# Compact modeling of metal Oxide TFTs based on bayesian search-based artificial neural network and genetic algorithm

A. HongMei Xie,[1] B. Author,[1] and C. Author[2]
[1)] *School of Automation, Guangdong University of Technology, Guangzhou, China*
[2)] *School of Automation, Guangdong University of Technology, Guangzhou, China*

(*Electronic mail: Second.Author@institution.edu.)

(Dated: 17 October 2022)

The thin-film transistors ( TFTs ) plays an important role in electronic information industry. The traditional physical model requires a lot of time and resources, and the generalization ability of the model is not strong. This paper presents a method of artificial neural network ( ANN ) model to quickly incorporate the characteristics of emerging devices for circuit simulation. The Multi-layer Perceptron (MLP)model is a typical ANN model with a simple structure and high modeling efficiency. Then,the pivotal steps of MLP is to determine its topology. For this hyperparameter problem can be easily solved by Bayesian search. In order to improve the accuracy of the model, a genetic algorithm is proposed to optimize the initial values of the weight and biase. Furthermore, by introducing a first derivative in the loss function, small signal parameters are considered. This modeling approach greatly reduces the time required for compact modeling of devices. Through experiments, the output of the model is in good agreement with the experimental data, which fully verifies the effectiveness of the proposed model. After extracting the trained model parameters, it is implemented with Verilog-A. This circuit-level experiment demonstrates that this hybrid ANN model can accurately estimate various physical devices and circuits.

## I. INTRODUCTION

Flexible metal oxide semiconductor thin-film transistors(TFTs) have attracted considerable attention over the past few years due to their excellent device properties, such as high transparency[1], low manufacturing cost, and temperature[2]. In recent years, due to its extremely low leakage current and high uniformity[3], the circuit can have a high level of integration, making it a great possibility in realizing integrated circuits.

Due to rapid changes in semi-conductor technology, developing a new model of transistor characteristics has become an essential part of a continuous design cycle. Therefore, it is necessary to design an accurate and reliable prediction model. Developing high-quality, physically-based compact models[4,5] is time-consuming. Therefore it is generally not suitable for emerging devices. To rapidly model emerging devices, a data-oriented modeling approach is used for technology evaluation without in-depth detailed device physics. Artificial neural networks are considered powerful tools for modeling and optimization problems[6], and their universal approximation properties[7,8] enable them to learn arbitrary nonlinear relationships from data. Therefore, it has a history of serving as compact models for semiconductor devices[9,10], with a particular success in radio frequency (RF) device applications.[11,12]

In this work, an automatic parameter tuning algorithm, Bayesian optimization[13,14] is adopted to determine the structure of the ANN model. Compared with manual experiments to determine the structure, it greatly reduces the time and computing power required. In addition, in order to improve the model accuracy, the genetic algorithm is integrated into the neural network. A hybrid neural network model is designed and explained the network modeling through the visual method. Finally, implementing this model in the circuit can achieve the effect of circuit simulation.

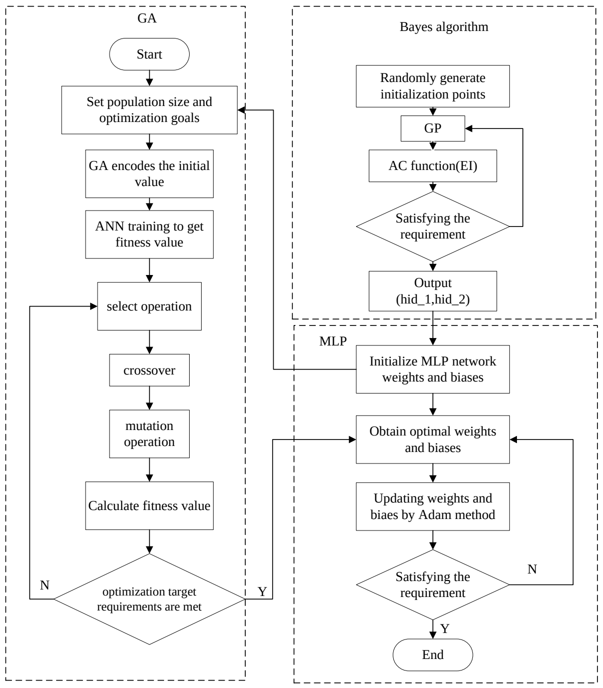
## II. ANN MODEL FRAMEWORK

### A. Model Building

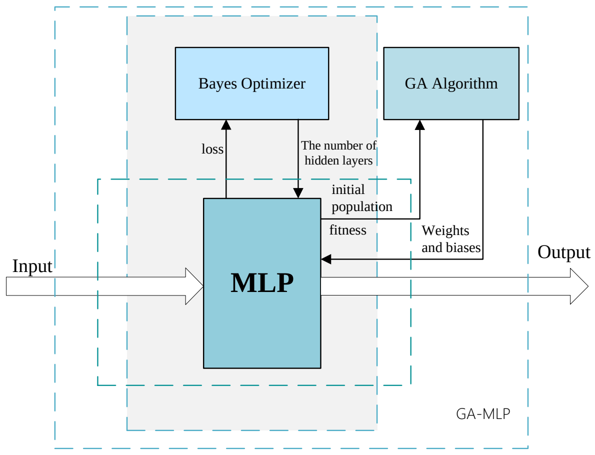
The model is divided into three parts, the flow chart and structure diagram is shown in the Fig.1. In the whole model, the part of MLP is used to fit the complex nonlinear relationship between TFTs input and output.[15] In order to give the network enough degrees of freedom to represent this complex nonlinear relationship, an appropriate model size must be chosen. However, the general approximation theorem does not dictate what the size of the neural network should be.

In the actual modeling of the network, it usually takes many attempts to determine the optimal topology, which invisibly reduces the modeling speed of the network. The number of neurons is called hyperparameter. There are many ways to adjust hyperparameters, such as grid search, random search, and various evolutionary computing algorithms[14] (such as genetic algorithm, particle swarm optimization, etc). However, the above methods are not suitable for this parameter optimization. Because hyperparameters evaluate the performance of a combination of parameters. The network formed by these parameters can only evaluate the performance of the network after training. In general, training a neural network requires a lot of computation and takes a long time. Therefore, the number of parameter combinations to be evaluated cannot be too many. Take these considerations, the most suitable method is Bayesian optimization. Bayesian optimization, a model-based method for finding the minimum of a function has recently been applied to machine learning hyperparameter tuning. The results show that the method can achieve better performance on the test set while requiring fewer iterations than random search.

Traditional optimization algorithms are generally based on gradient information, but this method is generally used


GA
Start
Set population size and optimization goals
GA encodes the initial value
ANN training to get fitness value
select operation
crossover
mutation operation
Calculate fitness value
optimization target requirements are met
N
Y
Bayes algorithm
Randomly generate initialization points
GP
AC function(EI)
Satisfying the requirement
Output (hid_1,hid_2)
MLP
Initialize MLP network weights and biases
Obtain optimal weights and biases
Updating weights and biaes by Adam method
Satisfying the requirement
N
Y
End

(a)Flowchart of the entire model



(b)model structure diagram

FIG. 1. Flowchart of the entire model and the model structure diagram

when constraints and objective functions can be derived. This causes it is easy to fall into the cycle of local optimal solutions[16]. Genetic algorithms[17]target all individuals in a population and use randomization techniques to guide an efficient search of an encoded parameter space. It uses a single individual to represent the optimal solution of the problem to be solved, and initializes a large number of different individuals to form a population. Compared with the traditional optimization algorithm, the genetic algorithm is an algorithm that searches for the optimal individual on the basis of the group, which has the characteristics of strong robustness, high search efficiency, and not easy to fall into local optimum. Therefore, combining it with MLP can make better use of neural networks to solve problems.

## B. Multi-Layer Perceptron

Artificial neural network,which originated from the information processing research of biological nerve, is a kind of parallel processing and high-performance mathematical model. The modeling process of neural network can be regarded as a process of function approximation. The network selected for this modeling is MLP(multi-layer perceptron). It has a simple network structure and constructs a global approximation of nonlinear input-output mapping. In the MLP, the first and last layers are called input and output layers, and the layers in between are called hidden layers. The input neuron receives the external excitation, its output becomes the excitation of the first hidden layer neuron, and so on. The output neuron receives the excitation from the last hidden layer neuron and completes the external output. Kosmogorov's theorem states that with a reasonable structure and proper weights, a three-layer feedforward network can approximate any continuous function. Too many hidden layers will not only increase the training time but also lead to overfitting[15]. The output to the node l in the hidden layer is given by:

$$net_l = g_l\left(\sum_{j=1}^{n} x_j w_{jk}^l + b_k^l\right) \qquad k = 1,2\cdots s \tag{1}$$

Where n and s are the number of neurons in the $L-1$ th and $L$th layers respectively, $b_k^l$is the $L$th bias term of kth node, $w_{ik}^l$is the weight factor, $g_l$is the activation function of the Lth hidden layer.

Common activation functions are Relu, Tanh, and Sigmoid. Due to circuit simulation requirements for first and higher order derivatives of transistors, the sigmoid type with smooth derivatives are preferred. Considering the simplicity of implementation on hardware circuits, the Tanh activation function is used. The final output is defined as follows:

$$\hat{I}_{ds} = g_2\left(g_1\left(x_{in} w_{ij}^1 + b_j^1\right) w_{jk}^2 + b_k^2\right) w_{kl}^3 + b_l^3 \tag{2}$$

In this work, the loss function takes the MSE. Considering the detection of small signal parameters, the loss function is defined as follows:

$$\text{loss } = \tfrac{1}{n}\left(\sum_{i=1}^{n}\left(I_{ds}-\hat{I}_{ds}\right)^2 + \sum_{i=1}^{n}\left(\text{grad}_1 - g_d\right)^2 + \sum_{i=1}^{n}\left(\text{grad}_2 - g_m\right)^2\right) \tag{3}$$

$$\frac{\partial \hat{I}_{ds}}{\partial x_m} = \frac{\partial f_{ANN}\left(V_{gs}, V_{ds}, W/L\right)}{\partial x_m} = \text{grad}_m \tag{4}$$

Where $\hat{I}_{ds}$is the output of the MLP, $I_{ds}$is the expected value. grad$_m$is the first derivative of the output concerning some input.

MLP networks are trained with Adaptive Momement Estimation algorithm.[16] It is a first-order optimization algorithm that can replace the traditional stochastic gradient descent process, which iteratively updates neural network weights based on training data[18].Computational graphs in Pytorch are often able to capture any order derivative of the device output by taking into account gradients in the network. Therefore, the gradient cannot be calculated manually. Typically device characteristics span multiple orders of magnitude. In order to avoid the neural network outputting negative current and compress the output range, it is necessary to logarithm the current. In addition, data need to be standardized and shuffed before training.

## C. Bayes Optimizer

Bayesian optimization finds the value that minimizes the objective function by building a surrogate function (probabilistic model) based on past evaluation results of the objective.[19]

The objective function of this optimization is chosen as the R2 coefficient of determination. It is defined as follows:

$$R^2\left(I_{ds},\hat{I}_{ds}\right)=1-\frac{\sum_{i=0}^{n_{\text{samples}}-1}\left(I_{ds}-\hat{I}_{ds}\right)^2}{\sum_{i=0}^{n_{\text{sampies}}-1}\left(I_{ds}-\bar{I}_{ds}\right)^2} \tag{5}$$

When R2 is closer to 1, the model is better.

For each hyperparameter scheme, there is a corresponding optimal score for the pair[20]. Combining it is $D=\{(x_{11},x_{21},\text{val}(x_{11},x_{21})),(x_{12},x_{22},\text{val}(x_{12},x_{22})),\cdots\}$ Where $x_{11},x_{21}$ means the number of first and second hidden layer respectively in the first search, $val(x_{11},x_{21})$ represents the corresponding R2 score. Others and so on.

The object to be searched is the number of neurons in each hidden layer(hid1 and hid2). The selection range is (0-25). With each iterative search, the Bayesian optimization algorithm selects a value for each hyperparameter from the domain space.

The regression target is obtained by combining a set of hidden layer neuron numbers, which is called the surrogate function. In this work, a Gaussian process is selected. Random initialization is used to obtain D, and Gaussian process regression is used to solve the mean of the unknown points, that is, update the posterior $P(\text{val}(x_{1^*},x_{2^*})\mid(x_{1^*},x_{2^*}),D)$ of the model. where $x_{1^*},x_{2^*}$ is the hyperparameter that has not yet been calculated. $D$ is the set of observed hyperparameters and corresponding scores. Since $val\sim GP(\mu,K)$ , it is predicted that $P(val\mid x,D)=N\left(val\mid\hat{\mu},\hat{\sigma}^2\right)$.

Through the posterior, use the acquisition function to obtain a new hyperparameter combination scheme. where $x_{1t}\in x_{1*},x_{2t}\in x_{2^*}$ . the acquisition function is used to evaluate how to select new sampling points (trade-off between exploration and exploitation). Finally, the new parameters are selected as:

$$(x_{1t},x_{2t})\leftarrow\arg\max S((x_{1^*},x_{2^*}),p(\text{val}\mid(x_{1^*},x_{2^*}),D)) \tag{6}$$

Where $S$ is the acquisition function.

Then Substitute the new $x_{1t},x_{2t}$ into the model, calculate the corresponding $\text{val}(x_{1t},x_{2t})$ , and incorporate it into the set

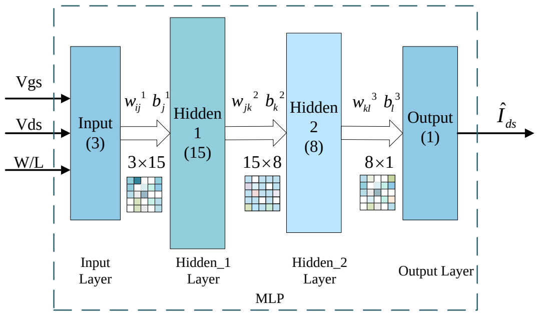


FIG. 2. topology of the MLP

$D$. Iterate the above steps until a suitable optimal hyperparameter is found. The final number of hidden layers is 15 and 8 respectively. The topology of the entire model is shown in Fig.2.

## D. Genetic Algorithm

Genetic algorithm is a global optimization algorithm that not based on gradient information[17,21]. It is a computational model that simulates the natural selection and genetic mechanism of Darwin's theory of biological evolution. The essence of genetic algorithm is to use group search technology. According to the principle of survival of the fittest, it evolves generation by generation, and finally obtains the optimal solution or quasi-optimal solution. The process of genetic algorithm is as follows:

1. The generation of the initial population. Encodes the initial weights and biases of the neuron. In the approximation problem of multi-dimensional and high-precision continuous functions, the commonly used coding method is real number coding. In this encoding method, each individual contains the weights and biases of the network.

2. fitness function. The fitness function is also the objective function optimized by the genetic algorithm. A set of initial values is chosen to assign to the MLP for training, taking the loss function as the fitness of the individual.

3. Choose operation. The survival of the fittest among individuals in a group requires selection operations. The algorithm chosen is Tournament Selection, it is a method of choosing the individual from the set of individuals. The winner of each tournament is selected to perform crossover.

4. Crossover operation. Two individuals in the population after the selection operation are arbitrarily selected as crossover objects. That is two parent individuals undergo chromosome exchange and recombination to generate two child individuals. The cross operation method of the l-th chromosome $a_l$ and the k-th chromosome $a_l$ at the j position is defined as:

$$\begin{cases} a_{kj}=a_{kj}(1-b)+a_{lj}b \\ a_{lj}=a_{lj}(1-b)+a_{kj}b \end{cases} \tag{7}$$

where b is a random number between$[0,1]$intervals .

5. Mutation operation. Mutation is a process in which one allele of a gene is replaced by another allele, resulting in a new genetic structure. Using non-uniform variation, the j-th gene $a_{ij}$ of the i-th individual can be defined as:

$$a_{ij} = \begin{cases} a_{ij} + (a_{ij} - a_{\max}) * f(g) & r > 0.8 \\ a_{ij} + (a_{\min} - a_{ij}) * f(g) & r \le 0.8 \end{cases} \tag{8}$$

$$f(g) = \delta\left(1 - \frac{g}{G_{\max}}\right)^2 \tag{9}$$

Where $a_{\max}$and $a_{\min}$ are the upper and lower bounds of the gene $a$, $\delta$ is a random number, $g$ is the current iteration number, $G_{\max}$ is the set maximum evolution number, and $r$ is a random number between $[0,1]$ intervals. Fig.3 show the iterative results of two other heuristics algorithm and genetic algorithm. It can be seen that as the number of iterations increases, the iteration error is closer to 0.

It is worth nothing that the genetic algorithm outperforms the other two algorithms. Although the convergence rate of GA is not the fastest, but the initial search is relatively stable. DE search stagnation, the individual is difficult to update, did not converge to the extreme point. PSO algorithm converges faster in the early stage of evolution. But in the later stage of evolution, the convergence speed of the algorithm slows down due to the lack of effective mechanism to make the algorithm escape from local extremum. Compared with PSO, chromosomes share information with each other in genetic algorithm, so the movement of the whole population is relatively uniform to the optimal region. For optimization problems, compared with GA, DE and PSO algorithm, the convergence speed is faster, but PSO is easy to fall into local optimal solution, and the algorithm is unstable.

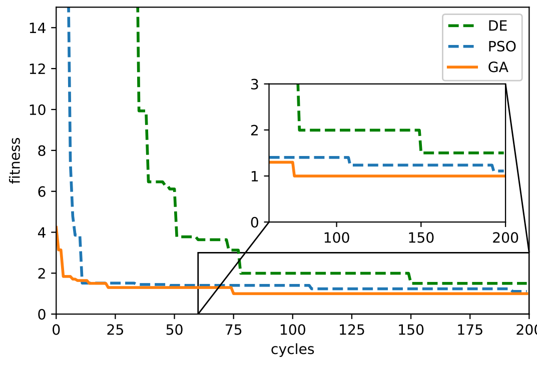
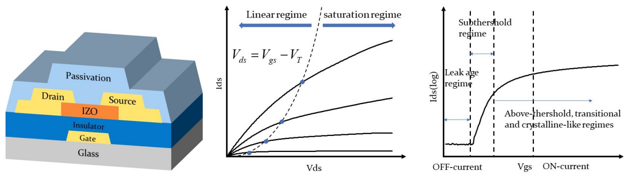


FIG. 3. result of DE PSO and GA

## III. RESULT AND DISCUSSION

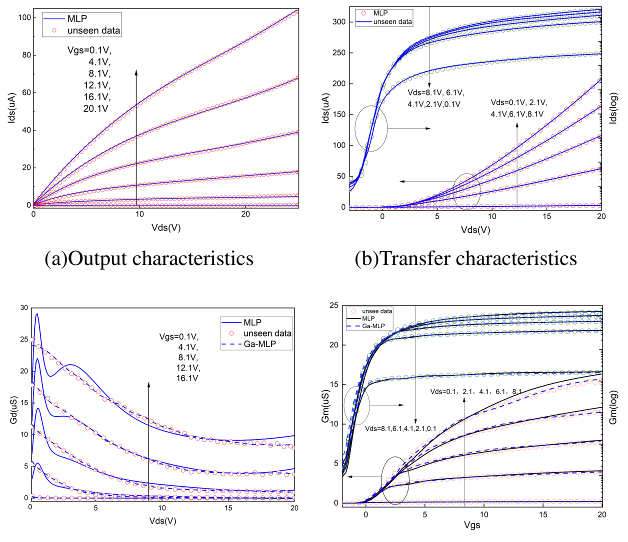
The schematic diagram and characteristic curve of an IZO-TFT is shown in Fig.4. Its drain-source voltage, gate-source voltage and channel width-length ratio are used as inputs to the proposed model. The current between the drain and source stages as the output of the model. In the following, visualization and sensitivity analysis are used to verify a possibility that the proposed model is intrinsically linked to the physical device.



(a)schematic diagram (b)characteristic curve

FIG. 4. schematic diagram and characteristic curve of an IZO-TFTs. (a) schematic diagram. (b) characteristic curve

The MLP with two hidden layers (denoted as 3-15-8-1, representing three inputs, the first hidden layer has 15 neurons, the second hidden layer has 8 neurons, and 1 output) applied to transistor model. The fitting results of ANN-based compact model are shown in Fig.5. In Fig.5 , dots and lines represent measurement data and model output, respectively. Fig.5(a)-(b)shows the fitting results of Ids–Vds, Ids–Vgs,Fig.5 (c)–(d) shows the fitting results of Vgs-Gm after GA initialization and without initialization, respectively. It can be clearly seen that when there is no initialization, MLP has a certain deviation in the prediction of small signal parameters. This may be because of local optima stuck in late stages of training. This problem is solved by using the excellent global optimization ability of the genetic algorithm.



(a)Output characteristics (b)Transfer characteristics

(c)Output conductance (d)Transconductance

FIG. 5. schematic diagram and characteristic curve of an IZO-TFT. (a) schematic diagram. (b) characteristic curve

By visualizing each hidden layer, it is clear that each neuron is functioning in a different area. parameter input ranges. The Fig.6 shows the neuron output with different W/L, drain

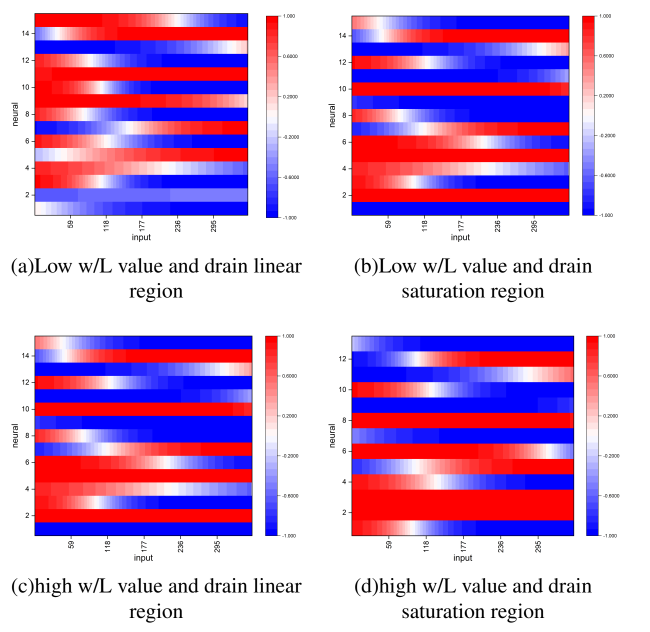

(a)Low w/L value and drain linear region (b)Low w/L value and drain saturation region (c)high w/L value and drain linear region (d)high w/L value and drain saturation region

FIG. 6. The output of the first hidden layer with different input regions visualizes the heat map

voltages, and gate voltages. It can be seen from the output heatmap that some neurons only work in the threshold region of the drain linear region with lower W/L value, and some work in other regions. Thus, the drain voltages can be divided into linear and saturation regions, and the gate voltages into subthreshold and above threshold regions. If the W/L value is only divided into high and low, then there will be 8 working methods in combination, that is, at least 8 neurons are required to achieve a better distinction. It can be seen from the output of each neuron that some areas are controlled by multiple neurons, and there is also one neuron that controls the output of multiple areas.

Finally, the 15 neurons were searched by Bayesian work in each division area. For the second hidden layer, the accepted input is the output of the previous hidden layer. That is, the neurons of this layer inherit the information of the previous layer, and make a more abstract feature representation on the basis of the previous layer. One possible situation is to integrate different regional features learned in the previous layer, and finally output the integrated information. Since there are 8 region classes in total, there should be 8 neurons by conjecture. The final result of Bayesian search is 8, which is in line with the conjecture.

When a neural network is trained, its weights are fixed. The output of the neuron is defined as: $Y = n^l(X)$. Defining the sensitivity of a neuron as the variance caused by the input, it can be expressed as the conditional expectation.[22] Fig.7 shows the sensitivity of some neurons to the input. This could also proves that different neurons work in different regions.

Write the mathematical equations for all training parameters of the built model into a Verilog-a file for further circuit simulations. Use the obtained model to build a second-order inverter, and the result is shown in the Fig.8 That is to say, it can realize circuit simulation.

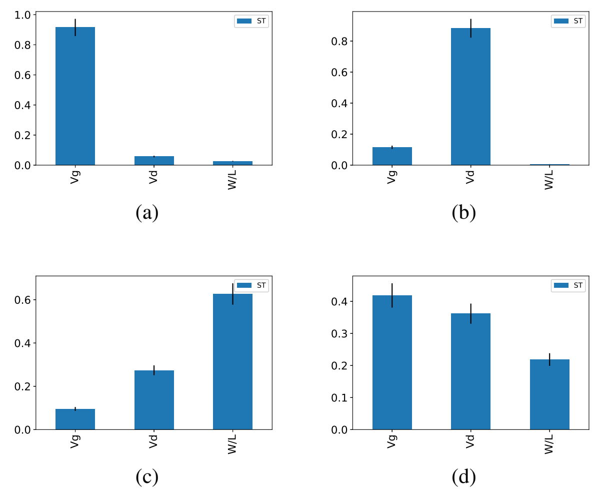


FIG. 7. Sensitivity of different neurons to different inputs

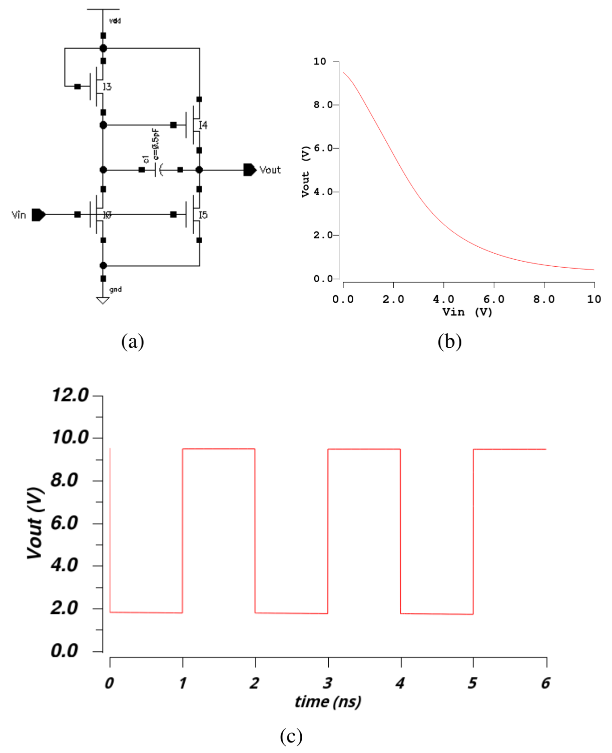


FIG. 8. Double-Stage invert. (a) Schematic diagram. (b) Output value of the double-stages inverter. (c) Voltage timing diagram

## IV. CONCLUSION

In this paper, an artificial neural network based on a general transistor compact model is designed. The accuracy and efficiency of the model are improved by the assisted combination of Bayesian optimizer and genetic algorithm. Finally, the reliability of the modeling is verified by the visual neural network hidden layer and sensitivity analysis method. Since the model does not depend on the specific physical characteristics of the device, it has great generalization performance and can be widely used in other device modeling.

## V. REFERENCE

[1] K. Nomura, H. Ohta, A. Takagi, T. Kamiya, M. Hirano, and H. Hosono, "Room-temperature fabrication of transparent flexible thin-film transistors using amorphous oxide semiconductors," nature **432**, 488–492 (2004).

[2] M. J. Mirshojaeian Hosseini and R. A. Nawrocki, "A review of the progress of thin-film transistors and their technologies for flexible electronics," Micromachines **12**, 655 (2021).

[3] D.-B. Ruan, P.-T. Liu, Y.-C. Chiu, P.-Y. Kuo, M.-C. Yu, K.-Z. Kan, T.-C. Chien, Y.-H. Chen, and S. M. Sze, "Effect of interfacial layer on device performance of metal oxide thin-film transistor with a multilayer high-k gate stack," Thin Solid Films **660**, 578–584 (2018).

[4] A. Khakifirooz, O. M. Nayfeh, and D. Antoniadis, "A simple semiempirical short-channel mosfet current–voltage model continuous across all regions of operation and employing only physical parameters," IEEE Transactions on Electron Devices **56**, 1674–1680 (2009).

[5] R. Vishnoi and M. J. Kumar, "A compact analytical model for the drain current of gate-all-around nanowire tunnel fet accurate from sub-threshold to on-state," IEEE Transactions on Nanotechnology **14**, 358–362 (2015).

[6] D. E. Root, "Future device modeling trends," IEEE Microwave Magazine **13**, 45–59 (2012).

[7] K. Hornik, M. Stinchcombe, and H. White, "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," Neural networks **3**, 551–560 (1990).

[8] K. Hornik, "Approximation capabilities of multilayer feedforward networks," Neural networks **4**, 251–257 (1991).

[9] Y. Lei, X. Huo, and B. Yan, "Deep neural network for device modeling," in *2018 IEEE 2nd Electron Devices Technology and Manufacturing Conference (EDTM)* (IEEE, 2018) pp. 154–156.

[10] L. Zhang and M. Chan, "Artificial neural network design for compact modeling of generic transistors," Journal of Computational Electronics **16**, 825–832 (2017).

[11] M. Lázaro, I. Santamaría, and C. Pantaleón, "Neural networks for large and small-signal modeling of mesfet/hemt transistors: a comparative study," in *Proceedings of the 17th IEEE Instrumentation and Measurement Technology Conference [Cat. No. 00CH37066]*, Vol. 3 (IEEE, 2000) pp. 1272–1277.

[12] A. Aoad, "Reconfigurable microstrip antenna optimization through artificial neural networks," in *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)* (IEEE, 2020) pp. 1–3.

[13] J. B. Mockus and L. J. Mockus, "Bayesian approach to global optimization and application to multiobjective and constrained problems," Journal of optimization theory and applications **70**, 157–172 (1991).

[14] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani, "Predictive entropy search for efficient global optimization of black-box functions," Advances in neural information processing systems **27** (2014).

[15] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: four layers versus three," IEEE Transactions on Neural Networks **8**, 251–255 (1997).

[16] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," arXiv preprint arXiv:1904.09237 (2019).

[17] J. D. Schaffer, "Some experiments in machine learning using vector evaluated genetic algorithms," Tech. Rep. (Vanderbilt Univ., Nashville, TN (USA), 1985).

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980 (2014).

[19] J. Mockus, *Bayesian approach to global optimization: theory and applications*, Vol. 37 (Springer Science & Business Media, 2012).

[20] P. I. Frazier, "A tutorial on bayesian optimization," arXiv preprint arXiv:1807.02811 (2018).

[21] S. Bornholdt and D. Graudenz, "General asymmetric neural networks and structure design by genetic algorithms," Neural networks **5**, 327–334 (1992).

[22] Y. Zhang and B. Wallace, "A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification," arXiv preprint arXiv:1510.03820 (2015).

[23] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," arXiv preprint arXiv:1012.2599 (2010).