Revista Innovación y Software
Vol. 7, No. 2, Mes Septiembre - Febrero, 2026
ISSN: 2708-0935
Pág. 18-26
https://revistas.ulasalle.edu.pe/innosoft

Esta obra está bajo una Licencia Creative Commons Atribución 4.0 Internacional.

Tipo de artículo: Artículos cortos
Temática: Inteligencia artificial
Recibido: 28/07/2026 | Aceptado: 27/08/2026 | Publicado: 30/09/2026

Identificadores persistentes:
DOI: 10.48168/innosoft.s30.a388
ARK: ark:/42411/s30.a388

La neurona artificial y el perceptrón simple: implementación desde cero y sus límites frente al problema XOR

The artificial neuron and the simple perceptron: implementation from scratch and its limitations against the XOR problem

Kris Ximena Colquehuanca Zapana[1][0009-0004-2666-3598]*, **Nilver Wilfredo Flores Pacco**[2][0009-0003-1436-4884]

[1]Facultad de Ingeniería Estadística e Informática, Universidad Nacional del Altiplano - Puno. Av. Floral 1153, Ciudad Universitaria, Puno, Perú. 75453117@est.unap.edu.pe
[2]Facultad de Ingeniería Estadística e Informática, Universidad Nacional del Altiplano - Puno. Av. Floral 1153, Ciudad Universitaria, Puno, Perú. nilwillflo749@gmail.com

*Autor para correspondencia: 75453117@est.unap.edu.pe

Resumen
El perceptrón simple constituye la unidad de cómputo fundamental de las redes neuronales artificiales y el punto de partida histórico del aprendizaje automático conexionista. El objetivo de este trabajo fue implementar un perceptrón desde cero, empleando únicamente la biblioteca NumPy y sin recurrir a marcos de aprendizaje profundo, con el fin de comprender su funcionamiento interno: la suma ponderada de las entradas, la función de activación escalón y la regla de aprendizaje por corrección de error. Se entrenó el modelo sobre las compuertas lógicas AND, OR y XOR, y se evaluó su exactitud mediante la comparación entre las salidas predichas y las esperadas. Los resultados mostraron que el perceptrón aprendió correctamente las compuertas AND y OR, alcanzando una exactitud del 100 %, dado que ambas son linealmente separables; en cambio, falló en la compuerta XOR, con una exactitud del 50 %, pues sus clases no pueden separarse mediante una sola recta, incluso al incrementar el número de épocas. Se comprobó así, de forma empírica, la frontera entre los problemas que el perceptrón simple puede y no puede resolver. Esta limitación estructural, documentada históricamente, motivó el desarrollo de las redes neuronales multicapa, lo que evidencia la relevancia del perceptrón como base conceptual del aprendizaje profundo actual.

Palabras claves: función de activación, neurona artificial, perceptrón, separabilidad lineal, problema XOR.

Abstract
The simple perceptron is the fundamental computing unit of artificial neural networks and the historical starting point of connectionist machine learning. The aim of this work was to implement a perceptron from scratch, using only the NumPy library and without resorting to deep learning frameworks, in order to understand its inner workings: the weighted sum of inputs, the step activation function, and the error-correction learning rule. The model was trained on the AND, OR, and XOR logic gates, and its accuracy was evaluated by comparing the predicted outputs with the expected ones. The results showed that the perceptron correctly learned the AND and OR gates, reaching 100 % accuracy, since both are linearly separable; in contrast, it failed on the XOR gate, with 50 % accuracy, because its classes cannot be separated by a single straight line, even when the number of epochs was increased. The decision boundaries learned for AND and OR were also visualized, confirming

that the perceptron only draws a straight separation line. This empirically confirmed the boundary between the problems that a simple perceptron can and cannot solve. This structural limitation, historically documented, motivated the development of multilayer neural networks, which highlights the relevance of the perceptron as a conceptual basis of current deep learning and as an essential step for understanding more complex architectures.

Keywords: *activation function, artificial neuron, linear separability, perceptron, XOR problem.*

Introducción

Las redes neuronales artificiales constituyen uno de los pilares de la inteligencia artificial moderna. Su unidad básica, la neurona artificial, se inspira en el funcionamiento simplificado de las neuronas biológicas: recibe varias señales de entrada, las combina de forma ponderada y produce una salida cuando la combinación supera un umbral. El primer modelo matemático formal de este comportamiento fue propuesto por McCulloch y Pitts [1], quienes demostraron que redes de unidades lógicas simples podían representar funciones proposicionales. Posteriormente, Rosenblatt [2] introdujo el perceptrón, incorporando una regla de aprendizaje capaz de ajustar automáticamente los pesos a partir de ejemplos, lo que lo convirtió en el primer modelo entrenable de clasificación.

El perceptrón simple resulta importante porque encapsula las ideas centrales del aprendizaje supervisado: representar el conocimiento en pesos, medir el error frente a una salida esperada y corregir esos parámetros de forma iterativa. Comprenderlo es un prerrequisito para entender arquitecturas más complejas, como el perceptrón multicapa o las redes convolucionales que hoy dominan tareas de visión y lenguaje [3]. Sus aplicaciones son amplias; por ejemplo, en ingeniería civil se emplean para predecir el comportamiento de estructuras y materiales [4].

El problema que aborda este trabajo es doble. Por un lado, se busca mostrar de forma transparente cómo aprende un perceptrón, implementándolo sin marcos de trabajo especializados; por otro, se pretende evidenciar su limitación fundamental. El perceptrón solo puede resolver problemas linealmente separables, es decir, aquellos en que una única recta o hiperplano basta para separar las clases. La compuerta lógica XOR constituye el contraejemplo clásico: sus clases no son separables por una recta, y el fracaso del perceptrón ante ella impulsó históricamente el desarrollo de las redes multicapa. El objetivo de este artículo es desarrollar la técnica, implementarla en Python y verificar de forma empírica esa frontera entre lo que el perceptrón puede y no puede aprender.

Revista Innovación y Software
Vol. 7, No. 2, Mes Septiembre - Febrero, 2026
ISSN: 2708-0935
Pág. 18-26
https://revistas.ulasalle.edu.pe/innosoft

Materiales y métodos

Inspiración biológica y modelos precursores. Las redes neuronales artificiales fueron diseñadas tomando como referencia el funcionamiento del cerebro humano. Cada neurona biológica recibe información por las dendritas, la procesa en el soma y, cuando el estímulo supera cierto umbral, envía una señal por el axón hacia otras neuronas [5]. A partir de ese comportamiento se construyó un modelo computacional simplificado en el que las entradas representan los estímulos, los pesos simulan la intensidad de las conexiones sinápticas y una función de activación determina si la neurona responde. La Figura 1 ilustra esta analogía y la estructura resultante de la neurona artificial.

El primer modelo formal, de McCulloch y Pitts [1], operaba sobre entradas binarias con pesos fijos: sumaba las entradas y las comparaba con un umbral, emitiendo 1 si lo superaba y 0 en caso contrario; su limitación era que no podía aprender. Rosenblatt [2] superó esa restricción con el perceptrón, cuyo modelo se esquematiza en la parte derecha de la Figura 1.

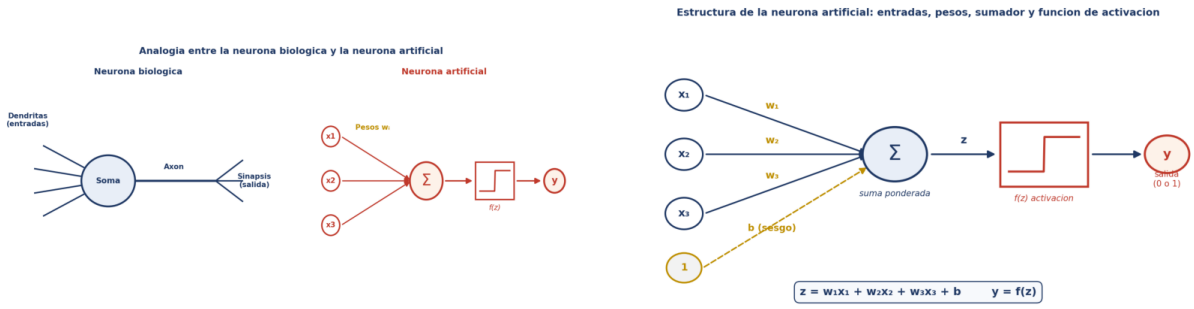


Figura 1. Analogía biológica (izquierda) y estructura de la neurona artificial: entradas, pesos, sumador, activación y salida (derecha). Elaboración propia.

Modelo de la neurona artificial. Una neurona con n entradas calcula una suma ponderada $z = w_1x_1 + w_2x_2 + \ldots + w_nx_n + b$, donde w_i son los pesos y b es el sesgo. Esta suma pasa por una función de activación que determina la salida. En el perceptrón clásico se emplea la función escalón, que devuelve 1 si $z \geq 0$ y 0 en caso contrario. Además del escalón, existen otras funciones de activación de uso extendido en redes modernas, como la sigmoide, la tangente hiperbólica y la ReLU [6], que se comparan en la Figura 2.

Revista Innovación y Software
Vol. 7, No. 2, Mes Septiembre - Febrero, 2026
ISSN: 2708-0935
Pág. 18-26
https://revistas.ulasalle.edu.pe/innosoft

Figura 1. Funciones de activacion

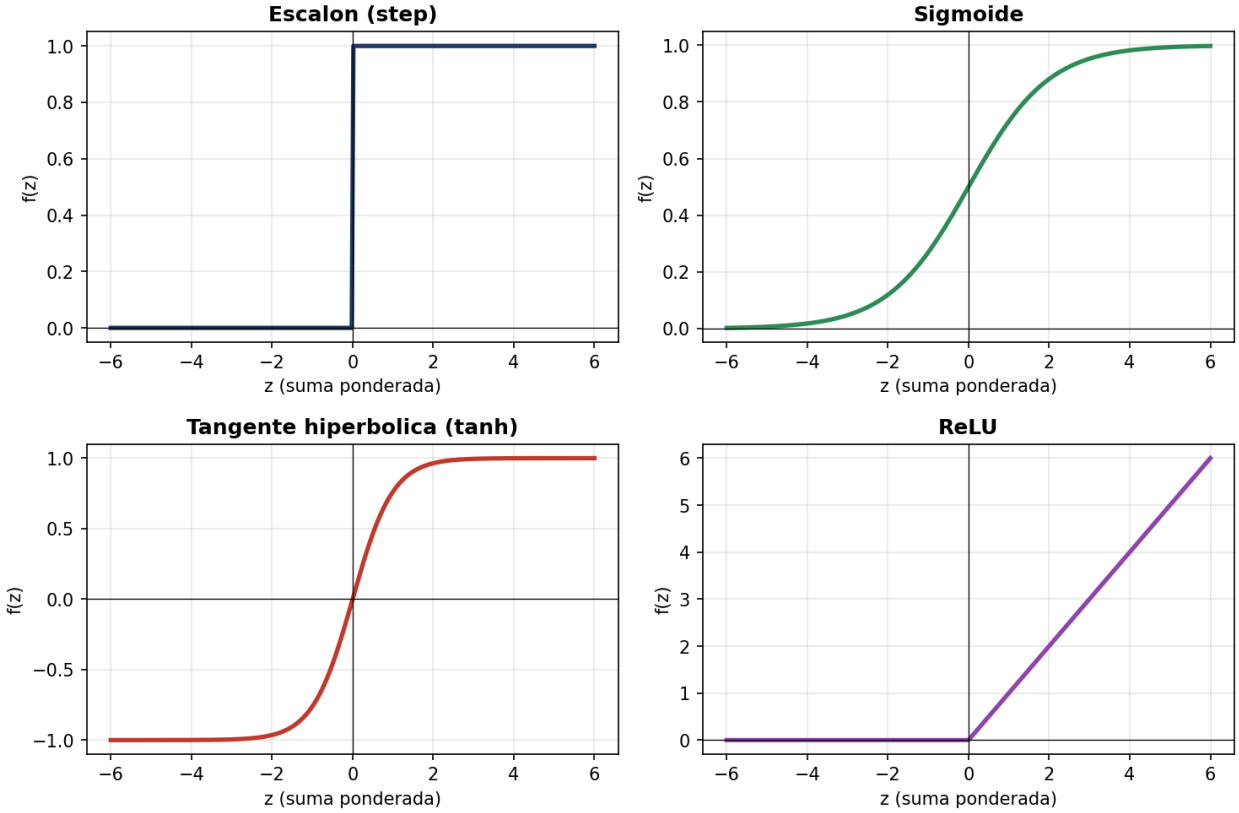


Figura 2. Funciones de activación: escalón, sigmoide, tangente hiperbólica y ReLU. Elaboración propia.

Regla de aprendizaje por corrección de error. El perceptrón ajusta sus pesos comparando la salida predicha con la esperada. Para cada ejemplo, la actualización se realiza según la Ecuación 1, donde η es la tasa de aprendizaje, d la salida esperada, y la salida obtenida y x_i el valor de la entrada. Si la predicción es correcta, no hay cambio; si es incorrecta, los pesos se desplazan en la dirección que reduce el error.

$$w_i \leftarrow w_i + \eta(d - y)x_i \tag{1}$$

Datos y herramientas. Se utilizaron las cuatro combinaciones de entrada de dos bits y las salidas de las compuertas AND, OR y XOR como conjunto de entrenamiento. La implementación se realizó en Python 3 sobre Google Colab, empleando únicamente NumPy y Matplotlib, sin marcos de aprendizaje automático. Se

fijó una semilla para la reproducibilidad, una tasa de aprendizaje de 0,1 y 10 épocas para AND y OR, y 50 para el experimento con XOR.

Resultados y discusión

Compuertas AND y OR. El perceptrón aprendió ambas compuertas sin error. Para AND, los pesos convergieron a $w_1 = 0{,}20$, $w_2 = 0{,}10$ y $b = -0{,}20$, estabilizándose a partir de la tercera época; para OR, a $w_1 = 0{,}10$, $w_2 = 0{,}10$ y $b = -0{,}10$. En ambos casos la exactitud fue del 100 %. En la Figura 3 se muestra la frontera de decisión aprendida para AND: una recta que separa correctamente las salidas 0 de las salidas 1.

Revista Innovación y Software
Vol. 7, No. 2, Mes Septiembre - Febrero, 2026
ISSN: 2708-0935
Pág. 18-26
https://revistas.ulasalle.edu.pe/innosoft

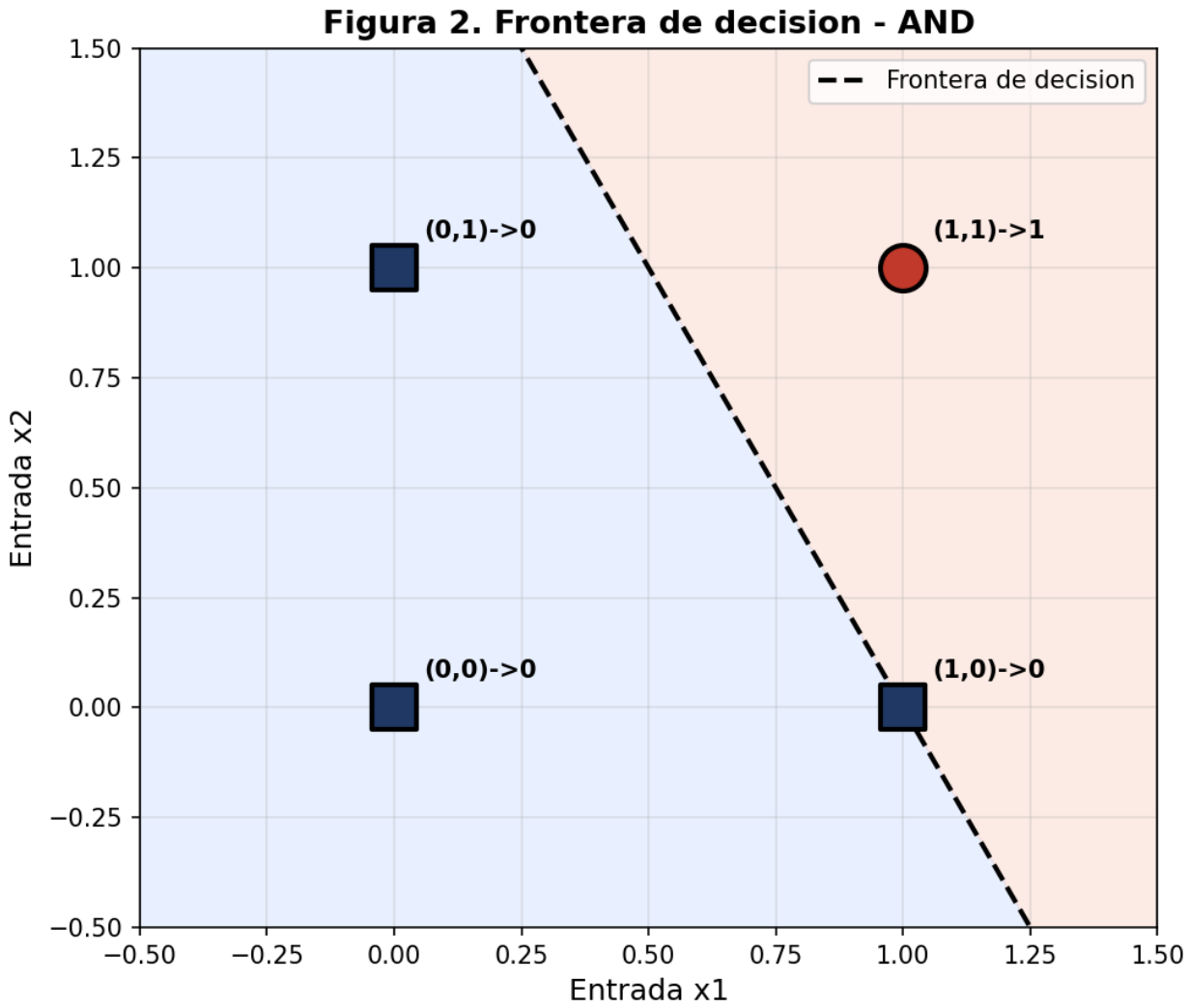


Figura 3. Frontera de decisión aprendida para la compuerta AND. Elaboración propia.

El problema XOR. Al entrenar el perceptrón sobre la compuerta XOR durante 50 épocas, el modelo no logró superar el 50 % de exactitud, y el número de correcciones por época nunca descendió a cero, lo que indica que el algoritmo no converge. En la Figura 4 se ilustra la causa: los cuatro puntos de XOR no pueden separarse mediante una sola recta, pues las salidas positivas ocupan esquinas opuestas del cuadrado unitario. Se comprobó además que el aumento del número de épocas no modifica este resultado, lo que confirma que la limitación es estructural y no un problema de ajuste.

Figura 4. Limitacion del perceptron: el problema XOR

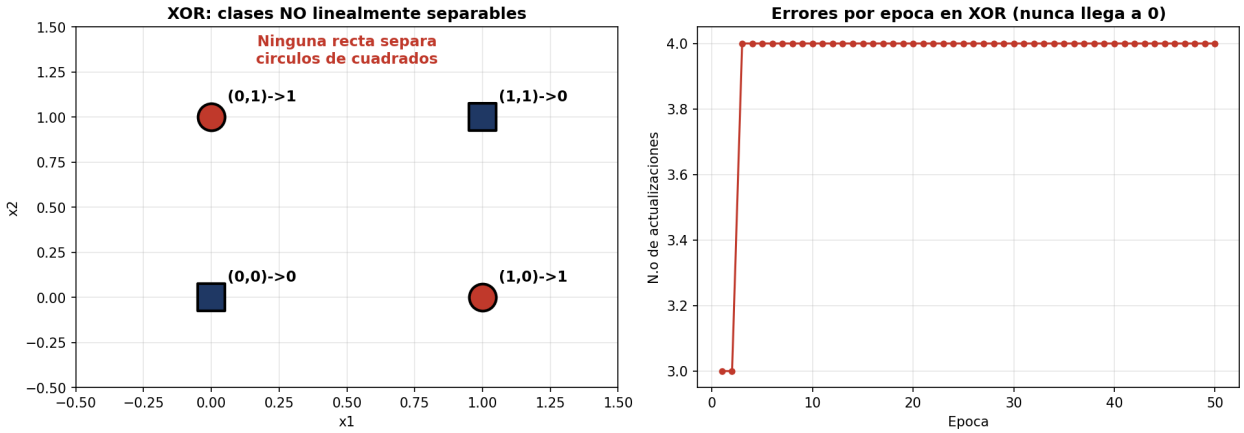


Figura 4. El problema XOR: las clases no son linealmente separables y el error nunca llega a cero. Elaboración propia.

Resumen comparativo. En la Tabla 1 se sintetizan los resultados obtenidos para las tres compuertas lógicas evaluadas.

Tabla 1. Desempeño del perceptrón simple por compuerta lógica.

Compuerta	**Exactitud**	**¿Separable?**	**Resultado**
AND	100 %	Sí	Aprendida
OR	100 %	Sí	Aprendida
XOR	50 %	No	Falla

Los resultados confirman empíricamente el alcance y el límite del perceptrón simple. Su éxito con AND y OR y su fracaso con XOR ilustran el concepto de separabilidad lineal: el perceptrón solo traza una frontera recta, por lo que únicamente resuelve problemas cuyas clases admiten esa separación. La limitación no es de ajuste de hiperparámetros, sino estructural, pues ninguna tasa de aprendizaje ni número de épocas resuelve XOR. Esta barrera motivó históricamente la incorporación de capas ocultas y del algoritmo de retropropagación [7], que dan origen al perceptrón multicapa. Como aplicación regional, un clasificador lineal simple podría apoyar tareas de decisión binaria con pocas variables en el Altiplano puneño, aunque el experimento con XOR advierte que las relaciones no lineales requieren arquitecturas más expresivas.

Conclusiones

El perceptrón simple aprende con exactitud perfecta las compuertas linealmente separables AND y OR, pero es incapaz de resolver la compuerta XOR, cuya exactitud se mantiene en 50 % con independencia del número de épocas. Este hallazgo, reproducido desde cero con NumPy, permite comprender tanto el mecanismo de aprendizaje por corrección de error como la razón histórica y matemática por la que las redes neuronales evolucionaron hacia arquitecturas multicapa. El trabajo constituye una base conceptual clara para el estudio del aprendizaje profundo. Como trabajo futuro se propone implementar un perceptrón multicapa que resuelva XOR y evaluar clasificadores lineales sobre datos reales del contexto altoandino.

Contribución de Autoría

Kris Ximena Colquehuanca Zapana: Conceptualización, Metodología, Software, Validación, Redacción del borrador original. Nilver Wilfredo Flores Pacco: Investigación, Análisis formal, Curación de datos, Visualización, Escritura, revisión y edición.

Referencias

[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943, doi: 10.1007/BF02478259.

[2] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958, doi: 10.1037/h0042519.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, doi: 10.1109/5.726791.

[4] F. J. Valderrama-Purizaca, D. A. Chávez-Barturen, S. P. Muñoz-Pérez, V. A. Tuesta-Monteza, and H. I. Mejía-Cabrera, "Importance of artificial neural networks in civil engineering: a systematic review of the literature," *ITECKNE*, vol. 18, no. 1, pp. 71–83, 2021, doi: 10.15332/iteckne.v18i1.2542.

[5] W. Zamora-Cárdenas, M. Zumbado, and I. Trejos-Zelaya, "McCulloch-Pitts artificial neuron and Rosenblatt's perceptron: An abstract specification in Z," *Technology Inside*, no. 5, pp. 16–29, 2020. [Online]. Available: https://revista.cpic.or.cr/technology-inside/article/view/19

[6] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th International Conference on Machine Learning (ICML)*, Haifa, Israel, 2010, pp. 807–814. [Online]. Available: https://icml.cc/Conferences/2010/papers/432.pdf

[7] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986, doi: 10.1038/323533a0.