Letter

Generalization of stochastic-resonance-based threshold networks with Tikhonov regularization

Saiya Bai,[1] Fabing Duan,[1,*] François Chapeau-Blondeau,[2,†] and Derek Abbott[3,‡]

[1]*Institute of Complexity Science, College of Automation, Qingdao University, Qingdao 266071, People's Republic of China*
[2]*Laboratoire Angevin de Recherche en Ingénierie des Systèmes, Université d'Angers, 62 Avenue Notre Dame du Lac, 49000 Angers, France*
[3]*Centre for Biomedical Engineering and School of Electrical and Electronic Engineering, University of Adelaide, Adelaide, South Australia 5005, Australia*

(Received 23 January 2022; revised 2 May 2022; accepted 17 June 2022; published 6 July 2022)

Injecting artificial noise into a feedforward threshold neural network allows it to become trainable by gradient-based methods and also enlarges the parameter space as well as the range of synaptic weights. This configuration constitutes a stochastic-resonance-based threshold neural network, where the noise level can adaptively converge to a nonzero optimal value for finding a local minimum of the loss criterion. We prove theoretically that the injected noise plays the role of a generalized Tikhonov regularizer for training the designed threshold network. Experiments on regression and classification problems demonstrate that the generalization of the stochastic-resonance-based threshold network is improved by the injection of noise. The feasibility of injecting noise into the threshold neural network opens up the potential for adaptive stochastic resonance in machine learning.

DOI: 10.1103/PhysRevE.106.L012101

Injecting noise into input data [1], weights, or a desired signal [2–6] has been substantially studied in artificial neural networks for improved *network generalization*, i.e., the ability to fit real data outside the initial training set [7]. Using a rigorous expansion of infinitesimal parameters, Bishop [1] proved that noise injection is equivalent to a smoothing regularization term that behaves as a generalized Tikhonov regularizer. However, the regularization term in the loss function contributed by the injected noise variance is effective only in the case of an infinite training set [2,3,8]. In terms of practical training, this is intractable in view of the associated time consumed. Moreover, a range of noise intensities has been manually trialed to generalize the network performance for regression and classification problems [1,2,4,9], from which an acceptable noise variance was finally determined as a hyperparameter for the neural network. The optimum injected noise variance is still unknown and cannot be adaptively learned as a model parameter of the network. This has severely limited the application of noise injection in artificial neural networks.

Recently, the study of noise injection in neural networks shifted its focus to the activation function of the hidden layer [6,9–15]. In particular, adding noise only to the activation function in its hard-saturated regimes allows the stochastic gradient-descent method to be more widely applicable in training neural networks with nondifferentiable functions [10,12–14]. The advantage of noise in the hidden layer can be explained theoretically by the case of minimizing the convex loss function of a feedforward neural network, which estimates a regression function via a nonlinear mapping

$$f(\boldsymbol{x},\boldsymbol{\theta})=\boldsymbol{\alpha}+\boldsymbol{\beta}\psi(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}), \tag{1}$$

where $\boldsymbol{x}\in\mathbb{R}^{N\times 1}$ is the input vector, $\boldsymbol{W}\in\mathbb{R}^{K\times N}$ and $\boldsymbol{\beta}\in\mathbb{R}^{M\times K}$ are weight matrices, $\boldsymbol{\alpha}\in\mathbb{R}^{M\times 1}$ and $\boldsymbol{b}\in\mathbb{R}^{K\times 1}$ denote bias vectors, the network parameter set $\boldsymbol{\theta}=\{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{W},\boldsymbol{b}\}$, and $\psi(x)$ is the activation function of hidden neurons. Let $\{\boldsymbol{x}(\ell),\boldsymbol{y}(\ell)\}_{\ell=1}^{L}$ denote L examples of the data set for training the network in a supervised learning manner, and the observation $\boldsymbol{y}\in\mathbb{R}^{M\times 1}$ is sampled from the regression function in the presence of observational error or noise. Then, by minimizing the empirical loss function

$$\mathcal{L}=\frac{1}{L}\sum_{\ell=1}^{L}\|f(\boldsymbol{x}(\ell),\boldsymbol{\theta})-\boldsymbol{y}(\ell)\|^2 \tag{2}$$

with respect to the parameter set $\boldsymbol{\theta}$, we can approximate the network output as the desired regression function.

Now, for noise injection, one implementation is to perturb the input $\boldsymbol{x}$ by introducing the artificial noise $\boldsymbol{\xi}\in\mathbb{R}^{N\times 1}$ [1–4]. Here the zero-mean injected noise $\boldsymbol{\xi}$ has the common probability density function $f_\xi(x)$ and the same variance σ_ξ^2. For the convex Euclidean 2-norm $\|\boldsymbol{x}\|$ and using the Jensen inequality, the expectation of the loss $\mathcal{L}$ satisfies

$$\begin{aligned}\mathbb{E}_\xi(\mathcal{L})&=\mathbb{E}_\xi\left[\frac{1}{L}\sum_{\ell=1}^{L}\|f(\boldsymbol{x}(\ell)+\boldsymbol{\xi},\boldsymbol{\theta})-\boldsymbol{y}(\ell)\|^2\right]\\&\geqslant\frac{1}{L}\sum_{\ell=1}^{L}\|\mathbb{E}_\xi[f(\boldsymbol{x}(\ell)+\boldsymbol{\xi},\boldsymbol{\theta})]-\boldsymbol{y}(\ell)\|^2\\&=\frac{1}{L}\sum_{\ell=1}^{L}\|\tilde{f}(\boldsymbol{x}(\ell),\boldsymbol{\xi},\boldsymbol{\theta})-\boldsymbol{y}(\ell)\|^2=\mathcal{L}_\xi,\end{aligned} \tag{3}$$

where the empirical loss $\mathcal{L}_\xi$ is achieved by a different neural network architecture indicated by the mapping

$$\tilde{f}(\boldsymbol{x},\boldsymbol{\xi},\boldsymbol{\theta})=\boldsymbol{\alpha}+\boldsymbol{\beta}\tilde{\psi}[\boldsymbol{W}(\boldsymbol{x}+\boldsymbol{\xi})+\boldsymbol{b}], \tag{4}$$

[*]fabingduan@qdu.edu.cn
[†]f.chapeau@univ-angers.fr
[‡]derek.abbott@adelaide.edu.au

with the transformed hidden neuron $\tilde{\psi}(u) = \mathbb{E}_\xi[\psi(u+\xi)] = \int \psi(u+\xi) f_\xi(\xi) d\xi$.

From Eq. (3), the contrast of the designed network architecture in Eq. (4) lies in (i) the smaller empirical loss achieved by injecting noise $\boldsymbol{\xi}$ into the hidden neurons $\tilde{\psi}(x)$ than the input data $\boldsymbol{x}$, (ii) allowing backpropagation training of neural networks with a family of nondifferentiable activation functions, and (iii) the practicability of adaptively finding the optimum noise variance σ_ξ^2 in the training phase. For instance, for the McCulloch-Pitts neuron [16]

$$\psi(u) = \begin{cases} 1, & u \geqslant 0 \\ 0, & u < 0 \end{cases} \tag{5}$$

yielding a binary output [12–14], the conventional backpropagation learning method is infeasible due to the zero gradient of $\psi(u)$ for $u \neq 0$ and the nondifferentiability at $u = 0$. However, with the help of the injected noise $\boldsymbol{\xi}$, the designed threshold network in Eq. (4) becomes trainable via the backpropagation learning method, because $\tilde{\psi}(x) = \int_{-x}^{\infty} f_\xi(\xi) d\xi$ has a proper nonzero gradient [13,14].

Next, an interesting question is that whether the injected noise $\boldsymbol{\xi}$ in the hidden layer still acts as a smoothing regularization for generalizing the performance of the stochastic-resonance-based threshold network or not? The answer is positive.

Assuming the zero-mean noise $\boldsymbol{\xi}$ has a small variance σ_ξ^2, the empirical loss $\mathcal{L}_\xi$ in Eq. (3) can be expanded in powers of the noise vector $\boldsymbol{\eta} = \boldsymbol{W}\boldsymbol{\xi}$ as

$$\begin{aligned}
\mathcal{L}_\xi &= \frac{1}{L}\sum_{\ell=1}^{L} \|\boldsymbol{\alpha} + \boldsymbol{\beta}\mathbb{E}_\xi[\psi(\boldsymbol{W}\boldsymbol{x}(\ell) + \boldsymbol{W}\boldsymbol{\xi} + \boldsymbol{b})] - \boldsymbol{y}(\ell)\|^2 \\
&\approx \frac{1}{L}\sum_{\ell=1}^{L} \left\| \boldsymbol{\alpha} + \boldsymbol{\beta}\mathbb{E}_\xi \left[\psi(\boldsymbol{W}\boldsymbol{x}(\ell) + \boldsymbol{b}) + \boldsymbol{J}_\psi^\top \boldsymbol{\eta} \right.\right. \\
&\quad \left.\left. + \frac{1}{2}\begin{pmatrix} \boldsymbol{\eta}^\top \boldsymbol{H}_{\psi_1} \boldsymbol{\eta} \\ \boldsymbol{\eta}^\top \boldsymbol{H}_{\psi_2} \boldsymbol{\eta} \\ \vdots \\ \boldsymbol{\eta}^\top \boldsymbol{H}_{\psi_K} \boldsymbol{\eta} \end{pmatrix} + \mathcal{O}(\boldsymbol{\eta}) \right] - \boldsymbol{y}(\ell) \right\|^2 \\
&\approx \frac{1}{L}\sum_{\ell=1}^{L} \left\| f(\boldsymbol{x}(\ell), \boldsymbol{\theta}) - \boldsymbol{y}(\ell) + \frac{\sigma_\xi^2}{2} \boldsymbol{\beta} \overbrace{\begin{pmatrix} \mathrm{tr}(\boldsymbol{H}_{\psi_1} \boldsymbol{W}\boldsymbol{W}^\top) \\ \mathrm{tr}(\boldsymbol{H}_{\psi_2} \boldsymbol{W}\boldsymbol{W}^\top) \\ \vdots \\ \mathrm{tr}(\boldsymbol{H}_{\psi_K} \boldsymbol{W}\boldsymbol{W}^\top) \end{pmatrix}}^{\boldsymbol{\lambda}} \right\|^2 \\
&\approx \mathcal{L} + \frac{\sigma_\xi^2}{L}\sum_{\ell=1}^{L} [\boldsymbol{y}(\ell) - f(\boldsymbol{x}(\ell), \boldsymbol{\theta})]^\top \boldsymbol{\lambda} + \mathcal{O}(\sigma_\xi^2),
\end{aligned} \tag{6}$$

where $\boldsymbol{J}_\psi$ is the Jacobi matrix of $\psi(\boldsymbol{W}\boldsymbol{x}(\ell) + \boldsymbol{b}) \in \mathbb{R}^{K \times 1}$, $\boldsymbol{H}_{\psi_k}$ is the Hessian matrix of $\psi_k = \psi([\boldsymbol{W}]^{(k)} \boldsymbol{x}(\ell) + b_k)$ for $k = 1, 2, \ldots, K$, $[\boldsymbol{W}]^{(k)}$ is the kth row of $\boldsymbol{W}$, $\mathbb{E}_\xi(\boldsymbol{\eta}^\top \boldsymbol{H}_{\psi_k} \boldsymbol{\eta}) = \mathrm{tr}[\boldsymbol{H}_{\psi_k} \mathbb{E}_\xi(\boldsymbol{\eta}\boldsymbol{\eta}^\top)] = \sigma_\xi^2 \mathrm{tr}(\boldsymbol{H}_{\psi_k} \boldsymbol{W}\boldsymbol{W}^\top)$, and the operator $\mathrm{tr}(\cdot)$ denotes the matrix trace. It is clearly seen from the second term in Eq. (6) that the injected noise in the hidden layer contributes a regularization term to the original empirical loss $\mathcal{L}$ of Eq. (2). In addition, it is seen in Eqs. (4)–(6) that only one hyperparameter of the injected noise level σ_ξ is introduced; thus the designing complexity of the stochastic-resonance-based neural network does not increase much.

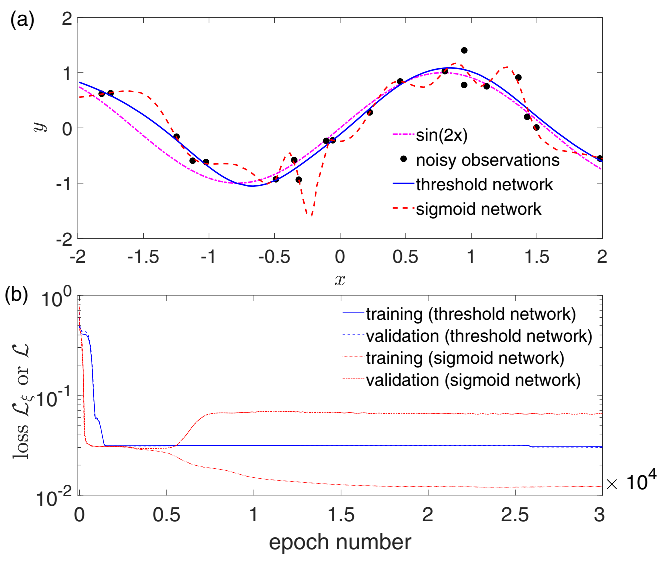

FIG. 1. (a) Outputs of the designed threshold network and the sigmoid network. For comparison, the noisy observations and the target function $\sin(2x)$ are also illustrated. (b) Learning curves of the loss versus the epoch number on training and validation data sets.

Next we will demonstrate experimentally the generalization of the designed threshold network in Eq. (4) for function approximation and classification problems. First, consider fitting a unidimensional target function $\sin(2x)$ on observations $y = \sin(2x) + \zeta$ from Gaussian background noise ζ. Here ζ is with zero mean and variance $\sigma_\zeta^2 = 0.25^2$. Also, $x(\ell)$ are randomly distributed with uniform distribution in the interval $[-2, 2]$, and an illustrative sample of the training set $\{x(\ell), y(\ell)\}_{\ell=1}^{L=20}$ (circles) is shown in Fig. 1(a). The injected noise $\boldsymbol{\xi}$ in Eq. (1) is assumed to be Gaussian distributed with a common level σ_ξ, and the designed threshold network has $K = 21$ hidden neurons of $\tilde{\psi}(x)$. The output (solid line) of the trained threshold network after 3×10^4 epochs is presented in Fig. 1(a). For comparison, Fig. 1(a) also plots the output (dashed line) of the sigmoid network with $K = 21$ activation functions $(1 + e^{-u})^{-1}$ in the hidden layer, which passes through or overfits almost training observations. The corresponding training curve of the loss $\mathcal{L}_\xi$ in Eq. (3) is illustrated in Fig. 1(b). Moreover, at every interval of 100 epochs, cross validation is carried out to validate the generalization of the designed threshold network on the validation set, which also contains another sample of 20 noisy observations. It is clearly seen from the validation curve of $\mathcal{L}_\xi$ in Fig. 1(b) that the error on the validation set in the sense of $\mathcal{L}_\xi$ also monotonically decreases to 0.031 during the training process. This is because the regularization term in Eq. (6) can restrict the magnitude of the weights and control the effective complexity of the

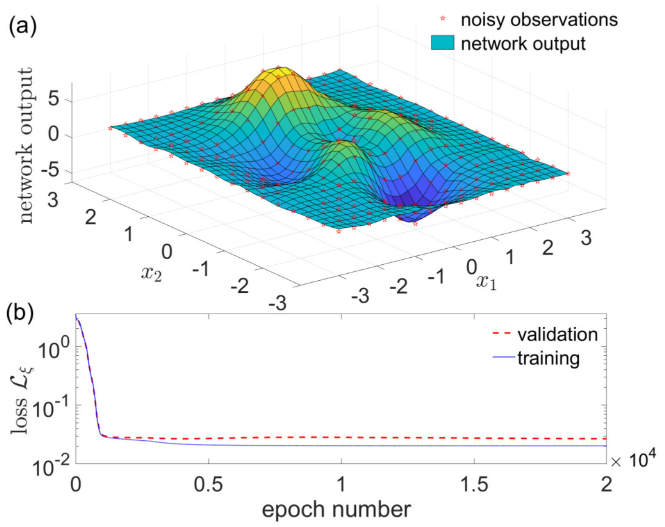

FIG. 2. (a) Output of the trained threshold neural network as the approximation (patched surface) to the two-dimensional function in Eq. (7). The validation data (⋆) are also plotted. (b) Learning curves of the loss $\mathcal{L}_\xi$ on training and validation sets.

designed threshold network with a smoother response to noisy observations. For comparison, the validation loss $\mathcal{L}$ of the sigmoid network rises from 0.0296 to 0.067 as the epoch number increases, as shown in Fig. 1(b).

Furthermore, we also validate the designed threshold neural network for approximating a two-dimensional function

$$z(x_1,x_2)=3(1-x_1)^2e^{-x_1^2-(x_2+1)^2}-\tfrac{1}{3}e^{-(x_1+1)^2-x_2^2}$$
$$-10\left(\tfrac{1}{5}x_1-x_1^3-x_2^5\right)e^{-x_1^2-x_2^2}. \quad (7)$$

Here the 16×16 training set contains the data $\boldsymbol{x}(\ell)=[x_1(\ell),x_2(\ell)]^\top$ that are uniformly spaced in the range $[-3,3]\times[-3,3]$ and the noisy observations $y(\ell)=z[x_1(\ell),x_2(\ell)]+\zeta$. Here the zero-mean Gaussian noise ζ is with variance $\sigma_\zeta^2=0.2^2$. The 32×32 validation set is also chosen in this way to validate the designed threshold network with the size $N\times K\times M=2\times30\times1$. The approximation (patched surface) of the trained threshold network and the validation set (stars) are both shown in Fig. 2(a). The corresponding training (solid line) and validation (dashed line) curves of the loss $\mathcal{L}_\xi$ in Eq. (3) are illustrated in Fig. 2(b). It is seen in Fig. 2(b) that the error on the validation set still does not go up for 2×10^4 training epochs, and the designed threshold network also generalizes well to approximate the two-dimensional function of Eq. (7).

Specifically, Fig. 3 shows the learning curves of the injected noise variance σ_ξ^2 in the designed threshold network for function approximation. The injected noise manifests its beneficial role in the training process as the noise variance converges on a nonzero (local) optimum value 0.813 for the unidimensional function or 0.446 for the two-dimensional function. Since the injected noise variance σ_ξ^2 is adaptively searched by the gradient-descent method, this prominent characteristic of the adaptive stochastic resonance effect [15,17–21] then greatly extends the practicability of the threshold neural network.

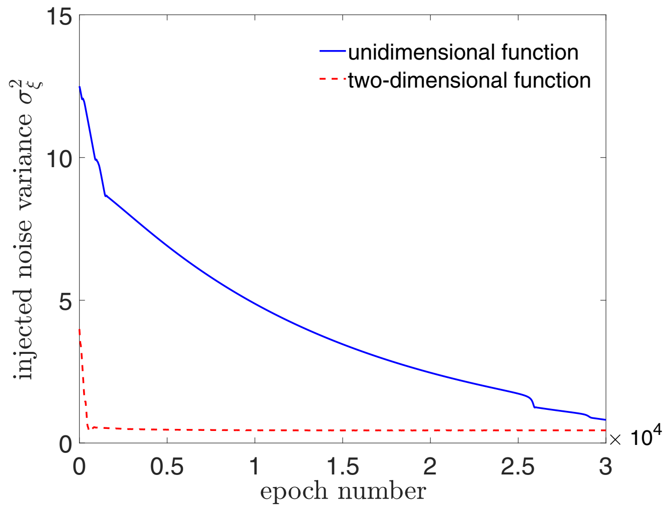

FIG. 3. Learning curves of the noise level σ_ξ in the stochastic-resonance-based threshold network for approximating unidimensional and two-dimensional functions. The other parameters are the same as in Figs. 1 and 2.

We further demonstrate the generalization of the designed threshold network for multiclassification problems. Here the empirical loss is chosen as the cross entropy

$$\mathcal{L}_\xi=\frac{1}{L}\sum_{\ell=1}^{L}\{-\boldsymbol{y}(\ell)^\top\ln f(\boldsymbol{x}(\ell),\boldsymbol{\theta})+[\mathbf{1}-\boldsymbol{y}(\ell)]^\top$$
$$\times\ln[1-f(\boldsymbol{x}(\ell),\boldsymbol{\theta})]\}. \quad (8)$$

First, consider a classical data classification problem [22], as shown in Fig. 4. There is a pair of regions A and B facing each other in an asymmetrically arranged manner, which represents two data patterns. The data points are described by the coordinates x_1 and x_2 that form the input vector $\boldsymbol{x}$ of the neural network with the size $2\times20\times1$. The vertical distance $d=-7$ separates two regions with respect to the x_2 axis and the smaller d means the larger the area in which the two regions

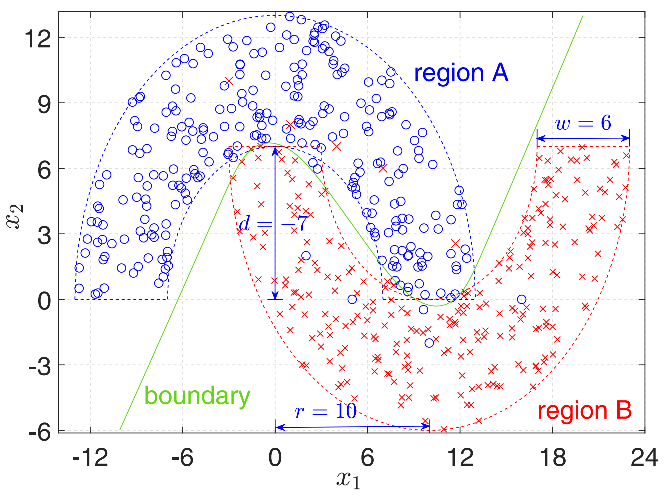

FIG. 4. Decision boundary given by the designed threshold network trained by 200 pairs of data points for vertical distance $d=-7$ between regions A and B. Each region is with radius $r=10$ and width $w=6$.

overlap [22]. The training set consists of 200 pairs of data points, where five pairs are mislabeled. Figure 4 presents the training result of the decision boundary given by the trained threshold network. For this difficult nonlinear separability problem, the classification accuracy of the trained threshold network is 97.8% for 2000 pairs of testing points, which is comparable to the classification error of 98.2% achieved by the radial basis function network with the common width 4.1 assigned to 20 Gaussian units [22].

We design a $4 \times 6 \times 3$ threshold network to classify three kinds of flowers on the Iris data set consisting of 150 samples [23]. A 4:1 ratio is selected for training and testing, and the mislabeled data take in $\frac{1}{9}$ of the training set. After 100 training epochs, the classification accuracy of the trained threshold network is 97.4% for the test set, which is comparable to the accuracy of 97.6% obtained by the elegant learning method of support vector machines [24]. For comparison, the sigmoid network with the activation function $1/(1+e^{-5u})$ only achieves the classification accuracy 61% on the same testing set, which is usually viewed as a substitute for the low-precision neural network with binary neurons [10,12–14].

The stochastic-resonance-based threshold network is also applied to recognize handwritten digits in the MNIST database. Here 10^4 images with the training set and testing set in a 4:1 ratio are employed. Each gray handwritten image has 28×28 pixels and so can be mapped into a 784×1 input vector $\boldsymbol{x}$ for the threshold neural network with the size of $784 \times 100 \times 10$. The categories of the digits are expressed by the target vector set $\boldsymbol{d} = \{d_i\}$ for $i = 0, 1, \ldots, 9$. For instance, the vector $d_1 = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]^\top$ represents the digit 1 and so on. The classification accuracy up to 97.0% on the testing set is obtained [13]. Furthermore, the 2000 testing images are attacked by the spatial Gaussian noise with variance 0.1, the salt and pepper noise with density 0.2, and the speckle multiplicative noise with variance 0.3 and the cropping of the 7×7 subpart of the image and the trained stochastic-resonance-based threshold network attains the classification accuracies of 74.60%, 80.70%, 95.0%, and 62.25%, respectively. For comparison, the sigmoid network with the same size $784 \times 100 \times 10$ only achieves the accuracies of 31.75%, 47.05%, 91.45%, and 56.25%. The generalization of the designed threshold network is significantly more efficient.

In this paper, we proved theoretically that the injected noise in the hidden layer plays the role of the generalized Tikhonov regularizer for the loss function of the neural network. Experiments on function approximation and classification problems showed that the generalization performance of the designed threshold network can be improved by the injection of noise at the optimal nonzero noise level obtained by the gradient-based learning rule. Compared with the conventional method of substituting the sigmoid function for the threshold neuron, the generalization performance of the designed threshold neural network is greatly satisfactory, even comparable to some elegant machine learning methods.

However, we must emphasize that, in the testing phase, the realization of the hidden neuron $\tilde{\psi}(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{\eta}+\boldsymbol{b}) \approx \frac{1}{T}\sum_{t=1}^{T}\psi(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{\eta}_t+\boldsymbol{b})$ in Eq. (4) needs to be asymptotically implemented by injecting a sufficiently large number T of mutually independent noise $\boldsymbol{\eta}_t$ into threshold neurons [12–14]. For instance, the number T takes the value 400 in the realization of the designed threshold neural network for recognizing handwritten digits. From this point of view, the heavy computations of testing designed threshold networks are inconvenient for more complex problems in practice [6,10,14]. We build a stochastic-resonance-based threshold convolutional neural network with one convolutional layer, one max-pooling layer, and two fully connected layers, wherein the rectified linear unit (ReLU) activation functions are replaced by the transformed neurons $\tilde{\psi}$ in Eq. (4). It is interesting to note that the implementation of the designed threshold convolutional neural network with $T = 20$ threshold neurons can achieve an accuracy of 96.75% for testing 10^4 images in the MNIST data set and 70.39% top-1 accuracy for testing 10^4 images in the CIFAR-10 data set, which are comparable to the accuracies achieved by the full precision convolutional neural network with ReLU activation functions. In such a case, the number T is greatly reduced, and the generalization capability of the deeper threshold convolutional neural network is worth further study.

This work was supported by the Natural Science Foundation of Shandong Province of China (Grant No. ZR2021MF051) and the Australian Research Council (Grant No. DP200103795).

[1] C. M. Bishop, Training with noise is equivalent to Tikhonov regularization, Neural Comput. **7**, 108 (1995).

[2] G. An, The effects of adding noise during backpropagation training on a generalization performance, Neural Comput. **8**, 643 (1996).

[3] Y. Grandvalet and S. Canu, Noise injection: Theoretical prospects, Neural Comput. **9**, 1093 (1997).

[4] A. K. Seghouane, Y. Moudden, and G. Fleury, Regularizing the effect of input noise injection in feedforward neural networks training, Neural Comput. Appl. **13**, 248 (2004).

[5] H. Noh, T. You, J. Mun, and B. Han, Regularizing deep neural networks by noise: its interpretation and optimization, in *Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, 2017*, edited by I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett (Curran Associates Inc., Red Hook, NY, 2017), pp. 5115–5124.

[6] Z. He, A. S. Rakin, and D. Fan, *IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, 2019* (IEEE, Piscataway, 2019), pp. 588–597.

[7] C. M. Bishop, *Neural Networks for Pattern Recognition* (Oxford University Press, London, 1996), Chap. 9, pp. 346–349.

[8] M. Burger and A. Neubauer, Analysis of Tikhonov regularization for function approximation by neural networks, Neural Netw. **16**, 79 (2003).

[9] N. C. Hammadi and H. Ito, Improving the performance of feedforward neural networks by noise injection into hidden neurons, J. Intell. Robot. Syst. **21**, 103 (1998).

[10] C. Gulcehre, M. Moczulski, M. Denil, and Y. Bingio, Noisy activation functions, in *Proceedings of the 33rd International*

Conference on International Conference on Machine Learning, edited by M. F. Balcan and K. Q. Weinberger (JMLR.org, New York, NY, 2016), pp. 3059–3068.

[11] B. Kosko, K. Audhkhasi, and O. Osoba, Noise can speed backpropagation learning and deep bidirectional pretraining, Neural Netw. **129**, 359 (2020).

[12] S. Ikemoto, F. Dallalibera, and K. Hosoda, Noise-modulated neural networks as an application of stochastic resonance, Neurocomputing **277**, 29 (2018).

[13] X. Liu, L. Duan, F. Duan, F. Chapeau-Blondeau, and D. Abbott, Enhancing threshold neural network via suprathreshold stochastic resonance for pattern classification, Phys. Lett. A **403**, 127387 (2021).

[14] L. Duan, F. Duan, F. Chapeau-Blondeau, and D. Abbott, Noise-boosted backpropagation learning of feedforward threshold neural networks for function approximation, IEEE Trans. Instrum. Meas. **70**, 1010612 (2021).

[15] Z. Liao, Z. Wang, H. Yamahara, and H. Tabata, Echo state network activation function based on bistable stochastic resonance, Chaos Soliton. Fract. **153**, 111503 (2021).

[16] W. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity, Bull. Math. Biophys. **5**, 115 (1943).

[17] S. Mitaim and B. Kosko, Adaptive stochastic resonance, Proc. IEEE **86**, 2152 (1998).

[18] J. J. Collins, C. C. Chow, and T. T. Imhoff, Stochastic resonance without tuning, Nature (London) **376**, 236 (1995).

[19] A. R. Bulsara and L. Gammaitoni, Tuning in to noise, Phys. Today **49**(3), 39 (1996).

[20] M. D. McDonnell, N. G. Stocks, C. E. M. Pearce, and D. Abbott, *Stochastic Resonance: From Suprathreshold Stochastic Resonance to Stochastic Signal Quantization* (Cambridge University Press, New York, 2008).

[21] Y. Kang, R. Liu, and X. Mao, Aperiodic stochastic resonance in neural information processing with Gaussian colored noise, Cogn. Neurodyn. **15**, 517 (2021).

[22] S. Haykin, *Neural Networks and Learning Machines* (Prentice Hall, New York, 2009).

[23] D. Dua and C. Graff, UCI machine learning repository, http://archive.ics.uci.edu/ml (2017).

[24] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning* (Springer, New York, 2008).