Parameter Extraction from a Resistive Load Inverter Circuit using Supervised Learning Methods

R. C. Valdés García, F. García Lamont, R. Z. García Lozano, A. López Chau, R. Sánchez Fraga, G. Lastra Medina

***Abstract*— This paper presents a proposal for parameter extraction of a resistive load inverter circuit, with a Thin Film Transistor (TFT), using Artificial Neural Networks, Random Forest, Decision Trees and Support Vector Regression. Although analytical and optimization methods are usually used for this purpose, they have disadvantages such as the need for expertise or complex implementation. This work shows that these supervised learning methods are useful for this task because they can learn the parameters of the device transfer curves, obtaining a good fit between the measurements and the extracted parameters. The different methods were trained using a data set constructed from simulations performed with AIM-Spice software, where the parameters affecting different regions of the inverter characteristic curve were extracted. In the experimental stage, the Neural Networks obtained better results, with an average error rate of 6.04%. The method was also applied to real NMOS measurements and yielded minimum errors of up to 0.43%.**

***Index Terms*— Inverter circuit, Parameter extraction, Supervised learning, Modeling, Electronic simulation.**

I. Introduction

Nowadays, electrical simulators have established themselves not only as electronic circuit design tools, but also, they have been widely used to study and analyze the electrical behavior of different devices [1].

These simulators employ mathematical models that describe the electrical behavior of the devices. The proper fit between the results obtained with the simulation regarding the experimental measurements requires that the mathematical model used receives the appropriate parameter values. Therefore, it is very important to know the values of the model parameters that correspond to the device to be simulated.

R. C Valdés García, is a PhD student in Computer Science at the Autonomous University of the State of Mexico (UAEM), Texcoco, State of Mexico; rvaldesg564@alumno.uaemex.mx

F. García Lamont, is a full-time professor in the area of Artificial Intelligence at the Autonomous University of the State of Mexico (UAEM), Texcoco, State of Mexico; fgarcial@uaemex.mx

R. Z. García Lozano, is a full-time professor in the area of Electronics at the Autonomous University of the State of Mexico (UAEM), Ecatepec, State of Mexico; rzgarcial@uaemex.mx

A. López Chau, is a full-time professor in the area of artificial intelligence at the Autonomous University of the State of Mexico (UAEM), State of Mexico; alchau@uaemex.mx

R. Sánchez Fraga, is a researcher from Center for Engineering and Industrial Development (CIDESI), Queretaro, Mexico; rodolfo.sanchez@cidesi.edu.mx

G. Lastra Medina, is a researcher from Center for Engineering and Industrial Development (CIDESI), Queretaro, Mexico; gonzalo.lastra@cidesi.edu.mx

Usually, the extraction of these electrical parameters is mostly done with extraction methods that are classified into analytical methods [1-3], and methods based on mathematical optimization such as genetic algorithms (GAs) [4-14] and even fuzzy logic [15]. In both cases, depending on the device and measurement conditions, the extraction methods depend on the human expertise or the computational load to obtain the parameters that adequately represent the actual behavior of the devices.

The parameter extraction using analytical methods can be slow, tedious and their difficultness may increase depending on the device model, and, besides, the process is subject to human errors [2, 3]. On the other hand, although GAs and other optimization techniques are useful for finding a solution in very large search spaces, they have notorious disadvantages e.g., GAs performance depends onto determine the fitness function to minimize or maximize, otherwise the solution may be far from the optimal value [16-18]. Supervised learning methods have advantages over optimization techniques, for example, they do not need to define a fitness function, moreover, many of them are able to extrapolate data and a trained model can provide an approximate solution to a large number of new samples [16-19].

Therefore, this paper proposes to employ the supervised learning methods Neural Networks (NN), Random Forest (RF), Support Vector Regression (SVR) and Decision Trees (DT) to extract the parameters of a TFT, which operates within a resistive load inverter circuit (IC). An inverter was selected to demonstrate the advantages of learning methods over analytical methods, which are sometimes limited to individual devices. On the other hand, TFTs were selected since they are major elements in everyday devices today, although Poly-Si TFTs were selected since it is a standard technology, to avoid complications by using technologies still under development. The need for sufficient experimental measurements for training was avoided by generating a set of simulated measurements. The contribution of this work is an important step towards improve the extraction of parameters, which could be useful for manufacturing centers of electronic devices. Having as advantage the easy implementation of these methods with many programming languages in comparison with GAs. In addition to applying for the first-time supervised learning methods to perform this task, since so far NN and SVR have been used for modeling an I-V curve [20-22], not for extraction; RF and DT have only been used for fault detection in wafer fabrication [23]. On the other hand, the extraction is not performed from

stand-alone devices, but from devices operating within application circuits. This feature offers the opportunity to analyze variations in device behavior as a function of operating time; for example, the effects of electrical stress on TFTs [24, 25]. On the other hand, the method was used to perform parameter extraction in experimental measurements of NMOS (Negative-channel Metal-Oxide Semiconductor) transistors manufactured by the Center for Engineering and Industrial Development (CIDESI), whose parameters were unknown. The method proved to provide parameters that allowed modeling the I-V curves with high accuracy.

This article is divided as follows. The current TFT model as well as supervised learning methods are presented in section II. Theoretical Concepts. The proposal is presented in section III. Methodology. The Discussion and Results are presented in section IV. The article closes with Conclusions in section V.

II. THEORICAL CONCEPTS

A. TFT Model

The transistor model (TFT) is presented by (1). This model is implemented in the Spice AIM-Spice type software [26]. For this article, the Poly-Si TFT Model PSIA2 level 16 transistor was selected. Table I lists some of the different TFT parameters with their name, units and the default value that the simulator assigns to them.

$$I_a = \begin{cases} \mu_{FET} C_{ox} \frac{W}{L}\left(V_{GT}V_{DS} - \frac{V_{DS}^2}{2\alpha_{sat}}\right), & V_{DS} < \alpha_{sat} V_{GT} \\ \mu_{FET} C_{ox} \frac{W}{L}\frac{V_{GT}^2 \alpha_{sat}}{2}, & V_{DS} \geq \alpha_{sat} V_{GT} \end{cases} \quad (1)$$

Where μ_{FET} represents the effective mobility, C_{ox} is the capacity of the oxide, α_{sat} is the Proportionality constant of Vs_{at}. From The device current when is below the threshold voltage is defined by (2).

$$I_{sub} = MUS \cdot C_{ox} \frac{W}{L} V_{sth}^2 \exp\left(\frac{V_{GT}}{V_{sth}}\right)\left[1 - \exp\left(-\frac{V_{DS}}{V_{sth}}\right)\right] \quad (2)$$

TABLE I
PARAMETERS OF TFT

Name	Parameter	Units	Default
α_{sat}	Proportionality constant of V_{sat}	-	1
AT	Drain induced barrier lowering 1	m/V	3×10^{-8}
BT	Drain induced barrier lowering 2	m·V	1.9×10^{-6}
ETA	Subthreshold ideality factor	-	7
I0	Leakage scaling constant	A/m	6
MMU	Low field mobility exponent	-	3
MU0	High field mobility	cm^2/Vs	100
μ_l	Low field mobility parameter	cm^2/Vs	0.0022
MUS	Subthreshold mobility	cm^2/Vs	1
T_{ox}	Thin-oxide thickness	m	1×10^{-7}
W	Channel width	m	-
L	Channel length	m	-
V_T=V_{GT}	Threshold voltage	V	-

Where *MUS* is the subthreshold mobility below. Other parameters such as V_{DS} and V_{GS} indicate the voltage applied at the transistor terminals drain (D) to source (S) or gate (G) to source. $V_{sth} = ETA \cdot V_{th}$, where V_{th} is the thermal voltage. Other common parameters of the TFT model are listed in Table 1.

B. Inverter Circuit

The IC is the basis of today's digital electronics, and it can be said that the design of any digital circuit requires the use of these inverter circuits. Although the inverter with resistive load is not the most used configuration in the application circuits of the TFTs, in this work this inverter has been selected for its simplicity, likewise the type of TFT used has a little complex mathematical model, these two points are helpful to test this proposal. Figure 1 shows the Electrical diagram of the circuit of a resistive load inverter. The circuit input signal (V_{in}) is supplied by the V_{GS} source, while the negated output will be obtained at the V_{out} terminal. The supply voltage is V_{DD} or V_{DS}. It is also observed an original transfer curve with a V_T=3V, R_L=510kΩ, *I0*=6A/m and *MMU*=3. The other dotted curves represent the change they have when modifying the parameters to V_T=1V, R_L=1MΩ, *I0*=10kA/m and *MMU*=3. Modifying one parameter at a time and keeping the others the same as the original.

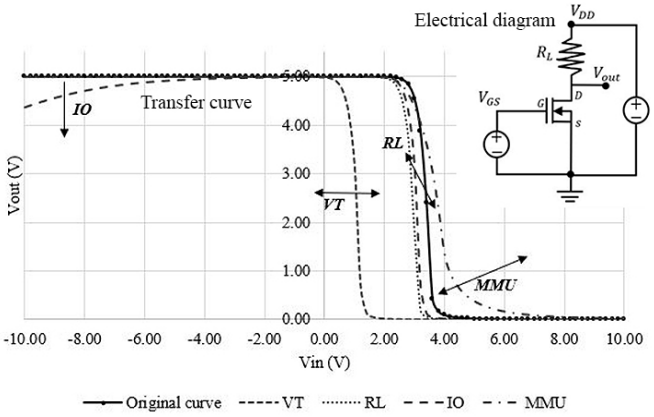

Fig. 1. Transfer curve, electrical diagram of a resistive load inverter circuit and its behavior with different parameters.

Although the TFT Poly-Si model has different parameters, to demonstrate feasibility of the proposal, in this work only four of them are extracted: V_T, is a critical parameter, since it defines the moment of activation and conduction of the transistor, this parameter shifts the curve to the left or to the right; R_L is a specific parameter of the circuit, by varying the current flow, it affects the slope of the transition region, something similar happens with *MMU*, which expands this region, at higher current flow, the transition from high state to low state is more abrupt, with lower current flow said the transition is slow; finally, *I0*, which mainly affects the subthreshold region, causing a decrease in the maximum output voltage. Parameter extraction analysis on inverters with more common configurations such as with active loads and materials currently used in the TFT will be performed in future work.

C. Neural Networks

NNs emulate the learning of living beings. They have proven to be a computational tool capable of solving a large number of problems in different disciplines [27]. In the case of regression problems, where the function can be arbitrary of the descriptive variables, NNs can estimate the parameters to approximate the mathematical model and find a solution. According to the universal approximation theorem, it says that a single-layer

network with a certain number of neurons in it can approximate any continuous regression function [28].

The neuron receives an input p that must be multiplied by a synaptic weight w and may or may not be added by a bias b, to have a preoutput n which is evaluated on a transfer function f to have the final output a [29]. Formally the output of the neuron is given by:

$$a = f(wp + b) \tag{3}$$

A network is built with a set of neurons that are interconnected with each other. The number of neurons and layers that a network must have, is not defined, since this changes with each problem to be solved. In supervised learning, the Backpropagation algorithm is usually used, which is based on the gradient. During training, the difference between the expected output, called target, with the one calculated by the network is used. This error allows to modify the synaptic weights w and bias b and thus minimize the mean square error (MSE) presented in (4).

$$F(x) = E[e^2] = E[(t - a)^2] \tag{4}$$

The update of the synaptic weights and the bias is carried out in each iteration with the following equations:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha \frac{\partial F}{\partial w_{i,j}^m} \tag{5}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha \frac{\partial F}{\partial b_i^m} \tag{6}$$

D. Decision Trees and Random Forest

Decision trees are a statistically based method, widely used to solve classification and regression problems. A tree is built by answering a series of yes/no questions, depending on the answer a branch is taken until reaching a final node, which corresponds to the output value. The DT algorithm must identify where to make a split in the input samples, these splits represent the branches of the tree.

The search for the regions $R_1, R_2, \dots, R_j$ where the splits are made is done iteratively, with the aim of finding the J regions that minimize the Residual Sum of Squares (RSS), defined by (7).

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2 \tag{7}$$

Where $\hat{y}_{R_j}$ is the mean of the output variable in the region R_J. Due to it is computationally expensive to take into account all the possible divisions in the space of the predictors (elements of the input sample), Recursive Binary Splitting is used. Applying this method, the predictor X_j and the cutoff point s, can be located in the iterations, to distribute the samples in regions of the type $\{X | X_j < s\}$ and $\{X | X_j \geq s\}$ [30-32]. After identifying the cutoff points in the predictors(X_1, X_2, $X_3, \dots, X_p$), the total RSS is given by (8).

$$\sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_j})^2 + \sum_{i:x_2 \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2 \tag{8}$$

The number of samples is i.

E. Support Vector Regression

SVR are the version of the Support Vector Machines (SVM) when the variable to be predicted is continuous. Vapnik raised the beginnings of SVMs in the 1990s [33]. Although they were initially applied for classification, today they are applied for regression. The principle of operation is to find a hyperplane that is in the middle of the closest samples of the classes and to get the maximum margin between them.

In SVM, the greater the margin between the hyperplane and the classes, the better the classification. In SVR the objective is to find the function that is closest to the sample points. The linear regression function is given by (9).

$$f(x) = (w_1 x_1 + \cdots + w_d x_d) + b \tag{9}$$

Where $w_i \in \mathbb{R}, \forall i = 1, \dots, d$ and $b \in \mathbb{R}$.

Given a dataset of the form $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. The function is given by (10)

$$\frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i^*) \tag{10}$$

For $i = 1, 2, \dots, n$.

$$y_i - \big((w, x_1) + b\big) \leq \varepsilon + \xi_i \tag{11}$$

$$\big((w, x_1) + b\big) - y_i \leq \varepsilon + \xi_i^* \tag{12}$$

$$\xi_i \geq 0, \xi_i^* \geq 0 \tag{13}$$

C is a constant that determines the balance between f and the measure of tolerances at deviances greater than a ε. The variables ξ_i and ξ_i^* control the error by the regression function when approximating the i-th sample [21], [34, 35].

III. METHODOLOGY

Due to the learning methods used in this work are supervised, the data must be labelled, i.e., each input pattern must have its corresponding desired output. The proposed methodology is summarized in the diagram in Fig 2. Therefore, data is generated, performing inverter simulations, making various sweeps in the V_T, R_L, *I0* and *MMU*. The dataset is rescaled so that it is easy to handle by the NN and the SVR, although it is not necessary for DT or RF.

The next step is to choose the parameters and configurations of the predictor models, for example, for the one NN the number of hidden layers and their activation function are specified. The steps of the proposed method are explained in detail below. For each trained model, the MSE and the coefficient of determination (R^2) are the main metrics for assessing performance, they are stored, and different combinations of hyper-parameters are tested to find the best model for each method. Once the best model of the different learning methods has been found, evaluation curves (20% of the samples) are randomly taken to simulate and graph the behavior of the IC with the expected and extracted parameters, to analyze the difference between them. Finally, each method is tested with samples outside of the knowledge learned.

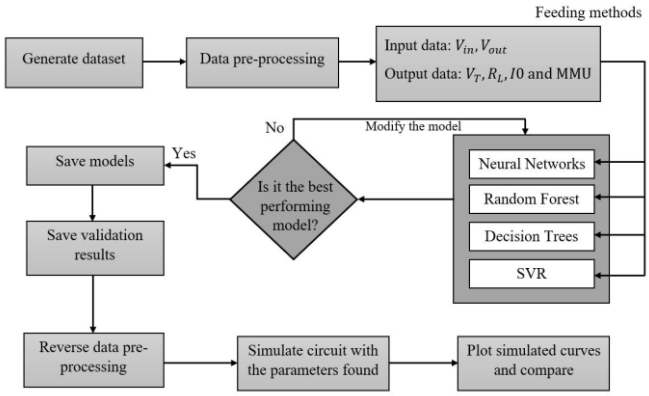

Fig. 2. General diagram of the proposed methodology.

A. Obtaining the Data

Supervised learning methods require a sufficiently large data set to build well-performing prediction models. In the case of experimental measurements of electronic devices, it is difficult to obtain large amounts of data, since the data must belong to the same device. If measurements from different devices under different conditions are used, the prediction models will not be accurate. To overcome this problem, a data set was generated from AIM-Spice simulations.

The simulations were made using a TFT with geometric parameters W=100um, L=10um and T_{ox}=10 nm with sweeps from V_T=1V to 5V at a rate of 1V, for each value of V_T a sweep was made in R_L with 10kΩ, 39kΩ, 100kΩ, 510kΩ and 1MΩ. For each value of R_L, a sweep was made in *I0* with 6A/m, 1kA/m, 5kA/m and 10kA/m. Those configurations were made 3 times for each value of *MMU*=0.8, 1.5 and 3. A V_{DD} was handled from 3 to 6V, to increase samples and knowledge, obtaining 1200 different curves.

Each curve has 102 points defined by V_{in} due to the sweep from -10 to 10V with 0.2V increments. This range allows to have complete transition regions for V_T and R_L, and in negative V_{in} the effect of *I0* is covered. For each value of V_{in} there is a different value of V_{out} and I_D, with a constant value of V_{DD}. Therefore, each entry is a vector of 304 elements. The data were normalized using a normal distribution.

B. Design and Training of Learning Methods

The methods mentioned in section II receive V_{in}, V_{DD}, I_D y V_{out}, as inputs variables, and V_T, R_L, *I0* and *MMU* as outputs variables. The methods must find the patterns that allow them to identify the parameters that each training sample has.

In the case of NNs, the best combination of hyper-parameters, as well as the number of layers and neurons, was found by a grid search, in which different values of learning rates, activation functions, number of layers and neurons are tested. Each training result is stored and, at the end, the one with the lowest MSE and the highest R^2 is selected as the best model.

For the DT, tree pruning was used, where the growth of the tree is not stopped, and then pruned, leaving a sufficiently robust tree that provides a low error, making use of cost complexity pruning.

For RF and SVR a grid search was also used to find the hyper-parameters that provide the best performance. The tables with the search ranges used for each method are shown in section C Experiments.

C. Experiments

The learning methods were implemented in the Jupiter environment, provided by Google Colaboratory, using the Python language. Currently there are specialized libraries in machine learning that allow the application of learning models in a simple way. In this work, the Scikit-learn library [36] was used for the application of RF, DT and SVR. The NN were applied using Keras from Tensorflow [37].

The values used for the parameters of each model during the grid search are presented below. The best model of each method and its performance can be found in section IV below.

In Table II, are the values in the different hyper-parameters of the NN. The number of epochs tested ranged from 500 to 5000.

TABLE II
HYPER-PARAMETER VALUES IN NN

Layer	Number of neurons	Activation function	Learning rate
1	32, 64, 128, 256, 512	relu, linear, sigmoid	0.1, 0.001, 0.0001
2	32, 64, 128, 256	relu, linear, sigmoid	0.1, 0.001, 0.0001
3	32, 64	relu, linear, sigmoid	0.1, 0.001, 0.0001

Table III shows the different values taken by the hyperparameters in RF. Where from 10 trees up to 150 trees, different numbers of features and depths up to 20 were tested. Table IV shows the different values taken by the hyperparameters in the DT. As it can be observed, there was no limit to the tree growth and there was no limit in final nodes, because at the end pruning was applied to reduce complexity and overtraining.

TABLE III
HYPER-PARAMETER VALUES IN RF

Number of trees	Max features	Max depth
10, 11, 12, …, 150	5, 7, 28, 45, 90	None, 3, 10, 20

Finally, Table V presents the different values tested for the SVR hyperparameters during the grid search.

TABLE IV
HYPER-PARAMETER VALUES IN DT

CCP Alpha	Min samples split	Max leaf nodes	Max depth
0, 1, 3, 5, 10	2, 4	None	None

TABLE V
HYPER-PARAMETER VALUES IN SVR

Kernel	C	Gamma
Poly	0.1, 1, 2, 3, 4	0.1, 1, 2, 3, 4
Linear	0.1, 1, 2, 3, 4	0.1, 1, 2, 3, 4

The processing time varies depending on the extent of the grid search, the amount of data and the capacity of the computer equipment. Considering the GPUs available in Colab [38], the training time (minutes) for NN was 141.5, for RF was 1.76, for DT was 2.86 and SVR was 18.98. Thanks to today's computing power, processing time is minimal.

The simulation of the parameters extracted by the different learning models was performed in two parts. The first consisted of taking directly the data from the validation set, calculating their R^2, MSE and additionally calculating the error percentage in (14) that each extracted parameter calculated with respect to the expected one.

$$\%Error = \left|\frac{parameter_{exp} - parameter_{ext}}{parameter_{exp}}\right| \times 100 \qquad (14)$$

The second way to evaluate the learning models was to feed them with completely new curves, but within the range of the knowledge acquired during training. These curves have parameters with intermediate values of those to be used in the learning process. For example, the models learned to identify values of V_T=1, 2, 3, 4 and 5V. The new curves have random values such as V_T=1.3, 4.5V and so on, this allows to approximate the tests to what it would be in the real world, with samples different from the knowledge learned in training. Finally, they are simulated and plotted to visually compare the differences between the actual curves and the ones used by the parameters extracted by the learning models. For these test samples the percentage error was calculated using the current of the curves. Between the simulated curve with the extracted parameters and the curve from which the extraction was done.

IV. Discussion of Results

A. Extraction in Inverter Circuit

This section shows the structures of the best performing methods, as well as their evaluation. It also shows graphically the comparison of the performance of the IC using the real parameters versus the extracted ones.

The NN architecture with the best performance was using 2 hidden layers, 256 and 64 neurons respectively, both with a "relu" activation function, a "linear" function in the output layer, a learning rate of 1×10^{-2} and 2000 training epochs.

For RF the best model had 146 trees, with a maximum of 7 characteristics and no maximum depth. In DT the best CCP AlPHA was zero, resulting in a tree depth of 14, with 623 terminal nodes after the pruning technique.

In the case of SVRs, one SVR was trained for each parameter to be extracted, i.e., there are 4 SVRs. For the SVR belonging to V_T there is a polynomial kernel, a *C*=1.0 and *gamma*=0.1. For the SVR belonging to R_L there is a polynomial kernel, *C*=1.0 and *gamma*=0.3. For the SVR belonging to *I0* and *MMU* there is a polynomial kernel, *C*=5.0 and *gamma*=0.4.

The tables (VI-IX) present the evaluation of the results of each model for the validation data. The tables have the R^2, *MSE* and percent error per parameter and an overall mean. Table VI shows the result of the NN where VT, RL and MMU have the best performance, with I0 being the parameter with the highest error.

TABLE VI
EVALUATION OF NN RESULTS

Parameter	R^2	MSE	Error (%)
VT	0.9991	2.470×10^{-2}	2.27
RL	0.999	2.690×10^{-2}	2.86
IO	0.9596	1.710×10^{-1}	16.45
MMU	0.9995	1.710×10^{-2}	2.57
Mean	0.98	6.003×10^{-2}	6.04

The NN obtains the best score because in three of the four parameters there is an R^2 greater than 0.99, which allows having the lowest percentage of error, except in *I0*, which had an error higher than 16%.

Table VII shows that RF has the best performance in the *MMU* parameter, followed by V_T and R_L, the *I0* parameter had the lowest performance with R^2=0.82 with an error percentage of 62%. Similar to NN, RF presents a high score in three parameters, *I0* being the one with the lowest R^2=0.82 less than 0.82, this causes the error percentage of this parameter to be 62.9% which affects the mean error percentage.

TABLE VII
EVALUATION OF RF RESULTS

Parameter	R^2	MSE	Error (%)
VT	0.9798	1.218×10^{-1}	8.90
RL	0.9688	1.590×10^{-1}	14.90
IO	0.8273	3.547×10^{-1}	62.96
MMU	0.9912	7.776×10^{-2}	11.10
Mean	0.94	1.766×10^{-1}	24.46

Table VIII shows the results obtained by SVR, which was the method with the third best performance. The parameters for which the estimation is best are V_T with R^2=0.96, *MMU* with R^2=0.95, followed by R_L with R^2=0.86. The parameter with the worst fit is *I0* with R^2=0.74.

TABLE VIII
EVALUATION OF SVR RESULTS

Parameter	R^2	MSE	Error (%)
VT	0.9698	1.488×10^{-1}	12.49
RL	0.861	3.213×10^{-1}	25.78
IO	0.745	4.310×10^{-1}	50.52
MMU	0.9578	1.704×10^{-1}	26.06
Mean	0.8834	2.679×10^{-1}	28.71

Table IX shows the DT results, this was the method with the lowest performance with an average R^2=0.84 and with an error percentage of 29.7%. of the parameters individually, it can be stated that the average performance is low due to the extraction of the *I0* parameter, which obtains an R^2=0.51, the other three parameters have a good score above 0.93. The same happens with the error percentage, the mean is affected by 92.5% in *I0*, but in the other parameters the highest error is in *MMU* of 12.5%.

TABLE IX
EVALUATION OF DT RESULTS

Parameter	R^2	MSE	Error (%)
VT	0.9728	1.414×10^{-1}	4.63
RL	0.9366	2.168×10^{-1}	9.14
IO	0.5151	5.943×10^{-1}	92.56
MMU	0.9578	1.705×10^{-1}	12.55
Mean	0.84557	2.808×10^{-1}	29.72

Fig. 3 shows the behaviour of the IC using the parameters extracted from one of the samples of the validation dataset. Fig. 4 presents the IC parameters extracted from one of the test curves, which was not learned during the training process (with parameter values intermediate to the learned ones).

Fig. 3 presents a curve of the validation set together with the curves modelled with the parameters extracted by the different learning methods. The percentage error (by 14) was calculated using the output voltages, where NN obtained 0.87%, RF obtained 4.28%, DT obtained 0.09% and SVR obtained 85.7%. Fig. 4 presents a test IC curve, with which the trained methods were tested to identify sample parameters that are not part of the training and validation set. NN had a percentage error of 2.59%, RF had 11.73%, DT had 25.57% and SVR had 13.53%.

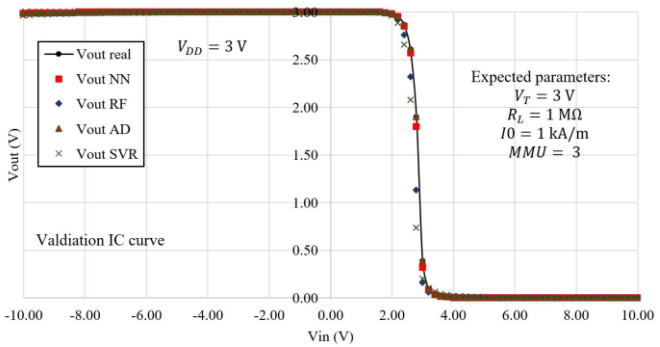

Fig. 3. Transfer curve of the IC, randomly selected from the validation samples.

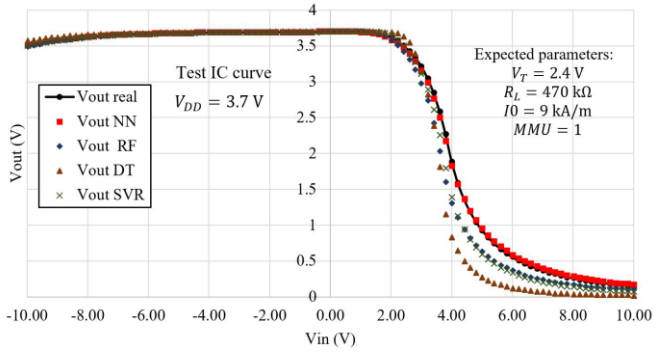

Fig. 4. Test curve with intermediate values in the parameters of the learned range.

The difference in the results of the learning methods is due to the training process performed by each of them. For both curves, a good agreement and fit of the curves with the extracted parameters is observed, in the first one, the DT fits the transition region perfectly, followed by the NN. Although the DTs obtain good adjustments in the validation stage, the DTs reduce their performance in the testing stage, since measurements with intermediate parameter values (with decimals) were used. The training and validation were performed with integer values and the DTs have the disadvantage of not being able to extrapolate, for this reason the performance decreased with test measurements.

During the training it is evident that the *I0* parameter is difficult to identify for the different methods, because this parameter affects a small region of the curve (negative voltage) and its effect is weak compared to the other parameters. So *I0* being a parameter that weakly affects the curve, it does not provide strong information to the methods to be identifiable. Even if the error in *I0* is large, the extraction still works because the effect it has is weak, that is, the change in the curve of an *I0*=100 A/m is almost the same as an *I0*=200 A/m.

One of the reasons why learning methods are not compared to analytical extraction is precisely because of the limitations when a transistor is part of a circuit. The analytical extraction of R_L, *MMU* and *I0* from the model defined in (1) cannot be performed independently. However, the parameter V_T can be found as follows, although it is not sufficient to model the full transferential curve:

$$I_{DS-sat} = \mu_{FET} C_{OX} \frac{W}{L} \frac{(V_{GS}-V_T)^2 \alpha_{sat}}{2} \quad (15)$$

Assuming, for simplicity, that α= 1 and considering that $V_{GS} = V_{in}$ the equation would be expressed as:

$$I_{DS-sat} = \mu_{FET} C_{OX} \frac{W}{L} \frac{(V_{in}-V_T)^2}{2} \quad (16)$$

The current flowing through the load resistance is defined by Ohm's law, that is:

$$I_R = \frac{V_R}{R_L} \quad (17)$$

The voltage across the load resistor would be equal to $V_R = V_{DD}-V_{out}$, so the expression for the current at the load resistance is:

$$I_R = \frac{V_{DD}-V_{out}}{R_L} \quad (18)$$

As the load resistance and the transistor are connected in parallel, the current flowing through both devices is the same, i.e.:

$$I_R = I_{DS-sat} \quad (19)$$

$$\frac{V_{DD}-V_{OUT}}{R_L} = \frac{\mu_{FET} C_{OX}}{2} \frac{W}{L} (V_{IN}-V_T)^2 \quad (20)$$

Clearing $V_{DD}-V_{OUT}$:

$$V_{DD}-V_{OUT} = \frac{R_L \mu_{FET} C_{OX}}{2} \frac{W}{L} (V_{IN}-V_T)^2 \quad (21)$$

$$\sqrt{V_{DD}-V_{OUT}} = \sqrt{\frac{R_L \mu_{FET} C_{OX}}{2} \frac{W}{L}} (V_{IN}-V_T) \quad (22)$$

This is the expression for a straight line, where the slope of the line (m) is equal to:

$$m = \sqrt{\frac{R_L \mu_{FET} C_{OX}}{2} \frac{W}{L}} \quad (23)$$

As can be seen, the threshold voltage can be extracted from the intercept of the line with the x-axis. That is, when VIN is equal to V_T, the value of $\sqrt{V_{DD}-V_{OUT}}$ will be equal to zero.

TABLE X
COMPARISON OF VT EXTRACTION WITH OTHER WORKS

Work	V_T extracted (V)	V_T expected (V)	%Error
This work	2.26	2.4	*5.83*
[8]	-1.07	-1.2	10.83
[10]	-11.2	1.0	1.22×10^3

Table X shows two parameter extraction studies, which present extraction values and expected values, with which it is possible to calculate their error percentage. Although they are not comparable with this work since they use other conditions technology and data, the table allows to observe that this research obtained good results.

B. Extraction in NMOS Transistor

This subsection presents the results of parameter extraction in experimental measurements of NMOS transistors with the proposed methodology. The transistors were manufactured by the Centre for Engineering and Industrial Development. The

methodology was applied for the extraction of surface mobility (*UO*), threshold voltage (V_T) and surface resistance (R_S) parameters. The parameters were extracted from transistors with the following dimensions. Device 1 (D1) with a *W*=100um, an *L*=8um and a T_{ox}=27nm; device 2 (D2) with a *W*=100um, an *L*=16um and a T_{ox}=32nm.

Because the DTs cannot extrapolate what is learned, only the NN, RF and SVR methods were trained in these tests. The training set was 540 samples, ranging from 0 to 500 Ω for R_S, with 100Ω increments, for V_T had a sweep from -1.5 to 3V with 0.5V increments and for *UO* we had a sweep from 100 to 900 cm^2/Vs. These learning ranges were taken due to the NMOS standard, for V_T, the range was established by experimental measurements (positive V_{GS} conduction). The tunning of the hyperparameters of the learning methods was performed with a grid search with the same range as in III. C, due to the length of the manuscript, in order not to exceed the allowed limit, the grid search tables are not repeated for this experiment. Table XI presents the best hyperparameters of the learning methods with their performance in the validation samples of the two different devices.

After the training of the learning methods, they are fed with the experimental NMOS measurements, to perform the extraction and use the extracted values to simulate the NMOS and observe if the behavior of the devices can be modeled.

TABLE XI
HYPER-PARAMETERS AND PERFORMANCE OF LEARNING METHODS

Method	Hyper-parameters	R^2 in D1	R^2 in D2
NN	2 hidden layers, 256 and 64 neurons, "relu" function in hidden layers and "linear" in output layer	0.99	0.87
RF	Depth of 15 (D1) and 20 (D2), 250 leaf nodes, 2 samples for slit and 250 estimators	0.77	0.75
SVR	Kernel RBF, C=4, gamma "scale" (D1) and 3 (D2)	0.74	0.84

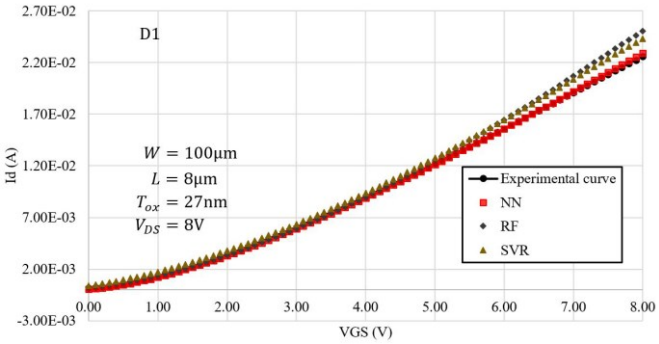

Fig. 5. Modelling of D1 with extracted parameters.

Fig. 5 shows the transferential curve of D1 and the curves modelled with the extracted parameters. Being a newly fabricated device, its parameters are unknown, NN extracted UO=1475cm^2/Vs, V_T=-0.22 V and R_S=166Ω, RF extracted 747 cm^2/Vs, -0.66 V and 86Ω respectively, and SVR extracted 802 cm2/Vs, -0.85V and 108Ω respectively. The percentage of error of each modelling is 0.43% for NN, 5.03% for RF and 6.65% for SVR. Fig. 6 shows the transferential curve of D2 and the curves modelled with the extracted parameters. NN extracted *UO*=333cm^2/Vs, V_T=0.6V and R_S=530Ω, RF extracted 294 cm^2/Vs, 0.44 V and 321Ω respectively, and SVR extracted 299cm2/Vs, 0.36V and 393Ω respectively. The percentage of error of each modelling is 0.44% for NN, 3.47% for RF and 0.57% for SVR

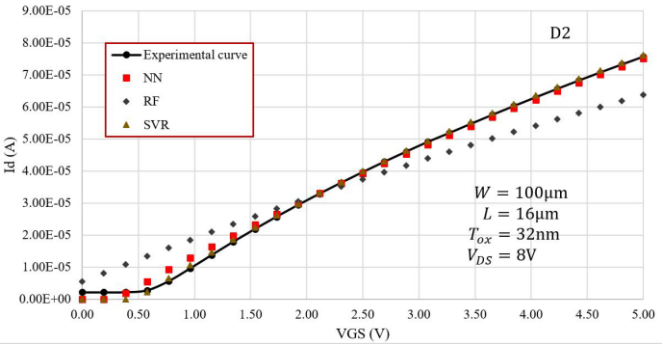

Fig. 6. Modelling of D1 with extracted parameters.

Fig. 5 shows the transferential curve of D1 and the curves modelled with the extracted parameters. Being a newly fabricated device, its parameters are unknown, NN extracted UO=1475cm^2/Vs, V_T= -0.22 V and R_S=166Ω, RF extracted 747 cm^2/Vs, -0.66 V and 86Ω respectively, and SVR extracted 802 cm2/Vs, -0.85V and 108Ω respectively. The percentage of error of each modelling is 0.43% for NN, 5.03% for RF and 6.65% for SVR. Fig. 6 shows the transferential curve of D2 and the curves modelled with the extracted parameters. NN extracted *UO*=333cm^2/Vs, V_T=0.6V and R_S=530Ω, RF extracted 294 cm^2/Vs, 0.44 V and 321Ω respectively, and SVR extracted 299cm2/Vs, 0.36V and 393Ω respectively. The percentage of error of each modelling is 0.44% for NN, 3.47% for RF and 0.57% for SVR.

V. CONCLUSION

In this study, supervised learning methods for parameter extraction in a resistive load inverter circuit were presented. Due to the early use of these methods for parameter extraction in electronic devices, it was strategically decided to start with such an inverter model, but also, the method was applied to the extraction of parameters in experimental measurements of NMOS-type transistors.

Using Neural Networks, Random Forest, Decision Trees and Support Vector Regression. Neural Networks were identified as having the best performance compared to the other methods employed, its evaluation presented an average R^2=0.98 with a percentage of error of 6.04%, and in test samples a percentage error of 15.85% was obtained. However, Random Forest performs a good extraction of parameters with an R^2=0.94, and a percentage of error of 24.64%, in tests, obtained an R^2=0.75 and a percentage error of 42.9%. Decision Trees with an R^2=0.84 and a percentage error of 29.7%, in tests R^2=0.34 and a percentage error of 78.2%. Support Vector Regression with R^2=0.88 and a percentage of error 28.7% in tests an R^2=0.38 and a percentage error of 50.3%.

The proposed method was applied for extraction in NMOS transistors, and real measurements were used for testing, obtaining minimum error percentages of 0.43% and 0.44 with Neural Networks, 6.65% and 3.47% with Random Forest, 6.65% and 0.57% with Support Vector Regression. This proved that the methods used can learn from simulated I-V curves and use their knowledge to extract parameters from real measurements. As future work, it is intended to increase the parameters to be extracted and apply the method in different technologies.

Data availability

The datasets generated during the current study are available from corresponding author.

REFERENCES

[1] A. Ortiz-Conde, F. J. García-Sánchez, J. Muci, A. Terán Barrios, J. J. Liou, and C.-S. Ho, "Revisiting MOSFET threshold voltage extraction methods," *Microelectron. Reliab*., vol. 53, no. 1, pp. 90–104, Jan. 2013, doi: 10.1016/j.microrel.2012.09.015.

[2] A. Cerdeira, M. Estrada, R. García, A. Ortiz-Conde, and F. J. García Sánchez, "New procedure for the extraction of basic a-Si:H TFT model parameters in the linear and saturation regions," *Solid. State. Electron*., vol. 45, no. 7, pp. 1077–1080, Jul. 2001, doi: 10.1016/S0038-1101(01)00143-5.

[3] C. Tanaka and K. Ikeda, "Comprehensive investigation on parameter extraction methodology for short channel amorphous-InGaZnO thin-film transistors," *in 2018 IEEE International Conference on Microelectronic Test Structures* (ICMTS), Mar. 2018, pp. 23–26, doi: 10.1109/ICMTS.2018.8383756.

[4] Y. H. Hu and S. Pan, "SaPOSM: an optimization method applied to parameter extraction of MOSFET models," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 12, no. 10, pp. 1481–1487, 1993, doi: 10.1109/43.256940.

[5] P. Moreno, R. Picos, M. Roca, E. Garcia-Moreno, B. Iniguez, and M. Estrada, "Parameter Extraction Method using Genetic Algorithms for an Improved OTFT Compact Model," *in 2007 Spanish Conference on Electron Devices*, Jan. 2007, pp. 64–67, doi: 10.1109/SCED.2007.383996.

[6] N. Akkan, M. Altun, and H. Sedef, "Parameter Extraction Method Using Hybrid Artificial Bee Colony Algorithm for an OFET Compact Model," *in 2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design* (SMACD), Jul. 2018, pp. 105–108, doi: 10.1109/SMACD.2018.8434861.

[7] I. Benacer and Z. Dibi, "Extracting parameters of OFET before and after threshold voltage using genetic algorithms," *Int. J. Autom. Comput*., vol. 13, no. 4, pp. 382–391, Aug. 2016, doi: 10.1007/s11633-015-0918-6.

[8] N. Akkan, M. Altun, and H. Sedef, "Modeling and Parameter Extraction of OFET Compact Models Using Metaheuristics-Based Approach," *IEEE Access*, vol. 7, pp. 180438–180450, 2019, doi: 10.1109/ACCESS.2019.2959474.

[9] T. Bendib and F. Djeffal, "Electrical Performance Optimization of Nanoscale Double-Gate MOSFETs Using Multiobjective Genetic Algorithms," *IEEE Trans. Electron Devices*, vol. 58, no. 11, pp. 3743–3750, Nov. 2011, doi: 10.1109/TED.2011.2163820.

[10] E. Gadjeva and M. Hristov, "Computer-aided extraction of small-signal model parameters of heterojunction bipolar transistors," *in Proceedings of the Joint* INDS'11 & ISTET'11, Jul. 2011, pp. 1–6, doi: 10.1109/INDS.2011.6024817.

[11] A. Huang, Z. Zhong, Y. Guo, and W. Wu, "A novel extrinsic parameter extraction method for the technology independent modeling of transistors," *in 2015 Asia-Pacific Microwave Conference* (APMC), Dec. 2015, pp. 1–2, doi: 10.1109/APMC.2015.7412946.

[12] A. Jarndal, "Combined genetic algorithm and neural network technique for transistor modeling," *in 2015 International Conference on Communications, Signal Processing, and their Applications* (ICCSPA'15), Feb. 2015, pp. 1–4, doi: 10.1109/ICCSPA.2015.7081300.

[13] S. I. Sayed, M. M. Abutaleb, and Z. B. Nossair, "Improved CNFET performance based on genetic algorithm parameters optimization," *in 2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference* (IEMCON), Oct. 2017, pp. 181–184, doi: 10.1109/IEMCON.2017.8117185.

[14] S. Moparthi, P. K. Tiwari, and G. K. Saramekala, "Genetic algorithm-based threshold voltage prediction of SOI JLT using multi-variable nonlinear regression," *in 2021 International Symposium on Devices, Circuits and Systems* (ISDCS), Mar. 2021, pp. 1–4, doi: 10.1109/ISDCS52006.2021.9397911.

[15] R. Picos, O. Calvo, B. Iñiguez, E. García-Moreno, R. García, and M. Estrada, "Optimized parameter extraction using fuzzy logic," *Solid. State. Electron*., vol. 51, no. 5, pp. 683–690, May 2007, doi: 10.1016/j.sse.2007.02.031.

[16] K. R. Chowdhary, Fundamentals of artificial intelligence. New Delhi: Springer India, 2020.

[17] J. J. Grefenstette, Ed., Genetic Algorithms for Machine Learning. Boston, MA: Springer US, 1994.

[18] S. Bandyopadhyay and S. Kumar, Classification and Learning Using Genetic Algorithms. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[19] L. Zhang, Y. Pan, X. Wu, and M. J. Skibniewski, Introduction to Artificial Intelligence, vol. 163. London: Springer London, 2021.

[20] R. H. Griffin, D. E. Root, J. Xu, A. Dadvand, T. Y. Chu, and Y. Tao, "Artificial Neural Network Modelling and Simulation of Organic Field Effect Transistors and Circuits," in 2019 *IEEE International Flexible Electronics Technology Conferenc*e, IFETC 2019, Aug. 2019, pp. 1–5, doi: 10.1109/IFETC46817.2019.9073711.

[21] J. Cai, J. King, C. Yu, J. Liu, and L. Sun, "Support Vector Regression-Based Behavioral Modeling Technique for RF Power Transistors," *IEEE Microw. Wirel. Component*s Lett., vol. 28, no. 5, pp. 428–430, May 2018, doi: 10.1109/LMWC.2018.2819427.

[22] J. Cai, C. Yu, J. Liu, and L. Sun, "Large Signal Behavioral Model of RF Transistor Using Least Square Support Vector Machine," *in 2020 International Conference on Microwave and Millimeter Wave Technology* (ICMMT), Sep. 2020, pp. 1–3, doi: 10.1109/ICMMT49418.2020.9386908.

[23] L. Puggini, J. Doyle, and S. McLoone, "Fault Detection using Random Forest Similarity Distance," IFAC-PapersOnLine, vol. 48, no. 21, pp. 583–588, 2015, doi: 10.1016/j.ifacol.2015.09.589.

[24] A. Ojha, Y. S. Chauhan, and N. R. Mohapatra, "A Channel Stress-Profile-Based Compact Model for Threshold Voltage Prediction of Uniaxial Strained HKMG nMOS Transistors," *IEEE J. Electron Devices Soc.*, vol. 4, no. 2, pp. 42–49, Mar. 2016, doi: 10.1109/JEDS.2016.2524536.

[25] C.-H. Shen, Y. Li, I.-H. Lo, P.-J. Lin, and S.-C. Chung, "Modeling temperature and bias stress effects on threshold voltage of a-Si:H TFTs for gate driver circuit simulation," *in 2011 International Conference on Simulation of Semiconductor Processes and Devices*, Sep. 2011, pp. 251–254, doi: 10.1109/SISPAD.2011.6035072.

[26] Berkeley, "Aim-Spice." California, [Online]. Available: http://www.aimspice.com/.

[27] B. M. Wilamowski and J. D. Irwin, The Industrial Electronics Handbook: Intelligent Systems, Second. CRC Press, 2011.

[28] A. B. Kock and T. Teräsvirta, "Forecasting performances of three automated modelling techniques during the economic crisis 2007–2009," Int. J. Forecast., vol. 30, no. 3, pp. 616–631, Jul. 2014, doi: 10.1016/j.ijforecast.2013.01.003.

[29] M. Hagan, H. Demuth, M. Beale, and O. De Jesus, Neural Network Design, Second. Ebook, 2014.

[30] S. Raschka and V. Mirjalili, Python Machine Learning, Second Edi. Birmingham: Packt Publishing Ltd., 2017.

[31] A. C. Muller and S. Guido, Introduction to Machine Learning with Python. United States of America: O'Reilly Media, 2016.

[32] G. James, D. Witten, T. Hastie, and R. Tibshirani, An Introduction to Statistical Learning, vol. 103. New York, NY: Springer New York, 2013.

[33] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn*., vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[34] M. Awad and R. Khanna, "Support Vector Regression," *in Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers, Berkeley*, CA: Apress, 2015, pp. 67–80.

[35] J. J. Martín, "Support Vector Regression: propiedades y aplicaciones," Universidad de Sevilla, 2016.

[36] F. Pendragosa et al., "Scikit-learn: Machine Learning in {P}ython," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011, [Online]. Available: https://scikit-learn.org/stable/about.html#citing-scikit-learn.

[37] M. Abadi et al., "{TensorFlow}: Large-Scale Machine Learning on Heterogeneous Systems," 2015. https://www.tensorflow.org/?hl=es-419 (accessed Sep. 28, 2021).
[35] Google, "Te damos la bienvenida a Colaboratory," https://colab.research.google.com/?hl=es (accessed Sep. 26, 2022).

Roberto Carlos Valdés García. He obtained the title of Computer Engineer and he obtained the Master's degree in Computer Science at the UAEM University Center of Ecatepec, Mexico in 2016 and 2019 respectively. He is currently a PhD student in Computer Science at UAEM University Center of Texcoco at the Autonomous University of the State of Mexico.

Farid Garcia Lamont. He obtained the title of Industrial Robotic Engineer from the ESIME Azcapotzalco of the IPN, Mexico in 2000. He obtained the degree of Master of Science in Automatic Control from the CINVESTAV-IPN, Mexico in 2004. He obtained the PhD of Computer Science from CINVESTAV-IPN, Mexico in 2010. He is currently a full-time professor at the University Center UAEM Texcoco, at the Autonomous University of State of Mexico.

Rodolfo Zolá García Lozano. He obtained the title of Electronics Engineer from the Technological Institute of Higher Studies of Ecatepec (TESE), Mexico in 1996. He obtained the PhD degree from CINVESTAV-IPN, Mexico in 2005. He is currently a full-time professor at the University Center UAEM Ecatepec at the Autonomous University of State of Mexico.

Asdrúbal López Chau. He obtained the title of Communications and Electronics Engineer from the ESIME of the IPN, Mexico in 1998. He obtained the degree of Master of Science in Computer Engineering from Center for Research in Computation, IPN, Mexico in 2004. He obtained the PhD of Computer Science from CINVESTAV-IPN, Mexico in 2013. He is currently a full-time professor at the Autonomous University of State of Mexico.

Rodolfo Sánchez Fraga. He obtained the degree of engineer in mechatronics in 2008, in 2013 he obtained the degree of master in computer engineering, and the degree of doctor in computer science in 2017. Having all his training at the National Polytechnic Institute of Mexico. He is currently a professor and researcher at the Center for Engineering and Industrial Development in Queretaro, Mexico.

Gustavo Lastra Medina. He obtained a degree in Electronic Engineering from the Autonomous University of Baja California in 2005. In 2010 he obtained a Master's degree in Materials Science and Engineering at the Sonora University and in 2014 he obtained a PhD degree in Materials Science and Engineering at the National Autonomous University of Mexico. He is currently a professor and researcher at the Center for Engineering and Industrial Development in Queretaro, Mexico.