Physics-Inspired Neural Networks (Pi-NN) for Effcient Device Compact Modeling

Mingda (Oscar) Li[1], Ozan İrsoy[2], Claire Cardie[2] and Huili Grace Xing[1]
1. School of Electrical and Computer Engineering, Cornell University, NY 14850, USA
2. Department of Computer Science, Cornell University, NY 14860, USA
Emails: ml888@cornell.edu, grace.xing@cornell.edu

***Abstract*—** We present a novel Physics-Inspired Neural Network (Pi-NN) approach for compact modeling. Development of high-quality compact models for devices is key to connect device science with applications. One recent approach is to treat compact modeling as a regression problem in machine learning. The most common learning algorithm to develop compact models is the Multilayer Perceptron (MLP) neural network. However, device compact models derived using MLP neural networks often exhibit unphysical behavior, which is eliminated in the Pi-NN approach proposed in this work since the Pi-NN incorporates fundamental device physics. As a result, smooth, accurate and computationally efficient device models can be learnt from discrete data points by using Pi-NN. This work sheds some light on the future of the neural network compact modeling.

I. Introduction

Device compact modeling bridges device science to applications, therefore it plays a very important role in device research. There are two extremes for device modeling, one is purely physical and the other is purely empirical. Looking at these two extremes, a purely physical modeling method, such as NEMO [1], is computational expensive for use in circuit simulations, and a purely empirical modeling method, such as look-up table, has limited generalization (extrapolation) ability. Therefore, to find a middle ground between purely physical and purely empirical models, the Electron Design Automation industry, represented by the Compact Model Coalition, chooses to promote physics-based compact models. These use fundamental device physics as the building blocks, then add empirical fitting to modify and merge different analytical physical expressions into smooth functions. However, developing high-quality physics-based compact models is very time-consuming, and therefore often not available for emerging devices. As an alternative, regression with machine learning can be used to model relationships between different variables with certain generalization abilities. Among different regression algorithms, the neural network modeling method has raised a lot of interests [2-4] given the fact that it is theoretically capable of arbitrarily accurate approximation to any function and its derivatives [5]. Previous works [2-4] used Multilayer Perceptron (MLP) neural networks to develop compact models (shown in Fig. 1), which are prone to having unphysical behavior (see Fig. 5(d-e)). To eliminate the unphysical behavior, we have developed a novel neural network structure: Physics-Inspired Neural Network (Pi-NN), with fundamental device physics embedded. As a result, the Pi-NN can be trained to generate an accurate, smooth, and computational efficient device compact model.

II. Thin-TFET and Training Procedure

To illustrate the principles of Pi-NN, we develop compact models for the DC I-V curves of a transistor. Physics-based device modeling is typically challenging because the I-V curves are highly nonlinear and requires different analytical physical expressions in different bias windows. Therefore it is usually difficult to handcraft an infinitely differentiable function from these physical expressions. Since high quality physics-based compact models are yet unavailable for emerging devices, such as Tunnel Field Effect Transistors (TFETs) [6], the neural network modeling approach has an added attraction. Here we used a novel device proposed in our group, a Thin-TFET [7] (**T**wo-dimensional **H**eterojunction **In**terlayer **T**unneling **F**ield **E**ffect **T**ransistor), as an example device for testing the neural network modeling techniques. The schematic device structure of an n-type Thin-TFET is shown in Fig. 3. The training data are simulated [7] for the top gate voltage (V_{TG}) from 0 to 0.4 V and the drain-source voltage (V_{DS}) from -0.1 to 0.4 V with an uniform step of 0.01 V, while the test data are for V_{TG} from 0.005 to 0.405 V and V_{DS} from -0.095 to 0.405 V with an uniform step of 0.01 V. The detailed training procedure is shown in Fig. 4.

III. MLP Neural Network Modeling and Unphysical Behavior

In this section, we use the MLP neural network to generate a compact model for the DC I-V curves of the Thin-TFET. After some initial training, we choose to use MLP neural networks with two hidden layers and defined its hyperparameter as *(i, j)*, where *i* is the number of neurons in the first hidden layer and *j* is the number of neurons in the second hidden layer. Each neuron uses the hyperbolic tangent function $tanh(x)=(e^x-e^{-x})/(e^x+e^{-x})$ as the activation function. By choosing the hyperparameter *(i, j)* to be (5, 5), (7, 7) and (9, 9), these three MLP neural networks were trained for 5 million epochs. Its well-established learning algorithms are described in (1) and (4) [8]. Using the loss function defined in (3), the root-mean-squared (R.M.S) deviations for training data and test data are plotted in Fig. 5(a). The test errors are used to evaluate the generalization ability of the model, namely how the model fit the unseen data. As shown in Fig. 5(a), the test errors stay close to the training errors, which indicated a good generalization. We choose to plot the I-V curves modeled by the MLP neural network with 7 *tanh* neurons in the first and second hidden layers, which gives a neural network with 15 neurons and 85 parameters in total. Figure 5(b-e) show the I-V curves generated by the MLP neural network compact model along with the training data and the test data. Good fitting in the linear scale is achieved for both the I_D-V_{DS} and the I_D-V_{TG} curves. However, if we zoom in the region near $V_{DS} = 0$, I_D is not zero when V_{DS} is zero, indicating the I_D-V_{DS} relationship is unphysical around $V_{DS} = 0$ (see Fig. 5(d) and the inset). Moreover, the I_D-V_{TG} relationship is also unphysical in the sub-threshold region (shown in Fig. 5(e)). The fundamental reason of these unphysical behaviors is that the MLP neural network has no knowledge of the device physics; therefore the fitting is no longer physical when I_D is very small. In order to eliminate these unphysical behaviors, we have to design a neural network with *a priori* knowledge of the fundamental device physics.

IV. A Physics-Inpired Neural Network Design

First, we note that the inputs V_{DS} and V_{TG} are related to two different physical effects: V_{DS} drives the current through the device while V_{TG} controls the channel potential profile to change the magnitude of the current. Therefore V_{DS} and V_{TG} should be fed to two different neural networks. According to the fundamental device physics, we know I_D-V_{DS} curves have a linear region at small V_{DS} and a saturation region at large V_{DS}. This behavior is similar to a *tanh* function. This indicates V_{DS} should be fed into a neural network with *tanh* activation functions (*tanh subnet*). To ensure I_D equals zero when V_{DS} equals zero, all the *tanh* neurons in the *tanh subnet* must have no bias terms. On the other hand, the I_D-V_{TG} curves have an exponential turn-on in the sub-threshold region and then become a polynomial in the ON region. This is best simulated as a sigmoid function $sig(x)=1/(1+e^{-x})$. Therefore, V_{TG} is fed into a neural network with sigmoid activation functions (*sig subnet*). It should be noted that we assumed gate leakage current is negligible, so V_{TG} would not change the sign of I_D. The final drain current is the entrywise product of the outputs of the *tanh subnet* and the *sig subnet*. This entrywise product reflects the control of V_{TG} on the drain current driven by V_{DS}. In addition, V_{DS} can affect the channel potential profile controlled by V_{TG} due to various non-ideal effects such as the short channel effects. A simple but effective remedy for this is to add weighted connections from each layer in the *tanh subnet* to its corresponding layer in the *sig subnet*. By embedding the above device physics in a neural network structure, we arrive at the Physics-Inspired Neural Network (Pi-NN) (shown in Fig 2). This novel neural network is reminiscent of the peephole Long-Short Term Memory (LSTM) [9], with the notable difference that the Pi-NN does not propagate through time. The pseudo-codes for the feed-forward and error back-propagation algorithms are shown in (2) and (5).

V. Physics-Inspired Neural Network Modeling

After initial training, we chose to use Pi-NNs with one hidden layer and define the hyperparameter as (*m, n*), where *m* is the number of the *tanh* neurons in the hidden layer and *n* is the number of the *sigmoid* neurons in the same hidden layer. The test errors stay close to the training errors as shown in Fig. 6(a), which indicates good generalization. Balancing between model complexity and accuracy, we chose the model with the hyperparameter (2, 3), which give a small Pi-NN model with only 7 neurons and 20 parameters in total. Excellent modeling is demonstrated in both the ON region (shown in Fig. 6(b-c)) and the sub-threshold region (shown in Fig. 6(e)). The I_D-V_{DS} relationship around V_{DS} equals zero is shown in Fig. 6(d). All the unphysical behaviors that appeared in the MLP neural network model have been eliminated. Moreover, thanks to the embedded device physics, the Pi-NN requires much less parameters than the MLP neural network, which results in a smaller, more efficient compact model.

VI. Conclusions

Motivated by the need of high-quality compact models for emerging devices, we have proposed a novel neural network: Pi-NN, for compact modeling. With fundamental device physics incorporated, the Pi-NN method can produce accurate, smooth and computational efficient transistor models with good generalization ability. Thin-TFET is presented as an example to illustrate the capabilities of Pi-NN: a relatively small compact model is achieved with excellent fitting in both the ON and the sub-threshold region of the Thin-TFET. Finally, the Pi-NN approach is readily implementable on commercial measurement and modeling systems.

Acknowledgment

This work was supported in part by the Center for Low Energy Systems Technology (LEAST), one of the six SRC STARnet Centers, sponsored by MARCO and DARPA.

References

[1] Steiger, Sebastian, et al. "NEMO5: a parallel multiscale nanoelectronics modeling tool." (2011).

[2] Xu, Jianjun, and David E. Root. "Advances in artificial neural network models of active devices." *2015 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*. IEEE, 2015.

[3] Hammouda, H. Ben, et al. "Neural based models of semiconductor devices for SPICE simulator." American Journal of Applied Sciences 5.4 (2008): 385-391.

[4] Wang, Fang, and Qi-Jun Zhang. "Knowledge-based neural models for microwave design." IEEE Transactions on Microwave Theory and Techniques 45.12 (1997): 2333-2343.

[5] Hornik, Kurt. "Approximation capabilities of multilayer feedforward networks." *Neural networks* 4.2 (1991): 251-257.

[6] Seabaugh, Alan C., and Qin Zhang. "Low-voltage tunnel transistors for beyond CMOS logic." *Proceedings of the IEEE* 98.12 (2010): 2095-2110.

[7] Li, Mingda Oscar, et al. "Two-dimensional heterojunction interlayer tunneling field effect transistors (thin-tfets)." *IEEE Journal of the Electron Devices Society* 3.3 (2015): 200-207.

[8] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." *Cognitive modeling* 5.3 (1988): 1.

[9] Gers, Felix A., Nicol N. Schraudolph, and Jürgen Schmidhuber. "Learning precise timing with LSTM recurrent networks." *Journal of machine learning research* 3.Aug (2002): 115-143.

	Previous works: *Multilayer Perceptron (MLP) Neural Network*	**This work:** *Physics-Inspired Neural Network (Pi-NN)*

Neural Network Achitecture

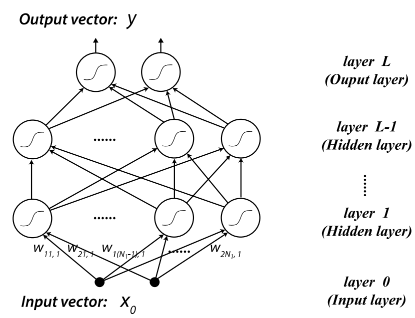


Figure 1: The Multiplayer Perception (MLP) neural network model

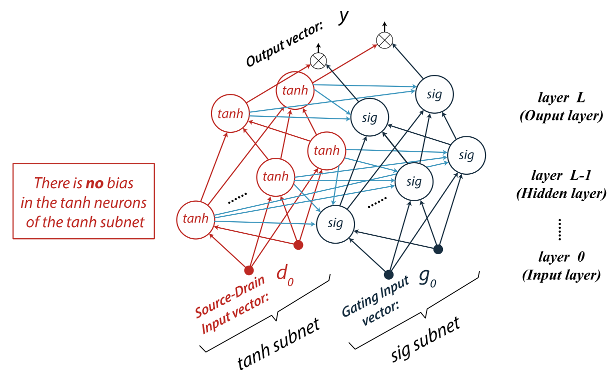


Figure 2: The Physics-Inspired Neural Network (Pi-NN) model

Symbols, Constraint and Parameters

Previous works (MLP)

Symbols:

- l : Layer Index
- N_l : No. of neurons in the l^{th} layer
- x_l : The output vector of the l^{th} layer
- z_l : The intermediate vector of the l^{th} layer
- L : No. of layers in the network
- y : The output vector of the network
- x_0 : The input vector of the network
- $\circ$: The entrywise product of two vectors

Parameters: W_l : the weight matrix, whose element $w_{ji,l}$ connects the i^{th} neuron of the *(l - 1)*th layer to j^{th} neuron of the l^{th} layer

b_l : the bias vector, whose element $b_{i,l}$ is the bias of the i^{th} neuron of the l^{th} layer

This work (Pi-NN)

Symbols:

- l : Layer Index
- $N_l^{T(S)}$: No. of neurons in the l^{th} layer of the ***tanh(sig) subnet***
- $d_l(g_l)$: The output vector of the l^{th} layer in the ***tanh(sig) subnet***
- z_l : The intermediate vector of the l^{th} layer in the ***tanh(sig) subnet***
- L : No. of layers in the network
- y : The output vector of the network
- $d_0(g_0)$: The input vector of the ***tanh(sig) subnet***
- $\circ$: The entrywise product of two vectors

Constraint: $N_L^T = N_L^S$ (The output layers of the two subnet have same no. of neurons)

Parameters: $W_l^{T(S)}$: the weight matrix for the *tanh(sig) subnet, whose element* $w_{ji,l}^{T(S)}$ connects the i^{th} neuron of the *(l - 1)*th layer to j^{th} neuron of the l^{th} layer

W_l^P : the weight matrix for the peephole connections from the *tanh subnet* to the *tanh subnet, whose element* $w_{ji,l}^P$ connects the i^{th} neuron of the l^{th} layer in the *tanh subnet* to j^{th} neuron of the l^{th} layer in the *sig subnet*

b_l^S : the bias vector, whose element $b_{i,l}^S$ is the bias of the i^{th} neuron of the l^{th} layer in the *sig subnet*

*(**Note**: There is **no** bias in the tanh neurons of the tanh subnet)*

Feedforward

Previous works (MLP)

```
for each layer l in [1 ,2 ...,L] :
```

$$z_l = W_l x_{l-1} + b_l$$
$$x_l = \sigma(z_l)$$
$$y = x_L \qquad (1)$$

This work (Pi-NN)

```
for each layer l in [1 ,2 ...,L] :
```

$$z_l^T = W_l^T d_{l-1} \quad \text{(* tanh subnet *)}$$
$$d_l = tanh(z_l^T) \quad \text{(* tanh subnet *)}$$
$$z_l^S = W_l^P d_l + W_l^S g_{l-1} + b_l^S \quad \text{(* sig subnet \& peephole connections *)}$$
$$g_l = sig(z_l^S) \quad \text{(* tanh subnet *)}$$
$$y = d_L \circ g_L \qquad (2)$$

Loss Function

$$E = \frac{1}{2}\|y - t\|^2 \quad \text{where } t \text{ is the desired output vector of } y \qquad (3)$$

Error Backpropagation

Previous works (MLP)

(* For the output layer *)

$$\frac{\partial E}{\partial x_l} = \frac{\partial E}{\partial y_l}$$

(* For the hidden layers *)

```
for each layer l in [L ,L - 1 ...,1] :
```

$$\frac{\partial E}{\partial z_l} = \frac{\partial x_l}{\partial z_l} \circ \frac{\partial E}{\partial x_l}$$
$$\frac{\partial E}{\partial x_{l-1}} = (W_l)^T \frac{\partial E}{\partial z_l} \qquad (4)$$

This work (Pi-NN)

(* For the output layer *)

$$\frac{\partial E}{\partial g_L} = d_L \circ \frac{\partial E}{\partial y}, \quad \frac{\partial E}{\partial d_L} = g_L \circ \frac{\partial E}{\partial y}$$

(* For the hidden layers *)

```
for each layer l in [L ,L - 1 ...,1] :
```

$$\frac{\partial E}{\partial z_l^S} = g_l \circ (\vec{1} - g_l) \circ \frac{\partial E}{\partial g_l} \quad \text{(* sig subnet *)}$$
$$\frac{\partial E}{\partial g_{l-1}} = (W_l^S)^T \frac{\partial E}{\partial z_l^S} \quad \text{(* sig subnet *)}$$
$$\frac{\partial E}{\partial z_{j,l}^T} = (\vec{1} - d_l \circ d_l) \circ \left(\frac{\partial E}{\partial d_l} + (W_l^P)^T \frac{\partial E}{\partial z_l^S}\right) \quad \text{(* tanh subnet \& peephole connections *)}$$
$$\frac{\partial E}{\partial d_{l-1}} = (W_l^T)^T \frac{\partial E}{\partial z_l^T} \quad \text{(* tanh subnet *)} \qquad (5)$$

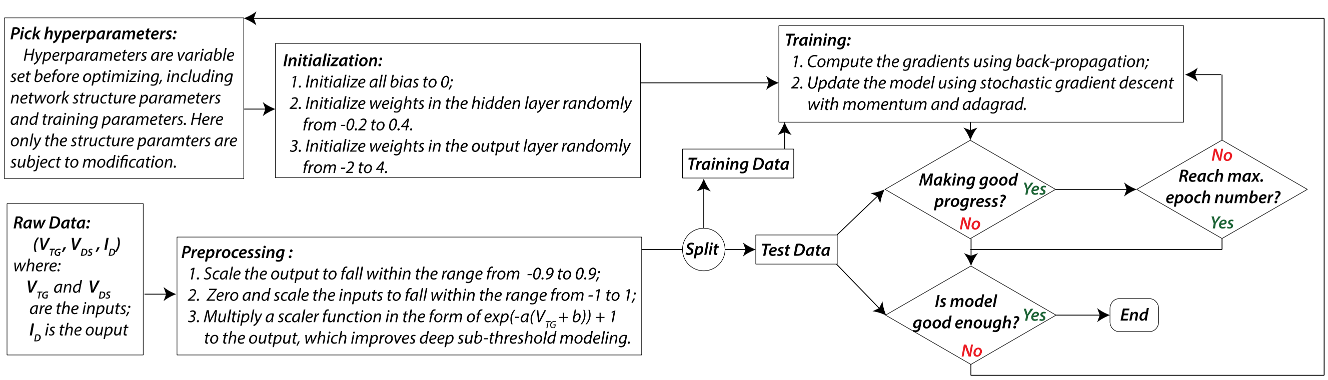
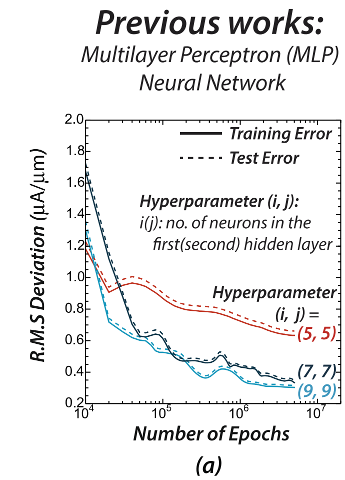


Figure 3: A training procedure for neural network device compact modeling.



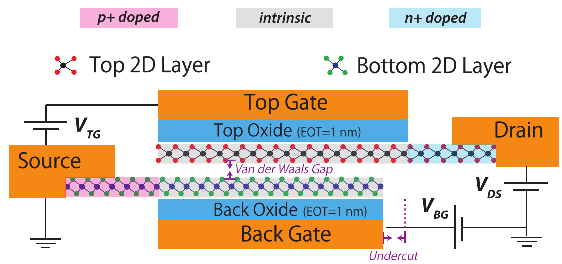
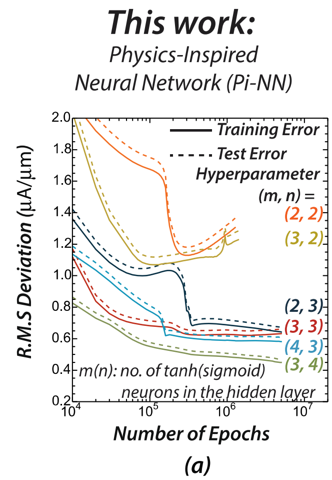
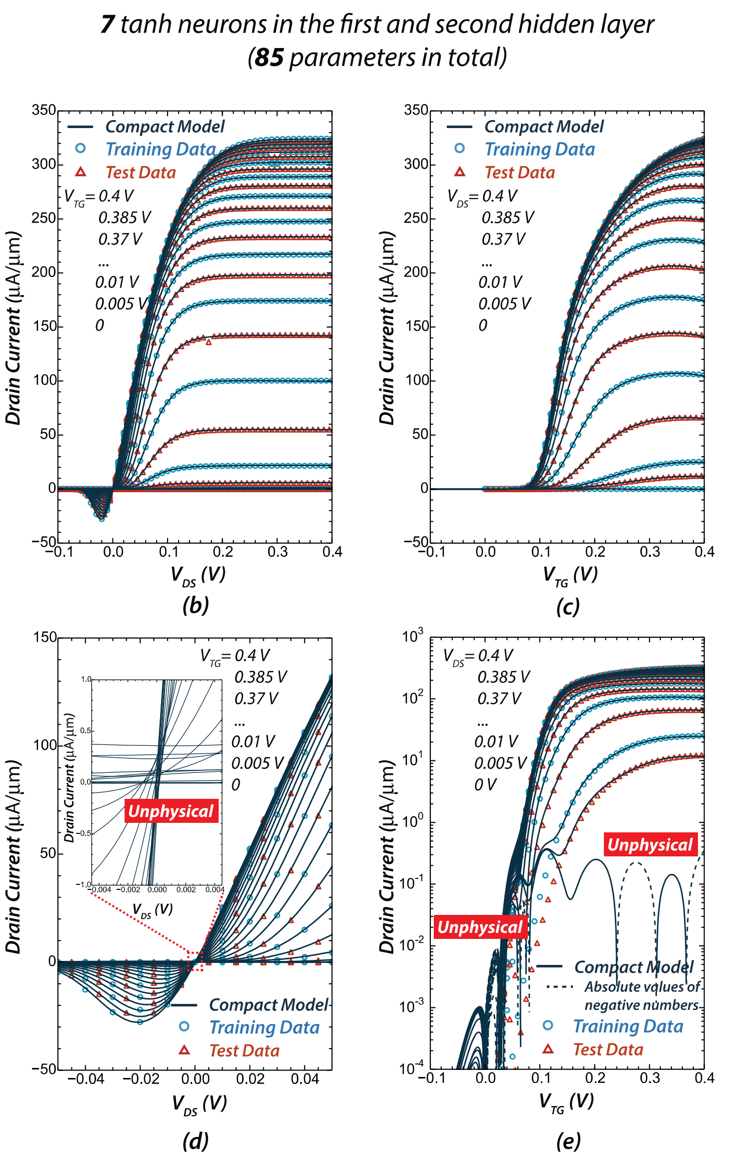


Figure 4: The schematic device structure of an n-type Thin-TFET [7]. Its I-V curves are obtained by sweeping the top gate (V_{TG}) with the back gate (V_{BG}) grounded.





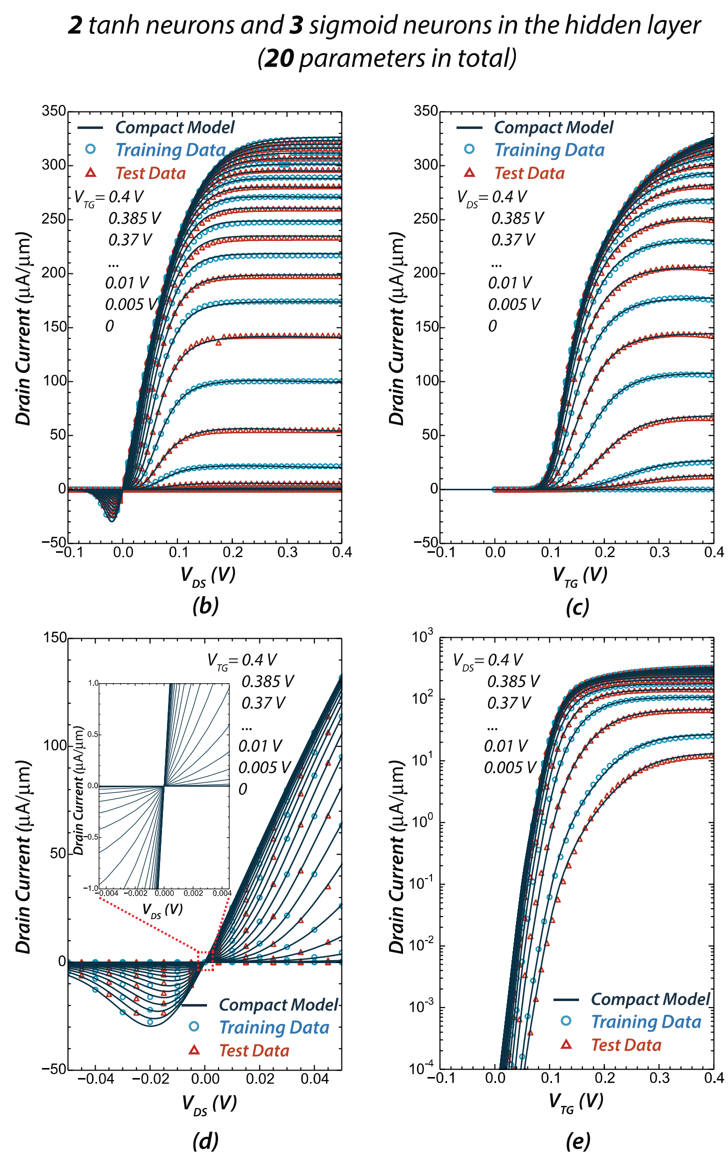


Figure 5: For the MLP neural network used in previous works [2-4], (a) the training errors and test errors for a variety of hyperparameters. From (b) to (e), the I-V curves generated by the MLP neural network model with 85 parameters are plotted along with the training data and the test data: (b) I_D versus V_{DS} at different V_{GS}; (c) I_D vs. V_{TG} at different V_{DS} in linear scale; (c) I_D vs. V_{DS} at different V_{TG} around $V_{DS} = 0$, the embeded plot shows unphysical I_D- V_{DS} relationships around V_{DS} equals 0; (d) I_D vs. V_{TG} at different V_{DS} in semi-log scale, unphysical oscillation of I_D around zero appears in the sub-threshold region and when $V_{DS} = 0$.

Figure 6: For the Pi-NN developed in this work, (a) the training errors and test errors for a variety of hyperparameters. From (b) to (e), the I-V curves generated by the Pi-NN model with 20 parameters are plotted along with the training data and the test data: (b) I_D versus V_{DS} at different V_{TG}; (c) I_D vs. V_{TG} at different V_{DS} in linear scale; (c) I_D vs. V_{DS} at different V_{TG} around $V_{DS} = 0$, the embeded plot shows well-behaved I_D- V_{DS} relationship around $V_{DS} = 0$; (d) I_D vs. V_{TG} at different V_{DS} in semi-log scale, good fitting is achieved in the sub-threshold region. All the unphysical behaviors of the MLP neural network are eliminated, and the size of the neural network is largely reduced.