

**Universidad Autónoma del Estado de México**  
**Centro Universitario UAEM Texcoco**  
**Doctorado en Ciencias de la Computación**

---



**T e s i s**

**Extracción de parámetros en curvas I-V de dispositivos tipo MOSFET utilizando  
métodos de aprendizaje supervisado**

**P R E S E N T A**

Roberto Carlos Valdés García

**Que para obtener el grado en:**

Doctor en Ciencias de la Computación

**Tutor académico:**

Dr. Farid García Lamont

**Tutores adjuntos:**

Dr. Rodolfo Zolá García Lozano

Dr. Asdrúbal López Chau

Texcoco, Estado de México

Agosto 2023



## CARTA DE CESIÓN DE DERECHOS DE AUTOR

El que suscribe Roberto Carlos Valdés García Autor del trabajo escrito de evaluación de grado en la opción de Tesis con el título “Extracción de parámetros en curvas I-V de dispositivos tipo MOSFET utilizando métodos de aprendizaje supervisado”, por medio de la presente con fundamento en lo supuesto en los artículos 5,18,24,25,27,30,32 y 148 de la Ley Federal de Derechos de Autor, así como los artículos 35, y 36 fracción II de la Ley de la Universidad Autónoma del Estado de México; manifiesto mi autoría y originalidad de la obra mencionada que se presentó en Universidad Autónoma del Estado de México CU Texcoco para ser evaluada con el fin de obtener el grado de Doctor en Ciencias de la Computación.

Así mismo expreso mi conformidad de ceder los derechos de reproducción, difusión y circulación de esta obra, en forma NO EXCLUSIVA, a la Universidad Autónoma del Estado de México, se podrá realizar a nivel nacional e internacional, de manera parcial o total a través de cualquier medio de información que sea susceptible para ello, en una o varias ocasiones, así como en cualquier soporte documental, todo ello siempre y cuando sus fines sean académicos, humanísticos, tecnológicos, históricos, artísticos, sociales, científicos u otra manifiesto de la cultura.

Entiendo que dicha cesión no genera obligación alguna para la Universidad Autónoma del Estado de México y que podrá a no ejercer los derechos cedidos.

Por lo que el autor da su consentimiento para la publicación de su trabajo escrito de evaluación profesional.

Se firma presente en Estado de México, a los 1 días del mes de agosto de 2023.

  
Roberto Carlos Valdés García

Nombre y firma de conformidad

# Índice

Índice de Figuras .....	VI
Índice de tablas .....	IX
Resumen .....	XI
Abstract.....	XII
Capítulo 1 Introducción .....	1
1.2. Hipótesis.....	3
1.3. Variables de estado .....	3
1.4. Objetivos .....	4
1.4.1. Objetivo general .....	4
1.4.2. Objetivos particulares .....	4
1.5. Metodología .....	4
Capítulo 2 Estado del arte.....	9
Capítulo 3 Marco teórico .....	32
3.1 Transistores .....	32
3.1.1. Tipos de transistores.....	32
3.1.2. Circuito inversor.....	35
3.2. Extracción analítica de $V_T$ en circuito inversor .....	37
3.3. Redes neuronales artificiales (RNAs) .....	39
3.3.1 Modelo de la neurona.....	40
3.3.2 Neurona con múltiples entradas .....	40
3.3.3 Red multicapa.....	41
3.3.4 Entrenamiento de las RNAs .....	42
3.4. Árboles de decisión .....	44
3.4.1. Árbol de clasificación .....	44
3.4.2 Árboles de regresión .....	45
3.4.3 Ensamblados de árboles de decisión .....	46
3.5 Support Vector Regression .....	47
Capítulo 4 Enfoque propuesto .....	50
4.1. Transistor NMOS .....	50
4.1.1. Análisis de parámetros y como afectan al NMOS .....	50
4.1.2. Extracción de 3 parámetros del NMOS .....	52

4.2. Transistor TFT IGZO.....	55
4.2.1. Análisis de parámetros y como afectan al TFT.....	55
4.2.2. Extracción de KP, $V_T$ y $R_C$ del TFT.....	58
4.3. Circuito de carga resistiva.....	60
4.3.1 Análisis de los parámetros y como afectan al circuito inversor.....	60
4.3.2. Extracción de 2 parámetros.....	72
4.3.3. Extracción de 4 parámetros utilizando diferentes métodos de aprendizaje automático.....	75
Capítulo 5. Resultados y discusión.....	78
5.1. Extracción de parámetros de un NMOS.....	78
5.1.1. Resultados de extracción en NMOS.....	78
5.2. Extracción de parámetros de un TFT IGZO.....	83
5.2.1. Extracción de KP y $V_T$ .....	83
5.2.2. Extracción de KP, $V_T$ y $R_C$ .....	92
5.3. Extracción de circuito de carga resistiva.....	95
5.3.1. Resultado de extracción de 2 parámetros.....	95
5.3.2 Resultados de extracción de 4 parámetros.....	100
Conclusiones.....	108
Trabajo Futuro.....	111
Agradecimientos.....	111
Artículos publicados.....	111
Referencias.....	113
Anexos.....	116
Anexo A.....	116
Anexo B.....	135

# Índice de Figuras

Figura 1. Metodología propuesta .....	8
Figura 2. Medidas de capacitancia .....	11
Figura 3. Resultados utilizando TLM .....	12
Figura 4. Comparación experimental y calculada.....	13
Figura 5. Diagrama de flujo del algoritmo ABC.....	17
Figura 6. Diagrama de flujo de dos AG .....	21
Figura 7. Sistema difuso para extracción .....	23
Figura 8. Funciones de fusificación .....	24
Figura 9. Comparación de curva experimental y la calculada .....	26
Figura 10. Comparación de curva experimental y la calculada dos OTFT.....	28
Figura 11. Características físicas y modelo de red.....	30
Figura 12. Diagrama general y curvas de un transistor.....	32
Figura 13. Diagrama de transistores pnp y npn .....	33
Figura 14. Símbolos de transistor MOSFET tipo P y N .....	34
Figura 15. Curva de transferencia del NMOS.....	34
Figura 16. Regiones de un NMOS .....	35
Figura 17. Curva de transferencia del inversor .....	36
Figura 18. Circuitos inversores más comunes.....	36
Figura 19. Modelo de una sola neurona y una entrada .....	40
Figura 20. Neurona con más de una entrada .....	41
Figura 21. Red Multicapa.....	42
Figura 22. Entrenamiento supervisado.....	43
Figura 23. Entrenamiento no supervisado.....	43
Figura 24. Ejemplo de árbol de decisión.....	44
Figura 25. Funcionamiento de la SVM.....	47
Figura 26. Funcionamiento de la SVR.....	48
Figura 27. NMOS cambiando VT.....	50
Figura 28. NMOS cambiando UO .....	51

Figura 29. NMOS cambiando $R_S$ .....	52
Figura 30. Comportamiento de TFT cambiando $V_T$ .....	56
Figura 31. Comportamiento de TFT cambiando $K_P$ .....	56
Figura 32. Comportamiento de TFT cambiando $R_C$ .....	57
Figura 33. $I_D$ cambiando $R_L$ y $V_T$ .....	61
Figura 34. $V_{out}$ cambiando $R_L$ y $V_T$ .....	62
Figura 35. $I_D$ y $V_{out}$ cambiando $V_{DD}$ .....	62
Figura 36. Estructura de un MOSFET canal n.....	63
Figura 37. $I_D$ y $V_{out}$ cambiando $T_{ox}$ .....	63
Figura 38. $I_D$ y $V_{out}$ cambiando $L$ .....	64
Figura 39. $I_D$ y $V_{out}$ cambiando $W$ .....	65
Figura 40. $I_D$ cambiando $I_0$ .....	66
Figura 41. $I_D$ de TFT cambiando $I_{00}$ .....	66
Figura 42. $I_D$ TFT cambiando $I_0$ e $I_{00}$ .....	67
Figura 43. $I_D$ TFT cambiando $I_0$ e $I_{00}$ subumbral .....	67
Figura 44. $I_D$ de TFT comparación entre $I_0$ e $I_{00}$ .....	68
Figura 45. $V_{out}$ inversor cambiando $I_0$ .....	69
Figura 46. $V_{out}$ inversor cambiando $I_0$ subumbral .....	69
Figura 47. $V_{out}$ inversor cambiando $I_{00}$ .....	69
Figura 48. $V_{out}$ inversor cambiando $I_{00}$ subumbral .....	69
Figura 49. $V_{out}$ inversor cambiando $I_0$ e $I_{00}$ .....	70
Figura 50. $V_{out}$ inversor cambiando $I_0$ e $I_{00}$ región subumbral .....	70
Figura 51. $V_{out}$ inversor comparación entre $I_0$ e $I_{00}$ región subumbral .....	70
Figura 52. $I_D$ de TFT cambiando $MMU$ .....	71
Figura 53. $I_D$ de inversor cambiando $MMU$ .....	72
Figura 54. $V_{out}$ de inversor cambiando $MMU$ .....	72
Figura 55. $R^2$ de métodos de aprendizaje T1 .....	80
Figura 56. Curva de validación T1 .....	81
Figura 57. $R^2$ de métodos de aprendizaje T2 .....	82
Figura 58. Curva de validación T2.....	83
Figura 59. $R^2$ de modelos de RN para extracción de $K_P$ y $V_T$ .....	84

Figura 60. $R^2$ obtenido por los modelos de RN por parámetro.....	85
Figura 61. Extracción y simulación de 2 parámetros del TFT IGZO .....	86
Figura 62. $R^2$ en entrenamiento de RNs (lineal) .....	87
Figura 63. $R^2$ en entrenamiento de RNs por parámetro (lineal).....	88
Figura 64. Simulación y modelado de TFT1 con parámetros extraídos .....	89
Figura 65. TFT1 y curvas con $R_c$ añadida .....	89
Figura 66. Simulación y modelado de curvas con parámetros extraídos.....	90
Figura 67. Simulación y modelado de TFTs al no depender de W y L .....	92
Figura 68. $R^2$ de cada parámetro obtenido por los métodos de aprendizaje .....	93
Figura 69. Simulación y modelado de TFT usando tres parámetros extraídos.....	94
Figura 70. Simulación de inversor curvas 1 y 2.....	97
Figura 71. Simulación de inversor curvas 3 y 4.....	98
Figura 72. Curva 1 modificada cuatro veces en W y L.....	100
Figura 73. Curva 2 modificada cuatro veces en W y L.....	100
Figura 74. Curva 3 modificada cuatro veces en W y L.....	100
Figura 75. $R^2$ de los métodos y por parámetros .....	103
Figura 76. Porcentaje de error de los métodos.....	103
Figura 77. Curva 1 de validación de CI con parámetros extraídos .....	104
Figura 78. Curva 2 de validación de CI con parámetros extraídos .....	105
Figura 79. Curva 1 de prueba de CI con parámetros extraídos.....	106
Figura 80. Curva 2 de prueba de CI con parámetros extraídos.....	107

# Índice de tablas

Tabla 1. Resultados de dos circuitos, medidos y teóricos.....	10
Tabla 2. Resultados de extracción.....	15
Tabla 3. Parámetros de AG.....	16
Tabla 4. Ejemplo de resultados.....	16
Tabla 5. Resultados de la extracción de parámetros.....	17
Tabla 6. Rango de datos.....	18
Tabla 7. Resultados de dos tipos de AG.....	22
Tabla 8. Resultados de extracción con LD.....	26
Tabla 9. Parámetros extraídos en T1.....	28
Tabla 10. Parámetros extraídos en T2.....	29
Tabla 11. Porcentajes de error de obtenidos en [21].....	30
Tabla 12. Rangos en los parámetros del NMOS.....	52
Tabla 13. Dimensiones de los dispositivos NMOS.....	53
Tabla 14. Search grid para RN del NMOS.....	54
Tabla 15. Search grid para RF del NMOS.....	54
Tabla 16. Search grid para SVR del NMOS.....	54
Tabla 17. Rango de parámetros para la extracción de $K_P$ y $V_T$ .....	58
Tabla 18. Rangos de parámetros para la extracción de $K_P$ , $V_T$ y $R_C$ .....	58
Tabla 19. Dimensiones de los dispositivos TFTs.....	59
Tabla 20. Search grid para RN del NMOS.....	59
Tabla 21. Search grid para RF del NMOS.....	59
Tabla 22. Search grid para SVR del NMOS.....	60
Tabla 23. Modelos de red entrenados.....	75
Tabla 24. Valores para cada parámetro.....	76
Tabla 25. Valores en hiperparámetros de RN.....	77
Tabla 26. Valores en hiperparámetros de AD.....	77
Tabla 27. Valores en hiperparámetros de RF.....	77
Tabla 28. Valores en hiperparámetros de SVR.....	77



Tabla 29. Características de corriente eléctrica eliminadas .....	78
Tabla 30. $R^2$ y MSE de RN para T1 .....	79
Tabla 31. $R^2$ y MSE de RF para T1 .....	79
Tabla 32. $R^2$ y MSE de SVR para T1.....	80
Tabla 33. $R^2$ y MSE de RN para T2.....	81
Tabla 34. $R^2$ y MSE de RF para T2 .....	81
Tabla 35. $R^2$ y MSE de SVR para T2.....	82
Tabla 36. Desempeño de RNs en entrenamiento para TFTs.....	84
Tabla 37. Desempeño de RNs por parámetro .....	84
Tabla 38. Parámetros extraídos en mediciones fijas de TFTs .....	86
Tabla 39. Desempeño de RNs en entrenamiento para TFTs (lineal) .....	87
Tabla 40. Parámetros extraídos de mediciones de TFTs (lineal) .....	88
Tabla 41. Parámetros extraídos de curvas con $R_C$ añadida .....	90
Tabla 42. Desempeño en entrenamiento en no dependencia de parámetros geométricos .....	91
.....	
Tabla 43. Parámetros extraídos en TFTs al no depender de W y L .....	91
Tabla 44. Desempeño general para la extracción de tres parámetros del TFT .....	93
Tabla 45. $R^2$ obtenido por métodos aprendizaje por parámetro.....	93
Tabla 46. Parámetros extraídos de TFTs (tres parámetros) .....	94
Tabla 47. Resultados en validación por tipo de normalización .....	95
Tabla 48. Extracción de $R_L$ .....	96
Tabla 49. Extracción de $V_T$ .....	96
Tabla 50. Porcentaje de error en extracción de $R_L$ .....	97
Tabla 51. Porcentaje de error de extracción en $V_T$ .....	97
Tabla 52. Curva 1 modificada cuatro veces en Wy L.....	98
Tabla 53. Curva 2 modificada cuatro veces en W y L.....	99
Tabla 54. Curva 3 modificada cuatro veces en W y L.....	99
Tabla 55. Evaluación de resultados de RN .....	101
Tabla 56. Evaluación de resultados de RF .....	101
Tabla 57. Evaluación de resultados de AD .....	102
Tabla 58. Evaluación de resultados de SVR .....	102

# Resumen

En este trabajo se propone utilizar Redes Neuronales Artificiales, Árboles de Decisión, Random Forest y Support Vector Regression, que son métodos de aprendizaje automático propios de la inteligencia artificial, para identificar los parámetros/características de dispositivos electrónicos con el objetivo de caracterizarlos y así modelar el comportamiento de dichos dispositivos en un simulador electrónico con exactitud.

Los diferentes modelos de inteligencia artificial se validaron al realizar varios experimentos en diferentes condiciones. Se comenzó a extraer 2 y después 4 parámetros de un circuito inversor de carga resistiva, el cual estaba constituido por un transistor de película delgada de silicio policristalino (TFT Poly-Si). Después se realizó la extracción de 3 parámetros en transistores tipo NMOS de manera individual y se llevaron a cabo pruebas de extracción utilizando mediciones físicas. Finalmente se realizó la extracción de tres parámetros en transistores tipo TFT de India Galio y Oxido de Zinc (IGZO), con los cuales también se hicieron pruebas de extracción con mediciones físicas.

En las tres condiciones de experimentación para realizar la extracción de parámetros se realizó básicamente lo siguiente: generar un conjunto de entrenamiento destinado al dispositivo de interés, usando un software de simulación eléctrico, con el cual se realizaron una serie de simulaciones donde se configuran los parámetros de interés. Las curvas I-V (entradas) resultantes se almacenan y se registran los parámetros (salidas) usados durante cada simulación. Con estas muestras de curvas I-V se entrenan los métodos de aprendizaje, en esta etapa aprenden a identificar los patrones y características de las curvas, que hacen posible la predicción de los parámetros que corresponden a cada una de ellas. Después del entrenamiento, se almacena el modelo, el cual está listo para recibir nuevas muestras de entrada y hacer una extracción/predicción. En esta etapa es posible usar las mediciones físicas de los dispositivos para probar a los métodos con datos reales e identificar al más exacto.

Con el método descrito en el párrafo anterior se probó que efectivamente los modelos de aprendizaje supervisado pueden ser utilizados para realizar la extracción de parámetros, con porcentajes de error menores al 10%.

# Abstract

In this work a series of experiments were carried out to prove that supervised learning methods such as Artificial Neural Networks, Decision Trees, Random Forest and Support Vector Regression, typical of machine learning and artificial intelligence, are capable of identifying the parameters/characteristics of electronic devices in order to characterize them and thus model the behavior of these devices in an electronic simulator accurately.

For this purpose, different experiments were performed under different conditions. We started by extracting 2 and then 4 parameters from a resistive load inverter circuit, which was constituted by a polycrystalline silicon thin film transistor (TFT Poly-Si). Then the extraction of 3 parameters in NMOS type transistors was performed individually and extraction tests were carried out using physical measurements. Finally, the extraction of three parameters was performed on India Gallium Zinc Oxide (IGZO) TFT type transistors, with which extraction tests were also performed using physical measurements.

In the three experimental conditions to perform the extraction of parameters, the following was basically done: generate a training set for the device of interest, using an electrical simulation software, with which a series of simulations were performed where the parameters of interest are configured. The resulting I-V curves (inputs) are stored and the parameters (outputs) used during each simulation are recorded. With these samples of I-V curves the learning methods are trained, at this stage they learn to identify the patterns and characteristics of the curves, which make possible the prediction of the parameters that correspond to each one of them. After training, the model is stored, which is ready to receive new input samples and make an extraction/prediction. At this stage it is possible to use the physical measurements of the devices to test the methods with real data and identify the most accurate one.

With the method described in the previous paragraph it was proven that supervised learning can indeed be used to perform parameter extraction, with error rates of less than 10%.

# Capítulo 1 Introducción

Por los años 50s, el tubo de vacío fue reemplazado por el transistor, este dispositivo, tenía un menor tamaño, mejor eficiencia, más fácil y económico de fabricar, además de no tener la necesidad de calentar para funcionar. El uso del transistor permitió reducir el tamaño de las computadoras y llegar a la tecnología que se utiliza día a día.

En las últimas décadas los transistores también han ido evolucionando, siendo más eficientes y pequeños. Esto dio inicio a un cambio que ha sido desapercibido por muchos por la sociedad alrededor del mundo, en los años 80s y 90s el objetivo era tener computadoras, laptops y celulares más pequeños y eficientes. Pero en los inicios del 2000, estos dispositivos comenzaron a ser más delgados, pero a tener una pantalla de mayor tamaño. Esto debido a la necesidad de las diferentes áreas en las que estos dispositivos son utilizados. Por ejemplo, un diseñador gráfico necesita de una pantalla de gran tamaño y buena resolución, o un usuario quiere ver de mejor manera las fotos que podía tomar con su celular. Así comenzó la demanda por celulares, pantalla de computadora/portátiles y televisores con pantallas cada vez más sofisticadas [1], [2]. Fue entonces cuando el transistor de película delgada (TFT por sus siglas Thin Film Transistor) obtuvo una gran importancia, ya que este transistor de efecto de campo, como su nombre lo indica es un dispositivo, que se fabrica colocando el material semiconductor en finas capas de sustrato (usualmente y en un principio vidrio), haciéndolo un transistor lo suficientemente pequeño y delgado para ser utilizado en pantallas modernas.

Con la carrera de las diferentes empresas tecnológicas como Samsung, Sony, Microsoft, entre muchas otras, llevo a la investigación y desarrollo de los TFTs para seguir mejorando e innovando a este transistor. Dichas mejoras del TFT se han logrado en los últimos años, lo que ha terminado en dispositivos más eficientes, rápidos, pequeños, incluso flexibles [3]–[7]. Pero con modelos matemáticos complicados, con un gran número de parámetros.

Una de las herramientas más importantes dentro de la investigación electrónica son los softwares de simulación eléctricos [8]. Este software especial nos permite diseñar, probar y depurar dispositivos sin la necesidad de fabricarlos físicamente [9][10]. Los softwares de

simulación funcionan gracias a modelos matemáticos que representan los fenómenos en la vida real. En el caso de la simulación electrónica, el software utiliza los modelos de los diferentes transistores, capacitores y otros dispositivos electrónicos. Estos modelos matemáticos están compuestos por variables y constantes conocidos como parámetros. El resultado de la simulación dependerá de los valores introducidos en cada uno de los parámetros del modelo.

Cada vez que se desarrolla un nuevo transistor, este debe ser caracterizado, es decir, conocer que valores tienen sus parámetros físicos y eléctricos, que hacen que tenga su comportamiento (modelen al dispositivo), y así poder ser comercializado. Cuando no se conocen los parámetros exactos que tiene un dispositivo, no podría ser simulado dentro de un software, ya que probablemente el comportamiento del dispositivo físico y la simulación resultante serían muy diferentes. De esa forma se pierde el objetivo y funcionalidad de un software simulador, ya que este al usar parámetros incorrectos llevaría a simulaciones erróneas de transistores y otros dispositivos comerciales, y la fabricación y desarrollo de circuitos volvería a lo que era antes (prueba y error) de los simuladores, desperdiciando, tiempo y recursos.

Debido a que conocer los parámetros que tiene un nuevo dispositivo desarrollado es de suma importancia, los investigadores llevan una serie de operaciones y simplificaciones utilizando el modelo matemático y la curva característica de los dispositivos para encontrar los valores en sus parámetros. A este proceso se le conoce como extracción de parámetros. Como se mencionó anteriormente, los dispositivos emergentes son cada vez mejores, pero a su vez, el modelo matemático es complejo, y realizar la extracción de parámetros puede ser una tarea compleja, el proceso que varía de un dispositivo a otro, y en ocasiones puede llevar demasiado tiempo. También, es necesario que la persona quien realiza la extracción tenga de conocimiento de los modelos utilizados y mucha experiencia para obtener parámetros que permitan modelar el comportamiento de los dispositivos.

Cuando una persona cuenta con la suficiente experiencia en la extracción de parámetros, su conocimiento le permite deducir que parámetros son los que están afectando la curva de transferencia de un dispositivo, por ejemplo, el voltaje de umbral ( $V_T$ ) que desplaza a la izquierda o derecha toda la curva. Por esta razón se propone utilizar métodos de aprendizaje supervisado como Redes Neuronales (RN), Árboles de decisión (AD),

Random Forest (RF) y Support Vector Regression (SVR), para llevar a cabo la tarea de extracción de parámetros. Debido a que el principio de funcionamiento de estos métodos, es similar al proceso de aprendizaje que tiene el humano, después de haber procesado una gran cantidad de curvas I-V de dispositivos. El objetivo de los diferentes métodos será identificar los valores de los parámetros que pertenecen a una curva I-V, y con estos parámetros será posible modelar el comportamiento del dispositivo en un simulador con un porcentaje de error mínimo. Siendo una alternativa novedosa, eficiente, rápida, con buena exactitud, sin las limitaciones que tiene el método analítico, tampoco es necesario tener gran experiencia en extracción de parámetros. Por otro lado, tampoco es necesaria ser experto en machine learning, ya que hoy en día, los lenguajes de programación más utilizados, cuentan con librerías especializadas para desarrollar modelos de aprendizaje automático de forma muy sencilla.

## **1.2. Hipótesis**

Como se mencionó anteriormente, una persona que realiza el proceso de extracción de parámetros, con el tiempo, adquiere conocimiento para identificar los parámetros que afectan a una curva I-V a simple vista, por el hecho de haber trabajado con muchas muestras. Por ello, se asume que los métodos de aprendizaje automático, serán capaces de adquirir conocimiento a través de muestras I-V, como lo hace una persona, y así predecir los parámetros pertenecientes a cada una de ellas.

## **1.3. Variables de estado**

Los parámetros de un dispositivo dependerán de su modelo matemático, en esta investigación se base en los parámetros más comunes que se extraen en la literatura, y de ellos, se seleccionaron los más relevantes y que permiten realizar el modelado de la curva I-V. Los parámetros de definen a continuación:

- Voltaje de umbral: es el voltaje mínimo de compuerta a fuente (gate-source) que se necesita para crear una conducción entre las la fuente y drenaje (source-drain).
- Movilidad eléctrica: se refiere a la capacidad que tiene un semiconductor para dejar que los portadores se muevan, con mayor o menor facilidad a lo largo del canal.

- Resistencias: es la oposición que presenta un semiconductor a la corriente eléctrica, cada material cuenta con un coeficiente de resistividad, y fuerza de oposición dependerá directamente a dicho coeficiente.

## **1.4. Objetivos**

### **1.4.1. Objetivo general**

Entrenar modelos de diferentes métodos de aprendizaje supervisado que sean capaces de identificar y predecir diferentes parámetros pertenecientes a una curva I-V, y dichos parámetros permitan modelar con un error mínimo el comportamiento del dispositivo en un simulador eléctrico. El aprendizaje o conocimiento será adquirido a partir de un conjunto de ejemplos de curvas I-V, las cuales serán obtenidas por medio de simulaciones, combinando múltiples valores en los parámetros de interés.

### **1.4.2. Objetivos particulares**

- Analizar los parámetros con mayor efecto sobre el comportamiento de los dispositivos.
- Generar los conjuntos de ejemplos (curvas I-V) a partir de simulaciones con los parámetros de mayor importancia.
- Entrenar métodos de aprendizaje supervisado para la predicción de parámetros en dispositivos electrónicos.
- Analizar el aprendizaje de los métodos para identificar el que brinde mejor desempeño.
- Validar los métodos, utilizando mediciones experimentales para comprobar que los métodos realizan una extracción con niveles altos de exactitud.

## **1.5. Metodología**

En esta subsección se describe la metodología propuesta en esta investigación para realizar la extracción de parámetros, la cual puede aplicarse a cualquier método de aprendizaje supervisado. Por otro lado, también se puede aplicar en diferente tecnología y dispositivo electrónicos, la única condición es que el dispositivo y parámetros de interés

puedan ser simulados en algún software electrónico para la creación de ejemplos para el entrenamiento de los modelos de aprendizaje.

La metodología propuesta es un híbrido entre la metodología de prototipos y la metodología incremental [9], [10].

Como su nombre lo indica, la primera metodología se basa en realizar prototipos rápidos y funcionales sin enfocarse en la presentación. En esta investigación se entrenan modelos de RNs, ADs, RFs y SVRs, identificando las mejores características de cada uno de ellos (número de neuronas, capas, nodos, ramas, árboles etc.) y se almacenan para evaluarlos posteriormente, ya sea con mediciones físicas de dispositivos reales o con mediciones simuladas en condiciones diferentes a lo aprendido durante el entrenamiento. Pero el modelo no se implementa en ningún sistema web o aplicación instalable para un sistema operativo.

De la metodología incremental se tomó la característica de ir avanzando la complejidad de la extracción, después de obtener buenos resultados de los primeros prototipos. Los primeros modelos de aprendizaje fueron entrenados para extraer 2 parámetros y en dispositivos NMOS (negative-channel metal-oxide semiconductor), una tecnología bien establecida y normalizada, lo cual ayuda a tener un control y asegurar que, si los resultados no son favorables, no se debe a la tecnología.

Una vez que los métodos probaron ser capaces de aprender de las curvas I-V e identificar 2 parámetros, se hizo un incremento en el número de parámetros a extraer. También se cambió el tipo de transistor y se utilizaron TFTs. En estas dos primeras etapas los modelos ya entrenados y almacenados fueron evaluados, realizando la extracción de mediciones físicas de transistores NMOS y TFTs.

En un tercer incremento, se pasó a realizar la extracción, ya no en un dispositivo individual, sino en un circuito inversor de carga resistiva. Esta etapa a su vez se realizó en dos incrementos, comenzando con una extracción de 2 parámetros y después hasta 4 parámetros.

A continuación, se describen los pasos que se realizan para la construcción del primer prototipo (obtención de los primeros modelos entrenados):

1. Selección de la tecnología o dispositivo del cual se desea realizar la extracción de parámetros.
2. Seleccionar los parámetros que se desean extraer del dispositivo.



3. Seleccionar un software de simulación eléctrica, en esta investigación se utilizó AIMSpice y LtSpice.
4. Seleccionar el conocimiento que va adquirir el modelo, es decir, el rango que tendrán los parámetros. Por ejemplo, un  $V_T= 1V$  hasta  $V_T=3V$ , dará como resultado un modelo que pueda extraer el  $V_T$  en curvas I-V en ese rango, curvas con menores o mayores voltajes, terminaría en una extracción incorrecta.
5. Seleccionar la sensibilidad que tendrá el modelo entrenado en los rangos que tendrá cada parámetro. Por ejemplo, en el rango de  $V_T= 1V$  hasta  $V_T=3V$ , el barrido será con incrementos de 0.1V, 0.2V, 0.5V etc. Entre más pequeño sea el incremento, mayor será el número de muestras, lo cual es positivo hasta cierto punto, pero podría ocurrir que entre la muestra que tiene un  $V_T= 1.1V$  y la que tiene un  $V_T= 1.2V$  no haya un gran cambio, y para el método de aprendizaje serán consideradas como la misma muestra y el desempeño final sea bajo.
6. Preprocesamiento de las muestras I-V, en este paso las curvas se deben acomodar, de manera que se cuenten con dos archivos, las entradas y las salidas, en el caso de las entradas se tendrá una matriz de  $n \times m$ , donde en cada renglón ( $n$ ) es un ejemplo, una curva I-V, y  $m$  es el total de puntos que componen a la curva I-V (puntos de  $I_D$ , que corresponden a  $V_{GS}$ ),  $m$  estará ligado al barrido establecido en el voltaje de alimentación  $V_{GS}$ . En el caso del archivo de entrada, es una matriz de  $n \times h$ , donde cada renglón representa la salida de la entrada correspondiente, y el número de columnas ( $h$ ) dependerá del número de parámetros a extraer.

También se pueden realizar normalizaciones sobre las curvas I-V, y los parámetros de salida, como por ejemplo usar la distribución Gaussiana (normal), o simplemente realizar una división entre el valor máximo para mantener los datos entre 0 y 1, o -1 y 1, dependiendo de la naturaleza de los datos. El objetivo de esto es que nuestros ejemplos brinden la mayor información para facilitar la predicción.

7. Una vez que los datos fueron ordenados en entradas y salidas, deben dividirse en dos conjuntos, el primero se conoce como conjunto de entrenamiento, del cual los métodos de aprendizaje adquieren el conocimiento, y el segundo es el

conjunto de validación, este conjunto de muestras sirve para medir que tan bueno fue el aprendizaje del modelo.

8. En este paso se procede a comenzar con el entrenamiento de los diferentes métodos de aprendizaje. Para cada uno de los métodos se debe encontrar el mejor modelo, ya que puede haber una gran cantidad de RNs, con diferentes números de capas, neuronas etc. Así que deben probarse diferentes modelos hasta encontrar el que mejor ajuste a los datos que está procesando. Lo mismo se debe hacer con AD, RF y SVR, y la búsqueda del modelo óptimo se detiene cuando el aprendizaje ya no aumenta, y siempre será mejor optar por el modelo más simple. Por ejemplo, si un árbol de decisión con 50 hojas finales obtiene la misma exactitud que el árbol de 200 hojas, claramente el árbol pequeño requiere de menor procesamiento y memoria. Cuando se detecta el mejor modelo de cada método estos son almacenados.
9. Tras el entrenamiento de los métodos, los modelos resultantes pasan a la etapa de evaluación. Esta etapa va en dos pasos, la primera es mediante las métricas convencionales que se obtienen durante el entrenamiento y el puntaje obtenido con el conjunto de evaluación, las métricas comunes son el error cuadrático medio (MSE) y en regresión se utiliza el coeficiente de determinación ( $R^2$ ). La segunda parte de la evaluación es alimentar a los modelos ya entrenados con muestras nuevas, cuyas salidas no fueron exactamente lo que aprendieron en el entrenamiento, pero si dentro de su rango de conocimiento, es decir, si aprendieron un  $V_T = 1, 1.5, \dots, 3 V$ , se utilizan muestras con un  $V_T = 1.8V$  y así sucesivamente con los otros parámetros. También se utilizaron mediciones físicas de las cuales se desconocían sus parámetros.

Finalmente se calcula el porcentaje de error obtenido entre las muestras de validación o medición física, y la curva resultante de la simulación con los parámetros extraídos con los diferentes métodos. Esto permite realizar una comparativa de las extracciones hechas por los métodos e identificar el desempeño de cada uno de ellos.

Para la creación de los prototipos que van en incremento, se deben realizar modificaciones en los pasos 2 a 5, en los cuales se establece el número de parámetros a extraer

y el conocimiento que brindara cada uno de ellos a los métodos de aprendizaje, los demás pasos no se modifican. En la figura 1 se presenta el diagrama general de la metodología, donde se resumen los pasos antes descritos y las tres etapas incrementales realizadas en esta investigación.

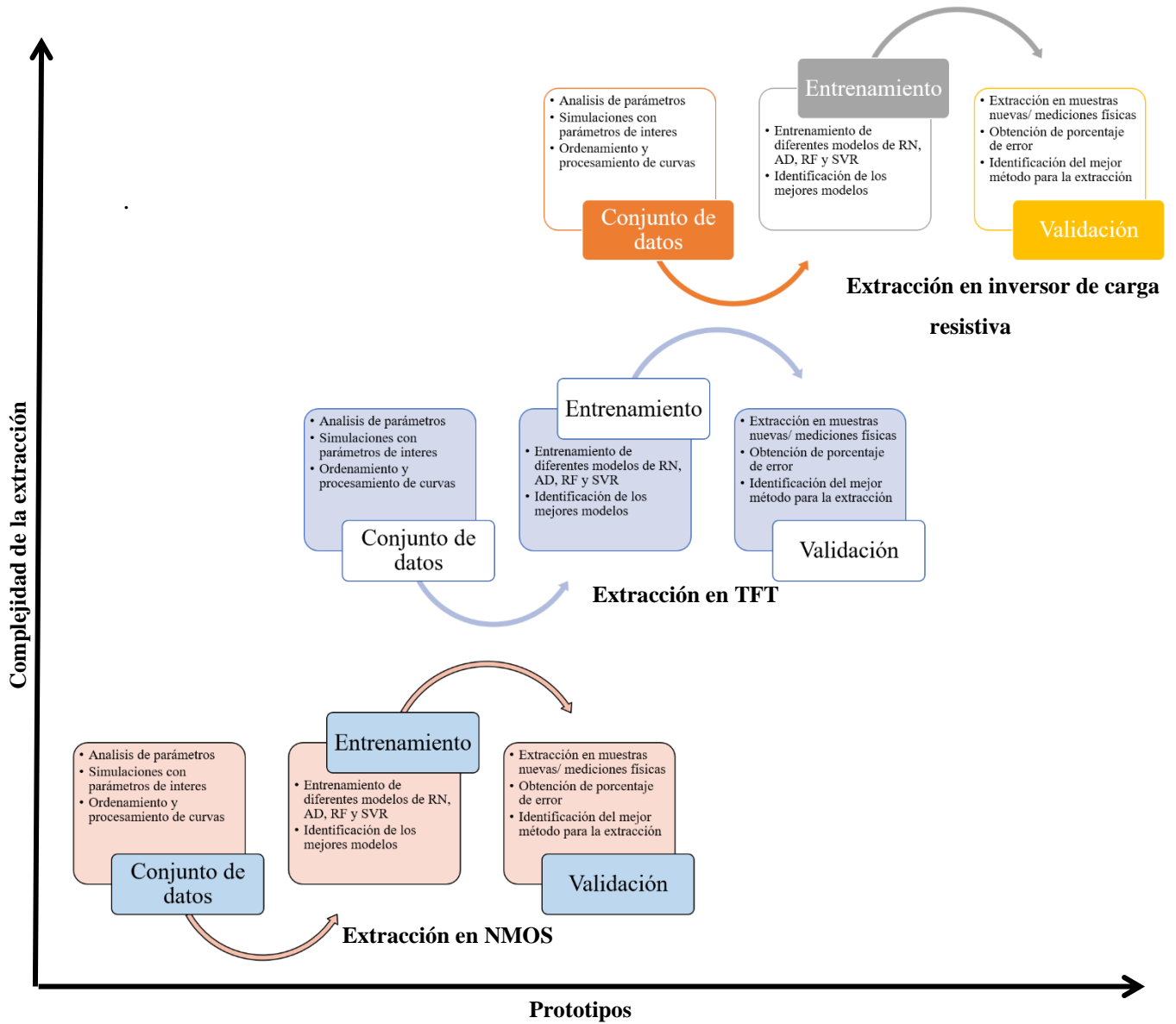


Figura 1. Metodología propuesta

## Capítulo 2 Estado del arte

La extracción de parámetros ha sido estudiada y abordada de diferentes formas; en este capítulo se presentan los trabajos relacionados a este tema que se revisaron y se hace un análisis de estos.

En el artículo [11], propuso una metodología exacta para hacer la extracción de ciertos parámetros en inversores de compuerta flotante CMOS, que cuentan con un interruptor para tener acceso temporal a la compuerta flotante. Los parámetros para extraer son el factor de ganancia  $\gamma$  y otras capacitancias parásitas que se acoplan en la compuerta flotante.

Los inversores de compuerta flotante con varias entradas se han convertido en circuitos muy útiles en diseño de circuitos analógicos modernos y de señal mezclada. La extracción de los parámetros es de suma importancia, ya que el desempeño y comportamiento de los circuitos está directamente ligada a ellos.

El principio de la metodología propuesta es comparar la respuesta transitoria invertida obtenida en el período de restablecimiento con el obtenido. Se tienen seis pasos a seguir y se enlistan a continuación:

**Paso 1:** se debe tomar una medición durante el periodo de restablecimiento de  $A_{VR}$  alrededor de un punto fijo en  $V_{IN}$ , el cual corresponde a una pendiente de ganancia baja en la salida  $V_{OUT}$ .

**Paso 2:** se mide durante el periodo de evaluación de  $A_{VR}$  alrededor de el mismo punto que en el paso 1.

**Paso 3:** el parámetro  $\gamma$  es calculado usando  $\gamma = \frac{A_{VE}}{A_{VR}} - \gamma_{par}A_{VE}$ . El segundo término del lado derecho de la ecuación es insignificante en la región de baja ganancia debido a que  $\gamma_{par} < \gamma$  y ya que  $A_{VE}$  es un valor pequeño, entonces  $\gamma$  se puede aproximar como:  $\gamma \approx$

$$\frac{A_{VE} |_{Paso\ 2}}{A_{VR} |_{Paso\ 1}}$$

**Paso 4:** se mide durante el periodo de restablecimiento de  $A_{VR}$  en un punto fijo cerca del punto de cambio del inversor, que corresponde a una pendiente de alta ganancia en la salida.

**Paso 5:** se toma una medición en el periodo de evaluación en  $A_{VE}$  en el mismo punto que el paso 4.

**Paso 6:** de la ecuación del paso 3, se calcula a  $\gamma_{par}$  usando los valores obtenidos en los pasos 3, 4 y 5 usando:  $\gamma_{par} = \frac{1}{A_{VR} |_{Paso\ 4}} - \frac{\gamma |_{Paso\ 3}}{A_{VE} |_{Paso\ 5}}$

Se fabricaron dos circuitos de prueba usando un proceso CMOS de doble-pli y doble-metal con  $1.2\ \mu m$ . Dichos circuitos fueron medidos utilizando la metodología descrita en los seis pasos. En la Tabla1 se muestra la tabla original del artículo con los resultados obtenidos en los experimentos.

Tabla 1. Resultados de dos circuitos, medidos y teóricos

Prueba	No. entradas	Capacitancia de entrada	Aspecto radio del transistor	Medidos		Teóricamente	
				$\gamma$	$\gamma_{par}$	$\gamma$	$\gamma_{par}$
1	3	0.256 pF	P= $27\ \mu m/1.2\ \mu m$ N= $9\ \mu m/1.2\ \mu m$	0.837	$6.71 \times 10^{-3}$	0.85	$6.71 \times 10^{-3}$
2	6	0.35 pF	P= $16.8\ \mu m/3\ \mu m$ N= $16.8\ \mu m/6\ \mu m$	0.923	-	0.926	-

En el artículo [12], se muestra un método sistemático para una extracción aproximada de transistores orgánicos de película delgada (OTFTs) que son usados en modelos compactos en *Spice*. Los modelos de transistores de película delgada orgánica de tipo universal junto con la plataforma *Smartspice* de *Silvaco* se utiliza para realizar simulaciones experimentales de *plastic logic* (PL) que se llevan a cabo sobre sustratos de plásticos flexibles.

El desarrollo de este tipo de transistores ha ganado popularidad debido a su tamaño amplio y capacidad de trabajo a bajas temperaturas. Su aplicación más común es en las pantallas. Sin embargo, con la investigación se han mejorado las características del mismo, lo cual permite agregar dieléctricos más delgados y así aumentar las áreas de aplicación.

Para realizar la extracción de los parámetros primero comienzan a obtener la capacitancia-voltaje para calcular la acumulación de la carga en el canal. Tomando medidas de prueba con diferentes medias del canal y la compuerta. Se extrae la capacitancia del canal por área ( $F/cm^2$ ) de la pendiente de la capacitancia medida contra la longitud. De este modo se asegura que solo sea considerada la capacitancia de la longitud del canal en la atracción.

En la Figura 2 se muestra la toma de la capacitancia donde a) de obtiene la capacitancia por área en el canal con diferentes longitudes (10, 15, 30 y  $50\ \mu m$ ), para cada punto se calcula la pendiente de la capacitancia medida contra la longitud con diferentes voltajes de

compuerta. El inciso b) muestra la curva de la derivada de  $C - V$  para extraer el voltaje de umbral ( $V_T$ ).

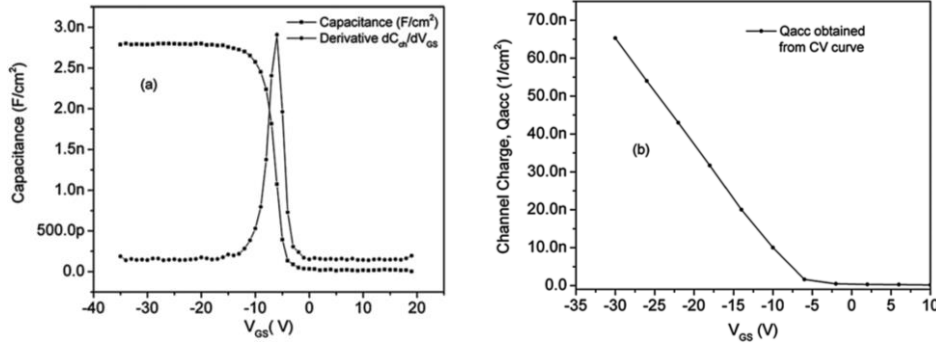


Figura 2. Medidas de capacitancia  
Fuente: [12]

El voltaje de umbral se calcula tomando el punto máximo de la derivada de las características  $C - V$ .  $V_{GS} = -5$  V lo que permite obtener a  $V_T$  sin depender del modelo o de algún ajuste. La carga del canal por área se calcula integrando el área bajo la curva de  $C - V$ .

$$\int_{10}^{V_{GS}} C_{ch} dV_{GS} = Q_{acc} \quad (1)$$

Después se determina el parámetro de movilidad con la ley del comportamiento de potencia. Para esto se necesita obtener la resistencia del canal utilizando el método de transferencia lineal (TLM) donde las características de salida son colocadas desde los dispositivos con diferentes longitudes de canal, para separar los efectos del mismo, con la resistencia de contacto. El resultado obtenido con TLM se muestran en la Figura 3.

De la Figura 3 (parte izquierda) es la extracción de las resistencias en el canal utilizando el método de línea de transferencia y (parte derecha) es la comparación de la movilidad dependiente de  $V_{GS}$  (son los puntos) con los calculados (la línea) utilizando la ecuación:

$$\mu_{Ch} = \left( \frac{MU_{ACC}}{1+(\theta V_{GS})} \right) \left( \frac{Q_{acc}}{C_i V_{ACC}} \right). \quad (2)$$

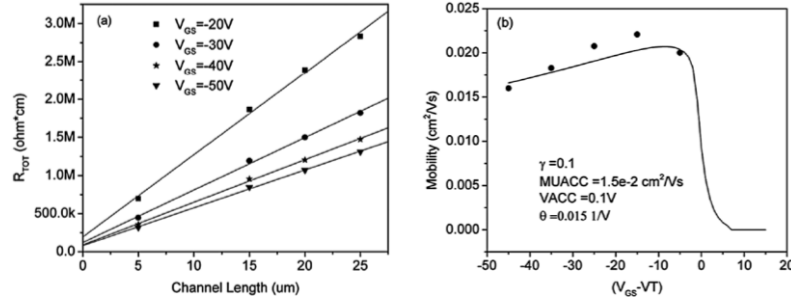


Figura 3. Resultados utilizando TLM  
Fuente: [12]

En este punto  $MUACC$  es la movilidad intrínseca del semiconductor orgánico con  $\gamma$  dando la ley de poder de dependencia. Definiendo la conductancia del canal y la movilidad de los transistores, la corriente en el drenador y fuente en la región de operación lineal se obtiene como:

$$I_{Dslin} = \frac{W}{L} (G_{Ch} V_{DS}) \quad (3)$$

Y la corriente en la región de operación en saturación está dada por la expresión:

$$I_{Dssat} = \frac{W}{L} (G_{Ch} V_{DS}) (1 + (\lambda V_{DS})) \quad (4)$$

Cuando se conoce la resistencia del canal usando TLM y la carga del canal de las curvas de corriente-voltaje. La movilidad del voltaje de compuerta del OTFT se calcula con:

$$\mu_{Ch} = \frac{1}{\left(\frac{W}{L}\right) R_{Ch} Q_{acc}} \quad (5)$$

En la Figura 4 se muestra la comparación de los resultados obtenidos de manera experimental (puntos) y los simulados (línea) de las curvas de transferencia con una longitud de canal de  $15 \mu\text{m}$  en el dispositivo con un voltaje  $V_{DS} = -1\text{V}$  y  $V_{DS} = -15\text{V}$ . En a) la escala esta dada de manera logarítmica, b) es el mismo conjunto de curvas pero con una escala lineal y c) es la salida del mismo dispositivo donde los datos experimentales fueron representados por puntos y los simulados por líneas.

Se concluye que se logró demostrar una metodología para la extracción de parámetros en un TFT orgánico universal de *Silvaco* usando datos experimentales de TFT de PL. El procedimiento requiere el método de línea de transferencia para calcular la resistencia en el canal y medir la capacitancia-voltaje para obtener la carga acumulada en el canal. También se muestra el análisis de la degradación en la movilidad.

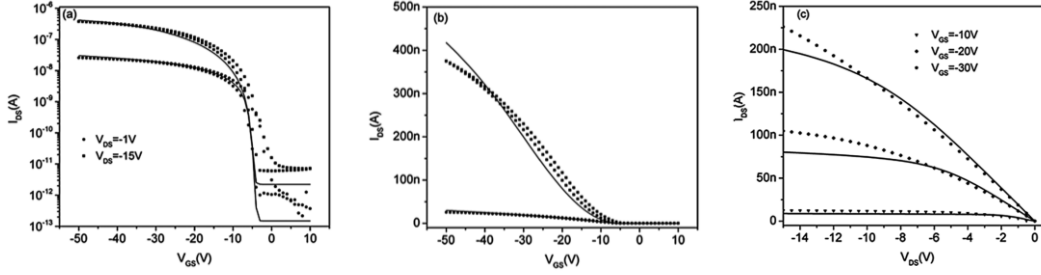


Figura 4. Comparación experimental y calculada  
Fuente: [12]

El artículo [13] se extraen los parámetros de un transistor MOSFET utilizando un nuevo procedimiento de optimización llamado SaPOSM. El cual une el algoritmo de optimización determinístico *Pseudo-Objective Function Substitution Method* (POSM) con el paradigma estocástico de optimización *Simulated Annealing* (SA).

Se hizo la extracción de los parámetros de un MOSFET de  $0.3 \mu m$  en la longitud del canal obteniendo resultados positivos. Una simulación precisa circuitos integrados requiere de software especializado como lo es *SPICE*, pero al mismo tiempo, los simuladores dependen de que tan preciso es el modelo matemático que representa el comportamiento de los dispositivos. La extracción de parámetros es un problema de optimización no lineal, que pretende minimizar el mínimo cuadrado de una función objetivo.

$$\begin{aligned}
 S(x) &= \sum_{i=1}^k [e_i(x)]^2 \\
 &= \sum_{i=1}^k \sum_{j=1}^m [(W_i(I(x, v_{ij}) - I_m v_{ij}))]^2
 \end{aligned} \tag{6}$$

En donde:

El vector  $x$  es el vector de los parámetros  $x = (x_1, x_2, \dots, x_n)$  y  $n$  es el número de parámetros.

- $v_{ij} = [v_{ds}(j)v_{gs}(j), v_{bs}(j)]$  es la  $j$ -ésima muestra en el punto  $i$ -ésimo de la curva corriente-voltaje (I-V).
- $I(x, v_{ij})$  es el modelo de corriente en el drenador con los parámetros  $x$  con los voltajes  $v_{ij}$ .
- $I_m(v_{ij})$  es la corriente del drenador medida (normal) en el voltaje  $v_{ij}$ .
- $W_i$  es el peso de la  $i$ -ésima curva obtenida por el diseñador.



De igual manera se menciona que en la extracción de los parámetros puede utilizar métodos clásicos deterministas de optimización no lineal, por ejemplo, el método modificado Gauss-Newton, el gradiente por descenso, el algoritmo Levenberg-Marquardt y métodos de búsqueda directa para complementar el proceso de extracción. Esto significa que bien pueden utilizarse redes neuronales artificiales para realizar la extracción de parámetros ya que estas utilizan métodos como el gradiente descendiente, Levenberg-Marquardt entre otros.

La unión de SA con POSM la describen de la siguiente manera:

- Inicialmente se ejecuta SA, en cada iteración se detectan los fallos, para reducir el costo de la función entre dos intervalos (de igual temperatura) consecutivos.
- Se utiliza POSM en el punto de búsqueda de la corriente. Aquí POSM será capaz de buscar un mínimo local de manera más eficiente que SA.
- El mínimo local es usado como condición inicial para comenzar el siguiente paso de SA con una menor temperatura. Si SA no presenta un mejoramiento de la función objetivo, se le permitirá continuar con el siguiente intervalo de temperatura sin utilizar POSM.

Para mejorar la eficiencia de búsqueda también utilizaron un cambio en el tamaño del paso de búsqueda de forma dinámica cuando SA está trabajando. Iniciando con un tamaño de paso  $\Phi_0$ . Dicho tamaño de paso  $\Phi$  aumentara cuando haya un movimiento de caída durante la etapa del SA con la finalidad de encontrar el mínimo local más rápido. Por otro lado, si la función objetivo no cambia antes y después del intervalo de corriente con la misma temperatura,  $\Phi$  se disminuye un 10%.

Para finalizar o detener el algoritmo se toman en cuenta alguna de las tres condiciones:

1.  $S(x_{optimo}) \leq \epsilon_1$ , el error en la función de costo se aproxima a cero.
2.  $T \leq \epsilon_2$  la temperatura en el recocido (*annealing temperature*) se aproxima a cero.
3.  $Count \geq M$ . Cuando la función objetivo no mejora para  $M$  ( $M \geq 2$ ) en dos intervalos consecutivos con la misma temperatura.

En la Tabla 2, se muestran los diferentes parámetros, con su valor mínimo ( $x_{bajo}$ ) y el más alto ( $x_{alto}$ ) que fueron extraídos. Así mismo se tienen los resultados de utilizando SA y el método combinado SaPOSM.

Al final los investigadores concluyen que el método que mezcla el SA con POSM llamado SaPOSM presentó obtener resultados que se consideran aceptables con un tiempo

computacional moderado. Además, que no se limita a la extracción de parámetros de un transistor MOSFET, sino que puede ser utilizado en problemas de optimización no lineales, donde hay gran cantidad de locales mínimos.

Tabla 2. Resultados de extracción

Parámetro	$x_{bajo}$	$x_{alto}$	SA	SaPOSM
VTO	-0.5	-1.5	-1.026	-1.016
UO	10	600	50.40	52.27
NSUB	1.5e16	1.0e20	0.161e19	0.5002e19
GAMMA	0.0	2.5	0.3234	0.4941
ETA	0	2.0	0.3213e-2	0.3121e-2
THETA	0	2.0	0.1841	0.2077
KAPPA	0	30	10.92	19.98
VMAX	1e4	1e8	0.2841e6	0.5126e6
XJ	1e-8	3e-7	0.2388e-6	0.1831e-6
TOX	-	-	1.0e-8(ajustado)	1.0e-8(ajustado)
LD	-	-	5.8e8(ajustado)	5.8e8(ajustado)
# de evaluaciones de función.	-	-	235,009	6,312
Error relativo final	-	-	0.2035e-4	0.514e-4

En este trabajo, se hizo la combinación de dos métodos para extracción de parámetros de un transistor MOSFET, mejorando el tiempo de búsqueda. Se menciona que la obtención de los parámetros es una tarea no lineal y se podrían utilizar redes neuronales artificiales, además de los algoritmos genéticos los cuales son usados en extracción de parámetros. Este punto es favorable, ya que las RNs no han sido estudiadas profundamente para extraer parámetros de circuitos electrónicos.

En el artículo [5] se utilizaron los algoritmos genéticos (AG) como una herramienta de extracción de parámetros de un transistor orgánico de efecto de campo (OFET), en la parte baja y superior de voltaje de umbral. Los parámetros que fueron seleccionados a buscar fueron el factor de la mejora de la movilidad  $\gamma$ , el voltaje de umbral  $V_{th}$ , la oscilación subumbral  $S$ , la modulación de la longitud del canal  $\lambda$  y la forma de la región donde cambia la pendiente  $m$ .

Los autores utilizan los AG para reemplazar el método tradicional para hacer la extracción de parámetros ya que es una tarea compleja basada en relación directa de dichos parámetros que requiere un análisis matemático del modelo matemático de cada dispositivo, este proceso se conoce como Modelo unificado y método de extracción de parámetros (UMEM). La desventaja de una persona con mucha experiencia para funcionar correctamente.

En la sección del desarrollo se describe que cada parámetro desconocido es denominado gen, y a cada vector de estos parámetros se le conoce como cromosoma. Su algoritmo comienza generando una población de unas variaciones aleatorias de una semilla. Después cada uno de los elementos o individuos es evaluado de acuerdo a su aptitud o capacidad de cumplir los criterios especificados por el diseñador. Cada elemento es una lista de valores de parámetros que serán extraídos. Se utilizaron los operadores de selección, *crossover* y mutación de forma iterativa. La función objetivo o de aptitud que se utilizó fue la función del error cuadrático medio (MSE) que se presenta a continuación:

$$f = \frac{1}{M} \sum V_{gs} \sum V_{ds} \left[ \frac{I_{d,real} - I_{d,GA}}{I_{d,real}} \right]^2 \quad (7)$$

Donde  $M$  es el número de ejemplos del conjunto de datos.

El trabajo también nos indica cuales fueron los parámetros utilizados en el diseño del AG, los cuales se observan en la Tabla 3.

Tabla 3. Parámetros de AG

Parámetros de los AG	Valor
Número de variables	8
Tamaño de la población	100
Número máximo de generaciones	700
Selección	Por torneo
Crossover	Aritmético
Mutación	Factible adaptativo
Fracción de crossover	0.8

Tabla 4. Ejemplo de resultados

Parámetro	T1		T2		T3	
	Valor extraído	Valor de referencia	Valor extraído	Valor de referencia	Valor extraído	Valor de referencia
$V_{th}$ (V)	-1.07	-1.2	-1.1	-1.04	-12.35	-12
$\gamma$	0.15	-	0.03	-	0.98	0.91
$S$ (V/dec)	0.16	0.1	0.1	0.078	3.7	4.1
$V_{aa}$ (V)	560	-	48	-	240	350
$\lambda$ (V <sup>-1</sup> )	$-5.2 \times 10^{-3}$	-	$-7 \times 10^{-3}$	-	$-3 \times 10^{-3}$	$-1.2 \times 10^{-3}$

Algunos de los ejemplos de sus resultados se muestran en la Tabla 4. En ella se muestran tres transistores con características físicas diferentes (W y L ambos  $\mu\text{m}$ ). Dos posibles deficiencias en este trabajo es que no mencionan los rangos de valores utilizados en cada parámetro, solamente informan que se usaron 70,000 conjuntos de parámetros. Otro punto es que tampoco indican el porcentaje de error en sus resultados. Con este trabajo se

puede estar seguro de que los AG permiten realizar la extracción de parámetros con buenos resultados, como una alternativa simple y rápida al método analítico.

En el trabajo [14], se propone un algoritmo híbrido basado en una colonia de abejas artificiales (Fig. 5) como una herramienta para la extracción de parámetros y se compara con los métodos convencionales de extracción de parámetros basados en algoritmos puramente matemáticos y genéticos. Debido a que la investigación sobre los transistores de efecto de campo orgánicos (OFET) se ha incrementado en la última década, para obtener una estructura ligera y flexible, así como como una producción práctica y con menor costo.

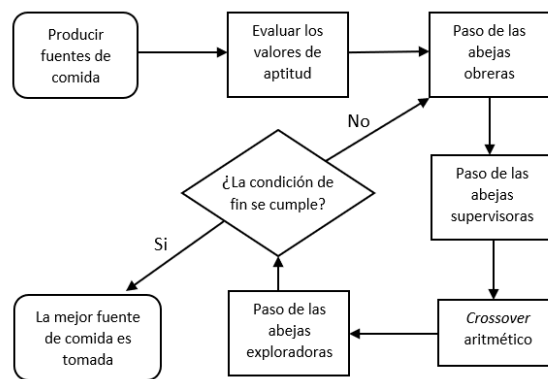


Figura 5. Diagrama de flujo del algoritmo ABC  
Fuente: [14]

Tabla 5. Resultados de la extracción de parámetros

#	Método	$V_T$ (V)	$\gamma$	$V_{AA}$ (V)	$R(\Omega)$	$\alpha_s$	$m$	$\lambda(I/V)$	NRMSE
T1	h-ABC	1.02	0.76	1.58	87k	0.57	2.69	0.0069	0.0026 ± 28%
	AG	1.10	0.51	1.66	65k	0.60	2.80	0.0037	0.0052 ± 21%
	Método analítico	1.05	0.37	3.94	11k	0.53	2.68	0.02	0.0602
T2	h-ABC	5.47	0.90	1983	59k	1.06	2.13	-0.0090	0.0183 ± 2%
	AG	5.17	0.96	1498	150k	1.05	2.06	-0.0084	0.0190 ± 8%
	Método analítico	6.90	1.08	891	47k	1.12	2.09	-0.0071	0.0225

Se aplicaron los métodos a un modelo compacto OFET conocido para dos transistores diferentes, ambos con pentaceno como semiconductor orgánico. El primer transistor (T1) está disponible en la literatura y el otro (T2) se fabrica en el laboratorio de los autores. Se concluye que la extracción de parámetros basados en algoritmos genéticos presenta un buen comportamiento siendo aceptables ya que con los datos experimentales de T1 tuvo un error RMS normalizado (NRMSE) del 0.26%. Sin embargo, es 1.83% para T2 debido a la falta de mediciones. En la Tabla 5 se muestran los resultados para ambos transistores en los diferentes

tipos de extracción aplicados. Se concluye que este tipo de AG permite realizar una extracción de parámetros, con resultados similares al método analítico.

En el artículo [15], se modeló y simuló un transistor de efecto de campo utilizando redes neuronales artificiales (RNAs). Este enfoque permite modelar los dispositivos de forma sencilla. Se utilizó el simulador 2-D Atlas para hacer la comparación de los resultados del transistor con los que brinda la red neural.

La red se entrenó en Matlab (2006a) y las características generales son las siguientes:

- Entradas:  $V_{gs}$ ,  $V_{ds}$ ,  $L$  y  $t$ .
- Salidas:  $I_D$
- Tipo de red: Feed-Forward Back-Propagation
- Función de transferencia:  $\text{logsig}(y) = \left(\frac{1}{1+e^{-x}}\right)$
- Función de entrenamiento: TRAINRP.

Se utilizó la fórmula del error relativo para conocer el porcentaje de error de los resultados que generó la red neuronal (RN) contra los experimentales simulados.

$$RE\% = \left(\frac{I_{exp} - I_{pred}}{I_{exp}}\right) 100 \quad (8)$$

En la Tabla 6 se muestran los rangos de los datos de la red neuronal. Al final la mejor red contó con dos capas ocultas de 4 y 10 neuronas respectivamente, con un MRE de 1.4103% y un factor de correlación de 0.9996.

Tabla 6. Rango de datos

Rango	Entradas			Salidas	
	Grosor $t$ (nm)	$L$ ( $\mu\text{m}$ )	$V_{GS}$ (V)	$V_{DS}$ (V)	$I_D$ ( $\mu\text{A}$ )
Mínimo	5.7	10	-3	-3	-4.96
Máximo	6	50	0	0	0

En el artículo [16], se propone un nuevo procedimiento para la extracción de los parámetros básicos de un transistor de película delgada amorfo, dicho método evita la optimización no lineal, el cual es utilizado en la actualidad. La extracción se basa en la integración de los datos experimentales medidos. Se realizó la extracción en las regiones lineal y de saturación por encima del voltaje de umbral. La exactitud de las curvas simuladas.

La exactitud de las curvas simuladas, utilizando los parámetros obtenidos con el nuevo método, se verificaron con los datos medidos y calculados con el modelo matemático. Dentro

del artículo, presentan las ecuaciones del transistor para mostrar el comportamiento de los diferentes parámetros ( $W$ ,  $L$ ,  $C_i$ ,  $\mu_0$ ,  $V_T$ ,  $R$ ,  $\gamma$  y  $V_{AA}$ ).

$$I_{DS} = \left( \frac{K/V_{AA}^\gamma}{1+R(K/V_{AA}^\gamma)(V_{GS}-V_T)^\gamma} \right) \left( \frac{(V_{GS}-V_T)^{1+\gamma} V_{DS} (1+\lambda V_{DS})}{\left[ 1 + \left[ \frac{V_{DS}}{V_{DS sat}} \right]^m \right]^{1/m}} \right) \quad (9)$$

Donde  $K = C_i \mu_0 \left( \frac{W}{L} \right)$ ,  $W$  es el ancho del canal,  $L$  es la longitud del canal,  $C_i$  es la capacitancia de compuerta,  $\mu_0$  es la movilidad de la banda,  $V_T$  es el voltaje umbral,  $R$  es la resistencia de la fuente en el drenador,  $\gamma$  y  $V_{AA}$  son parámetros empíricos que definen la variación de la movilidad con  $V_{GS}$ ;  $m$  es la agudeza en la región y  $\lambda$  es la modulación de la longitud del canal.

La ecuación 10 descuida la corriente de fuga y es el resultado de considerar que la movilidad del efecto de campo aumenta con el voltaje de la compuerta.

$$\mu_{FET} = \mu_0 \left( \frac{V_{GS}-V_T}{V_{AA}} \right)^\gamma \quad (10)$$

La conductancia intrínseca del canal  $g_{chi}$ , para el bajo voltaje en el drenador (región lineal) se expresa como:

$$\begin{aligned} g_{chi} &= \frac{W}{L} C_i \mu_{FET} (V_{GS} - V_T) \\ &= \frac{W}{L} C_i \mu_0 \frac{1}{V_{AA}^\gamma} (V_{GS} - V_T) \end{aligned} \quad (11)$$

La corriente en el drenador en la región lineal, de la ecuación 12 y para  $V_{GS} > V_T$  se escribe:

$$I_{DS lin} = \frac{K}{V_{AA}^\gamma} (V_{GS} - V_T)^{1+\gamma} V_{DS} \quad (12)$$

El procedimiento primero elimina a los parámetros  $\gamma$  y  $V_{AA}$  de la ecuación, ya que estos varían de un transistor a otro, y son altamente dependientes de la tecnología utilizada. Además, que  $\gamma$  se encuentra como exponente. Se extrae  $V_T$  y  $\gamma$  dentro de la región línea. La función  $H(V_{GS})$  esta definida como:

$$H(V_{GS}) = \frac{\int_0^{V_{GS}} I_{DS} dx}{I_{DS}(V_{GS})} \quad (13)$$

Después de la integración se divide entre  $I_{DS}$ , obteniendo la expresión:

$$H(V_{GS}) = \frac{1}{2+\gamma} (V_{GS} - V_T) \quad (14)$$

El voltaje de umbral  $V_T$  se obtiene del intercepto y  $\gamma$  de la pendiente de la región lineal cuando  $V_{GS} > V_T$ .

Después, tomando la ecuación 14, la expresión  $I_{DS}^{1/(1+\gamma)}$  contra  $V_{GS}$ , es obtenida, y calculando la pendiente  $S_1$  de  $I_{DS}^{1/(1+\gamma)}$ , el valor de  $V_{AA}$  puede ser extraído de:

$$V_{AA} = \left[ \frac{KV_{DS}}{S_1^{1+\gamma}} \right]^{1/\gamma} \quad (15)$$

El siguiente paso es la extracción en la región de saturación cuando  $V_{GS} = V_{DS}$ . Se obtiene la expresión  $I_{DS sat}^{1/(2+\gamma)}$  contra  $(V_{GS} - V_T)$  de la ecuación:

$$I_{DS sat} = \frac{K}{V_{AA}^\gamma} \alpha_s (V_{GS} - V_T)^{2+\gamma} \quad (16)$$

Se obtiene el parámetro  $\alpha_s$  como:

$$\alpha_s = \frac{S_s^{2+\gamma} V_{AA}^\gamma}{K} \quad (17)$$

Los parámetros  $m$  y  $\lambda$  se pueden extraer de la expresión de la corriente de salida de la ecuación 18.

$$m = \frac{\log 2}{\log} \left[ \frac{K}{V_{AA}^\gamma} \frac{\alpha_s (V_{GS} - V_T)^{2+\gamma}}{I_{DS sat} (V_{DS sat})} \right] \quad (18)$$

En el trabajo se concluye que con el procedimiento permite realizar la extracción de todos los parámetros de un modelo de nivel 15 implementado en AIM-Spice, sin el uso de una optimización no lineal u algún método gráfico. El valor de  $V_T$  y del exponente  $\gamma$  pueden ser extraídos independientemente uno del otro usando un solo proceso matemático incluyendo este método integral que reduce el ruido experimental.

En el artículo [17] se realizó la extracción de parámetros de un transistor OTFT de Infineon utilizando dos tipos de algoritmos genéticos: *Queen-Bee* y *Crossing-Mates*.

Se describe el funcionamiento de ambos algoritmos, comenzando por el *Queen-Bee*, el cual genera una población de individuos de manera aleatoria de una semilla. Cada elemento o individuo de la población es una lista de parámetros a extraer. Después de generar la población, se califica la aptitud de cada individuo. Se definió la función objetivo como el error total RMS  $\varpi$  representado como:

$$\varpi = \sqrt{\sum_{V_{GS} V_{DS}} (I_{real}(V_{GS}, V_{DS}) - I_{AG}(V_{GS}, V_{DS}))^2} \quad (19)$$

Se clasifican a los individuos con mejor aptitud o menor error y se utilizan como semilla base para la siguiente generación con la finalidad de mantener la mejor aptitud y la semilla forma parte de la nueva población.

Por otro lado, el algoritmo *Crossing-Mates* después de generar la población y evaluar su aptitud, se selecciona al individuo óptimo y este se cruza con otro seleccionado aleatoriamente hasta generar una nueva población que es del doble del tamaño que la primera. Después se toma un número  $N$  de nuevos padres óptimos que generaran la siguiente generación, repitiendo el proceso. El cruzamiento produce nuevos individuos a partir de dos padres, suponiendo que cada parámetro hijo es una combinación lineal de los padres. Los autores utilizaron el siguiente factor de cruzamiento con la finalidad de mejorar el funcionamiento.

$$p = \frac{(1+a)p_1+(1-a)p_2}{2} \quad (20)$$

En la Figura 6 se presenta el diagrama de flujo de los algoritmos descritos anteriormente donde a) es el algoritmo *Queen-Bee* y b) es el algoritmo *Crossing-Mates*.

Se aplicaron ambos algoritmos en un equipo con procesador de 2.0 Ghz, con 1024 Mb de memoria RAM con un sistema operativo Windows Xp. Las características del transistor utilizado fue un OTFT con  $SiO_2$  como dieléctrico en la compuerta con unas dimensiones de  $W = 170\mu m$ ,  $L = 130\mu m$ , espesor de oxido en compuerta de  $100nm$ ,  $\epsilon_r = 3.9$  con  $\mu_0 = 0.4 cm^2V^{-1}s^{-1}$ . Los resultados obtenidos por los algoritmos se presentan en la Tabla 7.

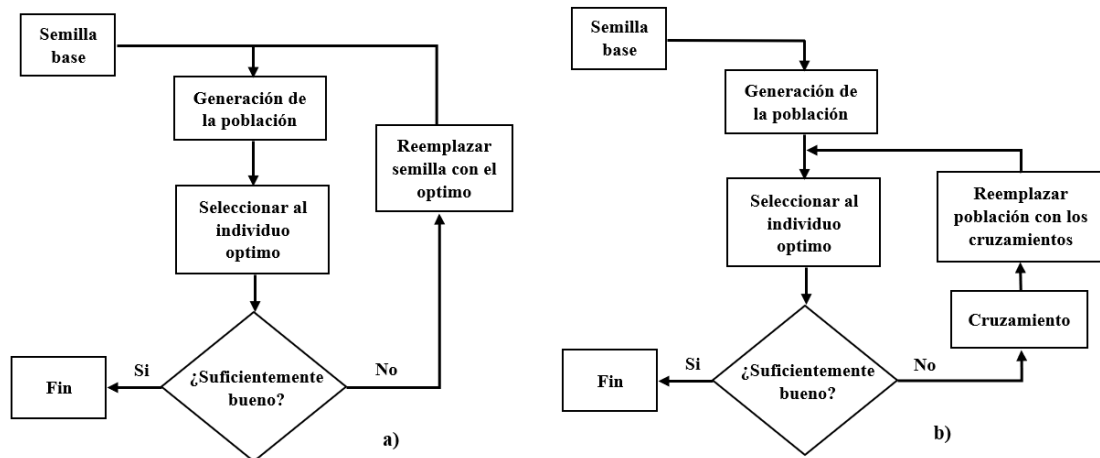


Figura 6. Diagrama de flujo de dos AG Basada en [17]



Tabla 7. Resultados de dos tipos de AG

Parámetro	Algoritmo <i>Queen-Bee</i>		Algoritmo <i>Crossing-Mates</i>	
	Valor promedio	$\sigma$ (%)	Valor promedio	$\sigma$ (%)
$V_t$ (V)	-11.49	3.4	-11.41	1.4
$\gamma_a$	1.27	2.4	1.27	2.2
$V_{aa}$ (V)	48.08	0.7	48.11	0.7
$\alpha_s$ ( $V^{\frac{1}{2}}$ )	0.41	0.9	0.41	0.8
$\lambda$ ( $V^{-1}$ )	1 $\mu$	10.1	1 $\mu$	9.3
$R$ ( $\Omega$ )	5.8 k	71.8	3.8 k	65.3
$m$ ( $\cdot 10^{-3}$ )	2.22	1.7	2.21	1.6
$\delta$ (V)	0.13	2.7	0.13	2.0
$i_{r0}$ (A)	3.4 p	8.5	3.4 p	6.1
$V_{r0}$ (V)	5.06	8.5	5.04	5.6
<b>Tiempo</b>	44.08	44.6	24.08	34.5

En el artículo [18] se realiza la extracción de parámetros de un modelo tipo MOS (Metal-Oxido-Semiconductor) de nivel 3 y en un modelo compacto TFT utilizando lógica difusa y se compara con el modelo unificado y método de extracción (UMEM) el cual, como ya se ha mencionado es basado en análisis analítico.

En este trabajo se hace la discusión sobre la extracción de parámetros en modelos de dispositivos utilizando las mediciones de I-V es una tarea compleja para modelos recientes. La cual debe encontrar cientos de parámetros, algunos correlacionados, requiriendo optimización global y experiencia humana.

Para facilitar el proceso se utilizan métodos directos de extracción, además de extraer solo algunos del total de ellos, en modelos con gran número de parámetros. Los métodos de extracción directa requieren de una segunda etapa para incluir la interacción entre los diferentes parámetros, lo que lleva a la necesidad de métodos globales como SaPOSM, difusión rápida, algoritmos genéticos entre otros, para encontrar los valores que mejor ajusta a los datos experimentales. La desventaja de SaPOSM o difusión rápida es que requieren de mucho poder de cómputo además de ser complicados de programar, por otro lado, los algoritmos genéticos se codifican más fácil, pero carecen de precisión. Estos métodos tienen algo en común, ellos dejan de lado el conocimiento de un humano experto en realizar esta tarea.

El conocimiento de un experto puede ser usado por la lógica difusa, la cual es una técnica utilizada en control e identificación, una de sus ventajas es que no requiere una descripción precisa del comportamiento del sistema. La lógica difusa solo necesita saber que

el aumento del voltaje de umbral recorre la curva I-V a la derecha o a la izquierda si el voltaje disminuye.

Se menciona de forma breve el funcionamiento de un sistema difuso, el cual tiene cuatro etapas. La primera se conoce como rusificación, en donde se definen las funciones de pertenencia de las variables de entrada, convirtiendo los valores reales a datos difusos. En la etapa de inferencia determina un valor de salida en cada regla, dependiendo del grado de pertenencia a un conjunto difuso. El tercer etapa llamada composición los conjuntos difusos de cada variable se unifican para conformar un solo conjunto difuso para cada variable de salida. La última etapa es la defuzzificación, en ella se hace la conversión de las salidas difusas a valores reales.

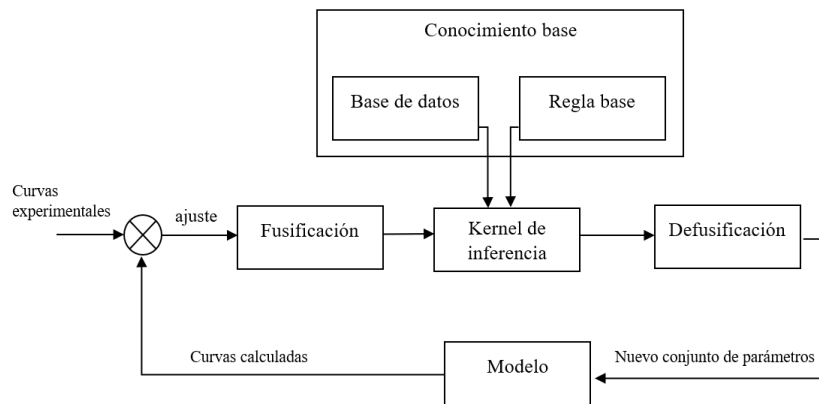


Figura 7. Sistema difuso para extracción

En la Figura 7 se presenta su diagrama de un controlador de una entrada y una salida, comúnmente la entrada es analógica, que debe convertirse en una señal discreta para su procesamiento.

La extracción de parámetros la realizan por partes, se encuentran ciertos parámetros que permiten calcular otros. Se comienza la extracción de  $\beta$ ,  $V_t$  y  $\theta$  ajustando los datos experimentales de las curvas  $I_{DS} - V_{GS}$  usando  $V_{DS} = 50mV$  y  $V_{BS} = 0mV$ . Conociendo como afectan estos parámetros al comportamiento de la curva se pueden establecer las reglas.

- $V_t$  recorre por completo la curva
- $\beta$  brinda un factor de escala
- $\theta$  describe la curvatura de  $I_{DS} - V_{GS}$  conforme  $V_{GS}$  incrementa.

Con el conocimiento anterior establecieron las reglas:

1. Si la curva calculada se encuentra a la derecha de la curva experimental, se debe disminuir  $V_t$  y viceversa.
2. Si la curva calculada se encuentra por encima de la experimental, se debe disminuir  $\beta$  y viceversa.
3. Si la curva calculada tiene más curvatura que la experimental, se debe disminuir  $\theta$  y viceversa.

Implementaron las reglas anteriores utilizando los conjuntos difusos de la Figura 8. Para la defuzzificación se usó el método de centro de área (COA). Además de normalizar los datos de entrada a valores entre -1 y 1.

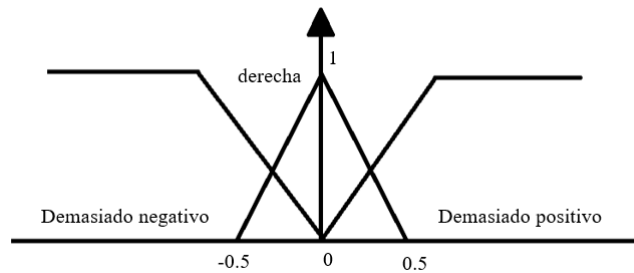


Figura 8. Funciones de fuzzificación

Cuando se ha encontrado el voltaje umbral  $V_T$  para  $V_{BS}=0V$ , se puede extraer  $\Gamma$  y  $\phi_B$  con diferentes valores en  $V_{BS}$ . Se sabe que el efecto de esos dos parámetros es manipular el espacio entre las curvas  $I_{DS} - V_{GS}$  cuando el voltaje general cambia. Con ellos proponen las siguientes reglas:

1. Si la curva calculada se encuentra a la izquierda de la experimental correspondiente para ese valor mínimo de  $V_{BS}$ , se debe aumentar  $\Gamma$  y viceversa.
2. Si la curva calculada, en un valor intermedio de  $V_{BS}$ , se encuentra a la izquierda de las mediciones correspondientes, se debe aumentar  $\phi_B$  y viceversa.

Después de obtener los parámetros antes mencionados de una forma iterativa, realizan la extracción de parámetros que afectan la región de saturación  $\alpha$  y  $\lambda$  desde las curvas  $I_{DS}-V_{GS}$ . Aquí hacen mención que han utilizado solo la curva correspondiente para el valor máximo del voltaje de compuerta ( $V_{GS}=3.3V$ ), debido a que se tienen un menor número de

parámetros a extraer. Usar modelos más complejos requerían el uso de más curvas. El conocimiento de los dos parámetros es:

- $\alpha$  establece la relación entre  $V_{GS}$  y  $V_{Dsat}$ .
- $\lambda$  controla la pendiente de la curva  $I_{DS}-V_{GS}$  en saturación.

Con ello proponen las siguientes reglas:

1. Si la curva calculada se satura muy rápido, se debe disminuir  $\alpha$  y viceversa. Comparando los valores de  $I_{DS}$  con el voltaje correspondiente al 50% de la corriente máxima experimental.
2. Si la pendiente de la curva calculada es baja, se debe aumentar  $\lambda$  y viceversa. Esto se logra comparando los valores máximos de  $I_{DS}$ , en  $V_{DS}$  y  $V_{GS}$ .
3. Por último, hacen la extracción de dos parámetros,  $A$  e  $I_D$  que corresponden a la corriente de volumen, su efecto se observa en la corriente en el drenador y en la terminal del *bulk*.  $A$  brinda un factor de escalado y  $B$  describe el reparto de la corriente del *bulk* con  $V_{DS}$ . Con lo anterior propusieron las siguientes reglas:
  1. Si la curva calculada  $I_{bulk}-V_{DS}$  es más alta que la medida, se debe disminuir  $A$  o viceversa. Se logra comparando los valores de  $I_{bulk}$  con el voltaje máximo experimental correspondiente y para el máximo  $V_{GS}$ .
  2. Si la curva calculada  $I_{bulk}-V_{DS}$  se eleva más rápido que la medida, se debe disminuir  $B$  o viceversa. Se logra comparando  $I_{bulk}$  con el voltaje correspondiente al 50% de la corriente máxima experimental para un valor intermedio de  $V_{GS}$ .

El controlador difuso se mantendrá en iteración calculando nuevos parámetros hasta llegar a una condición para finalizar. Utilizaron el error RMS global, en las curvas experimentales y las calculadas, tomando en cuenta cada punto de ellas. Se determino finalizar las iteraciones cuando se tenga un error RMS menor al 1%.

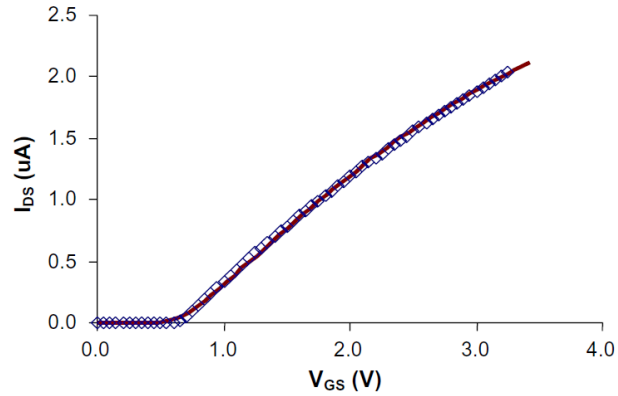


Figura 9. Comparación de curva experimental y la calculada

Fuente: [19]

En la Figura 9 se presenta una comparación de la curva real (línea) con la calculada, observando buena concordancia entre ellas. En la Tabla 8 se presentan los resultados de los parámetros que fueron extraídos, tanto con lógica difusa y el método UMEM. Se hizo una extracción de 9 parámetros, el procesamiento se llevó a cabo en un equipo con 2.0 GHz y 512MB de memoria RAM.

Tabla 8. Resultados de extracción con LD

Parámetro	Valor inicial	Valor extraído	Método directo
Para 2 V			
$V_t$	2 V	0.6459 V	0.6397 V
$\beta$	0.15	$21.2 \mu A/V^2$	$21.0 \mu A/V^2$
$\theta$	0.1	$0.1298 V^{-1}$	$0.1260 V^{-1}$
$\lambda$	0.01	$3.95 \times 10^{-10} V^{-1}$	$\sim 0$
$\alpha$	1	1.183	1.17
$\Gamma$	$1 V^{1/2}$	$0.65 V^{1/2}$	$0.65 V^{1/2}$
$\phi_b$	2 V	0.40 V	0.40 V
$A$	$3.0 V^{-1}$	$0.606 V^{-1}$	$0.61 V^{-1}$
$B$	30 V	14.2 V	14.1 V
Tiempo			26 s
Para 5 V			
$V_t$	5 V	0.6457 V	0.6397 V
$\beta$	0.001	$21.1 \mu A/V^2$	$21.0 \mu A/V^2$
$\theta$	0.01	$.1295 V^{-1}$	$0.1260 V^{-1}$
$\lambda$	$1 \times 10^{-10}$	$2.95 \times 10^{-10} V^{-1}$	$\sim 0$
$\alpha$	0.1	1.181	1.17
$\Gamma$	$4 V^{1/2}$	$0.66 V^{1/2}$	$0.65 V^{1/2}$
$\phi_b$	1 V	0.40 V	0.40 V
$A$	$1.5 V^{-1}$	$0.604 V^{-1}$	$0.61 V^{-1}$
$B$	1.0 V	14.1 V	14.1 V
Tiempo			29 s

También aplicaron esta nueva técnica a un modelo OTFT para determinar sus parámetros. Para facilitar el proceso se analizó un dispositivo de contactos óhmicos, de manera que solo se deben extraer seis parámetros:  $\alpha_S$  el efecto de saturación,  $V_{aa}$  el ajuste de movilidad,  $V_T$  el voltaje de umbral,  $\lambda$  la modulación de la longitud del canal,  $\gamma_a$  el efecto de campo eléctrico de la movilidad y  $m$  que es la curvatura o forma de la zona de la rodilla para la transición a saturación.

Los parámetros  $V_{aa}$ ,  $V_T$  y  $\gamma_a$  pueden ser calculados ajustando los datos experimentales  $I_{DS}-V_{GS}$  con un bajo  $V_{DS}$ . Se tiene el siguiente conocimiento de estos parámetros:

- $V_T$  recorre la curva completa.
- $V_{aa}$  brinda un factor de escala
- $\gamma_a$  toma en cuenta el efecto de campo eléctrico vertical en la movilidad.

Generaron las siguientes reglas:

1. Si la curva calculada se encuentra a la derecha de la experimental, se debe disminuir  $V_T$  o viceversa. Se realizaron incrementos o decrementos igual al 10% del valor máximo.
2. Si la curva calculada se encuentra sobre la experimental, se debe incrementar  $V_{aa}$  o viceversa.
3. Si la curva calculada es más curvada que la experimental, se debe incrementar  $\gamma_a$  o viceversa.

Después de que se obtienen los parámetros anteriores, los siguientes a extraer son  $\alpha_S$ ,  $\lambda$  y  $m$ . Se sabe que  $\alpha_S$  establece la relación entre  $V_{GS}$  y  $V_{DSat}$ .  $\lambda$  define la pendiente de la curva en saturación y  $m$  ajusta la forma de la rodilla de la región. Con lo anterior propusieron las siguientes reglas:

1. Si la curva calculada se satura muy rápido, se debe disminuir  $\alpha_S$  o viceversa.
2. Si la pendiente de la curva calculada en saturación es muy pequeña, se debe incrementar  $\lambda$  o viceversa.
3. Si la rodilla es muy puntiaguda o afilada, se debe reducir  $m$  o viceversa.

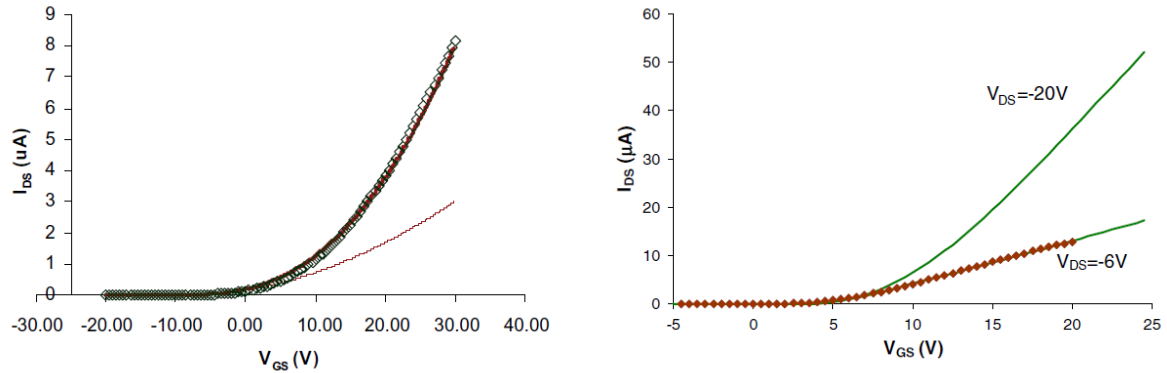


Figura 10. Comparación de curva experimental y la calculada dos OTFT  
Fuente: [19]

Se aplica el método de forma iterativa hasta que el error global es menor al 1%. Los OTFT usados son de dos tipos. El transistor T1 presenta  $W = 170\mu\text{m}$ ,  $L = 130\mu\text{m}$ , dieléctrico de compuerta de  $\text{SiO}_2$  con un espesor de 100nm. Para el transistor T2 se tiene una  $W = 500\mu\text{m}$ ,  $L = 50\mu\text{m}$ , dieléctrico de compuerta de PVP con un espesor de 120nm. El equipo utilizado para aplicar la extracción con el sistema difuso tiene las mismas características mencionadas anteriormente.

En la Figura 10 se presentan la comparación de las curvas calculadas con las experimentales/medidas (línea). En la parte superior se tiene el T1 usando un  $V_{DS} = -10\text{V}$  en las curvas reales, mientras que las dos calculadas presentaron  $V_{DS} = 0.1\text{V}$  y  $10\text{V}$ . El T2 en la parte inferior, las curvas experimentales tienen un  $V_{DS} = -6\text{V}$  y las calculadas presentaron  $V_{DS} = -6\text{V}$  y  $-20\text{V}$ . En las tablas 9 y 10 se presentan los parámetros que fueron extraídos y los resultados.

Tabla 9. Parámetros extraídos en T1

T1				
$W = 170\mu\text{m}$ , $L = 130\mu\text{m}$ , dieléctrico de compuerta de $\text{SiO}_2$ con un espesor de 100nm				
Parámetro	Valor inicial	Valor extraído	Método directo	
$V_t$	1.0	-11.2 V	-12.1	
$V_{aa}$	10 V	49.2 V	55.4	
$\gamma_a$	0.5	1.01	1.3	
$\lambda$	0.01	$6.27 \times 10^{-10}\text{V}^{-1}$	~0	
$\alpha_s$	0.5	0.383	0.40	
$m$	1.0	2.31	2.5	
Tiempo			19 s	

En este trabajo se concluye que fue aplicando un nuevo método para la extracción de parámetros usando lógica difusa, la cual hace uso del conocimiento de un humano experto,

esto tiene ventaja sobre otros métodos que deben aplicar cálculos como el jacobiano o ajustar funciones de una población de cientos de individuos como es el caso de los algoritmos genéticos, esto reduce la necesidad de poder de cómputo. Hacen mención que este método se puede aplicar a modelos de tipo BSIM, EKV o PSP.

Tabla 10. Parámetros extraídos en T2

T2				
$W = \mu\text{m}, L = 50 \mu\text{m}, \text{dieléctrico de compuerta de PVP con un espesor de } 120\text{nm}$				
Parámetro	Valor inicial	Valor extraído	Método directo	
$V_t$	1.0	-3.11 V	-3.9	
$V_{aa}$	10 V	1600 V	1400	
$\gamma_a$	0.5	0.17	0.15	
$\lambda$	0.01	$1.05 \times 10^{-4} V^{-1}$	$9.6 \times 10^{-5} V^{-1}$	
$\alpha_S$	0.5	1.25	1.4	
$m$	1.0	3.14	2.97	
Tiempo			21 s	

En el trabajo [19] se realizó la extracción/predicción de parámetros estáticos de un transistor bipolar de compuerta aislada usando redes neuronales artificiales. Los parámetros a predecir en este trabajo fueron: el voltaje de ruptura (BV), el voltaje de estado ( $V_{on}$ ) y el voltaje de bloqueo estático ( $V_{lu}$ ). Los autores propusieron el uso de las redes neuronales artificiales (ANN) multicapa para predecir dichos parámetros.

Como entradas, se utilizaron parámetros estructurales del dispositivo como N-drift doping ( $N_d$ ), N-buffer doping ( $N_b$ ), P-well doping ( $N_{well}$ ),  $P^+$  anode doping ( $N_{P^+}$ ), N-drift thickness ( $T_d$ ), N-buffer thickness ( $T_{buffer}$ ) y la longitud del canal ( $L$ ). Para un mejor entendimiento de estos parámetros, se presentan en Fig. 11a. La RN que utilizaron los autores conto con cuatro capas ocultas de 36, 20, 16 y 8 neuronas respectivamente, usando los 7 parámetros estructurales para predecir 5 parámetros eléctricos como salida (Fig 11b). Como resultados presentaron un porcentaje de error promedio del 3.98% para los 5 parámetros de salida al utilizar hasta 3750 muestras aproximadamente para el entrenamiento, de lo contrario obtienen un error promedio aproximado del 15%.



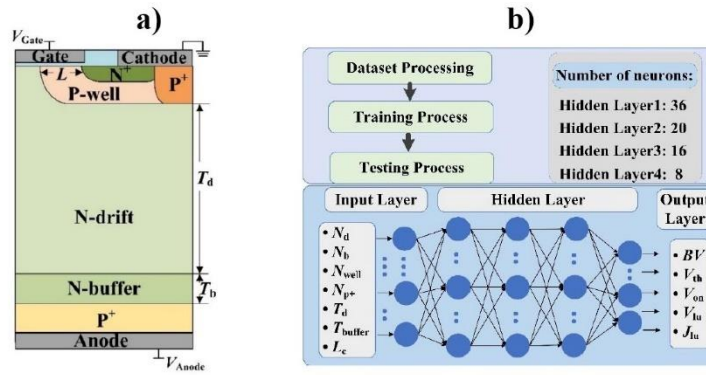


Figura 11. Características físicas y modelo de red  
Fuente: [19]

Este es uno de los trabajos propuestos más recientes, y de los que más se asemeja a lo propuesto en esta investigación. Aun así, hay notorias diferencias, los parámetros extraídos de este trabajo, con el que se propone, ya que en [19] se extraen ciertos voltajes, a partir de parámetros físicos como entradas, es decir se refieren a la estructura física del dispositivo.

En el trabajo [20] se realizó una clasificación de las características de transistores TFT IGZO, usando un enfoque de agrupamiento (K-means), el cual tenía como objetivo agrupar las curvas I-V en cuatro grados de rendimiento: alto, medio alto, medio y bajo. Dicha agrupación la realizaron acorde a las características de los transistores. Después del clustering los autores usaron redes neuronales profundas o DNN por sus siglas en inglés y las redes neuronales convencionales para extraer la movilidad eléctrica, el voltaje de umbral, la relación de corriente on/off y la pendiente subumbral. En la Tabla 11 se presenta un resumen de los porcentajes de error obtenidos en la extracción realizada en este trabajo.

Tabla 11. Porcentajes de error de obtenidos en [20]

	Movilidad	Voltaje umbral	Pendiente subumbral	Relación on/off
RNs	11.6%	28.8%	51.5%	96.3%
DNN	9.47%	14.1%	17.4%	26.9%

Una de las notables deficiencias de este trabajo es que los resultados se presentan solamente en tablas, con los resultados obtenidas de la extracción, pero no se cuenta con un valor objetivo, de esta manera no es posible tener certeza en que la extracción fue realmente exacta y no se presentan ejemplos de curvas I-V donde se comparen los comportamientos obtenidos de las diferentes extracciones.

En esta investigación se pretende identificar las fortalezas de los trabajos presentados en esta sección y adaptarlas al objetivo propuesto, también se pretenden identificar las desventajas para evitarlas y obtener un trabajo de investigación completo.

# Capítulo 3 Marco teórico

## 3.1 Transistores

El 23 de diciembre de 1947, Walter H. Brattain y John Bardeen presentaron el primer transistor en los laboratorios Bell el cual sería el remplazo del tubo de vacío y el triodo, elementos básicos en la televisión y la radio. El transistor, al ser un dispositivo de estado sólido con tres terminales era de menor tamaño y más ligero. Al contrario de los tubos de vacío, el transistor no tenía que calentarse ni perdía calor, además que necesitaba menos potencia [21], [22].

Como ya se mencionó el transistor es un dispositivo de tres terminales, una de ellas sirve para contralar a las otras dos, como si fuera una llave que controla el flujo de corriente entre las terminales. En la Figura 12, se presenta un diagrama general que describe el comportamiento de las tres terminales, donde la terminal C es la que controla la corriente  $i$  entre A y B, y se muestra una gráfica para tres valores de  $i_c$  o  $V_c$ . El comportamiento se puede describir con:

$$i = f(V_{AB}, V_C) \text{ o } i = f(V_{AB}, i_C) \quad (21)$$

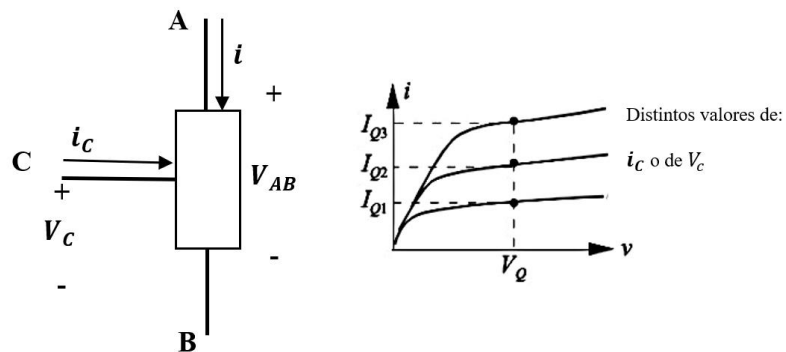


Figura 12. Diagrama general y curvas de un transistor

### 3.1.1. Tipos de transistores

El transistor se conforma de tres capas, dos de ellas de material tipo  $n$  y una tipo  $p$  o viceversa dos capas de material tipo  $p$  y una tipo  $n$ , denominando a los dispositivos como transistor  $nnp$  o transistor  $pnp$ . En la Figura 13 se muestra el diagrama de ambos

transistores, es importante notar que ya se etiquetan las terminales con sus nombres correspondientes, emisor (E), colector (C) y base (B), siendo la terminal base la que controla el comportamiento del dispositivo.

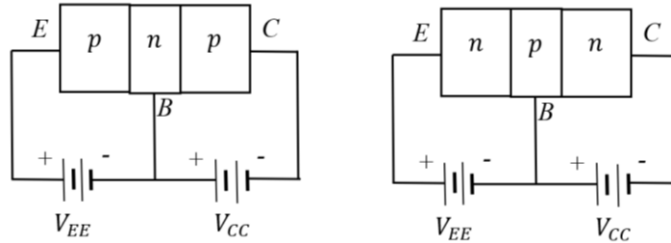


Figura 13. Diagrama de transistores pnp y npn

La clasificación o tipos de transistores está dada por el tipo de unión de los materiales con el que fue construido físicamente. A continuación, se presentan los tipos de transistores:

**Transistor bipolar:** también conocido solo como BJT (*Bipolar Junction Transistor*), recibe el nombre porque en él fluyen tanto electrones libres (material tipo *n*) como huecos (material tipo *p*) para generar la corriente eléctrica.

**Transistor unipolar:** también llamado transistor de efecto de campo o FET (*Field Effect Transistor*) ya que su funcionamiento depende del campo eléctrico. Por otro lado, es unipolar porque en él solo se mueve un tipo de carga, ya sean electrones o huecos.

Dentro de los transistores FET surgen dos tipos: el JFET, un transistor de efecto de campo de unión (en inglés *Junction Field Effect Transistor*), este puede controlarse con voltaje sin necesidad de una corriente de polarización y el MOSFET, su nombre viene de Metal-Óxido-Semiconductor, su compuerta metálica es separada del canal semiconductor por una capa de óxido. Dentro de estas clasificaciones han surgido nuevos tipos de transistores, los cuales adoptan las mismas características de los ya mencionados, pero con mejoras o nuevos materiales en su construcción, como lo es el TFT (*Thin-film transistor*) o transistor de película delgada es un tipo de MOSFET, que ha ganado mucha popularidad en su uso en pantallas de los dispositivos de hoy en día como Smartphones, Smart TV entre otros.

Debido a que en este trabajo se pretende utilizar un transistor TFT, solamente se agregaran las ecuaciones que describen el comportamiento de los transistores MOSFET. En la Figura 14 se presenta el símbolo utilizado para los transistores JFET, tipo N y P, donde se nombra a sus terminales en inglés D (*drain*), G (*gate*) y S (*source*) que la traducción directa

son drenador, puerta y fuente. Anteriormente se utilizaron los términos colectores, base y emisor, que es otra forma en la que son conocidos respectivamente. De igual manera se ilustran los voltajes y corrientes que circulan por el dispositivo.

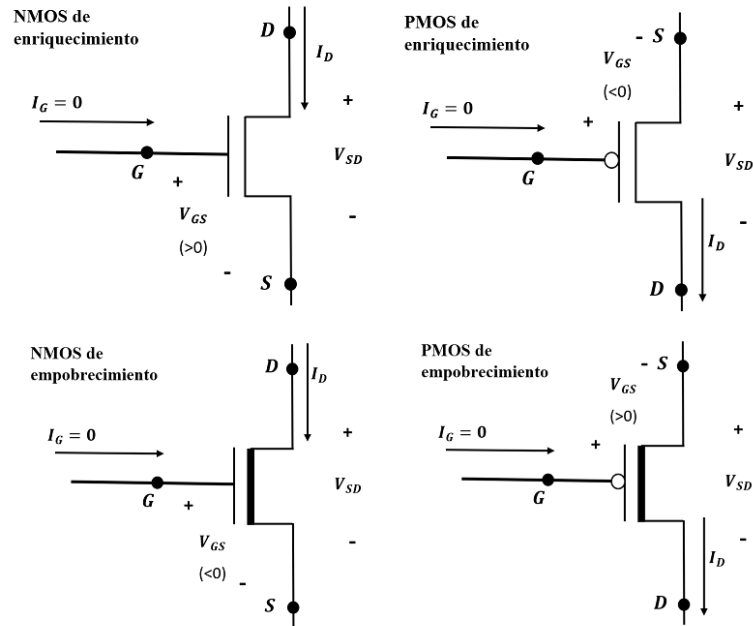


Figura 14. Símbolos de transistor MOSFET tipo P y N

En la Figura 15 se muestra la curva de transferencia de los transistores NMOS donde  $I_D = f(V_{GS}, V_{DS})$ . El transistor MOSFET permite tener valores  $V_{GS}$  positivos a diferencia del JFET y como se observa en la figura en el de enriquecimiento hay un desplazamiento  $V_T$ , que a partir de él la corriente eléctrica puede circular.

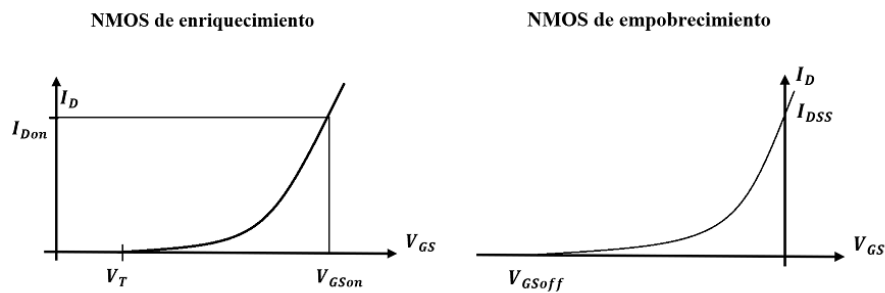


Figura 15. Curva de transferencia del NMOS

En la curva característica de los transistores se pueden destacar tres regiones: corte, óhmica y saturación. Cada una de ellas se presenta bajo ciertas condiciones de voltaje o

corriente, en la Figura 16 se presentan estas regiones de un NMOS de enriquecimiento [21], [22].

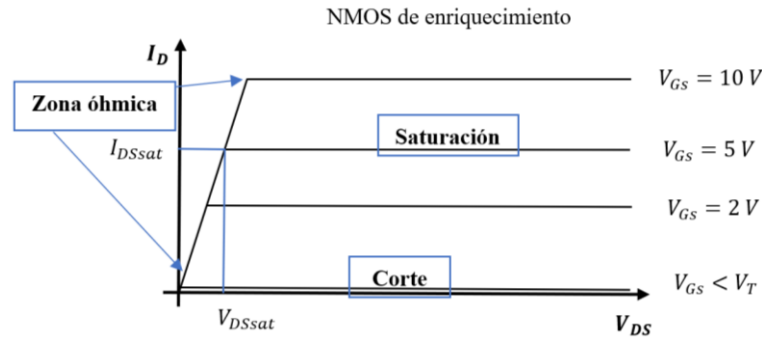


Figura 16. Regiones de un NMOS

El modelo matemático que describe el comportamiento del transistor MOS en las diferentes zonas se presenta a continuación:

*Zona de corte*

$$I_D = 0 \text{ cuando } V_{GSQ} \leq V_T \quad (22)$$

*Zona óhmica*

$$I_D = \frac{V_{DS}}{R_{DS}} \text{ cuando } \begin{cases} V_{GSQ} \geq V_T \\ V_{DSQ} \leq V_{DSsat} \end{cases} \quad (23)$$

*Zona de saturación*

$$I_D = \left( \frac{V_{GS} - V_T}{V_{GSon} - V_T} \right)^2 I_{Don} \text{ cuando } \begin{cases} V_{GSQ} \geq V_T \\ V_{DSQ} \geq V_{DSsat} \end{cases} \quad (24)$$

### 3.1.2. Circuito inversor

La creciente necesidad de superar las limitaciones que tienen los transistores MOSFET, ha empujado a la ingeniería a experimentar y desarrollar nuevos dispositivos con diferentes formas y materiales como son FD-SOIs, FinFets, Grafeno, Fets basados en nano cables y tubos de carbono entre otros. Una de las formas de probar y evaluar el funcionamiento de nuevos dispositivos es usándolos en compuertas lógicas ya que solo se debe configurar la conexión entre transistores. El inversor o compuerta *not* es de las más simples y utilizada por otras compuertas como la NAND y NOR [23].

Como su nombre lo indica, el circuito inversor genera una señal contraria a la que recibe, es decir, si recibe un voltaje bajo, en su salida habrá un voltaje alto y viceversa. En la Figura 17 se presenta la curva característica de un inversor.

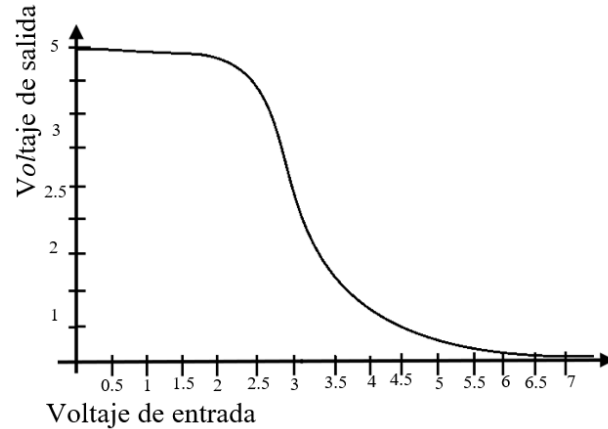


Figura 17. Curva de transferencia del inversor

Hay diferentes configuraciones de circuitos que tienen el comportamiento de invertir la señal que reciben de entrada, en este trabajo se analizará 2, el inversor de carga resistiva o pasiva y el inversor de carga activa. Ambos circuitos se pueden observar en la Figura 18. Donde a) es el inversor de carga resistiva y b) el de carga activa.

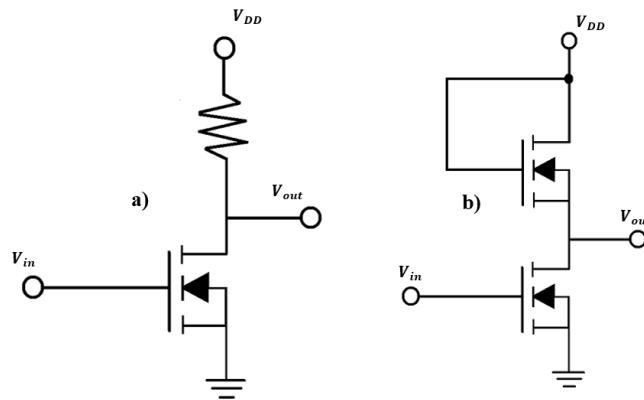


Figura 18. Circuitos inversores más comunes

Cuando el voltaje de entrada  $V_{in}$  no recibe voltaje el transistor se encuentra en corto, de modo que la salida  $V_{out}$  es igual al voltaje en  $V_{DD}$ . De lo contrario, cuando se aplica un voltaje en  $V_{in}$ , el transistor conduce y el voltaje en  $V_{DD}$  comienza a descender. La forma en la que cae depende del valor de la resistencia, entre más grande la pendiente aumenta. Este inversor fue sustituido por el inversor de carga activa ya que el anterior utiliza resistencias, las cuales son de un tamaño mucho mayor que un transistor. Al reemplazar la resistencia por un segundo transistor, el cual tiene conectada su base al drenador, fue posible disminuir drásticamente el tamaño del circuito, siendo más fácil la miniaturización de los circuitos.

### 3.2. Extracción analítica de $V_T$ en circuito inversor

En el caso del transistor de efecto de campo metal-óxido-semiconductor (MOSFET) de Si cristalino los procesos de extracción resultan relativamente sencillos, porque la dependencia de la corriente en el drenador  $I_D$  respecto al voltaje de compuerta es constante, como se muestra en las ecuaciones siguientes que describen el modelo de  $I_D$ :

$$I_{D \text{ lin}} = \mu C_{ox} \frac{W}{L} (V_G - V_T) \quad (25)$$

$$I_{D \text{ sat}} = \mu C_{ox} \frac{W}{L} (V_G - V_T)^2 \quad (26)$$

$I_{D \text{ lin}}$  es la corriente en el drenador en la región lineal, mientras que  $I_{D \text{ sat}}$  es la corriente del drenador en la región de saturación. También puede contraerse como:

$$I_{D \text{ lin}} = K(V_G - V_T) \quad (27)$$

$$I_{D \text{ sat}} = k (V_G - V_T)^2 \quad (28)$$

Siendo  $K = \mu C_{ox} \frac{W}{L}$

Donde:

- $\mu$  es la movilidad efectiva de los portadores de carga.
- $C_{ox}$  es la capacidad del óxido por unidad de área.
- $W$  es el ancho de la puerta.
- $L$  es la longitud de la puerta.

En este tipo de dispositivos los procedimientos analíticos de extracción de parámetros proporcionan resultados aproximados que posteriormente son ajustados con base a la experiencia de los diseñadores. En el caso de los TFT aun cuando son un tipo de transistor MOSFET el proceso de extracción analítica se vuelve complejo, debido a que la dependencia de la  $I_D$  respecto al  $V_{GS}$  no es constante, como se puede observar en el modelo de  $I_D$  de TFTs (ecuaciones 29 y 30). La  $I_D$  tiene una dependencia exponencial de  $\gamma + 1$  o  $\gamma + 2$ , según sean las condiciones de polarización.

$$I_{D \text{ Lin}} = K(V_G - V_T)^{\gamma+1} \quad (29)$$

$$I_{D \text{ Sat}} = k (V_G - V_T)^{\gamma+2} \quad (30)$$



Debido a que la extracción analítica sobre un circuito inversor de carga activa se complica debido al modelo del transistor. Se muestra como ejemplo la extracción de  $V_T$  en el transistor de carga resistiva.

Inicialmente se establece que para valores de voltaje de entrada ( $V_{in}$ ) bajos, el voltaje de salida de inversor ( $V_{out}$ ) será alto, cercano al valor de  $V_{DD}$  utilizado.

En base al circuito de la Fig. 18a,  $C_{DS} = V_{OUT}$  y  $V_{GS} = V_{in}$ , por lo que para las condiciones anteriores:

$$V_{DS} = V_{out} = V_{DD} \quad (31)$$

Por lo tanto,  $V_{DS} \geq V_{GS} - V_T$ , es decir, para voltajes de  $V_{IN}$  bajos el transistor se encontrará en régimen de saturación. En estas condiciones la corriente del transistor estará definida por la ecuación:

$$I_{DS-sat} = \mu_{FET} C_{OX} \frac{W}{L} \frac{(V_{GS} - V_T)^2 \alpha_{sat}}{2} \quad (32)$$

Se asume por simplicidad que  $\alpha_{sat} = 1$  y se considera que  $V_{GS} = V_{in}$  la ecuación se expresa como:

$$I_{DS-sat} = \mu_{FET} C_{OX} \frac{W}{L} \frac{(V_{in} - V_T)^2}{2} \quad (33)$$

La corriente eléctrica que circula por la resistencia de carga ( $R_L$ ) está definida por la ley de Ohm, esto es:

$$I_R = \frac{V_R}{R_L} \quad (34)$$

El voltaje  $R_L$  sería igual a  $V_R = V_{DD} - V_{out}$ , por lo que la expresión de la corriente en la resistencia de carga quedaría:

$$I_R = \frac{V_{DD} - V_{out}}{R_L} \quad (35)$$

Como la resistencia de carga y el transistor están conectados en paralelo, la corriente que circula por ambos dispositivos es la misma, esto es:

$$I_R = I_{DS-sat} \quad (36)$$

$$\frac{V_{DD} - V_{out}}{R_L} = \frac{\mu_{FET} C_{OX} W}{2L} (V_{in} - V_T)^2 \quad (37)$$

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

$$V_{DD} - V_{out} = \frac{R_L \mu_{FET} C_{OX} W}{2 L} (V_{in} - V_T)^2 \quad (38)$$

Se aplica la raíz cuadrada en ambos miembros, para linealizar la ecuación.

$$\sqrt{V_{DD} - V_{out}} = \sqrt{\frac{R_L \mu_{FET} C_{OX} W}{2 L}} (V_{in} - V_T) \quad (39)$$

Esta es la expresión de una recta, en donde la pendiente de la recta ( $m$ ) es igual a:

$$m = \sqrt{\frac{R_L \mu_{FET} C_{OX} W}{2 L}} \quad (40)$$

Como se puede observar, el voltaje de umbral se puede extraer a partir del intercepto de la recta con el eje  $x$ . Esto es, cuando  $V_{in}$  sea igual a  $V_T$ , el valor de  $\sqrt{V_{DD} - V_{OUT}}$  será igual a cero. De esta forma calcular el valor de  $V_T$  es razonablemente sencillo.

Por otro lado, de la pendiente de la recta es posible encontrar el valor del producto  $R_L \mu_{FET}$ . Sin embargo, con este procedimiento analítico no es posible obtener los valores de resistencia de carga y movilidad de forma individual. Sería necesario implementar un procedimiento de optimización matemática o realizar otras mediciones para poder hacer la extracción.

### 3.3. Redes neuronales artificiales (RNAs)

Las redes neuronales (RNs) puede definirse como un conjunto de elementos (llamadas neuronas) conectados entre sí para resolver problemas complejos no lineales o bien en donde se desconoce el modelo matemático. Surgen como una forma de imitar el funcionamiento de aprendizaje del cerebro [24].

Warren McCulloch y Walter Pitts, propusieron el modelo de una neurona simple en 1943. Más adelante las RNs fueron trabajadas por Windrow y Hoof, quienes modificando el trabajo previo de MacCulloch y Pitts, desarrollaron la red llamada Adaline, por su forma de aprendizaje (*Adaptative linear element*).

Las RNs perdieron fuerza, ya que al término de los 70's se demostró que la neurona simple no podría resolver el problema lógico XOR. Dicho problema fue resultado teniendo estructuras de redes más complejas (más neuronas) [25].

### 3.3.1 Modelo de la neurona

El modelo de una sola neurona es conocido como perceptrón, o perceptrón simple ya que existe la multicapa. En la Figura 19 se tienen la representación de una neurona. Donde se tiene una sola entrada  $p$  que es multiplicado por una  $w$  (representación de la conexión sináptica biológicamente) a esto se le suma una bias ( $b$ ), obteniendo un resultado preliminar  $n$ . El resultado  $n$  es evaluado en una función de transferencia  $f$  (esta se selecciona dependiendo del problema y tipo de datos utilizados), dando así la salida de la neurona  $a$ .

Colocando lo anterior de manera formal, se puede representar la salida de una neurona como:

$$a = f(wp + b) \quad (41)$$

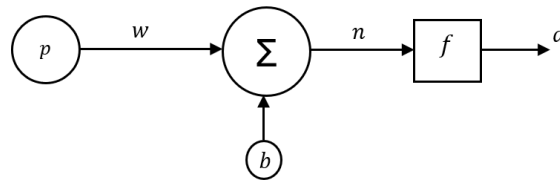


Figura 19. Modelo de una sola neurona y una entrada

Ejemplo:

Suponga que  $p = 4$ ,  $w = 2$  y  $b = 0.5$ . La salida de la red es:

$$a = f(2(4) + 0.5) = f(8.5)$$

El valor final de  $a$  depende de la función de transferencia que se va a utilizar. Si se utiliza la función llamada como *hard limit*  $a = 1$ .

### 3.3.2 Neurona con múltiples entradas

Normalmente los problemas existentes en la vida real tienen más de una variable a analizar, por lo que es muy común contar con más de una entrada en un RN. En la figura 8 se presenta el modelo de una neurona con varias entradas  $p_k$ . Donde cada elemento del vector  $p$  multiplica con el elemento correspondiente del vector de pesos  $w$ . Agregando un bias ( $b$ ) a la suma de todos los productos de cada elemento de  $p$  con  $w$ , se obtiene la salida preliminar  $n$ . Para obtener la salida total  $a$ , se debe evaluar  $n$  en la función de transferencia seleccionada [26], [27].

Formalmente se puede describir el modelo de la neurona (Fig. 20) como:

$$n = p_1w_1 + p_2w_2 + p_3w_3 + \dots + p_kw_k + b \quad (42)$$

Viendo la expresión anterior en su forma matricial se tiene:

$$n = Wp + b \quad (43)$$

Por último, la evaluación en la función de transferencia es:

$$a = f(Wp + b) \quad (44)$$

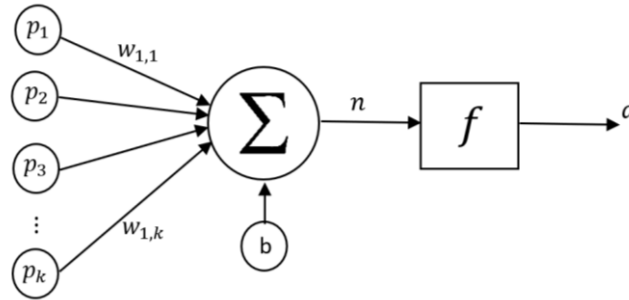


Figura 20. Neurona con más de una entrada

### 3.3.3 Red multicapa

Los investigadores observaron las limitantes que una sola neurona tenían, de manera que con el tiempo surgen las redes multicapa o perceptrón multicapa. Las capas son un conjunto de neuronas ordenado en forma de columna, conectadas a otro conjunto de neuronas (igualmente en columnas), teniendo un número  $c$  de capas en una red. Cada capa puede tener diferente número de neuronas en ella. La figura 21 muestra una forma general de una red con tres capas. Es importante notar que los pesos ya forman a una matriz  $W$ , cada elemento de las columnas de  $W$  pertenecen a una capa de la red. Lo mismo sucede con el bias, cada neurona cuenta con uno, de manera que se tiene una matriz  $B$ , donde cada columna pertenece a una capa.

De la Figura 21, se pueden obtener las salidas de cada capa como:

$$a^1 = f^1(w^1p + b^1) \quad (45)$$

$$a^2 = f^2(w^2a^1 + b^2) \quad (46)$$

$$a^3 = f^3(w^3a^2 + b^3) \quad (47)$$

En  $a^1$  el vector  $p$  son las entradas de la red, esto cambia para las siguientes capas, ya que la entrada de la capa 2 es la salida de la capa 1. Lo mismo sucede en la capa 3, su entrada es la salida de la capa 2. La salida de la capa final se puede expandir como:

$$a^3 = f^3(w^3(f^2(w^2(f^1(w^1p + b^1)) + b^2)) + b^3) \quad (48)$$

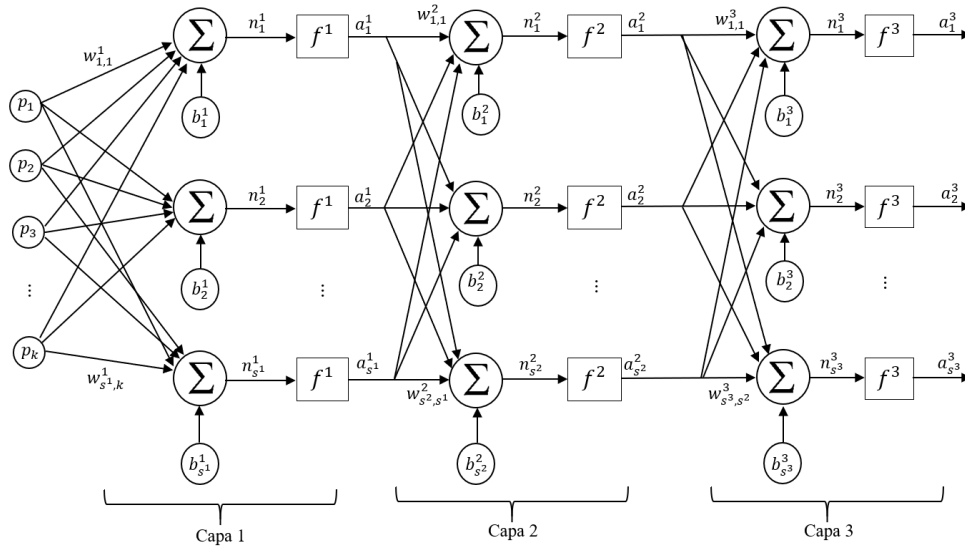


Figura 21. Red Multicapa

### 3.3.4 Entrenamiento de las RNAs

El entrenamiento o aprendizaje de las RNAs es fundamental, ya que aquí es donde se define si la red es capaz de resolver un problema. Las RNAs necesitan datos (patrones) para poder aprender de ellos, a estos datos se le conoce como conjunto de entrenamiento. Aunque normalmente dichos patrones se dividen en tres conjuntos, entrenamiento, pruebas y validación (se especificaran más adelante). El conjunto de entrenamiento debe contar con dos características:

**Significativo:** el conjunto debe contar con un gran número de ejemplos para que la red tenga suficiente información para aprender. Si los datos son insuficientes el aprendizaje podría ser escaso.

**Representativo:** los ejemplos o patrones deben ser variados. Si se tienen un mayor número de ejemplos para una salida, la red aprenderá únicamente para esos ejemplos, presentando un menor rendimiento para los demás tipos de ejemplos. Dicho de otra forma, los datos deben estar balanceados.

#### 3.2.4.1 Entrenamiento supervisado

En el entrenamiento supervisado se le presentan a la red dos tipos de datos, el patrón/ejemplo de entrada y la salida deseada correspondiente a dicha entrada. De manera

que cuando la entrada es procesada por toda la red se obtiene la salida  $a$  (secciones anteriores). Se compara la salida de la red con la salida real  $a_r$ . Si el error entre las salidas, la red modifica los valores de los pesos  $w$ , con la finalidad de reducir el error. La representación de este entrenamiento se puede observar en la Figura 22.

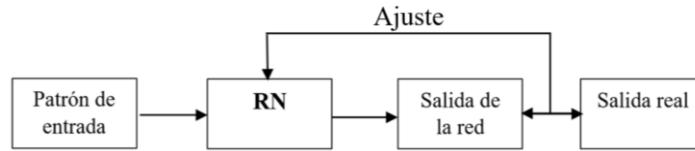


Figura 22. Entrenamiento supervisado

### 3.2.4.2 Entrenamiento no supervisado

En este entrenamiento no se brinda la salida deseada a la red, de modo que no es posible determinar si la salida generada es correcta. La red adapta los pesos  $w$  con la información que circula dentro de ella. Con el entrenamiento no supervisado el objetivo es encontrar las características de los datos. La Figura 23 representa este entrenamiento [25]–[27].

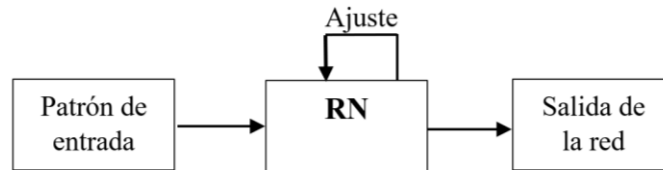


Figura 23. Entrenamiento no supervisado

Debido a que los métodos de aprendizaje como lo son las RN, y como se verá más adelante con RF y SVR. Estos aprenden de ejemplos y en este caso, los ejemplos son curvas I-V. Las curvas deben estar etiquetadas, es decir, para cada una de ellas se deben conocer las salidas que le corresponden (parámetros). Usando un software de simulación electrónico permite generar las muestras necesarias para los entrenamientos de los métodos, y al establecer las características y condiciones de la simulación, los parámetros son conocidos, lo que permite el enfoque de aprendizaje supervisado. En el caso de utilizar mediciones físicas de dispositivos, no siempre se conocen los valores de sus parámetros, por lo cual sería necesario aplicar el enfoque no supervisado. Para esta investigación solamente se estudiará el enfoque de aprendizaje supervisado, utilizando mediciones de las cuales se sabe cuáles son los parámetros de cada ejemplo para realizar los entrenamientos.

### 3.4. Árboles de decisión

Los árboles de decisión son métodos de aprendizaje utilizados para la clasificación y regresión. Su funcionamiento se basa en la toma de decisiones del tipo si/no a ciertas preguntas. Por ejemplo, si se desea saber en qué animal está pensando una persona, y las posibles respuestas son: perro, oso o un ave. Se comienza a realizar una serie de preguntas como ¿vuela?, ¿es más grande que una persona? De esta manera se forma un árbol de decisión como se muestra en la Figura 24 [28], [29].

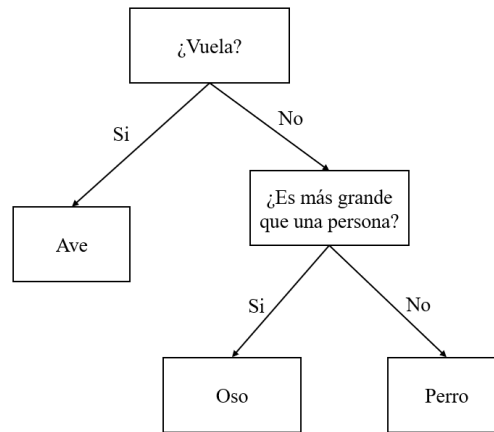


Figura 24. Ejemplo de árbol de decisión

En la vida real los datos de entrenamiento no vienen organizados en forma de preguntas, de manera que el algoritmo debe identificar las características de los datos que permiten hacer la mejor clasificación.

#### 3.4.1. Árbol de clasificación

Para construir un árbol se debe identificar en cuál de los predictores (los predictores son los elementos de cada ejemplo de entrada) se debe realizar un corte para la creación de una rama. Para ello se debe encontrar la combinación de divisiones de manera que brinda la mayor ganancia de información.

Para encontrar cuál de los predictores brinda más información, se define una función objetivo para optimizar. La función es la siguiente:

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^m \frac{N_j}{N_p} I(D_j) \quad (49)$$

Donde  $f$  es el predictor o característica donde se hace la división,  $D_p$  y  $D_j$  es el conjunto de datos  $j$  es el nodo hijo,  $I$  es la medida de impureza.  $N_p$  es el total de muestras en el nodo padre y  $N_j$  es el total de muestras en el nodo hijo. La ganancia de información será inversamente proporcional a la impureza del nodo hijo [30]–[32].

Para reducir el espacio de búsqueda es posible aplicar arboles de decisión binarios, de forma que los nodos padres se dividen en solo dos nodos descendientes (nodo izquierdo y nodo derecho) de la manera siguiente:

$$IG(D_p, f) = I(D_p) - \frac{N_{izq}}{N_p} I(D_{izq}) - \frac{N_{der}}{N_p} I(D_{der}) \quad (50)$$

De esta manera la impureza y la división son conocidos como la impureza Gini ( $I_G$ ), la entropía ( $I_H$ ) y el error de clasificación ( $I_E$ ). Donde la entropía está definida por:

$$I_H(t) = - \sum_{i=1}^c p(i|t) \log_2 p(i|t) \quad (51)$$

Donde  $p(i|t)$  es la cantidad de muestras que pertenecen a la clase  $c$  de cierto nodo  $t$ . Entonces, si todas las muestras pertenecen a la misma clase, el valor de la entropía será cero, y se dice que la entropía es máxima cuando se tiene una distribución uniforme en las clases. La ecuación para de la impureza Gini es la siguiente:

$$I_G(t) = \sum_{i=1}^c p(i|t)(1 - p(i|t)) = 1 - \sum_{i=1}^c p(i|t)^2 \quad (52)$$

El error de la clasificación también puede ser usado como una medida de la impureza y está dado por:

$$I_E = 1 - \max\{p(i|t)\} \quad (53)$$

### 3.4.2 Árboles de regresión

Los árboles de regresión es una variante de los árboles de clasificación y se utilizan cuando la salida es una variable continua. Para la construcción de un árbol de regresión también se deben encontrar los predictores en donde se crearán las ramas. Los predictores definen las regiones  $R_1, R_2, R_3, \dots, R_j$ . Con las cuales se calcula la suma residual de los cuadrados, por sus siglas en inglés (RSS).

Se debe encontrar la combinación de regiones  $J$ , que obtenga el menor RSS, que se define como:

$$RSS = \sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2 \quad (54)$$



Donde  $\hat{y}_{R_j}$  es la media de la variable de salida en la región  $R_j$ . Para reducir el espacio de búsqueda también se aplica la división binaria, con el cual se encuentra cada variable de entrada  $X_j$  y el punto para crear una rema  $s$ , de esta manera se pueden distribuir las muestras del conjunto de entrenamiento en  $\{X|X_j < s\}$  y  $\{X|X_j \geq s\}$  [30], [31], [33].

### 3.4.3 Ensamblados de árboles de decisión

Un ensamble es la combinación de diferentes modelos de aprendizaje, con la finalidad de obtener un modelo más robusto y con mayor exactitud. En la literatura se pueden encontrar una gran cantidad de métodos de aprendizaje, algunas variaciones y mejoras de otros, pero los ensambles que han demostrado obtener buenos resultados en clasificación y regresión son los que están conformados por árboles de decisión, a continuación, se presentan Random Forest y Gradient Boosting.

#### 3.3.3.1 *Random Forest*

Random forest es un ensamble conformado por un conjunto de árboles de decisión, cada uno de ellos entrenado con una muestra diferente del conjunto de entrenamiento. El principio es que si se tienen diferentes árboles con desempeño y sobre entrenamiento diferentes, se puede obtener un resultado promedio de todos ellos, reduciendo de forma general el sobre entrenamiento y mejorando la exactitud del resultado.

Por mencionar algunas de las ventajas del Random Forest es que no tiene problemas de sobre entrenamiento sin importar cuantos árboles se agreguen. Por otra parte, en la etapa de programación la mayoría de los hiperparámetros del modelo se pueden configurar en su valor por default, siendo el número de árboles el parámetro en el que se debe encontrar su valor óptimo.

#### 3.3.3.2 *Gradient Boosting*

El Gradient Boosting (GB) es otro ensamble construido por un conjunto de árboles de decisión, pero la diferencia entre GB y RF es la forma en la que se construyen los árboles. En GB los árboles se van creando de manera secuencial, uno tras otro, donde cada árbol nuevo disminuye los errores del árbol pasado. El principio es que al combinar árboles simples que en GB se les conoce como aprendiz débil o *weak learners*, cada uno de ellos con con

buenas predicciones de una muestra del conjunto de entrenamiento, al agregar más árboles se mejora el modelo [31], [34].

### 3.5 Support Vector Regression

Support Vector Regression o Regresión de soporte vectorial es la variante de la Support Vector Machine (SVM) cuando la variable de salida es continua. Ambos son otro método de aprendizaje supervisado, en la literatura lo consideran la mejora del perceptrón. El algoritmo de la SVM tiene como objetivo mejorar un margen, dicho margen es la distancia que separa las muestras del conjunto de entrenamiento con un hiperplano. La Figura 25 representa el funcionamiento básico [35], [36].

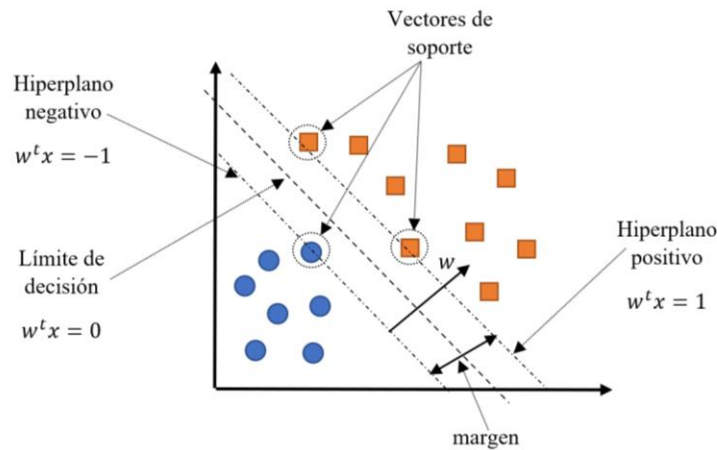


Figura 25. Funcionamiento de la SVM

Cuando se encuentran márgenes grandes, el error del aprendizaje es menor por consecuencia el desempeño en la predicción es alta, por el contrario, cuando se tienen errores cortos el modelo presenta un sobre entrenamiento.

Para resolver el problema de clases que se pueden separar linealmente de la forma  $C = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ , donde  $x_i \in \mathbb{R}^d$  y  $y_i \in \{+1, -1\}$ , es posible establecer el hiperplano con una función lineal como:

$$D(x) = (w_1x_1 + w_2x_2 + \dots + w_dx_d) + b \tag{55}$$

Donde  $w_i$  y  $b \in \mathbb{R}$ .

En el caso de un problema de regresión donde se tiene el mismo conjunto  $C$ , pero  $y_i \in \mathbb{R}^d$  la ecuación para encontrar el hiperplano se define de la siguiente manera:

$$\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*) \tag{56}$$

Donde  $C$  es una constante mayor que cero y da equilibrio entra la función  $f$  y la tolerancia de desviación mayor que  $\varepsilon$ . Las variables  $\xi$  permiten cierto rango de error con los datos de salida  $y_i$ . En la Figura 26 se puede observar el ejemplo del funcionamiento de la SVR.

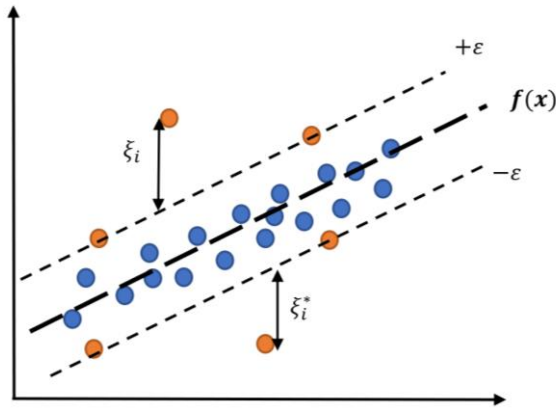


Figura 26. Funcionamiento de la SVR

Si la función a predecir no es continua y se tiene el conjunto  $C = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ , donde  $x_i \in \mathbb{R}^d$  y  $y_i \in \mathbb{R}$ . Suponiendo que  $\Phi: \mathbb{X} \rightarrow F$  sea la función perteneciente a cada elemento de  $x$ . Donde  $F$  es un espacio Hilbert (una generalización del espacio Euclidiano), dicho espacio podría ser de alta dimensión o hasta infinita.

Entonces la ecuación que se necesita para aproximar la función  $F$  se define de la siguiente forma:

$$f(x) = [w, \Phi(x)] + b \quad (57)$$

Cuando la dimensión del problema es muy grande se utiliza el Kernel, el cual brinda a cada par de elementos del conjunto de entrada  $X$ , un valor real que pertenece al producto escalar de las salidas  $y_i$  de cada elemento  $x_i$  en un nuevo espacio, de la manera:

$$K(x, x') = [\Phi(x), \Phi(x')] \quad (58)$$

Existen diferentes funciones Kernel, actualmente las más utilizadas son aquellas que han comprobado funcionar para resolver cierto tipo de problemas. El buen desempeño que realice el Kernel dependerá directamente de la naturaleza de los datos a los que se apliquen. A continuación, se presentan las funciones Kernel más comunes.

Kernel polinomial definido como:

$$K(x, z) = p(K_1(x, z)) \quad (59)$$

Kernel base radial Gaussiana definido como:

$$K(x, x') = \exp(-\gamma \|x - x'\|^2) \quad (60)$$

Kernel base radial exponencial definido como:

$$K(x, x') = \exp\left(-\frac{\|x-x'\|}{2\sigma}\right) \quad (61)$$

Kernel sigmoidal definido como:

$$K(x, x') = \tanh(\gamma(x, x') + R) \quad (62)$$

En caso de requerir mayor información sobre el capítulo de marco teórico es posible utilizar las referencias [21-36]. En la investigación se reporta brevemente cada tema con el objetivo de no extender el documento.

# Capítulo 4 Enfoque propuesto

## 4.1. Transistor NMOS

En esta sección se presenta el análisis del efecto que tienen los parámetros de voltaje de umbral ( $V_T$ ), movilidad de superficie ( $U_0$ ) y la resistencia de superficie ( $R_s$ ) en el transistor NMOS. También se presenta cómo se realizó la generación del conjunto de datos para el entrenamiento de los métodos de aprendizaje.

### 4.1.1. Análisis de parámetros y como afectan al NMOS

En esta subsección se presenta un breve análisis de los parámetros de interés a extraer del transistor NMOS, esto con el objetivo de mostrar como estos parámetros afectan al comportamiento del dispositivo y por qué fueron seleccionados para su extracción.

#### *Voltaje de umbral*

Como se definió anteriormente  $V_T$  en 1.3, lo convierte en uno de los parámetros más importantes ya que define el momento de la conducción del transistor. A continuación, se presenta la Figura 27, donde realizó un barrido en el parámetro, desde  $-2V$  hasta  $2V$  a razón de  $1V$ .

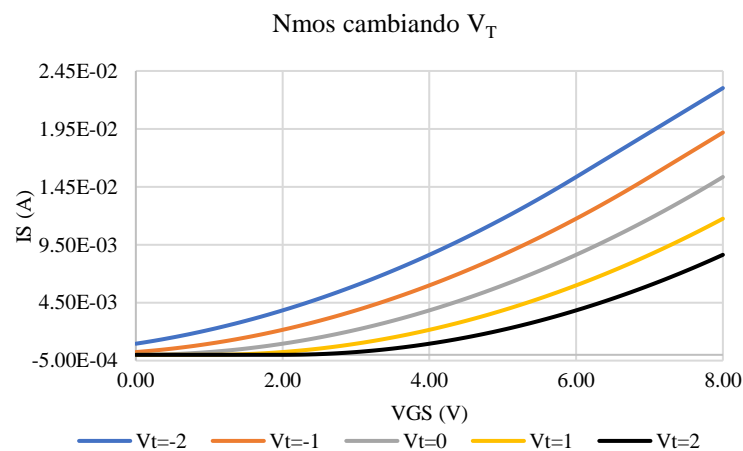


Figura 27. NMOS cambiando  $V_T$

A simple vista parece que la corriente  $I_S$  disminuye conforme aumenta  $V_T$ , pero no es así. El voltaje de umbral no modifica la cantidad de corriente, solo desplaza la curva de transferencia de izquierda o derecha según sea su valor. En este caso, la curva de color azul es la misma que la de color negro, solo que esta esta desplazada a la izquierda, ya que la corriente comenzó a moverse a partir de  $-2V$ , y la de color negro comenzó hasta los  $2V$ .

### **Movilidad de superficie**

La movilidad de superficie es un parámetro que, si modificará la curva de transferencia del NMOS, ya que la corriente eléctrica será directamente proporcional a  $UO$ . Con valores de movilidad altos, mayor será el flujo de corriente circulando en el dispositivo. Esto se observa en la Fig. 28, donde el  $V_T$  se mantiene fijo en  $1V$  y en todas las curvas la conducción inicia en el mismo punto, pero el flujo de la corriente eléctrica es diferente para cada una. Donde  $I_S \sim 0.0039A$  cuando  $UO = 100cm^2/Vs$  y se alcanza una corriente máxima de  $I_S \sim 0.019A$  cuando  $UO = 500cm^2/Vs$ . El nivel máximo de conducción que se puede tener dependerá del material del que sea fabricado el dispositivo. Por ejemplo, el silicio (Si) tiene una movilidad máxima de  $1350cm^2/Vs$  o el Arseniuro de Galio (AsGA) de hasta  $8500cm^2/Vs$  [37].

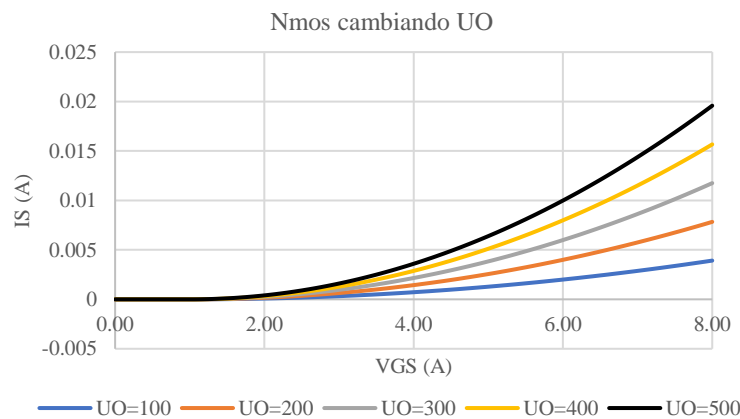


Figura 28. NMOS cambiando UO

### **Resistencia de superficie**

Como su nombre lo indica el parámetro  $R_S$  indica la resistencia que opondrá el material ante la corriente eléctrica, idealmente un transistor no debería presentarla, pero en la práctica esto no es así. Dependiendo del material se tendrá mayor o menor resistencia y la corriente eléctrica será inversamente proporcional a ella. Esto se observa en la Figura 29 en donde se

hace un barrido en  $R_S = 0\Omega$  hasta  $R_S = 400\Omega$ , obteniendo una corriente mínima de  $I_S \approx 0.005$  A en  $400\Omega$  y hasta  $I_S \approx 0.011$  A.

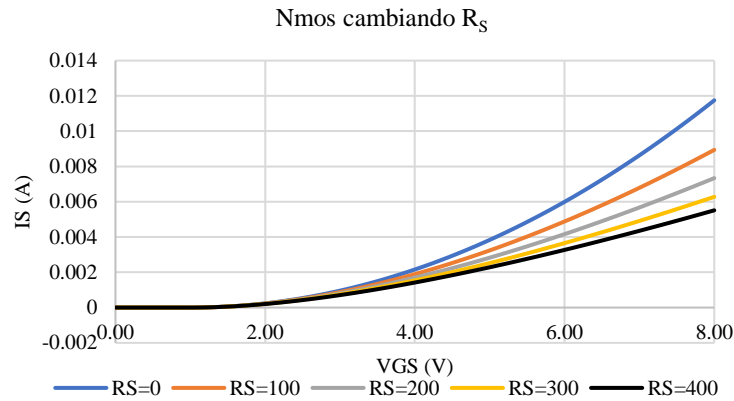


Figura 29. NMOS cambiando  $R_S$

#### 4.1.2. Extracción de 3 parámetros del NMOS

En esta sección se presenta como se realizó la generación del conjunto de datos para el entrenamiento, los rangos que se tomaron dentro de las simulaciones y el preprocesamiento que se les aplicó para la extracción en transistores NMOS. Así mismo se presenta la búsqueda en malla que se realizó para modificar los hiperparámetros de los métodos de aprendizaje. Los resultados obtenidos de esta extracción se encuentran en la sección 5.1.

##### *Generación del conjunto de datos*

Para realizar la simulaciones y crear el conjunto de datos del NMOS se utilizó el software LTspice XVII [38]. En el cual se realizaron un conjunto de configuraciones en los parámetros  $V_T$ ,  $R_S$  y  $U_O$ . El rango que tomaron los valores en los parámetros se encuentra en la Tabla 12. Obteniendo un total de 540 curvas de ejemplo diferentes para el aprendizaje.

Tabla 12. Rangos en los parámetros del NMOS

Parámetro	Valor inicial	Valor final	Unidades	Incremento
$U_O$	100	900	$\text{cm}^2/\text{Vs}$	100
$V_T$	-3	3	V	0.5
$R_S$	0	500	$\Omega$	100

El preprocesamiento que se realizó en las curvas de transferencia para reescalar los datos y mantener valores entre  $-1$  y  $1$  se llevó a cabo dividiendo cada vector de entrada/salida entre el valor máximo que este podía tener. Por ejemplo, para reescalar la salida  $U_O$ , se

dividió entre 900. El patrón de entrada correspondiente al voltaje  $V_{GS}$  fue dividido entre 8V, y así para cada elemento de entrada o salida para la red.

Debido a que se cuentan con mediciones experimentales en régimen de saturación (T1) y en régimen lineal (T2), se llevaron simulaciones para crear dos conjuntos de datos, el conjunto correspondiente a T1 se utilizó un  $V_{DS}= 8 \text{ V}$  y un  $V_{DS}= 0.01 \text{ V}$  para T2. En la Tabla 13 se presentan las dimensiones de ambos transistores.

Tabla 13. Dimensiones de los dispositivos NMOS

Dispositivo	Ancho del canal (W)	Largo del canal (L)	Espesor de oxido ( $T_{ox}$ )
T1	100 $\mu\text{m}$	8 $\mu\text{m}$	27nm
T2	100 $\mu\text{m}$	16 $\mu\text{m}$	32 $\mu\text{m}$

Una vez que los conjuntos de datos estuvieron reescalados y divididos en conjunto de entradas y conjunto de salidas, fueron utilizados por el algoritmo de los AD, el cual se encargó de identificar los elementos del vector entrada que fueran más significativas, es decir, aquellas que brindan mayor información para predecir la salida deseada. La selección de características permite reducir la dimensión de los datos, como resultado se obtienen entrenamientos más rápidos, por otro lado, al utilizar las entradas más importante se aumenta la probabilidad de tener una mayor exactitud por parte de los métodos de aprendizaje automático [28], [36], [39].

### ***Entrenamiento de los métodos de aprendizaje***

Al igual que en la sección 4.1.3.2, aquí se presenta como se realizó el entrenamiento de los métodos de aprendizaje. En esta ocasión utilizando dos conjuntos de datos pertenecientes a dispositivos NMOS (Tabla 19) que cuanta únicamente con datos de entrada significativos.

Para el aprendizaje se seleccionaron a las RN, RF y SVR, ya que los AD fueron utilizados para la selección de características. Como ya se mencionó, para el entrenamiento de los métodos de aprendizaje es necesario experimentar con diferentes estructuras de los mismo para encontrar cual es la que tiene un mejor aprendizaje y sin llegar al sobre entrenamiento.

En esta subsección se presentan los hiperparámetros de cada uno de los métodos y sus rangos que se tomó en el *Search grid*. Los hiperparámetros configurados son los mismos que en 4.1.3.2.1, pero en esta ocasión algunos valores fueron descartados ya que se sabe que no



brindan mejores resultados. La Tabla 14 muestran los hiperparámetros configurados durante el *Search grid*. Donde solo se probaron redes con dos capas ocultas, cada una con un número de neuronas que variaron desde 32 hasta 512 neuronas, no se probaron redes más grandes ya que anteriormente se estableció que el aprendizaje de la red no mejoraba al ser más grande. El número de épocas máximo fue de 2000, ya que, a partir de esta cantidad, el aprendizaje se mantenía constante.

Tabla 14. Search grid para RN del NMOS

Capas ocultas	No. neuronas	Función de activación	Learning rate	Épocas
1	64,128,256 y 512	Linear, relu y tanh	0.1, 0.01, 0.001 y 0.0001	500, 1000, 1500 y 2000
2	32,64 y128			

En la Tabla 15 se muestran los hiperparámetros configurados en el *search grid* del RF, donde se tomó una profundidad máxima para los árboles desde 3 hasta 25, el número de nodos terminales fue de 100 hasta 250 nodos, el número mínimo de muestras que un nodo pueda tener para que se divida fue de 2 a 4, el número mínimo de muestras en los nodos hijos para que haya una división fue de 2 a 5 y la cantidad de árboles que conformaron al bosque fue desde 10 hasta 250.

Tabla 15. Search grid para RF del NMOS

<b>max_depth</b>	<b>max_leaf_nodes</b>	<b>min_samples_split</b>	<b>min_samples_leaf</b>	<b>No_estimators</b>
3, 10, 15, 20, 25	100, 120, 150 y 250	2, 3 y 4	2, 3, 4 y 5	10, 30, 50, 120, 250

En la Tabla 16 se muestran los hiperparámetros y kernels que se utilizaron en el *search grid* del SVR. Se probaron los cuatro Kernel más utilizados, desde función sigmoïdal hasta el de función de base radial. Para C, se tomó un rango de 1 a 4 y en gamma de 1 a 3, “scale” y “auto”.

Tabla 16. Search grid para SVR del NMOS

<b>Kernel</b>	<b>C</b>	<b>gamma</b>
Sigmoid	1,2,3 y 4	scale, auto, 1, 2 y 3
Poly	1,2,3 y 4	scale, auto, 1, 2 y 3
Linear	1,2,3 y 4	scale, auto, 1, 2 y 3
RBF	1,2,3 y 4	scale, auto, 1, 2 y 3

En la sección 5.2 se presentan los resultados, tanto la estructura o modelo que presento un mejor desempeño de cada uno de los métodos de aprendizaje como ejemplos de la extracción de parámetros de curvas de validación.

## 4.2. Transistor TFT IGZO

Como se mencionó en la introducción el transistor de película delgada (TFT) es uno de los elementos principales en las pantallas usadas actualmente, en celulares, televisores u monitores. Cada vez se desarrollan pantallas con mejores desempeños, resoluciones, gama de colores entre otros, para eso se necesitan tener TFTs con excelentes características eléctricas [40]. El TFT de Indio-Galio y Oxido de Zinc (IGZO), es un dispositivo que tiene una alta movilidad, buen rendimiento y gran estabilidad ante estrés eléctrico [41], [42]. A continuación, se presenta el análisis de algunos de los parámetros del TFT y los experimentos desarrollados para la extracción de ellos.

### 4.2.1. Análisis de parámetros y como afectan al TFT

Los parámetros a extraer del TFT IGZO son el voltaje de umbral  $V_T$ , el parámetro de transconductancia  $K_P$  y la resistencia de contacto  $R_C$ .

#### *Voltaje de umbral*

El voltaje de umbral como en la mayoría de los dispositivos, es el voltaje necesario para activar la conducción y que el transistor comience a operar. En la Figura 51 se presenta el barrido hecho en  $V_T$  de 0.5 a 2V. Con un  $V_{GS}$  establecido de 0 a 6V,  $V_{DS} = 6V$ ,  $K_P = 1.5 \mu A/V^2$  y  $R_C = 0 \Omega$ . las dimensiones del dispositivo son un  $W = 80 \mu m$  y un  $L = 10 \mu m$  con un  $T_{ox} = 15 nm$ , estas dimensiones son las mismas que los TFTs fabricados por el Centro de Nanociencias y Micro y Nanotecnologías (CNMN) del IPN, de los cuales se extrajeron sus parámetros. En la Figura 30 se observan 4 curvas I-V, a simple vista parece que la de 0.5V tiene mayor movilidad que la de 2V, pero en realidad lo que sucede es que la primero comienza a conducir aproximadamente desde los 0.5V en  $V_{GS}$ , y es posible ver una parte mayor de la curva (de 0.5 a 6V) mientras que con la curva de 2V, solo se ve de 2 a 6V. Si se tuvieran los valores de  $I_D$  para valores en  $V_{GS}$  mayores, sería posible observar que la curva con  $V_T = 2V$  alcanza la misma corriente en aproximadamente  $V_{GS} = 8V$ .

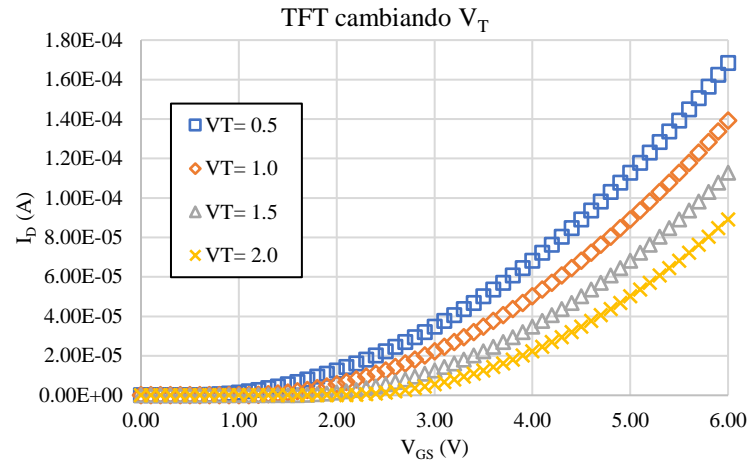


Figura 30. Comportamiento de TIT cambiando VT

**Parámetro de transconductancia**

El parámetro de transconductancia KP [43], es un parámetro que está definido por la movilidad de dispositivo  $\mu_{FET}$ , por la capacitancia  $C_{ox}$ , que a su vez la capacitancia se define por la constante dieléctrica ( $\epsilon_i$ ) y el espesor del óxido  $T_{ox}$  (ecuación 63).

$$KP = \mu_{FET}C_{ox} = \mu_{FET} \frac{\epsilon_i}{T_{ox}} \tag{63}$$

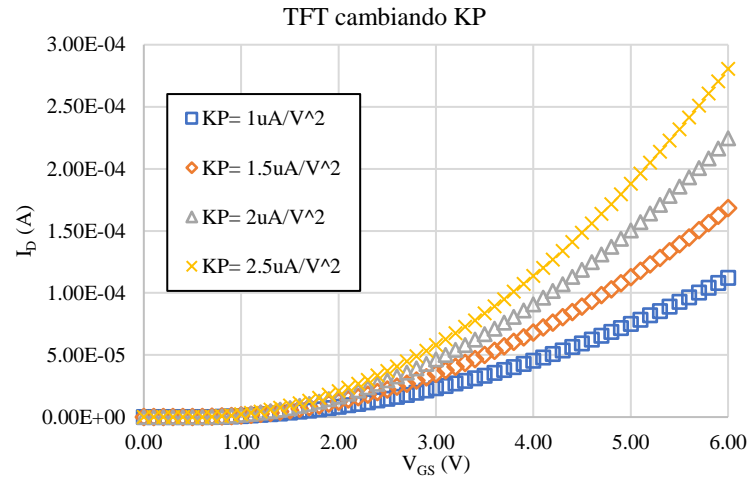


Figura 31. Comportamiento de TIT cambiando KP

La movilidad del dispositivo será directamente proporcional a KP. Estableciendo como constantes un  $V_T = 0.5$  V,  $R_C = 0$   $\Omega$  y los diferentes voltajes de la misma forma que en la subsección pasada (análisis de  $V_T$ ), la Figura 31 presentan las curvas I-V donde se hizo un barrido en KP de 1 a 2.5uA/V<sup>2</sup>, donde la movilidad si está siendo afectada directamente. La

condición del dispositivo es la misma en todas las curvas, comenzando en aproximadamente  $V_T = 0.5$  V.

### Resistencia de contacto

La resistencia de contacto  $R_C$ , es un parámetro que, como su nombre lo dice va a oponer una resistencia al flujo de corriente eléctrica, de manera que la corriente  $I_D$  será inversamente proporcional a  $R_C$ .  $R_C$  está definida por la siguiente ecuación [44]–[46]:

$$R_C = R_S + R_D \quad (64)$$

Donde  $R_S$  es la resistencia de en *source* y  $R_D$  es la resistencia de *drain*, en ambas terminales se presenta una resistencia, esta puede variar dependiendo del material utilizado durante la fabricación, en este caso los contactos de los TFTs IGZO fabricados por el CNMN son de Aluminio (Al). En la Figura 32 se presentan las curvas I-V donde se tiene constante un  $V_T = 0.5$  V y un  $K_P = 1.5$   $\mu\text{A}/\text{V}^2$ . Se hizo un barrido en  $R_C$  de 0 hasta 12  $\text{k}\Omega$ . Se puede observar que no hay un cambio significativo de 0 a 1  $\text{k}\Omega$ , curvas en las cual se alcanza la máxima corriente  $I_D$ , para  $R_C = 4$   $\text{k}\Omega$  ya hay una ligera disminución, pero es a partir de 8  $\text{k}\Omega$  que hay una pérdida de corriente evidente. Es importante notar que el comportamiento del dispositivo no es tan sensible al aumento de  $R_C$ , al menos en un rango de 0 a 3  $\text{k}\Omega$ , y para para notar un cambio importante de corriente entre un valor de  $R_C$  y otro, el aumento debe ser en aumentos de por lo menos de 2 a 4  $\text{k}\Omega$ .

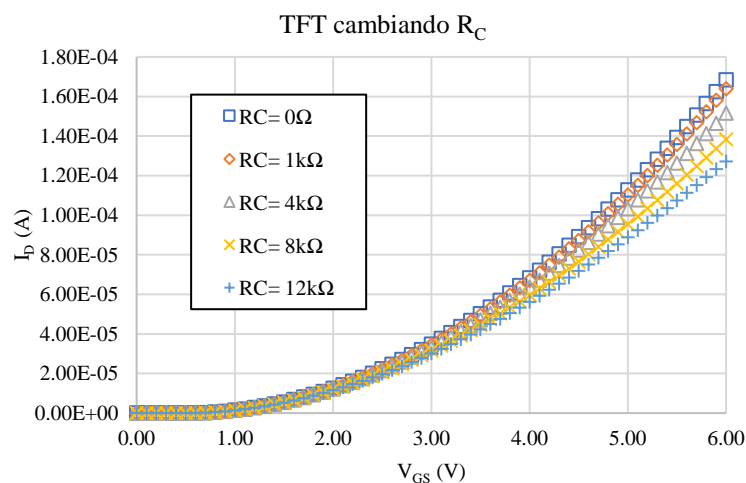


Figura 32. Comportamiento de TFT cambiando  $R_C$

#### 4.2.2. Extracción de KP, VT y RC del TFT

En esta sección se presenta como se realizó la generación del conjunto de datos para el entrenamiento, los rangos que se tomaron dentro de las simulaciones y el preprocesamiento que se les aplicó para realizar extracción de parámetros en TFTs. Así mismo se presenta la búsqueda en malla que se realizó para modificar los hiperparámetros de los métodos de aprendizaje. Los resultados obtenidos de esta extracción se encuentran en la sección 5.3.

##### *Generación del conjunto de datos*

Para realizar las simulaciones y crear el conjunto de datos del TFT se utilizó nuevamente LTspice. En el cual se realizaron un conjunto de combinaciones en los parámetros KP,  $V_T$  y  $R_C$ .

Se construyeron 2 conjuntos de entrenamiento, el primero de ellos para realizar la extracción de KP y  $V_T$ . El segundo se diseñó para la extracción de KP,  $V_T$  y  $R_C$ . El primer conjunto constó de 289 curvas I-V, los rangos de la simulación se presenta en la Tabla 17, el segundo conjunto tuvo 840 muestras, y el rango de las simulaciones se presenta en la Tabla 18.

El preprocesamiento de las muestras se realizó utilizando el valor máximo en cada una de las entradas. El vector  $V_{GS}$  de  $-6$  a  $6V$  fue dividido entre 6, KP fue dividido entre  $4.2 \times 10^{-6}$  y  $3.8 \times 10^{-6}$ , y así sucesivamente para cada una de las características de entrada o salida.

Tabla 17. Rango de parámetros para la extracción de KP y  $V_T$

Parámetro	Valor inicial	Valor final	Unidades	Incremento
KP	$1 \times 10^{-6}$	$4.2 \times 10^{-6}$	A/V <sup>2</sup>	$0.2 \times 10^{-6}$
$V_T$	1	4.2	V	0.2

Tabla 18. Rangos de parámetros para la extracción de KP,  $V_T$  y  $R_C$

Parámetro	Valor inicial	Valor final	Unidades	Incremento
KP	$1 \times 10^{-6}$	$3.8 \times 10^{-6}$	A/V <sup>2</sup>	$0.2 \times 10^{-6}$
$V_T$	1	3.6	V	0.3
$R_C$	0	$6 \times 10^{-6}$	$\Omega$	$1 \times 10^{-3}$

La dimensión del dispositivo en estas simulaciones fue de  $W=80\mu m$ ,  $L=10\mu m$  y  $T_{ox}=15nm$ . Como se mencionó anteriormente, el CNMN fabricó TFTs en los cuales se hará la extracción de parámetros, en la Tabla 19 se tiene una lista de los dispositivos fabricados y sus dimensiones, para algunos de ellos se tienen número de mediciones.

Tabla 19. Dimensiones de los dispositivos TFTs

Dispositivo	W (um)	L (um)	Tox (nm)
TFT1	80	10	15
TFT2	40	10	15
TFT3	80	5	15
TFT4	40	40	15
TFT5	80	20	15
TFT6	80	80	15
TFT7	40	5	15

### Entrenamiento de los métodos de aprendizaje

Como en la extracción del circuito inversor y en el transistor NMOS, para realizar la extracción en los TFTs IGZO se entrenaron los métodos de RN, RF y SVR. En la extracción de este dispositivo, y como se mencionó en la subsección pasada, se tienen dos conjuntos de datos, para la extracción de  $K_P$  y  $V_T$  se entrenaron únicamente RNs y para la extracción con  $R_C$  se utilizaron RN, RF y SVR.

En esta subsección se presentan los hiperparámetros de cada uno de los métodos y sus rangos que se tomó en el *Grid search*. La Tabla 20 presenta el top 8 de los modelos de red que presentaron los mejores desempeños, acompañados de las funciones de activación y factores de aprendizajes. El número de épocas establecido para el entrenamiento fue de 2000.

En la Tabla 21 se presentan los hiperparámetros y sus rangos configurados para realizar el *Grid Search* en el RF. En la Tabla 22 se presentan los hiperparámetros configurados en SVR.

Tabla 20. Search grid para RN del NMOS

Modelo de red	No. De capas	No. De neuronas	Función de activación	Factor de aprendizaje
1	2	128/64	Relu/Linear	0.001
2	2	256/32	Relu/Linear	0.001
3	2	256/64	Relu/Linear	0.001
4	2	256/128	Relu/Linear	0.001
5	2	256/256	Relu/Linear	0.001
6	3	128/64/32	Relu/Linear	0.001
7	4	256/218/64/32	Relu/Linear	0.001
8	5	256/128/64/32/16	Relu/Linear	0.001

Tabla 21. Search grid para RF del NMOS

max_depth	max_leaf_nodes	min_samples_split	No_estimators
3, 10, 15, 20 y 25	50, 70, 100, 120, 150, 180 y 200	2, 3 y 4	10, 30, 50, 80, 120, 150, 180 y 200

Tabla 22. Search grid para SVR del NMOS

Kernel	C	Gamma	Degree	Coef0
Sigmoid	0.01, 0.8, 1, 1.5, 2, 3 y 4	scale, auto, 0.001, 0.1, 0.5, 1, 1.5 y 2	1, 2 y 3	0, 0.5 y 1
Poly	0.01, 0.8, 1, 1.5, 2, 3 y 4	scale, auto, 0.001, 0.1, 0.5, 1, 1.5 y 2	1, 2 y 3	0, 0.5 y 1
Linear	0.01, 0.8, 1, 1.5, 2, 3 y 4	scale, auto, 0.001, 0.1, 0.5, 1, 1.5 y 2	1, 2 y 3	0, 0.5 y 1
RBF	0.01, 0.8, 1, 1.5, 2, 3 y 4	scale, auto, 0.001, 0.1, 0.5, 1, 1.5 y 2	1, 2 y 3	0, 0.5 y 1

### 4.3. Circuito de carga resistiva

En esta sección se presenta la extracción de parámetros de un circuito inversor de carga resistiva (CI), por su simplicidad se puede hacer la extracción utilizando la región de transición. Los parámetros relacionados con la movilidad tienen efectos similares al de la resistencia de carga que se utiliza, por ello se pueden omitir y solo encontrar el valor de dicha resistencia. A continuación, se presenta el comportamiento del CI cambiando sus parámetros.

#### 4.3.1 Análisis de los parámetros y como afectan al circuito inversor

La primera etapa de extracción se realiza con 2 parámetros ( $R_L$  y  $V_T$ ), usando otros 4 como entradas para alimentar la red,  $V_{in}$ ,  $V_{DD}$ ,  $V_{out}$  e  $I_D$ . El parámetro  $V_{DD}$  agrega robustez a la extracción, ya que esta se realiza en  $R_L$  y  $V_T$  para 5 diferentes valores de  $V_{DD}$ . Para realizar una extracción con un número mayor de parámetros es necesario realizar un estudio profundo de cómo afectan el comportamiento del circuito inversor para seleccionar los que más importantes y realizar una extracción de ellos por medio de las redes neuronales.

A continuación, se presentan un conjunto de figuras con gráficas con las cuales se puede observar cómo cambia el comportamiento del circuito inversor.

En la Figura 33 se presentan 3 gráficas con 8 curvas cada una. Se observa como la corriente  $I_D$  incrementa conforme se aumenta el voltaje de entrada, pero a su vez la corriente es menor si se utiliza una resistencia grande ( $R$  en  $\Omega$ ). Evidentemente, para resistencias bajas existirá un mayor flujo de corriente. Por otro lado, el voltaje de umbral no cambia el valor de la corriente, pero la recorre a izquierda o derecha según su valor.

En la Figura 34 se presenta el voltaje de salida del inversor, en la gráfica superior se tiene un  $V_T = 1V$ , lo que significa que el circuito comienza a invertir en  $V_{in} = 1V$ . Se observa como en resistencias (dada en kilo ohms) bajas, la región de transición es más larga y el voltaje de salida no llega hasta cero volts. Para resistencias de  $160\text{ k}\Omega$  la zona de transición

disminuye y se logran tener voltaje de menores a 0.5V. Por otro lado, en la gráfica inferior con un  $V_T = 5V$  las curvas se desplazan a la derecha y el circuito invierte a partir de  $V_{in} = 5V$ .

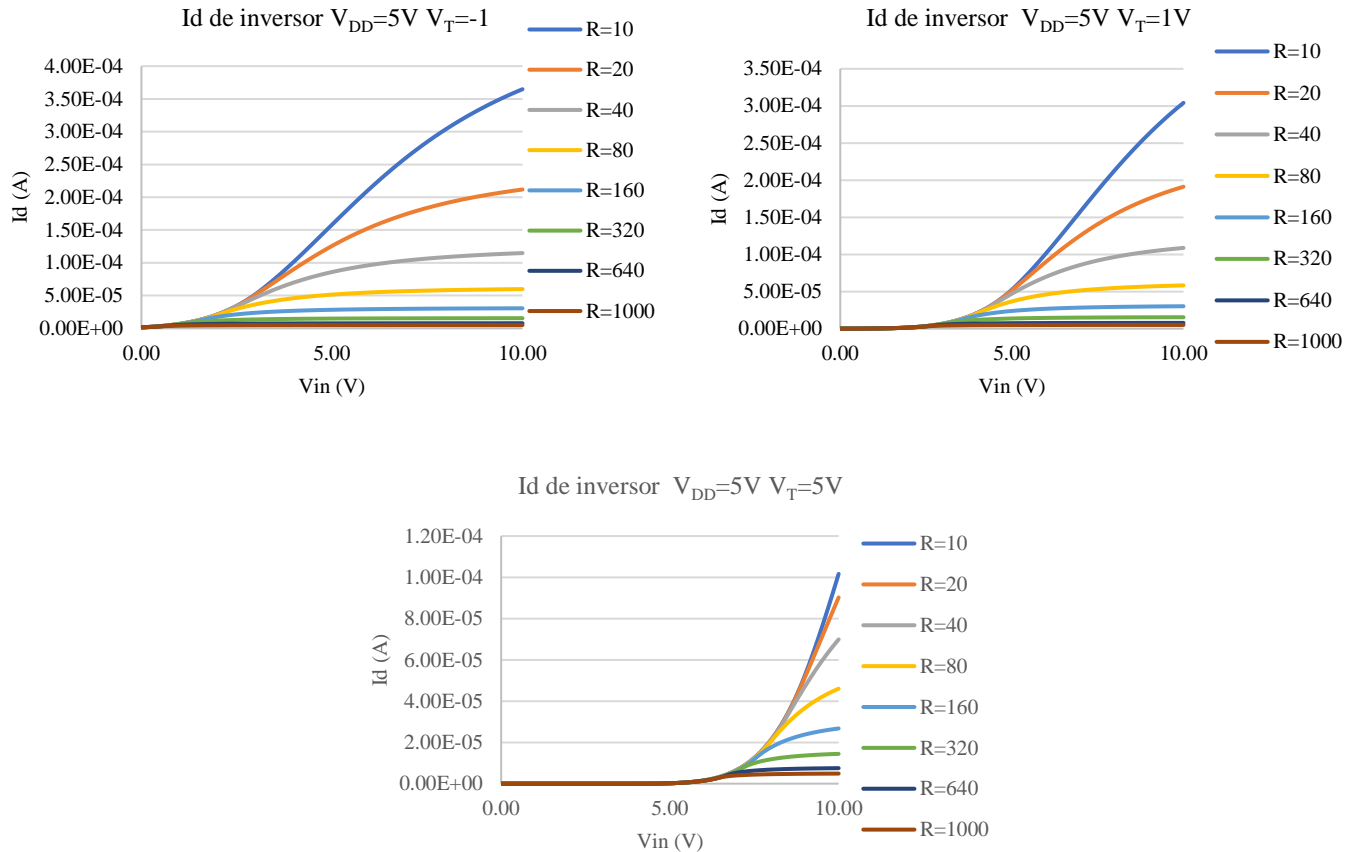


Figura 33.  $I_D$  cambiando  $R_L$  y  $V_T$

Como se observa en las gráficas 33 y 34 los parámetros de resistencia y voltaje de umbral influyen directamente en el comportamiento de la corriente y el voltaje de salida del inversor. En la Figura 35 se presentan las gráficas para  $I_D$  y  $V_{out}$ , cambiando el voltaje  $V_{DD}$ . Se colocaron como fijos los parámetros de  $R_L = 320k\Omega$  y  $V_T = 3V$  considerados como los valores medios que pueden tomar dichos parámetros. Se tiene que  $I_D$  es directamente proporcional a  $V_{DD}$ , es decir, a mayor voltaje en  $V_{DD}$  mayor será la corriente cuando  $V_{in} > V_T$ . Mientras que, en  $V_{out}$ ,  $V_{DD}$  define el valor ideal máximo que tendría  $V_{out}$  cuando se encuentre en salida alta.



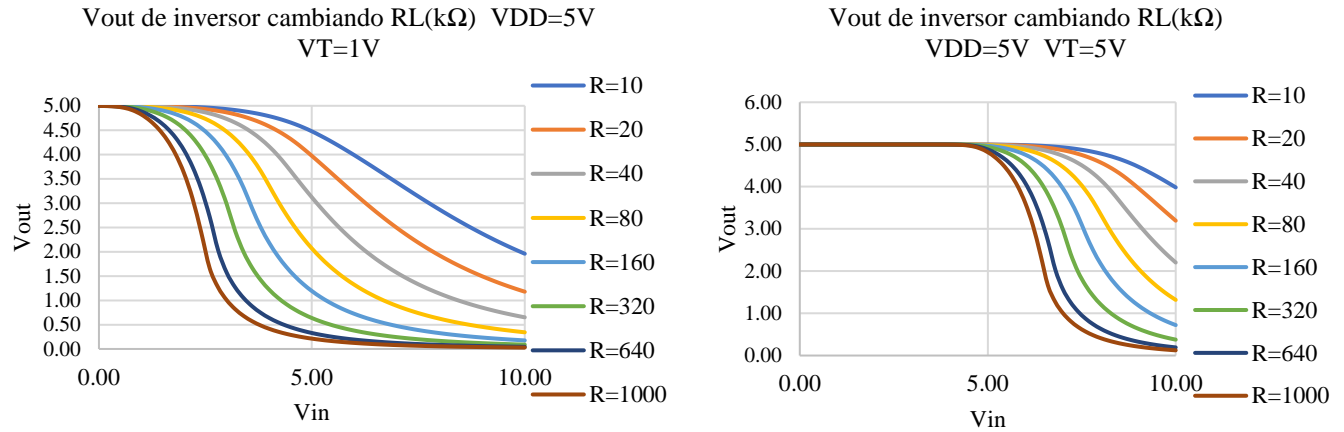


Figura 34.  $V_{out}$  cambiando  $R_L$  y  $V_T$

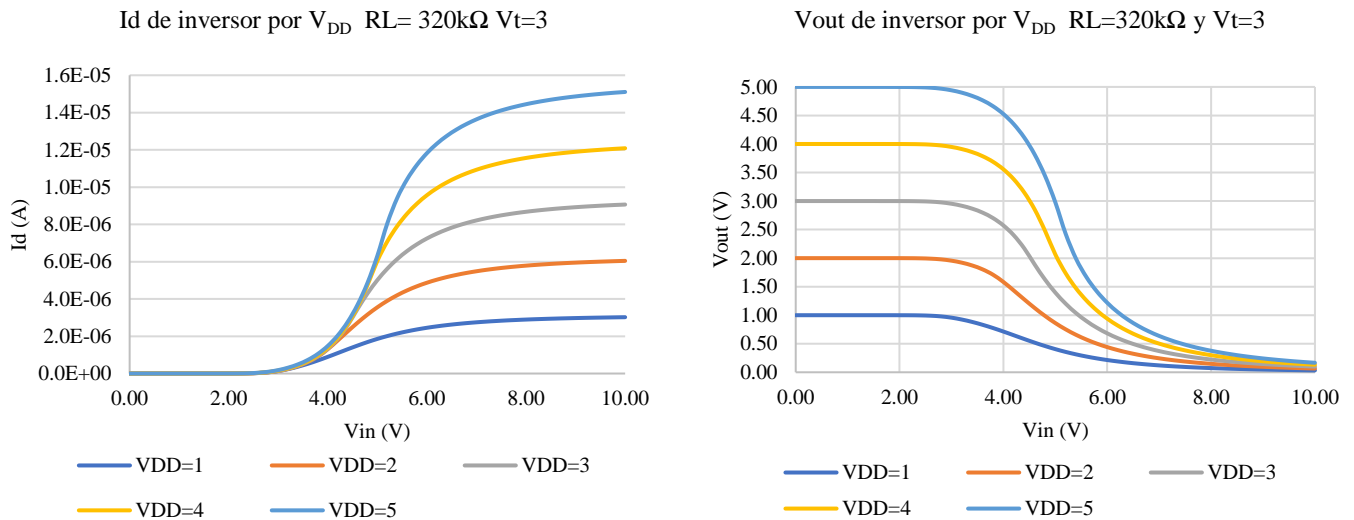


Figura 35.  $I_D$  y  $V_{out}$  cambiando  $V_{DD}$

A continuación, se presenta el análisis de 3 parámetros físicos con los que se construye el transistor, es decir, sus medidas.  $T_{ox}$  (*Thin-oxide thickness*) o el espesor de la capa de oxido,  $W$  y  $L$  (*Width y Length*) siendo el ancho y largo del canal del transistor. En la Figura 36 se presenta el diagrama que representa la estructura de un MOSFET en la cual se pueden identificar los parámetros mencionados.

En la Figura 37 se presentan el comportamiento de  $I_D$  y  $V_{out}$  cambiando el valor para el parámetro  $T_{ox}$ , dejando como fijos  $R_L= 320k\Omega$ ,  $V_T= 3V$ ,  $W= 1\times 10^{-3}m$  y  $L= 1\times 10^{-5}m$ . En la parte superior se puede notar como la corriente aumenta de manera exponencial cuando el valor de  $T_{ox}= 1\times 10^{-9}m$  y con menor  $V_{in}$  llega a la región activa. Entonces se puede decir que

la corriente  $I_D$  es inversamente proporcional con respecto a  $T_{ox}$ . Es importante mencionar que el valor común para  $T_{ox} = 1 \times 10^{-7} m$ . En la gráfica inferior, se tiene que el voltaje de salida cae de manera abrupta con el menor valor en  $T_{ox} = 1 \times 10^{-9} m$ , también se puede observar que se tiene un comportamiento similar que cuando se cambia el valor de  $R_L$ , cómo se observó en Fig. 34, la pendiente de la curva se hace más pronunciada. Ya que  $T_{ox}$  afecta al inversor de manera similar que la resistencia  $R_L$  y además,  $T_{ox}$  es un parámetro que se define al momento de la fabricación de los dispositivos, es un parámetro que se conoce y por lo tanto su extracción es innecesaria.

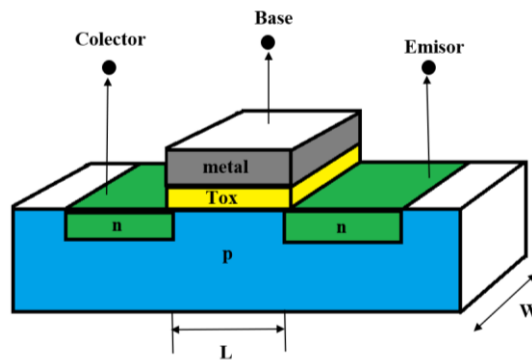


Figura 36. Estructura de un MOSFET canal n

Fuente: [30]

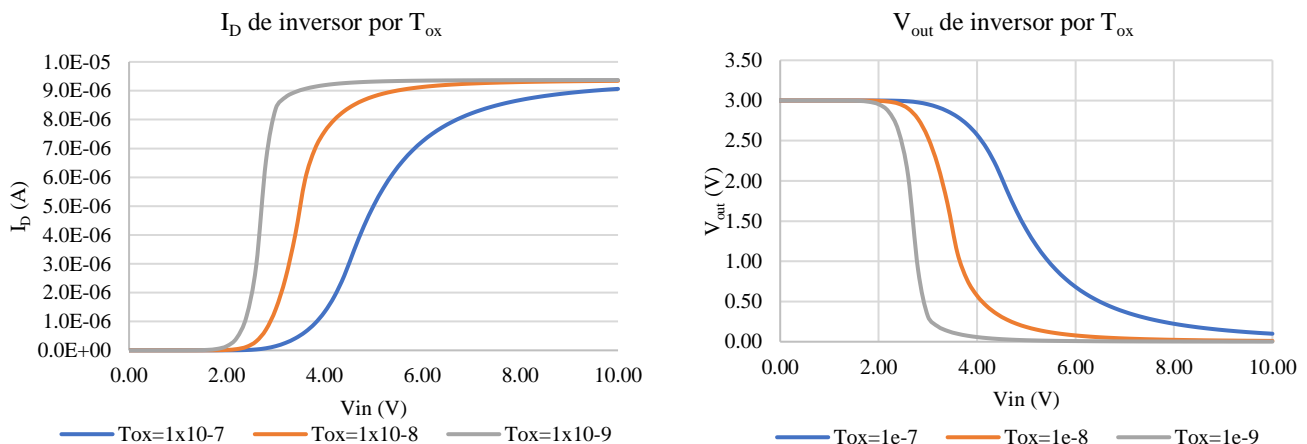


Figura 37.  $I_D$  y  $V_{out}$  cambiando  $T_{ox}$

Algo similar sucede con los parámetros  $W$  y  $L$ , ya que aumentan o disminuyen la corriente en el transistor. Cuando  $W$  se aumenta, el canal por donde circula la corriente es más ancho, lo que significa una mayor cantidad de corriente, entonces la corriente es proporcional con respecto a  $W$ . Por otro lado, cuando  $L$  es mayor se aumenta la longitud del

canal, lo que representa una mayor distancia que tiene que recorrer la corriente, teniendo  $L$  un comportamiento de resistencia, entonces a mayor  $L$  menor será la corriente que circula ya que es inversamente proporcional con respecto a  $L$ . En otras palabras, estos dos parámetros en el circuito inversor se comportan como resistencias (aumento y disminución), cuando se aumenta  $W$  circula más rápido la corriente teniendo una curva de  $V_{out}$  en el inversor similar a tener resistencias altas y al aumentar  $L$ , se disminuye la corriente provocando un  $V_{out}$  con una resistencia menor. Esto se puede observar en la Figura 38, donde se tienen los parámetros fijos usados anteriormente,  $W = 1 \times 10^{-6} \text{m}$  y lo que se cambia es  $L$ . Cuando este parámetro es menor, circula mayor corriente como se tiene en la gráfica superior, lo que provoca en  $V_{out}$  un descenso más abrupto como con resistencias de más grandes, como se ve en la gráfica inferior.

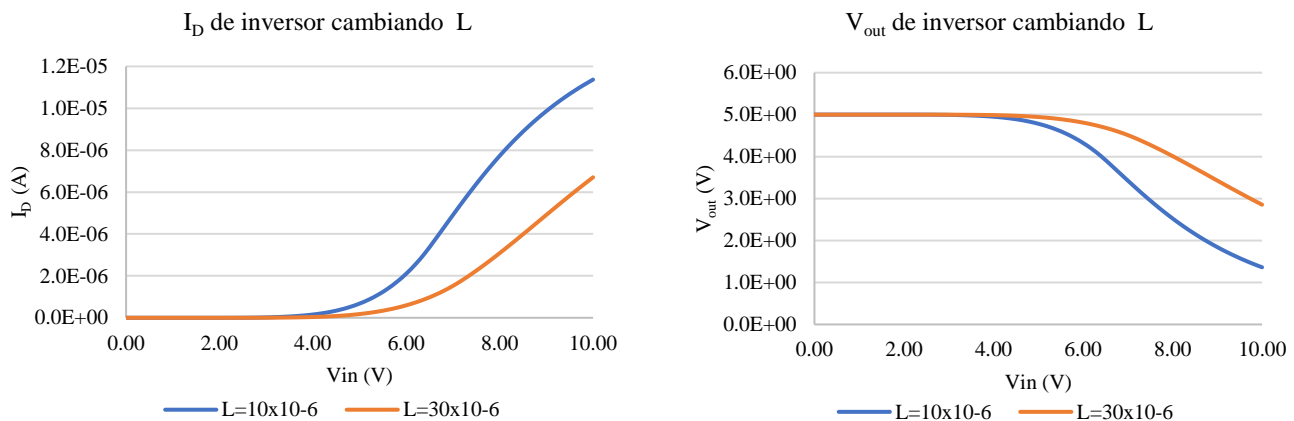


Figura 38. ID y  $V_{out}$  cambiando L

La Figura 39 muestra el cambio de  $I_D$  y  $V_{out}$  cuando se modifica  $W$ , se puede ver que cuando es mayor la corriente es proporcional (curva naranja gráfica superior) y  $V_{out}$  tiene un descenso abrupto en la gráfica inferior. Una vez más se presenta el comportamiento que se tiene al aumentar o disminuir  $R_L$ .

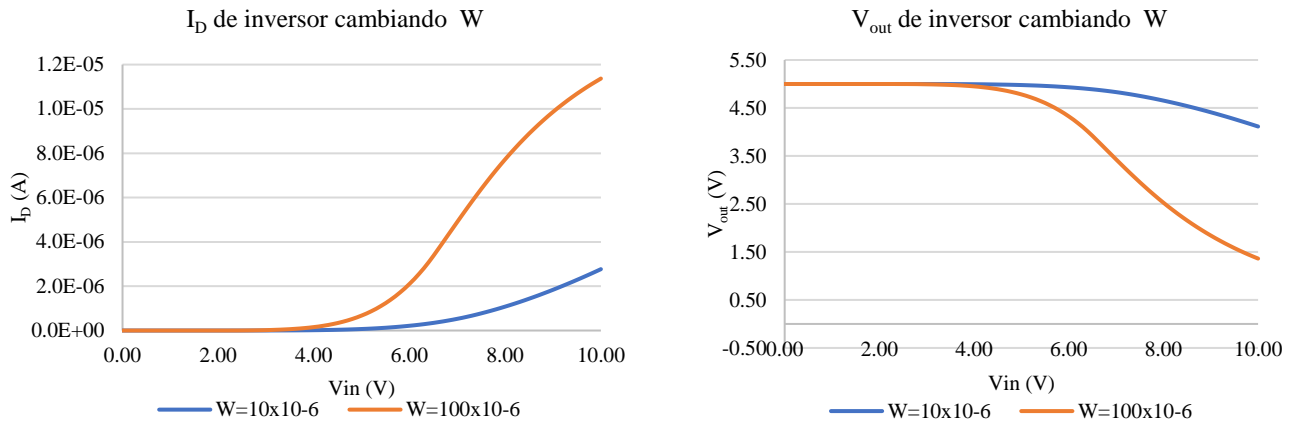


Figura 39. ID y V<sub>out</sub> cambiando W

### Parámetros I0 e I00 en TFT

Los parámetros I0 o constante de escala de fugas (*Leakage scaling constant*) e I00 o corriente de saturación de diodo inverso (*Reverse diode saturation current*) están relacionados con una fuga de corriente cuando el dispositivo no conduce y esto produce una pérdida de rendimiento. A continuación, se presentan un conjunto de curvas del transistor para observar cómo se ve afectado.

En la Figura 40 (parte izquierda) se muestra la curva característica del TFT donde se hace un barrido en I0 desde 10 a  $10 \times 10^6$  A/m, dejando fijo a I00 en su valor por default (150 A/m) se puede observar que no hay un cambio notorio en el comportamiento con excepción en la parte inicial, pero aun así parece mínimo el cambio. En la misma figura (parte derecha) se tienen las mismas curvas, pero solo se tomó la región subumbral (hasta 2V), de esta manera se observa de mejor manera que si hay un cambio en el comportamiento de la curva cuando el parámetro I0 es incrementado considerablemente.

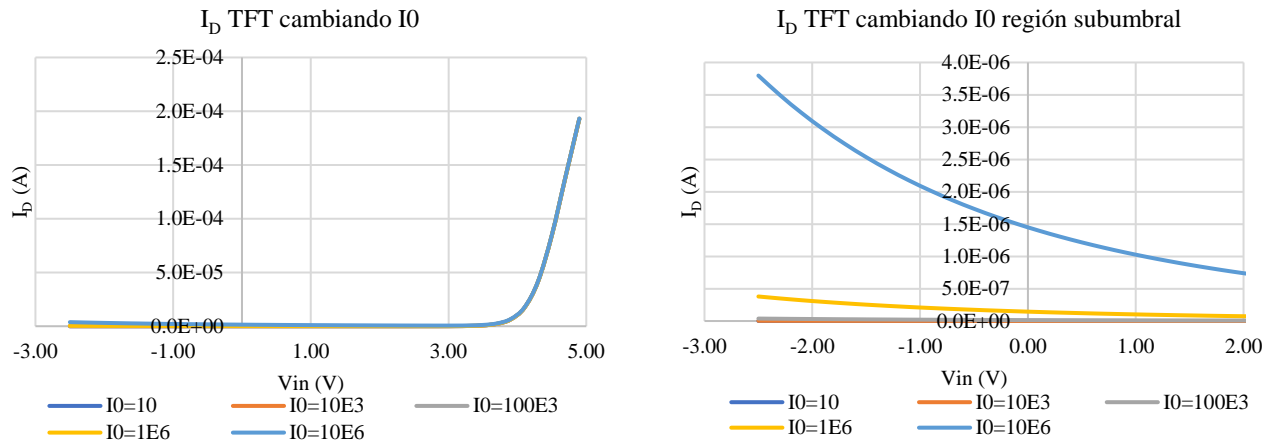


Figura 40.  $I_D$  cambiando  $I_O$

En la Figura 41 (parte izquierda) se muestra el comportamiento del TFT modificando el parámetro  $I_{O0}$ , en el que se hizo un barrido de 150 a  $10 \times 10^6$  A/m, dejando fijo a  $I_O$  en su valor por default (10A/m). Al igual que en la figura 40, a simple vista parece que el comportamiento no se ve afectado. En la misma figura (parte derecha), se tomó la región subumbral y en ella se puede ver que si hay un efecto en las curvas.  $I_{O0}$  provoca que exista una corriente constante previa al voltaje umbral, a diferencia de  $I_O$ , en donde la corriente de fuga es mayor y esta va descendiendo conforme el voltaje de entrada aumenta.

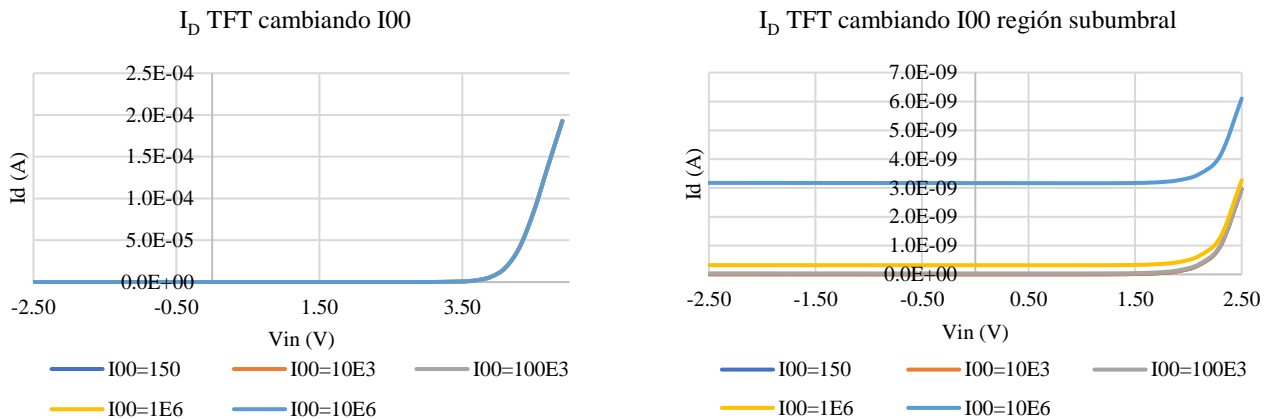


Figura 41.  $I_D$  de TFT cambiando  $I_{O0}$

A continuación, se presentan las curvas cambiando ambos parámetros al mismo tiempo de manera incremental. Con un barrido desde su valor por default correspondiente hasta  $10 \times 10^6$  A/m. En la Figura 42 es muy poco el cambio que se puede observar a simple vista como en las figuras anteriores, mientras que en la Figura 43, se ve el cambio de las curvas en

la región subumbral. El comportamiento es similar al que se tiene cambiando solamente  $I_0$  y se podría decir que  $I_{00}$  no está teniendo algún efecto cuando se aumenta junto a  $I_0$ . Por esta razón se juntaron las tres curvas (Fig. 44) donde se modifica por separado  $I_0$ ,  $I_{00}$  y la tercera modificando ambos tomando un valor de  $10 \times 10^6 \text{ A/m}$ .

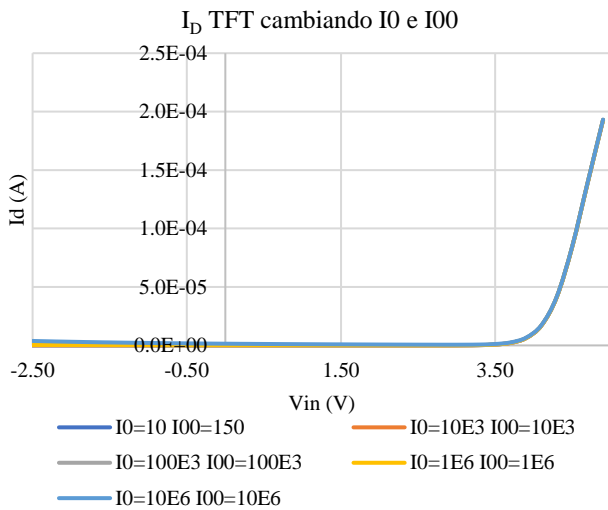


Figura 42. ID TFT cambiando  $I_0$  e  $I_{00}$

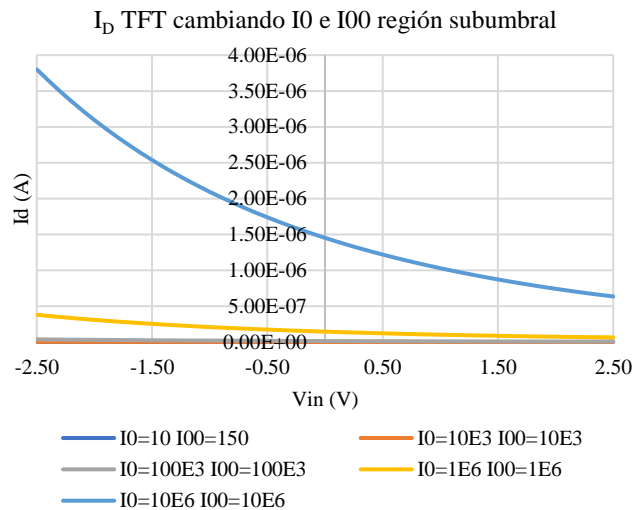


Figura 43. ID TFT cambiando  $I_0$  e  $I_{00}$  subumbral

En la Figura 44 se unieron las curvas donde  $I_0 = 10 \times 10^6 \text{ A/m}$ ,  $I_{00} = 10 \times 10^6 \text{ A/m}$ ,  $I_0$  e  $I_{00} = 10 \times 10^6 \text{ A/m}$ , en ellas se observa que entre la primera y tercera curva no hay diferencia, son prácticamente la misma, es decir,  $I_{00}$  no tiene efectos sobre  $I_0$  cuando vale  $I_{00} = 10 \times 10^6 \text{ A/m}$ . La segunda curva se encuentra muy por debajo ya que el valor de  $I_d$  es muy pequeño, por esto se agregaron dos curvas más donde  $I_{00} = 1000 \times 10^6 \text{ A/m}$  (color amarillo) e  $I_{00} = 1000 \times 10^7 \text{ A/m}$  (color azul), donde en esta última ya se acerca a la curva de  $I_0 = 10 \times 10^6 \text{ A/m}$ . Esto nos dice que el parámetro  $I_{00}$  debe tener valores bastante grandes para que ejerza un cambio significativo sobre el comportamiento del TFT pudiéndose tomar como un parámetro discriminativo y solo trabajar con  $I_0$ .

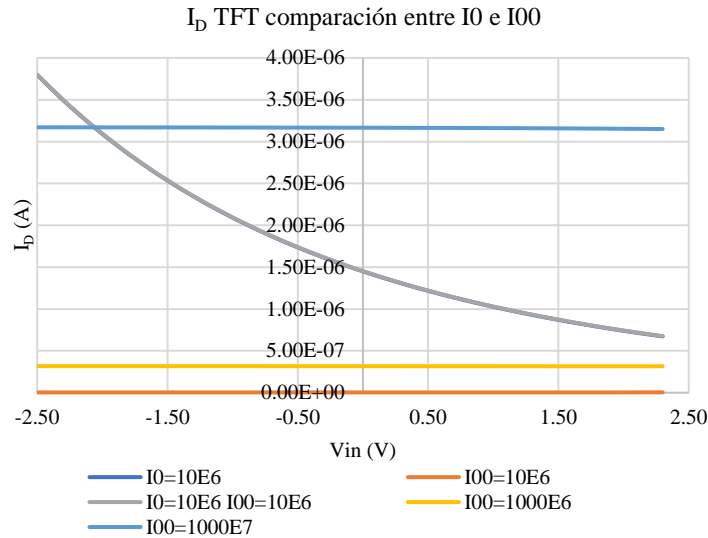


Figura 44. ID de TFT comparación entre I0 e I00

**Parámetros I0 e I00 en inversor**

En esta sección se analizará el efecto de los parámetros I0 e I00, pero en el circuito inversor, en la Figura 45 se tiene las curvas características haciendo el mismo barrido de I0 de 10 a  $10 \times 10^6$  A/m y un  $V_{in}$  desde -2.5 a 5V. En esta ocasión es evidente que hay un cambio en el comportamiento de las curvas, el voltaje  $V_{out}$  cuando la salida es alta defiende con valores grandes en I0. En la Figura 46 se toma solo la región subumbral para una mejor apreciación, se puede notar que no hay grandes cambios para los primeros valores de I0. Para el valor más alto  $V_{out}$  pierde hasta 1V.

El efecto del parámetro I00 sobre el inversor se encuentra en la Figura 47, que se podría decir que tal efecto no existe, ya que a simple vista no es posible diferenciar un cambio como lo fue con I0. En la Figura 48 se muestra solo la región subumbral en la cual ya se puede notar el descenso de  $V_{out}$ . La reducción fue de solo 0.003169V para el valor más alto de I00. Con esto se podría confirmar que I00 es un parámetro de poca importancia ya que sus efectos son mínimos.

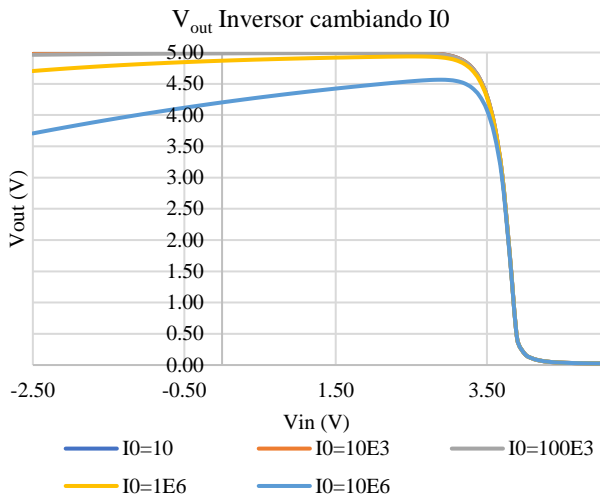


Figura 45.  $V_{out}$  inversor cambiando  $I_0$

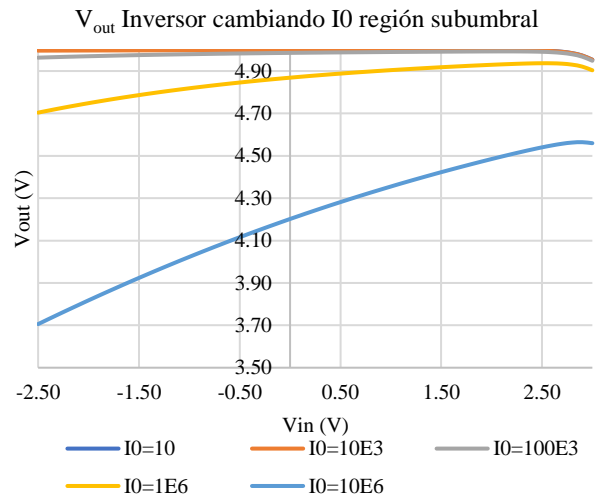


Figura 46.  $V_{out}$  inversor cambiando  $I_0$  subumbral

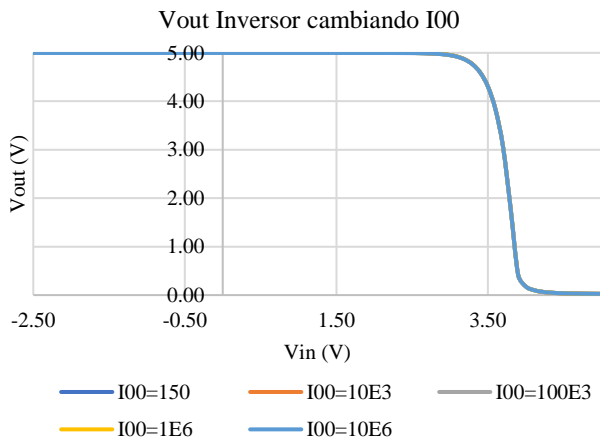


Figura 47.  $V_{out}$  inversor cambiando  $I_{00}$

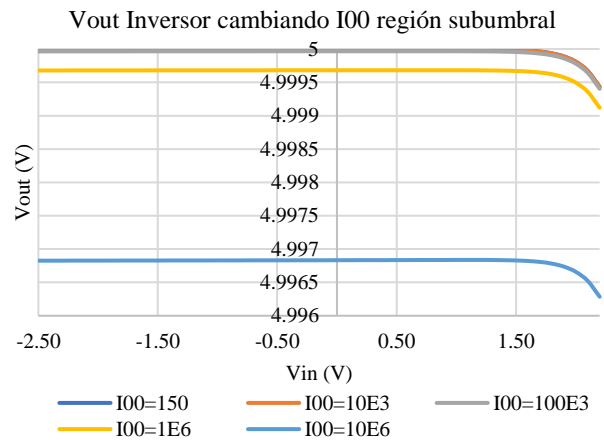


Figura 48.  $V_{out}$  inversor cambiando  $I_{00}$  subumbral

En las Figuras 49 y 50 se presentan las curvas en donde ambos parámetros fueron incrementándose desde su valor default hasta  $10 \times 10^6 \text{ A/m}$ . Se observa el mismo comportamiento que para el TFT, es decir un efecto de  $I_0$ . Hay un descenso en el voltaje  $V_{out}$  similar al que se presenta en la figura 46 para el valor más alto.

Para confirmar que los efectos de  $I_0$  sobre el inversor son más pronunciados que  $I_{00}$  se unieron nuevamente las curvas que pertenecen a  $I_0 = 10 \times 10^6 \text{ A/m}$ ,  $I_{00} = 10 \times 10^6 \text{ A/m}$  y en la que se aumentan al mismo tiempo  $I_0$  e  $I_{00}$  igual a  $10 \times 10^6 \text{ A/m}$ . En la Figura 51 se observa nuevamente que la primer y tercer curva son las mismas, dejando por debajo la segundo donde solo se cambia  $I_{00} = 10 \times 10^6 \text{ A/m}$ . Con esto se afirma que  $I_{00}$  se puede descartar como un parámetro que influya fuertemente en el circuito inversor. Se agregaron dos curvas más



donde  $I_{00} = 1000 \times 10^6 \text{ A/m}$  e  $I_{00} = 3000 \times 10^6 \text{ A/m}$  las cuales ya presentan un efecto significativo, sin embargo su valor debe ser bastante grande, algo que en circuitos físicos podría no ser algo común.

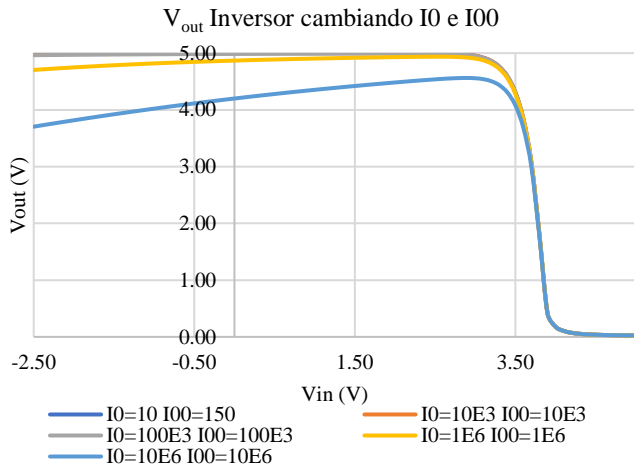


Figura 49. Vout inversor cambiando I<sub>0</sub> e I<sub>00</sub>

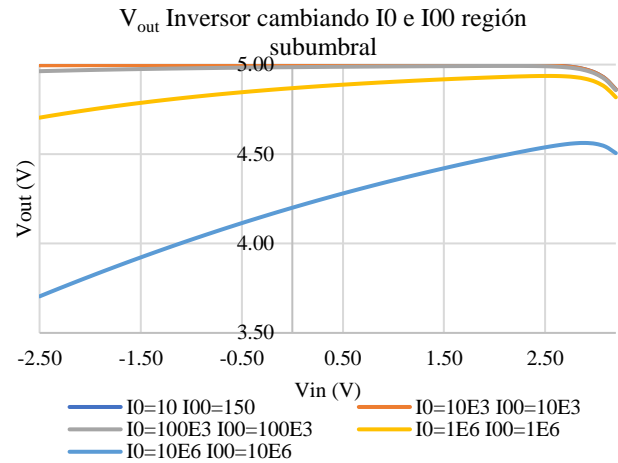


Figura 50. Vout inversor cambiando I<sub>0</sub> e I<sub>00</sub> región subumbral

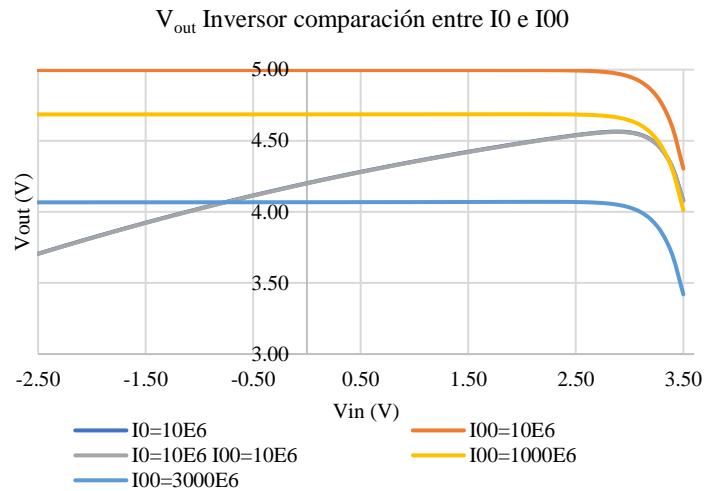


Figura 51. Vout inversor comparación entre I<sub>0</sub> e I<sub>00</sub> región subumbral

### Parámetro MMU en TFT

Uno de los parámetros más importantes dentro de los transistores, son aquellos los que afectan la movilidad de los electrones en el dispositivo, es decir, los parámetros de movilidad son los que definen el flujo de electrones. A mayor movilidad, mayor será el flujo y viceversa. El parámetro MMU (exponente de movilidad de campo bajo) es un parámetro que afecta la

movilidad en el TFT. Para comprender como afecta al dispositivo, en la Figura 52 se presenta la curva característica del TFT, donde se presenta un barrido en MMU de 0.8 a 3.

Se puede observar que la corriente eléctrica es muy baja para valores de 0.8 a 2, el cambio es casi imperceptible entre dicho rango, además que la corriente no aumenta conforme lo hace el voltaje de entrada ( $V_{in}$ ). Para  $MMU=2$ , se nota un cambio en la corriente, habiendo un aumento de esta, a partir de  $V_{in}=4$  V. Cuando  $MMU=3$ , el crecimiento de la corriente ya no tiene un comportamiento lineal, teniendo un acelerado aumento de la corriente desde  $V_{in}=3.5$  V. Tener mayor movilidad, es similar a tener una menor resistencia, de modo que la corriente fluye libremente y es cuando se observa un flujo de corriente mayor.

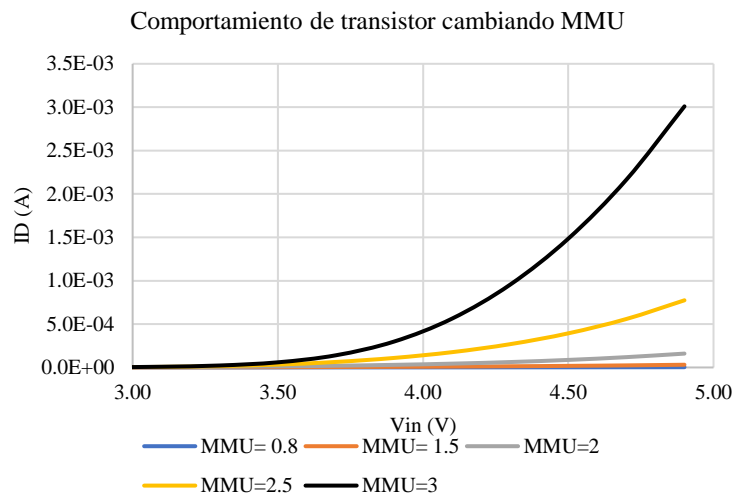


Figura 52.  $I_D$  de TFT cambiando MMU

### ***Parámetros MMU en Inversor***

En esta sección se presenta el comportamiento que tiene el circuito inversor al cambiar el parámetro MMU, anteriormente se mostró, que tiene puede aumentar o disminuir la corriente. En la Figura 53 se presenta la curva de la corriente  $I_D$  del inversor cambiando el valor de MMU. Se observa un cambio de estado bajo cuando  $V_{in} \leq 0$  V a estado alto, y hay una región de transición a estado algo cuando  $V_{in} > 0$  V (dependerá del voltaje umbral  $V_T$ ). Es aquí donde afecta MMU, el rango de  $V_{in}$  necesario para realizar la transición de estado bajo a alto puede ser mayor al tener un valor en MMU pequeño, ya que la corriente que pasa es menor, por el contrario, al tener un valor de MMU alto, hay un crecimiento exponencial en la corriente lo que significa que el rango de  $V_{in}$  para pasar de un estado a otro es menor.

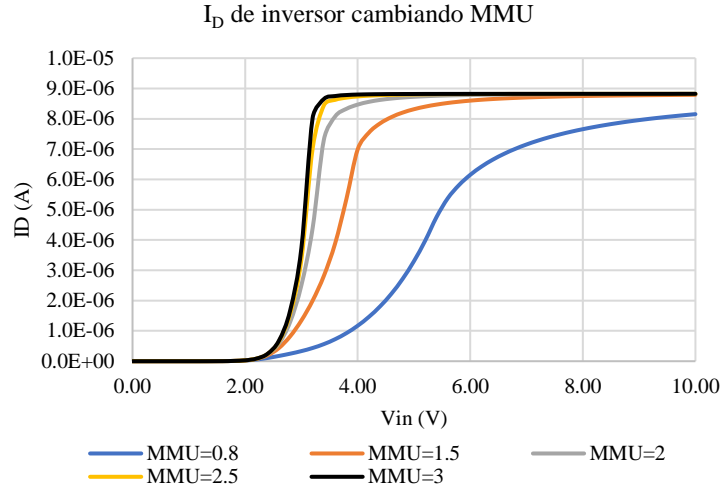


Figura 53.  $I_D$  de inversor cambiando MMU

En la Figura 54, se presenta la curva característica del inversor, se puede observar que la corriente  $I_D$  tiene un comportamiento inverso al voltaje de salida  $V_{out}$ . Cuando se tiene un valor grande de MMU, la corriente aumenta exponencialmente, en el caso de  $V_{out}$ , este disminuye, pasando rápidamente de una salida alta para  $V_{in} \leq 0$  V a una salida baja cuando  $V_{in} > 0$  V. Al tener valores pequeños en MMU, la región de transición es más grande, tomando un rango mayor en  $V_{in}$ .

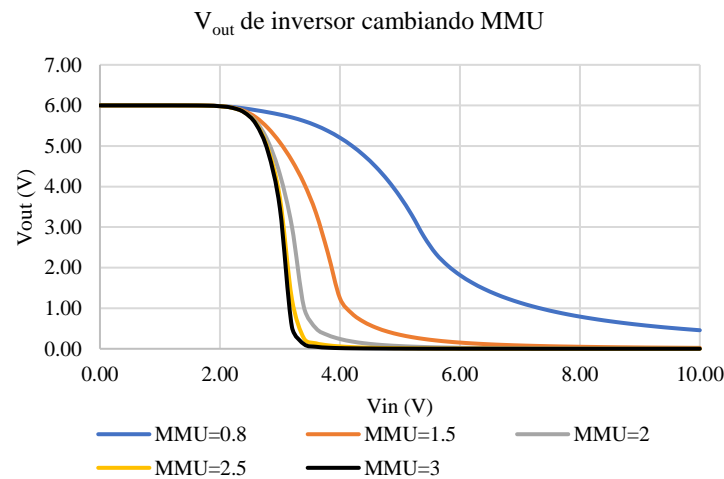


Figura 54.  $V_{out}$  de inversor cambiando MMU

### 4.3.2. Extracción de 2 parámetros

En esta sección se presenta la extracción de parámetros utilizando el lenguaje de programación Python. El objetivo de migrar a este lenguaje es aprovechar que tiene una

licencia de código abierto además que posee gran variedad de herramientas y librerías destinadas a la creación de sistemas inteligentes como lo son las RNAs. Se hace uso de las librerías Keras ahora propia de Tensorflow [47] entre otras para diseñar y entrenar la RN que hará la extracción de parámetros.

### ***Normalización del conjunto de datos***

El conjunto de datos debe llevar un preprocesamiento o normalización con la finalidad de tener la información en una escala que sea más fácil de tratar por la RN. Para tratar el conjunto de datos en estos experimentos se utilizaron dos formas más aparte de la ya usada (dividir entre la magnitud de cada vector). La segunda forma es dividir cada valor entre el máximo (ecuación 65) para tener un rango de  $-1$  a  $1$  y la tercera forma es usar una distribución normal (ecuación 66). Esto se hizo para encontrar cual forma de normalización brinda mejores resultados al momento de entrenar la RN.

$$p_i = \frac{p_i}{\max(p_i)} ; p_o = \frac{p_o}{\max(p_o)} \quad (65)$$

Donde  $p_i$  y  $p_o$  corresponden a los patrones de entrada y salida respectivamente.

$$p_i = \frac{p_i}{\left(\frac{p_i - \mu_{p_i}}{\sigma_{p_i}}\right)} ; p_o = \frac{p_o}{\left(\frac{p_o - \mu_{p_o}}{\sigma_{p_o}}\right)} \quad (66)$$

Donde  $\mu$  corresponde al promedio y  $\sigma$  la desviación estándar.

Por otro lado, se generó otro conjunto de datos que corresponde a la extracción del parámetro  $I_0$  (4.1.1). Este conjunto está conformado por 400 curvas, donde se hace un barrido en  $V_T$  de  $1$  a  $5V$  y en  $I_0$  de  $10A/m$  a  $1 \times 10^3 A/m$ . Se usaron cuatro valores de resistencia  $R_L$  ( $10k\Omega$ ,  $100k\Omega$ ,  $510k\Omega$  y  $1M\Omega$ ).

Anteriormente se estaba trabajando con los puntos de cada curva por separado, es decir, para cada valor de  $V_{in}$  ( $0$  a  $10V$  con incrementos de  $0.2V$ ) se tenía un valor de  $V_{out}$ ,  $I_D$  y un  $V_{DD}$  los cuales conforman un patrón o ejemplo que corresponden a una salida de  $V_T$  y  $R_L$ . De modo que el conjunto de 400 curvas, cada una con 51 puntos formarían un conjunto de 20,400 datos para la red. Sin embargo, una entrada de 4 elementos no brinda suficiente información para extraer parámetros que no afectan al inversor de manera evidente como lo hacen  $V_T$  y  $R_L$ . Por lo anterior el conjunto de entrenamiento de ordeno manejando las entradas con las curvas completas.

Cada vector de entrada es de 204 elementos, donde se dividen entre los cuatro parámetros ya mencionados ( $V_{in}$ ,  $V_{DD}$ ,  $I_D$  y  $V_{out}$ ). De esta manera las entradas brindan

información completa de cada curva y es más fácil extraer los parámetros de salida ( $I_0$ ,  $V_T$  y  $R_L$ ). Una desventaja de este método es que el tamaño del conjunto es menor, ya que solo se cuentan con las 400 como ejemplos.

El conjunto de datos compuesto por 400 curvas pretende usarse para la extracción del parámetro  $I_0$  el cual tiene un gran rango de valores. El preprocesamiento en donde cada entrada es dividido por el valor máximo para mantener valores de 0 a 1 ya no es suficiente ya que para  $I_0$ , la mayoría de sus valores estarán muy cerca de 0 y solo uno de ellos valdrá 1. Para disminuir el amplio rango que tiene  $I_0$ , se obtienen sus posibles valores en una escala logarítmica (base 10) y después dividir los datos entre el valor máximo obtenido. De esta forma se tiene una distribución más uniforme de los datos.

Para maximizar la distribución de los valores de  $I_0$  en escala logarítmica y divididos entre el valor máximo se aplicó la ecuación para calcular una distribución normal, presentada anteriormente (ecuación 65).

### ***Diseño y entrenamiento de la red***

Para encontrar el mejor modelo se debe hacer una búsqueda de los mejores parámetros para la red, es decir, cuantas capas ocultas, cuantas neuronas por capa, que funciones de activación, que factor de aprendizaje ( $l_r$ ) y cuantas épocas, son las que brindan un mejor aprendizaje en la red. En los experimentos se realiza una búsqueda de estos valores óptimos de tipo rejilla, donde se va probando la combinación de los parámetros. Por ejemplo, se prueba una red de una capa con 32 neuronas con un número de épocas considerable (1000 épocas es lo más usual) con un factor de aprendizaje fijo ( $l_r = 0.001$ ) y se lo que se buscará cual o cuales funciones de activación brindan un mejor resultado, entrenando modelos con diferentes funciones. Una vez que se identifica las mejores funciones, se hace un barrido en  $l_r$ , en este caso de 0.1 a 0.00000001, para identificar el mejor. El número de épocas puede depender del  $l_r$ , ya que, entre más pequeño, serán necesarias más épocas para alcanzar cierto valor de salida. Si el modelo de 32 neuronas brinda una exactitud considerablemente baja (a decisión del diseñador) se puede proponer un modelo más complejo con más capas y más neuronas utilizando los mejores parámetros que ya se identificaron. Cuando la exactitud (ACC) ya no aumenta y tampoco disminuye el error (MSE) aunque el modelo se haga más complejo se considera que será el mejor modelo posible para el problema y conjunto de datos dado. Si se presentan varios modelos que tienen el mismo desempeño (ACC o MSE) o muy

similar es preferible seleccionar el modelo más simple de ellos, ya que es el que requiere menos poder de cómputo.

En la Tabla 23 se muestra una lista de los modelos entrenados, donde en cada modelo se entrenó con 0.001 y 0.0001 como  $l_r$  ya que ambos fueron los que brindan mejores resultados. La mejor función de activación fue la *relu* en las capas ocultas, mientras que en la capa de salida se usó la *linear*, otras funciones como sigmoide o tangente hiperbólica se probaron, pero el aprendizaje de la red estaba por debajo del 60% de ACC. El número de épocas fueron de 1000 para  $l_r = 0.001$  y 2500 cuando  $l_r = 0.0001$ .

Tabla 23. Modelos de red entrenados

No. de capas ocultas	No. de neuronas	Función de activación	$l_r$
1	32	Relu-linear	0.001-0.0001
1	64	Relu-linear	0.001-0.0001
1	128	Relu-linear	0.001-0.0001
1	256	Relu-linear	0.001-0.0001
1	512	Relu-linear	0.001-0.0001
1	1024	Relu-linear	0.001-0.0001
1	2048	Relu-linear	0.001-0.0001
2	256-32	Relu-linear	0.001-0.0001
5	128-64-32-16-8	Relu-linear	0.001-0.0001

Cada modelo fue entrenado con los tres conjuntos de datos que fueron normalizados. El que presentó mejor rendimiento fue el que tenía dos capas ocultas de 256 y 32 respectivamente con el conjunto que uso normalización usando el valor máximo de cada vector. Los resultados se muestran en la sección 5.3.1.

#### 4.3.3. Extracción de 4 parámetros utilizando diferentes métodos de aprendizaje automático

Anteriormente se presentó la extracción de tres parámetros dentro del circuito inversor,  $V_T$ ,  $R_L$  e  $I_0$  utilizando únicamente redes neuronales. A continuación, se presenta la extracción de cuatro parámetros, los tres anteriores con un cuarto MMU, utilizando tres técnicas más de aprendizaje automático, propias del *Machine Learning*: *árboles de decisión*, *Random forest* y *SVR*. Dichas técnicas fueron implementadas utilizando las librerías pertenecientes a *scikit-learn* [48]. De esta manera se tiene un panorama más amplio de los resultados que brinda diferentes métodos que pertenecen a la inteligencia artificial.

#### Generación del conjunto de datos y normalización

Para realizar la extracción de cuatro parámetros es necesario contar con el conjunto de datos que contenga información de los mismos, es decir, se necesita un conjunto de curvas del circuito inversor que tengan diferentes combinaciones de los parámetros.

Se realizaron un total de 1200 simulaciones del circuito inversor considerando los cuatro parámetros  $V_T$ ,  $R_L$ ,  $I_0$  y  $MMU$ . En la Tabla 24, se presentan los diferentes valores que tomaron cada uno de ellos.

Tabla 24. Valores para cada parámetro

Parámetro	Valor	Unidades
$V_T$	1, 2, 3, 4 y 5	V
$R_L$	10k, 39k, 100k, 510k y 1M	$\Omega$
$I_0$	6, 1k, 5k y 10k	A/m
$MMU$	0.8, 1.5 y 3	-

### ***Diseño y entrenamiento de los modelos***

En esta sección se presenta como fueron entrenados los diferentes métodos de aprendizaje automático, buscando los mejores hiper parámetros para así encontrar el mejor modelo o arquitectura de cada uno de los métodos.

La manera más común para encontrar los mejores hiper parámetros de un método de aprendizaje, es conocido como *Grid Search*, o búsqueda en malla, que consiste en hacer un barrido en cada uno de los parámetros en un rango predeterminado. El error cuadrático medio obtenido por cada una de las combinaciones de hiper parámetros es almacenado. Al finalizar se selecciona la combinación que obtuvo el menor error, y esos hiper parámetros son seleccionados como los mejores para ajustar los datos de entrenamiento.

La búsqueda en malla (*Search Grid*) también conocida como búsqueda exhaustiva es el método que permite encontrar el mejor valor de los hiperparámetros de los modelos de aprendizaje automático, dicho valores controlan el aprendizaje. Los valores de los hiperparámetros son propios de los modelos, es decir, estos no se ajustan durante el entrenamiento. Cuando se configura la estructura de los modelos deben ser seleccionados los valores en los hiperparámetros, pero inicialmente no es posible conocer el valor óptimo.

La búsqueda en malla realiza una combinación del tipo todos contra todos para encontrar cuál de ellas brinda el mejor desempeño para el modelo. En este trabajo se utilizó la librerías *klearn.model\_selection.GridSearchCV* [33] para realizar la implementación de la búsqueda en malla. A continuación, en las Tablas 25-28 se presentan los valores de los diferentes hiperparámetros con los cuales se realizó la búsqueda en malla. En la sección de

resultados se presenta cuáles fueron los mejores hiperparámetros de cada modelo de aprendizaje.

En la Tabla 25 se presentan los hiperparámetros que fueron modificados en la RN para encontrar el mejor desempeño.

Tabla 25. Valores en hiperparámetros de RN

Capas ocultas	No. neuronas	Función de activación	Learning rate	Épocas
1	32,64,128,256 y 512	Sigmoide, Linear, relu	0.1, 0.01, 0.001,	100, 500, 1000,
2	32,64,128 y 256	y tanh	0.0001 y 0.00001	1500, 2000 y 5000

En la Tabla 26 se presentan los hiperparámetros que fueron modificados en el AD para encontrar el mejor desempeño. En la Tabla 27 están los hiperparámetros modificados en el RF, los cuales son muy similares al AD, aunque se consideran el número de ADs que tendrá el RF.

Tabla 26. Valores en hiperparámetros de AD

Max_depth	max_leaf_nodes	min_samples_split	min_samples_leaf
None, 5, 10, 15, 30, 60	None, 130, 180, 200, 250 y 300	2, 3 y 4	2, 3, 4 y 5

Tabla 27. Valores en hiperparámetros de RF

Max_depth	max_leaf_nodes	min_samples_split	min_samples_leaf	No_estimators
None, 5, 10, 15, 30, 60	None, 130, 180, 200, 250 y 300	2, 3 y 4	2, 3, 4 y 5	10, 50, 100, 150, 200

En la Tabla 28 se presentan los hiperparámetros que se modificaron en la búsqueda en malla para el SVR, algunos de ellos pueden ser omitidos, dependiendo del Kernel utilizado.

Tabla 28. Valores en hiperparámetros de SVR

Kernel	C	gamma
Sigmoid	1,2,3,4 y 5	0.1, 1, 2, 3, 4
Poly	1,2,3,4 y 5	0.1, 1, 2, 3, 4
RBF	1,2,3,4 y 5	0.1, 1, 2, 3, 4



# Capítulo 5. Resultados y discusión

## 5.1. Extracción de parámetros de un NMOS

En esta sección se presentan los resultados obtenidos de los entrenamientos que se realizaron utilizando RN, RF y SVR, tanto su evaluación y desempeño utilizando las métricas como el  $R^2$ , y porcentaje de error obtenido en curvas de validación seleccionadas como ejemplos para mostrar que tan bueno es el ajuste, entre la medición esperada y la simulada con los parámetros extraídos. También se presentan los resultados de la selección de características por parte de los AD sobre el conjunto de entradas.

### 5.1.1. Resultados de extracción en NMOS

Como se mencionó en 4.1.2., se realizó una selección de características en los ejemplos de entrenamiento, para contar únicamente con las más importantes. Antes de dicha selección los conjuntos de datos de T1 y T2, estaban conformados por muestras de entrada de 82 y 52 elementos respectivamente. Los 82 elementos de las muestras para T1, están conformados por 41 elementos de  $V_{GS}$  y para cada uno de ellos hay un valor diferente de  $I_S$ . Lo mismo sucede en las muestras de T2, 26 pertenecen a  $V_{GS}$  y 26 a  $I_D$  (medición en régimen lineal).

Después de la selección de características, para ambos conjuntos los datos pertenecientes a  $V_{GS}$  fueron eliminados, ya que no aportaban ninguna información para la predicción de la salida. En la Tabla 29 se presentan los datos de corriente eliminados en los dos conjuntos. El índice en la corriente representa la correspondencia con el valor de  $V_{GS}$ , es decir  $I_{S1}$  e  $I_{D1}$  corresponden al valor de  $V_{GS1}=0V$ ,  $I_{S2}$  e  $I_{D2}$  corresponden al valor de  $V_{GS1}=0.2V$  y así sucesivamente hasta llegar a 8V en el caso de T1 y a 5V en T2.

Tabla 29. Características de corriente eléctrica eliminadas

Conjunto de datos	Elemento/característica de eliminados
T1	$I_{S11}, I_{S24} - I_{S27}, I_{S29} - I_{S32}, I_{S35}$ y $I_{S37}$
T2	$I_{D9}$

Al finalizar la selección de características el tamaño de la muestra de entrada en T1 fue de 30 elementos y en T2 fue de 25 elementos. Para ambos conjuntos, las muestras de entrada

se redujeron más del 50%. Para facilitar la comprensión e interpretación de los resultados de la extracción de parámetros en ambos dispositivos NMOS, se dividieron en subsecciones, una que corresponde a T1 otra que corresponde a T2.

**Extracción de parámetros en T1**

A continuación, se presentan los modelos que obtuvieron el mejor desempeño en cada uno de los métodos de aprendizaje aplicados para la extracción de parámetros en el NMOS T1.

Para las RN, se encontró nuevamente que dos capas ocultas de 256 y 64 neuronas, función de activación “relu” y “linear” en la capa de salida, un factor de aprendizaje de 0.002 y 2000 épocas de entrenamiento brinda el mejor desempeño. En la Tabla 30 se presenta su  $R^2$  y MSE obtenido en la extracción en cada parámetro.

Tabla 30.  $R^2$  y MSE de RN para T1

Parámetro	$R^2$	MSE
UO	0.9934	0.02172871
$V_T$	0.9993	0.01160118
$R_S$	0.98	0.04802255
Promedio	0.9909	0.02711748

En RF, se encontraron los siguientes valores en sus los hiperparámetros como los mejores. Profundidad máxima de 15, número máximo de nodos terminales de 250, número mínimo de muestras en el nodo para realizar una división de 2 y 250 árboles en total. En la Tabla 31 se muestra su desempeño.

Tabla 31.  $R^2$  y MSE de RF para T1

Parámetro	$R^2$	MSE
UO	0.7037	0.14582785
$V_T$	0.9872	0.05068095
$R_S$	0.6303	0.20685545
Promedio	0.7737	0.13445475

En las SVR se debe entrenar una para cada parámetro, es decir que se tienen tres diferentes SVR. En la Tabla 32 se presentan los mejores hiperparámetros de cada SVR y su desempeño en cada parámetro. Como se puede observar, la estructura de las SVR para cada parámetro resultó ser la misma.

En la Figura 55 se presenta una gráfica con el  $R^2$  promedio por cada uno de los métodos, en el cual se ve reflejado el aprendizaje general que tuvieron, siendo las redes neuronales las

que alcanzaron la calificación más alta, ya que como se muestran en las tres tablas anteriores las RN, obtuvieron un  $R^2$  alto en los tres parámetros.

Tabla 32.  $R^2$  y MSE de SVR para T1

SVR	Kernel	C	Gamma	$R^2$	MSE
UO	“RBF”	4	“scale”	0.6208	0.16498569
$V_T$	“RBF”	4	“scale”	0.9373	0.11250000
$R_S$	“RBF”	4	“scale”	0.6731	0.19452746
Promedio	-	-	-	0.7437	0.1573377

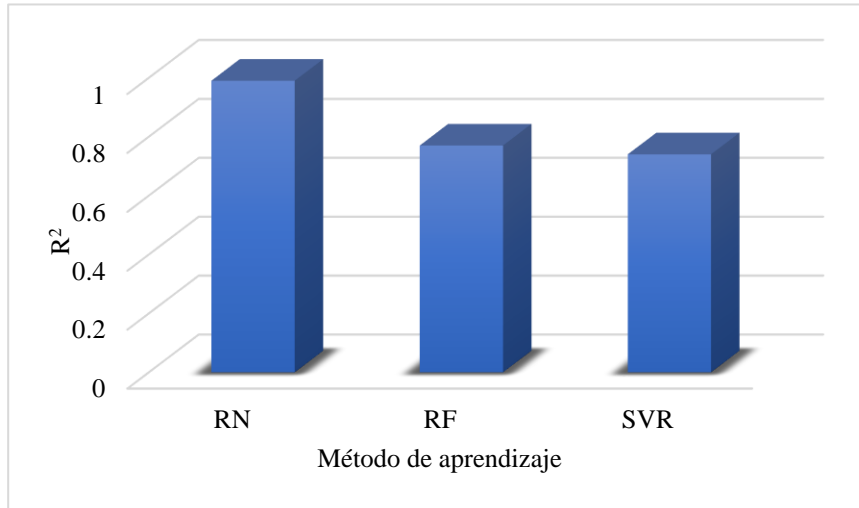


Figura 55.  $R^2$  de métodos de aprendizaje T1

A continuación, se presenta una curva de transferencia de validación perteneciente a T1, de la cual se extrajeron los parámetros. Se utilizaron los parámetros extraídos para simular al dispositivo y comparar el ajuste obtenido entre las curvas. En la figura 56 se muestra el ajuste obtenido entre las diferentes curvas. Los parámetros a extraer son  $UO= 500 \text{ cm}^2$ , un  $V_T= 0V$  y  $R_S= 0\Omega$ . Se calculó el porcentaje de error en cada una de ellas, las RN presentaron un error del 0.63%, RF del 2.7% y SVR del 0.16%.

Para realizar el cálculo del porcentaje de error utilizando la Eq. 67 en la pp 80, primero se calculó el área bajo la curva (Eq. 67) de cada una de las mediciones, y el resultado fue utilizado para el porcentaje de error. Esto se realizó ya que el porcentaje de error aplicado para cada valor de  $V_{GS}$  obtenía algunas regiones con un sesgo alto que afectaba el error promedio.

$$A = \int_a^b f(x)dx \approx (b - a) \left[ \frac{f(a)+f(b)}{2} \right] \quad (67)$$

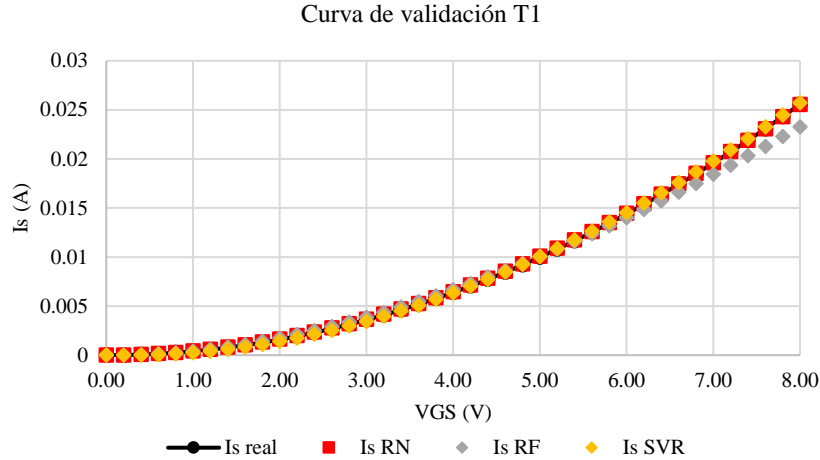


Figura 56. Curva de validación T1

### Extracción de parámetros en T2

A continuación, se presentan los modelos que obtuvieron el mejor desempeño en cada uno de los métodos de aprendizaje aplicados para la extracción de parámetros en el NMOS T2.

Se utilizó la misma estructura de red que en T1 para el entrenamiento utilizando el conjunto de datos de T2, dando como resultado un buen desempeño. En la Tabla 33 se presenta su  $R^2$  y MSE obtenido en la extracción en cada parámetro.

Tabla 33.  $R^2$  y MSE de RN para T2

Parámetro	$R^2$	MSE
UO	0.9981	0.01206389
$V_T$	0.9991	0.01419075
$R_S$	0.9294	0.09571976
Promedio	0.97	0.04065813

En RF, se encontraron los siguientes valores en sus los hiperparámetros como los mejores. Profundidad máxima de 20, número máximo de nodos terminales de 250, número mínimo de muestras en el nodo para realizar una división de 2 y 250 árboles en total, una estructura muy similar al RF de T1. En la Tabla 34 se muestra su desempeño.

Tabla 34.  $R^2$  y MSE de RF para T2

Parámetro	$R^2$	MSE
UO	0.8616	0.10427257
$V_T$	0.9904	0.04882441
$R_S$	0.4192	0.27457929
Promedio	0.7570	0.14255875

En las SVR para T2, también se entrenó una SVR para cada parámetro. En la tabla 35 se presentan sus hiperparámetros de cada SVR y su desempeño. Como se puede observar, la estructura de las SVR para cada parámetro resultó ser la misma, como lo fue en T1, pero en esta ocasión el valor en gamma cambió.

Tabla 35. R<sup>2</sup> y MSE de SVR para T2

SVR	Kernel	C	Gamma	R <sup>2</sup>	MSE
UO	“RBF”	4	3	0.9365	0.07062774
V <sub>T</sub>	“RBF”	4	3	0.9834	0.06424065
R <sub>S</sub>	“RBF”	4	3	0.6143	0.22376359
Promedio	-	-	-	0.8447	0.11954399

En la Figura 57 se presenta una gráfica con el R<sup>2</sup> promedio por cada uno de los métodos, en el cual se ve reflejado el aprendizaje general que tuvieron, siendo las redes neuronales una vez más las que alcanzaron la calificación más alta seguidas por SVR y al fina RF.

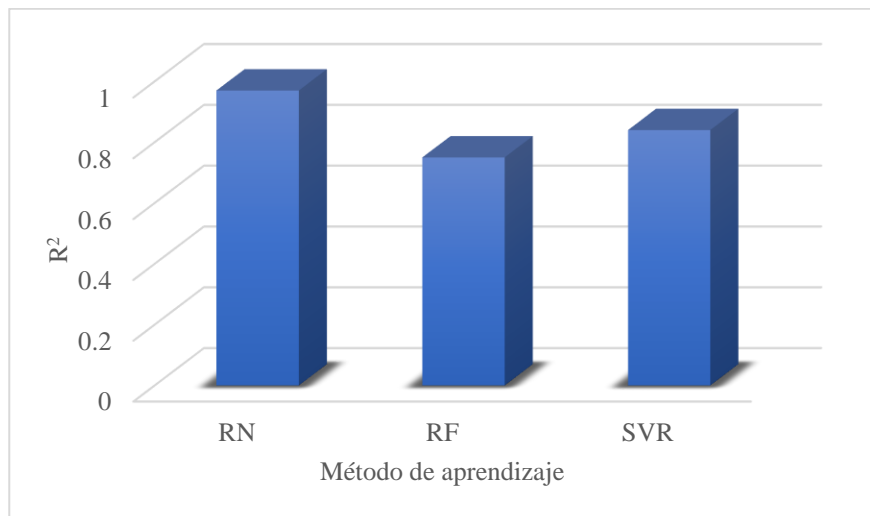


Figura 57. R<sup>2</sup> de métodos de aprendizaje T2

A continuación, se presenta una curva de transferencia de validación perteneciente a T2, de la cual se extrajeron los parámetros. Se utilizaron los parámetros extraídos para simular al dispositivo y comparar el ajuste obtenido entre las curvas. En la Figura 58 se muestra el ajuste obtenido entre las diferentes curvas. Los parámetros a extraer son UO= 500cm<sup>2</sup>/Vs, un V<sub>T</sub>= 2.5V y R<sub>S</sub>= 0Ω. Se calculó el porcentaje de error en cada una de ellas, las RN presentaron un error del 0.7%, RF del 5.1% y SVR del 15.2%.

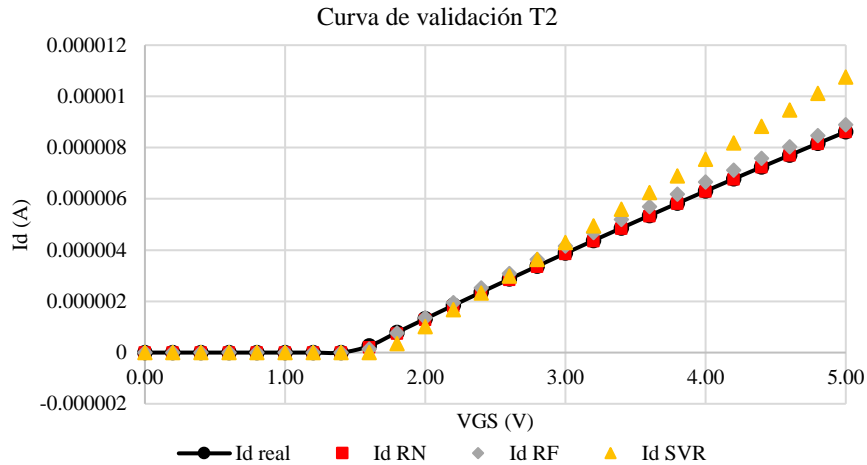


Figura 58. Curva de validación T2

## 5.2. Extracción de parámetros de un TFT IGZO

En esta sección se presentan los resultados de los entrenamientos de los diferentes métodos de aprendizaje usados para la extracción de parámetros en el TFT IGZO. Los resultados se dividen en la primera extracción que solamente incluyo a  $K_P$  y  $V_T$ , esta experimentación se realizó en región de saturación y lineal. La segunda extracción es de  $K_P$ ,  $V_T$  y  $R_C$ , esta experimentación solo se realizó en región de saturación.

La extracción de parámetros que se presentará en esta sección se realizó sobre mediciones experimentales de transistores desarrollados por el Centro de Nanociencias y Micro y Nanotecnologías del Instituto Politécnico Nacional.

### 5.2.1. Extracción de $K_P$ y $V_T$

#### *Extracción en régimen de saturación*

A continuación, se presenta el desempeño obtenido por los mejores 8 modelos de red entrenados para la extracción de  $K_P$  y  $V_T$ . En la Tabla 36 se tiene el MSE y  $R^2$  de cada modelo de red (las características se encuentran en la tabla 20). En la Figura 59 se graficó el  $R^2$  de los modelos donde es fácil identificar que, aunque no es grande la diferencia el modelo 4 obtuvo el mejor aprendizaje.

En la Tabla 37 se presenta el  $R^2$  que obtuvo cada modelo en ambos parámetros, de esta manera es posible identificar que parámetro es más difícil de identificar. Con la Figura 60 se puede identificar que para el parámetro  $K_P$  (gráfico izquierdo) tu obtuvo un menor

desempeño con el modelo 4 con el  $R^2$  más alto, en el gráfico derecho se observa que el parámetro  $V_T$  es más identificable, obteniendo un  $R^2$  mayor al 0.99 para todos los modelos de RN.

Tabla 36. Desempeño de RNs en entrenamiento para TFTs

Modelo de RN	MSE	$R^2$
1	$1.150 \times 10^{-4}$	0.997
2	$4.474 \times 10^{-4}$	0.991
3	$3.119 \times 10^{-4}$	0.994
4	$8.763 \times 10^{-5}$	0.998
5	$1.802 \times 10^{-4}$	0.996
6	$1.476 \times 10^{-4}$	0.997
7	$1.643 \times 10^{-4}$	0.996
8	$1.139 \times 10^{-4}$	0.997

Tabla 37. Desempeño de RNs por parámetro

Modelo de RN	KP	$V_T$
1	0.9964	0.9992
2	0.9915	0.9919
3	0.9896	0.9984
4	0.9974	0.9992
5	0.9935	0.9995
6	0.9964	0.998
7	0.9955	0.9983
8	0.9963	0.9993

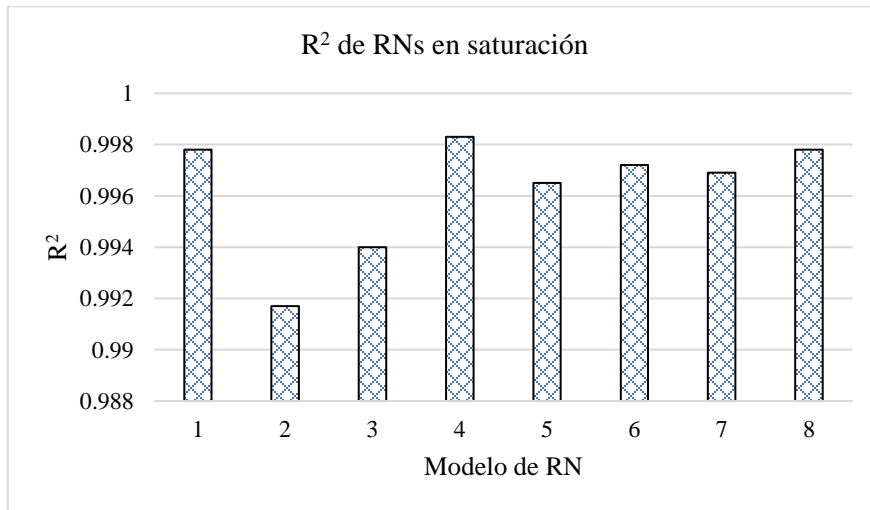


Figura 59. R2 de modelos de RN para extracción de KP y  $V_T$

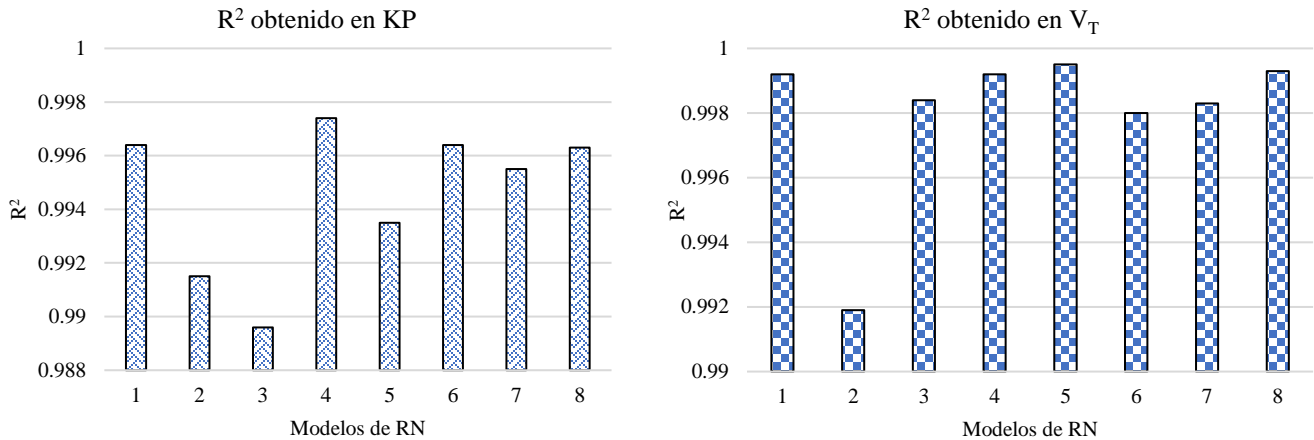


Figura 60. R<sup>2</sup> obtenido por los modelos de RN por parámetro

Debido a que son dispositivos recién fabricados no hay un parámetro exacto el cual usar como referencia para medir la exactitud que tendrá la extracción realizada por los métodos de aprendizaje. La métrica utilizada será el porcentaje de error que hay entre la curva experimental con las modeladas con los parámetros extraídos.

En la Figura 61 se muestra el modelado que se obtuvo en dos de las mediciones experimentales (TFT1 diferente bloque de la oblea). En ambas curvas I-V se observa un buen ajuste entre la medición real y las modeladas con los parámetros extraídos. Con la intención de no saturar el gráfico, solamente se presentan los modelados de los mejores 4 resultados. Para estas dos curvas los modelos de RN 1, 4 y 7 presentaron el porcentaje de error más bajos.

En la Tabla 38 se presentan los valores de los parámetros extraídos en ambas curvas, y el porcentaje de error que obtuvo el modelado con ellos. De la tabla se observa que para el primer dispositivo se tiene un KP promedio de  $1.5205 \times 10^{-6} \text{A/V}^2$  y un  $V_T$  promedio de 2.594 V, en el segundo dispositivo se tiene  $1.4716 \times 10^{-6} \text{A/V}^2$  y 3.19 V respectivamente. Aunque ambos transistores tienen la misma geometría el segundo presentó un voltaje umbral mayor, esto indica que no todos los dispositivos tendrán los mismos parámetros, aunque sean fabricados en una misma oblea. El porcentaje de error y los modelados obtenidos, aseguran que los parámetros extraídos son muy cercanos a los parámetros que tienen las mediciones experimentales.



Tabla 38. Parámetros extraídos en mediciones físcas de TFTs

Medición: TFT1 bloque 21				
Modelo de RN	KP extraído (A/V <sup>2</sup> )	V <sub>T</sub> extraído (V)	Error (%)	
1	1.6107×10 <sup>-6</sup>	2.667	0.54	
4	1.4631×10 <sup>-6</sup>	2.539	2.06	
5	1.4135×10 <sup>-6</sup>	2.555	2.32	
7	1.5947×10 <sup>-6</sup>	2.616	3.98	
Medición: TFT1 bloque 22				
Modelo de RN	KP extraído (A/V <sup>2</sup> )	V <sub>T</sub> extraído (V)	Error (%)	
1	1.5615×10 <sup>-6</sup>	3.248	1.25	
4	1.4054×10 <sup>-6</sup>	3.169	0.79	
7	1.4669×10 <sup>-6</sup>	3.181	2.23	
8	1.4529×10 <sup>-6</sup>	3.164	3.09	

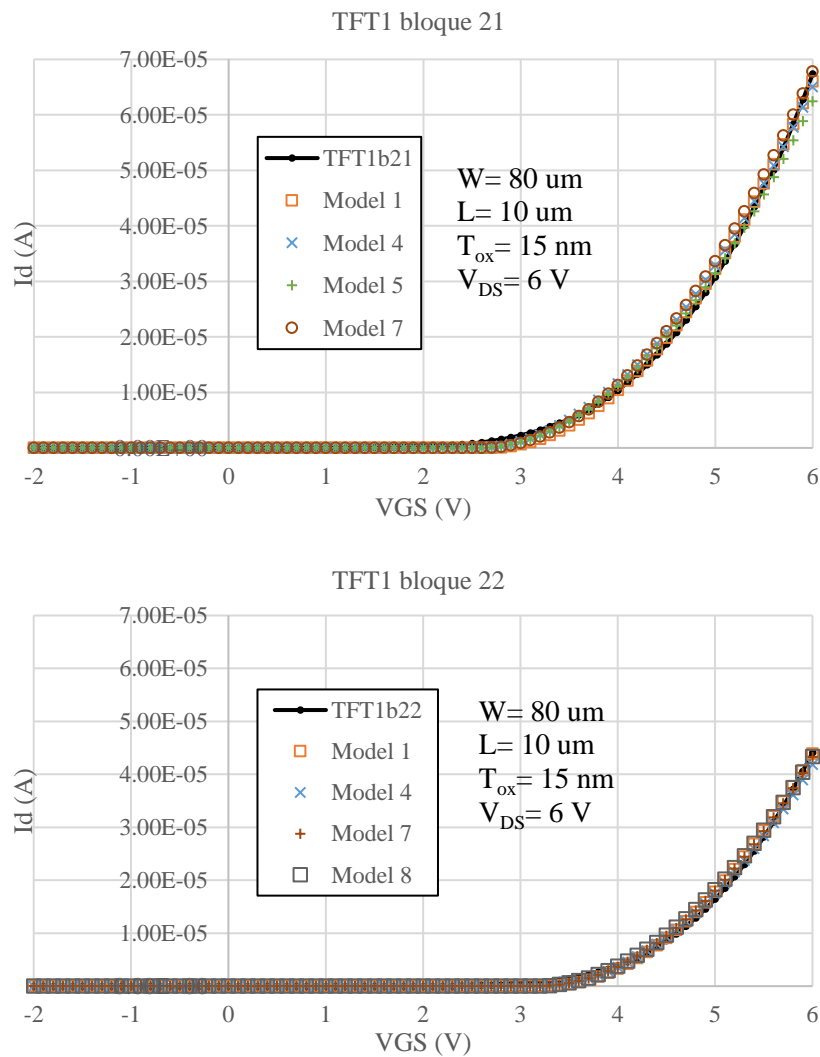


Figura 61. Extracción y simulación de 2 parámetros del TFT IGZO

**Extracción en régimen lineal**

A continuación, se presentan los resultados de los entrenamientos realizados para la extracción de parámetros en régimen lineal. Los resultados se presentan en la misma forma que en subsección pasada (saturación) por lo que se tendrá una explicación menos detallada.

En la Tabla 39 se presenta el MSE y  $R^2$  obtenido por los mejores 8 modelos de RN. En estos entrenamientos el  $R^2$  es muy similar al de régimen de saturación, por encima del 0.99, pero en régimen lineal el MSE se encuentra en  $\times 10^{-5}$ , aunque esto puede deberse a que en régimen lineal el valor de la corriente eléctrica es menor lo que reduce la escala del error, no necesariamente significa que hubo mayor exactitud. En la Figura 62 se presenta de manera grafica el  $R^2$  obtenido por los diferentes modelos de RN, donde es fácil notar que el modelo con el aprendizaje más bajo fue el modelo 7 y con mayor aprendizaje fueron los modelos 5 y 8. En la Figura 63 se tiene el  $R^2$  por parámetros, donde es posible observar en régimen lineal ambos parámetros están por encima del 0.99. Con estos resultados es posible afirmar que en régimen lineal las RN tienen un mayor aprendizaje con en saturación, aunque realizar extracción en lineal no suele ser tan común.

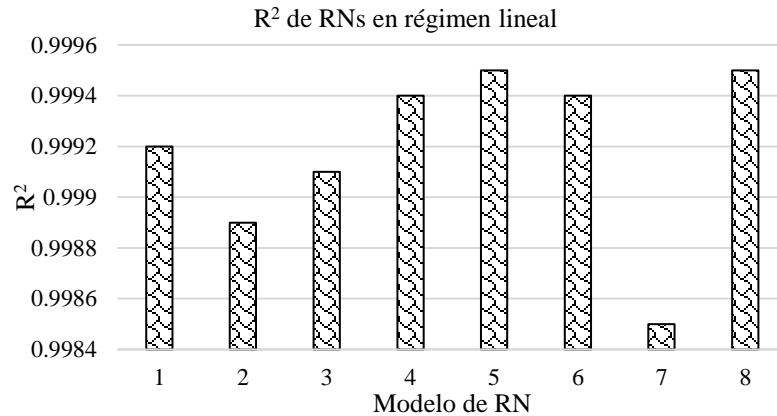


Figura 62.  $R^2$  en entrenamiento de RNs (lineal)

Tabla 39. Desempeño de RNs en entrenamiento para TFTs (lineal)

Modelo de RN	MSE	$R^2$
1	$3.964 \times 10^{-5}$	0.999
2	$5.598 \times 10^{-5}$	0.998
3	$4.621 \times 10^{-5}$	0.999
4	$2.714 \times 10^{-5}$	0.999
5	$2.432 \times 10^{-5}$	0.999
6	$3.141 \times 10^{-5}$	0.999
7	$7.609 \times 10^{-5}$	0.998
8	$2.348 \times 10^{-5}$	0.999

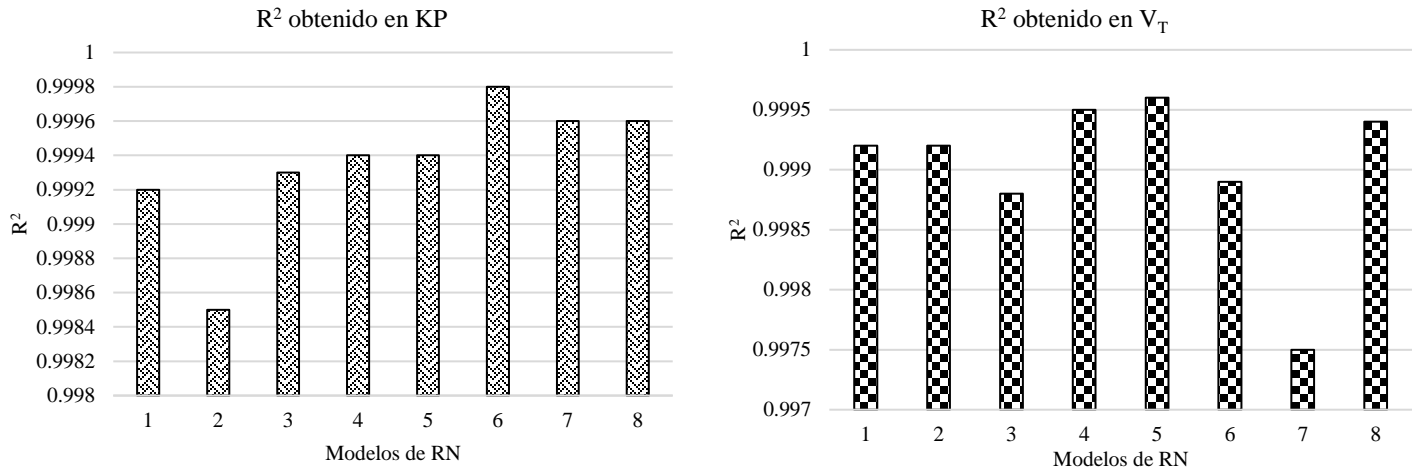


Figura 63. R<sup>2</sup> en entrenamiento de RNs por parámetro (lineal)

A continuación, se presentan las simulaciones realizadas con los parámetros extraídos en régimen lineal. La Figura 64 presenta la curva de transferencia del TFT1 en régimen de saturación, de la cual se observa un buen ajuste con las curvas modeladas con los parámetros extraídos, en este caso, los mejores modelos de RN fueron el 5, 6, 7 y 8. En la Tabla 40 se presentan los valores extraídos y los porcentajes de error para cada uno de los modelos. Donde se obtiene un promedio en KP de  $1.7344 \times 10^{-6}$  y un V<sub>T</sub> promedio de 2.943 V. En comparación con el KP=  $1.5205 \times 10^{-6} \text{ A/V}^2$  y V<sub>T</sub>= 2.594V obtenidos en saturación, parece que hay una mayor movilidad y un umbral mayor en régimen lineal, aunque estos resultados se podrían deber al modelo usado para simular el transistor (se usó el modelo 1), el cual es un modelo lineal y se pierde información.

Tabla 40. Parámetros extraídos de mediciones de TFTs (lineal)

Medición: TFT1 bloque 21				
Modelo de RN	KP extraído (A/V <sup>2</sup> )	V <sub>T</sub> extraído (V)	Error (%)	
5	$1.7167 \times 10^{-6}$	2.877	4.9	
6	$1.8162 \times 10^{-6}$	3.007	2.26	
7	$1.7082 \times 10^{-6}$	2.977	2.29	
8	$1.6967 \times 10^{-6}$	2.914	1.21	

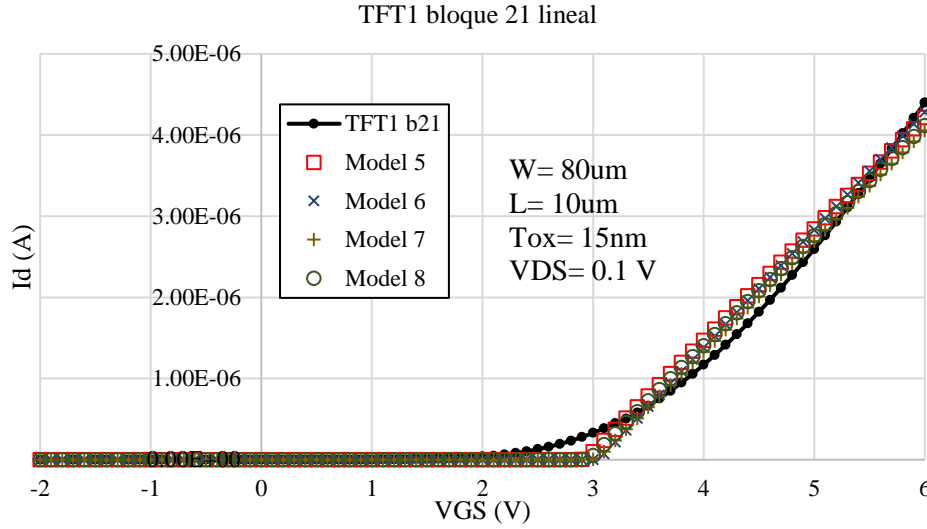


Figura 64. Simulación y modelado de TFT1 con parámetros extraídos

**Análisis de extracción con  $R_C$  añadida**

En esa subsección se presentan los resultados obtenidos de una extracción de parámetros, la cual se realizó usando el modelado del TFT1 en régimen de saturación a la cual se le añadió resistencia de contacto usando el simulador.

Dejando como constantes los parámetros de  $KP= 1.678 \times 10^{-6} A/V^2$  y  $V_T= 2.821 V$ , dentro de la simulación, se realizó un barrido en  $R_C$  desde  $0 \Omega$  hasta  $64 k\Omega$ . En la Figura 65 se presentan las diferentes curvas resultantes, donde la disminución de corriente eléctrica conforme la resistencia aumenta, teniendo una pérdida importante cuando  $R_C > 4 k\Omega$ .

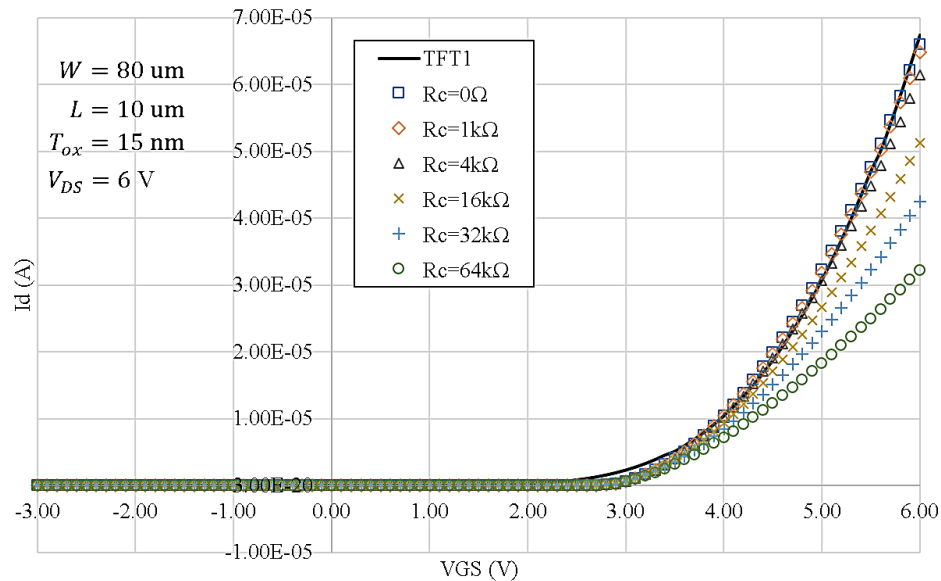


Figura 65. TFT1 y curvas con  $R_C$  añadida

Dichas curvas con  $R_C$  añadida se utilizaron para alimentar a una RN ya entrenada de la sección 5.2.1. con el objetivo de observar el resultado que las RNs brindarían, ya que no aprendieron de ejemplos con  $R_C$ . En la Figura 66a se presentan las curvas I-V correspondientes a  $R_C$  de 0 hasta 4k $\Omega$ , con el respectivo modelada usando los parámetros extraídos por la RN, mientras que en Fig. 66b se presentan las curvas con  $R_C$  de 16 a 64 k $\Omega$ , de igual manera con sus modelados con los parámetros extraídos. En este experimento se observó que las RNs fueron capaces de compensar los valores de KP y  $V_T$  para modelar con gran exactitud las curvas de entrada, aunque estas tuvieran  $R_C$  añadida. Los parámetros extraídos y el porcentaje de error para cada una de las curvas se presentan en la Tabla 41.

Tabla 41. Parámetros extraídos de curvas con  $R_C$  añadida

Curva	KP (A/V <sup>2</sup> )	$V_T$ (V)	Error (%)
0 $\Omega$	$1.63659 \times 10^{-6}$	2.708	1.20
1 k $\Omega$	$1.59228 \times 10^{-6}$	2.701	1.88
4 k $\Omega$	$1.48569 \times 10^{-6}$	2.662	1.21
16 k $\Omega$	$1.19525 \times 10^{-6}$	2.547	1.34
32 k $\Omega$	$9.45068 \times 10^{-7}$	2.494	2.38
64 k $\Omega$	$6382629 \times 10^{-7}$	2.498	11.48

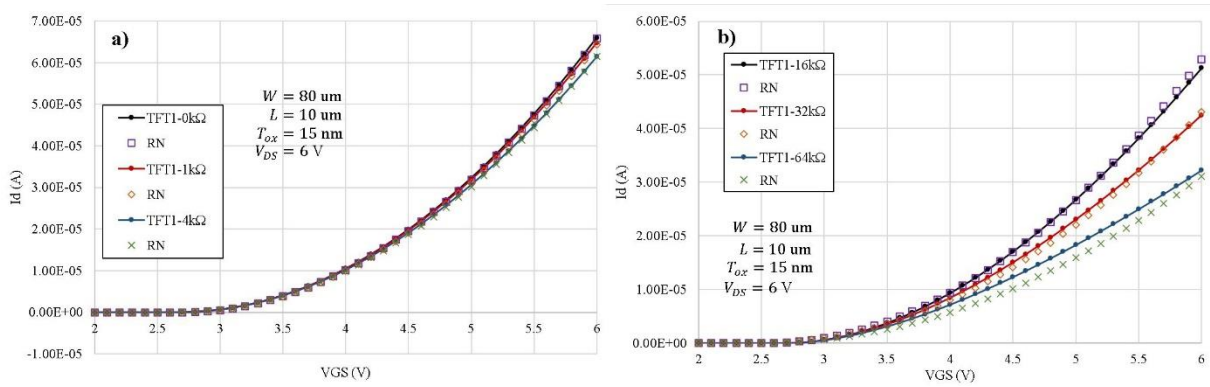


Figura 66. Simulación y modelado de curvas con parámetros extraídos

### No dependencia de parámetros geométricos ( $W$ y $L$ )

En esta subsección se presentan los resultados obtenidos de la extracción usando los métodos de RN, RF y SVR, los cuales fueron entrenados con las muestras de curvas I-V normalizadas. La normalización se realizó para eliminar los parámetros geométricos  $W$  y  $L$ , esto con la finalidad de poder utilizar un conjunto de entrenamiento, perteneciente al TFT1 ( $W = 80 \mu\text{m}$  y  $L = 10 \mu\text{m}$ ), para realizar extracción a dispositivos de otras dimensiones (Tabla 25). La normalización de cada muestra utilizó la Eq. (68).

$$I_{\text{norm}} = \frac{L}{W} I_D \quad (68)$$

En la Tabla 42 se presenta el desempeño de los métodos mencionados, tanto con el conjunto de entrenamiento como en validación, en la cual se identifica a las RN con el mejor aprendizaje, seguidas de RF y por último SVR. En la Tabla 43 se presentan los parámetros extraídos.

Tabla 42. Desempeño en entrenamiento en no dependencia de parámetros geométricos

Método	Entrenamiento		Validación	
	R <sup>2</sup>	MSE	R <sup>2</sup>	MSE
RN	0.999	1.58263×10 <sup>-5</sup>	0.999	2.737840×10 <sup>-5</sup>
RF	0.995	1.31892×10 <sup>-2</sup>	0.967	3.850242×10 <sup>-2</sup>
SVR	0.89	7.33846×10 <sup>-2</sup>	0.901	7.345997×10 <sup>-2</sup>

Se utilizaron 6 mediciones físicas, pertenecientes a dispositivos de diferentes dimensiones. Después de la extracción de parámetros, se obtuvo el siguiente porcentaje de error promedio: las RN obtuvieron un error del 1%, RF obtuvo un error del 21.04%, RF obtuvo un 22.23% y adicionalmente, se hizo una extracción analítica para comparar los resultados de los métodos de aprendizaje supervisado, con la cual se obtuvo un error promedio de 9.41%. En la Fig. 67a se presenta la medición física correspondiente al TFT2, donde se observa un buen ajuste por parte de RN y el AM. En la Fig. 67b se presenta la medición física del TFT3, donde nuevamente RN y AM presentan los mejores ajustes y para esta curva, SVR se acercó al comportamiento esperado.

Tabla 43. Parámetros extraídos en TFTs al no depender de W y L

Medición: TFT2 bloque 24				
Método	KP extraído (A/V <sup>2</sup> )	V <sub>T</sub> extraído (V)	Error (%)	
RN	1.70078×10 <sup>-6</sup>	2.508	1.06	
RF	1.27700×10 <sup>-6</sup>	1.969	16.71	
SVR	2.36728×10 <sup>-6</sup>	2.654	23.76	
AM	1.70384×10 <sup>-6</sup>	2.64	9.8	
Medición: TFT3 bloque 24				
Método	KP extraído (A/V <sup>2</sup> )	V <sub>T</sub> extraído (V)	Error (%)	
RN	1.39913×10 <sup>-6</sup>	2.616	1.23	
RF	1.21600×10 <sup>-6</sup>	2.163	28.23	
SVR	2.06868×10 <sup>-6</sup>	2.924	12.42	
AM	1.37439×10 <sup>-6</sup>	2.72	9.45	

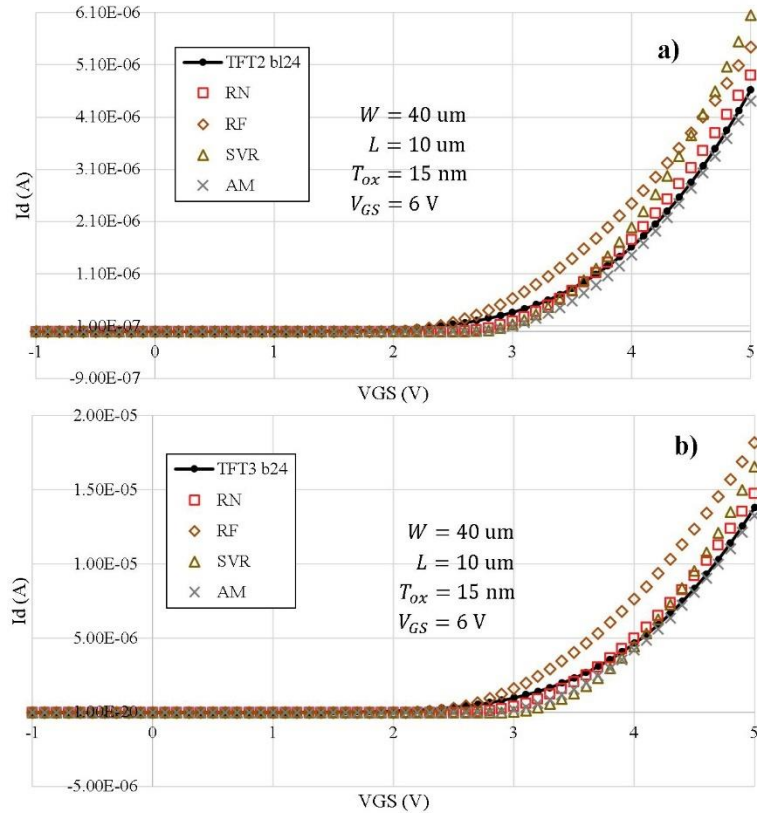


Figura 67. Simulación y modelado de TFTs al no depender de W y L

### 5.2.2. Extracción de $K_P$ , $V_T$ y $R_C$

En esta sección se presentan los resultados del entrenamiento de los métodos de aprendizaje que se utilizaron para extraer tres parámetros de los TFTs IGZO. Esta extracción se llevó a cabo únicamente en la región de saturación y el dataset fueron diseñados para los dispositivos con un  $W = 80 \text{ um}$  y  $L = 10 \text{ um}$ .

#### *Desempeño de los métodos de aprendizaje*

En esta subsección se presenta el desempeño que presentaron las RNs, RF y SVR en la extracción de  $K_P$ ,  $V_T$  y  $R_C$ . Después de realizar el entrenamiento métodos mencionados, se registró el desempeño que se presenta en la Tabla 44. Dados los datos se observa que las RNs obtuvieron el desempeño más bajo, tanto en el entrenamiento como en validación. En la Tabla 45 se presenta el  $R^2$  obtenido en cada uno de los parámetros, únicamente en las muestras de validación y con la gráfica presente en Fig. 68 se observa que el desempeño bajo por parte de las RNs se debe a la pobre identificación de  $R_C$ , siendo la más baja de los tres métodos utilizados.

Tabla 44. Desempeño general para la extracción de tres parámetros del TFT

Método	Entrenamiento		Validación	
	MSE	R <sup>2</sup>	MSE	R <sup>2</sup>
RNs	$3.16253 \times 10^{-2}$	0.6981	$3.54612 \times 10^{-2}$	0.6984
RF	$5.36498 \times 10^{-2}$	0.9446	$1.17946 \times 10^{-1}$	0.7425
SVR	$1.68253 \times 10^{-1}$	0.7069	$1.24951 \times 10^{-1}$	0.7462

Tabla 45. R<sup>2</sup> obtenido por métodos aprendizaje por parámetro

Método	Muestras de validación		
	Parámetros		
	KP	V <sub>T</sub>	R <sub>C</sub>
RNs	0.9485	0.9959	0.1508
RF	0.9708	0.9948	0.262
SVR	0.9473	0.9234	0.368

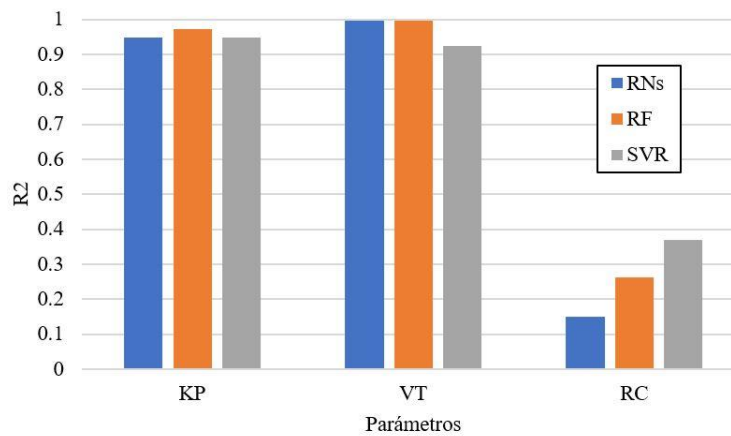


Figura 68. R<sup>2</sup> de cada parámetro obtenido por los métodos de aprendizaje

### Extracción y simulación de KP, V<sub>T</sub> y R<sub>C</sub>

En esta subsección se presenta los parámetros extraídos de las mediciones físicas del TFT1, y la simulación del dispositivo usando dichos parámetros. De esta manera se identificará el mejor método de aprendizaje para realizar la tarea de extracción de parámetros.

En la Tabla 46 se presentan los parámetros extraídos de la medición TFT1 la cual fue tomada con un V<sub>GS</sub> negativo a positivo del bloque 24. También se encuentra la medición del TFT1 del bloque 22, igualmente con V<sub>GS</sub> de negativo a positivo.

Usando los parámetros de la tabla anterior se simuló el transistor y se obtuvieron los ajustes de la Figura 69. En Fig. 69a, se observa el mejor ajuste para SVR con un error del 3.33%, seguidas de las RNs con un error del 4.39% y en tercer lugar RF con un 11.67%. A simple vista se observa que para mejorar el ajuste de SVR y RN, se debe disminuir ligeramente a R<sub>C</sub>, de esta manera la corriente aumentaría en V<sub>GS</sub>>5.5 V, permitiendo un mejor



ajuste. En Fig. 69b se tiene el mejor ajuste por parte de las RNs con un error del 7.35% seguidas de RF con 13.23% y en tercer lugar SVR con un 35.94%. En este caso para mejorar la extracción de RN se debería disminuir a  $R_C$ , en RF se debería aumentar  $V_T$  y en SVR se observa una corriente mucho mayor, por lo que se debería disminuir a  $K_P$ .

Tabla 46. Parámetros extraídos de TFTs (tres parámetros)

Parámetros extraídos				
Método	KP (A/V <sup>2</sup> )	V <sub>T</sub> (V)	R <sub>C</sub> (Ω)	Error (%)
TFT 1 medición NP bloque 24				
RN	1.30647×10 <sup>-6</sup>	2.700	3.077×10 <sup>3</sup>	4.39
RF	1.15076×10 <sup>-6</sup>	2.366	3.830×10 <sup>3</sup>	11.67
SVR	1.79588×10 <sup>-6</sup>	2.83	17.001×10 <sup>3</sup>	3.33
TFT 1 medición 5 bloque 22				
RN	1.38640×10 <sup>-6</sup>	3.192	3.055×10 <sup>3</sup>	7.35
RF	1.3229×10 <sup>-6</sup>	2.941	3.738×10 <sup>3</sup>	13.23
SVR	1.85535×10 <sup>-6</sup>	3.101	2.397×10 <sup>3</sup>	35.94

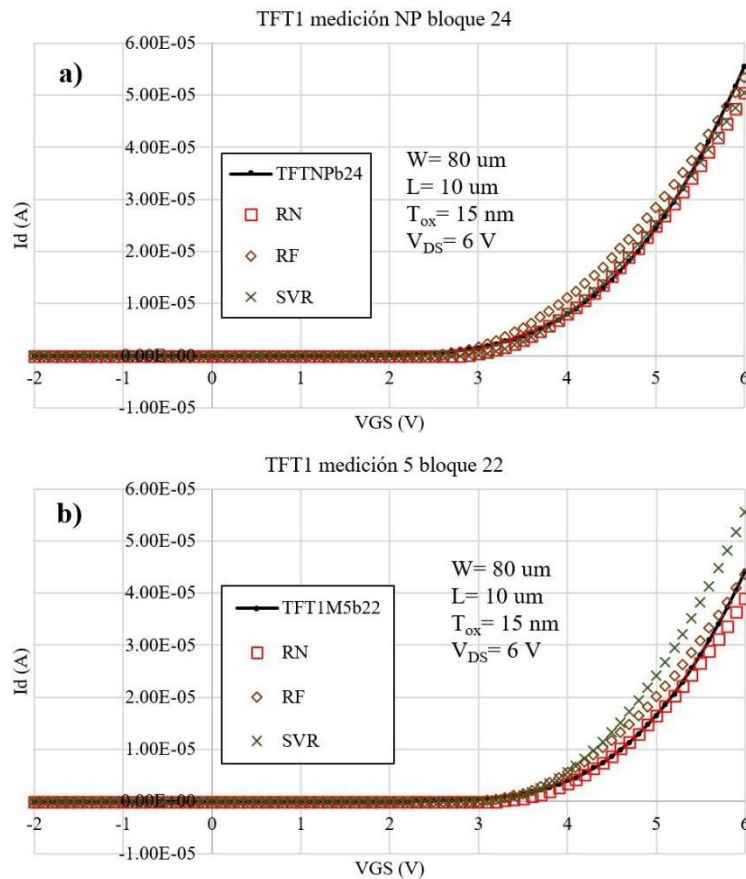


Figura 69. Simulación y modelado de TFT usando tres parámetros extraídos

En general se obtienen resultados con porcentajes de error aceptables, con los cuales se obtienen buenos ajustes. El resultado de la extracción por parte de los métodos de aprendizaje dependerá de la medición física de interés, y para algunas un método será mejor y en otras el mejor método otro.

### 5.3. Extracción de circuito de carga resistiva

En esta sección se presentan los resultados que brindaron las redes diseñadas y entrenadas con el conjunto de datos perteneciente al inversor de carga resistiva. El proceso de extracción se divide en dos, la primera extracción fue de 2 parámetros y la segunda fue de 4 parámetros.

#### 5.3.1. Resultado de extracción de 2 parámetros

En la Tabla 20 se presentó una lista de los modelos entrenados en Python. En esta sección se presenta la evaluación de la red que tuvo mejor desempeño. Ahora en la Tabla 47 se presentan los resultados que obtuvieron en la validación para los 3 conjuntos de datos. Se tienen dos de las métricas que brinda la librería de Keras al finalizar los entrenamientos, ACC y MSE, además se calculó el coeficiente de determinación  $R^2$  como una métrica extra.

Tabla 47. Resultados en validación por tipo de normalización

Red de 2 capas, 256-32n, lr=0.001 y 1000 épocas			
Tipo de reescalado	Accuracy	MSE	$R^2$
Magnitud	92.12%	9.0066E-03	0.87
Valor máximo	95.37%	1.1295E-02	0.90
Distribución normal	94.99%	1.2036E-01	0.86

Como ya se ha mencionado, una vez que finaliza el entrenamiento de la RN esta lista para usarse, para ello se necesitan de nuevos datos que no estuvieron presentes en el conjunto de datos tanto de entrenamiento como en validación. Los nuevos datos pueden tener valores intermedios dentro del rango de conocimiento que aprendió la red. Por ejemplo, el rango de  $V_T$  fue de  $-1$  a  $5V$  con aumentos de  $1V$ . Entonces se pueden utilizar datos donde  $V_T$  sea igual a  $1.5V$ ,  $2.3V$ ,  $4.4V$  etc. Lo mismo para  $R_L$ , si se utilizan datos fuera del rango la red generará resultados sin sentido.

**Simulación de RN**

La red fue alimentada con las curvas mencionadas y serán presentados de tres formas. La primera serán todos los datos que forman la curva completa (51 puntos por curva), que será llamada Ext 1. La segunda estará formada por los puntos de la región de transición, la cantidad depende del tamaño de dicha región, llamada Ext 2. La tercera simulación contara solamente con el punto de switcheo (punto medio de la región de transición), este se encuentra al calcular la derivada de la curva, llamado Ext 3. Lo anterior es para encontrar la mejor forma de dar los datos a la red para obtener los mejores resultados.

En las Tablas 48 y 49 se encuentran los resultados que brindo la red con las tres formas de darle los datos. A estos resultados se le calculo el porcentaje de error (Tablas 50 y 51) donde se puede confirmar que la Ext. 2 presenta el menor error de 0.2% en  $R_L$  y 3.27% en  $V_T$ , seguido de Ext. 3 con 1.03% y 4.75% respectivamente. En las tablas hay una columna DM (*Direct method*), que representa los resultados de la extracción utilizando el método analítico UMEM. Este último solo abarca la etapa analítica en donde se calcula el valor de los parámetros de forma matemática usando el modelo matemático del TFT, no se lleva la etapa de ajuste del valor a prueba y error.

Para calcular el porcentaje de error entre las diferentes curvas del circuito inversor, se utilizó la ecuación 58, para cada valor de  $V_{in}$ .

$$\%error = \left| \frac{V_{aprox} - V_{real}}{V_{real}} \right| \times 100 \tag{68}$$

Tabla 48. Extracción de  $R_L$

Curva	Valor real	Ext. 1	Ext. 2	Ext. 3	DM
1	70k $\Omega$	104.5k $\Omega$	69.47k $\Omega$	47.42k $\Omega$	149k $\Omega$
2	150k $\Omega$	158.08k $\Omega$	148.15k $\Omega$	149.35k $\Omega$	216k $\Omega$
3	550k $\Omega$	496.4k $\Omega$	566.51k $\Omega$	583.11k $\Omega$	353k $\Omega$
4	950k $\Omega$	910.18k $\Omega$	895.83k $\Omega$	925.86k $\Omega$	423k $\Omega$

Tabla 49. Extracción de  $V_T$

Curva	Valor real	Ext. 1	Ext. 2	Ext. 3	DM
1	2.3V	2.58V	2.43V	2.38V	1.8V
2	1.8V	2.02V	1.87V	1.81V	1.27V
3	3.5V	3.66V	3.55V	3.55V	2.84V
4	1.4V	1.56V	1.37V	1.59V	0.7V

Para observar mejor los resultados se hace la simulación del circuito inversor con los parámetros que obtuvo la red, de esta forma se puede observar que tan cerca están las curvas con respecto a la real. En las Figura 70 y 71 se presentan las cuatro curvas graficadas, donde

es posible afirmar que la Ext. 2 y Ext. 3 son las más cercanas a la real (línea en negro), mientras que la que representa los resultados de la extracción analítica es la más lejana.

Tabla 50. Porcentaje de error en extracción de  $R_L$

Curva	Ext. 1	Ext. 2	Ext. 3	DM
1	3.59%	0.07%	3.49%	6.77%
2	0.44%	0.10%	0.03%	3.05%
3	0.77%	0.22%	0.44%	3.35%
4	0.31%	0.42%	0.18%	5.87%
Promedio	1.27%	0.2%	1.03%	4.76%

Tabla 51. Porcentaje de error de extracción en  $V_T$

Curva	Ext. 1	Ext. 2	Ext. 3	DM
1	12.17%	5.65%	3.47%	21.73%
2	12.22%	3.88%	0.55%	29.44%
3	4.57%	1.42%	1.42%	18.85%
4	11.42%	2.14%	13.57%	50.0%
Promedio	10.09%	3.27%	4.75%	30.0%

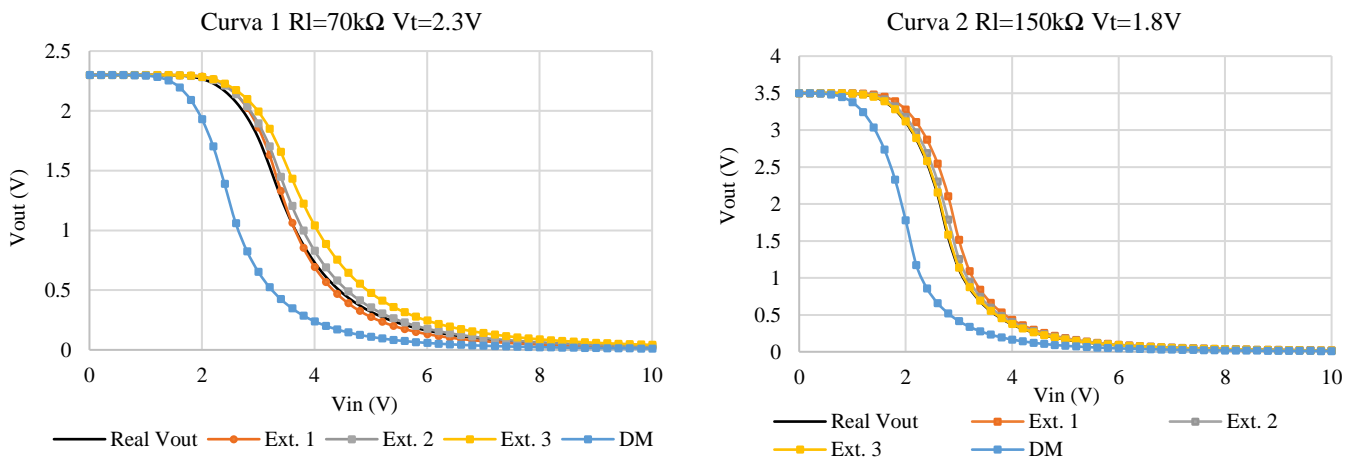


Figura 70. Simulación de inversor curvas 1 y 2

Se decidió que en los resultados de la extracción de parámetros utilizando la red entrenada, solo se realizaría la extracción en las 4 curvas de la Tabla 48 comparando los tres tipos de alimentación para la red (Ext. 1, Ext.2, y Ext. 3). Dejando a un lado la simulación de los parámetros que se obtuvieron en el conjunto de entrenamiento, ya que lo importante es probar la red con datos desconocidos por la misma.

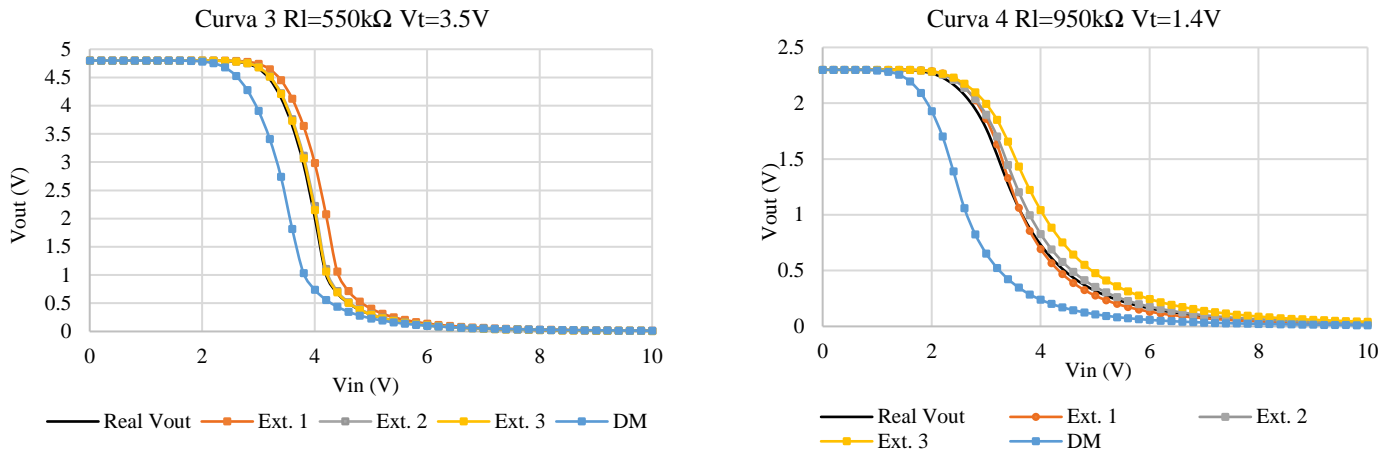


Figura 71. Simulación de inversor curvas 3 y 4

**Pruebas con datos fuera del conocimiento de la red**

En los resultados anteriores se demostró que la RN es capaz de extraer los parámetros de las curvas de un circuito inversor, con un error considerablemente mínimo, incluso con curvas que no aprendió directamente, pero que se encuentran dentro del conocimiento del conjunto de entrenamiento al tener valores intermedios en los rangos de los parámetros. A continuación, se presentan los resultados que la RN generó al ser alimentada con curvas a las que se les modificó el valor en los parámetros geométricos W y L. Con la finalidad de analizar la respuesta de la red.

Inicialmente la RN aprendió curvas con los parámetros de  $W = 1 \times 10^{-3}$  y  $L = 1 \times 10^{-5}$ , estos dos parámetros fueron modificados en cuatro diferentes variaciones en tres curvas aprendidas ( $V_T$  y  $R_L$  tienen valores aprendidos). En las siguientes Tablas (52-54) se muestran los valores usados para W y L, junto con los resultados generados por la red.

Tabla 52. Curva 1 modificada cuatro veces en W y L

<b><math>R_L</math> esperado= 70kΩ <math>V_T</math> esperado= 2.3V <math>V_{DD}</math>=2.3V</b>				
No. Curva	Parámetros geométricos		$R_L$ extraído	$V_T$ extraído
C1	W= 200 um	L=40 um	85.55 kΩ	5.21 V
C2	W= 100 um	L=10 um	79 kΩ	4.84 V
C3	W= 10 um	L=10 um	93.72 kΩ	5.81 V
C4	W= 40 um	L=40 um	87.73 kΩ	6.02 V

En las tablas se puede observar que el parámetro de resistencia  $R_L$  se ha extraído considerablemente bien a pesar de ser una curva totalmente nueva, ya que el valor deseado era de 70kΩ y la RN entregó desde 79 kΩ hasta 93.72 kΩ, sucede algo similar para las otras dos tablas. Aunque no sucedió lo mismo con el  $V_T$  ya que de 2.3V deseados el resultado más

cercano fue de 4.84V y el más lejano de 6.02V, lo mismo sucede en las tablas siguientes. El valor de  $V_T$  proporcionado por la red esta muy lejano del real, aunque esto no es totalmente negativo ya que la red está siendo alimentada con datos ajenos a lo que aprendió.

En las Figuras 72-74 se puede encontrar la razón por la cual la RN está dando un  $V_T$  mayor, como se vio en la sección 4.1.1. el voltaje umbral define en qué momento el transistor se activa, si es grande la curva se recorre a la derecha o viceversa. Cuando las curvas que alimentaron a la red son graficadas se puede observar que pareciera que el  $V_T$  aumento, ya que presentan un desplazamiento a la derecha en comparación con la curva aprendida (color azul a la izquierda). Entonces la RN está utilizando la experiencia adquirida durante el entrenamiento y al igual que una persona familiarizada con la extracción de parámetros deduce que se tiene un valor en  $V_T$  mayor, por el hecho de que las curvas están desplazadas a la derecha.

Tabla 53. Curva 2 modificada cuatro veces en W y L

<b>R<sub>L</sub> esperado= 150kΩ V<sub>T</sub> esperado = 1.8V V<sub>DD</sub>=3.5V</b>				
<b>No. Curva</b>	<b>Parámetros geométricos</b>		<b>R<sub>L</sub> extraído</b>	<b>V<sub>T</sub> extraído</b>
C1	W= 200 um	L=40 um	149.33 kΩ	4.93 V
C2	W= 100 um	L=10 um	145.32 kΩ	4.49 V
C3	W= 10 um	L=10 um	159.81 kΩ	5.41 V
C4	W= 40 um	L=40 um	160.76 kΩ	5.45 V

Tabla 54. Curva 3 modificada cuatro veces en W y L

<b>R<sub>L</sub> esperado= 320kΩ V<sub>T</sub> real = 3V V<sub>DD</sub>=5V</b>				
<b>No. Curva</b>	<b>Parámetros geométricos</b>		<b>R<sub>L</sub> extraído</b>	<b>V<sub>T</sub> extraído</b>
C1	W= 200 um	L=40 um	317.77 kΩ	5.5 V
C2	W= 100 um	L=10 um	322.29 kΩ	5.1 V
C3	W= 10 um	L=10 um	312.63 kΩ	5.86 V
C4	W= 40 um	L=40 um	310.34 kΩ	5.97 V

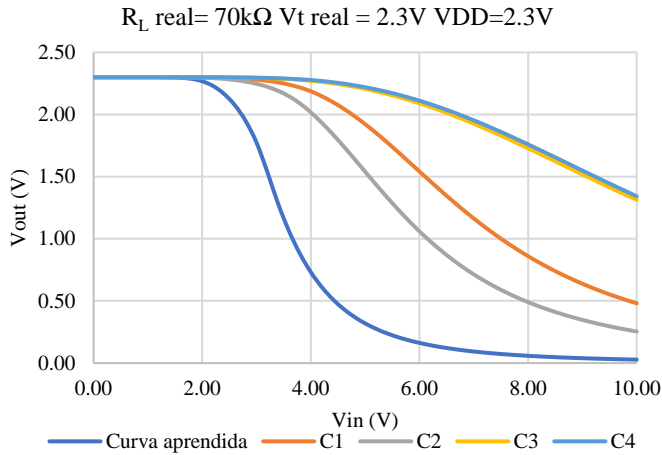


Figura 72. Curva 1 modificada cuatro veces en W y L

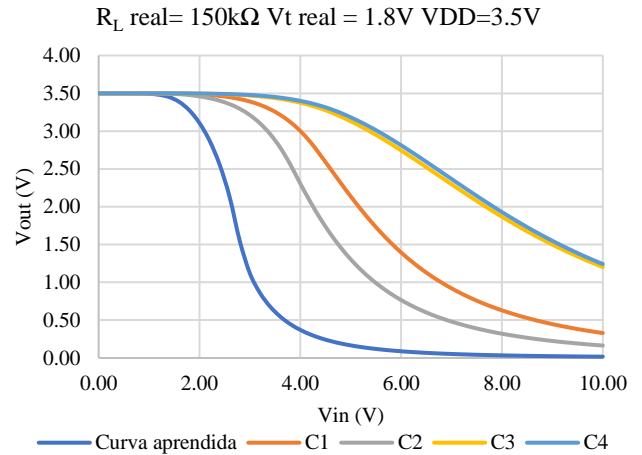


Figura 73. Curva 2 modificada cuatro veces en W y L

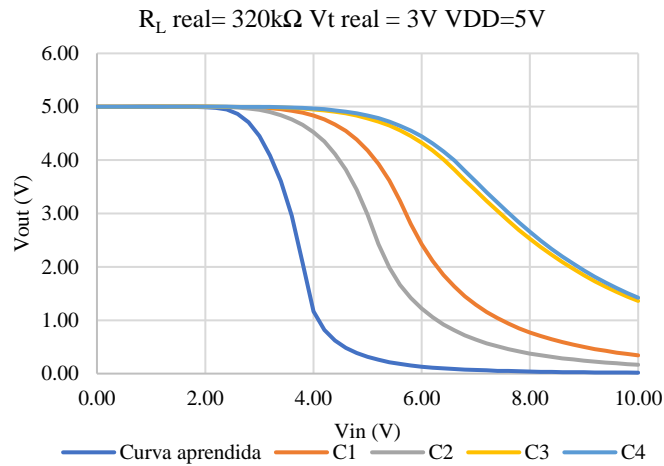


Figura 74. Curva 3 modificada cuatro veces en W y L

### 5.3.2 Resultados de extracción de 4 parámetros

En esta sección se presentan los resultados que obtuvieron los diferentes métodos de aprendizaje supervisado en la extracción de 4 parámetros ( $V_T$ ,  $R_L$ ,  $I_0$  y  $MMU$ ). Se presentan las características que tienen los mejores modelos, su evaluación y el comportamiento del circuito inversor con los parámetros extraídos y los reales.

#### *Mejores modelos de los métodos de aprendizaje*

El mejor modelo o arquitectura para la RN fue una red con 2 capas ocultas de 256 y 64 neuronas respectivamente, con función de activación *relu* mientras que la capa de salida de solo 4 neuronas tiene una función de activación *linear*, el mejor factor de aprendizaje fue de 0.001, con 2000 épocas de entrenamiento.

La estructura del RF con mejor desempeño presento un número de árboles o estimadores de 146, con un  $Max\_features=7$  sin profundidad máxima. En el caso de los AD, el mejor  $CCP\_ALPA=0$ . Dando como resultado un árbol con profundidad de 14, con 623 nodos terminales, con un  $min\_samples\_split=2$ .

En el caso de las SVR, se entrenó una SVR para cada parámetro a extraer, es decir, se tienen 4 SVR, algunas con hyper-parámetros diferentes, para la primer SVR perteneciente a  $V_T$  se tiene un Kernel polinomial, un  $C = 1.0$  y  $gamma = 0.1$ . Para la SVR de  $R_L$  un Kernel polinomial,  $C = 1.0$  y  $gamma = 0.3$ . En las SVR de  $IO$  y  $MMU$  se tienen los mismo hiper-parámetros, un Kernel polinomial,  $C = 5.0$  y  $gamma = 0.4$ .

A continuación, se presentan las cuatro Tablas (55-58), cada una pertenece a un método de aprendizaje, y tiene la evaluación de cada uno de ellos, dicha evaluación se realizó por parámetros y también se tiene un promedio general.

Tabla 55. Evaluación de resultados de RN

Parámetro	R <sup>2</sup>	MSE	%error
VT	0.9991	$2.47 \times 10^{-2}$	2.27
RL	0.999	$2.69 \times 10^{-2}$	2.86
IO	0.9596	$1.71 \times 10^{-1}$	16.45
MMU	0.9995	$1.71 \times 10^{-2}$	2.57
Promedio	0.98	$6.0039 \times 10^{-2}$	6.04

Como se observa, en la Tabla 55, se alcanza el promedio más alto en el  $R^2= 0.98$  perteneciente de las RN, esto se debe a que tiene un mayor equilibrio en los cuatro parámetros por encima del 0.95.

Tabla 56. Evaluación de resultados de RF

Parámetro	R <sup>2</sup>	MSE	%error
VT	0.9798	$1.2182 \times 10^{-1}$	8.90
RL	0.9688	$1.5909 \times 10^{-1}$	14.90
IO	0.8273	$3.5472 \times 10^{-1}$	62.96
MMU	0.9912	$7.7764 \times 10^{-2}$	11.10
Promedio	0.94	$1.766 \times 10^{-1}$	24.46

Por otro lado, en la Tabla 56 se observa que RF obtuvo un  $R^2= 0.94$ , colocándolo como segundo mejor método para la extracción de cuatro parámetros.



Tabla 57. Evaluación de resultados de AD

Parámetro	R <sup>2</sup>	MSE	%error
VT	0.9728	1.4142x10 <sup>-1</sup>	4.63
RL	0.9366	2.1689x10 <sup>-1</sup>	9.14
IO	0.5151	5.9435x10 <sup>-1</sup>	92.56
MMU	0.9578	1.7055x10 <sup>-1</sup>	12.55
Promedio	0.845575	2.808x10 <sup>-1</sup>	29.72

En la Tabla 57, se tiene promedio de R<sup>2</sup>= 0.94 para el AD, siendo el más bajo de todos los métodos, debido a que el desempeño en IO es de solo R<sup>2</sup>= 0.51 y este resultado afecta al momento de promediar. Sucede algo similar con SVR (Tabla 58), teniendo un R<sup>2</sup>= 0.74 en IO, aunque es mayor que el de AD, SVR tiene un R<sup>2</sup>= 0.86 en R<sub>L</sub>, aunque en promedio la SVR se mantiene por en encima con un R<sup>2</sup>= 0.88 general, siendo el tercer mejor método.

Tabla 58. Evaluación de resultados de SVR

Parámetro	R <sup>2</sup>	MSE	%error
VT	0.9698	1.4887x10 <sup>-1</sup>	12.49
RL	0.861	3.2137x10 <sup>-1</sup>	25.78
IO	0.745	4.31x10 <sup>-1</sup>	50.52
MMU	0.9578	1.7041x10 <sup>-1</sup>	26.06
Promedio	0.8834	2.6791x10 <sup>-1</sup>	28.71

La información de las cuatro tablas anteriores se puede observar de mejor manera en las siguientes dos figuras, donde se graficó el R<sup>2</sup> y el porcentaje de error (%error). En la Figura 75 (R<sup>2</sup>) se puede identificar que el parámetro IO es el que presenta mayor dificultad para extraer por parte de RF, AD y SVR, el parámetro R<sub>L</sub> fue difícil de extraer para las SVR. En la Figura 76, se presenta el porcentaje de error, en ella se puede observar que el parámetro IO presenta el mayor error, en especial para el AD, seguido de RF, SVR y por último NN. También se puede observar que, para los otros tres parámetros, el método con un mayor error es SVR, aunque se podría considerar como un erro bajo debido a que solo superó el 20% en R<sub>L</sub> y MMU.

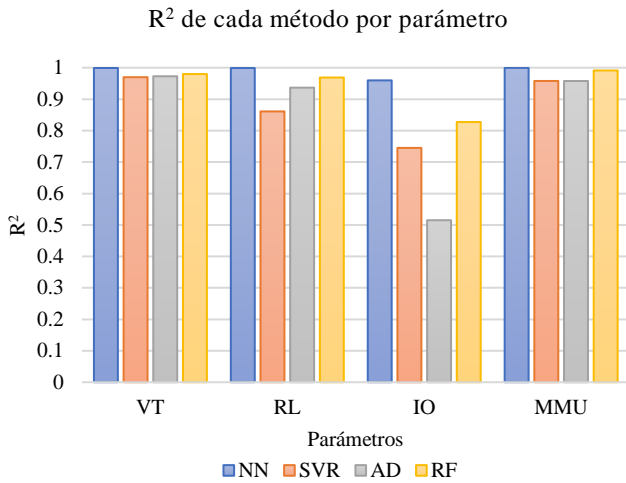


Figura 75. R2 de los métodos y por parámetros

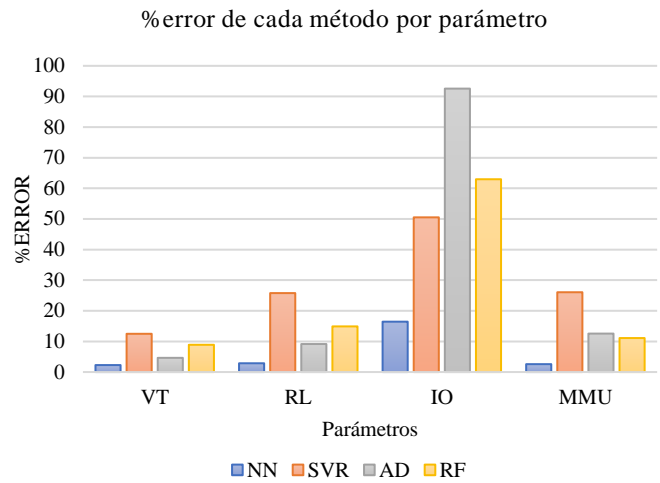


Figura 76. Porcentaje de error de los métodos

### Simulación de CI con parámetros extraídos

En esta sección se presenta la simulación del circuito inversor con los parámetros extraídos por los diferentes métodos de aprendizaje para comparar su comportamiento con el esperado. En la primera parte se muestran curvas del CI pertenecientes al conjunto de validación, la segunda parte muestran curvas del CI donde se tienen valores intermedios al conocimiento adquirido en el entrenamiento, con la finalidad de probar la capacidad de extrapolación de los diferentes métodos.

En las siguientes dos figuras se presentan dos curvas de validación del CI, las cuales están compuestas por tres gráficas, a) tienen el barrido completo de  $V_{in}$ ; con la finalidad de observar mejor las diferencias entre las diferentes curvas, en b) se enfocó la región de transición, donde afectan los parámetros  $V_T$ ,  $R_L$  y  $MMU$ , en c) se enfocó la región subumbral donde afecta el parámetro  $I_0$ .

En la Figura 77 a) las curvas ajustan de buena manera la original, algunas más que otras, en b) es posible ver que AD ajusta perfecto seguido de NN y en c) la que más se acerca es NN. En la Figura 78 a) sucede algo similar, todas las curvas ajustan a la original, pero en b) es posible ver que AD ajusta perfecto seguido de NN, en c), RF es la que más cerca esta del comportamiento esperado.

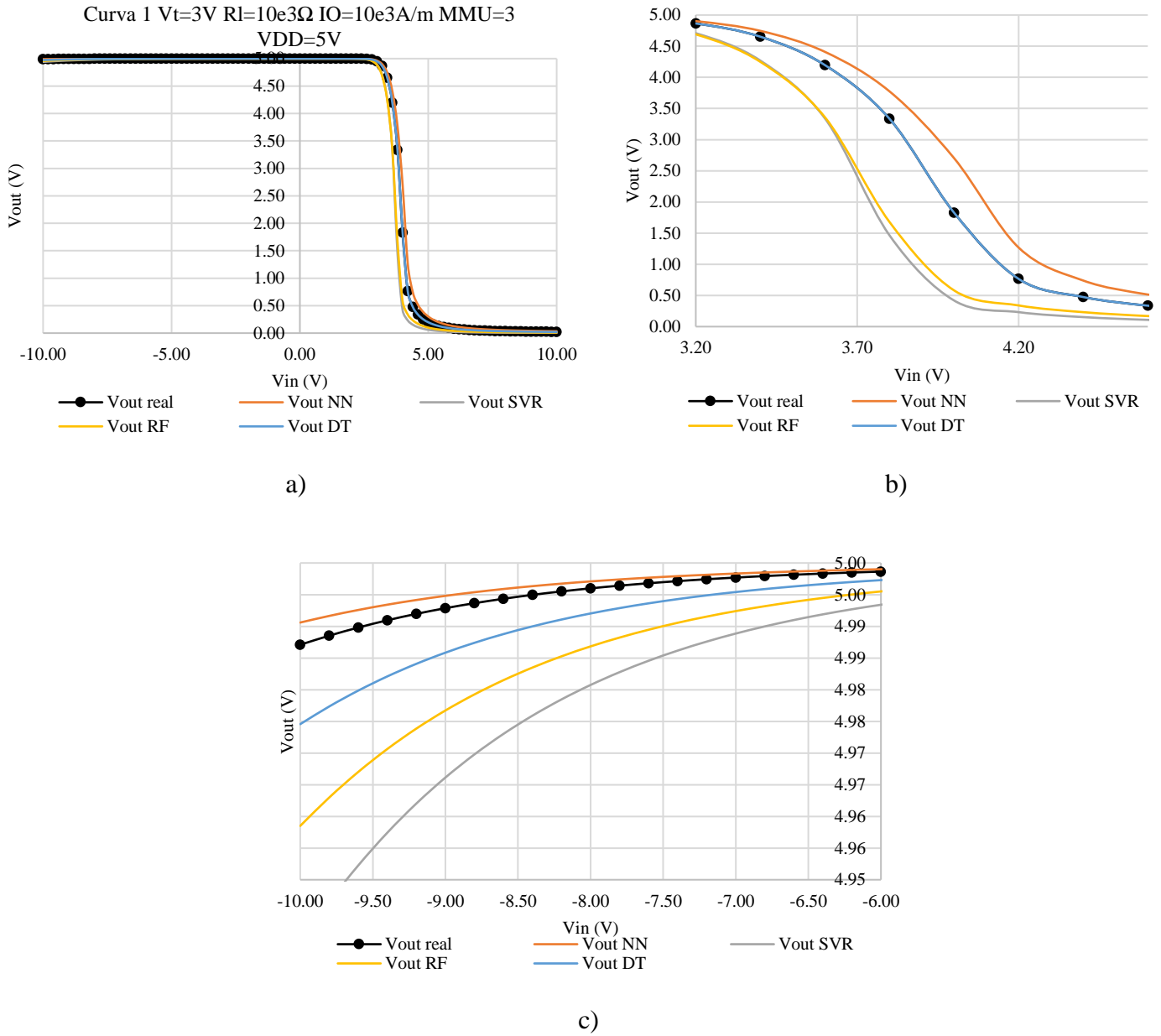
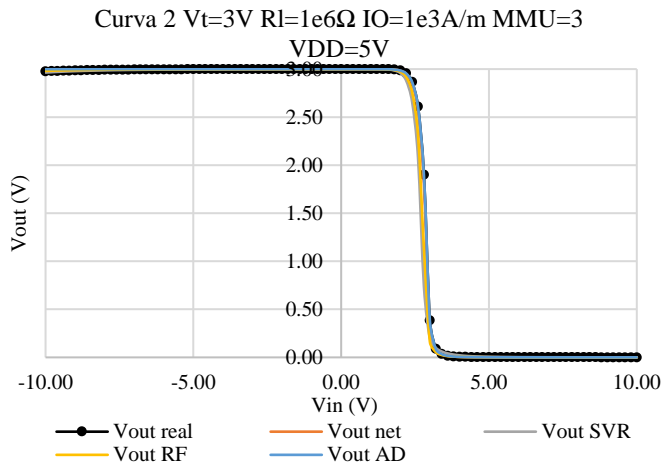


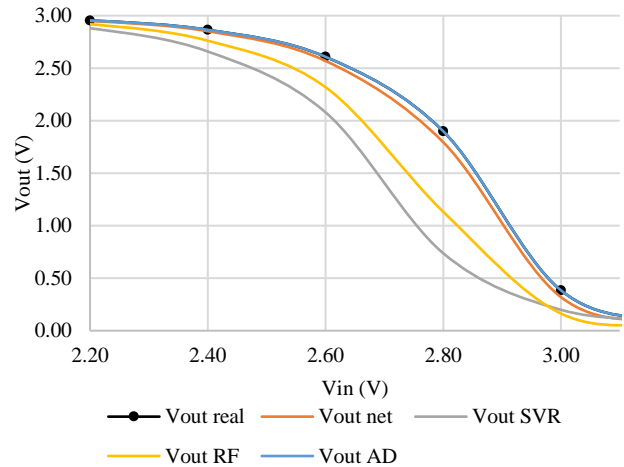
Figura 77. Curva 1 de validación de CI con parámetros extraídos

Las figuras 79 y 80 presentan las curvas donde los métodos fueron probados con valores intermedios en los parámetros a extraer, con la finalidad de probar cuál de ellos es mejor aproximando un resultado ante datos no aprendidos durante el entrenamiento.

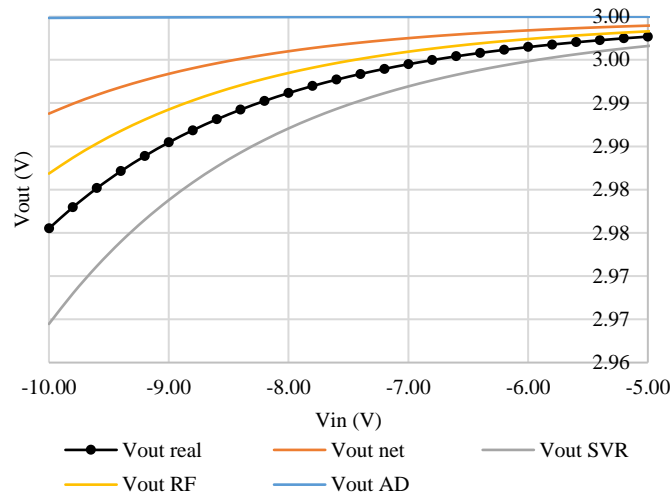
En la figura 79 a) se observa a simple vista que las diferentes curvas ajustan considerablemente bien a la original excepto AD, la cual se encuentra desplazada a la derecha, en b) esto es más evidente, además que se puede decir que RF es la que más se acerca a la esperada, seguida de NN y por último SVR en c) la que mejor ajusta en NN, seguida de RF, AD y por último SVR.



a)



b)



c)

Figura 78. Curva 2 de validación de CI con parámetros extraídos

En la figura 80 a) es posible observar que NN ajusta de manera perfecta toda la curva, los otros métodos se encuentran alejados del comportamiento esperado, en b) es posible notar que hay una pequeña región donde las NN no ajustan del todo aunque es mínimo y puede ser despreciable, en c) todos los métodos ajustan casi a la perfección, la distancia entre las curvas es casi nula, excepto para AD, la cual está encima de lo esperado.

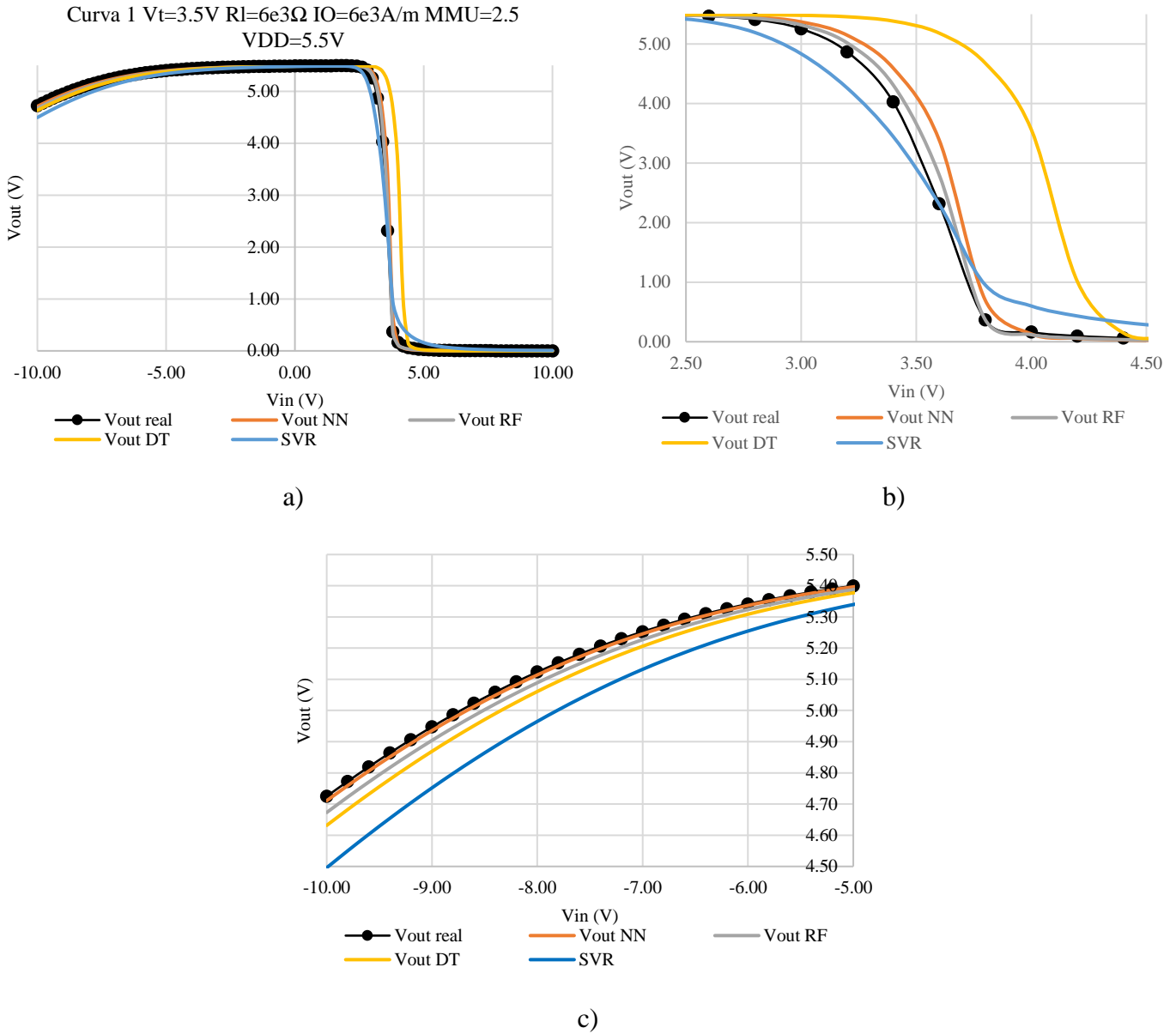
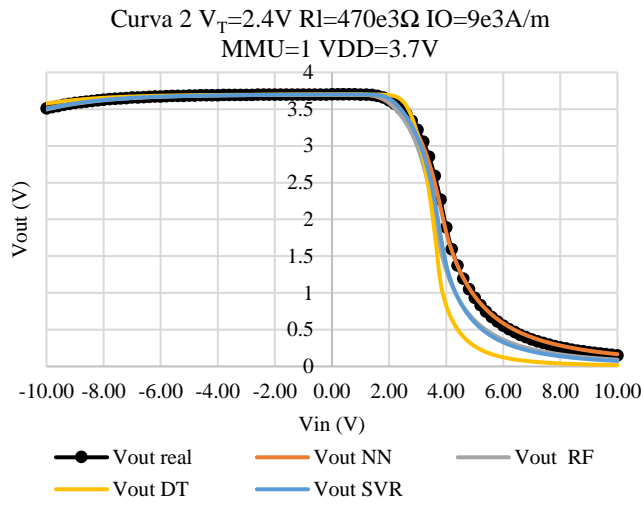
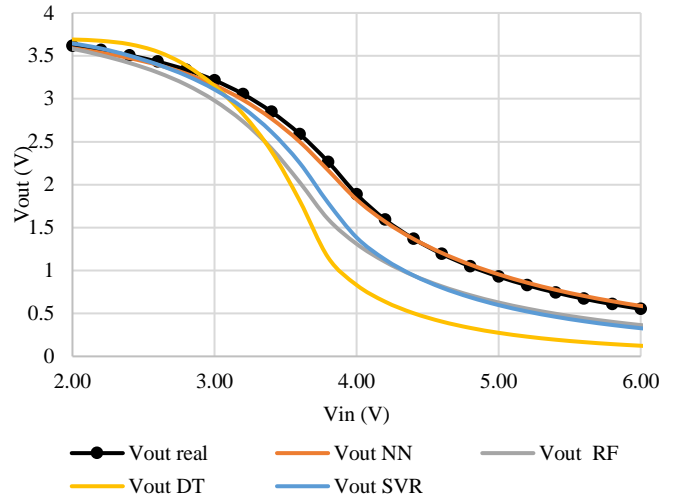


Figura 79. Curva 1 de prueba de CI con parámetros extraídos

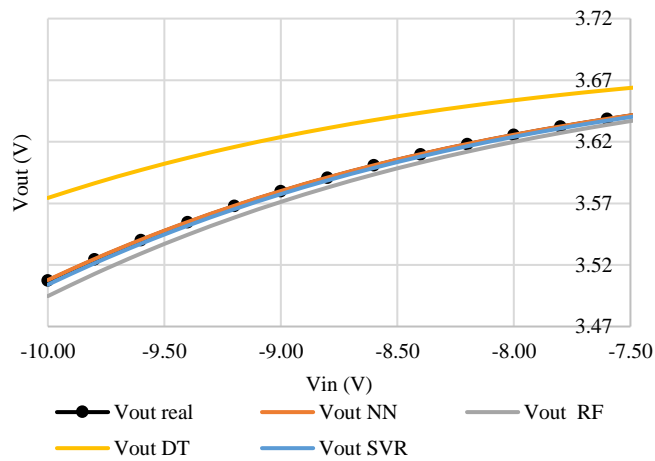
De manera general se puede afirmar que NN es el método que brinda mejores resultados cuando se alimenta con datos fuera del conocimiento adquirido, por otra parte, los resultados que generan RF y SVR también pueden ser aceptables. Los AD quedan en último lugar debido a su nula capacidad de dar un resultado aproximado con datos fuera de lo aprendido.



a)



b)



c)

Figura 80. Curva 2 de prueba de CI con parámetros extraídos

# Conclusiones

En este trabajo de investigación se presentó una propuesta para la extracción de parámetros en curvas I-V de diferentes dispositivos electrónicos. Se realizaron diferentes pruebas y experimentos, cuyos resultados demuestran que los métodos de aprendizaje supervisado son capaces de realizar la tarea de extracción de parámetros con una exactitud alta y son una alternativa sencilla que se puede aplicar en diferentes tecnologías para realizar la extracción. A continuación, se destacan y se discuten los resultados obtenidos en la investigación. Los resultados se dividen en etapas de experimentación que a su vez se pueden dividir en 2 o más partes.

En la primera etapa de experimentación se realizó la extracción de 3 parámetros en dispositivos NMOS, fabricados por el CIDESI. Se generó el conjunto de datos para el entrenamiento de los métodos de aprendizaje, para dos transistores de diferentes dimensiones (tabla). Antes de realizar el entrenamiento, el conjunto de datos pasó por una selección de característica, donde solamente se dejaron las entradas más significativas para predecir la salida deseada. En dicha selección se eliminaron las entradas correspondientes a  $V_{GS}$  y en la tabla (29) se presentan las entradas eliminadas correspondiente a la corriente eléctrica. Los resultados muestran que para T1 las RN tuvieron el mejor desempeño con un  $R^2= 0.99$ , seguido de RF con un  $R^2= 0.77$  y las SVR con un  $R^2= 0.74$ . Los parámetros extraídos de la medición experimental permitieron un buen ajuste, con un porcentaje de error de 0.63% por parte de las RN, RF presentó un 2.7% y las SVR con 0.16%. Para el dispositivo T2, las RN presentaron el mejor desempeño con un  $R^2= 0.97$ , seguidos de SVR con un  $R^2= 0.84$  y RF con un  $R^2= 0.75$ . Los parámetros extraídos de la medición experimental correspondiente a T2, permitieron un buen ajuste, con un porcentaje de error de 0.7% por parte de las RN, RF presentó un 5.7% y las SVR con 15.2%.

En la segunda etapa de experimentación se realizó la extracción de parámetros de TFTs IGZO con contactos de aluminio. El objetivo principal fue realizar la extracción de los parámetros  $K_P$ ,  $V_T$  y  $R_C$ . Se entrenaron tres métodos (RNs, RF y SVR), los cuales aprendieron de un conjunto de 840 muestras y después de su entrenamiento, se probaron para realizar la extracción de mediciones físicas de TFTs que fueron fabricados por el Centro de

Nanociencias y Micro y nanotecnologías del Instituto Politécnico Nacional de México. Se evalúa a las RNs con un  $R^2= 0.69$  con las muestras de validación, RF obtuvo un  $R^2=0.74$  y SVR también con un  $R^2=0.74$ . Esta evaluación puso a las RNs con el rendimiento más bajo, debido a que tuvieron una débil identificación de  $R_C$ . La prueba de extracción en mediciones físicas estableció a las RNs con los parámetros más exactos, permitiendo ajustes con porcentajes de error promedio de 5.10%, los RF con un error promedio del 11.78% y SVR con un error del 35.94%. Las pruebas con mediciones físicas indicaron que, aunque el aprendizaje de un método sea bajo (este caso las RNs), se pueden obtener parámetros extractos, todo dependerá de las condiciones de las nuevas mediciones.

Como tercera etapa se hizo la primera extracción en un circuito carga resistiva, del que se comenzaron a extraer dos parámetros ( $V_T$  y  $R_L$ ). Se utilizaron redes neuronales, implementadas en el lenguaje de Python en el entorno de *Jupyter* que brinda Google Colab, debido a que es un lenguaje de programación que cuenta con un gran número de librerías y herramientas destinados especialmente para la implementación de algoritmos de *Machine Learning* (aprendizaje maquina).

En este experimento se identificó como mejor modelo la red con dos capas ocultas de 256 y 32 neuronas respectivamente, con una exactitud del 95,37% de acuerdo con los resultados de Python, y un  $R^2= 0.9$  con un  $MSE= 1.129 \times 10^{-2}$ . Con el modelo de la red neuronal ya entrenada, se evaluó con un conjunto de curvas, que tenían valores de parámetros intermedios del conjunto utilizado para su aprendizaje. En esta evaluación se obtuvo un porcentaje de error del 3.27% en  $V_T$  y de 0.2% en  $R_L$  (escala logarítmica).

En una segunda parte de la tercera etapa, se realizó la extracción de cuatro parámetros del circuito inversor:  $V_T$ ,  $R_L$ ,  $I_0$  y  $MMU$ . En esta ocasión se utilizaron diferentes métodos de aprendizaje supervisado Random Forest, Árboles de Decisión y Support Vector Regression a parte de las Redes Neuronales. De los diferentes métodos el que presentó mejor desempeño fueron las RN con un  $R^2= 0.98$  y un porcentaje de error del 6.04%, en segundo lugar, se tienen a los RF con un  $R^2= 0.94$  y un porcentaje de error del 24.46%. Los AD y SVR tuvieron el desempeño más bajo debido a que la extracción del parámetro  $I_0$  la realizaron con poca exactitud, obteniendo un  $R^2= 0.84$  y un porcentaje de error del 29.72% por parte de los AD  $R^2= 0.88$  y un porcentaje de error del 28.71% por parte del SVR. Se probó a los métodos, alimentándolos con datos que tenían valores intermedios en los parámetros, para encontrar cuál de los métodos brinda mejores resultados, ante datos que no aprendieron durante el



entrenamiento. Se encontró que el mejor resultado lo brindan las RN, seguido de RF, SVR y al final AD, ya que estos últimos no pueden extrapolar ante nuevos datos.

Con los experimentos realizados y los resultados obtenidos se puede confirmar que en general los métodos de aprendizaje supervisado son capaces de identificar parámetros de mediciones I-V de dispositivos electrónicos, teniendo una buena concordancia entre las mediciones reales y las simuladas. De entre los diferentes métodos aplicados, las Redes neuronales obtuvieron los mejores resultados en la mayor de las veces.

## **Trabajo Futuro**

Como trabajo futuro se pretende obtener un modelo entrenado generalizado, es decir, un modelo robusto que sea capaz de realizar la extracción de parámetros de una o más tecnologías, además de eso, que sea capaz de extraer ante diferentes dimensiones geométricas de los dispositivos. También se tiene como objetivo implementar el modelo entrenado en un sistema web, de libre acceso para que lo investigadores en el área de la extracción puedan utilizarlo. Este sitio también permitiría compartir diferentes muestras I-V, para comenzar una colección de datos para seguir aumentando el conocimiento de los modelos y obtener mejores resultados.

## **Agradecimientos**

Se agradece al Consejo Nacional de Ciencia y Tecnología por la beca de estudios avanzados, con el número de apoyo 764880. También se agradece al Centro de Ingeniería y Desarrollo Industrial y al Centro de Nanociencias y Micho y Nanotecnologías por proveer mediciones físicas de dispositivos con las cuales fue posible probar la metodología propuesta en esta investigación.

## **Artículos publicados**

Con el desarrollo y resultados obtenidos de esta investigación se redactaron dos artículos, los cuales fueron aceptados y publicados por las revistas Journal of Material Science: Materials in Electronics e IEEE Latin America Transactions. A continuación, se presentan las dos referencias:

R. C. Valdés, F. García, R. Z. García, et al. “Parameter extraction in thin film transistors using artificial neural networks” *J Mater Sci: Mater Electron* 34, 555 (2023), Doi: <https://doi.org/10.1007/s10854-023-09953-z>.

Valdés García, R. C., García Lamont, F., García Lozano, R. Z., López Chau, A., Sánchez Fraga, R., & Lastra Medina, G. (2023). Parameter Extraction from a Resistive Load Inverter Circuit Using Supervised Learning Methods. *IEEE Latin America Transactions*, 21(5), 681–689. Retrieved from <https://latamt.ieee9.org/index.php/transactions/article/view/7483>

Estado del artículo: Publicado.

# Referencias

- [1] DigitalAVMagazine, “La demanda de pantallas de gran formato crecerá un 32% en 2018, según Sony y FutureSource.” <https://www.digitalavmagazine.com/2016/11/16/la-demanda-de-pantallas-de-gran-formato-crecera-un-32-en-2018-segun-sony-y-futuresource/> (accessed Nov. 13, 2019).
- [2] R. Dorantes, “Los 10 mejores celulares del mercado en 2018,” 2018. <https://www.altonivel.com.mx/tecnologia/mejores-celulares-smartphones-2018/> (accessed Nov. 13, 2019).
- [3] C. A. Torres, D. Murillo, and C. Restrepo, “Diseño y construcción de un inversor trifásico,” *Sci. Tech.*, vol. 40, pp. 37–42, 2008, [Online]. Available: <https://dialnet.unirioja.es/descarga/articulo/4733118.pdf>.
- [4] A. S. M., M. E. M., M. A. G., and E. C. B., “Retos Sobre el Modelado del Transistor de Compuerta Flotante de Múltiples Entradas en Circuitos Integrados,” *ReCIBE. Rev. electrónica Comput. Informática, Biomédica y Electrónica*, 2012, [Online]. Available: <https://www.redalyc.org/articulo.oa?id=512251559005>.
- [5] 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.
- [6] Y. Cao, T. Sato, M. Orshansky, D. Sylvester, and C. Hu, “New paradigm of predictive MOSFET and interconnect modeling for early circuit simulation,” in *Proceedings of the IEEE 2000 Custom Integrated Circuits Conference (Cat. No.00CH37044)*, pp. 201–204, doi: 10.1109/CICC.2000.852648.
- [7] N. Kaufmann and A. Konczykowska, “Interconnect layout macromodelling and simulation in high speed circuits,” in *2000 IEEE International Symposium on Circuits and Systems. Emerging Technologies for the 21st Century. Proceedings (IEEE Cat No.00CH36353)*, vol. 3, pp. 125–128, doi: 10.1109/ISCAS.2000.856012.
- [8] A. Álvarez, “Fundamentos físicos y tecnológicos de la informática,” *docplayer*, 2015. <https://docplayer.es/10832340-Tema-5-el-transistor-mos.html> (accessed Feb. 16, 2022).
- [9] M. Genero, J. Cruz-Lemus, and M. Piattini, *Métodos de investigación en ingeniería del software*. Madrid: Ra-Ma, 2015.
- [10] R. Pressman, *Ingeniería del software: Un enfoque práctico*, Séptima. Mc Graw Hill, 2010.
- [11] J. E. Molinar-Solis, V. H. Ponce-Ponce, R. Z. García-Lozano, A. Díaz-Sánchez, and J. M. Rocha-Pérez, “Electrical Parameters Extraction of CMOS Floating-Gate Inverters,” *Ing. Investig. y Tecnol.*, vol. 11, no. 3, pp. 315–323, Jul. 2010, doi: 10.22201/ft.25940732e.2010.11n3.027.
- [12] B. Yaglioglu, T. Agostinelli, P. Cain, S. Mijalkovic, and A. Nejim, “Parameter Extraction and Evaluation of UOTFT Model for Organic Thin-Film Transistor Circuit Design,” *J. Disp. Technol.*, vol. 9, no. 11, pp. 890–894, Nov. 2013, doi: 10.1109/JDT.2013.2260130.
- [13] 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.
- [14] 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.
- [15] I. Benacer and Z. Dibi, “Modeling and Simulation of Organic Field Effect Transistor (OFET) Using Artificial Neural Networks,” *Int. J. Adv. Sci. Technol.*, vol. 66, pp. 79–88, May 2014, doi: 10.14257/ijast.2014.66.07.
- [16] 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.
- [17] 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.


- [18] 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.
- [19] Q. Yao *et al.*, “Prediction of Static Characteristic Parameters of an Insulated Gate Bipolar Transistor Using Artificial Neural Network,” *Micromachines*, vol. 13, no. 1, p. 4, Dec. 2021, doi: 10.3390/mi13010004.
- [20] J. Oh, H. Song, E. Shin, H. Yang, J. Lim, and J.-H. Hwang, “Machine Learning–Assisted Thin-Film Transistor Characterization: A Case Study of Amorphous Indium Gallium Zinc Oxide (IGZO) Thin-Film Transistors,” *ECS J. Solid State Sci. Technol.*, vol. 11, no. 5, p. 055004, May 2022, doi: 10.1149/2162-8777/ac6894.
- [21] R. Boylestad and L. Nashelsky, *Electrónica: teoría de circuitos y dispositivos electrónicos*, Décima. México: Prentice Hall, 2009.
- [22] T. Ruiz, O. Arbelaitz, I. Etxeberria, and A. Ibarra, *Análisis básico de circuitos eléctricos y electrónicos*, Romo, Migu. España: Prentice Hall, 2004.
- [23] B. Cevallos, “MOSFET,” 2010. <https://es.slideshare.net/totycevallos/mosfet-13832167> (accessed Feb. 16, 2022).
- [24] A. Torres, S. Nelson, and C. Hernandez, “Redes neuronales y predicción de tráfico,” *Tecnura*, vol. 15, no. 29, pp. 90–97, 2011.
- [25] P. Ponce, *Inteligencia artificail con aplicaciones a la ingeniería*, First. México: Alfaomega, 2010.
- [26] P. Isasi and I. Galván, *Redes neuronales artificiales: un enfoque práctico*. Madrid: Prentice Hall, 2004.
- [27] M. Hagan, H. Demuth, M. Beale, and O. De Jesus, *Neural Network Design*, Second. Ebook, 2014.
- [28] X. Li, J. Cervantes, and W. Yu, “Fast classification for large data sets via random selection clustering and Support Vector Machines,” *Intell. Data Anal.*, vol. 16, no. 6, pp. 897–914, Nov. 2012, doi: 10.3233/IDA-2012-00558.
- [29] 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.
- [30] K. R. Chowdhary, *Fundamentals of artificial intelligence*. New Delhi: Springer India, 2020.
- [31] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, vol. 103. New York, NY: Springer New York, 2013.
- [32] M. Flasiński, *Introduction to Artificial Intelligence*. Cham: Springer International Publishing, 2016.
- [33] L. Zhang, Y. Pan, X. Wu, and M. J. Skibniewski, *Introduction to Artificial Intelligence*, vol. 163. London: Springer London, 2021.
- [34] A. C. Muller and S. Guido, *Introduction to Machine Learning with Python*. United States of America: O’Reilly Media, 2016.
- [35] 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.
- [36] J. Cervantes, F. García Lamont, A. López-Chau, L. Rodríguez Mazahua, and J. Sergio Ruíz, “Data selection based on decision tree for SVM classification on large data sets,” *Appl. Soft Comput.*, vol. 37, pp. 787–798, Dec. 2015, doi: 10.1016/j.asoc.2015.08.048.
- [37] “Propiedades de los semiconductores,” 2018. <http://www2.ual.es/te/icons/tabsem1.pdf> (accessed Mar. 24, 2022).
- [38] A. Devices, “LTspice.” 2022, [Online]. Available: <https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html>.
- [39] N. E. I. Karabadi, H. Seridi, I. Khelf, N. Azizi, and R. Boulkroune, “Improved decision tree construction based on attribute selection and data sampling for fault diagnosis in rotating machines,” *Eng. Appl. Artif. Intell.*, vol. 35, pp. 71–83, Oct. 2014, doi: 10.1016/j.engappai.2014.06.010.
- [40] L. Zhang, H. Wang, J. Wei, P. Hu, G. Hu, and D. Li, “Improvement in the electrical properties of a-IGZO TFTs by using PA-PI as a modification layer,” *Microelectronics J.*, vol. 127, no. 105516, 2022, doi: 10.1016/j.mejo.2022.105516.
- [41] J. Lee, S. Lee, D. Y. Jeong, and J. Jang, “Highly stable self-aligned coplanar a-IGZO TFTs under high temperature stress,” in *2018 9th International Conference on Computer Aided Design for Thin-Film Transistors (CAD-TFT)*, Nov. 2018, pp. 1–1, doi: 10.1109/CAD-TFT.2018.8608117.
- [42] Y. Li *et al.*, “A dynamic current hysteresis model for IGZO-TFT,” *Solid. State. Electron.*, vol. 197, p. 108459, Nov. 2022, doi: 10.1016/j.sse.2022.108459.
- [43] R. J. Baker, *CMOS: circuit design, layout, and simulation*, 3rd ed. Wiley-IEEE Press, 2010.

- [44] A. Pacheco-Sanchez, M. Claus, S. Mothes, and M. Schröter, “Contact resistance extraction methods for short- and long-channel carbon nanotube field-effect transistors,” *Solid. State. Electron.*, vol. 125, pp. 161–166, Nov. 2016, doi: 10.1016/j.sse.2016.07.011.
- [45] M. E. Rivas-Aguilar *et al.*, “Specific contact resistance of IGZO thin film transistors with metallic and transparent conductive oxides electrodes and XPS study of the contact/semiconductor interfaces,” *Curr. Appl. Phys.*, vol. 18, no. 7, pp. 834–842, Jul. 2018, doi: 10.1016/j.cap.2018.04.002.
- [46] S. Nautiyal and P. Mittal, “Contact resistance in organic transistors: Extraction using variable length method,” in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, May 2017, pp. 1558–1562, doi: 10.1109/CCAA.2017.8230051.
- [47] M. Abadi *et al.*, “{TensorFlow}: Large-Scale Machine Learning on Heterogeneous Systems,” 2015. <https://www.tensorflow.org/?hl=es-419> (accessed Sep. 28, 2021).
- [48] F. Pedragosa *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>.

*J Mater Sci: Mater Electron* (2023)34:555



## Parameter extraction in thin film transistors using artificial neural networks

Roberto C. Valdés<sup>1,\*</sup> , Farid García<sup>1</sup>, Rodolfo Z. García<sup>1</sup>, Asdrúbal López<sup>1</sup>, and Norberto Hernández<sup>2</sup>

<sup>1</sup>Autonomous University of the State of Mexico, Toluca, State of Mexico, Mexico

<sup>2</sup>Nanoscience, Micro and Nanotechnologies Centre of the IPN, Mexico City, Mexico

Received: 9 September 2022

Accepted: 20 January 2023

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

### ABSTRACT

This work presents a method based on supervised learning for the extraction of parameters in Indium Gallium Zinc Oxide Thin-Film Transistors with aluminium contacts, as an alternative regarding analytical and optimisation methods. The method consists of generating a set of  $I$ - $V$  curves of the device of interest using Spice software. These curves are the input samples of the Artificial Neural Networks, from which it is intended to predict the different parameters such as threshold voltage, transconductance and contact resistance, from each sample curve. By generating the training set itself, it is possible to label each sample curve, which allows the type of learning to be supervised. The results show that ANNs provide parameters with which it is possible to model physical measurements with error rates of less than 5% when extracting the first two parameters, and errors of between 0.06% and 4.62%, when extracting the three parameters. In addition, a comparison was made between the results of the ANNs and the analytical extraction of parameters.

### 1 Introduction

Nowadays, electronic simulation allows the design and testing of electronic devices or circuits before their manufacture, without the need to commit resources [1, 2]. Although it also allows the analysis and evaluation of devices already manufactured, which allows understanding their operation and making future optimisations. This is of great support to the growing emergence of new technologies and increasingly efficient devices [3].

Simulation software works by using mathematical models of the different devices supported, these models represent the behaviour of the devices in the real world. The models are made up of variables known as parameters, which can change from one device to another. The value assigned to them will define whether the simulation will correspond to the real-world behaviour.

For this reason, it is of utmost importance to know the parameters of the devices, for which a process known as parameter extraction is performed. Analytical methods can be found in the literature to

Address correspondence to E-mail: roberto.cvg@hotmail.com

<https://doi.org/10.1007/s10854-023-09953-z>

Published online: 16 February 2023



perform this task [4–15], and methods based on Genetic Algorithms (GA) [16–19] and Fuzzy Logic (FL) [20], have also been proposed as an alternative. These methods require a lot of experience and knowledge in the operation of the device models or defining the correct fitness function as in the case of GAs, to obtain good results. Not to mention emerging technologies, although more efficient devices, result in complex models with a large number of parameters, making parameter extraction a complex task. Recently, the application of machine learning in the process of parameter extraction has begun to be explored [21, 22].

Therefore, a method using ANNs is proposed to perform parameter extraction applied on TFT IGZO with Al contacts. The ANNs are trained to identify the parameters of a set of samples, consisting of  $I$ – $V$  curves of the device. This work shows that using ANNs are an alternative to provide parameters that allow a good fit between physical measurements and those simulated with the extracted parameters. The contribution of this study is a method of parameter extraction with which it is not necessary to have extensive experience in this task, nor to have a deep knowledge of the mathematical models of the devices. Although basic knowledge of the operation of NNs is required, their programming and implementation is not complicated due to the specialised Machine Learning libraries available today in different programming languages. On the other hand, using this method it is not necessary to process each  $I$ – $V$  curve individually as it is in the analytical methods. A trained NN can receive a single file with many curves and provide results almost instantaneously. In this paper not only a method for extraction is reported, but there are also two experiments, one in which the flexibility of NNs to compensate for the parameters to be extracted using their acquired experience is tested. The second one proposes a quick solution for when there are not enough samples to perform extraction in devices with different dimension.

Due to the early application of supervised learning for parameter extraction, the extraction of the most relevant transistor parameters such as threshold voltage, transconductance and contact resistance has been limited. The method was tested by extracting in different transistors manufactured by the Nanoscience, Micro and Nanotechnologies Centre of the National Polytechnic Institute, Mexico.

The rest of the paper is divided as follows: Sect. 2 presents a quick review of the state of the art; Sect. 3 introduces the TFT model and briefly presents the fundamentals of ANNs; Sect. 4 introduces the proposed method for parameter extraction; Sect. 5 presents the experiments carried out; Sect. 6 presents the results and discussion; and the paper closes with conclusions in Sect. 7.

## 2 Previous works

This section presents work related to the extraction of parameters in different types of transistors. Amongst the methodologies based on mathematical analysis, the following works stand out.

In [4], a method for parameter extraction in a short-channel amorphous InGaZnO TFT using experimental and simulated measurements was proposed. In [5], a comprehensive review of the most commonly used methods for calculating the threshold voltage in MOSFET devices was presented and includes work where the methods were applied to real devices. In [6] a parameter extraction for the AIM-Spice model of an amorphous TFT, avoiding non-linear optimisation, was proposed, the method was based on the integration of experimental measurements, and was tested in the linear and saturation regimes. A discussion on the use of technology based on OTFTs, OLETs, OLEDs and OPVs was presented in [7], concentrating on OLET devices, which have optical and electrical properties, the authors analysed the device in terms of drive current, threshold voltage, mobility and others. With the transfer curve analysis, they extracted the mentioned parameters. In [8], a method for parameter extraction of a polycrystalline silicon (poly-Si) TFT in weak conduction and triode region was proposed. They used two functions based on the integration of experimental measurements. In [9], an analysis of the behaviour of the OTFT, in the top (TC) and bottom (TB) contact, was made by modifying the shell drift model with respect to the series resistance. Using the mathematical model, the contact resistance, mobility and drain current, in linear and saturation regime, were extracted. In [10], a comparison of the performance of top and bottom contact OTFT structures was performed using two-dimensional numerical simulations. An estimation of the contact resistance using the conventional transmission line method and modified transmission line



method (M-TLM) was also performed. In [11], an accurate and broadband method for extracting parameters from a small-signal model in hetero-junction bipolar transistors (HBTs) was presented. An equivalent HBT circuit was used for the extraction of access resistance and parasitic conductance. In [12], the limited behaviour of the contact resistance in an organic pentacene transient is described by simulation and mathematical modelling. An analysis of a difference to the Shockley model, which is due to a non-linear behaviour of the contacts in organic devices, was performed. In [13], the specific contact resistance ( $p_c$ ) was determined between an amorphous indium-gallium-zinc-oxide (IGZO) semiconductor and different contact electrodes was found using TFT transistors. Chemical states of the contacts/semiconductor interfaces were used and analysed with X-ray photoelectron spectroscopy (XPS) to explain the differences in resistance. It was found that the lowest  $p_c$  was obtained using Ti/Au. In [14] three methods were used to extract the contact resistance of CNTFETs. The transfer length method (TLM) and two variants of the Y-function method were applied. It was found that the standard Y-function method does not give correct resistance values. In [15], a methodology was proposed to determine the asymmetrical source and drain resistances  $R_S$  and  $R_D$  from MoS<sub>2</sub> field-effect transistors (EM-FETs). By combining capacitance–voltage (C–V) and current–voltage characteristics, these resistances were extracted separately. First, the authors used C–V frequency dispersion from 0.3 to 10 kHz, then  $R_S$  and  $R_D$  were characterised by removing parasitic capacitances from the pad. The proposal was compared with the channel resistance method.

Also, works for parameter extraction based on GAs can be found. In [16] the authors proposed a hybrid algorithm for parameter extraction in OTFT transistors, based on the bee colony evolutionary algorithm (EA) to which they added a GA operator. The proposed algorithm extracted parameters from two devices and the results were compared with a simple GA. In [17] a machine learning approach to nonlinear regression with six input variables was used to measure the impact of process variability on the threshold voltage of a silicon-on-insulator (SOI) junctionless transistor (JLT). The GA was implemented in MATLAB to test the stated hypothesis. In [18], a GA was used to optimise the parameters of a

Carbon Nanotube Field Effect Transistor (CNFET), the performance of these devices depends on parameters such as the CNT diameter, the number of nanotubes and the spacing between the inter-tubes. The aim of this work was to minimise the Power-Delay Product (PDP). A compact analytical model for organic field effect transistors (OFETs) is presented in [19]. The proposed model describes the behaviour of the device in the above-threshold and below-threshold regime. This was achieved by calculating the total OFET current as the sum of both components where was added a transitive function to smooth the junction. A GA-based approach was also used as a tool for parameter extraction.

In [21] there is one of the few works where machine learning is used to extract parameters from IGZO TFTs, where 618 samples of I-V curves were used. In this work the authors start with an unsupervised learning approach using K-means to group the samples into 4 performance categories, and then use conventional NNs and Deep Neural Networks (DNN) to perform the parameter identification. By using physical measurements, they first performed analytical extraction, which allowed them to use the supervised learning approach. Their results show error rates of up to 10.6% in mobility extraction, up to 131.25% in subthreshold swing, 26.28% in threshold voltage and up to 71.36% in on/off current ratios using NNs. In [22] a work of extraction of extraction in Insulated Gate Bipolar Transistor (IGBT) is presented, where the extracted parameters were breakdown voltage ( $BV$ ), on-state voltage ( $V_{on}$ ), static latch-up voltage ( $V_{lu}$ ), static latch-up current density ( $J_{lu}$ ) and threshold voltage ( $V_T$ ). The authors propose a two-layer NN to predict the above parameters using as inputs device structural parameters (not I–V curves) such as N-drift doping, N-buffer doping, P-well doping, P + well doping, N-drift thickness, N-buffer thickness and channel length. Their results showed average error rates of up to 7.7%. In both works the extraction of  $R_C$ , which is a complicated parameter to extract due to the need for physical measurements of devices with different dimensions, was not performed.

### 3 Theoretical background

#### 3.1 TFT model

TFTs (Thin Film Transistors) are MOSFET (Metal Oxide Semiconductor Field Effect Transistor) devices, and are used in today's displays, such as those in mobile phones and smart TVs, whose development is progressing rapidly. The mathematical model describing the current behaviour of a TFT is given by Eq. (1) (model implemented in AIM Spice [23]).

Current  $I_D$  above the threshold voltage occurs when  $V_{GT} > 0$ .

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

where  $\mu_{FET}$  is the effective mobility of the device,  $C_{ox}$  is the capacitance of the oxide layer, given by the dielectric permittivity constant ( $\epsilon_i$ ) divided by the thickness of the oxide layer ( $T_{ox}$ ),  $W$  and  $L$  are the width and length of the channel respectively;  $V_{GT}$  is the result of the applied gate and source voltage, minus the threshold voltage ( $V_T$ );  $V_{DS}$  is the applied voltage between drain and source terminals. Finally,  $\alpha_{sat}$  is the proportionality constant of  $V_{sat}$  (saturation voltage).

The current  $I_D$  below the threshold voltage is given by Eq. (2).

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

where MUS stands for electric mobility;  $V_{sth}$  is the product of ETA (subthreshold ideality factor) and  $V_{th}$  (Eq. 3).

$$V_{th} = k_B \times \text{TEMP}/q, \quad (3)$$

where  $k_B$  is the Boltzmann's constant, TEMP is the temperature and  $q$  is the electron charge magnitude. A part of the model can be summarised by the parameter of transconductance ( $KP$ ), this parameter is widely used by circuit and device designers, which is given by Eq. 4, where  $\epsilon_0$  is the vacuum permittivity.

$$KP = \mu_{FET} C_{ox} = \mu_{FET} \frac{\epsilon_i \epsilon_0}{T_{ox}}. \quad (4)$$

$KP$  is one of the parameters to be extracted in this research, and with which it is possible to calculate the mobility which is a parameter used by people who analyse the behaviour of devices [24]. When the induced channel extends from the source to the drain, the transconductance can be rewritten as:

$$\beta = KP \frac{W}{L}. \quad (5)$$

Considering that the material of the device presents resistance, the mobility and electric current will be inversely proportional to the resistance. And the total resistance  $R_T$  of a device is given by Eq. (6).

$$R_T = R_C + R_{CH} = V_{DS}/I_D, \quad (6)$$

$$R_C = R_S + R_D, \quad (7)$$

where  $R_C$  is the contact resistance,  $R_{CH}$  is the channel resistance,  $R_S$  and  $R_D$  are the resistances at the source and drain terminals respectively [12, 14].

#### 3.2 Artificial neural networks

ANNs or just NNs are a simplified approximation of the brain, represented by an ensemble of artificial neurons. The artificial neuron concept was proposed by Warren S. McCulloch and Walter Pitts in 1943 [25].

NNs have been widely used for pattern recognition, big data analysis, feature extraction, classification, regression, system identification, amongst other applications [26]. NNs are inspired on the biological functioning of how living things learn from experience. NNs acquire knowledge from a set of representative examples or samples in a process known as training. There are three basic ways in which the network learns: supervised, unsupervised and reinforcement learning.

In supervised learning, the set of samples is pre-labelled, i.e. for each of the examples, the correct answer is known, usually used for classification and regression problems. In unsupervised learning, the answer corresponding to each training sample is not known, the NN adopts the underlying structure of the training set; usually, the unsupervised NNs are employed for clustering problems. In reinforcement learning, the goal is to obtain intelligent agents capable of taking the best action in given situations or environments, with the aim of obtaining the

maximum reward or the least punishment. Reinforcement learning is used in control, autonomous systems and game theory [26].

Formally, the output of a neuron is given by Eq. (8), where the product of an input  $p$  and a synaptic weight  $w$ , plus a bias  $b$ , is evaluated in an activation function  $f$ .

$$a = f(wp + b). \quad (8)$$

The activation function  $f$  is responsible for setting a range of values over which the network will be working, and the function is selected by the nature of the problem.

A neuron can have multiple inputs ( $p_R$ ), and the output of the neuron is given by Eq. (9), where each input is multiplied by a corresponding synaptic weight. The output of the multi-input neuron can be written in vector form (Eq. 10), where  $W$  is the vector of weights and  $p$  is the vector of inputs.

$$a = f(w_{1,1}p_1 + w_{1,2}p_2 + w_{1,3}p_3 + \dots + w_{1,R}p_R + b), \quad (9)$$

$$a = f(Wp + b). \quad (10)$$

In a layer of neurons  $s$ , each neuron has an output  $a_s$ . Each individual input  $p_1 \dots p_R$  of the input vector  $p$  is connected to each neuron and is multiplied by the corresponding weight of the matrix  $W$ . Each neuron has its bias  $b_i$  and its activation function  $f$ . So, the outputs of the neurons are given by Eqs. (11) and (13).

$$a_1 = f_1(w_{1,1}p_1 + w_{1,2}p_2 + w_{1,3}p_3 + \dots + w_{s,R}p_R + b_1), \quad (11)$$

$$\begin{aligned} a_2 &= f_2(w_{2,1}p_1 + w_{2,2}p_2 + w_{2,3}p_3 + \dots + w_{s,R}p_R + b_2) \\ &\vdots \end{aligned} \quad (12)$$

$$a_s = f_s(w_{s,1}p_1 + w_{s,2}p_2 + w_{s,3}p_3 + \dots + w_{s,R}p_R + b_s). \quad (13)$$

In vector form the output of a layer of neurons is given by Eq. (14). Where,  $b$  is the bias vector.

$$a = f(Wp + b). \quad (14)$$

In a network with a number  $n$  of layers, the output generated by each neuron in the first layer is the input to the second layer and so on. So, each layer has its vector of inputs  $p$ , its matrix of weights  $w$ , a bias

vector and its activation function. The output of 3 layers and the total network output is given by the following equations. The output of network layer  $n$  is given by Eq. (19).

$$a^1 = f^1(W^1p + b^1), \quad (15)$$

$$a^2 = f^2(W^2a^1 + b^2), \quad (16)$$

$$a^3 = f^3(W^3a^2 + b^3), \quad (17)$$

$$a^3 = f^3(W^3f^2(W^2f^1(W^1p + b^1) + b^2) + b^3), \quad (18)$$

$$a^n = f^n(W^n a^{n-1} + b^n). \quad (19)$$

As mentioned, learning is obtained in the training stage, where the error between the expected output and the obtained output is obtained in an iterative way, known as epochs. In supervised learning, the backpropagation algorithm is the most widely used. This gradient-based algorithm aims to reduce the mean square error MSE (Eq. 20). In training, at each epoch, the synaptic weights  $w$  and the bias are modified, obtaining the minimum error or up to the defined number of epochs.

$$F(x) = E[e^2] = E[(t - a)^2], \quad (20)$$

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha \frac{\partial F}{\partial w_{i,j}^m}, \quad (21)$$

$$b_i^m(k+1) = b_i^m(k) - \alpha \frac{\partial F}{\partial b_i^m}, \quad (22)$$

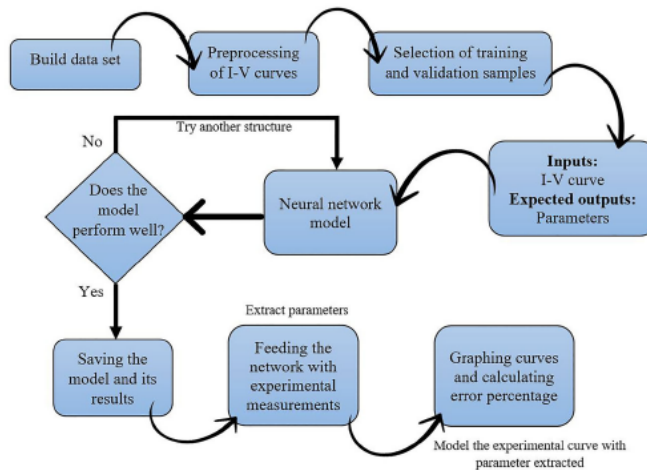
where  $t$  is the target, or desired output, and  $a$ , is the output obtained,  $k$  is the epoch number,  $m$  is the layer number,  $i$  and  $j$  are the identifier of the weights and bias,  $\alpha$  is the learning factor, which determines the change in  $w$  and  $b$  [27, 28].

#### 4 Parameter extraction method

Figure 1 summarises the methodology for parameter extraction using NNs. The extraction steps are summarised as follows:

- (1) A training dataset is constructed using I-V curves obtained by LTspice [29] simulation.
- (2) Curve data such as  $V_{GS}$  and  $I_D$  is normalised or rescaled with Eq. (24).
- (3) The total samples ( $I$ - $V$  curves) are divided into two sets, one for training and one for validation.

**Fig. 1** General diagram of the methodology for the extraction of parameters



- (4) For each simulated curve (input) its parameters (outputs or targets) are known.
- (5) With the data sets ready, the training of NNs with different numbers of layers and neurons is started.
- (6) If the determination coefficient of the trained NN model ( $R^2$ ) is greater than 90%, then, the model and its results are saved.
- (7) Once the NN has been trained, it is used to extract the parameters of physical measurements.

Finally, the extracted parameters are employed to simulate the device, and compare with the physical measurements and calculate the percentage error between the curves.

#### 4.1 Training dataset and preprocessing

The dataset of samples for training the NNs consists of a set of  $I$ - $V$  curves of the transistor. These curves are obtained by simulation, using a given model (defined in the simulator). In this case, level 1 of the SPICE Model for MOSFETs is used, although it is a basic model, it was used because it allowed modelling the behaviour of the TFTs. To build the training dataset, the first step is to define the parameters of interest, i.e. the parameters that will be extracted. In this case the parameters of interest are  $KP$ ,  $V_T$  and  $R_C$ .

The reason of this selection is that these electrical parameters are frequently used to characterise and model the TFTs electrical behaviour. The simulations will be performed by varying the values of the parameters of interest. The range of variation and the number of simulations that make up the training set will define how knowledgeable or robust the NN model will be. In the present work, an initial analytical parameter extraction was performed on one of the physical transistors, and these results were used as the basis for defining the range of values with which the simulation would be performed. It is important to note that this range of values must be sufficiently wide for the NN to provide adequate results (with low error percentages). Depending on the results obtained from the extraction using the trained NNs, the training sets can be complemented with more samples to improve the performance of the NNs.

The process for carrying out the simulations is as follows. To extract  $KP$ ,  $V_T$  and  $R_C$ , starting to set in the simulator (optional) in this case LTspice,  $KP$  and  $V_T$  are set to their initial value  $KP = 1 \times 10^{-6} \text{ A/V}^2$  and  $V_T = 1.5 \text{ V}$ , keeping  $KP$  and  $V_T$  as constants,  $R_C$  is swept from 0 to  $6 \times 10^3 \Omega$  with increments of  $1 \times 10^3 \Omega$ . Then the value of  $V_T$  increases,  $KP$  remains constant,  $V_T = 1.8 \text{ V}$  and again  $R_C$  is swept. When  $R_C$  was swept for each value of  $V_T$  (up to 3.6 V), now  $KP$

is increased and  $V_T$  starts with the initial value,  $KP = 1.4 \times 10^{-6} \text{ A/V}^2$ ,  $V_T = 1.5 \text{ V}$  and so on until  $KP$  reaches its maximum value. This is the most time-consuming step, if done by one person. In each run, the  $I_D$  data are stored. The approximate time is 120 samples of  $I-V$  curves in 1 h. The process in which the parameters are swept could be done by programming and using the mathematical models of the device Eqs. (1) to (7), although the possibility of human error in programming is not ruled out, so it was decided to use simulation software which uses the same mathematical models and ensure that the  $I-V$  samples are accurate. For the experimentation the following data sets were created. The first set (289 samples) to extract the parameter  $KP$  and  $V_T$  is given by Eqs. (23) and (24). A second set was made to extract  $KP$ ,  $V_T$  and  $R_C$  (448 samples), which is given by Eqs. (25), (26) and (27). The third set (392 samples) is given by Eqs. (28), (29) and (30). And finally, the fourth set is made up of the union of the second and third set with a total of 840 samples.

$$K = \{1 \times 10^{-6}/V^2 + k\Delta | k = 0, 1, \dots, 16\}, \tag{23}$$

where  $\Delta = 0.2 \text{ A/V}^2$ .

$$V = \{1.0V + k\Delta | k = 0, 1, \dots, 16\}, \tag{24}$$

where  $\Delta = 0.2 \text{ V}$ .

$$K = \{1 \times 10^{-6} \text{ A/V}^2 + k\Delta | k = 0, 1, \dots, 7\}, \tag{25}$$

where  $\Delta = 0.4 \text{ A/V}^2$ .

$$V = \{1.0V + k\Delta | k = 0, 1, \dots, 7\}, \tag{26}$$

where  $\Delta = 0.3 \text{ V}$ .

$$R = \{0\Omega + k\Delta | k = 0, 1, \dots, 6\}, \tag{27}$$

where  $\Delta = 1 \times 10^3 \Omega$ .

$$K = \{1.2 \times 10^{-6} \text{ A/V}^2 + k\Delta | k = 0, 1, \dots, 6\}, \tag{28}$$

where  $\Delta = 0.4 \text{ A/V}^2$ .

$$V = \{1.0V + k\Delta | k = 0, 1, \dots, 7\}, \tag{29}$$

where  $\Delta = 0.3 \text{ V}$ .

$$R = \{0\Omega + k\Delta | k = 0, 1, \dots, 6\}, \tag{30}$$

where  $\Delta = 1 \times 10^3 \Omega$ .

The voltages used were  $V_{GS}$  from  $-6$  to  $6 \text{ V}$  and  $V_{DS} = 6 \text{ V}$ , because the physical measurements were taken at these voltages.

The pre-processing step consists of rescaling the data, usually in the ranges  $[-1, 1]$  or  $[0, 1]$ , depending on the nature of the data. This step allows each sample to be transformed in a way that gives more information to the NN to facilitate pattern detection. In this case, each input was divided between the highest value it could have, thus obtaining the range  $[0, 1]$ .

### 4.2 NNs training

When solving a classification or regression problem using NNs it is necessary to train more than one network model (number of layers and neurons), as there is no way of knowing which NN will be the most suitable for each problem. Although it has been observed that for regression problems, the higher the number of layers and neurons, the higher the learning rate, this is not always true [27]. So, networks with different numbers of layers, neurons, activation functions and learning factors must be tested until the models with the highest learning from the data are found.

The best way to find the correct values of the hyperparameters (characteristics) of the network is to use the grid search [30]. This technique consists of sweeping through the different hyperparameters from a pre-determined initial value to a final value, combining all of them to identify which one provides the best performance. As mentioned above, in order to find the best learning NN model, approximately 24 NNs were trained, of which 8 of them performed the best. Table 1 shows the top 8 models that were trained to perform TFT parameter extraction. The function in the hidden layers is *Relu* and in the output layer *Linear*. Each model was trained with 2000 epochs. NN models with a smaller number of layers

**Table 1** Top 8 trained NNs models

NN model	No. layers	No. of neurons in each layer
1	2	128/64
2	2	256/32
3	2	256/64
4	2	256/128
5	2	256/256
6	3	128/64/32
7	4	256/128/64/32
8	5	256/128/64/32/16

and neurons presented underfitting (lack of learning) and with larger NNs, learning does not increase (a complex NN will not always be the best), and in computing when two NN models have the same performance, it is better to select the smaller one.

The training of the different NNs models was implemented using Google Colab [31], which is a Jupyter-based environment for programming in the Python language, and it is possible to make use of Google computing resources. The library used was Keras from Tensorflow [32] which is intended for training different types of NNs.

### 4.3 NN evaluation

Metrics for evaluating the results of parameter prediction (extraction) are presented in this subsection. In addition to the MSE and  $R^2$  in the parameters in each of the NN models. In the extraction of parameters from the physical measurements, the percentage error was calculated using the area under the curve (trapezoid method) in Eq. (31) between the experimental curve and those modelled with the extracted parameters.

$$\% \text{ error} = \left| \frac{\text{area}_{\text{ext}} - \text{area}_{\text{real}}}{\text{area}_{\text{real}}} \right| \times 100. \quad (31)$$

In computing, the selection of the best NN model is done by simply taking the model with the highest accuracy in case of classification or  $R^2$  in regression, considering only the results on the validation samples. In this work, the best model would be the one that obtained the highest  $R^2$  in the validation samples, for example, model 8 in Table 3. But it is possible to use physical measurements as a test set to perform the parameter extraction, using not only the model 8 that was the best, in this work we tested the best 8 out of 24 models. In this way, and for this problem, it is recommended to have more than one NN model trained, and the best one will be the one that allows to obtain the lowest error percentage.

## 5 Experiments

This section describes the different experiments and analyses that were performed during the training of the NNs. This research was developed incrementally, starting with the extraction of two parameters ( $KP$  and  $V_T$ ), once it was verified that the NNs extracted

parameters from physical measurements with good accuracy, extraction of  $KP$ ,  $V_T$  and  $R_C$  was carried out. For the extraction tests once the training of the NNs models was completed, physical measurements of IGZO TFTs were used, which were made with coming glass and at room temperature (25 °C), it should be mentioned that the proportionality saturation voltage is not required by the Spice model used. The geometrical parameters are presented in Table 2. Because there is more than one TFT transistor, and also with different dimensions (Table 2), from now on, each physical measurement will be identified with a number, e.g. T1M1 refers to transistor 1 with dimensions  $W = 80 \mu\text{m}$  and  $L = 80 \mu\text{m}$ , M1 refers to the first of all measurements with the same dimensions. Also, can be found "lin" to refer to measurement in the linear regime, if not present the physical measurement is in the saturation regime.

### 5.1 Experiment 1

In the first experiment, the first data set was used to train NNs to learn to extract the  $KP$  and  $V_T$  parameters in both the saturation and linear regimes. The training data are intended for a TFT with a  $W = 80 \mu\text{m}$ ,  $L = 80 \mu\text{m}$  and a  $T_{\text{ox}} = 15 \text{ nm}$  (T1). After the NN models are trained and stored, they are tested to perform the extraction of physical measurements. The values provided by the NN must be treated in the inverse way in which they were pre-processed to obtain the real value of the parameter to be used in the electronic simulator.

The physical measurements used for the tests in this experiment correspond to T1 with two measurement conditions ( $V_{\text{GS}}$  application). It was observed that the behaviour changed slightly if  $V_{\text{GS}}$  was applied from negative to positive (NP) and vice

**Table 2** Dimensions of the transistors from experimental measurements

Transistor	W ( $\mu\text{m}$ )	L ( $\mu\text{m}$ )	$T_{\text{ox}}$ (nm)	No. of measurements
T1	80	10	15	3
T2	40	10	15	2
T3	80	5	15	2
T4	40	40	15	1
T5	80	20	15	1
T6	80	80	15	1
T7	40	5	15	1

versa (PN). If no NP or PN is specified, the measurement was taken with  $V_{GS}$  sweep from negative to positive.

Figure 2 shows how the device (T1) changes its transfer curve when measured from negative to positive and from positive to negative. Not all transistors were measured in both ways, though. If not specified with a PN, it means that the physical measurement was obtained with  $V_{GS}$  from negative to positive. The analysis of the hysteresis that occurred in the fabricated transistors will be reported in a separate article.

## 5.2 Experiment 2

In this experiment, no training of NNs is performed, here was used one of the NN models that was trained in experiment 1. During experiment 1, the NNs learned from samples that did not have contact resistance. The aim of this second experiment is to analyse the results of the NN already trained when it is fed with a sample to which  $R_C$  was added.

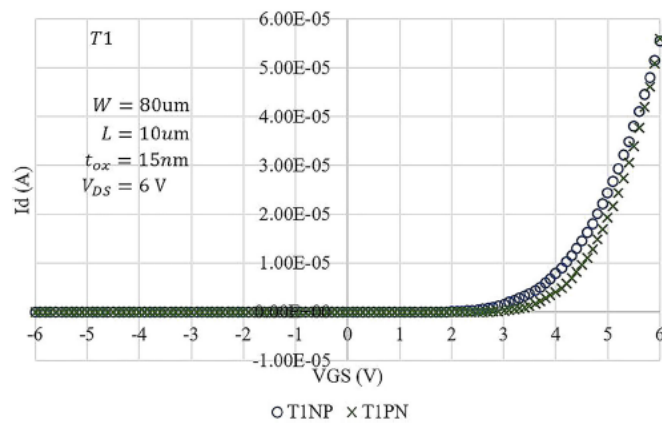
To achieve this, the parameters extracted from  $KP$  and  $V_T$  in experiment 1 were used, which allowed the behaviour of T1 to be modelled accurately. Using again the LTspice simulator, the parameters  $KP$  and  $V_T$  were set as constants, and 6 runs were performed, where in each of them the contact resistance had the following values:  $0 \Omega$ ,  $1 \times 10^3 \Omega$ ,  $4 \times 10^3 \Omega$ ,  $16 \times 10^3 \Omega$ ,  $32 \times 10^3 \Omega$  and  $64 \times 10^3 \Omega$ .

The 6 simulated samples with  $R_C$  increasing from 0 to  $64 \times 10^3 \Omega$  were stored and pre-processed to feed the NN model 1 and extract its  $KP$  and  $V_T$  parameters. This experiment allows us to demonstrate how NNs can make use of the knowledge acquired during training and approximate results for input samples that have information that was not necessarily learned.

## 5.3 Experiment 3

The main disadvantage of NNs and other machine learning methods is the need for sufficient samples to learn, plus the fact that each dataset is problem-specific. So, if an NN was trained to extract parameters with certain dimensions such as  $W = 80 \mu\text{m}$  and  $L = 10 \mu\text{m}$ , as in experiment 1, the NN would not be able to extract parameters from a transistor with different dimensions and it would be necessary to generate the training set for that other device. The process of making a data set for each of the transistors available in this research (Table 2) would become a tedious task. Furthermore, for some combination of  $W$  and  $L$  the  $I_D$  current would be the same as in the case of T4 and T6. In a future extractor, or even in a commercial extractor, generalised and specialised system for extraction for an established technology, where the geometrical factors are always the same, including the parameters  $W$ ,  $L$  and  $T_{ox}$  during NN's training would be worthwhile and absolutely necessary.

**Fig. 2** T1 transferential curve with different measurement condition



In the third experiment, the geometrical parameters of the samples that make up the first dataset were set to be removed. This is achieved by normalising the  $I_D$  current of each of the samples. This normalisation was performed using Eq. (32), where  $W = 80$   $\mu\text{m}$  and  $L = 10$   $\mu\text{m}$ . The NN models (first three in Table 1) were then re-trained with the normalised sample set. After training, the NNs are ready to perform the extraction tests on the physical measurements, in order to feed the NNs, the measurements must also be normalised using their own channel widths and lengths.

In this experiment, a way was proposed to avoid the need for examples for each problem to be solved by the NNs.

$$I_{\text{norm}} = \frac{L}{W}(I_D). \quad (32)$$

#### 5.4 Experiment 4

In the fourth experiment, training of NN models was carried out for dataset number 2 (Eq. 25, 26, 27), training was carried out for dataset number 3 (Eqs. 28, 29, 30) and training was also carried out by joining these datasets. This fourth experiment aimed to obtain NNs capable of extracting  $KP$ ,  $V_T$  and  $R_C$ . The data were not normalised to remove the  $W$  and  $L$  parameters (Eq. 32). For this experiment the datasets are designed to extract from a TFT with  $W = 80$   $\mu\text{m}$  and  $L = 10$   $\mu\text{m}$ .

The training performed on the first dataset was done as would commonly be done in computing, focussing on obtaining the most learning possible with the highest score on the validation samples. As mentioned in 4.2, this is achieved by treating the data in a way that provides the most knowledge for better pattern detection. For set 3 and the fourth set formed by the union of the second and third sets, the  $I-V$  curve samples did not undergo a special pre-processing to increase learning and have an excellent score in the validation samples, only the  $I_D$  current was increased one hundred times, with the purpose of having larger values and facilitating the identification of  $R_C$ , since as shown in Fig. 5a, the effect of  $R_C < 4 \times 10^3 \Omega$  is not noticeable to the naked eye, and only the  $I_D$  values were used when  $V_{GS} > 1.9$  V, which is the most significant section of the curve.

After completion of the NN models training for the different datasets, the extraction tests were performed using the physical measurements of the T1.

## 6 Results and discussion

This section presents the results obtained in the different experiments described in the previous section. To validate the results, a comparison of the modelling obtained from the physical measurements was made, using the parameters extracted by the NN models and an extraction performed by the analytical method (using Eq. 1). Thereby, the accuracy achieved by the proposed method is demonstrated. The analytical extraction was only performed to extract  $KP$  and  $V_T$  in saturation regime, analytical extraction of  $R_C$  is not available.

### 6.1 Results of experiment 1

The training of the NN models (Table 1) showed good performance (no overfitting or underfitting). Table 3 presents the MSE and  $R^2$  obtained by each model in the validation samples, in both saturation and linear regimes. The models can predict the parameters  $KP$  and  $V_T$  up to 99% average precision rate.

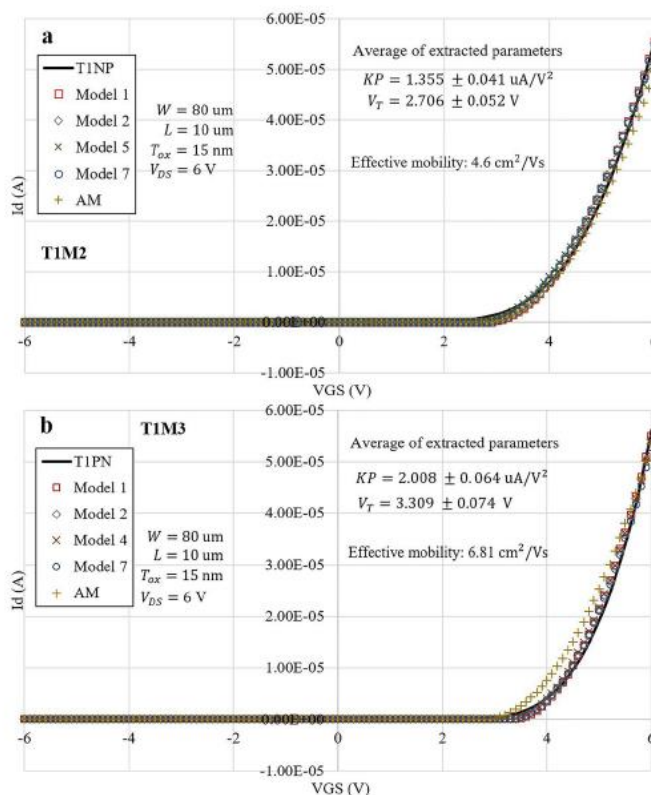
Figure 3a shows the modelling with the extracted parameters at T1, measured from negative to positive (measurement 2), where NN model 1 obtained an error rate of 1.67%, model 2 obtained 3.01%, model 5 obtained 2.07%, model 7 obtained 1.42% and AM obtained 9.67%. Figure 3b shows the modelling of the T1 measurement, measured from PN (measurement

**Table 3** Dimensions of the transistors from experimental measurements

NN model	Saturation		Linear	
	MSE	$R^2$	MSE	$R^2$
1	$1.150 \times 10^{-4}$	0.997	$3.964 \times 10^{-5}$	0.999
2	$4.747 \times 10^{-4}$	0.991	$5.598 \times 10^{-5}$	0.998
3	$3.119 \times 10^{-4}$	0.994	$4.621 \times 10^{-5}$	0.999
4	$8.763 \times 10^{-4}$	0.998	$2.714 \times 10^{-5}$	0.999
5	$1.802 \times 10^{-4}$	0.996	$2.432 \times 10^{-5}$	0.999
6	$1.476 \times 10^{-4}$	0.997	$3.141 \times 10^{-5}$	0.999
7	$1.643 \times 10^{-4}$	0.996	$7.609 \times 10^{-5}$	0.998
8	$1.139 \times 10^{-4}$	0.998	$2.348 \times 10^{-5}$	0.999



**Fig. 3** TFT modelling with extracted parameters in negative-to-positive and positive-to-negative measurement (saturation regime), a corresponds to the measurement supplied from negative to positive, and b from positive to negative



3), where NN model 1 obtained an error rate of 2.61%, model 2 obtained 2.03%, model 4 obtained 3.43%, model 7 obtained 0.21% and AM obtained 19.81%.

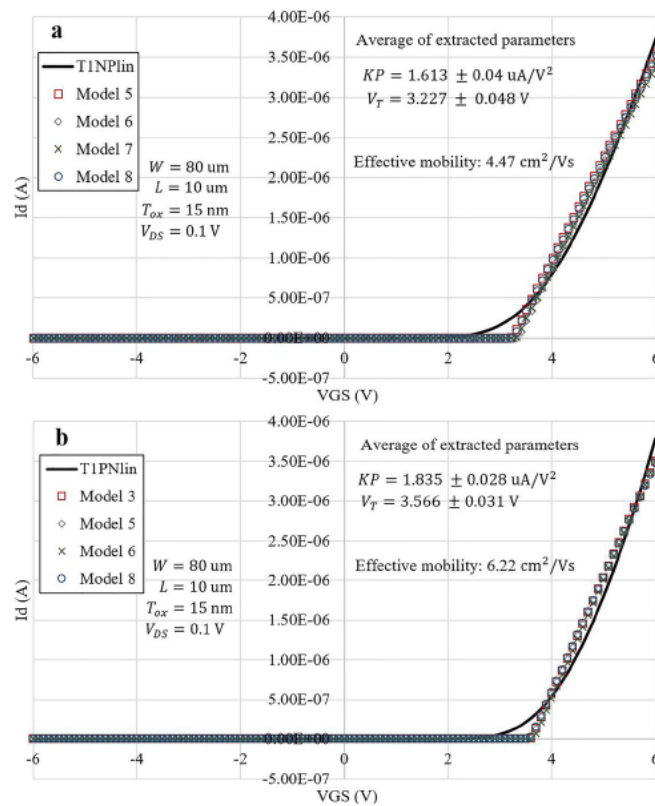
Figure 4a shows the modelling with the parameters extracted from T1 in linear regime. In Fig. 4a (negative to positive measurement), NN model 5 obtained an error rate of 3.61%, model 6 obtained 1.25%, model 7 obtained 3.47% and model 8 obtained 2.03%. In Fig. 4b (positive to negative measurement), NN model 3 had an error rate of 2.09%, model 5 had 2.26%, model 6 had 0.17% and model 8 had 2.20%. In the linear regime measurements, extraction with an analytical method is not available.

## 6.2 Results of experiment 2

In order to analyse the effect of  $R_C$  the TFT was simulated with the parameters extracted in physical measurement 1, of T1 by model 1. Where  $KP = 1.6107 \times 10^{-6} \text{ A/V}^2$  and  $V_T = 2.667 \text{ V}$  are set as constants. Figure 5a shows that there is not significant change from  $0$  to  $4 \times 10^3 \Omega$ , but from  $16 \times 10^3 \Omega$  the electric current starts to decrease, having a large loss at  $64 \times 10^3 \Omega$ .

By feeding the NN model 1 with the transfer curves with different resistances (Fig. 5a) it was found that the NN compensates the value of  $KP$  and  $V_T$  to obtain parameters that fit the input curves. Figure 5b shows the fittings obtained by using the

**Fig. 4** TFT modelling with extracted parameters in negative-to-positive and positive-to-negative measurement (linear regime), **a** corresponds to the measurement supplied from negative to positive, and **b** from positive to negative



parameters extracted by the NN model 1,  $I$ - $V$  curve with a  $R_C$  of  $16 \times 10^3 \Omega$ , obtained an error rate of 1.64%,  $I$ - $V$  curve with  $32 \times 10^3 \Omega$ , obtained 2.38% and  $I$ - $V$  curve with  $64 \times 10^3 \Omega$  obtained 11.48%.

Figure 6a shows the percentage of error obtained between the curves with increased resistance and those modelled with the extracted parameters, where it can be seen that the error increases when  $R_C$  is higher. Which means that the NN is able to compensate the parameters at  $KP$  and  $V_T$  on curves with resistance of about  $32 \times 10^3 \Omega$ , for larger values the error starts to become significant.

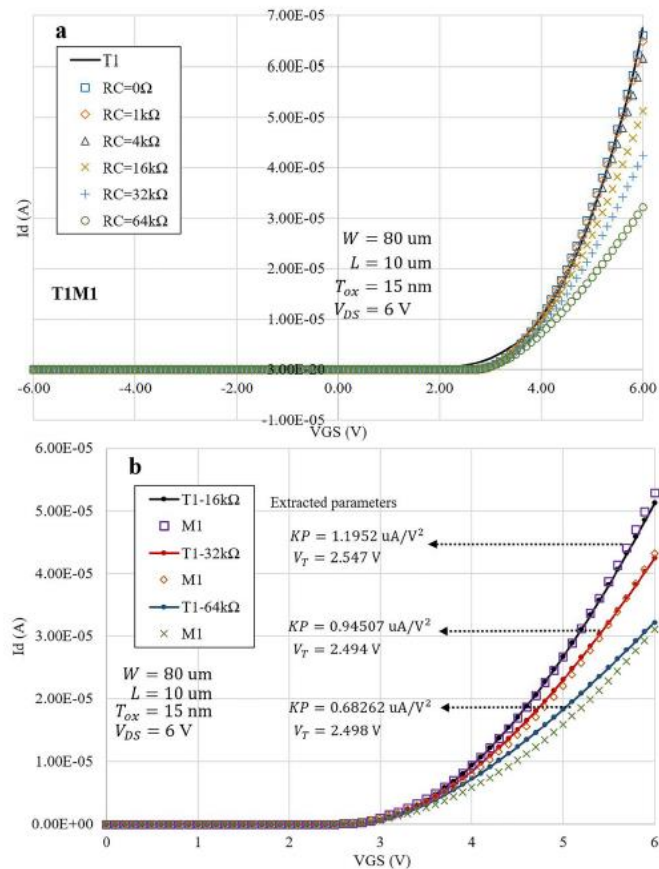
Figure 6b and c show the trend of the parameters  $V_T$  and  $KP$ , where it is observed that  $V_T$  decreases for

resistance values from  $10 \times 10^3 \Omega$ , remaining constant at approximately 2.5 V for resistance values higher than  $32 \times 10^3 \Omega$ .  $KP$  also appears to be inversely proportional to  $R_C$  in an almost linear trend, this behaviour is the same for mobility (Fig. 6d), these two parameters are directly affected by increasing resistance.

### 6.3 Results of experiment 3

With the dataset where the  $I_D$  current was normalised so as not to depend on the geometrical parameters ( $W$  and  $L$ ), the first 3 NN models were trained (Table 1). This experiment was performed only in the saturation regime. Table 4 shows the evaluation

**Fig. 5** Behaviour of T1 with  $R_C$  added and modelled with parameters extracted by NN model 1, **a** corresponds to the I-V curve and its variation with resistance, and **b** presents the fit of three curves with extracted parameters

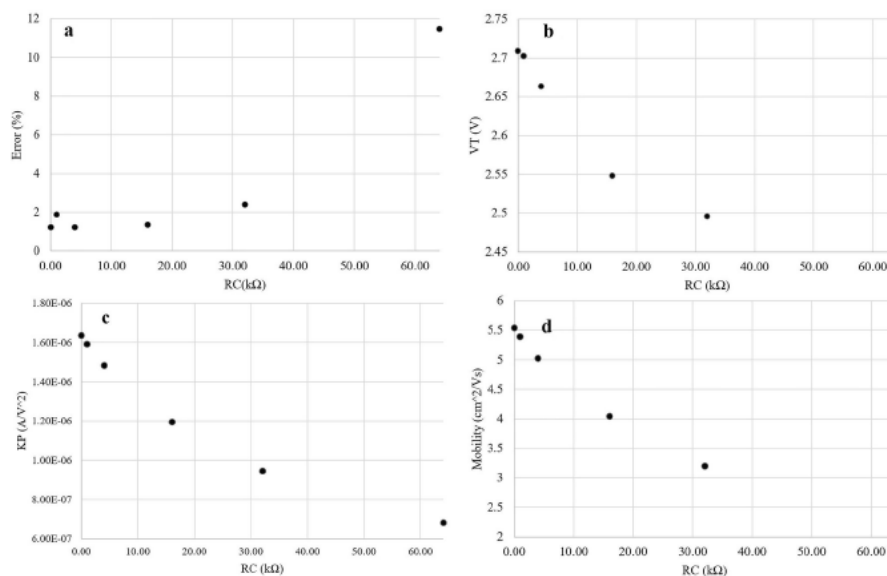


achieved by the models with the validation measurements samples, where the 3 models reached  $R^2 = 0.99$ .

Figure 7a presents the T3 measurement and the curves modelled with the parameters extracted by three NN models trained with the normalised current (Eq. 24) and by the analytical method. Where model 1 obtained an error rate of 12.4%, model 2 obtained 16.49%, model 3 obtained 22.98% and AM obtained 9.86%. Figure 7b presents the modelling with the extracted parameters for T7, where model 1 obtained

an error rate of 11.79%, model 2 obtained 14.22%, model 3 obtained 17.62% and AM obtained 15.52%. The results show an increase in modelling error with the parameters extracted by the NN models that were trained with the normalised  $I-V$  curves to remove the channel width and length. In this comparison, the AM has a smaller error rate compared to the NNs.

Figure 8 shows the average error rate (the 3 NN models) obtained for the 7 transistor sizes (Table 2), where an error between 12 and 19% is observed, with a peak of up to 25% for the  $W = 80 \text{ }\mu\text{m}$  and  $L = 10 \text{ }\mu\text{m}$



**Fig. 6** Percentage error in parameter extraction (a), threshold voltage (b), transconductance (c) and mobility (d) as a function of  $R_C$

**Table 4** Evaluation of the NN models using normalised current

NN model	MSE	$R^2$
1	$2.209 \times 10^{-5}$	0.996
2	$2.258 \times 10^{-5}$	0.999
3	$2.244 \times 10^{-5}$	0.998

transistor. The total average error of 16.89% could be acceptable considering that it was not necessary to have samples for each transistor and the extracted parameters are close to the desired one, these values could be adjusted by the expert personnel in charge of the extraction task, this manual adjustment is also often used in analytical methods when the result presents a notable error between the physical measurement and the one modelled with extracted parameters.

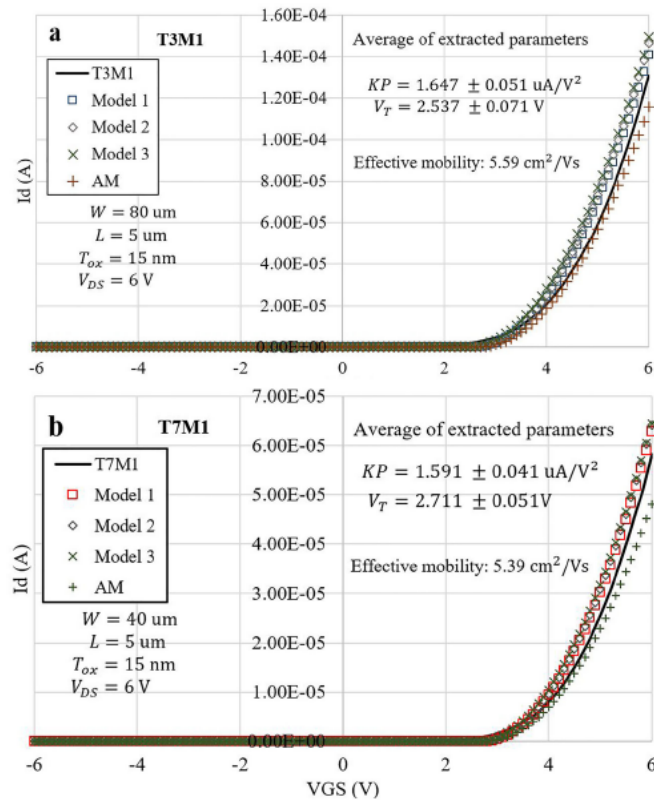
#### 6.4 Results of experiment 4

This section presents the results of the extraction of  $KP$ ,  $V_T$  and  $R_C$ , for which the training of NN models

was carried out using three different datasets. Figure 9a presents the average performance obtained by the eight trained NN models (Table 1) using the second dataset. This was achieved by increasing the current of each sample, dividing them by the maximum current of the set, which was  $I_D = 2.856384 \times 10^{-4}$  A. For example, if the current value was  $I_D = 6.8323 \times 10^{-5}$  A, when  $V_{GS} = 6$  V, after division the value would be  $I_D = 2.392 \times 10^{-1}$  A. This increase allows to have a greater distance between each sample and thus facilitate the identification between them and as a result, to have a better training of the NNs. After training, a performance of 99.4% for  $KP$ , 99.7% for  $V_T$  and 87.5% for  $R_C$  was obtained. Having the lowest learning in  $R_C$  in 87.5% due to the slight effect that this parameter has in comparison with  $KP$  and  $V_T$ .

Figure 9 presents the physical measurement of T1NP, together with the curves obtained using the extracted parameters. Where the four models with the lowest error percentages were selected, model 4 had 14.25% error, model 5 had 45.16% error, model 6 had 64.39% error and model 7 had 19.53% error.

**Fig. 7** Modelling of two devices, T3 (a) and T7 (b) using parameters extracted with NN trained with normalised current



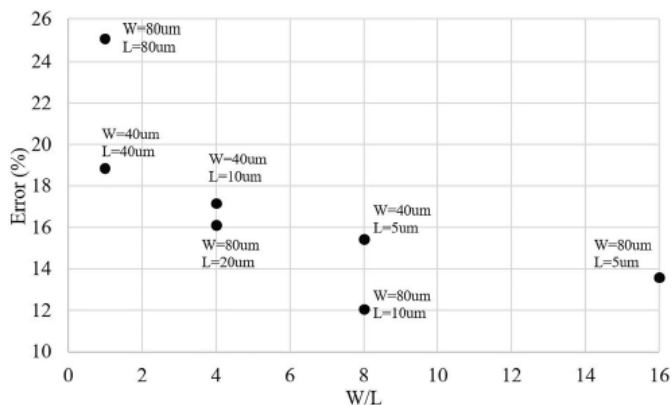
The lack of fit of the physical measurements with the parameters extracted by the NNs was due to low  $KP$  values. This happened because the focus was on the NNs predicting the parameters as accurately as possible during their training, falling into what is known as overtraining, which means that the NNs learn the input and output data “by heart”, but with new samples the results are erroneous.

To improve the results, training was performed with the third dataset, but for this training, the samples were not pre-processed using the maximum current value of the set, but only increased using  $I_D = I_D \cdot 100$ . For example, if in one sample the current when  $V_{GS} = 6 \text{ V}$  was  $I_D = 6.5256 \times 10^{-5} \text{ A}$ , after

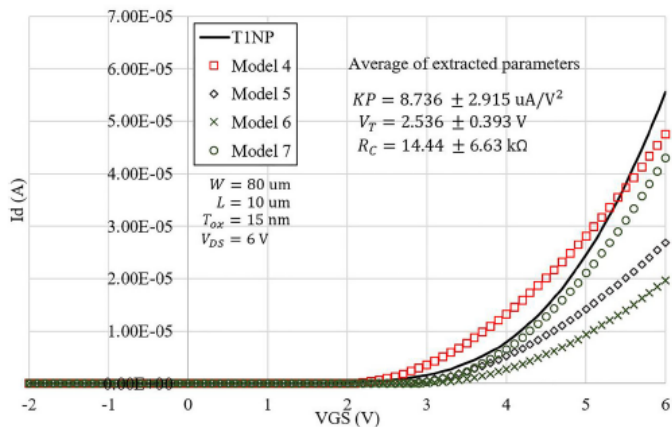
multiplication, it would be  $I_D = 6.5256 \times 10^{-3} \text{ A}$ , this slight increase allows to improve the identification of  $KP$  and  $V_T$ , but not to the point of overtraining again. After training the eight NN models, an average performance of 94.6% was obtained for predicting the  $KP$  parameter, 99.1% for predicting  $V_T$  and only 21.3% for predicting  $R_C$ . The decrease in the identification of the  $R_C$  parameter is noticeable when the  $I_D$  of the samples is not increased, and the NNs are allowed to gain as much knowledge as they can from the data themselves.

Figure 10a presents the physical measurement of T1NP, and the curves obtained using the extracted parameters, again, with the intention of not

**Fig. 8** Average extraction error rate for different transistors



**Fig. 9** Extraction and modelling of TINP with extracted parameters. The training of the NNs was done using the second dataset

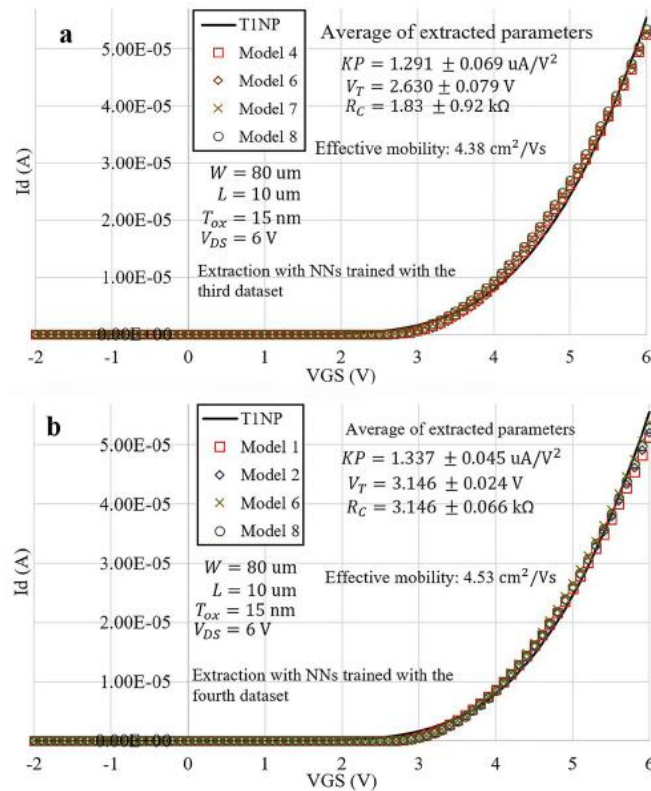


saturation of the graph, the 4 models with the lowest percentage error are presented. Where model 4 obtained an error percentage of 0.93%, model 6 obtained 1.69%, model 7 obtained 3.62% and model 8 obtained 4.62%. The decrease in error is noticeable when overtrained models are not obtained. Although, the trained NN models for the third set only learned 21.3% in the  $R_C$  parameter, they identified with high accuracy the parameters of the physical measurements.

The training of NN models using the union of the second and third sets aimed to test whether  $R_C$

learning improves with more examples (840 samples). Using the same form of preprocessing as for the third set ( $I_D = I_{D,100}$ ), a performance of 94.7% for  $KP$ , 99.2% for  $V_T$  and 12.03% in  $R_C$  was obtained. The learning of  $R_C$  did not improve by having more examples, on the contrary it decreased, this is because the union of the second and third set shortened the difference in the  $KP$  parameter, so that two samples, one with a  $KP = 1.2 \times 10^{-4} \text{ A/V}^2$  and another with a  $KP = 1.4 \times 10^{-4} \text{ A/V}^2$ , both with the same value of  $R_C$ , have no great difference, considering that  $R_C$  affects the last region of  $V_{GS}$ . Figure 10b shows the

**Fig. 10** Extraction and modelling of TINP-PN with extracted parameters. Using the third and fourth dataset during training, where **a** corresponds to the supplied measurement from negative to positive and **b** from positive to negative



physical measurement of TINP and the curves obtained using the extracted parameters, where again the four with the lowest error are shown. Where model 1 obtained a percentage error of 0.92%, model 2 obtained 0.06%, model 6 obtained 2.42% and model 8 obtained 0.51%. It is possible to observe that having a greater number of examples did improve the extraction of  $KP$  and  $V_T$ , which was reflected in the fact that more models obtained an error of less than 1%.

## 7 Conclusion

In this research, Artificial Neural Networks are proposed for the parameter extraction process in Thin Film Transistors IGZO with Al contacts. The parameters extracted were  $KP$ ,  $V_T$  and  $R_C$ , as they are parameters that allow modelling the behaviour of the TFTs. Four different experiments were carried out to prove that given the correct dataset of  $I$ - $V$  sample curves, the Neural Networks are capable to identify and extract their parameters. In this work, errors were comparable to and in some cases smaller than in [21, 22].

In the first experiment, different models of NNs were trained to extract the parameters  $KP$  and  $V_T$ ,

where the average performance in the validation measurements reached an  $R^2 = 0.998$ , and in physical measurements with minimum error rates of 1.68–2.04% (average of the best 4 models) because the data sets designed for this experiment were in the perfect range in the  $KP$  and  $V_T$  sweeps. The result of the NNs extraction showed higher accuracy compared to the analytical method applied. In this experiment, model 7 (4 layers of 256, 128, 64 and 32 neurons) obtained the lowest error rates in 3 of the 4 extractions presented.

In the second experiment, a trained NN model was fed with a measurement of  $T_{IM1}$  to which, by means of the simulator, different values of  $R_C$  were added. It was found that the trained NNs are able to compensate the  $KP$  and  $V_T$  values to fit measurements to which a contact resistance was added from 0 to  $64 \times 10^3 \Omega$ , with a percentage of error of about 2% when  $R_C < 64 \times 10^3 \Omega$ . This allows the extraction of parameters from  $I$ - $V$  curves even with medium-high resistances.

In the third experiment, the geometrical factor was removed from the first dataset, multiplying the  $I_D$  current by  $L/W$ , in order to use the first dataset to extract the parameters of physical measurements of different sized transistors, which were also multiplied by their own  $L/W$ . In this experiment only 3 different models were trained, which obtained an average  $R^2$  of 0.997 in the validation samples, and from the 7 available dimensions an average error rate of 16.89% was obtained. With the increased error in this experiment, the extraction result with the analytical method was more accurate with an error of about 9%. This experiment was proposed as a simple way to overcome the need for samples to extract parameters in transistors with different channel widths and lengths. In this experiment, model 1 (2 layers of 128 and 64 neurons) was the one with the lowest extraction error rate.

In the fourth experiment, the extraction of the parameters  $KP$ ,  $V_T$  and  $R_C$  was performed, for which training was done using the second, third and a union of these datasets. In the extraction using the second dataset, an example was presented of how NNs can learn to identify the parameters of validation samples with average  $R^2$  of 95.53% but fail in extraction tests against physical measurements due to overtraining. On the other hand, by using the third

dataset, overtraining was solved, avoiding excessive preprocessing. In this way, an average  $R^2$  of 71.6% was obtained, although it could be considered as low, these trained NN models obtained minimum error percentages of up to 0.93% in the extraction of physical measurement. Finally, training was performed with a dataset consisting of the second and third data sets, from which an average  $R^2$  of 68.6% was obtained, again, a low overall performance, but with these trained NN models, error rates of up to 0.06% were obtained in extraction tests on physical measurements. In the extraction of  $KP$ ,  $V_T$  and  $R_C$  in this experiment, model 8 (5 layers of 256, 128, 64, 32 and 16 neurons) had one of the lowest errors in the extractions presented, using both the third and fourth training set.

### Acknowledgements

Nanoscience, Micro and Nanotechnologies Centre of the National Polytechnic Institute is thanked for the fabrication of devices whose transferential curves were used to test the methodology proposed in this research. Thanks, are also due to the National Council of Science and Technology for the scholarship for advanced studies.

### Author contributions

RCV performed the parameter extraction using NNs and drafted the manuscript, NHC manufactured the transistors with which the method proposed in this work was tested, RZG performed the parameter extraction using the analytical method, FGL and ALC analysed the proposed method and improved the form and wording of the manuscript.

### Funding

The authors have not disclosed any funding.

### Data availability

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



## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

## References

- S. Raczynski, *Modeling and simulation*, 1st edn. (Wiley, Hoboken, 2014), pp.1–14
- H. Khan, M. A. Bazaz, S. A. Nahvi, Simulation Acceleration of High-Fidelity Nonlinear Power Electronic Circuits Using Model Order Reduction, in: 5th IFAC Conference on Advances in Control and Optimization of Dynamical Systems (2018) <https://doi.org/10.1016/j.ifacol.2018.05.069>
- P. Moreno, R. Picos, M. Roca, E. García, B. Iniguez, M. Estrada, Parameter extraction method using genetic algorithms for an improved OTFT compact model. Spanish Conf. Electron Devices (2007). <https://doi.org/10.1109/SCED.2007.383996>
- C. Tanaka, 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 (IEEE, 2018) doi: <https://doi.org/10.1109/ICMTS.2018.8383756>
- A. Ortiz-Conde, F.J. García-Sánchez, J. Muci, A. Terán Barrios, J.J. Liou, C.-S. Ho, Microelectron. Reliab. (2013). <https://doi.org/10.1016/j.microrel.2012.09.015>
- A. Cerdeira, M. Estrada, R. García, A. Ortiz, F.J. García, Solid State Electron (2001). [https://doi.org/10.1016/S0038-1101\(01\)00143-5](https://doi.org/10.1016/S0038-1101(01)00143-5)
- S.K. Ojha, B. Kumar, SILICON (2022). <https://doi.org/10.1007/s12633-021-01149-6>
- C. Avila, A. Ortiz, J.A. Caraveo, M.A. Quevedo, Trans. Electr. Electron. Mater. **22**(4), 550–556 (2021). <https://doi.org/10.1007/s42341-020-00268-y>
- P. Mittal, Y.S. Negi, R.K. Singh, J. Comput. Electron (2015). <https://doi.org/10.1007/s10825-015-0719-8>
- K. Bhargava, V. Singh, J. Comput. Electron (2014). <https://doi.org/10.1007/s10825-014-0574-z>
- S. Xin-zhi, L. Hai-wen, S. Xiao-wei, C. Yan-feng, C. Zhi-qun, L. Zheng-fan, Wuhan Univ. J. Nat. Sci. (2005). <https://doi.org/10.1007/BF02830676>
- S. Nautiyal, P. Mittal, Contact resistance in organic transistors: Extraction using variable length method, in: 2017 International Conference on Computing, Communication and Automation, (IEEE, 2017) doi: <https://doi.org/10.1109/CCA.A.2017.8230051>
- M.E. Rivas-Aguilar et al., Cur. Appl. Phys. (2018). <https://doi.org/10.1016/j.cap.2018.04.002>
- A. Pacheco-Sanchez, M. Claus, S. Mothes, M. Schröter, Solid State Electron. (2016). <https://doi.org/10.1016/j.sse.2016.07.011>
- H. Bae et al., IEEE Electron Device Lett. (2016). <https://doi.org/10.1109/LED.2015.2509473>
- N. Akkan, M. Altun, 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 (IEEE, 2018) doi: <https://doi.org/10.1109/SMACD.2018.8434861>
- S. Moparthi, P. K. Tiwari, 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 (IEEE, 2021) doi: <https://doi.org/10.1109/ISDCSS2006.2021.9397911>
- S. I. Sayed, M. M. Abutaleb, Z. B. Nossair, Improved CNFET performance based on genetic algorithm parameters optimization, in: 2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEEE, 2017) doi: <https://doi.org/10.1109/IEMCON.2017.8117185>
- I. Benacer, Z. Dibi, Int. J. Autom. Comput. (2016). <https://doi.org/10.1007/s11633-015-0918-6>
- R. Picos et al., Solid. State. Electron. (2007). <https://doi.org/10.1016/j.sse.2007.02.031>
- J. Oh et al., ECS J. Solid State Sci. (2022). <https://doi.org/10.1149/2162-8777/ac6894>
- Q. Yao et al., Micromachines (2021). <https://doi.org/10.3390/mi13010004>
- Berkeley, Aim-Spice, (Software, 2022), <http://www.aimspice.com/>. Accessed 08 August 2022
- R.J. Baker, CMOS, 3rd edn. (Wiley-IEEE Press, Hoboken, 2010), pp.131–145
- M. Flasiński, *Introduction to artificial intelligence*, 1st edn. (Springer, Cham, 2016), pp.156–157. <https://doi.org/10.1007/978-3-319-40022-8>
- P. Isasi, I. Galván, *Rates neuronales artificiales* (Pearson Prentice Hall, Madrid, 2004), pp.1–60
- M. Hagan, H. Demuth, M. Beale, O. De Jesus, *Neural network design*, 2nd edn. (Ebook, Nedlands, 2014), p.2.2-2.10
- W. Ertel, *Introduction to artificial intelligence*, 2nd edn. (Springer, Cham, 2017), pp.245–260. <https://doi.org/10.1007/978-3-319-58487-4>
- Analog Devices Inc., LTSpice, (Software, 2022), <https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html>. Accessed 09 August 2022
- Scikit.Learn, sklearn.model\_selection.GridSearchCV, (Website, 2022), [https://scikit-learn.org/stable/modules/generated/sklearn.model\\_selection.GridSearchCV.html#sklearn.model\\_selection.GridSearchCV](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html#sklearn.model_selection.GridSearchCV). Accessed 09 August 2022

31. Google, Colaboratory, (Website, 2022), <https://research.google.com/colaboratory/intl/es/faq.html>. Accessed 09 August 2022
32. M. Abadabi et al., Tensorflow, (Website, 2022), <https://www.tensorflow.org/?hl=es-419> Accessed 09 August 2022

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

# 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; rvaldeg564@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 (CIDEI), Queretaro, Mexico; rodolfo.sanchez@cidesi.edu.mx

G. Lastra Medina, is a researcher from Center for Engineering and Industrial Development (CIDEI), 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 difficulty 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. THEORETICAL 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.

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

Where  $\mu_{FET}$  represents the effective mobility,  $C_{ox}$  is the capacity of the oxide,  $\alpha_{sat}$  is the Proportionality constant of  $V_{sat}$ . 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
$\alpha_{sat}$	Proportionality constant of $V_{sat}$	-	1
AT	Drain induced barrier lowering 1	m/V	$3 \times 10^{-8}$
BT	Drain induced barrier lowering 2	m/V	$1.9 \times 10^{-6}$
ETA	Subthreshold ideality factor	-	7
IO	Leakage scaling constant	A/m	6
MMU	Low field mobility exponent	-	3
MUO	High field mobility	$cm^2/Vs$	100
$\mu_0$	Low field mobility parameter	$cm^2/Vs$	0.0022
MUS	Subthreshold mobility	$cm^2/Vs$	1
$T_{ox}$	Thin-oxide thickness	m	$1 \times 10^{-7}$
$\overline{W}$	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  $V_T=3V$ ,  $R_L=510k\Omega$ ,  $IO=6A/m$  and  $MMU=3$ . The other dotted curves represent the change they have when modifying the parameters to  $V_T=1V$ ,  $R_L=1M\Omega$ ,  $IO=10kA/m$  and  $MMU=3$ . Modifying one parameter at a time and keeping the others the same as the original.

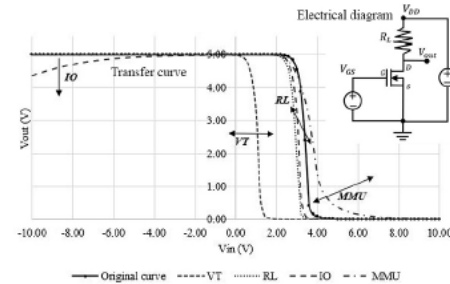


Fig. 1. Transfer curve, electrical diagram of a resistive load inverter circuit and its behavior with different parameters.

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,  $IO$ , 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:

$$a = f(wp + b) \quad (3)$$

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).

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

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

$$w_{ij}^m(k+1) = w_{ij}^m(k) - \alpha \frac{\partial F}{\partial w_{ij}^m} \quad (5)$$

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

#### 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).

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

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).

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

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).

$$f(x) = (w_1 x_1 + \dots + w_d x_d) + b \quad (9)$$

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)

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

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

$$y_i - ((w, x_i) + b) \leq \varepsilon + \xi_i \quad (11)$$

$$((w, x_i) + b) - y_i \leq \varepsilon + \xi_i^* \quad (12)$$

$$\xi_i \geq 0, \xi_i^* \geq 0 \quad (13)$$

$C$  is a constant that determines the balance between  $f$  and the measure of tolerances at deviances greater than a  $\varepsilon$ . The variables  $\xi_i$  and  $\xi_i^*$  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  $V_T, R_L, I_0$  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.

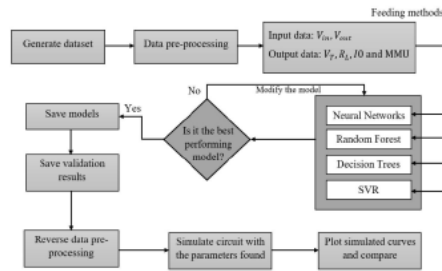


Fig. 2. General diagram of the proposed methodology.

### 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  $W=100\mu\text{m}$ ,  $L=10\mu\text{m}$  and  $T_{ox}=10\text{ nm}$  with sweeps from  $V_T=1\text{V}$  to  $5\text{V}$  at a rate of  $1\text{V}$ , for each value of  $V_T$  a sweep was made in  $R_L$  with  $10\text{k}\Omega$ ,  $39\text{k}\Omega$ ,  $100\text{k}\Omega$ ,  $510\text{k}\Omega$  and  $1\text{M}\Omega$ . For each value of  $R_L$ , a sweep was made in  $I_O$  with  $6\text{A/m}$ ,  $1\text{kA/m}$ ,  $5\text{kA/m}$  and  $10\text{kA/m}$ . 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  $6\text{V}$ , 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  $10\text{V}$  with  $0.2\text{V}$  increments. This range allows to have complete transition regions for  $V_T$  and  $R_L$ , and in negative  $V_{in}$  the effect of  $I_O$  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  $V_T$ ,  $R_L$ ,  $I_O$  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.

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

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  $1 \times 10^{-2}$  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  $I_O$  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  $V_T$ ,  $R_L$  and  $MMU$  have the best performance, with  $I_O$  being the parameter with the highest error.

Parameter	$R^2$	MSE	Error (%)
$V_T$	0.9991	$2.470 \times 10^{-2}$	2.27
$R_L$	0.999	$2.690 \times 10^{-2}$	2.86
$I_O$	0.9596	$1.710 \times 10^{-1}$	16.45

$MMU$	0.9995	$1.710 \times 10^{-2}$	2.57
Mean	0.98	$6.003 \times 10^{-2}$	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  $I_O$ , 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  $I_O$  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,  $I_O$  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.

Parameter	$R^2$	MSE	Error (%)
$V_T$	0.9798	$1.218 \times 10^{-1}$	8.90
$R_L$	0.9688	$1.590 \times 10^{-1}$	14.90
$I_O$	0.8273	$3.547 \times 10^{-1}$	62.96
$MMU$	0.9912	$7.776 \times 10^{-2}$	11.10
Mean	0.94	$1.766 \times 10^{-1}$	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  $I_O$  with  $R^2=0.74$ .

Parameter	$R^2$	MSE	Error (%)
$V_T$	0.9698	$1.488 \times 10^{-1}$	12.49
$R_L$	0.861	$3.213 \times 10^{-1}$	25.78
$I_O$	0.745	$4.310 \times 10^{-1}$	50.52
$MMU$	0.9578	$1.704 \times 10^{-1}$	26.06
Mean	0.8834	$2.679 \times 10^{-1}$	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  $I_O$  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  $I_O$ , but in the other parameters the highest error is in  $MMU$  of 12.5%.

Parameter	$R^2$	MSE	Error (%)
$V_T$	0.9728	$1.414 \times 10^{-1}$	4.63
$R_L$	0.9366	$2.168 \times 10^{-1}$	9.14
$I_O$	0.5151	$5.943 \times 10^{-1}$	92.56
$MMU$	0.9578	$1.705 \times 10^{-1}$	12.55
Mean	0.84557	$2.808 \times 10^{-1}$	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%.

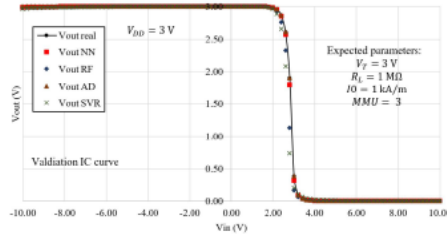


Fig. 3. Transfer curve of the IC, randomly selected from the validation samples.

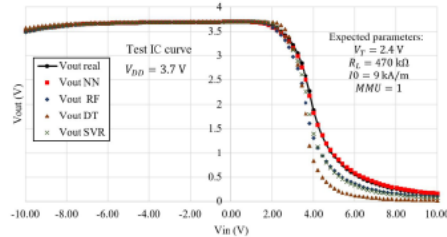


Fig. 4. Test curve with intermediate values in the parameters of the learned range.

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  $IO$  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  $IO$  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  $IO$  is large, the extraction still works because the effect it has is weak, that is, the change in the curve of an  $IO=100$  A/m is almost the same as an  $IO=200$  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  $IO$  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  $\alpha=1$  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.:

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

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

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

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

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

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

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

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

TABLE X  
COMPARISON OF  $V_T$  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	$1.22 \times 10^3$

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  $W=100\mu\text{m}$ , an  $L=8\mu\text{m}$  and a  $T_{ox}=27\text{nm}$ ; device 2 (D2) with a  $W=100\mu\text{m}$ , an  $L=16\mu\text{m}$  and a  $T_{ox}=32\text{nm}$ .

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  $\Omega$  for  $R_S$ , with 100 $\Omega$  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  $\text{cm}^2/\text{Vs}$ . 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 tuning 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.

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 Depth of 15 (D1) and 20 (D2), 250 leaf nodes, 2 samples for slit and 250 estimators	0.99	0.87
RF	Kernel RBF, $C=4$ , gamma "scale"	0.77	0.75
SVR	(D1) and 3 (D2)	0.74	0.84

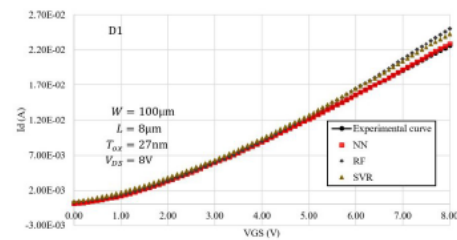


Fig. 5. Modelling of D1 with extracted parameters.

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  $UO=1475\text{cm}^2/\text{Vs}$ ,  $V_T=-0.22\text{V}$  and  $R_S=166\Omega$ , RF extracted  $747\text{cm}^2/\text{Vs}$ ,  $-0.66\text{V}$  and  $86\Omega$  respectively, and SVR extracted  $802\text{cm}^2/\text{Vs}$ ,  $-0.85\text{V}$  and  $108\Omega$  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  $UO=333\text{cm}^2/\text{Vs}$ ,  $V_T=0.6\text{V}$  and  $R_S=530\Omega$ , RF extracted  $294\text{cm}^2/\text{Vs}$ ,  $0.44\text{V}$  and  $321\Omega$  respectively, and SVR extracted  $299\text{cm}^2/\text{Vs}$ ,  $0.36\text{V}$  and  $393\Omega$  respectively. The percentage of error of each modelling is 0.44% for NN, 3.47% for RF and 0.57% for SVR.

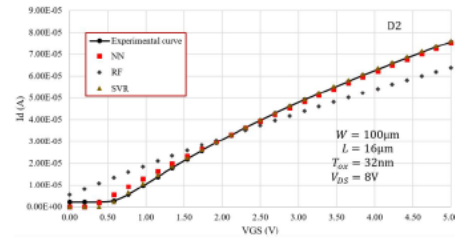


Fig. 6. Modelling of D2 with extracted parameters.

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  $UO=1475\text{cm}^2/\text{Vs}$ ,  $V_T=-0.22\text{V}$  and  $R_S=166\Omega$ , RF extracted  $747\text{cm}^2/\text{Vs}$ ,  $-0.66\text{V}$  and  $86\Omega$  respectively, and SVR extracted  $802\text{cm}^2/\text{Vs}$ ,  $-0.85\text{V}$  and  $108\Omega$  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  $UO=333\text{cm}^2/\text{Vs}$ ,  $V_T=0.6\text{V}$  and  $R_S=530\Omega$ , RF extracted  $294\text{cm}^2/\text{Vs}$ ,  $0.44\text{V}$  and  $321\Omega$  respectively, and SVR extracted  $299\text{cm}^2/\text{Vs}$ ,  $0.36\text{V}$  and  $393\Omega$  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. Garcia-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. Cedeira, M. Estrada, R. Garcia, A. Ortiz-Conde, and F. J. Garcia 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 OFET 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. Dreffal, "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. Jandali, "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. Saramakala, "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. Iniguez, E. Garcia-Moreno, R. Garcia, 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 Conference, 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. Components 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.aimsipice.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 Ed. 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. Pedragosa 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 García 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.