Modelo de Hopfield de red neuronal

Dominik Pastuszka

28 de junio de 2022

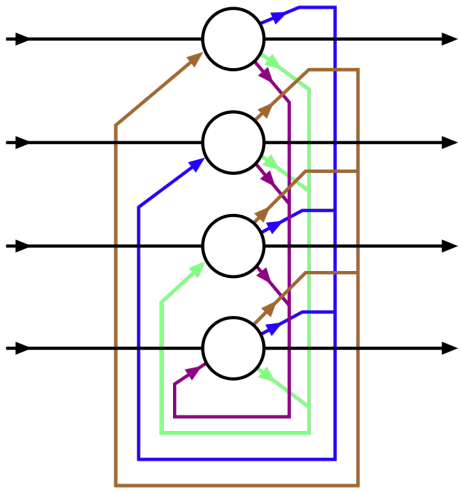


Figura 1: Modelo de Hopfield con cuatro nodos [1].

Resumen

El objetivo de este informe es aplicar el algoritmo de Metrópolis utilizado en el modelo de Ising para simular y estudiar el comportamiento emergente de una red neuronal de Hopfield para cualquier temperatura. Estudiaremos la capacidad del sistema para recordar patrones almacenados, calcularemos el solapamiento entre la salida y el patrón inicial, y determinaremos cómo decae la recuperación de la memoria en función del número de patrones almacenados, aportando además posibles aplicaciones físicas de estos algoritmos.

Índice

1. Introducción y problema a tratar

Una red de Hopfield (también conocida como modelo de Ising de red neuronal) es un tipo de red neuronal artificial recurrente popularizado por John Hopfield en 1982. [2] Estas redes sirven como memorias de contenido direccionable con nodos de umbral binario (o bien con variables continuas) y su principal uso es en el ámbito del reconocimiento de patrones y problemas de optimización, llegando hasta a ser utilizadas en física de partículas para la reconstrucción de trazas. [3] [4]

Vamos a considerar una red cuadrada bidimensional con $N \times N$ nodos y condiciones de contorno periódicas, donde cada nodo de la red representa una neurona que puede estar en dos estados (inactivo o activo): $s_{ij} = \{0, 1\}$.

El Hamiltoniano (energía) para cada configuración del sistema se expresa como:

$$H(s) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N}\omega_{ij,kl}s_{ij}s_{kl} + \sum_{i=1}^{N}\sum_{j=1}^{N}\theta_{ij}s_{ij} \tag{1}$$

Las interacciones entre las distintas neuronas se denominan pesos sinápticos ($\omega_{ij,kl}$) y representan el estado de las conexiones entre neuronas. Estos pesos sinápticos vienen dados en términos de P configuraciones de la red previamente almacenadas o patrones $\xi_{ij}^{\mu} = \{0, 1\}$. Estos patrones minimizan el Hamiltoniano y corresponden a los mínimos de la energía libre cuando $T \to 0$, de forma que tenemos:

$$\omega_{ij,kl}\begin{cases}\dfrac{1}{N^2}\displaystyle\sum_{\mu=1}^{P}(\xi_{ij}^{\mu} - a^{\mu})(\xi_{kl}^{\mu} - a^{\mu}) & \text{si } (ij) \neq (kl) \\ 0 & \text{si } (ij) = (kl)\end{cases} \tag{2}$$

Podemos ver que se trata de una red de largo alcance totalmente conectada, pero donde las autoconexiones no están permitidas. El parámetro a^{μ} es la proporción de neuronas activas del patrón ξ^{μ} almacenado:

$$a^{\mu} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\xi_{ij}^{\mu} \tag{3}$$

Por otro lado, el término θ_{ij}, denominado umbral de disparo, viene dado en términos de los pesos sinápticos:

$$\theta_{ij} = \frac{1}{2}\sum_{k=1}^{N}\sum_{l=1}^{N}\omega_{ij,kl} \tag{4}$$

Implementando el algoritmo de Metrópolis, la probabilidad de transición de un estado s a un estado s' viene dado por:

$$p_{s\to s'} = \text{mín}(1, e^{-\Delta H/T}) \quad \text{con } \Delta H = H(s') - H(s) \tag{5}$$

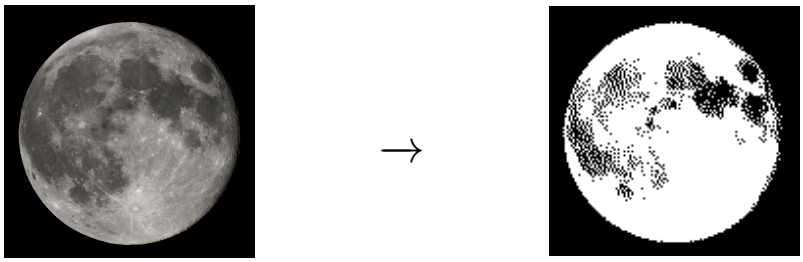
donde s' es la configuración que obtenemos cuando elegimos al azar un spin de la configuración s, por ejemplo, el spin (ij) con valor s_{ij}, y cambiamos su valor a $s'_{ij} = 1 - s_{ij}$. La expresión de la variación en el Hamiltoniano, ΔH, cuyo desarrollo viene detallado en el apartado (4.1) del apéndice, viene dada por:

$$\Delta H = \sum_{k}^{N}\sum_{l}^{N}\frac{1}{2}\omega_{ij,kl}s_{kl}(2s_{ij} - 1) + \theta_{ij}(1 - 2s_{ij}) \tag{6}$$

2. Resultados

2.1. Un solo patrón almacenado

En primer lugar, debemos encontrar un patrón que podamos introducir a nuestra red neuronal como entrada. Para ello, hemos tomado una imagen y la hemos transformado en un fichero, donde la representamos en formato binario, obteniendo así una matriz con unos y ceros, lo cual podemos ver representado en la figura (2). De esta forma, hemos generado el primer patrón que recordará nuestra red neuronal, la cual tendrá $128 \times 128 = 16384$ neuronas.



(a) Imagen inicial [5]. (b) Imagen transformada.

Figura 2: Procesado aplicado a la imagen (a) para transformarla en la imagen binaria (b).

2.1.1. Condición inicial aleatoria

Comenzamos imponiéndole al sistema un estado inicial completamente aleatorio. A continuación podemos ver la evolución del sistema utilizando el patrón previamente mencionado para una temperatura $T = 10^{-4}$.

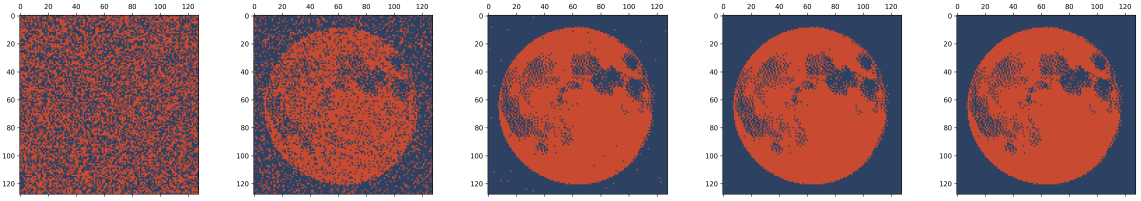


Figura 3: Evolución de nuestro sistema para temperatura $T = 10^{-4}$ con un patrón almacenado de dimensión 128×128. En concreto, se muestran las matrices obtenidas para las iteraciones (pasos Monte-Carlo) $\{0, 1, 5, 10, 20\}$.

Como podemos ver, a pesar de haber comenzado con un patrón completamente aleatorio, con tan solo un paso Monte-Carlo (definido como $N \times N$ iteraciones donde se elige una neurona al azar y se cambia o no su estado) la red neuronal es capaz de recordar el patrón almacenado, de manera que podemos distinguirlo a pesar de un cierto ruido en la imagen. Si dejamos al sistema evolucionar un poco más, podemos ver que a partir del paso 15, el sistema ha sido capaz de recrear el patrón prácticamente a la perfección.

Para cuantificar cómo la red se aproxima al patrón almacenado en función del tiempo, haremos uso del solapamiento, el cual se define para un patrón μ como:

$$m^{\mu}(s) = \frac{1}{N^2 a^{\mu}(1-a^{\mu})} \sum_{i=1}^{N} \sum_{j=1}^{N} (\xi_{ij}^{1} - a^{\mu})(s_{ij} - a^{\mu}) \in [-1, 1]$$

donde un solapamiento de 1 indica que el patrón se ha recuperado en su totalidad, 0 cuando no hay correlación entre el estado actual y el patrón almacenado, y -1 cuando se recupera el antipatrón (patrón con todos sus nodos invertidos). Para la configuración anterior, se ha obtenido la curva de solapamiento frente a los pasos Monte Carlo que podemos ver representada en la figura (4).

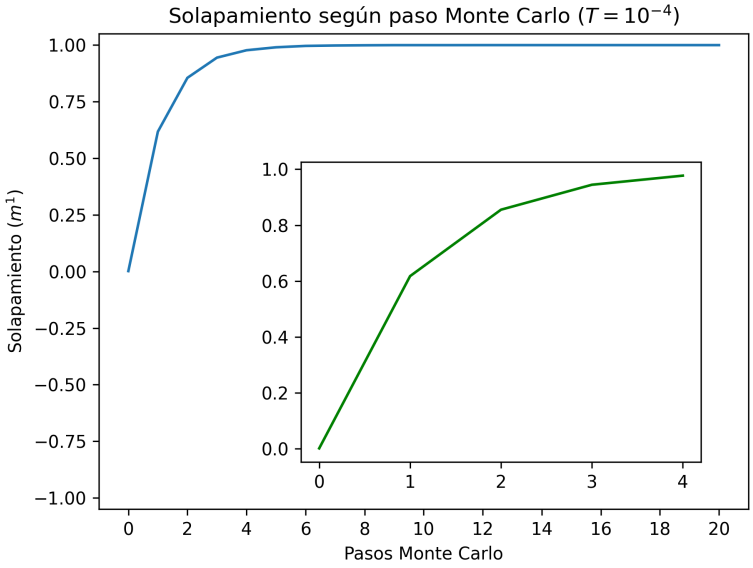


Figura 4: Solapamiento en función de los pasos Monte Carlo para temperatura $T = 10^{-4}$ para un sólo patrón almacenado.

Como podemos observar, el estado aleatorio tiene un solapamiento de 0, y con tan sólo un paso Monte Carlo, éste parámetro llega a ser superior a 0.5. Vemos también que con tan sólo 5 pasos, el solapamiento es muy cercano a la unidad ($m^{\mu}(s) = 0.99$), lo cual nos indica que la red es capaz de recordar el patrón con relativamente pocos pasos requeridos.

A continuación, vamos a estudiar el solapamiento de nuestro patrón en función de la temperatura del sistema. Para ello, hemos realizado la simulación partiendo de una condición inicial aleatoria para las temperaturas $T = \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$, obteniendo la gráfica (5), donde podemos observar el efecto que tiene la temperatura en este parámetro.

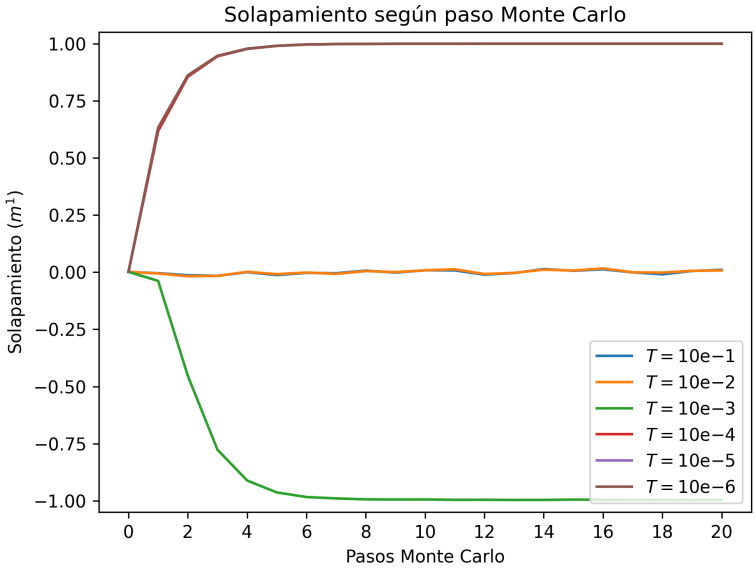


Figura 5: Solapamiento en función de los pasos Monte Carlo para cada una de las temperaturas consideradas y un sólo patrón almacenado.

Para temperaturas del orden de 10^{-2} o mayores, nuestro sistema no consigue recuperar el patrón inicial, y el solapamiento se mantiene cerca del valor 0. Para $T = 10^{-3}$, vemos que para las dos primeras iteraciones el solapamiento no cambia tan rápidamente como para temperaturas menores, pero a partir del paso 4 consigue recordar, en este caso, el antipatrón.

Conforme disminuimos la temperatura del sistema, podría parecer que la velocidad a la que llegamos a recuperar el estado almacenado es mayor. Si consideramos por ejemplo las curvas para $T = 10^{-6}$ y $T = 10^{-5}$, podemos ver que la primera crece ligeramente más rápido que la segunda. Sin embargo, esta diferencia es muy pequeña, por lo cual podemos concluir que, a partir de 10^{-4}, utilizar temperaturas más bajas no afecta sustancialmente a la velocidad de recuperación del patrón.

Resulta interesante estudiar el comportamiento del solapamiento para temperaturas entre 10^{-3} y 1, intervalo donde más cambio experimenta este parámetro. Considerando que, a partir del paso 8, el solapamiento llega prácticamente a su valor final y se estabiliza, se ha ejecutado la simulación 100 veces obteniendo la figura (6).

Para obtener la curva, hemos considerado una semilla inicial de forma que se recuerde el patrón con un solapamiento $m^1 = 1$, y se ha ido aumentando la temperatura y calculando el solapamiento tras un cierto número de pasos Monte Carlo. Alternativamente, se podría haber inicializado estados aleatorios distintos para cada temperatura. Sin embargo, se ha visto que este cambio no influye en la curva de solapamiento más que en algunos valores del solapamiento que serán negativos.

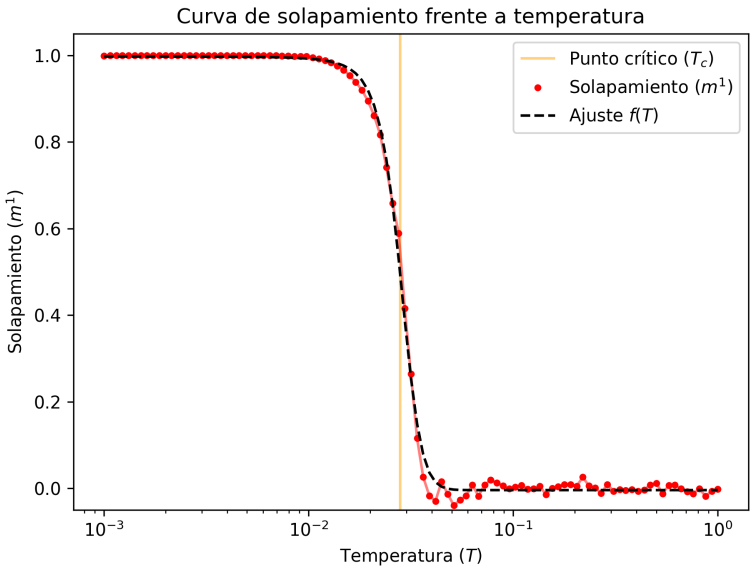


Figura 6: Solapamiento obtenido para un solo patrón almacenado frente a la temperatura, junto con el ajuste realizado.

Para determinar el comportamiento del solapamiento en función de la temperatura, se ha realizado un ajuste utilizando una función tangente hiperbólica, obteniendo los siguientes parámetros:

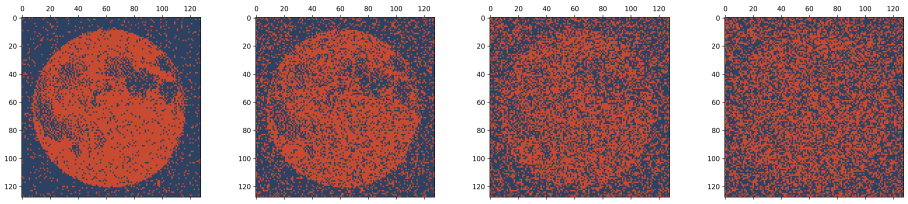
$$f(x) = A\tanh(Bx + C) + D = 0.501\tanh(-145.179x + 4.064) + 0.497$$

Podemos ver que los datos se ajustan muy bien a esta función, obteniendo un $\chi^2 = 0.0233$. Tomando un grado de significación de $\alpha = 0.05$ con 97 grados de libertad, vemos que el valor obtenido es mucho menor al tabulado, por lo cual podemos confirmar con un 95 % de confianza que nuestros datos siguen el modelo sugerido.

La función tangente hiperbólica planteada contiene un punto de inflexión en $T_c = 0.027996$, una temperatura que denominaremos crítica. Para esta temperatura, tras 8 pasos Monte Carlo el sistema ha sido capaz de recuperar el patrón con un solapamiento del 50 %. Para temperaturas superiores, el sistema no será capaz de recordar el patrón almacenado y el solapamiento decaerá a cero, mientras que para temperaturas menores si lo recordará.

2.1.2. Patrón deformado como condición inicial

Vamos a estudiar ahora la evolución del sistema para un estado inicial igual al patrón almacenado, aunque deformado. Para ello, hemos definido el parámetro λ, el cual nos indicará el grado de deformación. Para un valor $\lambda = 0$, el estado inicial será igual al patrón almacenado, mientras que para un valor $\lambda = 1$, se cambiarán de estado $N \times N$ neuronas (i, j) aleatorias. Podemos ver el efecto de este parámetro sobre el estado inicial en la figura (7).



(a) $\lambda = 0.1$ (b) $\lambda = 0.3$ (c) $\lambda = 0.6$ (d) $\lambda = 0.9$

Figura 7: Estado inicial a partir del patrón almacenado, deformado según el parámetro λ.

De nuevo, hemos ejecutado la simulación para los mencionados valores de λ para así obtener la figura (8), donde se muestra el solapamiento en función de los pasos Monte Carlo para cada grado de deformación.

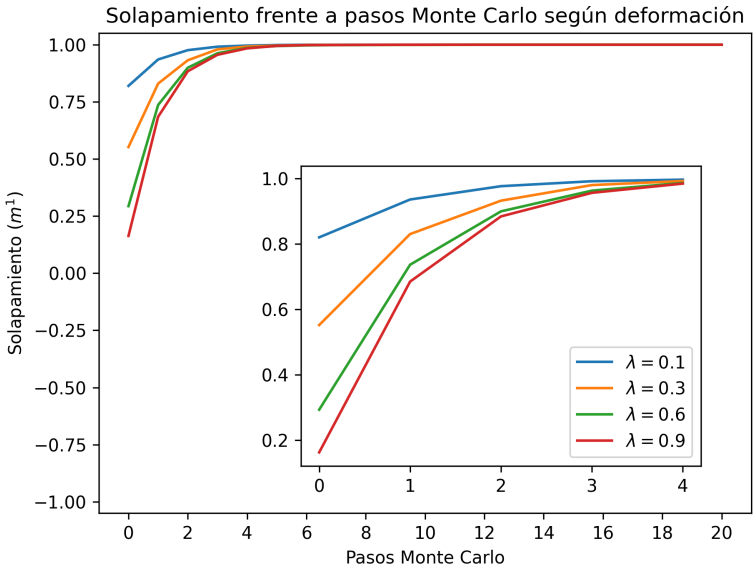


Figura 8: Solapamiento en función de los pasos Monte Carlo para cada uno de los patrones iniciales deformados, con un parámetro de deformación λ, con una temperatura de $T = 10^{-4}$.

Vemos que el parámetro λ afecta principalmente al solapamiento inicial (como era de esperar, para $\lambda = 0.1$, el solapamiento es muy cercano a 1). Consecuentemente, la velocidad a la que se consigue recuperar el patrón también será más alta para configuraciones iniciales con un grado de deformación bajo. Sin embargo, comparando con la gráfica (4), este parámetro no parece influir de manera significativa en el comportamiento general del sistema más allá del instante inicial.

Al igual que hicimos en el apartado anterior, vamos a estudiar el comportamiento del solapamiento del patrón frente a la temperatura. Realizando el mismo procedimiento que el explicado anteriormente, llegamos a los resultados expuestos en la figura (9).

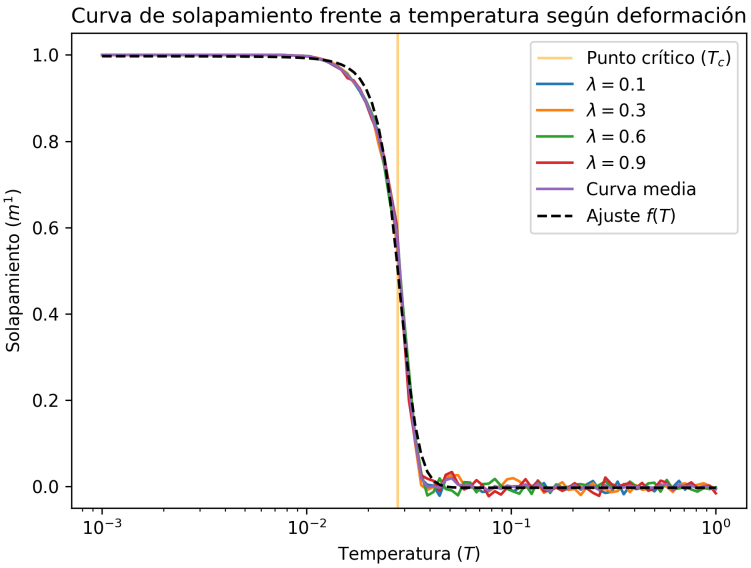


Figura 9: Curva de solapamiento en función de la temperatura para varios parámetros de deformación, junto con el ajuste realizado, la curva media, y la temperatura crítica.

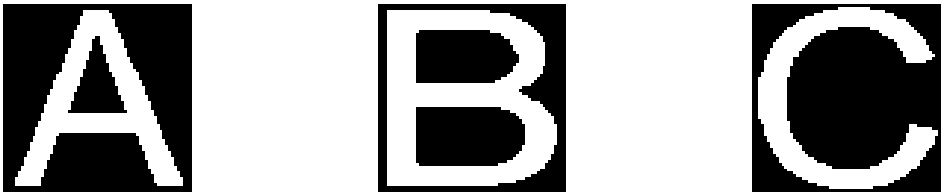
Podemos ver que las curvas para los distintos grados de deformación λ no difieren significativamente unas de otras, por lo cual hemos aplicado el ajuste lineal a una curva cuyos valores de solapamiento son el valor medio de todas las demás curvas. De esta forma, hemos obtenido los siguientes parámetros:

$$f(x) = A\tanh(Bx + C) + D = 0.499\tanh(-148.497x + 4.150) + 0.497$$

Como podemos ver, los parámetros del ajuste son muy similares, con una temperatura crítica de $T_c = 0.027947$, ligeramente menor a la obtenida para un estado inicial aleatorio aunque no lo suficiente como para poder concluir una relación definitiva. Además, el parámetro $\chi^2 = 0.0174$ sigue siendo menor al tabulado, por lo cual nuestros datos se ajustan al modelo planteado con un 95 % de confianza.

2.2. Varios patrones almacenados

Para estudiar la red neuronal de Hopfield con varios patrones, se han considerado (inicialmente) tres patrones de dimensiones 64×64 correspondientes a las letras del alfabeto [1]. Nuestro objetivo final, además de estudiar el comportamiento del sistema utilizando un procedimiento similar al detallado en los apartados anteriores para un sólo patrón, será conseguir que la red pueda reconocer correctamente la letra correspondiente a partir de una imagen con la letra deformada o manuscrita.



(a) Patrón 1. (b) Patrón 2. (c) Patrón 3.

Figura 10: Patrones almacenados de las tres primeras letras del alfabeto.

2.2.1. Condición inicial aleatoria

Consideramos una temperatura de $T = 10^{-4}$. Si dejamos que el sistema evolucione a partir de un estado inicial aleatorio, el sistema evolucionará hasta recordar uno de los tres patrones almacenados. Resulta interesante en este caso estudiar el solapamiento de nuestro sistema con cada uno de los patrones, lo cual podemos ver representado en la figura (11).

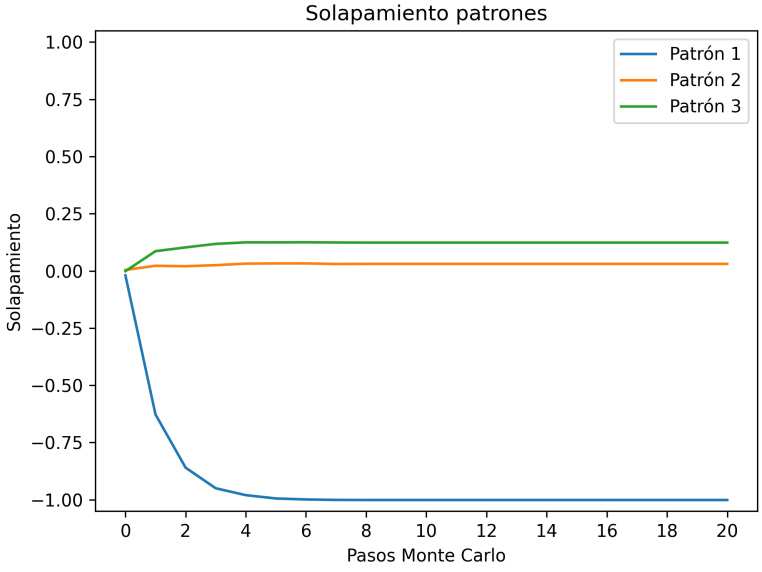


Figura 11: Solapamientos obtenidos para tres patrones almacenados con $T = 10^{-4}$ en función del tiempo, en pasos Monte Carlo.

[1] Para solventar, dentro de lo posible, el problema expuesto en el apartado 2.2.4, se han tomado los patrones con muy poco margen a los lados, de forma que la diferencia entre los patrones es mayor que si hubiésemos dejado la letra de un tamaño menor en el centro del recuadro.

Podemos ver que el comportamiento es muy similar al caso de un sólo patrón: conforme el sistema evoluciona, la red va recordando uno de los patrones hasta que, tras aproximadamente 8 pasos Monte Carlo, el solapamiento es casi la unidad en valor absoluto. Algo curioso que podemos observar es que el solapamiento con el patrón 1 no se mantiene nulo. Esto se debe a que hay cierto parecido entre ambos patrones, sin embargo esto no supone un problema ya que son lo suficientemente diferentes como para que la red pueda distinguirlos.

Realizando la simulación para varias temperaturas distintas, vemos que la red deja de recordar el patrón cuando la temperatura es del orden de $T \sim 0.1$. Por tanto, vamos a realizar, al igual que antes, 100 simulaciones con temperaturas en el rango $[10^{-4}, 1]$ y 8 pasos Monte Carlo para estudiar el comportamiento de los solapamientos en función de la temperatura. Esta curva de solapamiento la podemos encontrar representada en la gráfica (12).

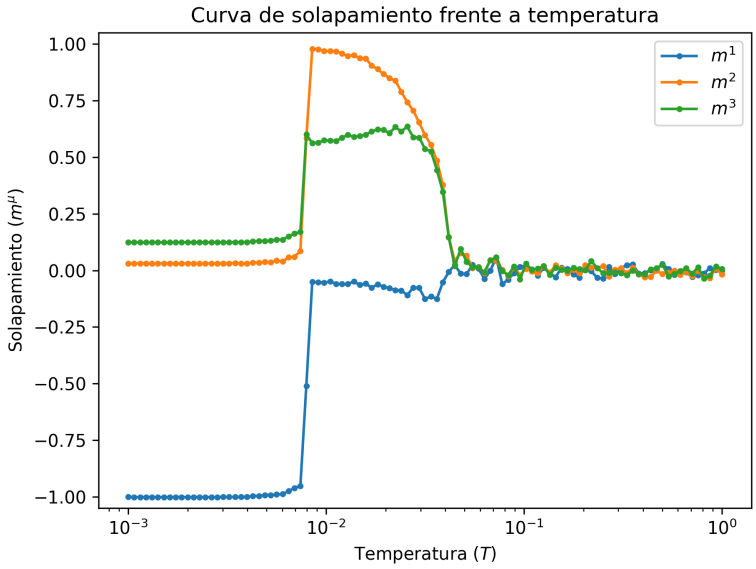


Figura 12: Solapamientos de cada patrón en función de la temperatura.

Vemos que la curva de solapamiento tiene tres zonas claramente diferenciadas, caracterizadas por dos valores de temperaturas $T_1 \approx 7.943 \cdot 10^{-3}$ y $T_2 \approx 4.467 \cdot 10^{-2}$.

Para temperaturas inferiores a T_1, la red es capaz de recordar uno de los patrones. Para temperaturas en la zona intermedia entre las dos temperaturas características definidas, ocurre una transición donde el patrón que inicialmente se recordó pasa a tener un solapamiento cercano al cero, mientras que los solapamientos con los otros dos patrones aumentan. Finalmente, para temperaturas mayores a T_2, los solapamientos con todos los patrones decaen a cero y la red es incapaz de recordar el patrón.

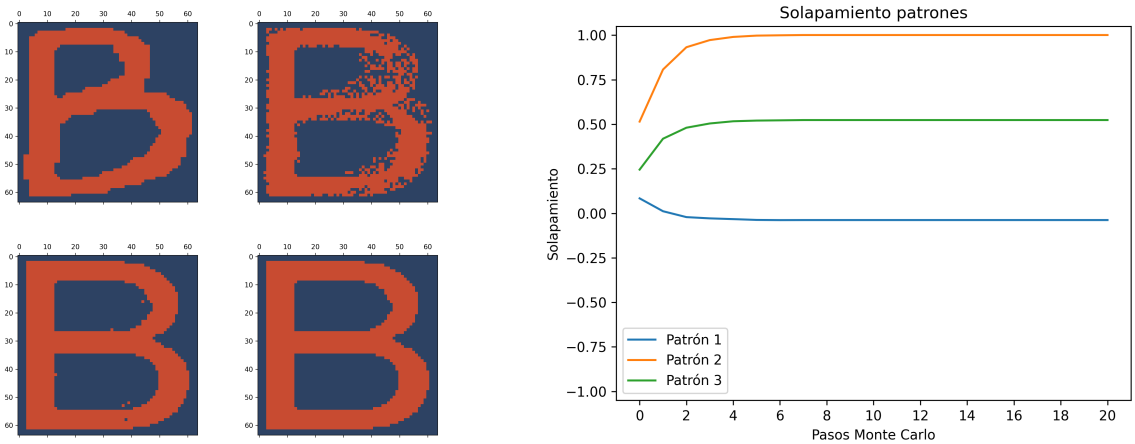
2.2.2. Uno de los patrones deformados como condición inicial

Para estudiar el comportamiento de la red neuronal con una entrada igual a uno de los patrones deformados, y debido a que ya estudiamos el comportamiento de la red frente a deformaciones iniciales generadas aleatoriamente mediante un parámetro de deformación, vamos a considerar cada una de las letras escritas a mano (figura (13)).

(a) Patrón 1 deformado. (b) Patrón 2 deformado. (c) Patrón 3 deformado.

Figura 13: Patrones deformados (escritos a mano).

En la figura (14) podemos ver el comportamiento del sistema cuando le introducimos un patrón inicial igual al patrón 2 deformado. Observamos que el sistema es capaz de recordar el patrón correcto con bastante rapidez.



(a) Matrices obtenidas para los pasos $\{0, 1, 5, 10\}$. (b) Solapamiento de cada patrón.

Figura 14: Comportamiento del sistema con tres patrones donde podemos ver el solapamiento de los tres patrones.

Al igual que para el caso del estado inicial aleatorio, vamos a estudiar el solapamiento entre los patrones en función de la temperatura del sistema. De esta forma, obtenemos la figura (15), la cual nos muestra que el sistema es capaz de reconocer el patrón para temperaturas menores a 10^{-2} de manera muy eficaz. A partir de este orden de temperatura, el patrón 2 decae ligeramente mientras el patrón 3 parece subir; sin embargo, a partir de una temperatura crítica de $T \approx 3.9 \cdot 10^{-2}$, el solapamiento entre ambos patrones decae rápidamente, y el sistema deja de ser capaz de recordar cualquiera de los patrones, de manera que el solapamiento entre los tres patrones oscila en torno al valor nulo.

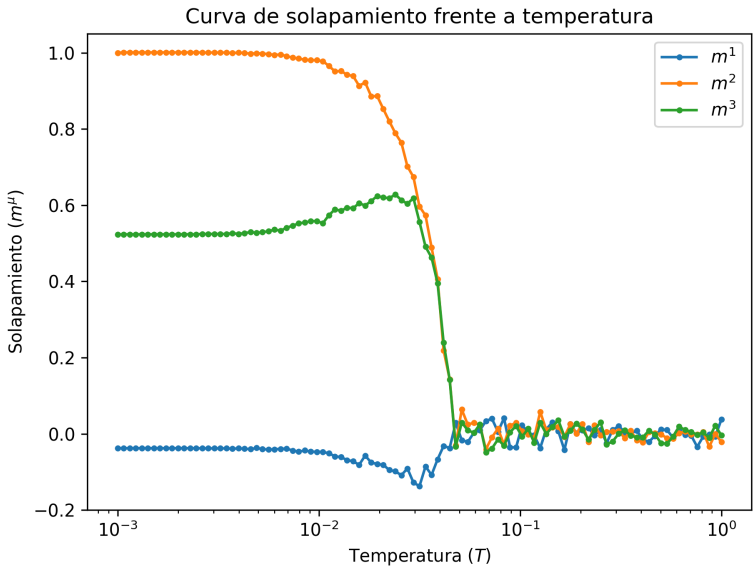


Figura 15: Solapamientos de cada patrón en función de la temperatura.

Para determinar la eficacia de nuestra red neuronal para reconocer las letras, hemos realizados numerosas simulaciones para $T = 10^{-4}$ y para cada una de las letras escritas a mano que además serán deformadas con un parámetro de deformación λ elegido aleatoriamente. En el gráfico (16) podemos ver representado el solapamiento medio para cada una de las letras iniciales deformadas.

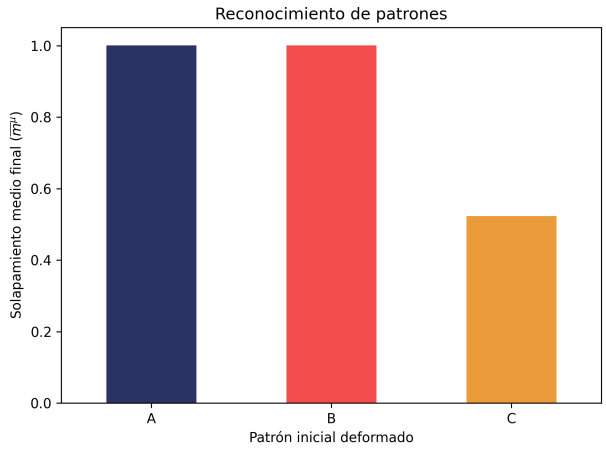


Figura 16: Solapamiento medio para cada patrón.

Como podemos ver, la red consigue recordar con bastante eficacia los patrones de las letras A y B, con un solapamiento final medio prácticamente la unidad. Sin embargo, el solapamiento medio obtenido para la letra C es mucho menor, apenas llega al 50 %. Esto puede deberse a que, con este estado inicial, el estado final suele ser un estado superposición de las otras letras. Este fenómeno lo describiremos más a fondo en el siguiente apartado (2.2.4).

Esta red la podríamos ampliar para contener todas las letras del alfabeto. Sin embargo, si lo hacemos con los patrones almacenados que mostramos anteriormente, es muy común que el sistema converja en estados superposición, y no se consiga reconocer el patrón correcto. Para solventar esto, podríamos utilizar unos patrones iniciales que se diferencien más entre sí, utilizar otros símbolos (como, por ejemplo, los números del 0 al 9 [6]), o bien colocar cada letra en un recuadro situado en una posición determinada de la matriz, de manera que las patrones almacenados sean ortogonales. Además, para obtener unos resultados mejores, también podríamos considerar combinar la red neuronal de Hopfield con otras redes [7].

2.2.3. Recuperación de la memoria según el número de patrones almacenados

A lo largo de esta sección, vamos a estudiar la relación que existe entre la recuperación de la memoria con el número de patrones almacenados. Para ello, fijaremos una temperatura de $T = 10^{-4}$ con un estado inicial aleatorio. Además, para obtener los patrones que almacenaremos, utilizaremos matrices 20×20 aleatorias.

Vamos a considerar que un patrón se consigue recordar sin apenas error cuando su solapamiento cumple:

$$m^{\mu} \geq 0.75$$

Para estudiar la recuperación de la memoria, vamos a ejecutar la red neuronal varias veces con un número de patrones en el intervalo $n_{pat} = [0, 1, 2, ..., 50]$. Para asegurarnos de que los solapamientos finales se mantienen estables, se han realizado además 100 pasos Monte Carlo por simulación [2].

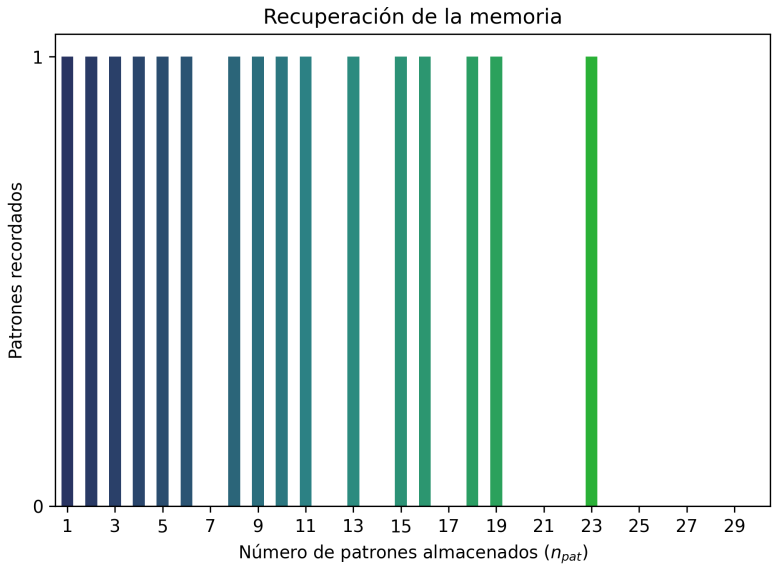


Figura 17: Número de patrones recordados en función del número de patrones almacenados.

Como podemos ver en la figura (17), el sistema es capaz de recordar un único patrón para un número de patrones almacenados pequeño (por debajo de 5 patrones, siempre es capaz de recordar un patrón). Sin embargo, conforme aumentamos n_{pat}, van apareciendo cada vez con mas frecuencia valores para los cuales el sistema no es capaz de recordar ningún patrón. Finalmente, para $n_{pat} \geq 24$, el sistema ya no es capaz de recordar ninguno de los patrones almacenados inicialmente.

A partir de estos datos podemos determinar también la fracción máxima de patrones que la red podrá almacenar de manera que todos los patrones se puedan recordar:

$$\alpha_c = \frac{P_c}{N^2}$$

[2]Hemos considerado 100 pasos Monte Carlo ya que, graficando el solapamiento para 50 patones, es donde mayor estabilidad hay. Dado que trabajaremos con un número menor de patrones, consideramos que 100 pasos deben ser más que suficientes.

De esta forma, obtenemos la gráfica (18), que es simplemente un reescalamiento de la gráfica (17). Como podemos ver, la fracción toma un valor $\alpha_c = 0.003$ cuando el patrón es recordado con un solapamiento mayor al 75 %, y es nula para los demás puntos.

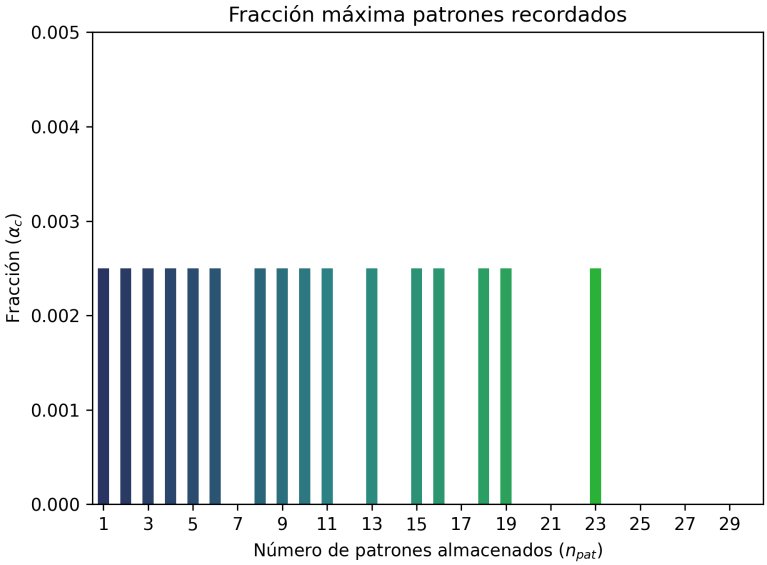


Figura 18: Fracción máxima de patrones que la red podrá almacenar de manera que todos los patrones se puedan recordar en función del número de patrones almacenados.

2.2.4. Estados de superposición

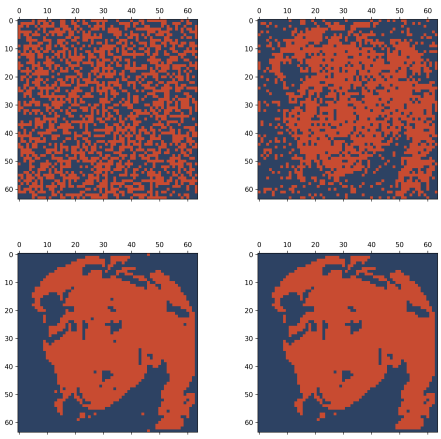
Inicialmente, para estudiar la red neuronal con varios patrones almacenados, hemos elegido los siguientes tres patrones de $64 \times 64 = 4096$ neuronas.

(a) Patrón 1. (b) Patrón 2. (c) Patrón 3.

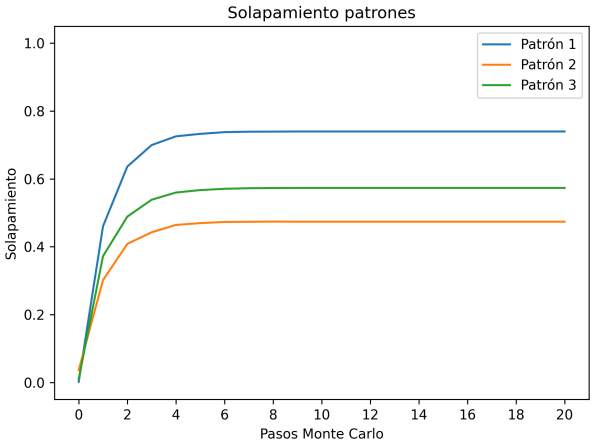
Figura 19: Procesado aplicado a la imagen (a) para transformarla en la imagen binaria (b).

Partiendo de un estado inicial aleatorio con una temperatura de $T = 10^{-4}$, hemos obtenido resultados similares a los del apartado anterior: tras un número relativamente bajo de pasos Monte Carlo (en torno a 8), la red consigue recordar uno de los patrones con un solapamiento muy cercano a la unidad.

Sin embargo, conforme ejecutamos la simulación varias veces, en ocasiones observamos que el sistema converge a un estado que no es exactamente igual a ninguno de los patrones, sino una superposición de varios. Este comportamiento lo podemos ver en la figura (20).



(a) Matrices obtenidas para los pasos $\{0, 1, 5, 10\}$.



(b) Solapamiento de cada patrón.

Figura 20: Comportamiento del sistema con tres patrones donde podemos ver el solapamiento de los tres patrones.

Este resultado inusual puede explicarse a partir de la ortogonalidad de los patrones almacenados. Dos patrones se dicen ortogonales si difieren en $N/2$ bits [8]. Debido a la similitud entre los tres, una superposición de los patrones, que se corresponde con la salida obtenida de nuestra red neuronal, es un mínimo local de energía. Para solventar este problema, es preferible utilizar patrones donde la diferencia entre sus bits sea mayor, evitando así las salidas no deseadas [9].

3. Conclusiones

A lo largo de este informe, se ha estudiado el modelo de red neuronal de Hopfield y las distintas características de esta red neuronal artificial mediante la implementación del algoritmo de Metrópolis.

En primer lugar, hemos estudiado el comportamiento de la red con un único patrón almacenado, y hemos visto cómo la red es capaz de recordar este patrón a partir de una condición inicial completamente aleatoria, al igual que con condiciones iniciales iguales al patrón pero deformado un cierto grado.

Para cada uno de estos casos, hemos determinado también la relación entre el parámetro del solapamiento en función de los pasos Monte Carlo realizados en cada simulación y de la temperatura del sistema. Para esta última, hemos conseguido además obtener una expresión analítica, una función tangente hiperbólica, que consigue predecir correctamente la temperatura crítica del sistema, a partir de la cual la red ya no consigue recrear el patrón.

Se ha observado que el sistema, para $N = 64$ y $T = 10^{-4}$, converge muy rápidamente, y con tan sólo 8 pasos Monte Carlo ya es capaz de recordar el patrón almacenado prácticamente sin error. También, se ha concluido que el grado de deformación del patrón no influye significativamente en la curva de solapamiento, afectando solamente al solapamiento inicial que presenta el sistema. Además, se ha concluido que disminuir la temperatura más allá de 10^{-4} no mejora la velocidad a la que converge el sistema de una manera importante.

En segundo lugar, hemos estudiado, siguiendo el mismo procedimiento, la red neuronal de Hopfield para varios patrones almacenados. Para ello, hemos considerado 3 patrones que el sistema ha sido capaz de recordar de manera correcta. Además, hemos obtenido la curva de solapamiento frente a la temperatura para cada uno de los patrones, y se han conseguido identificar las temperaturas características para las cuales el sistema consigue recuperar o no los patrones almacenados.

Se ha intentado aplicar esta red a un simple programa que permita reconocer una letra a partir de su imagen deformada o escrita a mano. Para dos de los patrones introducidos, la red ha sido capaz de reconocer la letra correcta con bastante porcentaje de acierto. Sin embargo, para uno de los patrones, el sistema parecía no conseguir reconocerlo, y el estado final obtenido era una superposición de varios patrones.

También, se ha realizado un breve estudio de la capacidad de recuperación de la memoria del sistema en función del número de patrones almacenados. En particular, hemos encontrado que para un número bajo de patrones almacenados (menor de 20), el sistema suele ser capaz de recordar uno de los patrones, excepto en casos puntuales. Sin embargo, si aumentamos el número de estados almacenados, el número de patrones que el sistema consigue recordar pasa a cero de manera muy rápida.

Finalmente, se ha intentado encontrar una explicación del fenómeno de los llamados estados superposición a partir de los patrones iniciales almacenados, basándonos en que son mínimos locales de la energía, y se ha intentado determinar la condición necesaria que los patrones deben cumplir para evitar estos estados indeseados: su ortogonalidad.

Podemos concluir por tanto que el modelo de red neuronal de Hopfield, a pesar de ser relativamente básico y contener bastantes limitaciones, tiene varias aplicaciones físicas útiles, principalmente en el ámbito del reconocimiento de patrones.

4. Apéndice

4.1. Expresión de la variación en el Hamiltoniano ΔH

El Hamiltoniano para el estado s, donde hemos fijado un valor (ij) aleatorio, viene dado por:

$$H(s) = -\frac{1}{2}\sum_{k}^{N}\sum_{l}^{N}\omega_{ij,kl}s_{ij}s_{skl} + \theta_{ij}s_{ij}$$

Para el estado s', hemos fijado un valor (ij) aleatorio y hemos cambiado el valor de su spin de forma que $s'_{ij} = 1 - s_{ij}$. Por tanto, el Hamiltoniano será:

$$H(s') = -\frac{1}{2}\sum_{k}^{N}\sum_{l}^{N}\omega_{ij,kl}(1 - s_{ij})s_{skl} + \theta_{ij}(1 - s_{ij})$$

Vamos a calcular la variación del Hamiltoniano:

$$\begin{aligned}\Delta H = H(s') - H(s) = &-\frac{1}{2}\sum_{k}^{N}\sum_{l}^{N}\omega_{ij,kl}(1-s_{ij})s_{skl} + \theta_{ij}(1-s_{ij}) \\ &-\left(-\frac{1}{2}\sum_{k}^{N}\sum_{l}^{N}\omega_{ij,kl}s_{ij}s_{skl} + \theta_{ij}s_{ij}\right)\end{aligned}$$

Operando llegamos a la expresión final utilizada (6):

$$\begin{aligned}\Delta H =& \sum_{k=1}^{N}\sum_{l=1}^{N}\left(-\frac{1}{2}\omega_{ij,kl}(1-s_{ij})s_{kl} + \frac{1}{2}\omega_{ij,kl}s_{ij}s_{kl}\right) + \theta_{ij}(1-s_{ij}) - \theta_{ij}s_{ij} \\ =& \frac{1}{2}\sum_{k=1}^{N}\sum_{l=1}^{N}\left(w_{ij,kl}(s_{ij}-1)s_{kl} + w_{ij,kl}s_{ij}s_{kl}\right) + \theta_{ij}(1-s_{ij}) - \theta_{ij}s_{ij} \\ =& \frac{1}{2}\sum_{k=1}^{N}\sum_{l=1}^{N}\left(\omega_{ij,kl}s_{kl}(2s_{ij}-1)\right) + \theta_{ij}(1-2s_{ij}) \quad \square\end{aligned}$$

4.2. Programas utilizados

Para obtener los resultados obtenidos en cada uno de los apartados de este informe, hemos utilizado varios programas, que enumeraremos a continuación y describimos brevemente su funcionamiento. En la carpeta comprimida *Hopfield.zip* adjunta a este informe, podemos encontrar los siguientes ficheros.

```
/Hopfield.zip/
├── single pattern/ ..... Apartado (2.1)
│   ├── gsl_rng.h ..... Librería para generar números aleatorios.
│   ├── hopfield.cpp ..... Programa red neuronal de Hopfield para un patrón.
│   ├── hopfield_graph.py ..... Animaciones y gráficas red neuronal.
│   ├── hopfield_sol.cpp ..... Programa curva de solapamiento.
│   ├── hopfield_sol.py ..... Gráfica curva de solapamiento
│   ├── image.py ..... Generador matriz binaria a partir de foto.
│   ├── moon.jpg ..... Foto usada en el informe.
│   └── moon.txt ..... Fichero patrón usado.
└── multi pattern/ ..... Apartado (2.2)
    ├── a.txt, b.txt, c.txt, a_dist.txt, b_dist.txt, c_dist.txt ..... Ficheros letras
    ├── gsl_rng.h ..... Librería para generar números aleatorios.
    ├── curva_solapamiento.cpp ..... Programa para generar curva de solapamiento
    ├── curva_solapamiento.pt ..... Gráficas curva solapamiento
    ├── hopfield.cpp ..... Programa red Hopfield varios patrones
    ├── hopfield.py ..... Gráficas red Hopfield varios patrones.
    ├── memoria.cpp ..... Apartado (2.2.3)
    ├── memoria.cpp ..... Gráficas memoria
    ├── reconocimiento.cpp ..... Apartado (2.2.2)
    └── reconocimiento.py ..... Gráficas reconocimiento
```

Cada programa está debidamente comentado, de manera que ejecutándolos se debería poder obtener todos los resultados expuestos en este informe. Para los códigos de C++, es necesario instalar la librería *GSL Gnu Scientific Library* e incluir su ruta a la hora de compilar. Los programas para obtener las gráficas se han realizado con Python3.

5. Referencias

[1] Zawersch. A hopfield net with four units. CC BY-SA 3.0 `https://upload.wikimedia.org/wikipedia/commons/4/44/Hopfield-net-vector.svg`.

[2] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. U. S. A.*, 79(8):2554–2558, April 1982.

[3] Georg Stimpfl-Abele and Lluís Garrido. Fast track finding with neural networks. *Computer Physics Communications*, 64(1):46–56, 1991.

[4] Hermann Kolanoski. Application of artificial neural networks in particle physics. *Artificial Neural Networks — ICANN 96*, page 1–14, 1996.

[5] Luc Viatour. Full Moon. CC BY-SA 3.0 `https://upload.wikimedia.org/wikipedia/commons/d/dd/Full_Moon_Luc_Viatour.jpg`.

[6] Tarun Varshney. Noise corrupted pattern recognition using hopfield neural network. 01 2009.

[7] Poonam Dabas and Umesh Kumar. Pattern recognition using artificial neural network. *International Journal of Computer Applications Technology and Research*, 3(6):358–360, 2014.

[8] Hao Dong. Energy-based models - Hopfield Network. Peking University `https://deep-generative-models.github.io/files/ppt/2020/Lecture`.

[9] ESCOM. Redes Neuronales de Hopfield. `https://es.slideshare.net/mentelibre/redes-neuronales-de-hopfield`.