Capítulo 4

RED NEURONAL DE HOPFIELD

Introducción

En este capítulo veremos un tipo de red neuronal dinámica, considerada así porque en esencia corresponde a un sistema cuyos estados dependen de la variable tiempo. Iniciaremos con una breve revisión a las memorias asociativas bidireccionales con el fin de dejar las bases necesarias para comprender el modelo de red dinámica propuesto, en 1982, por Hopfield. Abordaremos los modelos discretos y continuos propuestos por Hopfield, iniciando con la arquitectura de este tipo de red, continuaremos con el procesamiento de los datos y los respectivos algoritmos de aprendizaje.

Al final, mostraremos algunas aplicaciones en optimización de sistemas y reconocimiento y clasificación de patrones.

John Hopfield

Profesor de la Universidad de Princeton, formuló en 1982 un novedoso modelo de redes neuronales que se denominó Red de Hopfield.

Las redes dinámicas de Hopfield han sido muy usadas en modelado de sistemas no lineales, aunque se pueden aplicar a problemas de diversa índole, desde la clasificación de patrones hasta la optimización de funciones

MEMORIA AUTOASOCIATIVA BIDIRECCIONAL (BAM)

Una de las características más poderosas que tiene la inteligencia humana es la capacidad para asociar hechos, datos, situaciones, etc. Por ejemplo, si lanzamos una pregunta ¿quién es Rosenblatt?, ¿qué reacciones podríamos esperar?. Como es natural entre la comunidad general, muy probablemente, no se presentará una reacción especial y muy seguramente, lo asociarán al nombre de una persona de origen anglosajón. Pero entre nosotros los lectores de temas relacionados con redes neuronales artificiales, esperamos que sí se haya generado algún tipo de reacción o asociación especial; sin temor a equivocarnos, al escuchar este nombre muy seguramente a nuestras mentes se viene de inmediato el nombre de la red neuronal tipo Perceptron.
Esta maravillosa capacidad de nosotros los seres humanos de realizar asociaciones entre ideas, conceptos, cosas, etc. trataremos de emularla con este nuevo tipo de red neuronal llamada BAM.

Arquitectura de la BAM

La memoria asociativa bidireccional (BAM) presentada en la figura 4.1, está constituida por dos capas de neuronas que son los elementos básicos de procesamiento de la información. Estas capas están completamente interconectadas y el flujo de la información va desde la capa de entrada hacia la capa de salida y desde la capa de salida hacia la de entrada, por lo que los pesos son bidireccionales.

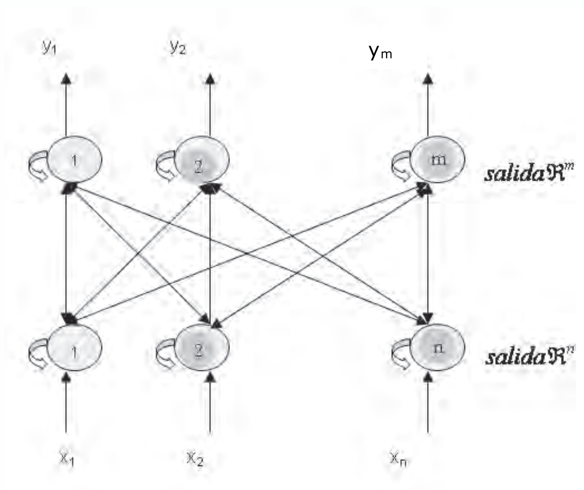

Figura 4.1 Memoria Asociativa Bidireccional

Si el problema está completamente definido, entonces conocemos todos los vectores de entrenamiento por anticipado y podemos determinar la matriz de pesos ***W*** utilizando la ecuación 4.1, donde *P* es el número de patrones de entrenamiento.

$$\mathbf{W} = \mathbf{y}_1\mathbf{x}_1^{\mathrm{T}} + \ldots + \mathbf{y}_i\mathbf{x}_i^{\mathrm{T}} + \ldots + \mathbf{y}_P\mathbf{x}_P^{\mathrm{T}} \qquad (4.1)$$

Con esta expresión obtenemos la matriz ***W*** de pesos sinápticos de la red neuronal a partir de los pares ordenados de vectores de entrada y salida del conjunto de entrenamiento.

Memoria autoasociativa

Al igual que la BAM, la memoria autoasociativa está constituida por dos capas de neuronas que son los elementos básicos de procesamiento de la información (figura 4.2.). Estas capas están completamente interconectadas y el flujo de los datos va desde la capa de entrada hacia la de salida y desde la capa de salida hacia la de entrada, por lo que las conexiones y sus respectivos pesos son bidireccionales.

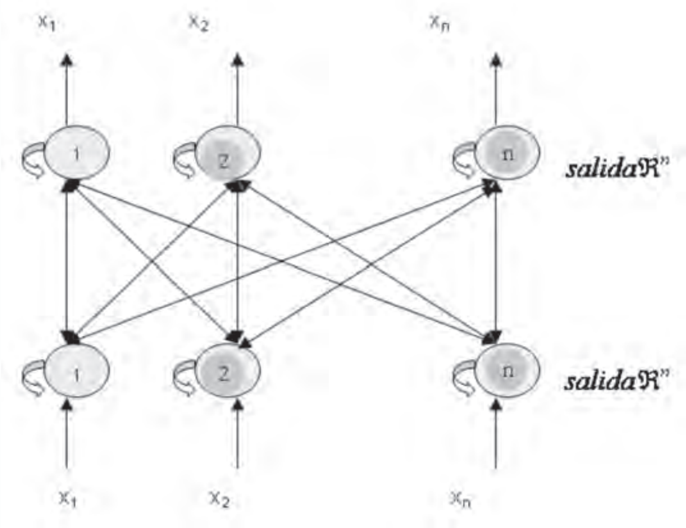

Fig. 4.2 Memoria Autoasociativa

Si conocemos todos los vectores de entrenamiento por anticipado, podemos determinar los pesos utilizando la ecuación 4.2.

$$\mathbf{W} = \mathbf{x}_1\mathbf{x}_1^{\mathrm{T}} + \ldots + \mathbf{x}_i\mathbf{x}_i^{\mathrm{T}} + \ldots + \mathbf{x}_P\mathbf{x}_P^{\mathrm{T}} \qquad (4.2)$$

Con esta expresión obtenemos la matriz ***W*** de pesos sinápticos de la red neuronal, con base en los pares ordenados de vectores de entrada y salida del conjunto de entrenamiento. Nótese que por tratarse de un proceso de autoasociación, los vectores de entrada y salida utilizados para el entrenamiento son iguales. En este caso las dos capas son de igual dimensión por lo que la matriz de pesos $\boldsymbol{W}$ es simétrica y de dimensión *nxn*.

Una vez obtenemos la matriz de pesos, podemos utilizar la red para asociar los vectores de entrada y salida, garantizando que ésta es capaz de responder a datos contaminados con ruido.

Procesamiento de Información en la BAM

Una vez construida la matriz de pesos con base en la ecuación 4.1, la BAM puede usarse para recordar la información que almacenan los vectores del conjunto de entrenamiento. Los pasos que llevaremos a cabo para procesar la información son los siguientes:

1. Aplicamos un par de vectores (x_i, y_i) a las neuronas de la BAM.
2. Propagamos la información de la capa $\boldsymbol{x}$ a la capa $\boldsymbol{y}$ y se actualizan las salidas de las unidades de la capa $\boldsymbol{y}$. En este caso se inicia de $\boldsymbol{x}$ hacia $\boldsymbol{y}$, sin embargo, puede ser de $\boldsymbol{y}$ hacia $\boldsymbol{x}$ y se conoce como contra-propagación.
3. Propagamos la información $\boldsymbol{y}$ actualizada en el paso anterior hacia la capa $\boldsymbol{x}$, y se actualizan estas unidades.
4. Repetimos los pasos 2 y 3 hasta que no haya cambios en las salidas de las dos capas.

Si el entrenamiento es el adecuado, el sistema convergerá a un punto que corresponderá a uno de los vectores utilizados para construir la matriz de pesos en la fase de aprendizaje. La salida para una entrada $\boldsymbol{x}$ cualquiera será el $\hat{O}(x_i) = y_i$ más cercana.

Este algoritmo nos muestra la naturaleza bidireccional del procesamiento de los datos en la BAM. Para ampliar este concepto, consideremos un problema donde es necesario diseñar con este tipo de red un sistema que asocie la huella dactilar y nombre de un usuario, la diferencia con una tabla tradicional, es que si introducimos una huella dactilar contaminada con ruido, el sistema está en capacidad de recuperar el nombre correcto, en cambio una solución basada en una tabla no lo hace.

El caso inverso también el sistema puede resolverlo, por ejemplo, si introducimos el apellido "*Pérec*" a la BAM, ésta a lo largo de varias iteraciones por las conexiones de la misma sería capaz de recuperar el apellido correcto "*Pérez*" y su correspondiente huella dactilar. Una base de datos con un sistema de búsqueda tradicional no sería capaz de recuperarse de este tipo de errores.

Modelo discreto de hopfield

El modelo discreto de la red neuronal de Hopfield se asimila a una memoria autoasociativa, donde la propuesta es reducir las dos capas de neuronas de la BAM a una sola, donde la salida de estas neuronas se lleva a la entrada. En la figura 4.3, mostramos este modelo de red dinámica propuesto por Hopfield y del cual recibió su nombre. A cada una de las neuronas les llega la entrada I_i, que son transformada por la unidad de procesamiento con los pesos sinápticos w_{ij}, para generar la salida y_i.

La arquitectura corresponde a una red de una capa de neuronas o unidades de procesamiento, por lo que esta red pertenece a las denominadas redes neuronales monocapa. Esta capa está conformada por *n* neuronas y la salida de dichas neuronas constituye el estado de la red.

La salida de cada neurona es propagada hacia la entrada de cada una de las otras neuronas, pero no a si misma, por lo que no existe auto-recurrencia; es decir, la salida de una neurona no afecta la entrada de si misma.

Para representar los pesos de la red neuronal de Hopfield usaremos la notación matricial, en la cual profundizaremos más adelante.

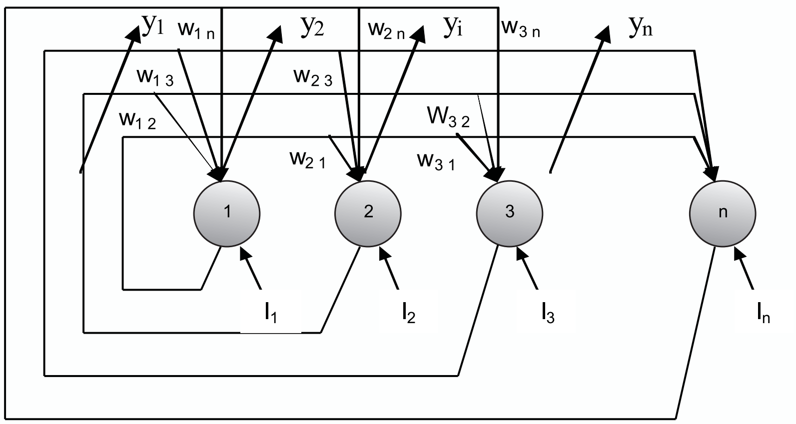

Fig. 4.3 Modelo discreto de Hopfield

Proceso de aprendizaje

En este apartado presentamos el proceso de aprendizaje de una red de Hopfield discreta. La información que esta red va a procesar es del tipo binario, pudiéndose codificar los datos como cero y uno (0,1), o como más uno y menos uno (+1,-1).

En la ecuación 4.3, presentamos la forma de calcular el peso w_{ij} para patrones codificados con valores 0 y 1, y en la ecuación 4.4 la expresión matemática para calcular el peso w_{ij} para patrones codificados con valores

-1 y 1. Donde e_i^k y e_j^k, son las componentes *i-ésima* y *j-ésima* del *k-ésimo* patrón de entrenamiento, *n* es el número de patrones a memorizar en la red neuronal.

$$w_{ij} = \begin{cases} \sum_{k=1}^{n} (2e_i^{\,k} - 1)(2e_j^{\,k} - 1) \Rightarrow 1 \le i; j \le n; i \ne j \\ 0 \Rightarrow 1 \le i; j \le n; i \ne j \end{cases} \tag{4.3}$$

$$w_{ij} = \begin{cases} \sum_{k=1}^{n} (e_i^{\,k} e_j^{\,k}) \Rightarrow 1 \le i; j \le n; i \ne j \\ 0 \Rightarrow 1 \le i; j \le n; i \ne j \end{cases} \tag{4.4}$$

Principio de funcionamiento

Como ya estamos en capacidad de calcular la matriz de pesos de una red de Hopfield discreta, ahora vamos a estudiar su funcionamiento, es decir, vamos a ver como actualizamos la salida de la red.

Como lo hemos descrito a lo largo del texto, cada una de las neuronas procesa la información de entrada calculando su entrada total como la sumatoria de las entradas por sus respectivos pesos, pero ahora adicionamos el efecto de la recurrencia que se realiza desde las salidas de las otras neuronas, tal como se muestra en la expresión 4.5 que calcula la entrada neta $Neta_i$.

Una vez calculado el valor total de entrada con la ecuación 4.5, seleccionamos de manera aleatoria una neurona de la red para calcular la salida que la misma va a generar. En el paso siguiente calculamos el valor de la entrada neta de dicha neurona. Teniendo el valor de la entrada neta calculada, actualizamos la salida de la neurona con la ecuación 4.6, para valores codificados como {-1,+1}. En estas expresiones, I_i corresponde a la entrada externa de la *i-ésima* neurona, y_j es la salida de la neurona *j-ésima*. Cuando se plantea que i≠j es porque este tipo de redes no admite la auto-recurrencia. Si las entradas están codificadas como {0,1} la ecuación 4.6 conserva su estructura, considerando que cuando la Neta es negativa, la salida es cero.

$$Neta_i = \sum_{j=1}^{N} y_j w_{ij} + I_i \quad \text{con}\, i \ne j \tag{4.5}$$

$$x_i(t+1) = \begin{cases} 1; Neta > 0 \\ x_i(t); Neta = 0 \\ -1; Neta < 0 \end{cases} \tag{4.6}$$

Como podemos observar si la entrada neta es igual a cero, el estado o salida de la neurona se hace igual a la salida de la neurona en el instante anterior; de esta manera, podemos visualizar una posible condición para que la salida de la red se estabilice en un valor determinado.

Concepto de energía en el modelo discreto de hopfield

La red de Hopfield es un sistema dinámico porque las salidas de la red pueden asimilarse a un vector de estado, que a medida que se procesa la información, cambia con el tiempo. Para verificar la convergencia de la misma se hace uso del criterio de estabilidad de Lyapunov.

Para efectos de mostrar el cumplimiento del criterio de estabilidad de Lyapunov, definimos en la ecuación 4.7, una función de energía para la red. Podemos verificar que a medida que el estado de una red de Hopfield evoluciona la energía de la red se hace cada vez menor. En otras palabras, la energía de la red tiende a decrecer de manera asintótica.

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}x_i x_j + \sum_{i=1}^{N}\theta_i x_i \tag{4.7}$$

Donde,

w_{ij} Valor del peso entre la neurona *i-ésima* y la *j-ésima*
x_i Salida de la neurona *i-ésima*
x_j Salida de la neurona *j-ésima*
N Número de neuronas en la capa
θ_i Umbral de la neurona *i-ésima*

Ejemplo de procesamiento

Con el fin de aclarar el procedimiento para calcular la matriz de pesos, el funcionamiento de la red y el procedimiento para de la energía, revisemos el siguiente ejemplo donde el problema viene descrito por dos vectores de entrenamiento x_1 y x_2.

$x_1^T = [\ 1\ \ 1\ -1\ -1]$
$x_2^T = [-1\ -1\ \ 1\ \ 1]$

Si aplicamos las expresiones vistas en los apartados anteriores, podemos calcular la matriz de pesos, así:

$$\mathbf{W} = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \cdot [\,1 \;\; 1 \; \text{-}1 \; \text{-}1] + \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \cdot [\text{-}1 \; \text{-}1 \;\; 1 \;\; 1]$$

$$\mathbf{W} = \begin{bmatrix} 0 & 2 & -2 & -2 \\ 2 & 0 & -2 & -2 \\ -2 & -2 & 0 & 2 \\ -2 & -2 & 2 & 0 \end{bmatrix}$$

Una vez calculada la matriz de pesos, podemos pasar a la fase de funcionamiento y para ello supongamos que le presentamos a la entrada de la red el patrón .

La red responderá ante el estímulo de este patrón de entrada modificando sus salidas en cada una de sus neuronas:

Para la neurona 1, calculamos la entrada neta:

$$Neta_1 = x_2 w_{12} + x_3 w_{13} + x_4 w_{14} + I_1 = 1 * 2 + 1 * -2 + -1 * -2 + 1 = 3$$

Luego, evaluamos esta entrada en la función de activación y obtenemos su respectiva salida:

$$x_1 = 1$$

Para la neurona dos, calculamos la entrada neta:

$$Neta_2 = x_1 w_{21} + x_3 w_{23} + x_4 w_{24} + I_2 = 1 * 2 + 1 * \text{-}2 + \text{-}1 * \text{-}2 + 1 = 3$$

Luego, evaluamos esta entrada en la función de activación y obtenemos su respectiva salida:

$$x_2 = 1$$

Para la neurona tres, calculamos la entrada neta:

$$Neta_3 = x_1 w_{31} + x_2 w_{32} + x_4 w_{34} + I_3 = 1 * \text{-}2 + 1 * \text{-}2 + \text{-}1 * 2 + 1 = \text{-}5$$

Luego, evaluamos esta entrada en la función de activación y obtenemos su respectiva salida:

$$x_3 = \text{-}1$$

Observemos que en este caso la salida x_3 ha cambiado su valor, por lo que para calcular la neta de la neurona cuatro se debe tener en cuenta este nuevo valor:

$$Neta_4 = x_1 w_{41} + x_2 w_{42} + x_3 w_{43} + I_4 = 1*-2 + 1*-2 + -1*2 - 1 = -7$$

Luego, evaluamos esta entrada en la función de activación y obtenemos su respectiva salida:

$$x_4 = -1$$

Finalmente, la salida de la red viene determinada por el vector $\boldsymbol{x_f}$:

$$x_f = \begin{bmatrix} 1 & 1 & -1 & -1 \end{bmatrix}$$

Como podemos observar, la salida de la red converge al primer patrón que fue almacenado en la red. Ahora verifiquemos como es el comportamiento de la energía de la red, para ello retomemos el vector de entrada $\boldsymbol{x_0}$.

$$x_0 = \begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix}$$

Al iniciar la salida de la red es el mismo patrón de entrada y, por lo tanto, la energía de la red en este caso la calculamos así:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji} x_i x_j \ , \quad i \neq j$$

Para *i=1*

$$E_1 = w_{21} x_1 x_2 + w_{31} x_1 x_3 + w_{41} x_1 x_4 = 2$$

Para *i=2*

$$E_2 = w_{12} x_2 x_1 + w_{32} x_2 x_3 + w_{42} x_2 x_4 = 2$$

Para *i=3*

$$E_3 = w_{13} x_3 x_1 + w_{23} x_3 x_2 + w_{43} x_3 x_4 = -6$$

Para *i=4*

$$E_4 = w_{14}x_4x_1 + w_{24}x_4x_2 + w_{34}x_4x_3 = 2$$

La energía total de la red la calculamos con la sumatoria de los elementos parciales anteriores:

$$E = -\frac{1}{2}\sum_{i=1}^{4} E_i = 2 + 2 - 6 + 2 = 0$$

Ahora calculemos la energía de la red para la salida final $\boldsymbol{x_f}$, hacia donde converge la red.

$$x_f = \begin{bmatrix} 1 & 1 & -1 & -1 \end{bmatrix}$$

Para i=1

$$E_1 = w_{21}x_1x_2 + w_{31}x_1x_3 + w_{41}x_1x_4 = 6$$

Para i=2

$$E_2 = w_{12}x_2x_1 + w_{32}x_2x_3 + w_{42}x_2x_4 = 6$$

Para i=3

$$E_3 = w_{13}x_3x_1 + w_{23}x_3x_2 + w_{43}x_3x_4 = 6$$

Para i=4

$$E_4 = w_{14}x_4x_1 + w_{24}x_4x_2 + w_{34}x_4x_3 = 6$$

La energía total de la red, para la salida final que implicó la convergencia, la calculamos con la sumatoria de los elementos parciales anteriores:

$$E = -\frac{1}{2}\sum_{i=1}^{4} E_i = -\frac{1}{2}(6 + 6 + 6 + 6) = -12$$

Observemos que la salida a la cual converge la red tiene una energía menor que la salida inicial. Esto es lógico pues cuando la red de Hopfield cambia su salida es por que está convergiendo a un mínimo de la función de energía.

La explicación de este hecho radica en que la red de Hopfield fue diseñada de tal forma que, por si misma, constituye un sistema estable. Lo anterior significa que la salida de la red converge a un valor en el cual la red se estabiliza. Lo que esperamos es que esta salida a la cual converge la red, sea uno de los patrones almacenados en ella. Sin embargo, debe tenerse en cuenta que este comportamiento se garantiza si los patrones a memorizar son ortogonales. Si no se cumple esta condición la red se puede estabilizar en una salida que no fue la almacenada, pero que constituye un mínimo de la función de energía.

MODELO CONTINUO DE HOPFIELD

Hemos estudiado con detenimiento el modelo discreto de red neuronal propuesto por Hopfield, ahora nos detendremos en el análisis de la arquitectura, la convergencia y las aplicaciones del modelo continuo de la red neuronal de Hopfield. En la figura 4.4, observamos el modelo para esta red propuesto por Hopfield, que a partir de dispositivos eléctricos y electrónicos.

En el modelo continuo de Hopfield representaremos a cada neurona con un circuito constituido por un condensador, una conductancia y un amplificador no lineal con función de transferencia sigmoidal, que genera la salida de la misma.

La salida de una neurona se lleva hacia las demás como señal de excitación o inhibición, con magnitud positiva o negativa respectivamente; por esta razón, adicionaremos al modelo un inversor a la salida de cada una de las neuronas.

La recurrencia propia de este tipo de redes, la representamos llevando la salida de cada neurona hacia la entrada de las demás. Con pequeños círculos hemos marcado las conexiones que se establecen, para el caso de una neurona que es excitada por la salida de otra, la conexión la marcamos sobre la salida positiva, en caso de existir una inhibición, la conexión la marcamos sobre la salida negativa. Esta situación en el modelo matemático lo reflejamos con la transconductancia T_{1n} que representa el valor de la conexión.

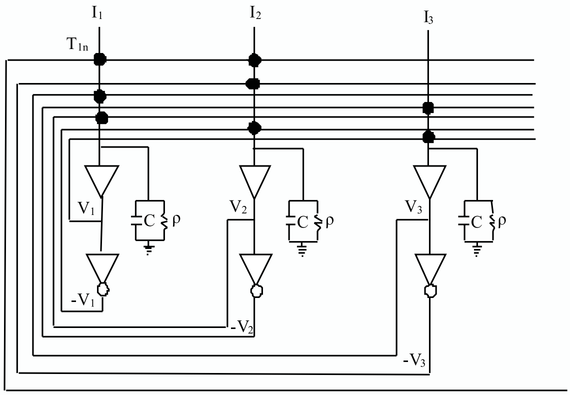

Fig. 4.4 Modelo Continuo de Hopfield

Modelo continuo de hopfield de una neurona

Vamos a detenernos en el modelo continuo que Hopfield propone para una neurona en el marco del modelo global de red neuronal. Como podemos observar de la figura 4.5, asimilamos la neurona a un nodo que recibe el aporte en corriente de las salidas de las otras neuronas a través de las transconductancias T_{ij}. Por otro lado, el nodo recibe un aporte de corriente de una fuente externa, para representar la entrada a la neurona.

El valor total de corriente se lleva al circuito paralelo conformado por el condensador y la conductancia, que representa la resistencia y la capacitancia de la membrana celular. El potencial u_i que se genera en este circuito se lleva a la entrada del amplificador no lineal, con función de transferencia sigmoidal, como una aproximación de la salida de la neurona en respuesta al potencial de activación total.

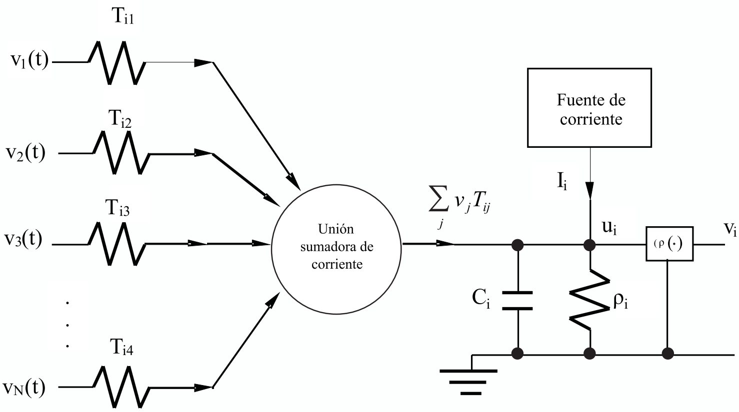

Fig. 4.5 Modelo Continuo de Hopfield de una Neurona

Si aplicamos el análisis de circuitos a este modelo continuo de Hopfield de una neurona y usando las leyes de Kirchoff, para el nodo u_i, la ecuación equivalente es la 4.8.

$$I_i + \sum_j (v_j - u_i) T_{ij} = C_i \frac{du_i}{dt} + \frac{u_i}{\rho_i} \tag{4.8}$$

Si reorganizamos la ecuación 4.8, esta expresión la podemos transformar así:

$$\begin{aligned} C_i \frac{du_i}{dt} &= I_i + \sum_j v_j T_{ij} - \frac{u_i}{\rho_i} - \sum_j u_i T_{ij} \\ C_i \frac{du_i}{dt} &= I_i + \sum_j v_j T_{ij} - u_i\left(\frac{1}{\rho_i} + \sum_j T_{ij}\right) \end{aligned} \tag{4.9}$$

Si definimos el segundo término de la sumatoria como,

$$\frac{1}{R_i} = (\frac{1}{\rho_i} + \sum_j T_{ij}) \tag{4.10}$$

entonces podemos simplificar la expresión de la corriente total que llega a la capacitancia, así:

$$C_i \frac{du_i}{dt} = I_i + \sum_j v_j T_{ij} - \frac{u_i}{R_i} \tag{4.11}$$

Si dividimos los dos lados de la ecuación entre C_i, obtenemos la siguiente ecuación diferencial 4.12 que modela el comportamiento del potencial u_i.

$$\frac{du_i}{dt} = \frac{I_i}{C_i} + \sum_j \frac{v_j T_{ij}}{C_i} - \frac{u_i}{R_i C_i} \tag{4.12}$$

Con el fin de simplificar la anterior ecuación, vamos a definir los siguientes términos:

$$\omega_{ij} = \frac{T_{ij}}{C_i} \qquad a_i = \frac{1}{R_i C_i} \qquad b_i = \frac{1}{C_i}$$

De esta manera, vemos en la ecuación 4.13 como el cambio en el voltaje u_i queda representado en función de la entrada externa Ii, la sumatoria de las salidas de las demás neuronas y el valor del potencial u_i.

$$\frac{du_i}{dt} = b_i I_i + \sum_j \omega_{ij} v_j - a_i u_i \tag{4.13}$$

Si observamos detenidamente esta expresión, encontraremos su similitud al modelo matemático que hemos venido utilizando para representar a una neurona artificial.

La ecuación 4.13 que representa el modelo continuo de Hopfield para una neurona la podemos representar en formato matricial usando la ecuación 4.14.

$$\frac{\mathbf{dU}}{\mathbf{dt}} = \mathbf{WV - AU + BI} \tag{4.14}$$

Si definimos al vector de estado como vector de potenciales de activación total de las diferentes neuronas, obtenemos las siguientes expresiones:

X= U (4.15)
V= Φ (U)=Φ(X) (4.16)

Si reemplazamos los vectores de estado definidos en el modelo continuo para la neurona, obtenemos la ecuación 4.17 que representa en el espacio de estado el mode-lo continuo de Hopfield .

$$\frac{\mathbf{dX}}{\mathbf{dt}} = \mathbf{W\Phi(X) - AX + BI} \qquad (4.17)$$

Para aclarar estos conceptos apliquemos la ecuación 4.17, en el caso de una red neuronal de dos estados x_1 y x_2, cuya representación matricial quedará de la siguiente manera:

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \begin{bmatrix} \phi(x_1) \\ \phi(x_2) \end{bmatrix} - \begin{bmatrix} a_{11} & 0 \\ 0 & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_{11} & 0 \\ 0 & b_{22} \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix}$$

La figura 4.6, representa el diagrama de bloques nos muestra el modelo continuo de Hopfield de la red neuronal.

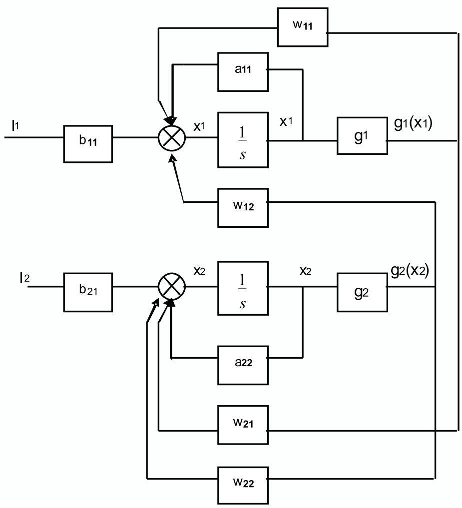

Fig. 4.6 Representación en Espacio de Estado del Modelo Continuo de Hopfield para una Red de dos Neuronas

Función de energía para el modelo continuo de hopfield

De manera similar al caso de la red discreta de Hopfield, la convergencia de la red continua de Hopfield, podemos estudiarla con base en el criterio de estabilidad de Lyapunov.

Para el modelo continuo Hopfield propuso la función de energía, ecuación 4.18, que depende de los potenciales de salida de las neuronas, la transconductancia de conexión, la entrada externa y la función de transferencia del amplificador inversor.

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} T_{ij} v_i v_j + \frac{1}{\lambda}\sum_{i=1}^{n}\frac{1}{R_i}\int_0^{vi} \phi_i^{-1}(v)dv - \sum_{i=1}^{n} I_i v_i \tag{4.18}$$

Para realizar el análisis de estabilidad de la red con base en el criterio de estabilidad de Lyapunov, es necesario utilizar la derivada de la función de energía calculada con la ecuación 4.19.

$$\frac{dE}{dt} = -\sum_i C_i \left(\frac{d(\phi_i^{-1}(v_i))}{dt}\right)\left(\frac{dv_i}{dt}\right)^2 \tag{4.19}$$

Si analizamos la expresión de la derivada, vemos que ésta depende de tres términos:

- La capacitancia C_i que siempre será positiva.
- La derivada de la salida de la *i-ésima* neurona, que al estar elevada al cuadrado, igualmente siempre será positiva.
- La derivada de la inversa de la función de activación de la neurona *i-ésima*, que de igual manera, siempre será positiva.

En la figura 4.7 observamos la función de activación sigmoidal y su inversa. En la gráfica de la función inversa tracemos una recta tangente imaginaria que va recorriendo todos sus puntos, entonces podremos observar que la pendiente de esta recta siempre será positiva y por esta razón la derivada será positiva.

Como todos los términos de la sumatoria de la derivada de la función de energía son positivos, podemos concluir que esta derivada siempre será positiva, y si nos acogemos al criterio de Lyapunov, podemos estar seguros que cuando la red neuronal está en funcionamiento va a converger a un valor estable.

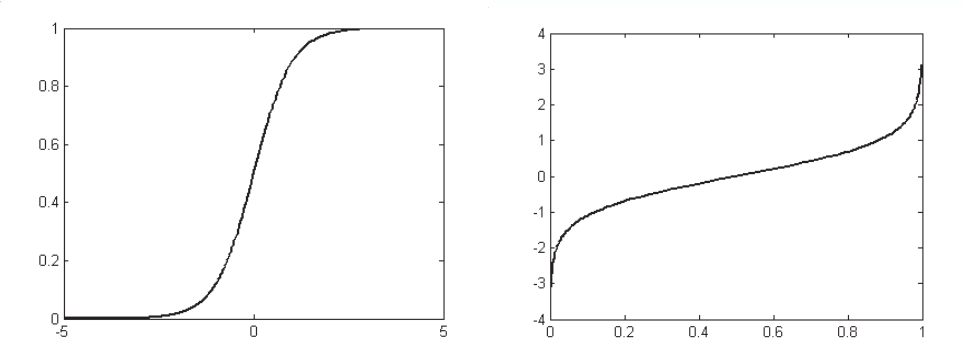

Fig. 4.7 Función de Activación Sigmoidal y su Inversa

APROXIMACIÓN PRÁCTICA

Red tipo hopfield con MATLAB®

En este primer proyecto quedaremos en capacidad de construir una Red de Hopfield usando MATLAB®, con base en el siguiente programa. Si aprovechamos la capacidad que tienen estas redes para memorizar datos, la utilizaremos para almacenar dos puntos en el plano.

```
% Programa que construye una red neuronal tipo HOPFIELD y
almacena dos puntos en el plano

close all;
T=[1 1;-1 -1]';

% Se grafican los valores a almacenar
figure;
plot(T(1,:),T(2,:),'*r');
hold on;
axis([-1.1 1.1 -1.1 1.1]);

% Se crea la red tipo Hopfield
red=newhop(T);

% Se verifica si los estados fueron almacenados
Ai = T;
[Y,Pf,Af] = sim(red,2,[],Ai);
Y
```

```
% Se prueba con un estado diferente
Ai = {[0.8; 0.8]};

% Se simula la red para N pasos
N=10;
[Y,Pf,Af] = sim(red,{1 N},{},Ai);
Y{1};
Evolucion=[cell2mat(Ai) cell2mat(Y)];
Inicio=cell2mat(Ai);
plot(Inicio(1,1),Inicio(2,1),'b+');
plot(Evolucion(1,:),Evolucion(2,:));

disp('Oprima una tecla para continuar');
pause;
for i=1:10
Ai = {2*rand(2,1)-1};
% Se simula la red por N pasos
N=25;
[Y,Pf,Af] = sim(red,{1 N},{},Ai);
Y{1};
EvolucionAux(:,:,i)=[cell2mat(Ai) cell2mat(Y)];
InicioAux(:,:,i)=cell2mat(Ai);
plot(InicioAux(1,1,i),InicioAux(2,1,i),'b+');
plot(EvolucionAux(1,:,i),EvolucionAux(2,:,i));
pause(1)
end;
title('Rojo = Patrones Memorizados Azul =Patrones de Prueba')

hold off;
```

En la ejecución de este proyecto propuesto es importante destacar que la red me-moriza los puntos suministrados, pero si le presentamos un patrón diferente a dichos puntos, obtenemos como resultado que la salida de la red evoluciona desde el patrón presentado hacia uno de los puntos utilizados en el entrenamiento, efecto que visualizamos en la figura 4.8.

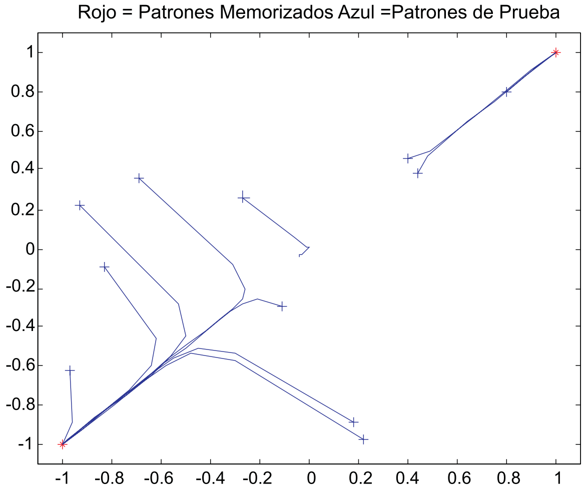

Fig. 4.8 Visualización de la evolución de la salida de la red ante diferentes patrones de entrada

Proyectos propuestos

1. Proponga y simule una red tipo Hopfield para que aprenda los siguiente puntos definidos en un espacio de cuatro dimensiones. Defina patrones con los siguientes vectores:

   P_1 =[1 1 0 0] y P_2 =[0 0 1 1].
   Verifique el comportamiento de la red ante diferentes entradas en MATLAB®

2. El objetivo ahora, es verificar como la red va minimizando su energía a medida que converge al patrón almacenado, para verificar este comportamiento, valide la red con los siguientes patrones.
   Patrón de entrada: [1 1 0 1 1]

Tabla 1 Comportamiento de la red ante el patrón de entrada [1 1 0 11]

Patrón de salida	Energía	Iteración

Patrón de entrada: [0 1 0 1 1]

Tabla 2 Comportamiento de la red ante el patrón de entrada [0 1 0 1 1]

Patrón de salida	Energía	Iteración

Patrón de entrada: [1 0 0 1 1]

Tabla 3 Comportamiento de la red ante el patrón de entrada [1 0 0 1 1]

Patrón de salida	Energía	Iteración

Patrón de entrada: [0 1 0 1 0]

Tabla 4 Comportamiento de la red ante el patrón de entrada [0 1 0 1 0]

Patrón de salida	Energía	Iteración

Patrón de entrada: [1 0 0 0 1]

Tabla 5 Comportamiento de la red ante el patrón de entrada [1 0 0 0 1]

Patrón de salida	Energía	Iteración

Patrón de entrada: [0 0 0 0 1]

Tabla 6 Comportamiento de la red ante el patrón de entrada [0 0 0 0 1]

Patrón de salida	Energía	Iteración

Patrón de entrada: [1 0 0 0 0]

Tabla 7 Comportamiento de la red ante el patrón de entrada [1 0 0 0 0]

Patrón de salida	Energía	Iteración

Patrón de entrada: [1 1 1 1 1]

Tabla 8 Comportamiento de la red ante el patrón de entrada [1 0 0 0 0]

Patrón de salida	Energía	Iteración

Realice un programa en MATLAB® que permita simular la red de Hopfield discreta. Dicho programa debe permitir visualizar la manera como cambia la energía de la red a medida que va iterando.