Finite-Time Lyapunov Exponents of Deep Neural Networks

Citation for the original published paper (version of record):
Storm, L., Linander, H., Bec, J. et al (2024). Finite-Time Lyapunov Exponents of Deep Neural Networks. Physical Review Letters, 132(5). http://dx.doi.org/10.1103/PhysRevLett.132.057301

Finite-Time Lyapunov Exponents of Deep Neural Networks

L. Storm,[1] H. Linander,[2] J. Bec,[3,4] K. Gustavsson,[1] and B. Mehlig[1]
[1]Department of Physics, University of Gothenburg, 41296 Gothenburg, Sweden
[2]Department of Mathematical Sciences, Chalmers Technical University and University of Gothenburg, Gothenburg, Sweden
[3]MINES Paris, PSL Research University, CNRS, Cemef, Valbonne, F-06900, France
[4]Université Côte d'Azur, Inria, CNRS, Cemef, Valbonne, F-06900, France

(Received 16 June 2023; revised 5 November 2023; accepted 3 January 2024; published 1 February 2024)

We compute how small input perturbations affect the output of deep neural networks, exploring an analogy between deep feed-forward networks and dynamical systems, where the growth or decay of local perturbations is characterized by finite-time Lyapunov exponents. We show that the maximal exponent forms geometrical structures in input space, akin to coherent structures in dynamical systems. Ridges of large positive exponents divide input space into different regions that the network associates with different classes. These ridges visualize the geometry that deep networks construct in input space, shedding light on the fundamental mechanisms underlying their learning capabilities.

DOI: 10.1103/PhysRevLett.132.057301

Deep neural networks can be trained to model complex functional relationships. The expressivity of such neural networks—their ability to unfold intricate data structures—increases exponentially as the number of layers increases [1]. Recent breakthrough applications of neural networks [2] use deep feed-forward layouts with many layers of neurons [3]. These are hard to train due to the multiplicative amplification of signals propagating through the layers, causing signals to either explode or vanish in magnitude if the number of layers is too large. Mathematical analysis in the asymptotic limit of infinitely wide layers reveals how deep networks can nevertheless learn to solve classification tasks [4–7]. Recent results indicate that finite-width networks may learn in different ways [8–10]. It is not understood, however, when and how such networks use their exponential expressivity to represent data features needed for a classification task, how the representation affects prediction accuracy and uncertainty, and how it depends on the network layout.

Here we use dynamical-systems theory to answer these questions. Deep feed-forward networks [Fig. 1(a)] are discrete dynamical systems. Inputs $\boldsymbol{x}^{(0)}$ are mapped iteratively through $x_i^{(\ell)} = g(\sum_{j=1}^{N_\ell} w_{ij}^{(\ell)} x_j^{(\ell-1)} - \theta_i^{(\ell)})$, where $g(\cdot)$ is a nonlinear activation function [11], the layer index $\ell = 0, \ldots, L+1$ plays the role of time, L is the number of hidden layers, N_ℓ is the number of neurons in layer ℓ, and the weights $w_{ij}^{(\ell)}$ and thresholds $\theta_i^{(\ell)}$ are parameters. Sensitivity of $\boldsymbol{x}^{(\ell)}$ to small changes in the inputs $\boldsymbol{x}^{(0)} = \boldsymbol{x}$ corresponds to exponentially growing perturbations in a chaotic system with positive maximal Lyapunov exponent [12,13] $\lim_{\ell\to\infty} \lambda_1^{(\ell)}(\boldsymbol{x})$, with growth rate $\lambda_1^{(\ell)}(\boldsymbol{x}) = \ell^{-1} \log |\delta\boldsymbol{x}^{(\ell)}|/|\delta\boldsymbol{x}|$. The latter is called maximal finite-time Lyapunov exponent (FTLE).

The network weights $w_{ij}^{(\ell)}$ are usually initialized as random numbers, independently Gaussian distributed with zero mean and variance σ_ℓ^2. In this case, the Lyapunov exponents are initially determined by a product of random matrices, and the multiplicative ergodic theorem guarantees that $\lambda_1^{(L)}(\boldsymbol{x})$ converges as $L \to \infty$, to a limit that is independent of $\boldsymbol{x}$ [14]. In the limit $N_\ell = N \to \infty$, one finds $\lambda_1^{(L)} \sim \log(GN\sigma^2)$, explaining why the initial weight variance should be chosen so that $GN\sigma^2 = 1$, because then signals neither contract nor expand [15–17], stabilizing initial stages of the learning (the constant G depends on the choice of activation function [15]).

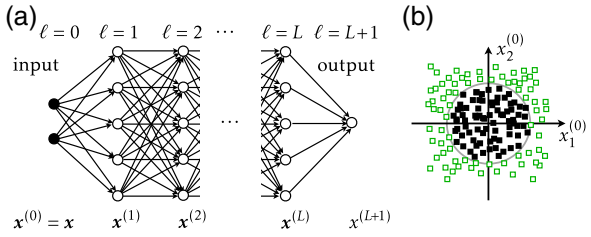

FIG. 1. Classification with a fully connected feed-forward network. (a) Layout with two input components $x_1^{(0)}$ and $x_2^{(0)}$, L hidden layers with five neurons each, and one output $x^{(L+1)}$. (b) Input plane (schematic) for a classification problem with a circular decision boundary that separates input patterns with targets $t = +1$ (empty green square) from those with $t = -1$ (filled black square).

In the limit $N \to \infty$, the hidden weights of deep networks barely change under training [4,18,19]. For finite width, by contrast, the weights tend to change [5]. How the network output changes in response to the changing weights indicates how the network learns.

To understand how the trained network expresses the input-data features needed for classification, we ask how the network output depends on small input changes, encoded in the $\boldsymbol{x}$ dependence of the maximal-FTLE field $\lambda_1(\boldsymbol{x})$ of the trained network. For a classification problem with two-dimensional inputs $\boldsymbol{x}$, divided into two classes with targets $t(\boldsymbol{x}) = \pm 1$ [Fig. 1(b)], we determine how the $\boldsymbol{x}$ dependence of the maximal FTLE changes when changing N and L. For narrow networks, we find that the maximal-FTLE field forms ridges in the input plane, much like Lagrangian coherent structures in dynamical systems [20–22]. These ridges provide insight into the learning process, illustrating how the network learns to change its output by order unity in response to a small shift of the input pattern across the decision boundary. The ridges disappear as the network width grows, suggesting a different learning mechanism. For more complex classification problems (MNIST [23] and CIFAR-10 [24]), we show that FTLE structures in input space explain variations in classification accuracy and predictive uncertainty.

Finite-time Lyapunov exponents.—Figure 1(a) shows a fully connected feed-forward network with L hidden layers, N_0 input components, N neurons per hidden layer, and $N_{L+1} = 1$ output neuron. The network maps every input $\boldsymbol{x}^{(0)} = \boldsymbol{x}$ to an output $x^{(L+1)}$. Weights and thresholds are varied to minimize the output error $[x^{(L+1)} - t(\boldsymbol{x})]^2$, so that the network predicts the correct target $t(\boldsymbol{x})$ for each input $\boldsymbol{x}$. The sensitivity of $\boldsymbol{x}^{(\ell)}$ to small changes $\delta\boldsymbol{x}$ is determined by linearization,

$$\delta\boldsymbol{x}^{(\ell)} = \mathbb{D}^{(\ell)}\mathbb{W}^{(\ell)} \cdots \mathbb{D}^{(2)}\mathbb{W}^{(2)}\mathbb{D}^{(1)}\mathbb{W}^{(1)}\delta\boldsymbol{x} \equiv \mathbb{J}_\ell \delta\boldsymbol{x}. \quad (1)$$

Here, $\mathbb{W}^{(\ell)}$ are the weight matrices with elements $w_{ij}^{(\ell)}$, and $\mathbb{D}^{(\ell)}$ are diagonal matrices with elements $D_{ij}^{(\ell)} = g'(b_i^{(\ell)})\delta_{ij}$, with $b_i^{(\ell)} = \sum_{j=1}^{N_\ell} w_{ij}^{(\ell)} x_j^{(\ell-1)} - \theta_i^{(\ell)}$ and $g'(b_i^{(\ell)}) = (\mathrm{d}/\mathrm{d}b)g(b)|_{b=b_i^{(\ell)}}$. The Jacobian $\mathbb{J}_\ell(\boldsymbol{x})$ characterizes the growth or decay of small perturbations to $\boldsymbol{x}$ [12,13]. Its maximal singular value $\Lambda_1^{(\ell)}(\boldsymbol{x})$ increases or decreases exponentially as a function of ℓ, with rate $\lambda_1^{(\ell)}(\boldsymbol{x}) \equiv \ell^{-1} \log \Lambda_1^{(\ell)}(\boldsymbol{x})$. The singular values $\Lambda_1^{(\ell)}(\boldsymbol{x}) > \Lambda_2^{(\ell)}(\boldsymbol{x}) > \ldots$ are the square roots of the non-negative eigenvalues of the right Cauchy-Green tensor $\mathbb{J}_\ell^\top(\boldsymbol{x})\mathbb{J}_\ell(\boldsymbol{x})$. The maximal eigenvector of $\mathbb{J}_\ell^\top(\boldsymbol{x})\mathbb{J}_\ell(\boldsymbol{x})$ determines the direction of maximal stretching, i.e. in which input direction the output changes the most.

FTLEs and Cauchy-Green tensors are used in solid mechanics to identify elastic deformation patterns [25], and to find regions of instability in plastic deformation [26] and crack initiation [27]. More generally, FTLEs help to characterize the sensitivity of complex dynamics to initial conditions [28–31]. In fluid mechanics, they explain the alignment of particles transported by the fluid [32,33], providing valuable insight into the stretching and rotation of fluid elements over time and space [34]. FTLEs allow us to identify Lagrangian coherent structures [20–22]; fluid-velocity structures that help to organize and understand complex spatiotemporal flow patterns [35]. These geometrical structures appear as ridges of large maximal FTLEs, orthogonal to the maximal stretching direction.

In applying these methods to neural networks, one should recognize several facts. First, in deep neural networks, the weights change from layer to layer. Therefore the corresponding dynamical system is not autonomous. Second, the number N_ℓ of neurons per layer may change as a function of ℓ, corresponding to a changing phase-space dimension. Third, the neural-network weights are trained. This limits the exponential growth of the maximal singular value $\Lambda_1^{(L)}$, and it causes the FTLE $\lambda_1^{(L)}(\boldsymbol{x})$ to remain $\boldsymbol{x}$ dependent, even in the limit of large L (discussed in more detail below). Fourth, one can use different activation functions, such as the piecewise linear ReLU function [11], or the smooth tanh function [15]. Here we use $g(b) = \tanh(b)$, so that the network map is continuously differentiable just like the dynamical systems for which Lagrangian coherent structures were found and analyzed.

Two-dimensional dataset.—To illustrate the geometrical structures formed by the maximal FTLE, we first consider a toy problem. The dataset [Fig. 1(b)] comprises 4×10^4 input patterns, with 90% used for training, the rest for testing. We trained fully connected feed-forward networks on this dataset by stochastic gradient descent, minimizing the output error $[x^{(L+1)} - t(\boldsymbol{x})]^2$. In this way we obtained classification accuracies of at least 98%. We used different layouts, changing the numbers of layers and hidden neurons per layer. The weights were initialized as independent Gaussian random numbers with zero mean and variance $\sigma_\ell^2 \sim N_\ell^{-1}$. The thresholds were initialized to zero (see the Supplemental Material [36] for details). After training, we computed the maximal FTLE in layer L and the associated stretching direction from Eq. (1) as described in Refs. [38,39].

The results are summarized in Fig. 2, which shows maximal-FTLE fields for trained networks with different layouts. We see that the ridges of large positive $\lambda_1^{(L)}(\boldsymbol{x})$ align with the decision boundary between the two classes [Fig. 1(b)]. The network learns by grouping the inputs into two different basins of attraction for $t = \pm 1$, separated by a ridge of positive $\lambda_1^{(L)}(\boldsymbol{x})$. A small shift of the input across the decision boundary leads to a substantial change in the output. In other words, a large maximal FTLE quantifies exponential expressivity of the network near the ridge. This is consistent with the observation that the output is

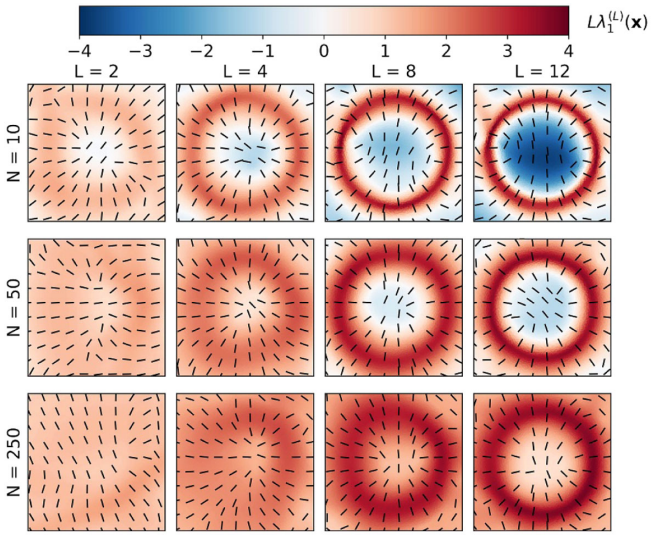

FIG. 2. Geometrical FTLE structures in input space for different widths N and depths L of fully connected feed-forward neural networks trained on the dataset from Fig. 1(b). Shown is the magnitude of $L\lambda_1^{(L)}(\boldsymbol{x})$, and the maximal stretching directions (black lines).

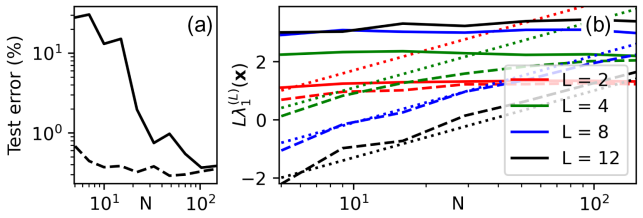

FIG. 3. (a) Classification error for a fully connected feed-forward network with $L = 2$ hidden layers with untrained, random hidden weights, and trained output weights, as a function of N (solid black line). Also shown is the classification error for the fully trained network (dashed line). Both curves were obtained for the dataset shown schematically in Fig. 1. (b) Quantification of the crossover seen in Fig. 2, for fully trained networks. Shown are the averages $\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{d}}$ (solid) and $\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{c}}$ (dashed), see text. The data was obtained by averaging over independent initial-weight realizations. Also shown is a fit to $L\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{c}} \approx -C_{\mathrm{c}}L + \log N$ (dotted).

particularly sensitive to weight changes near decision boundaries [8]. The ridges are most prominent for small N and large L. The contrast between ridge and background increases as L becomes larger, quantifying the exponential expressivity of deep networks [1]. For larger L, the network can resolve smaller input distances $\delta\boldsymbol{x}$ because the singular values increase or decrease exponentially from layer to layer.

The ridges of large maximal FTLE are Lagrangian coherent structures, because the maximal stretching directions (solid lines in Fig. 2) become orthogonal to the ridges for large L. This demonstrates that there is a stringent analogy between the FTLE ridges of deep neural networks and Lagrangian coherent structures.

The ridges gradually disappear as the number N of hidden neurons per layer increases, because the maximal singular value of $\mathbb{J}_L(\boldsymbol{x})$ approaches a definite $\boldsymbol{x}$-independent limit as $N \to \infty$ at fixed L. But how can the network distinguish inputs with different targets in this case, without ridges indicating decision boundaries? One possibility is that the large number of hidden neurons allows the network to embed the inputs into a high-dimensional space where they can be separated thanks to the universal approximation theorem [40]. In this case, training only the output weights (and threshold) suffices, as demonstrated by Fig. 3(a). That the classification error with random hidden weights is larger than that of the fully trained network is not surprising, since different random embeddings have different classification errors when the number of patterns exceeds twice the embedding dimension [11,41]. In summary, large-N networks can classify with random hidden weights. This implies that the hidden weights do not need to change during training, just as in kernel regression with the neural-tangent kernel [4,18]. In other words, the learning in this regime is lazy [19].

Figure 3(b) describes in more detail the crossover between the two learning regimes in Fig. 2. Shown are local averages of the maximal FTLE on the decision boundary, $\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{d}}$, and in the center of the input plane, $\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{c}}$, as functions of N for different values of L. The results were obtained by training for 200 epochs to reach classification accuracies $\geq 99\%$. Although the details of the maximal-FTLE field change upon training further, Fig. 3(b) allows us to draw the following conclusions about the transition. First, $L\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{d}}$ tends to an N-independent constant as L increases, $L\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{d}} \to C_{\mathrm{d}}$. This saturation is due to training: the network learns to produce output differences of the order of $\delta x^{(L+1)} \sim 1$, and to resolve input differences $\delta\boldsymbol{x}$ on the scale of the mean distance $|\delta\boldsymbol{x}^{(0)}|$ between neighboring inputs over the decision boundary. Second, the data shown in Fig. 3(b) suggest that $L\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{c}} \approx -C_{\mathrm{c}}L + \log N$ for large enough L. Third, the contrast between ridges and background disappears upon increasing N, when the background reaches the ridge level, $\langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{c}} \approx \langle\lambda_1^{(L)}(\boldsymbol{x})\rangle_{\mathrm{d}}$. At this point the learning mechanism transitions from learning by ridges to random embedding. In Fig. 3(b) this happens around $N_{\mathrm{c}} \sim \exp(C_{\mathrm{d}} + C_{\mathrm{c}}L)$. While the precise form of the law may depend on the training details, the general conclusion is that N_{c} depends very sensitively on L, because the N dependence of the relation is logarithmic, just as the $N \to \infty$ result for the Lyapunov exponent quoted above. We remark that the L scaling discussed above implies that $\lambda_1^{(L)}(\boldsymbol{x})$ remains $\boldsymbol{x}$ dependent for large L.

One may wonder how the FTLE fields in Fig. 2 depend on the initial weight variance σ^2. For $GN\sigma^2 < 1$, the FTLEs are negative on average, initially. This implies a slowing down of the initial training (vanishing-gradient problem). To see this, consider the fundamental forward-backward dichotomy of deep neural networks [11]: weight updates in the stochastic-gradient algorithm are given by $\delta w_{mn}^{(\ell)} \propto \Delta_m^{(\ell)} x_n^{(\ell-1)}$, where

$$[\boldsymbol{\Delta}^{(\ell)}]^{\mathsf{T}} = [\boldsymbol{\Delta}^{(L)}]^{\mathsf{T}} \mathbb{D}^{(L)} \mathbb{W}^{(L)} \cdots \mathbb{D}^{(\ell+1)} \mathbb{W}^{(\ell+1)} \mathbb{D}^{(\ell)}, \quad (2)$$

and $\Delta_j^{(L)} = g'(b^{(L+1)})[x^{(L+1)} - t(\boldsymbol{x})] g'(b_j^{(L)}) w_j^{(L+1)}$. It follows from Eq. (2) that negative FTLEs cause small weight increments $\delta w_{mn}^{(\ell)}$. Conversely, when the maximal FTLE is positive and too large, the weights grow rapidly, leading to training instabilities.

Remarkably, training has a self-organizing effect. After training, the maximal-FTLE distribution becomes independent of the initial σ^2, provided that the initial maximal FTLE is not too large (Fig. S1 in [36]). For small enough N, in particular, the distribution centers around zero. This is explained by the fact that the network learns by creating maximal-FTLE ridges in input space: in order to accommodate positive and negative $\lambda_1^{(L)}(\boldsymbol{x})$, the distribution must center around zero, alleviating the unstable-gradient problem. We remark that the shape of the distribution, the tails in particular, continues to evolve as one trains further.

MNIST dataset.—This dataset consists of 60 000 images of handwritten digits 0 to 9. Each grayscale image has 28×28 pixels and was pre-processed to facilitate machine learning [23]. Deep neural networks can achieve high precision in classifying this data, with accuracies of up to 99.77% on a test set of 10 000 digits [42].

We determined the maximal-FTLE field for this dataset for a network with $L = 16$ hidden layers, each containing $N = 20$ neurons, and a standard softmax layer with ten outputs [11]. To visualize the geometrical structures in the 28^2-dimensional input space, we projected it onto two dimensions as follows [43]. We added a bottleneck layer with two neurons to the fully trained network, just before the softmax-output layer. We retrained only the weights and thresholds of this additional layer and the output layer, keeping all other hidden neurons unchanged. The local fields b_1 and b_2 of the two bottleneck neurons are the coordinates of the two-dimensional representation shown in Fig. 4(a). We see that the input data separate into ten distinct clusters corresponding to the ten digits. The maximal FTLEs at the center of these clusters are very small or even negative, indicating that the output is not sensitive to small input changes. These regions are delineated by areas with significantly larger positive FTLEs [see 3× enlargement in panel (a)]. Figure 2 leads us to expect that patterns with large $\lambda_1^{(L)}(\boldsymbol{x})$ are located near the decision boundaries in high-dimensional input space. This is verified

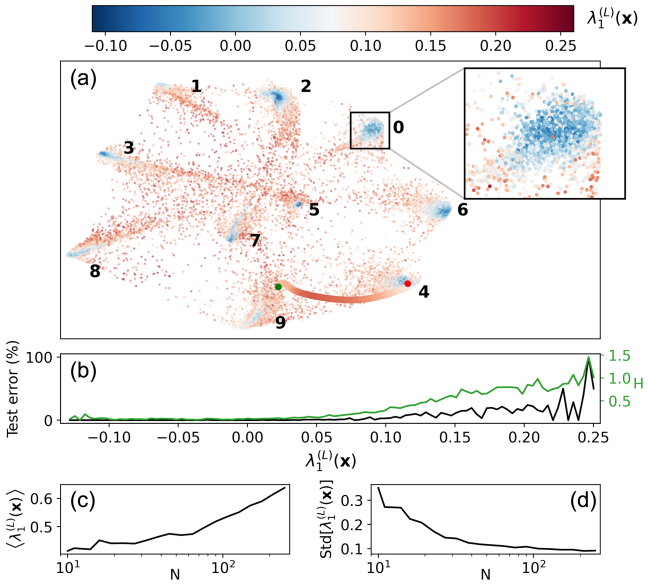

FIG. 4. Maximal-FTLE field for MNIST [23]. A fully connected feed-forward network with $N = 20$ neurons per hidden layer, $L = 16$ hidden layers, and a softmax layer with ten outputs was trained to a classification accuracy of 98.88%. The maximal FTLE was calculated for each of the 28^2-dimensional inputs and projected onto two dimensions (see text). (a) Training data in the nonlinear projection. For each input, the maximal FTLE $\lambda_1^{(L)}$ is shown color coded. The box contains 93% of the recognized digits 0. A 3× zoom of this box is also shown. The colored line represents an adversarial attack from 9 to 4 (see text), with $\lambda_1^{(L)}(\boldsymbol{x})$ color coded. (b) Classification error on the test set and predictive uncertainty H (see text) as functions of $\lambda_1^{(L)}(\boldsymbol{x})$. (c),(d) Mean and standard deviation of maximal-FTLE distribution versus N.

by strong correlations between $\lambda_1^{(L)}(\boldsymbol{x})$ and both the classification error and the predictive uncertainty. Figure 4(b) shows that the classification error on the test set is larger for inputs $\boldsymbol{x}$ with larger $\lambda_1^{(L)}(\boldsymbol{x})$, and that large values of $\lambda_1^{(L)}(\boldsymbol{x})$ correlate with high predictive uncertainty, measured by the entropy H of the posterior predictive distribution [44]. For softmax outputs, where $x_i^{(L+1)}$ can be interpreted as probabilities, $H = -\sum_i \langle x_i^{(L+1)} \rangle \log \langle x_i^{(L+1)} \rangle$, where $\langle\cdot\rangle$ denotes an average over an ensemble of networks with the same layout but different weight initializations [45,46]. These observations confirm that ridges of maximal FTLEs localize the decision boundaries in input space. Similar conclusions hold for other architectures (Fig. S2 in [36]), and for the more complex CIFAR-10 dataset (Fig. S3).

Figure 4(a) also shows $\lambda_1^{(L)}(\boldsymbol{x})$ along a path generated by an adversarial attack. The attack begins from a sample within the cluster corresponding to the digit 9 and aims to transform it into a digit 4 by making small perturbations to the input data [47] toward class 4. We see that the maximal FTLE is small at first, then increases as the path approaches the decision boundary, and eventually decreases again. This

indicates, first, that our conclusions regarding the correlations between large maximal FTLEs and decision boundaries extend to neighborhoods of the MNIST training set that contain patterns the network has not encountered during training. Second, the maximal stretching direction correlates with the direction in input space found by the adversarial attack (Fig. S4 [36]). Third, since it is hard to locate the FTLE ridges in the high-dimensional input space, we characterized the transition between the two learning regimes in terms of the mean and the standard deviation of the FTLE distribution. Figure 4(c) shows that the mean becomes positive and that the variance tends to zero as N grows, leading to the uniform FTLE field characteristic of learning by random embedding.

Conclusions.—For deep neural networks trained on different classification problems, we explored geometrical structures of finite-time Lyapunov exponents in input space. In fluid mechanics, such Lagrangian coherent structures appear as ridges of large exponents, and they are used with great success to organize the phase space of complex spatiotemporal flow patterns. The same is true for deep neural networks: FTLE ridges partition input space into different regions associated with different classes. Our analysis shows how the network exploits its exponential expressivity to form the ridges. Their sharpness determines how quickly classification errors and prediction uncertainty decreases as one moves away from the ridge. As the width of the network increases, the contrast between ridge and background disappears, leading to a different learning mechanism, random embedding, with qualitative differences regarding classification errors and predictive uncertainties. The transition to this lazy-learning regime [4,18,19] occurs for very wide networks, with widths $\sim \exp(L)$. The transition may explain why wider networks are more robust against adversarial attacks [47]: the less important the ridges are for representing the relevant data structures, the harder it is to realize adversarial attacks. The geometrical method presented here may extend to other network architectures, such as resnets [48] or transformers [49], and could help to visualize and understand the mechanisms that allow such neural networks to learn, but this remains a question for the future.

We thank A. Dubey and J. Meibohm for discussions concerning FTLE distributions for neural networks with random weights. L. S. was supported by grants from the Knut and Alice Wallenberg (KAW) Foundation (No. 2019.0079) and Vetenskapsrådet (VR), No. 2021-4452. J. B. received support from UCA-JEDI Future Investments (Grant No. ANR-15-IDEX-01). H. L. was supported by a grant from the KAW Foundation. K. G. was supported by VR Grant No. 2018-03974, and B. M. by VR Grant No. 2021-4452. Some of the numerical computations for this project were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC).

[1] B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli, Exponential expressivity in deep neural networks through transient chaos, Adv. Neural Inf. Process. Syst. **29**, 3360 (2016), https://proceedings.neurips.cc/paper_files/paper/2016/file/148510031349642de5ca0c544f31b2ef-Paper.pdf.

[2] J. Jumper *et al.*, Highly accurate protein structure prediction with AlphaFold, Nature (London) **596**, 583 (2021).

[3] M. Tan and Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in *International Conference on Machine Learning* (PMLR, Cambridge, 2019), pp. 6105–6114.

[4] A. Jacot, F. Gabriel, and C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks, Adv. Neural Inf. Process. Syst. **31**, 8580 (2018), https://proceedings.neurips.cc/paper_files/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

[5] M. Geiger, L. Petrini, and M. Wyart, Landscape and training regimes in deep learning, Phys. Rep. **924**, 1 (2021).

[6] P. M. Nguyen and H. T. Pham, A rigorous framework for the mean field limit of multilayer neural networks, arXiv:2001.11443.

[7] I. Seroussi, G. Naveh, and R. Zohar, Separation of scales and a thermodynamic description of feature learning in some CNNs, Nat. Commun. **14**, 908 (2023).

[8] A. Baratin, T. George, C. Laurent, R. D. Hjelm, G. Lajoie, P. Vincent, and S. Lacoste-Julien, Implicit regularization via neural feature alignment, in *International Conference on Artificial Intelligence and Statistics* (PMLR, Cambridge, 2021), pp. 2269–2277.

[9] J. Paccolat, L. Petrini, M. Geiger, K. Tyloo, and M. Wyart, Geometric compression of invariant manifolds in neural networks, J. Stat. Mech. (2021) 044001.

[10] R. Pacelli, S. Ariosto, M. Pastore, F. Ginelli, M. Gherardi, and P. Rotondo, A statistical mechanics framework for Bayesian deep neural networks beyond the infinite-width limit, Nat. Mach. Intell. **5**, 1497 (2023).

[11] B. Mehlig, *Machine Learning with Neural Networks: An Introduction for Scientists and Engineers* (Cambridge University Press, Cambridge, England, 2021).

[12] E. Ott, *Chaos in Dynamical Systems*, 2nd ed. (Cambridge University Press, Cambridge, England, 2002).

[13] P. Cvitanović, R. Artuso, R. Mainieri, G. Tanner, and G. Vattay, *Chaos: Classical and Quantum* (Niels Bohr Institute, Copenhagen, 2020).

[14] A. Crisanti, A. Vulpiani, and G. Paladin, *Products of Random Matrices in Statistical Physics* (Springer, Berlin, 1993).

[15] J. Pennington, S. Schoenholz, and S. Ganguli, Resurrecting the sigmoid in deep learning through dynamical isometry: Theory and practice, Adv. Neural Inf. Process. Syst. **30**, 4785 (2017), https://proceedings.neurips.cc/paper_files/paper/2017/file/d9fc0cdb67638d50f411432d0d41d0ba-Paper.pdf.

[16] I. Sutskever, J. Martens, G. Dahl, and G. E. Hinton, On the importance of initialization and momentum in deep learning, in *Proceedings of the 30th International Conference on Machine Learning—Volume 28* (PMLR, Cambridge, 2013), pp. III–1139–III–1147.

[17] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *Proceedings of the Thirteenth International Conference on Artificial*

Intelligence and Statistics, Proceedings of Machine Learning Research Vol. 9, edited by Y. W. Teh and M. Titterington (JMLR Workshop and Conference Proceedings, Chia Laguna Resort, Sardinia, Italy, 2010), pp. 249–256.

[18] S. Chen, H. He, and W. Su, Label-aware neural tangent kernel: Toward better generalization and local elasticity, Adv. Neural Inf. Process. Syst. **33**, 15847 (2020), https://proceedings.neurips.cc/paper_files/paper/2020/file/b6b90237b3ebd1e462a5d11dbc5c4dae-Paper.pdf.

[19] L. Chizat, E. Oyallon, and F. Bach, On lazy training in differentiable programming, Adv. Neural Inf. Process. Syst. **32**, 2933 (2019), https://proceedings.neurips.cc/paper_files/paper/2019/file/ae614c557843b1df326cb29c57225459-Paper.pdf.

[20] G. Haller and G. Yuan, Lagrangian coherent structures and mixing in two-dimensional turbulence, Physica (Amsterdam) **147D**, 352 (2000).

[21] V. Lucarini and T. Bódai, Edge states in the climate system: Exploring global instabilities and critical transitions, Nonlinearity **30**, R32 (2017).

[22] M Beneitez, Y Duguet, P Schlatter, and D. S. Henningson, Edge manifold as a Lagrangian coherent structure in a high-dimensional state space, Phys. Rev. Res. **2**, 033258 (2020).

[23] Y. LeCun, C. Cortes, and C. J. C. Burges, The MNIST database of handwritten digits, yann.lecun.com/exdb/mnist (2018).

[24] A. Krizhevsky, G. Hinton *et al.*, Learning multiple layers of features from tiny images, Technical report, Computer Science University of Toronto, Canada, (2009).

[25] C. Truesdell and W. Noll, in *The Non-Linear Field Theories of Mechanics*, Encyclopedia of Physics Vol. III, edited by S. Flügge (Springer, Berlin, 1965), pp. 1–579.

[26] J. L. Ren, C. Chen, Z. Y. Liu, R. Li, and G. Wang, Plastic dynamics transition between chaotic and self-organized critical states in a glassy metal via a multifractal intermediate, Phys. Rev. B **86**, 134303 (2012).

[27] X. Jin, Y. Chen, L. Wang, H. Han, and P. Chen, Failure prediction, monitoring and diagnosis methods for slewing bearings of large-scale wind turbine: A review, Measurement **172**, 108855 (2021).

[28] G. Haller, Lagrangian coherent structures, Annu. Rev. Fluid Mech. **47**, 137 (2015).

[29] S. Vannitsem, Predictability of large-scale atmospheric motions: Lyapunov exponents and error dynamics, Chaos **27**, 032101 (2017).

[30] A. Uthamacumaran, A review of dynamical systems approaches for the detection of chaotic attractors in cancer networks, Patterns **2**, 100226 (2021).

[31] A. Morozov, K. Abbott, K. Cuddington, T. Francis, G. Gellner, A. Hastings, Y.-C. Lai, S. Petrovskii, K. Scranton, and M. L. Zeeman, Long transients in ecology: Theory and applications, Phys. Life Rev. **32**, 1 (2020).

[32] R. Ni, N. T. Ouellette, and G. A. Voth, Alignment of vorticity and rods with Lagrangian fluid stretching in turbulence, J. Fluid Mech. **743**, R3 (2014).

[33] V. Bezuglyy, M. Wilkinson, and B. Mehlig, Poincare indices of rheoscopic visualisations, Europhys. Lett. **89**, 34003 (2009).

[34] P. L. Johnson, S. S. Hamilton, R. Burns, and C. Meneveau, Analysis of geometrical and statistical features of Lagrangian stretching in turbulent channel flow using a database task-parallel particle tracking algorithm, Phys. Rev. Fluids **2**, 014605 (2017).

[35] J. F. Gibson, J. Halcrow, and P. Cvitanovic, Visualizing the geometry of state space in plane Couette flow, J. Fluid Mech. **611**, 107 (2008).

[36] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.132.057301 for detailed information regarding the experimental setup and the theoretical models. It also contains Ref. [37].

[37] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, Automatic differentiation in PyTorch, Technical report, (2017), https:// openreview.net/forum?id=BJJsrmfCZ.

[38] J. P. Eckmann and D. Ruelle, Ergodic theory of chaos and strange attractors, Rev. Mod. Phys. **57**, 617 (1985).

[39] T. Okushima, New method for computing finite-time Lyapunov exponents, Phys. Rev. Lett. **91**, 254101 (2003).

[40] P. Kidger and T. Lyons, Universal approximation with deep narrow networks, arXiv:1905.08539.

[41] T. M. Cover, Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition, IEEE Trans. Electron. Comput. **EC-14**, 326 (1965).

[42] D. Ciregan, U. Meier, and J. Schmidhuber, Multi-column deep neural networks for image classification, in *2012 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE Computer Society, Silver Spring, 2012), pp. 3642–3649.

[43] Y. Wang, H. Yao, and S. Zhao, Auto-encoder based dimensionality reduction, Neurocomputing **184**, 232 (2016).

[44] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher *et al.*, A survey of uncertainty in deep neural networks, arXiv:2107.03342.

[45] B. Lakshminarayanan, A. Pritzel, and C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, Adv. Neural Inf. Process. Syst. **30**, 6402 (2017), https://proceedings.neurips.cc/paper_files/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf.

[46] L. Hoffmann and C. Elster, Deep ensembles from a Bayesian perspective, arXiv:2105.13283.

[47] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, Towards deep learning models resistant to adversarial attacks, in *International Conference on Learning Representations* (International Conference on Learning Representations, Vancouver, 2018).

[48] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.

[49] A. Dosovitskiy, L. Beyer, A Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, An image is worth 16×16 words: Transformers for image recognition at scale, arXiv:2010.11929.