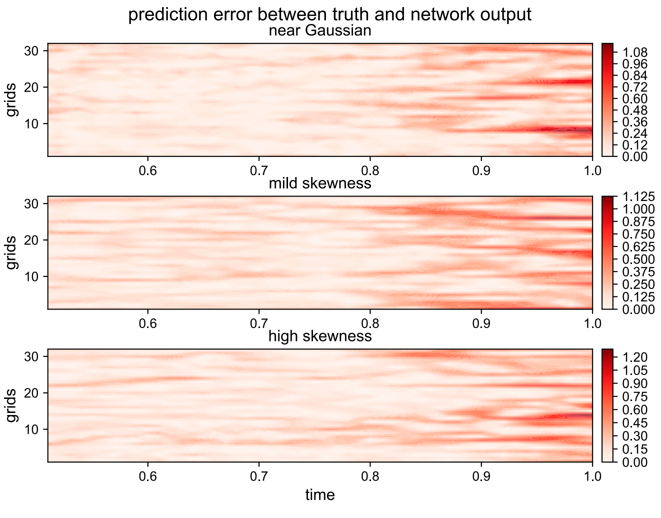

Fig. 5. The prediction error in the absolute difference between the truth and model output in the three tested regimes with different statistics.

further improve the prediction skill and push the final optimized error to an even lower value, with the cost of a larger computational requirement. Still, from the comparison, it shows that a moderate number of layers (such as $L = 80$) is sufficient to produce accurate results with relatively low cost. By pushing the network to deeper layers with $L = 120$, the improvement in error just becomes small and may not be necessary with the additional computational cost. The last row of Fig. 3 shows already the quite accurate recovery of the solution field purely learned from data.

Predicting Extreme Events Using Deep Neural Network. In checking the prediction skill of the optimized network, we pick the neural network with $L = 80$ densely connected layers as the standard model to test its predictions among different statistical regimes. It has been shown, in the training process, to have a high skill in recovering the original flow structures. Next, we should confirm that the neural network has really learned the dynamical structure of the original model, instead of merely overfitting the data.

Three statistical regimes ranging from near-Gaussian, mildly skewed, and highly skewed PDFs as shown in Fig. 1 are taken, for testing the range of prediction skill in the neural network model. An ensemble of 500 new trajectories from the tKdV solutions in different statistical regimes is used to show the robustness of the method. The mean and variance of the relative square errors [**9**] among the samples for the state u and the errors under the exponential function $\exp(u)$ are listed in Table 1 for different statistical regimes. Uniform high accuracy in the mean with tiny variance is achieved among the vastly different regimes with distinct statistical features.

Especially, we are interested in the case with highly non-Gaussian statistics representing the frequent occurrence of extreme events. In Fig. 4, the network is used to predict the flow solutions in the regime with highly skewed statistics (results for the other two cases can be found in *SI Appendix*, section B). The extreme values are represented by high peaks of a dominant wave moving along the field. This feature is not shown at all in the training data, where only near-Gaussian statistics is presented. As shown in the results, the trained network displays uniform skill among all of the tested samples in capturing the exact dynamical solutions in the extreme event domain unknown from the training data. By looking at the errors in the scaled data using the exponential function, the error amplitude even becomes smaller, confirming the accurate characterization of large extreme values through the network. In the typical snapshot of one sample, both the extreme values in the transporting waves and nonextreme detailed turbulent fluctuating structures are captured by the model.

Furthermore, the prediction errors as the absolute difference between the truth and the model prediction in the three statistical regimes are displayed in Fig. 5. The network predictions maintain accuracy with small errors in a much longer time scale than the decorrelation time $T_{\mathrm{decorr}} < 0.1$ of the states. This gives final confirmation of the general high skill in the deep neural network for capturing key representative features in complex dynamical systems once the essential structures are learned from the training procedure.

Concluding Remarks

A strategy using a densely connected multiscale deep neural network with relative entropy loss function for calibrating rescaled output data is proposed for the prediction of extreme events and anomalous features from data. It needs to be noticed that the extreme events are often represented by highly skewed PDFs and have frequent occurrence in the turbulent field (3, 8, 12), in contrast to the other situation of isolated rare events which can be studied with machine learning models (9). The prediction skill of the optimized deep neural network is tested on the tKdV equation, where different Gibbs states create a wide range of statistics from near-Gaussian to highly skewed distributions. By adopting the densely connected and multiscale structures, the deep neural network is easy to train with a standard model setup and fewer model hyperparameters. Many further investigations about the optimized structure of the network, such as the minimum required number of layers as well as a cost function as a combination of the L_1 or L_2 loss function together with the relative entropy loss, for efficient and accurate prediction in general turbulent systems can be considered next, based on the basic framework.

Using training data only drawn from the near-Gaussian regime of the dynamical model, the deep neural network displays high skill in learning the essential dynamical structures of the complex system and provides uniformly accurate prediction among a wide range of different regimes with distinct statistics. The network also shows robustness among tests in a large ensemble of samples. The robust performance in the test model implies the potential of more general applications using the neural network framework for the prediction of extreme events and important statistical features in a wider group of more realistic high-dimensional turbulent dynamical systems.

ACKNOWLEDGMENTS. This research of A.J.M. is partially supported by the Office of Naval Research Grant N00014-19-S-B001. D.Q. is supported as a postdoctoral fellow on the grant.

1. C. T. Bolles, K. Speer, M. N. J. Moore, Anomalous wave statistics induced by abrupt depth change. *Phys. Rev. Fluids* **4**, 011801 (2019).
2. G. Dematteis, T. Grafke, E. Vanden-Eijnden, Rogue waves and large deviations in deep sea. *Proc. Natl. Acad. Sci. U.S.A.* **115**, 855–860 (2018).
3. D. Qi, A. J. Majda, Predicting extreme events for passive scalar turbulence in two-layer baroclinic flows through reduced-order stochastic models. *Commun. Math. Sci.* **16**, 17–51 (2018).
4. M. A. Mohamad, T. P. Sapsis, Sequential sampling strategy for extreme event statistics in nonlinear dynamical systems. *Proc. Natl. Acad. Sci. U.S.A.* **115**, 11138–11143 (2018).
5. D. Qi, A. J. Majda, Predicting fat-tailed intermittent probability distributions in passive scalar turbulence with imperfect models through empirical information theory. *Commun. Math. Sci.* **14**, 1687–1722 (2016).
6. A. J. Majda, X. T. Tong, Intermittency in turbulent diffusion models with a mean gradient. *Nonlinearity* **28**, 4171–4208 (2015).
7. C. Viotti, F. Dias, Extreme waves induced by strong depth transitions: Fully nonlinear results. *Phys. Fluids* **26**, 051705 (2014).
8. D. Cai, A. J. Majda, D. W. McLaughlin, E. G. Tabak, Dispersive wave turbulence in one dimension. *Phys. D Nonlinear Phenom.* **152**, 551–572 (2001).
9. S. Guth, T. P. Sapsis, Machine learning predictors of extreme events occurring in complex dynamical systems. *Entropy*, **21**, 925 (2019).
10. W. Cousins, T. P. Sapsis, Unsteady evolution of localized unidirectional deep-water wave groups. *Phys. Rev. E* **91**, 063204 (2015).
11. A J Majda, Y. Lee, Conceptual dynamical models for turbulence. *Proc. Natl. Acad. Sci. U.S.A.* **111**, 6548–6553 (2014).
12. I. Grooms, A. J. Majda, Stochastic superparameterization in a one-dimensional model for wave turbulence. *Commun. Math. Sci.* **12**, 509–525 (2014).
13. N. Chen, A. J. Majda, Beating the curse of dimension with accurate statistics for the Fokker–Planck equation in complex turbulent systems. *Proc. Natl. Acad. Sci. U.S.A.* **114**, 12864–12869 (2017).
14. A. J. Majda, M. N. J. Moore, D. Qi, Statistical dynamical model to predict extreme events and anomalous features in shallow water waves with abrupt depth change. *Proc. Natl. Acad. Sci. U.S.A.* **116**, 3982–3987 (2019).
15. A. J. Majda, D. Qi, Statistical phase transitions and extreme events in shallow water waves with an abrupt depth change. *J. Statist. Phys.*, 10.1007/s10955-019-02465-3.
16. J. Schmidhuber, Deep learning in neural networks: An overview. *Neural Netw.* **61**, 85–117 (2015).
17. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. *Nature* **521**, 436–444 (2015).
18. M. I. Jordan T. M. Mitchell, Machine learning: Trends, perspectives, and prospects. *Science* **349**, 255–260 (2015).
19. I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning* (MIT Press, 2016).
20. C. Ma, J. Wang, W. E, Model reduction with memory and the machine learning of dynamical systems. *Commun. Comput. Phys.* **25**, 947–962 (2018).

21. J. Pathak, B. Hunt, M. Girvan, Z. Lu, E. Ott, Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.* **120**, 024102 (2018).
22. M. Raissi, G. E. Karniadakis, Hidden physics models: Machine learning of nonlinear partial differential equations. *J. Comput. Phys.* **357**, 125–141 (2018).
23. T. Bolton, L. Zanna, Applications of deep learning to ocean data inference and subgrid parameterization. *J. Adv. Model. Earth Syst.* **11**, 376–399 (2019).
24. J. A. Weyn, D. R. Durran, R. Caruana, Can machines learn to predict weather? Using deep learning to predict gridded 500-hPa geopotential height from historical weather data. *J. Adv. Model. Earth Syst.* **11**, 2680–2693 (2019).
25. J. Han, A. Jentzen, W. E, Solving high-dimensional partial differential equations using deep learning. *Proc. Natl. Acad. Sci. U.S.A.* **115**, 8505–8510 (2018).
26. W. E, J. Han, A. Jentzen, Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Commun. Math. Statist.* **5**, 349–380 (2017).
27. N. D. Brenowitz, C. S. Bretherton, Prognostic validation of a neural network unified physics parameterization. *Geophys. Res. Lett.* **45**, 6289–6298 (2018).
28. D. M. Pelt, J. A. Sethian, A mixed-scale dense convolutional neural network for image analysis. *Proc. Natl. Acad. Sci. U.S.A.* **115**, 254–259 (2018).
29. A. J. Majda, X. Wang, *Nonlinear Dynamics and Statistical Theories for Basic Geophysical Flows* (Cambridge University Press, 2006).