El Laboratorio de Neurona Analógica

La Guía Completa de Deep Learning Analógico

v1.0 – De los LEDs a los NTKs

Índice general

Parte I

Introducción

Capítulo 1

Descripción general del kit

Analog Neuron Lab es un laboratorio para tocar, medir y ajustar una red neuronal real, construida con componentes analógicos, donde cada peso y cada sesgo es un elemento físico que puedes girar con un destornillador y verificar con un multímetro.

La mayoría de estudiantes conocen las redes neuronales como una caja negra: introduces números, ocurre algo en silencio, y sale una predicción. Aquí vamos a romper esa sensación por completo. Esta guía y esta placa están pensadas para que una red neuronal deje de ser un dibujo en una diapositiva o un bloque de código y se convierta en algo que se puede entender y tocar con las manos.

La idea es simple: si una neurona artificial es, en esencia, una suma de entradas con pesos y una no linealidad, entonces debería ser posible construirla con electrónica básica y observarla trabajar a tiempo real. El valor educativo no está solo en que funcione, sino en que puedas ver cada paso del procesamiento. Si una salida cambia, puedes rastrear el porqué: qué entrada empuja, qué peso amplifica, qué sesgo desplaza, dónde aparece la saturación, y en qué punto la no linealidad decide apagar o dejar pasar señal.

También hay una segunda idea que hace este kit especial: en el mundo real nada es perfecto. Los componentes tienen tolerancias, los op amps saturan, los diodos no tienen un umbral mágico, y las medidas tienen ruido. En este laboratorio, esa imperfección no es un enemigo. Es parte de la historia. Aprenderás qué significa que un modelo "funcione" cuando está construido con materia y no con números ideales, y por qué eso ayuda a entender mejor tanto la electrónica como la inteligencia artificial.

Cómo leer esta guía

Esta guía está escrita para varios niveles a la vez. Si vienes de instituto, podrás seguir el hilo experimental sin necesidad de una base fuerte en matrices o electrónica. Si vienes de universidad, encontrarás el puente natural entre ecuaciones y señales. Si eres ya un experto, verás el mismo sistema con el lenguaje correcto para hablar de modelos, hipótesis, limitaciones físicas, incertidumbre y validación.

Esta sección describe la placa del laboratorio desde un punto de vista práctico. Su objetivo es que cualquier estudiante pueda identificar los elementos físicos del sistema, conectarlo correctamente y saber cómo interactuar con él antes de empezar los experimentos. No es necesario comprender todavía qué hace internamente la red. Aquí solo se describe cómo se trabaja con la placa.

1.1 Qué es, y qué no es, este laboratorio

Analog Neuron Lab es un kit educativo de computación analógica aplicada a redes neuronales. No es un microcontrolador que simula una red por dentro, ni una aplicación que "hace magia" con

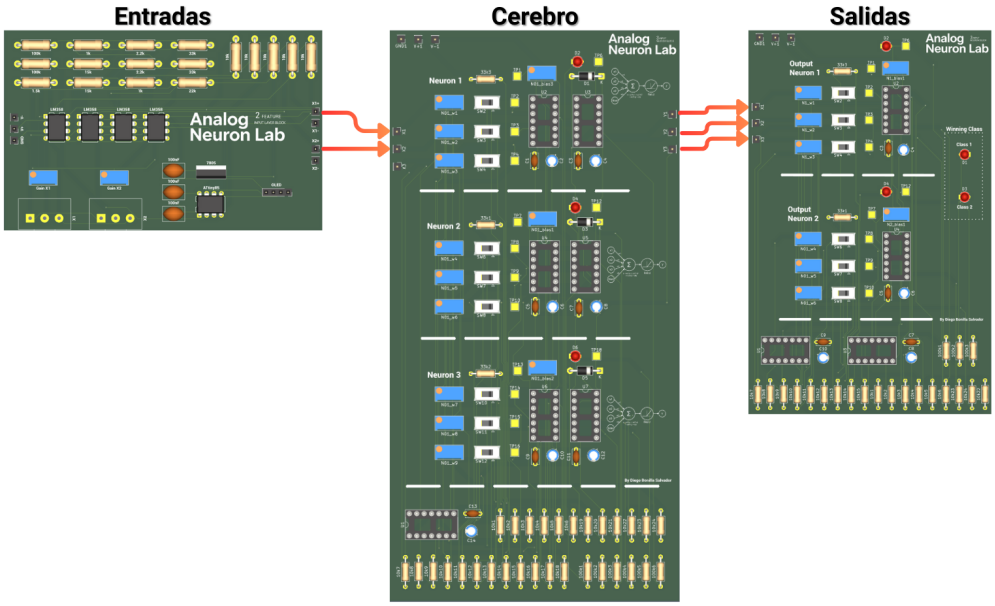


Figura 1.1: Foto general de las partes de la(s) PCB(s) de *Analog Neuron Lab*.

software. La red ocurre en el propio circuito. Eso significa que el comportamiento está determinado por tensiones, corrientes, resistencias y dispositivos reales.

Esto tiene consecuencias prácticas importantes. La primera es que la red es transparente: hay puntos accesibles para medir señales internas, y cada bloque se puede inspeccionar. La segunda es que el resultado no depende solo de una fórmula ideal, sino también del entorno: alimentación, cableado, instrumento de medida y variaciones normales de componentes. Esta guía te enseña a trabajar con todo ello de forma ordenada, sin frustración, y con criterio experimental.

1.2 Estructura física de la placa

La placa está organizada como una cadena de procesamiento que se recorre de izquierda a derecha. Primero se definen las entradas analógicas, después se procesan por bloques que implementan neuronas, y finalmente se obtienen salidas que puedes interpretar como decisión o como nivel de activación. A lo largo del recorrido verás tres tipos de elementos repetirse con consistencia: en primer lugar, conectores y puntos de entrada, pensados para introducir voltajes controlados de manera estable. En segundo lugar, módulos de neurona, donde ocurre la suma ponderada y la no linealidad. En tercer lugar, elementos de interacción y diagnóstico: potenciómetros multivuelta para ajustar parámetros, puntos de test para medir, y LEDs para dar una lectura rápida del estado.

La repetición es intencional. Cuando entiendes un bloque, has entendido el patrón, y podrás recorrer el resto de la placa con seguridad.

1.2.1. Puntos de test

Cada señal relevante del sistema dispone de un punto de test accesible para ajustar los pesos de la red neuronal. La filosofía del laboratorio es siempre la misma: si algo no cuadra, no se adivina. Se mide. Los puntos de test están pensados para que puedas comprobar, paso a paso, dónde aparece una desviación, un error de ajuste o una saturación.

1.2.2. LEDs de estado

Los LEDs no sustituyen al multímetro, pero son una herramienta de lectura rápida. Te permiten ver de un vistazo si una neurona está activa, si una salida está dominando a la otra, o si algo está claramente apagado cuando no debería. En los experimentos se explicará cuándo fiarse de ellos y cuándo no.

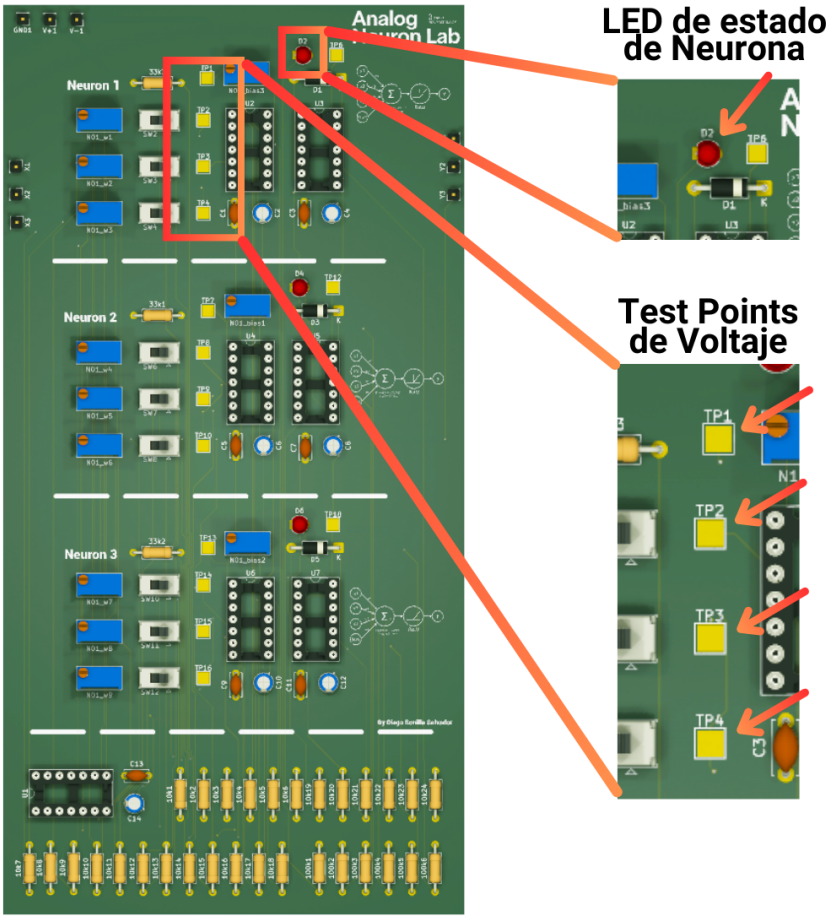


Figura 1.3: Posición de los LEDs de estado de neurona (activa o no activa) y de los Test Points para medir voltajes para ajustar los pesos de cada neurona.

1.3 Conectores de alimentación

La placa requiere una alimentación simétrica para poder representar señales positivas y negativas y para que los amplificadores operacionales trabajen con margen suficiente. En la práctica, esto se consigue con dos fuentes de 12V que forman los dos raíles, uno positivo y uno negativo, compartiendo una referencia común (GND). En otras palabras: el circuito vive entre un raíl superior y un raíl inferior, y todo lo que midas se expresa respecto a masa.

Antes de encender el sistema, acostúmbrate a un hábito que en electrónica evita casi todos los sustos: comprobar alimentación primero, señales después. Con el multímetro en modo voltaje DC, verifica que el raíl positivo está donde debe, que el raíl negativo está donde debe, y que GND es realmente tu referencia.

Masa y referencia

Toda medida de voltaje en este laboratorio se realiza respecto a GND.
Durante un experimento, la punta negra del multímetro debe permanecer siempre conectada a un punto GND de la placa. La punta roja se utiliza para medir los distintos puntos de test. Si cambias la referencia sin querer, tus medidas dejarán de ser comparables y el diagnóstico se vuelve confuso.

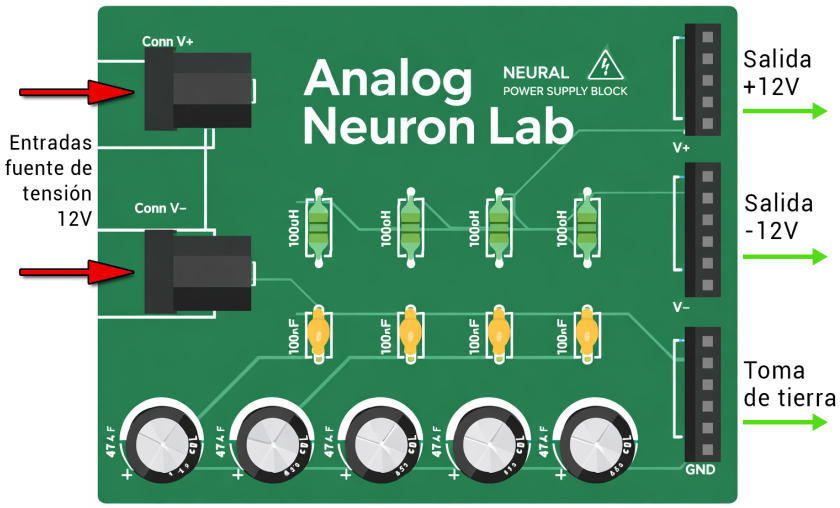


Figura 1.4: Conexión de las dos alimentaciones necesarias para el funcionamiento del sistema.

1.3.1. Módulo de entradas 2D ajustable con potenciómetros (X1, X2)

Además de las entradas externas, el kit incluye una PCB de entradas 2D diseñada para generar dos voltajes analógicos controlables de forma muy intuitiva. La idea es que puedas "mover" un punto en un plano (x_1, x_2) girando dos potenciómetros grandes: uno controla el eje horizontal y el otro el eje vertical. Este módulo es especialmente útil para experimentar con fronteras de decisión en 2D, porque permite barrer el espacio de entrada de manera continua y repetible.

Para hacer esa interacción más visual, la placa incorpora una pequeña pantalla OLED que actúa como un osciloscopio didáctico: dibuja unos ejes simples y marca la posición del punto (x_1, x_2) en tiempo real. De un vistazo puedes ver hacia dónde te estás moviendo y, por tanto, qué región del espacio de entrada estás explorando. La OLED también muestra barras que ayudan a interpretar rápidamente los niveles de señal.

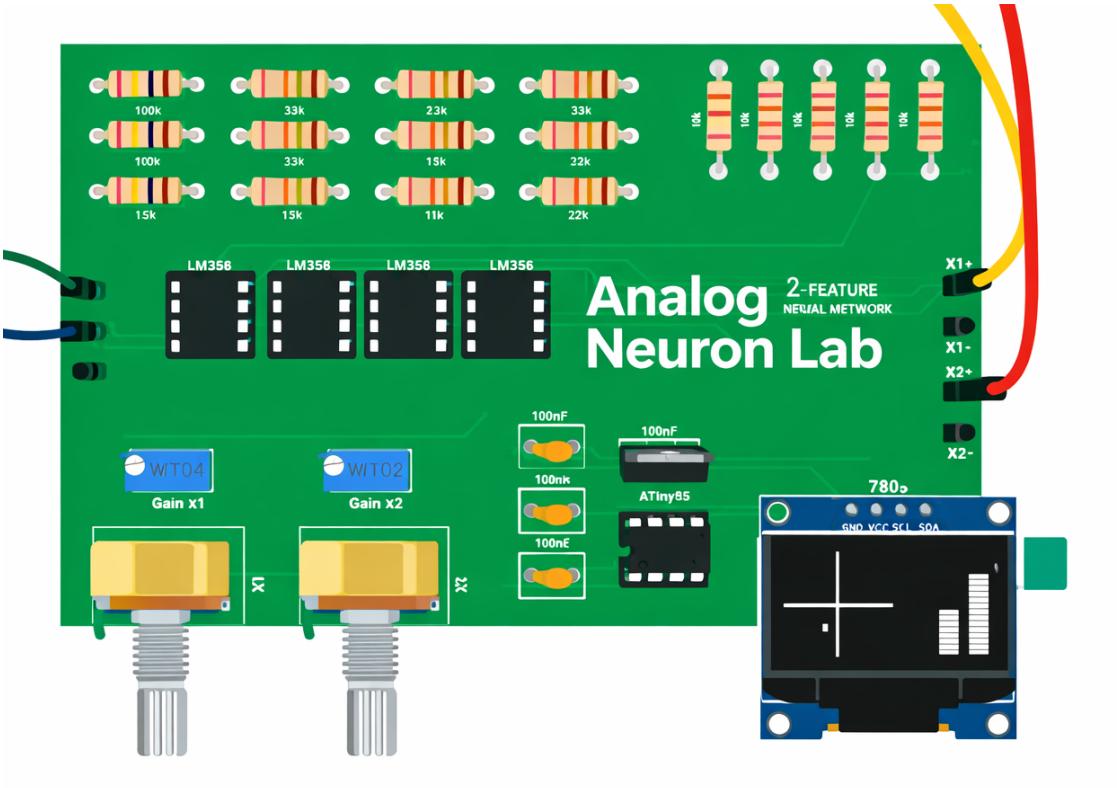


Figura 1.5: PCB de entradas 2D con los dos potenciómetros grandes (control de x_1 y x_2), los dos trimmers azules de calibración y la pantalla OLED.

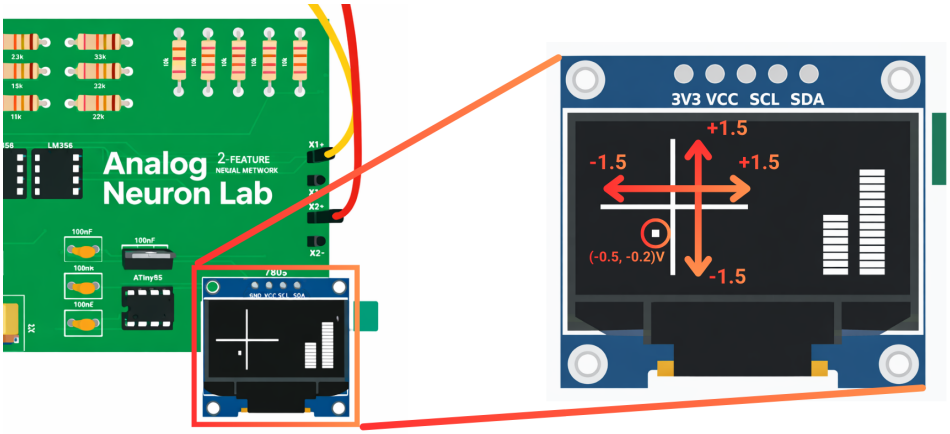


Figura 1.6: Zoom de la OLED mostrando el plano 2D y el punto (x_1, x_2).

Aviso de seguridad: rango permitido

Este módulo está pensado para trabajar exclusivamente con entradas en el rango $-\mathbf{1.5V}$ a $+\mathbf{1.5V}$.
Antes de usar los potenciómetros grandes para explorar el plano 2D, es obligatorio calibrar el rango de salida midiendo $\mathbf{X1}$ y $\mathbf{X2}$ en los pines de la derecha. Si generas un rango mayor, puedes forzar etapas posteriores fuera de especificación y dañar componentes.

Calibración previa de $X1$ y $X2$ (obligatoria). Encima de cada potenciómetro grande hay un trimmer azul. Esos trimmers ajustan el escalado efectivo de la salida para que el recorrido completo del potenciómetro corresponda exactamente al rango seguro $[-1.5V, +1.5V]$. La calibración no se hace "a ojo" ni por posición mecánica: se hace midiendo voltaje.

Con la placa alimentada y el multímetro en modo voltaje DC, conecta la punta negra a un punto GND del sistema y usa la punta roja para medir el pin $\mathbf{X1}$ (y después $\mathbf{X2}$) en el conector de la derecha. A continuación, gira el potenciómetro grande correspondiente hasta un extremo del recorrido (por ejemplo, el máximo). Ajusta su trimmer azul hasta leer aproximadamente $+1.5V$. Después gira el potenciómetro grande al extremo opuesto y verifica que cae aproximadamente a $-1.5V$. Repite una segunda pasada si hace falta, porque los extremos se influyen ligeramente entre sí. Haz exactamente el mismo procedimiento para $X2$ con su trimmer.

Cuando ambos canales estén bien calibrados, ya puedes usar los potenciómetros grandes con tranquilidad: el punto se moverá por el plano 2D dentro del rango seguro, y lo que veas en la OLED corresponderá a entradas reales que no fuerzan el circuito.

Criterio práctico para saber que está bien

Si al llevar cada potenciómetro grande a sus extremos siempre obtienes aproximadamente $+1.5V$ y $-1.5V$ en los pines $X1$ y $X2$, la calibración es correcta. A partir de ahí, cualquier experimento que use esta PCB como generador de entradas parte de una base segura y repetible.

1.4 Potenciómetros de ajuste

La placa incluye potenciómetros multivuelta para ajustar los parámetros internos del sistema. En este laboratorio, un potenciómetro no se trata como una resistencia que hay que "poner a un valor", sino como un control que fija un comportamiento observable. Por eso, el ajuste nunca se hace midiendo resistencia. Se hace midiendo voltaje en el punto de test correspondiente.

Verás dos familias de ajustes. Por un lado, los pesos, que determinan cuánto influye cada entrada en una neurona concreta. Por otro lado, los sesgos (bias), que desplazan el punto a partir del cual una neurona empieza a activarse. En el mundo del software esto se escribe como números en una matriz y un vector. Aquí se traduce a estos potenciómetros, referencias y tensiones.

Una consecuencia útil de este enfoque es que ajustar deja de ser un ritual y se convierte en un procedimiento: eliges una condición de entrada conocida, observas una señal interna objetivo (normalmente una referencia de entrada de 1.5 V), y corriges el potenciómetro hasta alcanzar ese valor dentro de un margen razonable.

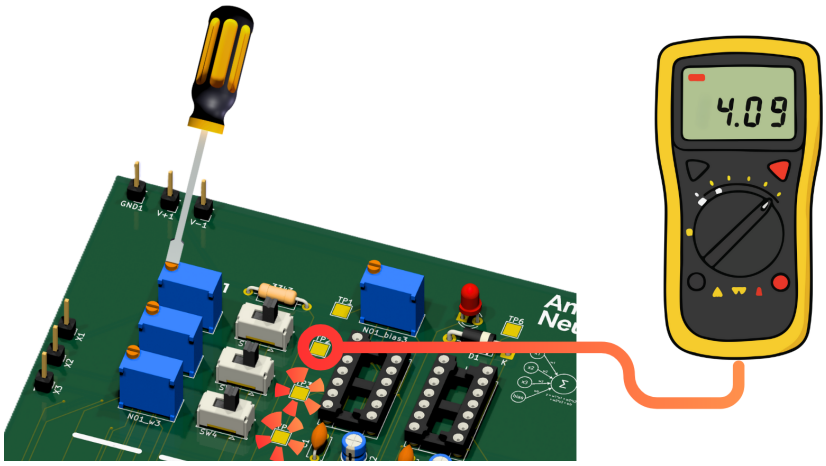


Figura 1.7: Medición de los pesos de la red. En la imagen, la red tiene 3 entradas por lo que hay que ajustar 3 pesos, lo que se traduce a la placa como 3 potenciómetros. Esto lo hacemos poniendo el multímetro en el Test Point que le corresponde (siguiendo la fila) y midiendo el voltaje de DC. Normalmente, y como convención en esta guía, usaremos 1.5 V como entrada para ajustar el voltaje que toca en cada potenciómetro.

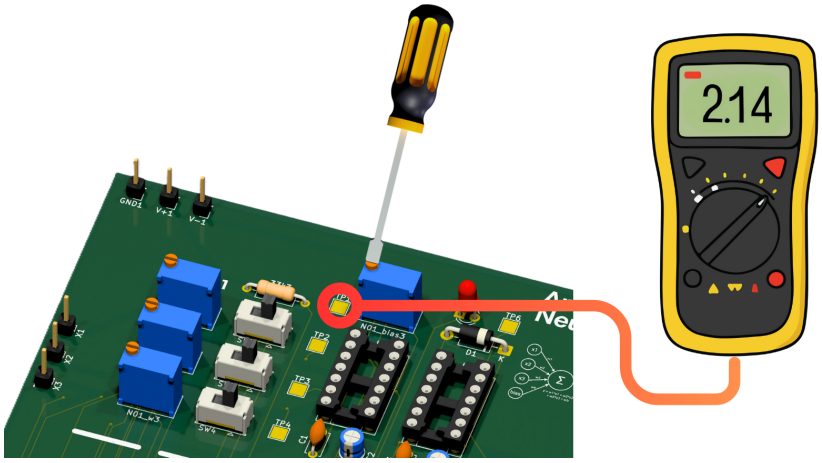


Figura 1.8: De la misma forma que ajustamos los pesos, ajustamos las bias.

Regla de oro del ajuste

Si un ajuste no se puede verificar en un punto de test, es un ajuste ciego. En *Analog Neuron Lab*, todo ajuste importante tiene una medida asociada.

1.5 Cómo está organizada esta guía

Esta guía está pensada como un único camino con tres formas de recorrerlo. El hardware es el mismo para todo el mundo. Los experimentos también. Lo que cambia es el nivel de explicación, el tipo de lenguaje y la profundidad con la que justificamos cada paso.

1.5.1. Parte I: El Aprendiz

El Aprendiz es el itinerario para empezar sin miedo. Aquí se aprende a trabajar con la placa como se aprende a trabajar en un laboratorio real: conectando correctamente, midiendo señales, y entendiendo el comportamiento de manera visual e intuitiva. La matemática se mantiene ligera y siempre ligada a lo que estás viendo en el multímetro y en los LEDs. Si sabes lo que es una función y te suena la idea de “entrada y salida”, puedes seguirlo.

1.5.2. Parte II: El Practicante

El Practicante une el lenguaje de la electrónica con el de las redes neuronales de forma explícita. Aquí aparece la notación de vectores y matrices, la interpretación de capas, y la forma correcta de pensar en una red como composición de transformaciones. El objetivo es que puedas mirar una tabla de pesos, ajustar potenciómetros, y entender qué forma tendrá la frontera de decisión o cómo se transforma un espacio de entrada.

1.5.3. Parte III: El Maestro

El Maestro entra en los detalles que normalmente se omiten: limitaciones físicas, saturación real, sensibilidad a tolerancias, propagación de incertidumbre, repetibilidad y validación experimental. Aquí la red no es solo una función, es un sistema físico. Verás cómo y por qué aparecen desviaciones respecto al ideal, qué parámetros dominan el error, y cómo diseñar procedimientos continuos con señales reales.

En la práctica, puedes leer esta guía de dos maneras. La primera es lineal, como un libro de laboratorio: sigues los capítulos en orden y construyes una base sólida. La segunda es selectiva: si ya dominas una parte, puedes saltar directamente a los experimentos y volver a las secciones teóricas cuando algo te despierte curiosidad o cuando necesites justificar un resultado.

El objetivo del kit es que, sea cual sea tu punto de partida, terminarás con una comprensión más concreta y más honesta de lo que significa una red neuronal. No como metáfora, sino como mecanismo. No solo como ecuación, sino como circuito.

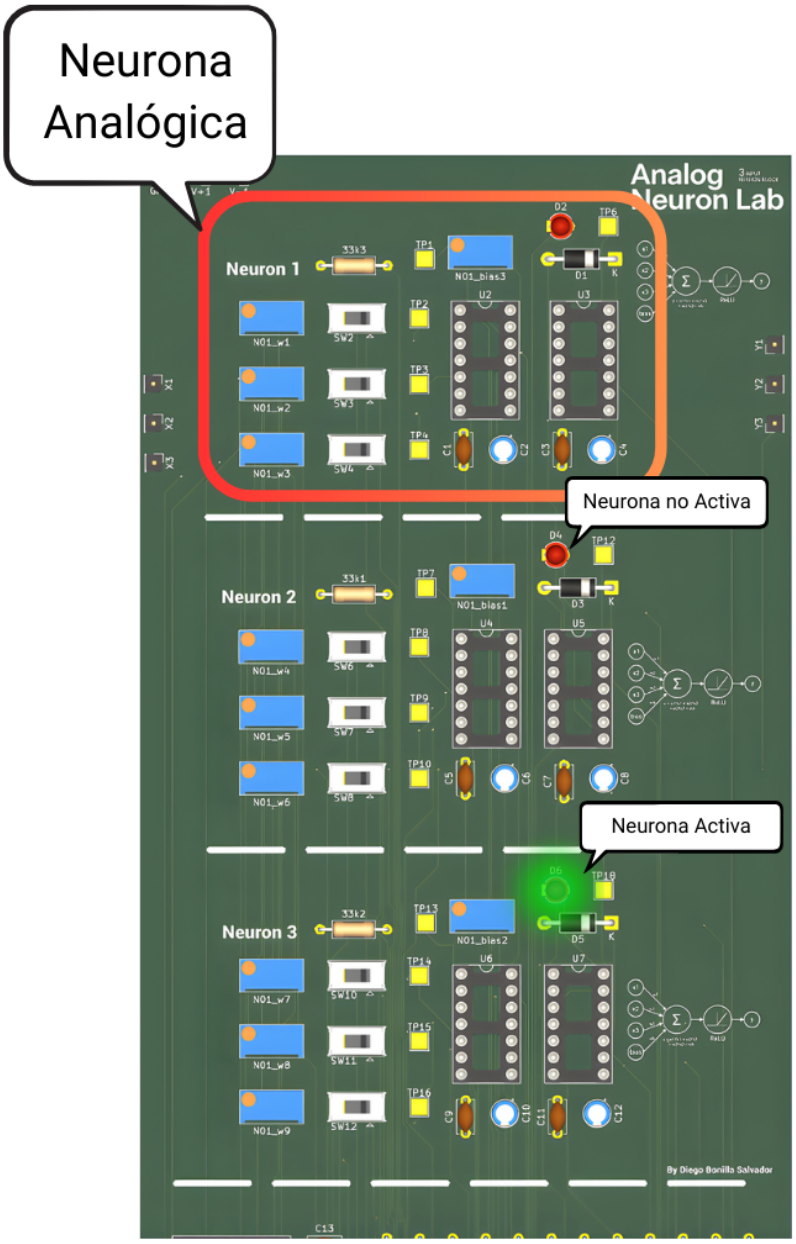


Figura 1.2: Cuatro neuronas formando una capa. Cada LED indica si la neurona está activa.

Parte II

El Aprendiz: Viendo la Luz

Capítulo 2

Bienvenido a la Máquina

Este kit existe por una razón muy concreta: hacer que las redes neuronales dejen de ser algo abstracto y pasen a ser algo que se puede ver, tocar y entender con intuición física.

Normalmente, cuando se habla de inteligencia artificial, todo ocurre dentro de un ordenador. Los datos entran como números, las operaciones se realizan de forma invisible y el resultado aparece en una pantalla. El problema es que, aunque el resultado funcione, es difícil hacerse una idea clara de qué está pasando realmente por dentro.

Esta placa hace exactamente lo mismo que una red neuronal de software, pero de una forma radicalmente distinta: utilizando electricidad real en lugar de código. Los cálculos no se ejecutan en pasos discretos, sino que ocurren de manera continua, impulsados por voltajes, corrientes y componentes electrónicos.

El objetivo de este capítulo es entender qué es esta máquina y por qué tiene sentido aprender redes neuronales de esta forma.

2.1 ¿Qué es esta placa en tu escritorio?

La placa que tienes delante es una red neuronal física. Cada neurona es un pequeño circuito electrónico diseñado para realizar una operación muy concreta: combinar varias entradas, decidir si el resultado es relevante y producir una salida.

En una red neuronal tradicional, estas operaciones se hacen con números dentro de un programa. Aquí se hacen con voltajes. Un voltaje representa una entrada. Otro voltaje representa una salida. Las conexiones entre neuronas están hechas con resistencias y amplificadores, no con líneas de código.

Esto tiene una consecuencia importante: todo sucede en tiempo real. Si cambias una entrada, no hay que esperar a que se ejecute ningún programa. El sistema responde inmediatamente, porque la electricidad fluye de forma continua.

Los LEDs de la placa no están ahí para decorar. Cada LED está conectado a la salida de una neurona y muestra su nivel de activación. Cuando una neurona produce una salida fuerte, el LED brilla con intensidad. Cuando la salida es débil o nula, el LED se atenúa o se apaga.

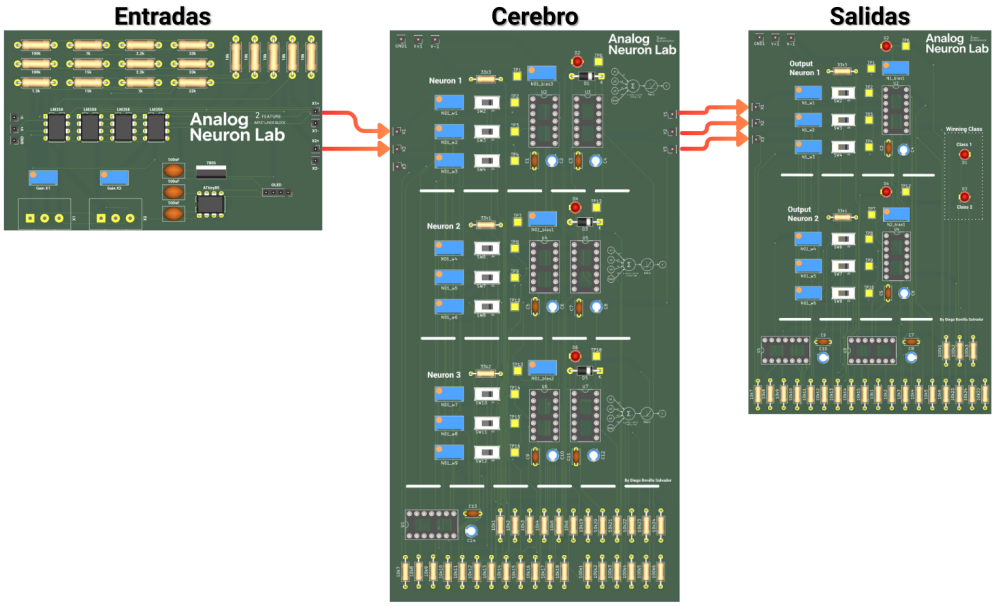


Figura 2.1: Esquemático de la red neuronal analógica: entradas, neuronas visibles y salidas. Cada una es una placa (PCB) diferente.

Para el Aprendiz

¿Sabías que...?

El brillo de un LED indica cuán fuerte es la salida de la neurona. Un LED muy brillante significa que la neurona está "segura" de su decisión. Un LED apagado indica que la neurona no se ha activado.

Lo importante no es memorizar qué componente hace qué, sino entender la idea general: esta máquina toma decisiones utilizando señales físicas, no instrucciones escritas.

2.2 Aprendizaje Automático para Humanos

El aprendizaje automático no consiste en programar reglas complicadas. Consiste en ajustar un sistema para que se comporte mejor después de ver ejemplos.

Un ordenador no aprende como una persona, pero el proceso tiene una lógica similar. Se le muestran situaciones, se observa cómo responde y se corrige su comportamiento poco a poco.

2.2.1. ¿Cómo aprenden los ordenadores?

Supongamos que queremos que una máquina distinga entre dos tipos de objetos, por ejemplo puntos que están dentro de un círculo y puntos que están fuera. No le decimos cómo hacerlo. En su lugar, le mostramos muchos ejemplos ya clasificados.

Al principio, la máquina responde mal. Sus decisiones son casi aleatorias. Pero cada vez que se equivoca, se ajustan ligeramente unos parámetros internos llamados pesos. Esos pesos determinan cómo de importante es cada entrada a la hora de tomar una decisión.

Después de repetir este proceso muchas veces, los pesos acaban tomando valores que producen buenas respuestas para la mayoría de los casos.

En esta placa, esos pesos no son números escondidos en memoria. Son potenciómetros que puedes girar con un destornillador. Ajustar un peso significa literalmente cambiar cómo fluye la electricidad en una neurona.

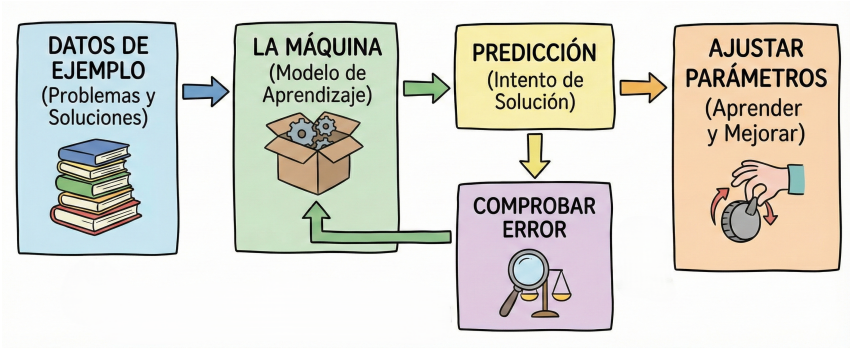


Figura 2.2: El proceso de aprendizaje ajusta automáticamente los parámetros internos de la máquina en función de los errores que comete con respecto a unos datos de ejemplo.

2.2.2. La "Neurona" en resumen

Una neurona artificial no es una copia exacta de una neurona biológica. Es una versión simplificada que realiza una tarea muy concreta.

Recibe varias entradas, combina esa información y decide si su salida debe activarse o no. Cada neurona individual es muy simple, pero al conectar muchas de ellas se pueden resolver problemas complejos.

Puedes pensar en una neurona como un pequeño filtro. Algunas entradas le importan mucho. Otras apenas influyen. El resultado final depende de la combinación de todas ellas.

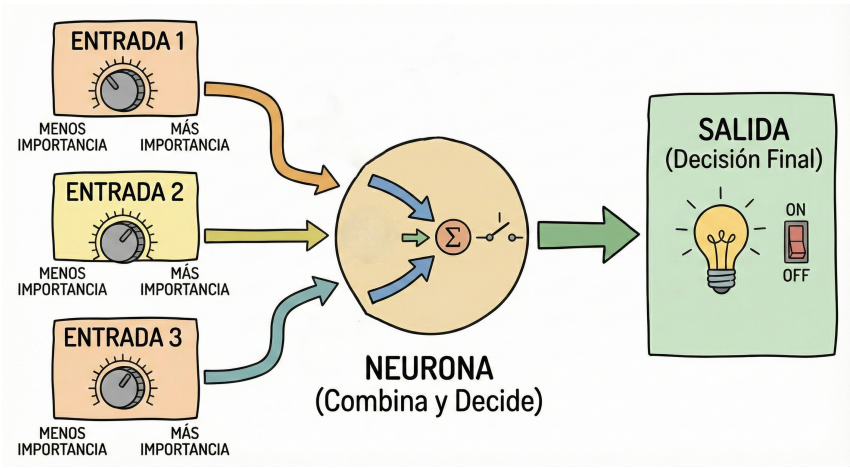


Figura 2.3: Esquema de una neurona artificial: entradas, combinación interna y salida.

En esta placa, cada neurona es un circuito real. Puedes seguir el camino de la señal con el dedo, desde la entrada hasta la salida. No hay magia oculta ni cajas negras.

Para el Aprendiz

Qué deberías quedarte claro

Esta máquina no piensa como una persona. Pero tampoco es un simple circuito fijo. Cambia su comportamiento cuando cambias sus pesos. Eso es lo que significa aprender.

En los siguientes capítulos empezarás a ver cómo se construye una neurona por dentro y cómo, a partir de bloques muy simples, se puede crear un sistema que clasifica, decide y responde de forma sorprendentemente inteligente.

Capítulo 3

Tu Primer Experimento

Hasta ahora hemos hablado de ideas generales: neuronas, aprendizaje y decisiones. En este capítulo toca hacer algo mucho más importante: **encender la máquina y verla reaccionar**.

Antes de entrenar nada, antes de resolver problemas y antes de hablar de inteligencia, vamos a observar cómo se comporta el sistema en su estado más básico. Esto es clave para entender una idea fundamental:

> Ahora mismo, este "cerebro" no sabe hacer nada en concreto.

No está entrenado. No reconoce patrones. No toma decisiones útiles.
Y precisamente por eso es el punto de partida perfecto.

3.1 Conéctalo: El Encendido

Lo primero que vamos a hacer es darle energía a la placa. Sin energía no hay neuronas, no hay decisiones y no hay LEDs.

La placa utiliza **dos alimentaciones de 12 V**. A primera vista puede parecer extraño, pero tiene todo el sentido del mundo y es muy común en electrónica analógica.

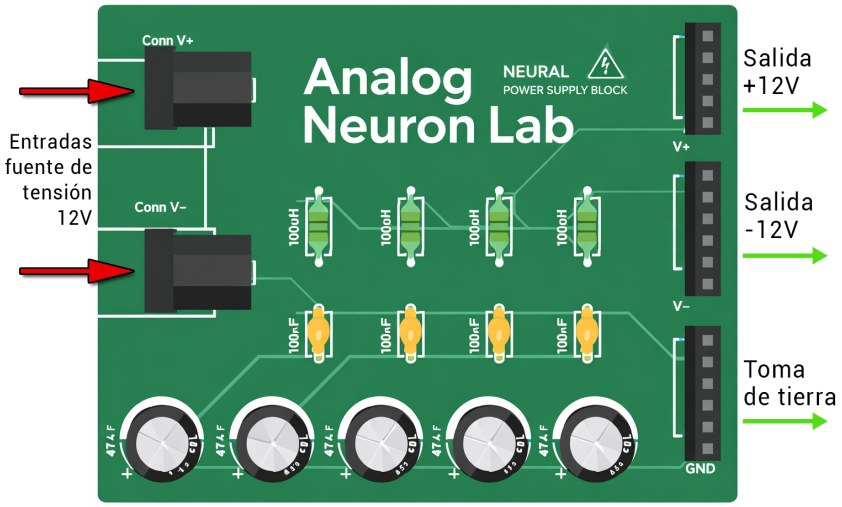


Figura 3.1: Conexión de las dos alimentaciones necesarias para el funcionamiento del sistema.

Conecta los dos adaptadores de 12 V a sus respectivos conectores jack. No hace falta conectar nada más todavía.

Al hacerlo, deberías observar que algunas luces se encienden inmediatamente. Esto indica que la placa está viva y que los bloques principales reciben energía.

Para el Aprendiz

¿Por qué dos alimentaciones de 12 V?

La placa trabaja con voltajes positivos y negativos. Al usar +12 V y -12 V, las neuronas pueden representar números tanto positivos como negativos. Esto es muy importante para poder tomar decisiones más flexibles y realistas.

Si alguna luz no se enciende o algo parece extraño, no pasa nada. En este punto solo queremos confirmar que la placa recibe energía y responde.

3.2 Tutorial de Luces Parpadeantes

Una vez encendida, la placa entra en un estado muy interesante. No está entrenada para ninguna tarea concreta. Sus parámetros internos no han sido ajustados con ningún objetivo.

Es decir: el cerebro está **en bruto**.

3.2.1. El “Hola Mundo” de las Neuronas

Mira la placa grande con atención. Verás varios LEDs verdes distribuidos por el PCB. Cada uno de esos LEDs corresponde a una **neurona**.

No es una metáfora. Cada LED está conectado a un circuito que implementa una neurona artificial completa. Cuando el LED se enciende, esa neurona está activa. Cuando está apagado, esa neurona no está contribuyendo a la decisión final.

En este experimento inicial, estamos usando una pequeña capa de **cuatro neuronas**. Eso puede parecer muy poco, pero es suficiente para empezar a entender qué está pasando.

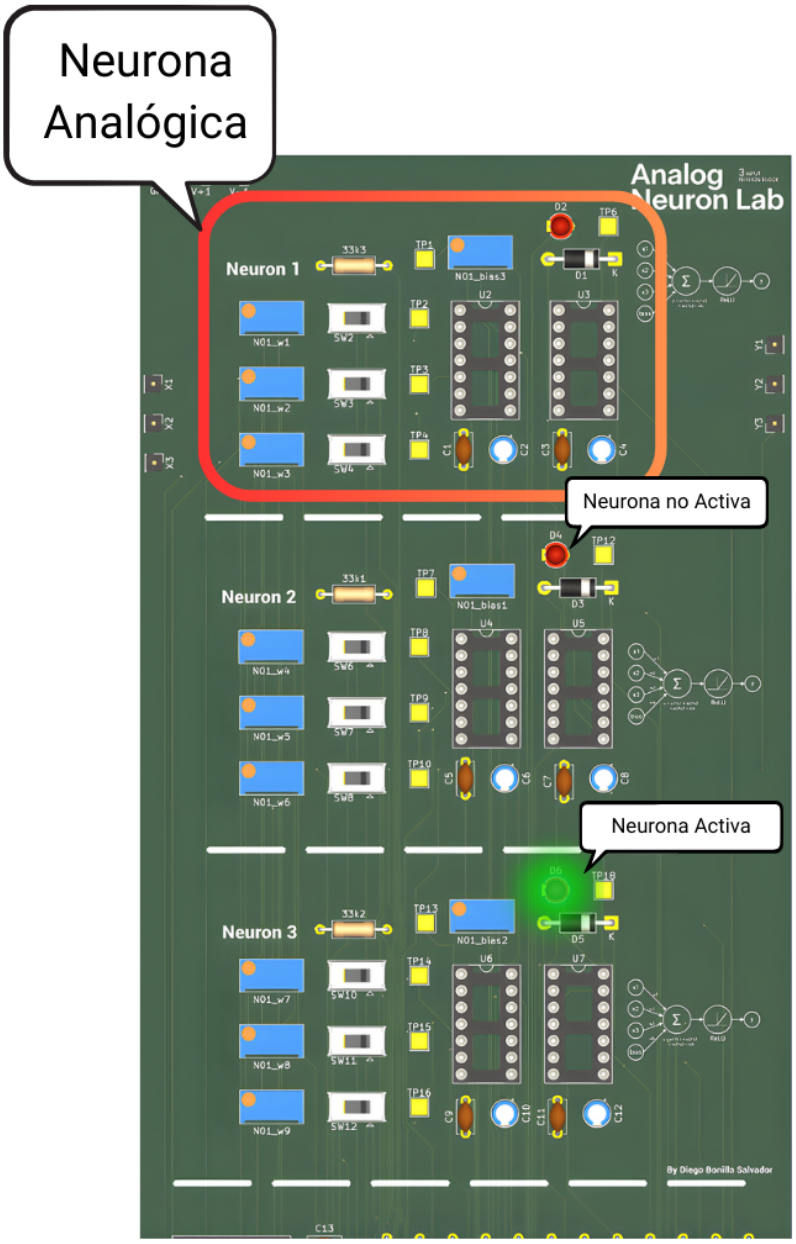


Figura 3.2: Cuatro neuronas formando una capa. Cada LED indica si la neurona está activa.

Para ponerlo en contexto: sistemas como ChatGPT utilizan miles de millones de neuronas. Aquí solo tenemos cuatro, pero eso es una ventaja enorme. Podemos observarlas individualmente, entender su papel y ver cómo influyen en la decisión.

Para el Aprendiz

Idea importante

Más neuronas no significa automáticamente mejor. Significa más complejidad. Esta red pequeña es perfecta para aprender cómo funciona todo desde dentro.

En este punto, las neuronas se activan y desactivan de forma aparentemente caótica. Eso es normal. Todavía no hemos ajustado nada.

3.2.2. Jugando con los Mandos

Ahora viene la parte divertida.

En la placa de entrada verás varios potenciómetros. Estos mandos controlan los **valores de entrada** que recibe el cerebro. Al girarlos, estás cambiando directamente los voltajes que alimentan a las neuronas.

Gira uno de los potenciómetros lentamente y observa qué ocurre en la placa principal. Verás que algunos LEDs empiezan a encenderse, otros se apagan y algunos cambian de intensidad.

Eso es el cerebro reaccionando.

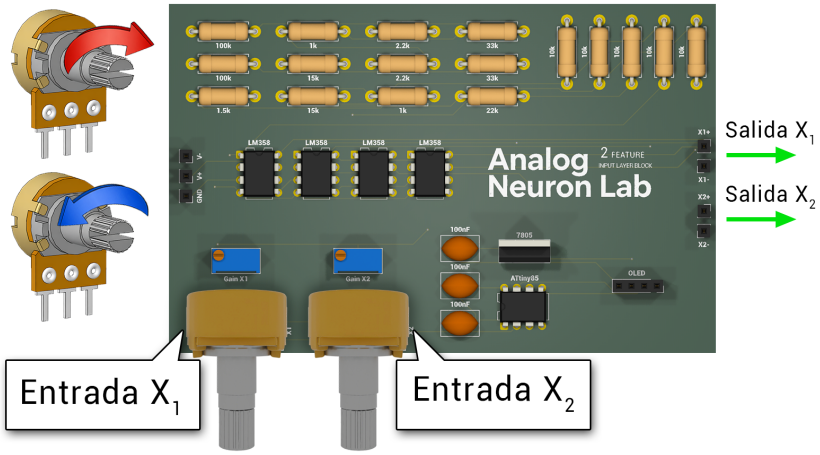


Figura 3.3: Al modificar los potenciómetros de entrada, las neuronas cambian su nivel de activación en tiempo real.

Cada potenciómetro representa una señal que el cerebro está "viendo". Cada neurona decide si esa señal es importante o no según sus parámetros internos. Como esos parámetros todavía no están ajustados, el comportamiento parece aleatorio.

Y eso es exactamente lo que queremos ver ahora.

Para el Aprendiz

Qué está pasando realmente

Las neuronas están tomando decisiones, pero nadie les ha dicho aún qué decisiones son buenas. Por eso algunas se activan sin sentido aparente. Entrenar la red consistirá en ajustar esos parámetros para que las activaciones tengan significado.

Quédate un rato jugando. Gira los mandos. Mira qué neuronas se activan juntas. Observa si hay alguna que casi nunca se enciende o alguna que siempre lo hace.

Todavía no estamos resolviendo ningún problema. Estamos aprendiendo a **escuchar** al cerebro.

Experimento 1: Ajustar una red neuronal real para clasificar círculos

En este experimento vas a convertir una red neuronal analógica en una máquina que clasifica puntos.
Vas a hacer dos cosas:

1. Ajustar los **pesos** y **bias** (con un multímetro y un destornillador).
2. Probar puntos de entrada y ver cómo aparece una **frontera de decisión no lineal**.

El problema: círculo interior vs anillo exterior

La red recibe dos entradas, x e y, que representan la posición de un punto.

- Puntos con x e y pequeños (cerca de 0 V) suelen pertenecer al **círculo interior**.
- Puntos con x e y más grandes (cerca de ± 1.5 V) suelen pertenecer al **anillo exterior**.

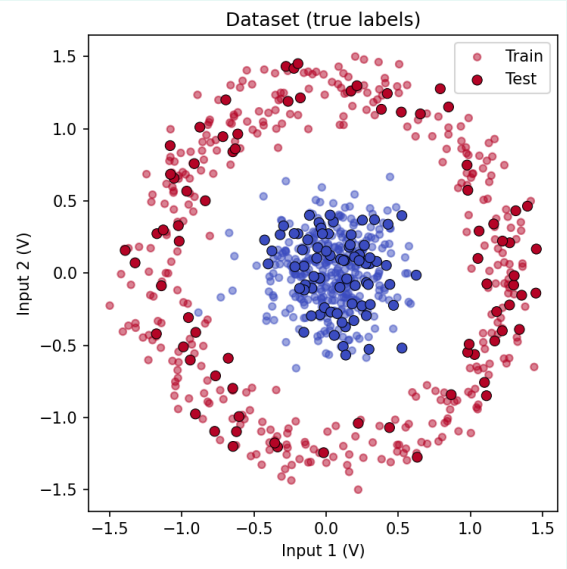


Figura 3.4: Dataset: puntos del centro frente a puntos del anillo.

Arquitectura de la red

Esta red tiene esta forma:

2 entradas → 3 neuronas ocultas → 2 neuronas de salida

- Las **3 neuronas ocultas** son como tres detectores, cada uno se activa en una zona distinta del plano.
- Las **2 neuronas de salida** combinan esas tres señales y deciden qué clase gana.

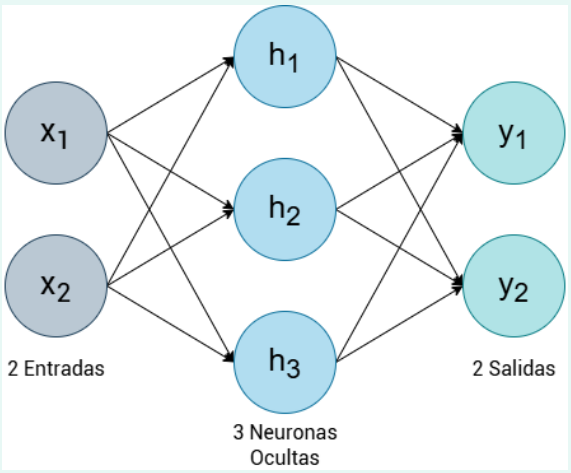
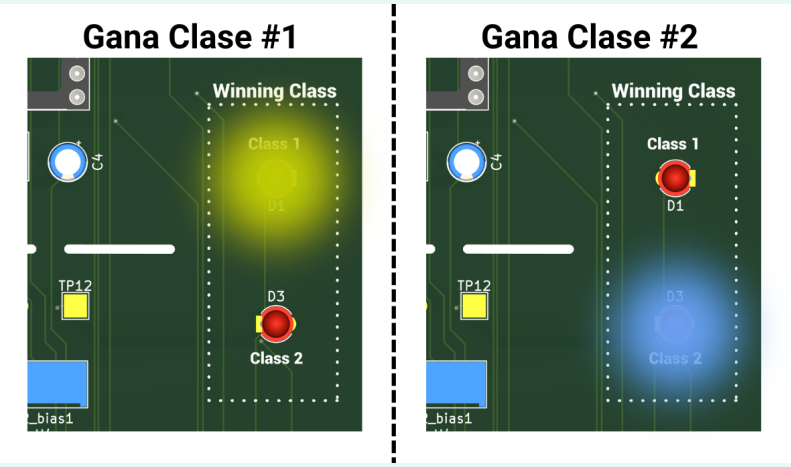


Figura 3.5: Red neuronal completamente conectada (*fully connected (FFN)* o *muli-layer perceptron (MLP)*) como la que tenemos en el kit.

La decisión final es muy simple:

gana la salida con mayor voltaje.



Material y postura de medida

- Multímetro en modo voltaje DC.
- Destornillador para trimmers.

Coloca la punta negra del multímetro en **GND** y no la muevas durante todo el experimento. Con la punta roja vas tocando los **test points** indicados.

Paso 0: comprobar la referencia del inversor

Antes de ajustar nada, comprueba que la referencia de inversión está bien. Aplica $+1.5$ V en la entrada de referencia.

Paso 1: ajustar la capa oculta (3 neuronas)

Vas neurona por neurona. En cada neurona hay:

- Un peso desde x.
- Un peso desde y.
- Un bias (un voltaje fijo propio de esa neurona).
- Dejamos el interruptor del 3er peso **cerrado** ya que solo tenemos 2 entradas en esta capa.

Neurona oculta 1

Ajusta estos voltajes (respecto a GND):

- Peso desde x: $\boxed{+1.006\,\text{V}}$
- Peso desde y: $\boxed{+0.142\,\text{V}}$
- Bias: $\boxed{+0.733\,\text{V}}$

Neurona oculta 2

Ajusta estos voltajes:

- Peso desde x: $\boxed{+1.055\,\text{V}}$
- Peso desde y: $\boxed{+1.387\,\text{V}}$
- Bias: $\boxed{-0.064\,\text{V}}$

Neurona oculta 3

Ajusta estos voltajes:

- Peso desde x: $\boxed{+1.430\,\text{V}}$
- Peso desde y: $\boxed{-1.291\,\text{V}}$
- Bias: $\boxed{-0.129\,\text{V}}$

Para el Aprendiz

Qué estás haciendo en esta capa

Estás construyendo tres detectores distintos. Cada uno responde a una combinación diferente de x e y. Juntos, crean una base para dibujar una frontera compleja.

Paso 2: ajustar la capa de salida (2 neuronas)

Ahora vas a ajustar las dos neuronas finales. Cada una mira las 3 neuronas ocultas y decide una clase. De momento, desconectamos la capa anterior de neuronas y conectamos sus 3 entradas x_1, x_2, x_3 a 1.5V.

Neurona de salida 1: "círculo interior"

Ajusta estos voltajes:

- Peso desde neurona oculta 1: $\boxed{+1.310\,\text{V}}$
- Peso desde neurona oculta 2: $\boxed{-1.407\,\text{V}}$
- Peso desde neurona oculta 3: $\boxed{-1.253\,\text{V}}$
- Bias: $\boxed{-3.244\,\text{V}}$

Neurona de salida 2: "anillo exterior"

Ajusta estos voltajes:

- Peso desde neurona oculta 1: $\boxed{-1.167\,\text{V}}$
- Peso desde neurona oculta 2: $\boxed{+1.424\,\text{V}}$
- Peso desde neurona oculta 3: $\boxed{+1.327\,\text{V}}$
- Bias: $\boxed{-2.302\,\text{V}}$

Paso 3: probar puntos y ver la frontera

Ahora viene lo importante: comprobar que la red separa el plano en dos regiones.

Cómo probar sin cálculos

Piensa en x e y como dos mandos:

- Cerca del centro: x y y cerca de 0 V.
- Anillo: x o y con valores grandes, cerca de ± 1.5 V.

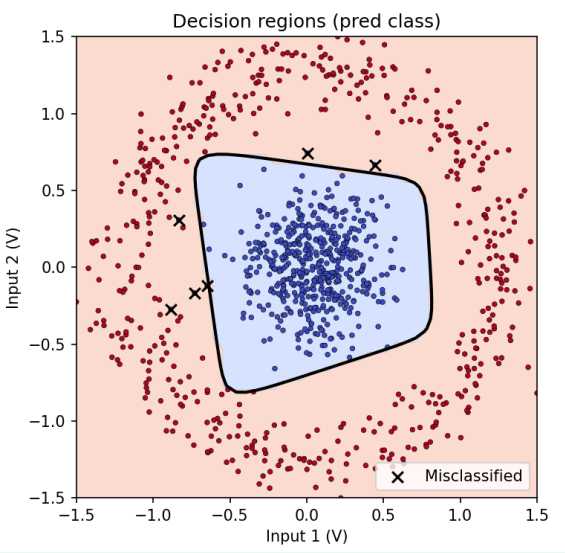


Figura 3.6: Frontera de decisión: la red divide el plano en dos regiones.

Qué acabas de lograr

Has ajustado una red neuronal real con tus manos. Los pesos y los bias han quedado fijados en voltajes concretos. Con eso, la red crea una frontera no lineal en el plano y separa “centro” de “anillo”.

Parte III

El Practicante: Conectando la Lógica

Capítulo 4

Bajo el Capó

Este capítulo explica qué está pasando dentro de la placa, sin quedarse en "esto se enciende y ya". La idea central es muy simple: cada neurona del kit es un **pipeline analógico** de dos etapas con un objetivo claro:

- **Etapa 1 (op-amp 1):** construir una preactivación z que es una suma ponderada de entradas, más un bias.
- **Etapa 2 (op-amp 2):** acondicionar la señal (inversión y margen) y dejarla lista para aplicar la no-linealidad tipo ReLU (que se detalla en la sección siguiente).

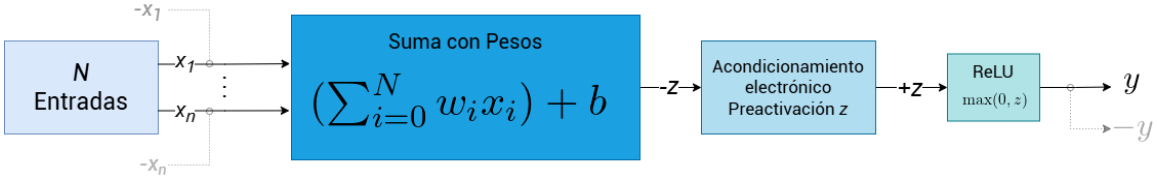


Figura 4.1: Vista general: entradas $\rightarrow$ suma con pesos $\rightarrow$ preactivación z $\rightarrow$ acondicionamiento $\rightarrow$ rectificación tipo ReLU $\rightarrow$ salida y.

4.1 Recordatorio: qué significa "neurona" aquí

Antes de entrar en el detalle del circuito, conviene aclarar qué significa exactamente *"neurona"* en el contexto del aprendizaje automático moderno y cómo se traduce esa idea a un sistema físico real.

4.1.1. La neurona artificial en pocas líneas

En redes neuronales, una neurona no es más que una operación matemática muy concreta. Dadas varias entradas numéricas, la neurona:

1. combina las entradas mediante una suma ponderada,
2. añade un desplazamiento constante (bias),
3. y aplica una función no lineal.

Formalmente, se escribe como

$$y = \sigma\left(\sum_{i=1}^{N} w_i x_i + b\right),$$

donde:

- x_i son las entradas,
- w_i son los pesos que determinan la importancia de cada entrada,
- b es el bias,
- $\sigma(\cdot)$ es la función de activación.

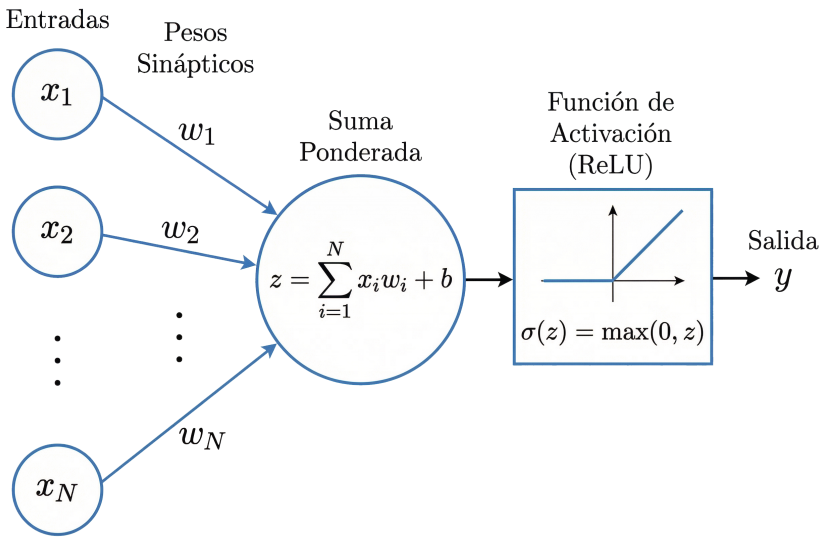


Figura 4.2: Esquema conceptual de una neurona artificial: entradas $x_1, \ldots, x_N$, pesos $w_1, \ldots, w_N$, suma z y activación no lineal $\sigma(z)$.

En esta guía trabajaremos casi exclusivamente con la activación **ReLU** (*Rectified Linear Unit*), definida como

$$\sigma(z) = \max(0, z).$$

Esta función deja pasar valores positivos y anula los negativos, introduciendo una no linealidad esencial.

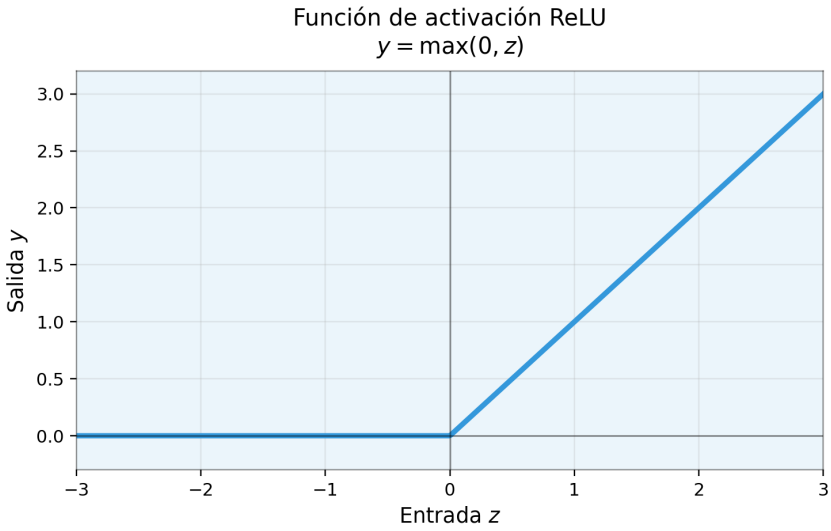


Figura 4.3: Función de activación no lineal ReLU.

4.1.2. De una neurona a una red

Una sola neurona es un clasificador muy limitado: solo puede separar el espacio de entrada mediante una frontera lineal (un plano, una recta, etc.).

La potencia real aparece al conectar muchas neuronas en **capas**:

- una **capa de entrada**, que recibe los datos,
- una o varias **capas ocultas**, que transforman progresivamente la información,
- una **capa de salida**, que produce la decisión final.

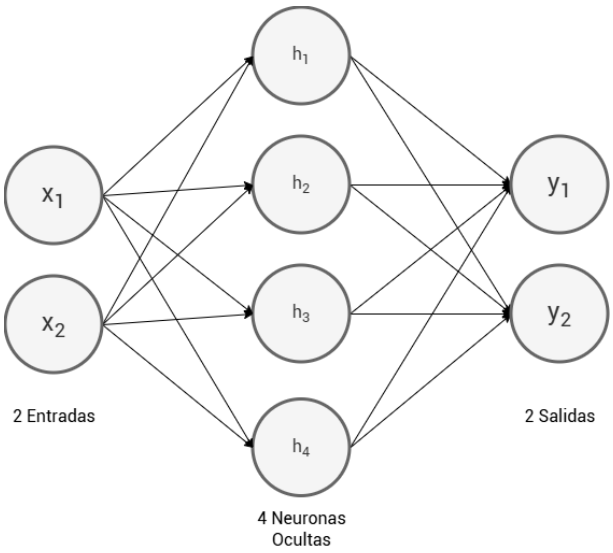


Figura 4.4: Esquema de una red neuronal con 2 entradas, 1 capa oculta de 4 neuronas, y 2 salidas.

Este tipo de arquitectura se conoce como **MLP** (*Multi-Layer Perceptron*). Cada capa aplica una transformación afín seguida de una no linealidad, lo que permite modelar relaciones altamente no lineales entre las variables de entrada.

4.1.3. Correspondencia directa con el hardware

En este kit, la neurona matemática se implementa de forma casi literal usando electrónica analógica. No hay simulación ni aproximación digital: la ecuación ocurre físicamente.

La correspondencia es directa:

- Las **entradas** x_i son voltajes reales.
- Los **pesos** w_i se implementan mediante una red resistiva ajustable (potenciómetros).
- El **bias** es un voltaje constante inyectado al sumador.
- La **activación** σ se implementa físicamente mediante un circuito rectificador tipo ReLU.

Cada neurona de la placa realiza exactamente

$$z = \sum_i w_i x_i + b$$

como un voltaje analógico continuo, y solo después aplica la no linealidad.

Visión Técnica

Cómo leer una neurona del kit

La primera mitad de la neurona construye el valor intermedio $z = \sum_i w_i x_i + b$ como un voltaje físico. La segunda mitad aplica una rectificación que impide valores negativos.
Si entiendes qué voltaje representa z, entiendes el comportamiento completo de la neurona.

4.2 Anatomía de la Placa

La placa está diseñada para que puedas seguir la neurona como si fuera una cadena de montaje. En términos prácticos, una "neurona" en este kit no es una abstracción: es una parte concreta de la PCB, con resistencias (pesos), un bias, dos amplificadores operacionales y un bloque de rectificación.

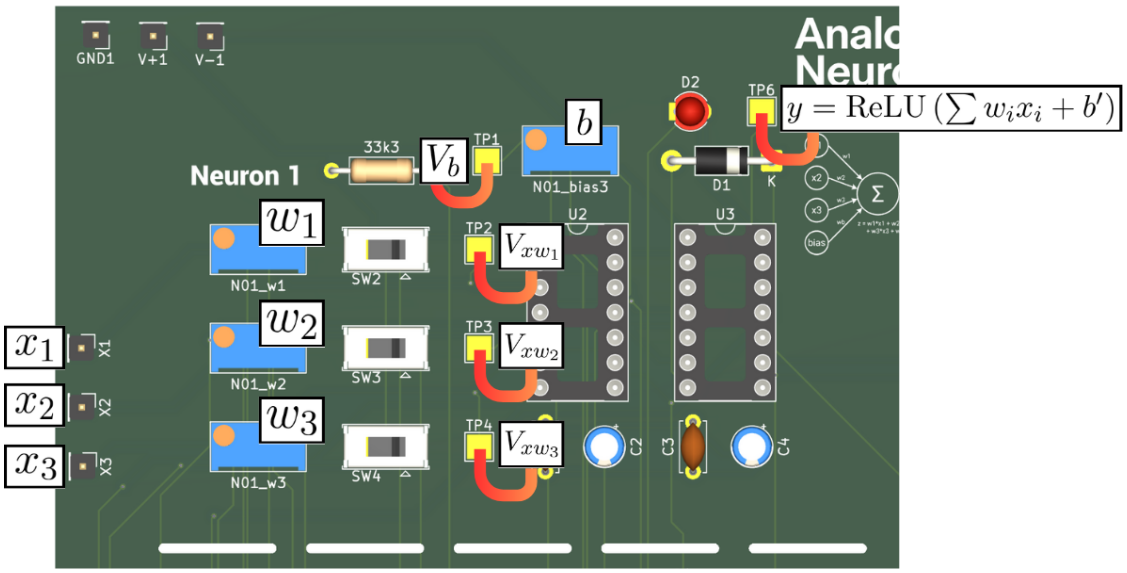


Figura 4.5: Anotaciones en la PCB: entradas x_i, el bloque de pesos w_i, amplificadores operacionales LM358, nodo z, ReLU y salida y.

4.2.1. Qué es un amplificador operacional

Un amplificador operacional, o *op-amp*, es un componente analógico diseñado para hacer una cosa muy bien: **escalar y combinar voltajes de forma controlada y estable**.

En esta placa no se usa como amplificador de audio ni como comparador, sino como una herramienta para implementar operaciones matemáticas simples: sumar, restar y aplicar ganancias.

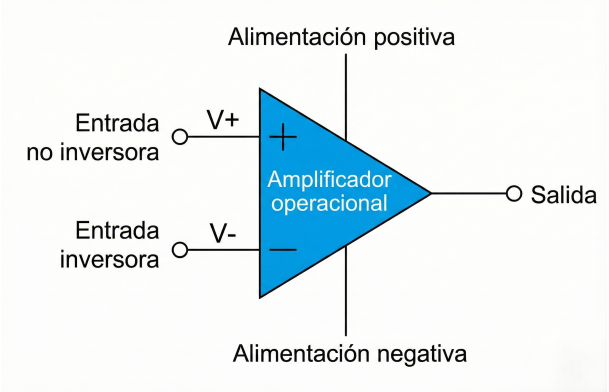


Figura 4.6: Amplificador operacional simple. Tiene entrada inversora (V^-) y no inversora (V^+). La salida nunca puede ser ni mayor ni menor que la alimentación del op-amp.

Amplificar no es "hacer más intenso" sin más

Un op-amp no amplifica un voltaje absoluto, sino la **diferencia** entre sus dos entradas:

$$V_{\text{out}} = A\,(V^+ - V^-),$$

donde A es una ganancia interna enorme. Esto significa que el op-amp odia que sus entradas sean distintas, y hará lo que sea necesario en su salida para forzarlas a ser casi iguales.

Por sí solo, esto sería inestable. La clave está en la realimentación.

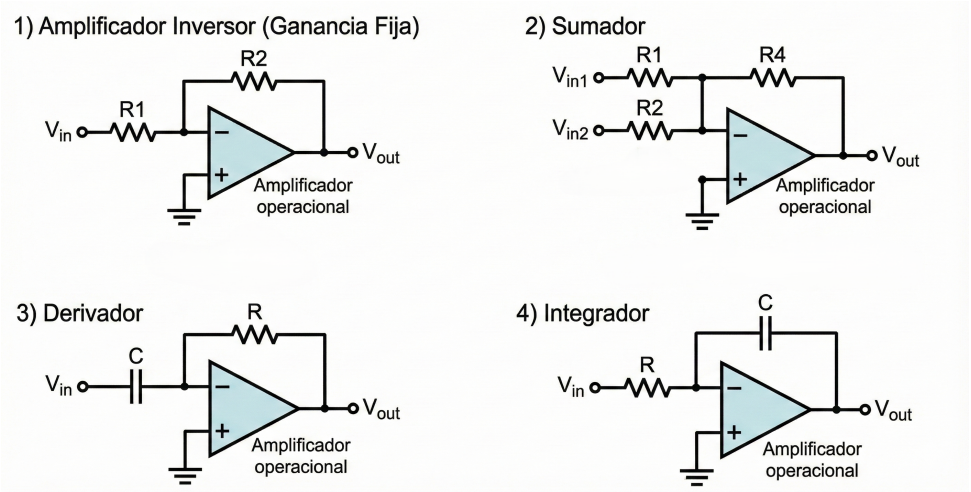


Figura 4.7: La realimentación del operacional da como resultado una serie de comportamientos muy variada.

Realimentación: dejar que el circuito decida la ganancia

Cuando conectamos parte de la salida de vuelta a la entrada inversora, el op-amp entra en un equilibrio automático. La ganancia real del circuito deja de depender del chip y pasa a depender de las resistencias externas.

Gracias a esto, el mismo op-amp puede comportarse como:

- un amplificador de ganancia fija,
- un sumador,

- o un bloque matemático más complejo (derivadas, integrales, ...).

En una neurona, esto es exactamente lo que queremos: controlar cuánto pesa cada entrada.

La idea clave: la masa virtual

Si conectamos la entrada no inversora V^+ a masa, el op-amp ajusta su salida para que la entrada inversora V^- esté también muy cerca de 0 V. Ese punto se llama **masa virtual**.

No es una masa real, pero se comporta como tal en voltaje.

La consecuencia importante es esta:

> Las corrientes que llegan a la masa virtual se suman de forma natural.

El op-amp se limita a generar el voltaje de salida necesario para absorber esa suma.

Por qué esto encaja perfecto con una neurona

En el kit, cada entrada aporta una corriente proporcional a su voltaje y a su resistencia. Todas esas corrientes se suman en la masa virtual, y el op-amp convierte ese resultado en un voltaje de salida.

Eso es exactamente una combinación lineal:

$$z = \sum_i w_i x_i + b.$$

Visión Técnica

En una frase

El op-amp crea un punto donde las corrientes se suman solas y transforma esa suma en un voltaje. Por eso es el componente ideal para construir neuronas analógicas.

4.2.2. El chip LM358: dos op-amps en un encapsulado

El corazón del bloque es el **LM324**. Es un integrado muy común que incluye **cuatro amplificadores operacionales** dentro del mismo chip.

Visión Técnica

Componente Destacado: El Op-Amp (LM324)
Un amplificador operacional no es un “interruptor” como un transistor. En esta placa se usa como una máquina analógica para sumar señales y aplicar ganancias, trabajando con voltajes de manera continua.

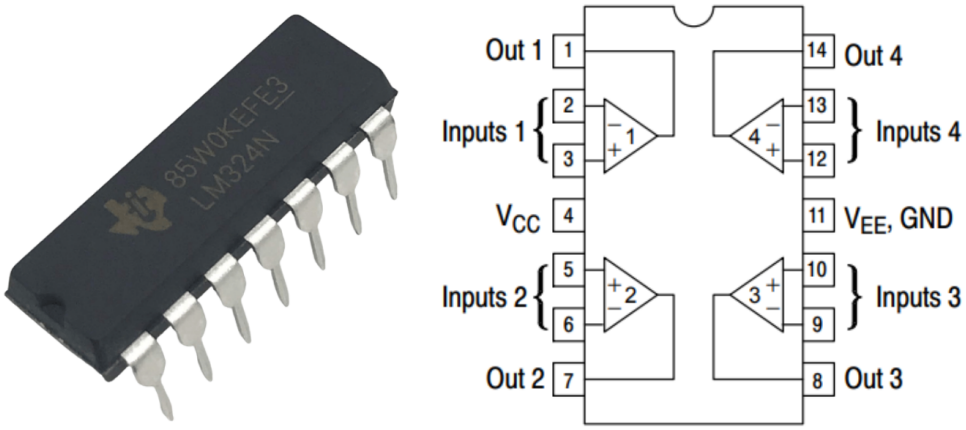


Figura 4.8: LM358: encapsulado y pinout. El integrado contiene dos opamps dentro que comparten alimentación.

4.2.3. Etapa 1: sumador inversor con pesos

La primera etapa recoge N entradas (voltajes) y las combina en un solo voltaje z. El montaje base es el sumador inversor:

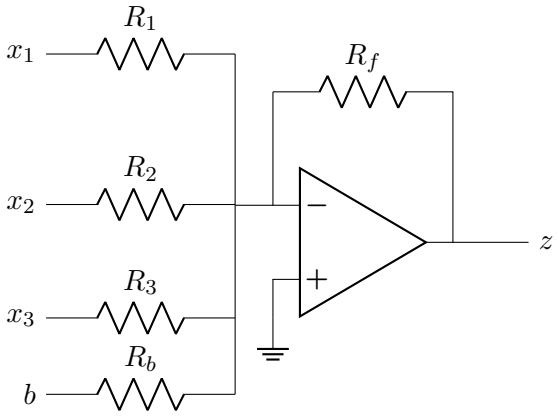


Figura 4.9: Sumador inversor: varias entradas x_i (y el bias b) se convierten en corrientes hacia el nodo sumador. La realimentación R_f fija la escala global y la salida z representa la preactivación.

La ecuación estándar del sumador inversor es:

$$z = V_{\text{out}} = -R_f \left(\frac{x_1}{R_1} + \frac{x_2}{R_2} + \cdots + \frac{x_N}{R_N} + \frac{b}{R_b} \right).$$

Aquí aparece la interpretación directa:

$$z = -\sum_{i=1}^{N} \left(\frac{R_f}{R_i} \right) x_i \; - \; \left(\frac{R_f}{R_b} \right) b.$$

Es decir, el peso efectivo de cada entrada es:

$$w_i \equiv \frac{R_f}{R_i}.$$

En el kit, R_f y parte de R_i se eligen para que la **ganancia global** de esta etapa sea aproximadamente constante y fácil de razonar. Por diseño, trabajamos con una magnitud de ganancia fija:

$$G \approx 3.3,$$

y el signo depende de si la etapa es inversora (signo negativo) o si más adelante se vuelve a invertir. A nivel de intuición, puedes pensar la preactivación como:

$$z \approx -G\left(\sum_i \tilde{w}_i x_i + \tilde{b}\right),$$

donde $\tilde{w}_i$ y $\tilde{b}$ son los parámetros ajustables con los potenciómetros, normalizados para ser cómodos.

Visión Técnica

Qué significa "peso" en hardware:
En software un peso es un número. En esta placa un peso es, de forma efectiva, un cociente de resistencias. Girar un potenciómetro cambia una resistencia equivalente, y por tanto cambia $w_i = R_f/R_i$.

Cómo se implementan pesos positivos y negativos

En una red neuronal real necesitas pesos con signo. En un circuito, una resistencia no tiene "signo", así que el kit implementa el signo con arquitectura:

- Cada entrada x_i dispone de dos ramas conceptuales: una que contribuye en el sentido "positivo" y otra en el "negativo".
- La configuración del potenciómetro decide cuánto contribuye cada rama, resultando en un peso neto con signo.

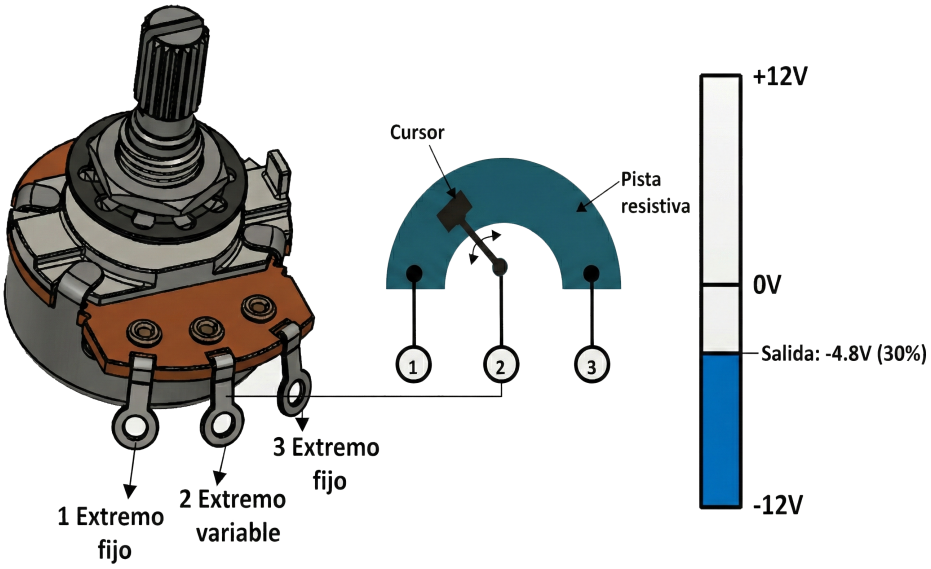


Figura 4.10: Potenciómetro como selector de contribución positiva y negativa de una entrada. En este ejemplo, el potenciómetro se encuentra en el 30 % de su valor total entre $\pm$12V o $\simeq -4.8$V. Sigue la ecuación de $V_2 = \alpha * V_1 + (1-\alpha) * V_3$, en este caso $V_1 = 12$V y $V_3 = -12$V.

No hace falta memorizar el detalle exacto en este punto. Lo importante para entender el bloque es que, al final, el circuito se comporta como una suma ponderada con signo.

4.2.4. Etapa 2: inversión y preparación para la ReLU

Tras la etapa de suma, tenemos z, que ya es "la neurona sin activación". La segunda mitad del LM358 se usa para:

- **Recolocar el signo** (si interesa que el sentido del peso sea intuitivo o que la activación funcione en la dirección correcta).
- **Acondicionar la señal** para que el siguiente bloque trabaje limpio: baja impedancia de salida, margen, estabilidad.
- **Alimentar el rectificador** que implementa la ReLU aproximada con diodo y R_{pull} a GND.

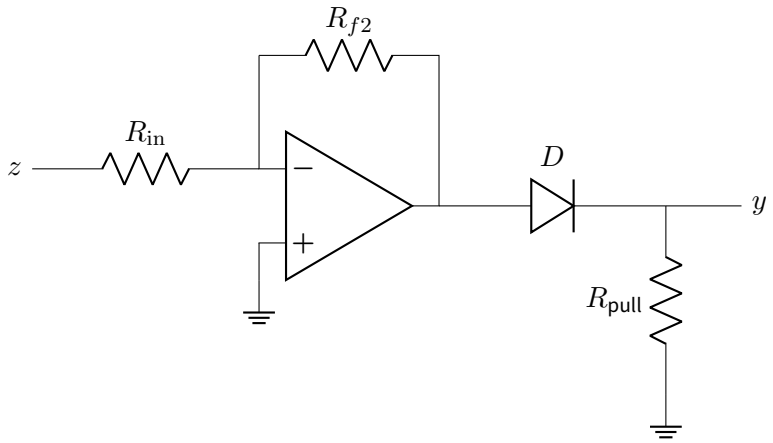


Figura 4.11: Segunda etapa: acondicionamiento (inversión y ganancia) seguido del rectificador tipo ReLU. El diodo define la conducción en positivo y R_{pull} fuerza $y \approx 0$ cuando el diodo está cortado.

Visión Técnica

Lectura rápida del pipeline:
La etapa 1 construye el "cálculo" z. La etapa 2 deja la señal lista para que el diodo haga de frontera y convierta z en una salida tipo ReLU y. En la siguiente sección se analiza esa frontera con detalle.

Experimento (Practicante): Sigue la señal de una neurona con un multímetro

Objetivo: verificar que tu neurona está calculando z y que la activación rectifica después.

Qué necesitas: un multímetro en DC y la placa encendida.

Pasos:

1. Fija las entradas x_1, x_2 a valores sencillos (por ejemplo $x_1 = 0.5\,\text{V}, x_2 = -0.5\,\text{V}$).
2. Localiza el nodo z (salida de la etapa 1) y mide su voltaje.
3. Localiza el nodo y (salida después del rectificador) y mide su voltaje.
4. Cambia un único peso (un potenciómetro) y repite medidas. Deberías ver cómo cambia z de forma continua, y cómo y se mantiene cerca de 0 cuando z es negativo.

Qué observar: si z se mueve pero y no, el problema suele estar en la etapa de rectificación o en la referencia a masa del nodo de salida.

4.3 El Circuito Rectificador (ReLU)

Una red neuronal analógica necesita, igual que una digital, una pieza esencial para ser útil: una **no-linealidad**. En el kit, esa no-linealidad se implementa con un rectificador basado en diodo que convierte la preactivación z en una salida y con comportamiento tipo ReLU:

$$y \approx \max(0, z).$$

La preactivación z es la señal que sale del bloque de suma y ganancia (op-amp), típicamente una combinación afín de entradas y bias:

$$z = w_1 x_1 + w_2 x_2 + \cdots + b.$$

El rectificador actúa a continuación como una frontera física: valores negativos se apagan y valores positivos pasan, con matices realistas debidos al diodo y a la carga.

Visión Técnica

Diodo $\simeq$ ReLU:
La salida del op-amp alimenta un diodo en directa hacia el nodo y; una resistencia R_{pull} a masa descarga y referencia y cuando el diodo está cortado. El resultado es una activación tipo ReLU en hardware.

4.3.1. Por qué la linealidad es aburrida

Si todas las capas de una red fueran lineales ($y = z$), la red entera se podría reemplazar por una sola capa. No es una magia, es álgebra.

Visión Técnica

Transformación afín:
Una transformación afín es una función geométrica que preserva líneas rectas y paralelismo, combinando una transformación lineal (como rotación, escalado o corte) con una traslación (desplazamiento). Es decir, toma un punto y lo mueve manteniendo su "forma" lineal, cambiando su posición, tamaño o inclinación sin curvarlo.

Considera una red de L capas donde cada capa hace una transformación afín:

$$h_k = W_k h_{k-1} + b_k, \qquad k = 1, \ldots, L, \quad h_0 = x.$$

Expandiendo iterativamente:

$$h_1 = W_1 x + b_1,$$

$$h_2 = W_2 h_1 + b_2 = W_2(W_1 x + b_1) + b_2 = (W_2 W_1)x + (W_2 b_1 + b_2),$$

y así sucesivamente. Al final siempre aparece la misma estructura:

$$h_L = \underbrace{(W_L W_{L-1} \cdots W_1)}_{W_{\text{eq}}} x + \underbrace{\left(W_L \cdots W_2 b_1 + \cdots + W_L b_{L-1} + b_L\right)}_{b_{\text{eq}}}.$$

Es decir:

$$h_L = W_{\text{eq}} x + b_{\text{eq}}.$$

Da igual si son 10 capas o 1000 capas, o si hay un millón de neuronas: si no hay activaciones no lineales entre medias, la red no gana capacidad de representación. En clasificación en 2D, por ejemplo, eso significa fronteras de decisión que son siempre rectas (o hiperplanos en dimensiones mayores).

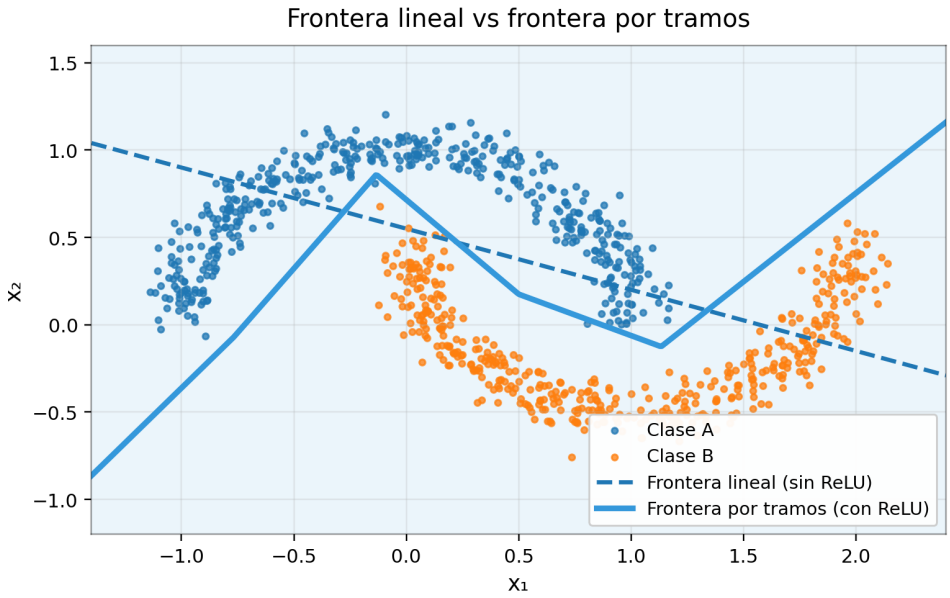


Figura 4.12: Una red puramente lineal produce fronteras lineales. Con ReLU, la frontera se vuelve por tramos (piecewise linear), y puede aproximar curvas complejas sumando segmentos.

La ReLU resuelve este bloqueo de manera elegante porque introduce una no-linealidad simple y estable. Además, tiene una propiedad práctica: es **lineal por tramos**. Para $z > 0$ el comportamiento es aproximadamente lineal, y para $z < 0$ la salida se apaga. Esa combinación es suficiente para que varias capas encadenadas construyan funciones mucho más ricas que una sola transformación afín.

Visión Técnica

Idea clave:
La potencia de una red profunda no viene de apilar multiplicaciones por matrices sin más. Viene de intercalar una operación no lineal entre capas. La ReLU es el ejemplo moderno más común porque es simple, funciona bien y en hardware es muy natural de implementar.

4.3.2. El Diodo como frontera de decisión

Antes de hablar de ReLU, conviene entender qué hace realmente un diodo, sin fórmulas innecesarias.

Un diodo es un componente **asimétrico**: no trata igual a los voltajes positivos y negativos. Su comportamiento básico puede resumirse así:

- En una dirección, el diodo **bloquea** la corriente.
- En la dirección contraria, el diodo **conduce**, pero solo cuando el voltaje es suficiente.

Esa asimetría convierte al diodo en una especie de *válvula eléctrica*.

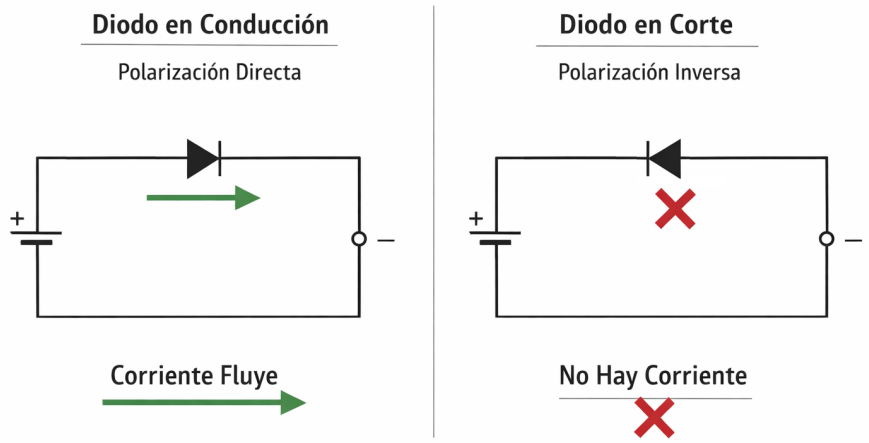


Figura 4.13: El diodo actúa como una válvula eléctrica: en polarización directa permite el paso de corriente, y en polarización inversa bloquea completamente la conducción.

Conducción directa y bloqueo

Cuando el diodo está polarizado en directa (el ánodo a mayor voltaje que el cátodo), empieza a conducir. Sin embargo, no lo hace inmediatamente: necesita superar una tensión mínima llamada **tensión directa** V_F, que en diodos de silicio típicos está alrededor de 0.6–$0.7\,\mathrm{V}$.

Si el voltaje aplicado es menor que ese umbral, el diodo prácticamente no conduce. Si es mayor, la corriente aumenta rápidamente.

En polarización inversa ocurre lo contrario: el diodo bloquea la corriente y se comporta, a efectos prácticos, como un circuito abierto.

Visión Técnica

Idea clave

Un diodo no deja pasar voltajes negativos y solo deja pasar voltajes positivos a partir de cierto umbral.

Del diodo a una frontera de decisión

Esta propiedad es exactamente lo que se necesita para implementar una función de activación tipo ReLU.

Recuerda que la ReLU ideal es:

$$\mathrm{ReLU}(z) = \mathrm{máx}(0, z).$$

Eso significa:

- si $z < 0$, la salida debe ser 0,
- si $z > 0$, la salida debe crecer con z.

Un diodo hace algo muy parecido de forma natural:

- para z negativo, el diodo está bloqueado,
- para z positivo y suficientemente grande, el diodo conduce.

La transición entre ambos regímenes define una **frontera física** entre apagado y activo.

El papel de la resistencia R_{pull}

Cuando el diodo está bloqueado, el nodo de salida quedaría flotante. Eso no es deseable, porque un nodo flotante puede retener carga o captar ruido.

La resistencia R_{pull} conecta la salida a masa y cumple una función muy simple:

- fuerza la salida a $0\,\text{V}$ cuando el diodo no conduce,
- define de forma clara el estado "apagado" de la neurona.

Cuando el diodo conduce, su resistencia efectiva es mucho menor que R_{pull}, por lo que la salida sigue al voltaje de entrada z.

ReLU física

El resultado del conjunto diodo + R_{pull} es una activación que se comporta como una ReLU realista:

$$y \approx \begin{cases} 0, & z \leq 0 \\ z - V_F, & z > 0 \end{cases}$$

No es una ReLU matemática perfecta, pero sí una frontera clara entre valores negativos y positivos, con un comportamiento continuo y estable.

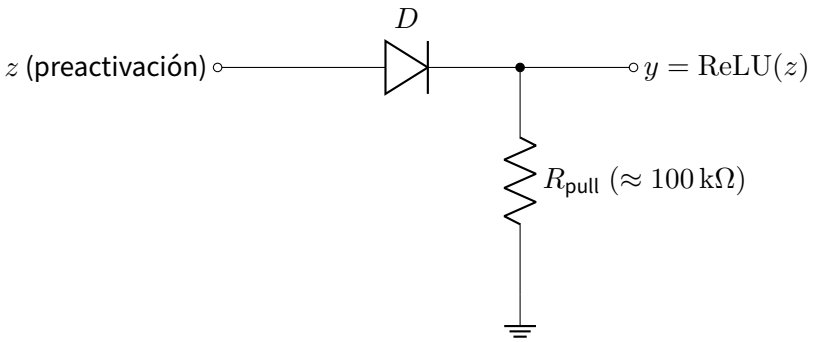


Figura 4.14: Implementación física de una activación tipo ReLU usando un diodo como frontera entre apagado y activo.

Visión Técnica

Lectura directa del circuito

El diodo decide si la neurona está activa o no. La resistencia define qué significa exactamente "cero". Juntos convierten un voltaje continuo en una decisión no lineal.

Modelo ideal (intuición)

Si el diodo fuese ideal, el circuito implementaría exactamente:

$$y = \begin{cases} 0, & z \leq 0, \\ z, & z > 0. \end{cases}$$

Cuando $z \leq 0$, el diodo no conduce y el nodo y queda forzado a masa por R_{pull}. Cuando $z > 0$, el diodo conduce y y sigue a z.

Qué ocurre en la realidad

Un diodo real no se comporta como un interruptor perfecto. Una aproximación estándar para su curva es la ecuación de Shockley:

$$I_D = I_S \left(e^{\frac{V_D}{n V_T}} - 1 \right),$$

donde I_S es la corriente de saturación, n es el factor de idealidad y V_T es el voltaje térmico (del orden de $26\,\mathrm{mV}$ a temperatura ambiente). Lo importante aquí no es memorizar la fórmula, sino entender su consecuencia:

- La conducción no aparece de golpe. Aumenta de manera exponencial con V_D.
- La "caída directa" V_F no es un número fijo. Depende de la corriente, del tipo de diodo y de la temperatura.

Por eso, el comportamiento real se parece más a una ReLU con umbral efectivo:

$$y \approx \begin{cases} 0, & z \le 0, \\ z - V_F(I), & z > V_F. \end{cases}$$

En la zona cercana a cero hay una transición suave: el diodo conduce poco, R_{pull} sigue tirando de y hacia 0, y el resultado es una "rodilla" en lugar de un corte perfecto.

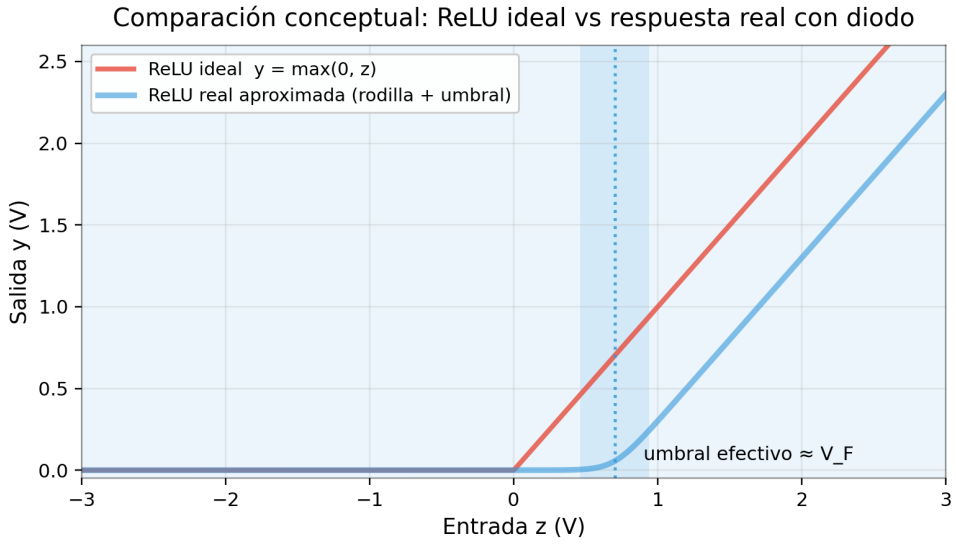


Figura 4.15: Comparación conceptual: ReLU ideal $\max(0, z)$ frente a respuesta real del rectificador con diodo (umbral efectivo y transición suave).

Por qué existe $R_{\text{pull}} \approx 100\,\mathrm{k\Omega}$

La resistencia R_{pull} no está de decoración. Resuelve problemas prácticos:

1. **Define el cero.** Cuando el diodo no conduce, y queda referenciado a GND y no se queda flotando.
2. **Descarga el nodo.** Sin esa descarga, pequeñas capacitancias del nodo pueden retener carga y producir salidas que tardan en volver a 0.

3. **Mejora estabilidad.** Un nodo flotante es sensible al ruido y a la carga del siguiente bloque.

Visión Técnica

La ReLU del kit no es una “función” en abstracto. Es un rectificador real con diodo y pull-down. Entender dónde aparece el umbral, por qué existe la rodilla y qué papel juega R_{pull} te permite diseñar mejores experimentos y entrenar modelos que luego funcionan en la placa sin sorpresas.

Laboratorio 1: Descenso por gradiente manual)

En este laboratorio vas a entrenar una red neuronal analógica extremadamente pequeña usando **descenso por gradiente** de forma manual. La idea no es "ajustar a ojo", sino seguir el mismo ciclo que haría un optimizador: **forward** $\rightarrow$ **pérdida** $\rightarrow$ **gradiente** $\rightarrow$ **actualización**.

Objetivo

Implementar una transformación lineal 2D (dos entradas, dos salidas) y observar cómo la pérdida disminuye iteración a iteración, a pesar de:

- ruido y offsets analógicos,
- resolución limitada de trimmers,
- pequeñas no linealidades internas.

Arquitectura del experimento

Vas a usar únicamente la **capa de salida** con dos neuronas. No hay capa oculta.

2 entradas $\rightarrow$ 2 neuronas de salida (modo lineal)

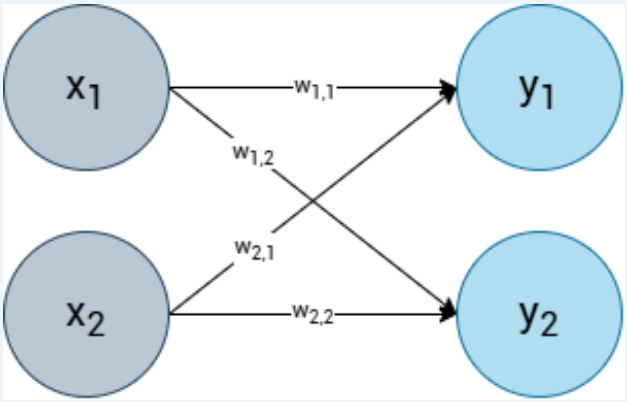


Figura 4.16: Arquitectura del laboratorio: dos entradas y dos neuronas de salida. Cada neurona implementa una combinación lineal.

Modelo matemático (lo que vamos a entrenar)

Para cada neurona de salida $k \in \{1, 2\}$, definimos:

$$\hat{u}_k = \alpha_{k1} x_1 + \alpha_{k2} x_2 + b_k \qquad \text{con} \qquad \alpha_{k1}, \alpha_{k2} \in [-1, 1]$$

En la placa, lo que tú mides como preactivación (test point antes del diodo) se aproxima por:

$$z_k \approx G \cdot \hat{u}_k \qquad \text{donde típicamente} \qquad G \approx 3.3$$

En este laboratorio **trabajaremos con** z_k (salida pre-diodo) para mantener el comportamiento lo más lineal posible, y además mantendremos z_k en un rango moderado para evitar saturaciones.

Visión Técnica

Por qué no usamos la salida con diodo

El diodo implementa una ReLU suave con un offset aproximado de $V_d \approx 0.6$ V y además existe saturación del operacional. Para no mezclar estos efectos con el descenso por gradiente, aquí medimos y entrenamos usando el nodo preactivación z, donde el comportamiento es mucho más cercano a una función lineal. El esquema de ponderación por interpolación entre $+x$ y $-x$ y el modelo aproximado del bloque están descritos en el documento de referencia del kit. :contentReference[oaicite:0]index=0

Cómo representa un peso tu hardware (muy importante)

En tu placa, un "peso" no es una multiplicación directa con un número negativo. En su lugar, cada entrada se duplica como $+x_i$ y $-x_i$, y el trimmer selecciona un valor intermedio.
Si llamamos $w \in [0, 1]$ a la posición normalizada del trimmer (conceptualmente), el voltaje en el cursor (wiper) es:

$$V_b(x_i) = (1 - w) \cdot (+x_i) + w \cdot (-x_i) = (1 - 2w)\, x_i$$

Definimos el **peso efectivo firmado** como:

$$\alpha = 1 - 2w \quad \Rightarrow \quad \alpha \in [-1, 1]$$

En este laboratorio tú vas a entrenar α y b (conceptualmente), pero los ajustarás físicamente girando trimmers.

Tarea a aprender: rotación 2D de $45°$

La red debe aprender a rotar el vector de entrada $[x_1, x_2]^\top$ $45°$ en sentido antihorario.
Objetivo (en el dominio $\hat{u}$):

$$\begin{bmatrix} u_1^\star \\ u_2^\star \end{bmatrix} = \begin{bmatrix} \cos 45° & -\sin 45° \\ \sin 45° & \cos 45° \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Como $\cos 45° = \sin 45° \approx 0.707$, el óptimo ideal sería:

$$\alpha_{11} \approx 0.707,\ \alpha_{12} \approx -0.707,\ b_1 \approx 0 \qquad \alpha_{21} \approx 0.707,\ \alpha_{22} \approx 0.707,\ b_2 \approx 0$$

Dataset (pocos puntos, bien condicionados)

Usa estos 6 puntos. Están dentro de $[-1, 1]$ V para mantener z lejos de saturación.

Muestra	x_1 [V]	x_2 [V]	$u_1^\star$ [V]	$u_2^\star$ [V]
A	$+1.0$	0.0	$+0.707$	$+0.707$
B	0.0	$+1.0$	-0.707	$+0.707$
C	-1.0	0.0	-0.707	-0.707
D	0.0	-1.0	$+0.707$	-0.707
E	$+1.0$	$+1.0$	0.0	$+1.414$
F	-1.0	$+1.0$	-1.414	0.0

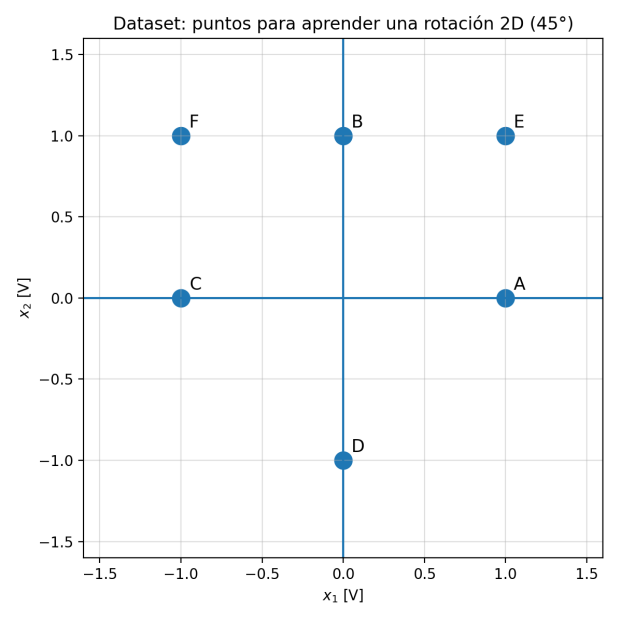


Figura 4.17: Puntos del dataset para aprender una rotación.

Qué vas a medir en la placa

Para cada una de las dos neuronas de salida:

- z_1: test point de **preactivación** de la neurona 1.
- z_2: test point de **preactivación** de la neurona 2.

Además, necesitas fijar las entradas x_1, x_2 con una fuente DC, los potenciómetros de entrada del kit, o un generador en modo DC.

Material

- Multímetro en modo voltaje DC (imprescindible).
- Destornillador para trimmers.
- Opcional: osciloscopio para visualizar estabilidad del voltaje y ruido.

Paso 0: configuración para modo lineal

1. Conecta la punta negra del multímetro a **GND** y no la muevas durante todo el laboratorio.
2. Asegúrate de que estás midiendo el nodo **pre-diodo** (z), no el nodo post-activación.
3. Si la neurona tiene un tercer peso (entrada 3) que no vas a usar, ajústalo a **cero efectivo**: pon el trimmer en el centro para que $\alpha_{k3} \approx 0$ (equivalente a $w \approx 0.5$).

Paso 1: calibración rápida de parámetros (cómo traducir pot $\leftrightarrow \alpha$ y b)

Esta parte hace que el laboratorio sea viable. No vas a intentar "adivinar" α mirando un potenciómetro. En su lugar, vas a **inferir** α con una medida directa.
Para una neurona k:

$$z_k \approx G(\alpha_{k1}x_1 + \alpha_{k2}x_2 + b_k)$$

1A. Medir (o fijar) el bias b_k

Pon $x_1 = 0$ V y $x_2 = 0$ V. Entonces:

$$z_k \approx G b_k \quad \Rightarrow \quad b_k \approx \frac{z_k}{G}$$

Ajusta el trimmer de bias hasta que z_k sea el deseado.

1B. Medir (o fijar) el peso α_{k1}

Pon $x_1 = +1.0$ V y $x_2 = 0$ V. Con bias ya fijado:

$$z_k \approx G(\alpha_{k1} \cdot 1 + b_k) \quad \Rightarrow \quad \alpha_{k1} \approx \frac{z_k}{G} - b_k$$

Ajusta el trimmer asociado a x_1 hasta que z_k coincida con el objetivo.

1C. Medir (o fijar) el peso α_{k2}

Pon $x_1 = 0$ V y $x_2 = +1.0$ V:

$$\alpha_{k2} \approx \frac{z_k}{G} - b_k$$

Ajusta el trimmer asociado a x_2.

Visión Técnica

Consejo práctico

En la práctica, siempre que quieras "poner" un valor numérico de α o b, lo haces en modo calibración y usando z. Esto convierte un problema difícil (leer el valor de un trimmer) en uno fácil (ajustar hasta ver un voltaje).

Paso 2: función de pérdida (lo que minimizamos)

Para cada muestra, trabajaremos en el dominio u, donde:

$$\hat{u}_k = \frac{z_k}{G}$$

Definimos el error por salida:

$$e_k = \hat{u}_k - u_k^\star$$

Y la pérdida por muestra:

$$\mathcal{L} = \tfrac{1}{2}(e_1^2 + e_2^2)$$

Paso 3: gradientes y regla de actualización (SGD manual)

Para una muestra (x_1, x_2):

$$\hat{u}_k = \alpha_{k1} x_1 + \alpha_{k2} x_2 + b_k$$

Gradientes:

$$\frac{\partial \mathcal{L}}{\partial \alpha_{k1}} = e_k x_1 \qquad \frac{\partial \mathcal{L}}{\partial \alpha_{k2}} = e_k x_2 \qquad \frac{\partial \mathcal{L}}{\partial b_k} = e_k$$

Actualización con learning rate η:

$$\alpha_{k1} \leftarrow \alpha_{k1} - \eta e_k x_1 \quad \alpha_{k2} \leftarrow \alpha_{k2} - \eta e_k x_2 \quad b_k \leftarrow b_k - \eta e_k$$

Elección de η (para humanos)

Un valor inicial razonable: $\eta \in [0.05, 0.2]$.

- Si ves que el error oscila o empeora, reduce η.
- Si baja muy lento y los cambios son imperceptibles, aumenta η.

Paso 4: bucle de entrenamiento guiado (muestra a muestra)

Vas a hacer 2 o 3 épocas (recorrer A-F varias veces). Con 6 muestras, esto es realista en una sesión.
Para cada muestra del dataset:

1. Pon x_1, x_2 según la tabla.
2. Mide z_1 y z_2.
3. Calcula $\hat{u}_1 = z_1/G, \hat{u}_2 = z_2/G$.
4. Calcula $e_1 = \hat{u}_1 - u_1^\star, e_2 = \hat{u}_2 - u_2^\star$.
5. Calcula los nuevos parámetros $\alpha'_{k1}, \alpha'_{k2}, b'_k$ con las ecuaciones de actualización.
6. Entra en **modo calibración** y ajusta trimmers para fijar esos nuevos valores:
   a) Bias: con $x_1 = 0, x_2 = 0$, ajusta hasta que $z_k = Gb'_k$.
   b) Peso de x_1: con $x_1 = +1, x_2 = 0$, ajusta hasta que $z_k = G(\alpha'_{k1} + b'_k)$.
   c) Peso de x_2: con $x_1 = 0, x_2 = +1$, ajusta hasta que $z_k = G(\alpha'_{k2} + b'_k)$.

Plantilla de tabla (para registrar el entrenamiento)

Rellena una fila por muestra (y por época).

Muestra	x_1	x_2	z_1	z_2	$\hat{u}_1$	$\hat{u}_2$	e_1	e_2
A								
B								
C								
D								
E								
F								

Ejemplo numérico de una actualización (para no perderse)

Supón que estás en la muestra A: $x_1 = 1.0, x_2 = 0.0$, y el objetivo es $u_1^\star = 0.707$.
Si mides $z_1 = 2.10$ V y usas $G = 3.3$, entonces:

$$\hat{u}_1 = z_1/G \approx 0.636, \qquad e_1 \approx 0.636 - 0.707 = -0.071$$

Con $\eta = 0.1$:

$$\alpha_{11} \leftarrow \alpha_{11} - 0.1 \cdot (-0.071) \cdot 1 = \alpha_{11} + 0.0071$$
$$\alpha_{12} \leftarrow \alpha_{12} - 0.1 \cdot (-0.071) \cdot 0 = \alpha_{12}$$
$$b_1 \leftarrow b_1 - 0.1 \cdot (-0.071) = b_1 + 0.0071$$

Después, vuelves al modo calibración y ajustas hasta cumplir $z_1 = G(\alpha_{11} + b_1)$ en la condición $x_1 = 1, x_2 = 0$, y $z_1 = Gb_1$ en $x_1 = 0, x_2 = 0$.

Paso 5: evaluación por época (ver que realmente mejora)

Al final de cada época (después de A-F), mide z_1, z_2 para todas las muestras sin actualizar parámetros, y calcula:

$$\mathcal{L}_{\text{media}} = \frac{1}{6}\sum_{n=1}^{6} \tfrac{1}{2}\left(e_1^{(n)2} + e_2^{(n)2}\right)$$

Resultados esperados y discusión

- Deberías ver una disminución clara de la pérdida en 1 a 3 épocas.
- Los valores finales se acercarán a los coeficientes de la rotación $45°$, pero no tienen por qué clavarse por:
  - offsets y no idealidad del inversor/buffer,
  - pequeñas diferencias en G,
  - resolución mecánica del ajuste.
- Si alguna vez notas saturación (por ejemplo z acercándose a varios voltios altos de forma sistemática), reduce amplitud del dataset o reduce los α objetivo.

Visión Técnica

Qué demuestra este laboratorio

No estás “afinando” una red pre-entrenada. Estás ejecutando un optimizador de forma manual sobre un sistema físico real. Cada actualización que haces corresponde a una regla matemática de descenso por gradiente, y el hecho de que la pérdida baje valida tanto el modelo como la controlabilidad del hardware.

Capítulo 5

Entrenando el Mundo Físico

5.1 El Kit de Software

Introducción a la librería de Python que acompaña al hardware para facilitar la interacción y el control.

5.2 Entrenamiento Consciente del Hardware

5.2.1. Por qué la simulación no es suficiente

Discusión sobre el ruido real, la deriva térmica y las imperfecciones que los simuladores de software suelen ignorar.

5.2.2. El "Bucle": Ordenador $\leftrightarrow$ Placa

Explicación del flujo de trabajo híbrido donde el PC ajusta los pesos y la placa realiza la inferencia analógica.

5.3 Construyendo el Clasificador de Círculos

Guía práctica paso a paso para configurar la red y resolver el problema clásico de clasificación no lineal de círculos concéntricos.

Parte IV

El Maestro: La Física de la Inteligencia

Capítulo 6

Fundamentos Matemáticos

TODO write the entire part.