Lyapunov exponent analysis for multilayer neural networks

Misaki Kondo\textsuperscript{1a)}, Satoshi Sunada\textsuperscript{2b)}, and Tomoaki Niiyama\textsuperscript{2}

\textsuperscript{1} Graduate School of Natural Science and Technology, Kanazawa University, Kakuma, Kanazawa, Ishikawa, 920-1192, Japan
\textsuperscript{2} Faculty of Mechanical Engineering, Institute of Science and Engineering, Kanazawa University, Kakuma-machi Kanazawa, Ishikawa 920-1192, Japan

\textsuperscript{a)} konmisaki@stu.kanazawa-u.ac.jp
\textsuperscript{b)} sunada@se.kanazawa-u.ac.jp

Received April 8, 2021; Revised June 21, 2021; Published October 1, 2021

Abstract: A multilayer neural network can be regarded as a type of discrete-time dynamical system in the sense that layer-to-layer information propagation can be expressed as the time evolution of a particular dynamical system. In this study, we investigate the stability of information propagation in multilayer neural networks for classification problems using finite-time maximum Lyapunov exponents, and we discuss how multilayer neural networks classify inputs. The dynamical stability analysis in this study reveals the input-dependent stability of trained multilayer neural networks. Multilayer neural networks are trained such that the information propagation is highly sensitive to input data vectors near a decision boundary for classification whereas it is less sensitive to input data vectors far from the decision boundary. This implies that the decision boundary in classification problems is characterized by a set where the finite-time maximum Lyapunov exponents of the information propagation are relatively large. These results offer a new perspective on the estimation of uncertainty of classification using multilayer neural networks.

Key Words: neural network, lyapunov exponent, dynamical system

1. Introduction

Artificial neural networks, which consist of multiple layered nodes to transmit and process signal, are emulation of neurotransmission in the human brain and are the foundation of deep learning known as a subset of machine learning \cite{1}. Nowadays, neural networks have become groundbreaking tools for data processing owing to their high-level performance. This has prompted a paradigm shift to many cross-disciplinary areas. Despite their great successes, full understanding on the mechanism of the information processing in neural networks, particularly deep neural networks with multilayered structures, remains a challenge. Recent studies suggested that a neural network can be regarded as a type of dynamical system \cite{2–4}. Based on this, an input data vector (e.g., an image) can be regarded as an initial state, and the layer-to-layer information propagation in a multilayer network system can be modeled as the time evolution of a dynamical system, as shown in Fig. 1. The link between

Nonlinear Theory and Its Applications, IEICE, vol. 12, no. 4, pp. 674–684 ©IEICE 2021 DOI: 10.1587/nolta.12.674

674
two different fields, neural networks and dynamical system theory [6, 7], offers a new perspective on understanding how multilayer neural networks processes information by introducing measures developed in the context of dynamical system theory [8].

In this study, we characterize the classification by multilayer neural networks using Lyapunov exponents, which have commonly been used as a measure of dynamical stability in context of dynamical system theory. The Lyapunov exponent analysis reveals that the “dynamical stability” for a well-trained multilayer neural network strongly depends on the inputs. It is shown that a decision boundary for the classification, which is determined by the trained network system, can be characterized by the distribution of large Lyapunov exponents. This provides us an alternative approach for estimating the uncertainty of the classification using multilayer neural networks.

Fig. 1. Schematic of multilayer neural network. From the viewpoint of dynamical system theory, a multilayer neural network can be regarded as a discrete time dynamical system because the layer-to-layer information propagation corresponds to the time evolution of the system. In this sense, an input data vector can be regarded as an initial state.

2. Lyapunov exponents of multilayer neural networks

In this section, we consider a multilayer neural network with \( n \) layers, as shown in Fig. 1. Let a dataset to be learned be composed of \( K \) inputs, \( x_{0,k} \), and their corresponding target vectors, \( t_k \), where \( k \in \{1, 2, \ldots, K\} \). The goal of supervised learning using a multilayer network is to find a function that transforms the inputs into their corresponding targets. Thus, the layer-to-layer information propagation in a multilayer neural network needs to be trained such that transformation to the appropriate targets is facilitated.

In a multilayer neural network, each layer works as a mapping, which, for example, consists of a linear matrix operation with a weight vector and a nonlinear transformation [due to an activation function, for example, hyperbolic tangent (tanh) or rectified linear unit (ReLU)] for the input. The mapping can simply be expressed as \( x_{j+1} = F_j(x_j, w_j) \), where \( x_j \) and \( w_j \) (\( j \in \{0, 1, 2, \ldots, n\} \)) denote the input vector and weight vector in the \( j \)th layer, respectively. The last layer (the \( n \)th layer) is a classifier that classifies an input as belonging to a certain target class, where the output \( x_n \) generated by the classifier represents the probability distribution that the input belongs to one of target classes. It is worth noting that the multilayer neural networks with such structures can be considered as discrete-time nonlinear dynamical systems [2–5].

In this study, we investigate how a multilayer neural network after training classifies each input into an appropriate class. For this, we introduce a finite-time version of maximum Lyapunov exponents for discrete-time dynamical systems. The traditional Lyapunov exponent is a measure of dynamical stability, which characterizes the rate of separation between two trajectories starting from slightly different initial states. A maximum positive Lyapunov exponent, \( \lambda > 0 \), indicates that the system is chaotic, whereas \( \lambda < 0 \) indicates that the system is dynamically stable. In this study, the Lyapunov exponent is used to characterize how two arbitrary inputs separate according to the layer-to-layer propagation in the multilayer network system.
For traditional dynamical systems, the Lyapunov exponents are computed using approaches shown in [9, 10] for a long period of time, ensuring that the maximum increasing direction does not depend on the initial direction. However, the number of the layers in multilayer network system may not be substantially large to validate such an assumption. In addition, the dimension of nonlinear mapping $F_j$ may be different in each layer. Therefore, we roughly estimated the maximum Lyapunov exponent, more precisely speaking, finite-time maximum Lyapunov exponent (FTMLE), for trained multilayer neural networks in a similar manner with [8] as follows:

1. Set input $x_{0,k}$ as an initial state.
2. Compute the output $x_{n,k}$ generated from input $x_{0,k}$.
3. Randomly generate an initial perturbation $\delta x_{0,k}(l) (l \in \{1, 2, \cdots, L\})$ and compute the deviation of output $\delta x_{n,k}(l)$ for each $\delta x_{0,k}(l)$.
4. Compute the FTMLE $\lambda_{n,k}$.

In the fourth step, we first estimated the FTMLE obtained from initial deviation $\delta x_{0,k}(l)$ as follows:

$$\lambda_{n,k}(l) \approx \frac{1}{n} \log \frac{\left| \delta x_{n,k}(l) \right|}{\left| \delta x_{0,k}(l) \right|} = \frac{1}{n} \sum_{j=0}^{n-1} \log \frac{\left| \delta x_{j,k}(l) \right|}{\left| \delta x_{j-1,k}(l) \right|},$$  \hspace{1cm} (1)

where $\delta x_{j,k}(l)$ is given by the following equation:

$$\delta x_{j} = \frac{\partial F_{j-1}}{\partial x_{j-1}} \delta x_{j-1}. \hspace{1cm} (2)$$

Subsequently, we chose the maximum value from the set of $\{\lambda_{n,k}, \lambda_{n,k}, \cdots, \lambda_{n,k}(L)\}$,

$$\lambda_{n,k} = \max_{l} \lambda_{n,k}(l). \hspace{1cm} (3)$$

In the computation of the FTMLE, we used the automatic differentiation package provided by the Python package PyTorch (torch.autograd), which is useful for computing $\partial F_j/\partial x_j$ for various transformations, for example, a convolutional neural network.

3. Results

In this section, we compute the FTMLE for different input data using Eq. (3) in trained multilayer neural networks and show the relationship between the dynamical stability and decision boundary in classification problems.

3.1 Two-dimensional classification problems

We here introduce simple two-dimensional classification problems. The goal is to classify two-dimensional (2D) input vectors $x_{k,0} = (x_k, y_k) (k \in \{1, 2, \cdots, K\})$ into several classes. Figures 2(a)–2(d) show the datasets used in this study for binary classifications. For each dataset, two types of data points (indicated by blue and red colors) are randomly distributed in one-dimensional (1D) donut, 2D donut, spiral, and square forms. Figures 3(a)–3(d) show other datasets for multiclass classifications, where three or four types of data points (indicated by blue, red, green, and yellow colors) are randomly distributed in 1D donut, 2D donut, spiral, and square forms, respectively.

Fig. 2. (a) 1D donut dataset, (b) 2D donut dataset, (c) spiral dataset, and (d) square dataset for binary classifications.
Fig. 3. (a) 1D donut dataset, (b) 2D donut dataset, (c) spiral dataset, and (d) square dataset for multiclass classifications.

For the binary classification problems, a multilayer neural-network model, which consists of three fully-connected hidden layers having 10 neurons with tanh activation functions at $j$th layer ($1 < j < n$) and softmax classifier with two outputs at the last layer ($j = n$), was trained using a 1000-point dataset ($K = 1000$). For the multiclass classification problems, a model, which consists of four fully-connected hidden layers having 10 neurons with tanh activation functions for $1 < j < n$ and softmax classifier at the last layer, was trained with a 2000-point dataset ($K = 2000$). For the optimizations of the models, we used the stochastic gradient descent (SGD) optimizer with learning rate 0.05 and momentum 0.9.

The results of the classifications by those models after the training over epoch 100 are displayed in Figs. 4(a), 4(c), 4(e) and 4(g) for binary classifications and in Figs. 5(a), 5(c), 5(e) and 5(g) for multiclass classifications. The interfaces between red-colored area and blue-colored area in these figures represent the decision boundaries obtained by the learning, which separate data points belonging to different classes. Subsequently, we computed the FTMLEs using Eq. (3) for the trained networks. Figures 4(b), 4(d), 4(f) and 4(h) show the distributions of the FTMLEs in each 2D space for binary classification problems, whereas Fig. 5(b), 5(d), 5(f) and 5(h) show the distributions of the FTMLEs in each 2D space for multiclass classification problems. As can be observed in these figures, relatively large FTMLEs concentrate near decision boundaries, that is, information propagation starting from a data point near the decision boundaries exhibits a large FTMLE.

Fig. 4. (a, c, e, g) Results of binary classifications for (a) a 1D donut dataset, (c) 2D donut dataset, (e) spiral dataset, and (g) square dataset. In each dataset, most of data points colored as blue and red were successfully classified into each class labeled by “0” and “1,” shown in blue-colored area and red-colored area, respectively. The decision boundaries can be observed as interfaces between the blue-colored and red-colored areas. (b, d, f, h) Distributions of FTMLEs for (b) 1D donut dataset, (d) 2D donut dataset, (f) spiral dataset, and (h) square dataset. Large FTMLEs concentrate near the decision boundaries.

The above-mentioned results clearly show that the trained network has a separation property for the data points near the decision boundary; that is, any two input data points across the boundary are classified into different classes regardless of the closeness of these points. We also confirmed the similar feature in various networks with different activation functions and different layers in binary and multiclass classification problems. See Appendix A for details.
3.2 MNIST handwritten digit recognition

To investigate the FTMLEs for more large-scale neural networks, we use the MNIST handwritten digit recognition dataset, which has commonly been used as a standard benchmark for machine learning [11, 12]. The dataset is 28 × 28 pixel gray-scale images of handwritten digits including a training set of 60,000 images and test set of 10,000 images. We used a model consisting of three layered convolutional neural networks and the softmax classifier with 10 outputs at \( n = 4 \). In the model, the first convolutional layer uses eight kernels of size 3 × 3 and the ReLU activation function, followed by max pooling of size 2 × 2. The second (third) convolutional layer uses 16 (32) kernels of size 2 × 2 and the ReLU activation function, followed by max pooling of size 2 × 2. Subsequently, a fully connected matrix of size 128 × 10 is used as the input for the softmax classification. After 50-epoch training by the SGD optimizer with learning rate 0.01 and momentum 0.9, the classification accuracy was 97%. For this image classification, it is impossible to visualize the decision boundary owing to its high dimensionality; therefore, we employed the t-distributed stochastic neighbor embedding (t-SNE) [13] to visualize the features in a low dimensional space. Figure 6(a) shows a part of the MNIST data embedded in a 2D output data space, where the softmax output \( x_n = \{x_n^{(m)}\}_{m=0}^{9} \) (10 dimensional vector) is embedded. The outputs are clearly grouped as some slender clusters by each class.

We also computed the FTMLEs for each input image in the MNIST recognition problem and plotted them in the same embedded 2D output data space, as shown in Fig. 6(b). One can see that relatively large FTMLEs localize near one end of each cluster. Similar characteristics were confirmed for the Fashion-MNIST dataset [14], as shown in Appendix A.

4. Discussions

4.1 FTMLE distribution in an output data space

As shown in Subsec. 3.2, we confirmed that the outputs from the large-scale neural networks for MNIST classification problem formed slender clusters in a 2D output data space, and the FTMLEs were relatively high near one end of each cluster [Fig. 6(b)]. This feature of the FTMLE distribution can be understood from the perspective of the decision boundaries as follows: considering that the outputs form slender clusters in an output data space, only one end of a group cluster has contact with one end of another group cluster. As an example, we show a cluster formation in an output data space, \( \{x_n^{(m)}\}_{m=0}^{2} \), for a three-class classification problem in Fig. 7(a), where each cluster can be visualized without any information loss. The contact interfaces correspond to the decision boundaries for the classification problems, and large FTMLEs are distributed near the contact interfaces, which
are the decision boundaries. This can be observed in the comparison of Figs. 7(a) and 7(b), where the three line-like clusters are colored blue, red, and yellow. In other words, the localization of large FTMLEs near the end of each cluster suggests that large FTMLEs are distributed near the decision boundaries. This relationship between the decision boundaries and FTMLEs in the relatively low-dimensional dataset implies that the concentration of large FTMLEs shown in Fig. 6(b) ensures a decision boundary is present even in the high-dimensional output space.

The above interpretation is also supported by the consideration of entropy of the (softmax) output from the networks. Evidently, from the definition of the softmax classifier, the softmax output $x_n = \{x_n^{(m)}\}_{m=0}^{9}$ represents the probability distribution that an input image belongs to a certain class $m$. For example, if $x_n^{(p)} \geq x_n^{(m \neq p)}$ for an input, the input is classified into class $p$. If the output is far from a decision boundary in an output data space and belongs to a class $p$, $x_n^{(p)} \approx 1 \gg x_n^{(m \neq p)}$. However, if the output is near a decision boundary between classes $p$ and $q$, $x_n^{(p)} \approx x_n^{(q)} \gg x_n^{(m \neq p, q)}$. Thus, relatively high entropy of the probability distribution, $S = -\sum_{m=0}^{9} x_n^{(m)} \log x_n^{(m)}$, obtained from an output $x_n$ indicates that the output is near one of the decision boundaries of the network.

Figure 8 shows the correlation map between the FTMLEs and logarithm of the entropy for the MNIST classification problem. The figure clearly reveals that the FTMLE is closely related to the entropy. This correlation affirms that the FTMLEs increase around decision boundaries even in the large-scale
Fig. 7. Example of the classification result for a three-class classification problem. (a) The data points to be classified into three classes are displayed in an output data space \( \{x_n^{(m)}\}_{m=0}^2 \), where \( x_n^{(m)} \) is the softmax output representing the probability distribution that an input belongs to a certain class \( m \). Each labeled data point forms a cluster in the space, and the decision boundaries are given as a set of the points in which the clusters have contact with each other. (b) FTMLE distribution. Large FTMLEs distribute in decision boundaries shown in (a).

Fig. 8. Correlation map between FTMLEs and the logarithm of entropy \( S \) for the MNIST dataset.

neural networks for a high-dimensional data classification problem.

4.2 Inference of classification accuracy from the FTMLE

The above-mentioned relationship between FTMLEs and decision boundaries suggests that the value of an FTMLE may have a high correlation with the classification accuracy as well as the decision boundaries. To clarify this, we investigated the relationship between the FTMLE and classification accuracy. In Figs. 9(a)–(d), each classification accuracy is shown as a function of FTMLE for a 1D donut dataset for binary and three-class classifications, MNIST hand-written digit dataset, and Fashion-MNIST dataset. The figures clearly show that each classification accuracy decreases with relatively large FTMLE. These results suggest that the FTMLE analysis can be used to infer the classification accuracy and detect decision boundaries in the input data space.

4.3 Information propagation

The analysis presented in this study can be applied to investigate how information is processed in hidden layers in terms of the dynamical stability. We computed the magnification of the deviation \( \delta x_{j,k} \) at the \( j \)th layer for \( \delta x_{j-1,k} \) at the \((j-1)\)th layer for the \( k \)th input information, that is, \( |\delta x_{j,k}|/|\delta x_{j-1,k}| \). Magnification \( |\delta x_{j,k}|/|\delta x_{j-1,k}| \) represents the stability of the information propagation from the \((j-1)\)th layer to the \( j \)th layer for the \( k \)th input. For display purpose, we computed the average of \( |\delta x_{j,k}|/|\delta x_{j-1,k}| \) for a group of trajectories with similar FTMLEs of \( \lambda_{n,k_g} \pm \Delta \lambda_n \ (k_g \in \{1, 2, \ldots, K_g\}) \); that is, \( \gamma_{j,k_g} = \langle |\delta x_{j,k}|/|\delta x_{j-1,k}| \rangle_{k_g} \), where \( k_g \) is the group number and \( \langle \cdot \rangle_{k_g} \) denotes the average for the \( k_g \)th group.

Figure 10 shows \( \gamma_{j,k_g} \) computed from various models for a binary classification problem, a three-class
Fig. 9. Classification accuracy as a function of the FTMLE for 1D donut datasets for (a) binary classification (b) three-class classification, (c) MNIST handwritten digit dataset, and (d) Fashion-MNIST dataset.

Fig. 10. The magnification $\gamma_{j,k}$ for (a, c) binary and (b) three-class classifications with a spiral dataset and (d) the MNIST handwritten digit problem. The network models used in (a) and (b) are similar to that shown in Sec. 3.1. The models used in (c) and (d) are the ResNet with $n = 10$ (See Appendix) and that shown in Sec. 3.2, respectively.
classification problem, and the MNIST handwritten digit problem. $\gamma_{j,k} > 1$ denotes a separation between two neighboring trajectories (signals) at the $j$th layer, whereas $\gamma_{j,k} < 1$ denotes the approach of two signals at the $j$th layer. Each $\gamma_{j,k}$ is colored by the magnitude of the FTMLE $\lambda_{n,k}$, that is, the distance from the decision boundary. As seen in these figures, the stability of the information propagation drastically changes in the hidden layers in a complex way, depending on the distance from the decision boundary. For the binary and three-class classification problems, as shown in Fig. 10, $\gamma_{j,k}$ can be larger than unity in the hidden layers for inputs near the decision boundary. We can identify the hidden layer where a large separation occurs, which plays a role in nonlinear classifications for the datasets. For the MNIST problem, $\gamma_{j,k} < 1$ is kept even in the hidden layers. This indicates that the separation property does not play a role in this classification problem. As such, our dynamical stability analysis could be useful for gaining an insight into the role of the hidden layers in classifications.

5. Conclusion

In this study, we investigated how multilayer neural networks classify inputs based on FTMLE analysis. Our analysis revealed that multilayer neural networks are trained such that information propagation starting from an input near the decision boundary exhibits a relatively large FTMLE. In other words, information propagation in the multilayer networks is trained to facilitate the separation of two inputs belonging to different classes near a decision boundary. An input far from the boundary exhibits a negative FTMLE; that is, information propagation of the input is stable for perturbations in order to facilitate classifications into the same class. Therefore, the input-dependent dynamical stability plays a crucial role in the classification. Moreover, we showed that the difficulty of classification can be estimated using the FTMLE analysis; that is, the FTMLE can be used as a measure to detect the decision boundary.

Acknowledgments

This work was supported, in part, by JSPS KAKENHI (Grant No. 20H04255) and JST PRESTO (Grant No. JPMJPR19M4).

Appendix

A. FTMLE distributions

We investigated FTMLE distributions for multilayer neural network models with different activation functions and different layers. Figures A-1(a)–A-1(h) show the classification results and corresponding FTMLE distributions for a square dataset, respectively, where we used neural network models containing 3 hidden fully-connected layers with the ReLU activation function, residual network (ResNet) models [15] containing 5 and 10 hidden fully-connected layers with tanh activation function, and neural ordinary differential equation model [3] with tanh activation function corresponding to $n = 50$. We used 10 neurons at each layer in all models used in this study. For the training of the models, the SGD optimizer with a learning rate of 0.05 and momentum of 0.9 was used. As mentioned in Sec. 3.1, large FTMLEs distribute near each decision boundary.

Figures A-1(i) and A-1(j) show the Fashion-MNIST image data embedded by t-SNE in a 2D output data space and FTMLE distribution, respectively. The model used in this task consists of convolutional neural networks. The first (second) convolutional layer uses 16 (32) kernels of size $3 \times 3$ and the ReLU activation function, followed by max pooling of size $2 \times 2$. The third convolutional layer uses 64 kernels of size $2 \times 2$ and the ReLU activation function, followed by max pooling of size $2 \times 2$. A fully-connected matrix of size $256 \times 32$ with tanh activation function is subsequently used. Finally, fully-connected matrix of size $32 \times 10$ is used for the input to the softmax classification. The model was trained using the SGD optimizer with a learning rate of 0.05 and momentum of 0.9. As seen in Figs. A-1(i) and A-1(j), relatively large FTMLEs localize near a part of the end of each cluster.
Fig. A-1. (a–h) Results of classification and FTMLE distributions for a square dataset. We used (a–b) neural network models with three layers and the ReLU activation function, ResNet models with (c–d) five layers and (e–f) ten layers with tanh activation functions, and (g–h) a NODE model with 50 layers. (i) Classification results and (j) FTMLE distribution in a 2D output data space for the Fashion-MNIST dataset.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature, vol. 521, p. 436, 2015.
[2] G.-H. Liu and E.A. Theodorou, “Deep Learning Theory Review: An Optimal Control and Dynamical Systems Perspective,” arXiv, 1908.10920v2, 2019.
[3] T.Q. Chen, Y. Rubanova, J. Bettencourt, and D.K. Duvenaud, “Neural ordinary differential equations,” Adv. Neural Inf. Process. Syst., vol. 31, pp. 6572–6583, 2018.
[4] E. Haber and L. Ruthotto, “Stable architectures for deep neural networks,” Inverse Problems, vol. 34, p. 014004, 2017.
[5] M. Benning, E. Celledoni, M. J Ehrhardt, B. Owren, and C.-B. Schönlieb, “Deep learning as optimal control problems: models and numerical methods,” arXiv:1904.05657, 2019.
[6] V.I. Arnold, Mathematical Methods of Classical Mechanics, Springer, New York, 2013.
[7] H. Goldstein, Classical Mechanics (2nd ed.), Addison-Wesley, Boston, 1980.
[8] H. Li, “Analysis on the Nonlinear Dynamics of Deep Neural Networks: Topological Entropy and Chaos,” arXiv:1804.03987, 2018.
[9] A.J. Lichtenberg and M.A. Lieberman, “Regular and Chaotic Dynamics (Applied Mathematical Sciences, 38),” Springer, New York, 1992.
[10] J.R. Dorfman, “An Introduction to Chaos in Nonequilibrium Statistical Mechanics (Cambridge Lecture Notes in Physics),” Cambridge University Press, Cambridge, 1999.
[11] Y. LeCun, C. Cortes, and C.J.C. Burges, The MNIST database of handwritten digits, http://yann.lecun.com/exdb/mnist
[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, p. 2278, 1998.
[13] L. van der Maaten and G. Hinton, “Visualizing Data using t-SNE,” Journal of Machine Learning Research, vol. 9, pp. 2579–2605, 2008.
[14] https://github.com/zalandoresearch/fashion-mnist

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.