Red Neuronal: Modelo de Hopfield.

Eduardo Pérez Álvarez.

2 de julio de 2018

Física Computacional - Grado en Físca - U.G.R.

1. Introducción.

Las Redes Neuronales son un campo muy importante dentro de la Inteligencia Artificial. Inspirándose en el comportamiento conocido del cerebro humano (principalmente el referido a las neuronas y sus conexiones), trata de crear modelos artificiales que solucionen problemas difíciles de resolver mediante técnicas algorítmicas convencionales.

Desde la década de los 40, en la que nació y comenzó a desarrollarse la informática, el modelo neuronal la ha acompañado. De hecho, la aparición de los computadores digitales y el desarrollo de las teorías modernas acerca del aprendizaje y del procesamiento neuronal se produjeron aproximadamente al mismo tiempo, a finales de los años cuarenta.

Desde entonces hasta nuestros días, la investigación neurofisiológica y el estudio de sistemas neuronales artificiales (ANS, Artificial Neural Systems) han ido de la mano. Sin embargo, los modelos de ANS no se centran en la investigación neurológica, si no que toma conceptos e ideas del campo de las ciencias naturales para aplicarlos a la resolución de problemas pertenecientes a otras ramas de las ciencias y la ingeniería.

Se puede decir que la tecnología ANS incluye modelos inspirados por nuestra comprensión del cerebro, pero que no tienen por qué ajustarse exactamente a los modelos derivados de dicho entendimiento.
Los primeros ejemplos de estos sistemas aparecen al final de la década de los cincuenta. La referencia histórica más corriente es la que alude al trabajo realizado por Frank Rosenblatt en un dispositivo denominado perceptrón. Hay otros ejemplos, tales como el desarrollo del Adaline por el profesor Bernard Widrow.

Durante todos estos años, la tecnología ANS no siempre ha tenido la misma consideración en las ramas de la ingeniería y las ciencias de la computación, más ansiosas de resultados que las ciencias neuronales. A partir de 1969, el pesimismo debido a las limitadas capacidades del perceptrón hizo languidecer este tipo de investigación.

A principios de los 80, por un lado Hopfield y sus conferencias acerca de la memoria autoasociativa y por otro lado la aparición del libro Parallel Distributed Processing (PDP), escrito por Rumelhart y McClelland reactivaron la investigación en el campo de las redes neuronales. Hubo grandes avances que propiciaron el uso comercial en campos tan variados como el diagnóstico de enfermedades, la aproximación de funciones o el reconocimiento de imágenes.

En los últimos años se han conseguido grandes avances gracias a la mejora de los ordenadores y al uso de GPUs para este tipo de computaciones.
Un ejemplo importante de aplicación de ellas viene de la mano de Google cuando montó Street View: una red neuronal convolucional que lograba una precisión del 96 % a la hora de reconocer números de calle en las imágenes que toman sus coches.
También han sido usadas por Android en los reconocimientos de voz.

2. Fundamento teórico.

El cerebro humano es capaz de desarrollar tareas que ni los ordenadores más potentes diseñados hasta la fecha son capaces de emular. Por ejemplo, en tareas tan ordinarias hoy en día como la de reconocer, después de muchísimos años, a un amigo o identificar y escribir imágenes. Su unidad básica son las neuronas.

Obviando toda complejidad, la función de la neurona consiste esencialmente en disparar cuando la suma de señales que le llegan desde otras neuronas supera un determinado umbral de excitación y, por lo que respecta a fenómenos debidos a la cooperación, las neuronas se pueden imaginar como variables binarias.

Hopfield, en su modelo considera una red con N neuronas binarias $s_i = 0, 1$, donde los dos estados corresponden a reposo y disparo respectivamente. Cada neurona está relacionada con todas las demás mediante sinapsis cuyos "pesos" o "fuerza" respectivos vienen descritos por una variable real w_{ij} . Aquí los índices i y j, cambian de 1 a N para describir todas las neuronas y sus relaciones sinápticas. En cada instante discreto t , el estado del sistema, llamado configuración del sistema o de la red de neuronas, se caracteriza mediante el conjunto de valores de las actividades de las neuronas y pesos sinápticos. Suponiendo que los pesos sinápticos son siempre los mismos, el modelo queda determinado al detallar la dinámica de las neuronas mediante una regla para los disparos y un método para ir actualizando las actividades.

Dicha regla es la llamada regla de Hebb, según la cual el fortalecimiento de algunas sinapsis (y por tanto, el aumento relativo del peso) correspondiente es consecuencia de la activación repetida de las neuronas que conectan, mientras que las sinapsis entre neuronas predominantemente inactivas tenderían a debilitarse y, eventualmente, desaparecerían.

Supongamos, por ejemplo, una neurona i conectada a otras muchas que representamos genéricamente por j , como se muestra en la imagen:

En este diagrama, la neurona i recibe una señal de cada una de las otras que se verá modulada por el peso sináptico de cada una de las conexiones que las une, por lo que la señal que llegue a la neurona i desde la neurona j , será $w_{ij} \cdot s_j$. De esta forma que la señal depende de la actividad de la neurona j (tan solo llegara señal a la neurona i, si la neurona j esta disparando) y de la eficacia de la transmisión w_{ij} , que puede ser negativa (cuando la sinapsis es inhibidora). Sumando todas estas señales para todo valor de j , se obtiene la señal total o corriente sináptica neta, h_i. Con esto se quiere decir que la señal en i sera igual a la suma de todos los pesos sinápticos cuando todas las demás neuronas estén activas, $s_j = 1$; en caso de que las neuronas estén en reposo $s_j = 0$, no aportaran nada a la señal total.

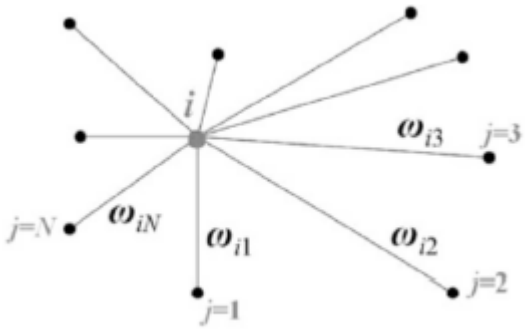

Figura 1: Esquema simplicado de la estructura que tiene una red neuronal de tipo Hopfield. En ella una neurona esta conectada con el resto y sus conexiones se indentifican con la variable w_{ij} , indicando así la eficacia con la que puede transmitirse información.

La regla básica para introducir una dinámica en el sistema consiste en inducir al disparo neuronas que están en un estado silencioso de reposo. Supongamos que tenemos una neurona en el instante t en estado de reposo. Para activar dicha neurona necesitamos que la señal neta que llegue a la neurona desde sus vecinos supere un umbral, específico de cada neurona, θ_i. Esto se puede modelar con una función de tipo escalón en la que si la señal total supera al umbral ($h_i > \theta_i$) la neurona puede disparar, y en caso contrario ($h_i < \theta_i$) se quedará en reposo. Inspirándose en los estudios del magnetismo, esta regla de actualización se hace probabilística, esto es, en lugar de hacer el cambio de s_i siempre que $h_i > \theta_i$, se hace con una probabilidad que depende de la diferencia entre $h_i \, y \, \theta_i$ o mas concretamente de $\beta(h_i - \theta_i)$ donde β es un parámetro y cuya $T = 1/\beta$, denominado temperatura, controla la estocasticidad del proceso. En la práctica se procede de la siguiente manera: partiendo de una conguración inicial dada para todas las actividades s_i, se calculan todas las señales h_i. Esto permite decidir que actividades han de intercambiarse usando la regla de disparo especificada. Una vez actualizados los estados s_i de las neuronas, se repite el calculo de las h_i para la nueva configuración de la red. La actualización

de la red se hace, en este caso, de manera secuencial: visitando las neuronas de una en una de forma aleatoria cambiando solo la actividad del lugar visitado a cada paso.
Los pesos sinápticos vienen dados por la regla del aprendizaje hebbiano:

$$w_{ij} = \frac{1}{a(1-a)N} \sum_{k=1}^{m} \left(\xi_i^k - a\right)\left(\xi_j^k - a\right) \tag{1}$$

Donde $\xi_i^k = 0, 1$, con i = 1, ... , N son cada una de las m configuraciones particulares, y a el promedio de ellas.

Cuando las sinapsis se construyen de este modo, esto es, como combinación lineal de una serie de patrones, se dice que estos patrones están almacenados en la red de neuronas. Este es el mecanismo del modelo para aprender y acumular experiencias. Es bastante general ya que los patrones pueden representar, además de imágenes, cualquier tipo de información compleja.

No se permiten autoconexiones, $w_{ii} = 0$. Los umbrales de disparo vienen dados por

$$\theta_i = \frac{1}{2} \sum_{j}^{N} w_{ij} \tag{2}$$

Con todo ello ya se puede escribir el hamiltoniano del sistema:

$$H = -\frac{1}{2} \sum_{i,j}^{N} w_{ij} s_i s_j + \sum_{i}^{N} \theta_i s_i \tag{3}$$

A partir del mismo, aplicando el método Monte Carlo y basándonos en el modelo de Ising, se modela para la programación.

Para ello, se sigue el siguiente algoritmo:

1. Se elige una neurona k al azar de la red, s_k.
2. Se evalua el valor $p = mín(1, exp(-\Delta H/T))$, donde ΔH queda

$$\Delta H = (2s_k - 1)\left[\frac{1}{2} \sum_{i=1, i\neq k}^{N} w_{ik} s_i - \frac{1}{2} \sum_{j=1, j\neq k}^{N} w_{kj} s_j + \theta_k\right] \tag{4}$$

3. Se genera un número aleatorio uniforme $0 < q < 1$. Si q $<$ p entonces

$$s_k = 1 - s_k \tag{5}$$

4. Ir a (1)

De esta forma se va actualizando la red.

A partir del solapamiento,

$$g^k = \frac{1}{a(1-a)N} \sum_{i=1}^{n} \left(\xi_i^k - a\right)\left(\xi_i^k - \frac{1}{2}\right) \tag{6}$$

Se puede observar como evoluciona la red. Cuando esta converge a un patrón concreto, ξ^k, toma el valor de 1,

Esto se entenderá mucho mejor cuando se vea, en la parte de resultados.

3. Descripción del programa y método.

El programa se ha desarrollado en lenguaje C++, utilizando un *buen generador de números aleatorios* proporcionado por el Departamento de Electromagnetismo y Física de la Materia Condensada de la Universidad de Granada, y que se basa en algoritmos tipo x(i)=x(i-p).xor.x(i-q).
Para la adquisición de imágenes en formato binario (ceros y unos), se ha utilizado la página "https://www.dcode.fr/binary-image" en la que introduciendo cualquier imagen *sencilla* la devuelve en formato de escritura en unos y ceros; y en el tamaño que se desee, que en general ha sido 20x20 o 30x30, ya que en el programa se ha trabajado con 400 o 900 neuronas principalmente.

Una vez las imágenes en formato .txt, para facilitar la entrada y salida de datos en el programa C++ se han utilizado 2 programas Fortran, a través de los cuáles las imágenes binarias en 20x20 se han pasado a columnas. Estas columnas son leídas por el programa principal, que trabaja con la red. Cuando este acaba, la forma final de la red de nuevo sale a un fichero por columnas, y mediante el otro programa Fortran se muestra por pantalla en 20x20 (o 30x30). Esto se ha hecho porque este lenguaje da más facilidades a la hora de leer y escribir en un formato determinado que C++, con el que sería más difícil leer los .txt en 20x20 (cosa qué también se puede hacer utilizando cadenas de carácteres por ejemplo, pero me ha parecido mucho más sencillo implementar 2 códigos Fortran).

Esto puede parecer un poco engorroso de hacer cada vez que se quiera ejecutar el programa, pero utilizando un archivo .sh se hace en una sola instrucción.

4. Resultados y análisis.

Patrones a bloques.

En primer lugar, con una red de 400 neuronas, se ha analizado la situación mas sencilla posible: Con 4 patrones almacenados, cada uno de los cuáles consta de 100 neuronas activadas (en 1) situadas en distintos bloques: el primer patrón tiene activadas las 100 primeras neuronas, y las demás desactivadas (en 0), el 2º de la neurona 100 a la 200 activadas, el tercero de la 200 a la 300 y el cuarto las cien últimas.

Partiendo de una configuración incial en la que las 25 primeras neuronas están activadas y el resto aleatoriamente activadas o desactivadas, la red neuronal evoluciona al primer patrón almacenado, tal y como se espera:

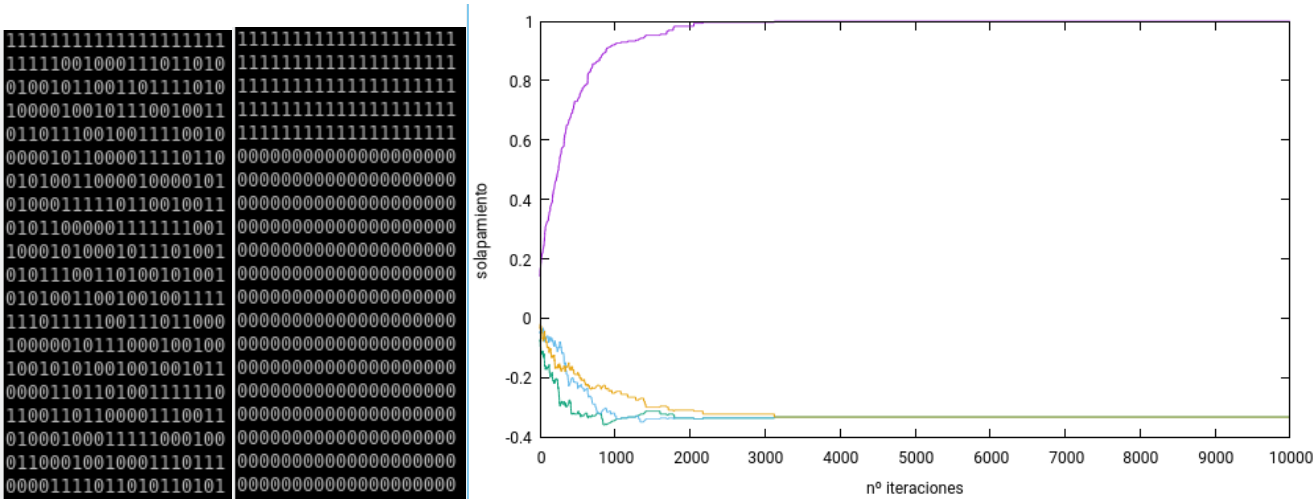

Figura 2: Configuración inicial / Config. final / Evolución de la red (solapamiento). Temperatura de 0.01

Si se sube la temperatura la red se comporta igual hasta la temperatura de 0.026, en la que mezcla patrones almacenados. Partiendo de la misma configuración inicial, a dicha temperatura.

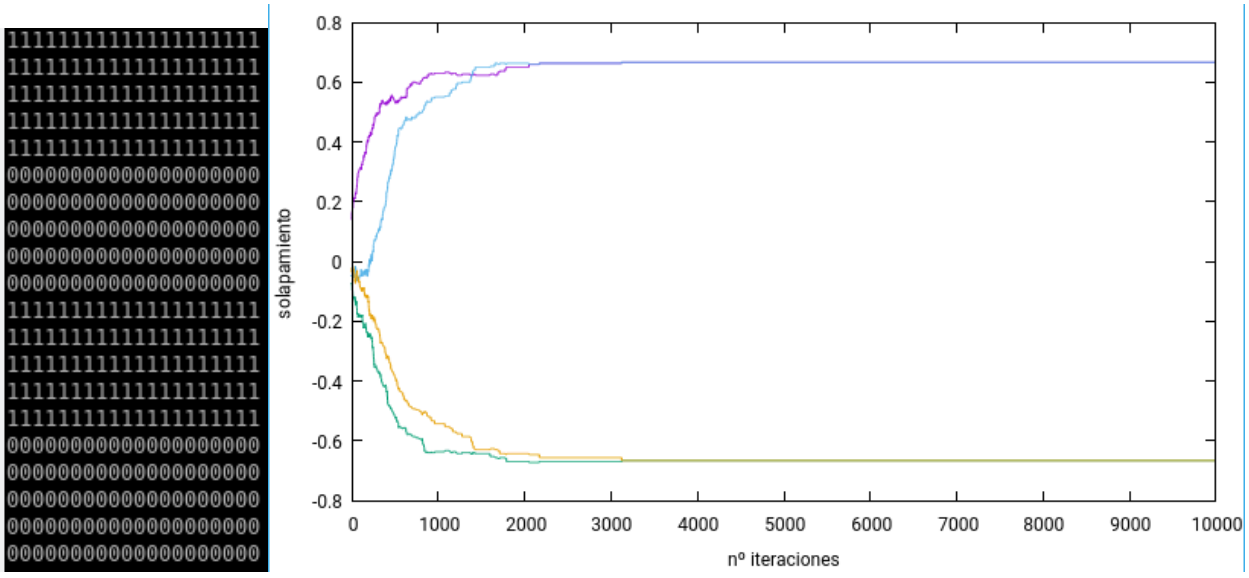

Figura 3: Config. final y solapamiento para $T = 0{,}026$.

En la gráfica se ve como en un primer momento, el solapamiento de la memoria uno (función morada) es mayor, lo que tiene sentido ya que se parte de una configuración parecida a esta. Los otros solapamientos empiezan un poco por debajo del 0, lógico teniendo en cuenta la aleatoriedad en la red inicial excepto en las 25 primeras. Sin embargo, a medida que la red neuronal converge a la memoria 1,también se activan las neuronas correspondientes a la memoria 3 (función azul). Se ve también que ahora el solapamiento no converge a 1, término correspondiente al reconocimiento de un patrón, sino a un valor por debajo de este, ya que ahora la red converge a la "suma" de dos memorias.

Si para esta misma situación se aumenta más la temperatura, la red se sigue comportando igual, hasta que se supera la temperatura crítica 0.23. A partir de ahí, la red empieza a fallar y deja de converger a los 2 patrones anteriores de forma limpia, es decir se activan neuronas fuera de ellos y se desactivan dentro. Cuando la temperatura es relativamente alta (entorno al doble de 0.23) la red ya es totalmente incapaz de distinguir las memorias, y las configuraciones finales que adapta esta son totalmente aleatorias:

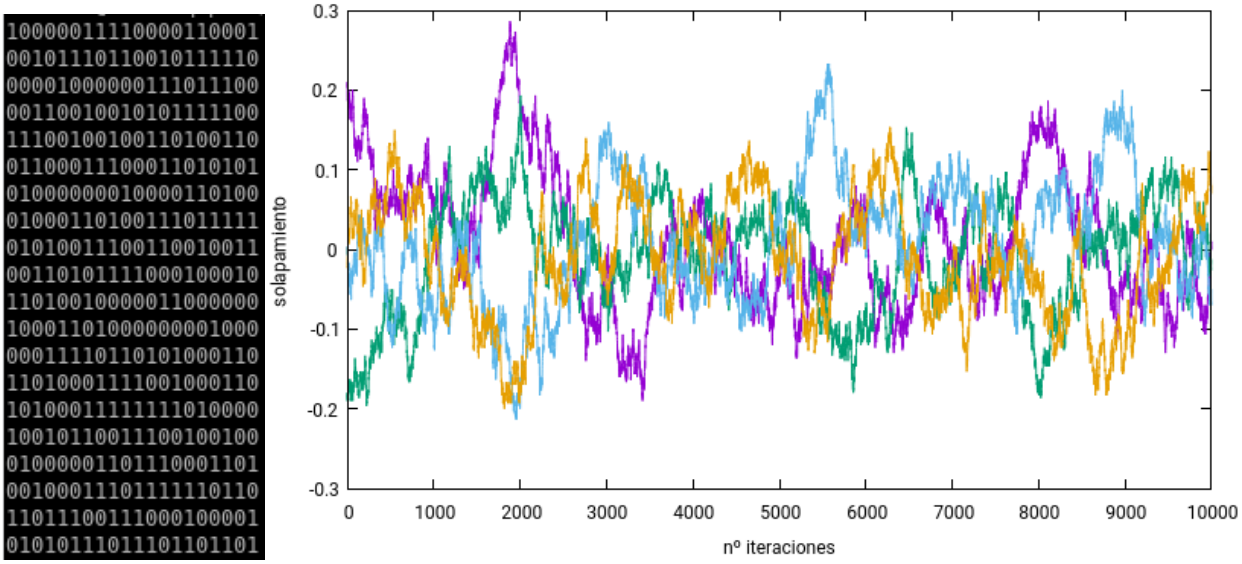

Figura 4: Config. final / solapamiento $T = 0{,}5 > T_{crítica}$

Si se parte de una configuración inicial totalmente aleatoria:

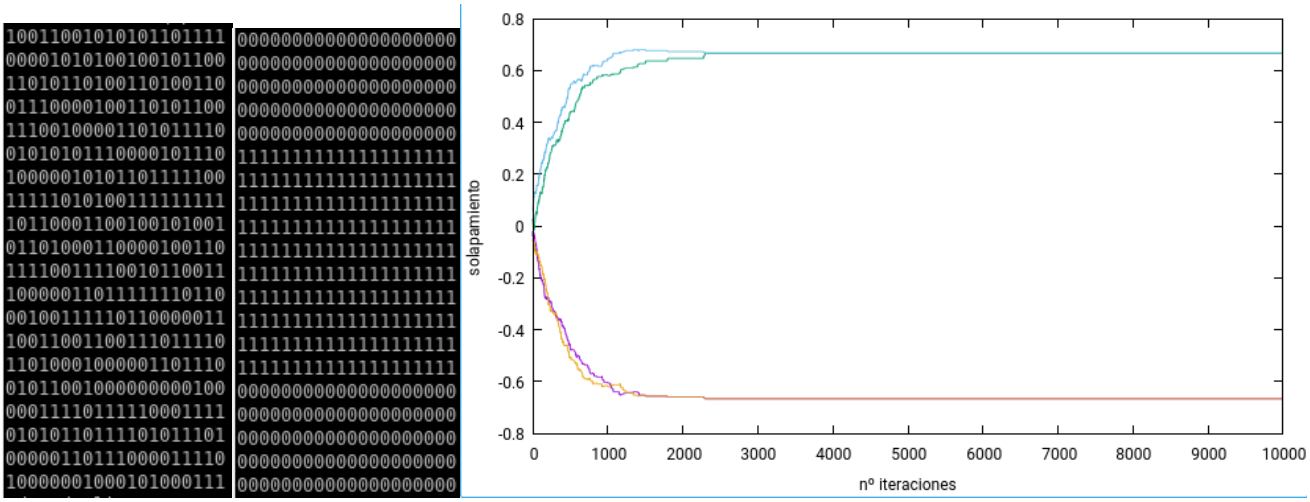

Figura 5: Config. inicial / Config. final / solapamiento, $T = 0{,}01$

Aunque la temperatura inicial sea muy baja, al partir de una configuración aleatoria, la red no es capaz de converger a un patrón concreto.

Aquí se puede apreciar, analizando la gráfica entorno al punto (0,0) que la configuración inicial aleatoria favorece a la memoria 3 (función azul), sin embargo la red activa también las neuronas correspondientes a la memoria 2 (función verde).

En otras palabras, el sistema no es capaz de recuperar un patrón concreto si partimos de una config. inicial aleatoria. Es necesario partir de un patrón deformado.

Si aumentásemos la temperatura por encima de la crítica se vería un comportamiento similar al que se da en el caso anterior (fig.4).

Se ha analizado hasta ahora el comportamiento de la red para el nº fijo de 4 patrones según la temperatura, llegando a la conclusión de que esta funciona correctamente por debajo de T=0.026, y partiendo de un patrón deformado.

Cabe añadir, que el comportamiento de la red neuronal es el mismo en líneas generales si se cambian los números aleatorios o si se cambian las config. iniciales. Es decir, lógicamente si la config inicial favorece a la memoria 2 en vez de a la 1, esta convergerá a la 2, y al aumentar la temperatura, convergerá a 2 memorias, la segunda y alguna de las otras.

Claramente, si se realiza este análisis de nuevo, variando la semilla de los números aleatorios, se ve que la red, para el caso en el que el sistema está a la temperatura de 0.026, converge de nuevo a la "suma" de 2 memorias, una de ellas será la primera (ya que las cond. iniciales la favorecen) y la otra no tiene por qué ser la tercera, como se ha visto en este análisis, de hecho seguramente no lo será, pues cada una de las otras 3 tiene 1/3 de probabilidades.

Otra importante cuestión a analizar es como de deformada está la configuración inicial con respecto a alguna memoria.

En este análisis se ha partido de 25 neuronas activadas y las demás aleatoriamente desactivadas o activadas. Si se parte de 25 neuronas activadas y las demás desactivadas, es claro que el sistema lo tiene "más fácil" ya que la condición inicial es más cercana a la memoria. En este caso, la red sigue convergiendo a la memoria 1 aunque se supere la temperatura de 0.026. De hecho, no mezcla patrones a menos que superemos la temperatura crítica de 0.23, a partir de la cuál el sistema empieza a no distinguir patrones, y para temperaturas más altas ya ocurre lo mismo que para el caso anterior en el que la config. final se vuelve totalmente aleatoria (fig. 4)

Se analiza ahora la influencia que tiene en el sistema aumentar el nº de patrones almacenados.

Partiendo ahora de 8 memorias (el doble), de forma similar a la analizada antes (sólo que ahora la división es 400/8 en lugar de 400/4 como fue antes), se puede apreciar que al sistema le cuesta mucho más evolucionar favorablemente hacia la memoria más parecida a su condición inicial. Concretamente, a menos que partamos de una config. inicial muy parecida a la deseada, la red mezcla varios patrones.

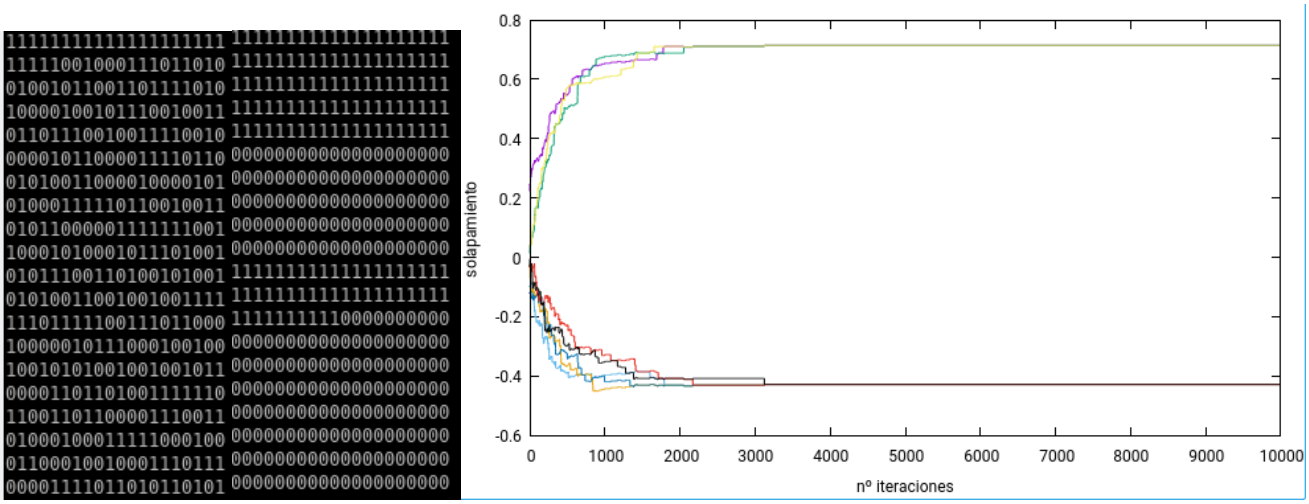

Figura 6: Config. inicial / Config. final / solapamiento, $T = 0{,}01$. 8 memorias almacenadas.

Se repite ahora la situación primera de todas, aquella en la que la red convergía sin problema al primer patrón. La temperatura sigue siendo 0.01.

Ahora la red mezcla 3 patrones, a pesar de que el patrón 1 esté claramanete favorecido por la configuración inicial. Por mucho que se minimice la temperatura, el comportamiento del sistema sigue siendo el mismo.

Al aumentarla se tiene una situación similar hasta una temperatura crítica de 0.095, a partir de la cuál ocurre lo mismo que en la figura 4.

Para que el sistema converja correctamente a una memoria concreta, sería necesario introducir una configuración inicial muy similar al patrón guardado, tal y como se verá a continuación en el análisis con 6 memorias, que permite matizar más que con 8.

Se analiza ahora un caso intermedio entre los anteriores. Si se almacenan 7 memorias, el comportamiento es practicamente el mismo que con 8, y si se almacenan 5 es el mismo que con 4.

Introduciendo ahora 6 memorias se encuentra un comportamiento intermedio entre el de 4 y el de 8 (ahora la división es 400/6 neuronas activadas cada patrón).

Para que la red converja al patrón es necesario que la configuración inicial no tenga muchas neuronas activadas fuera de la memoria deformada. Si se tiene una config. inicial con unos y ceros aleatorios excepto en una pequeña zona concreta, el sistema no evolucionará a un patrón concreto, sino a la suma de varios, igual que pasó antes con 8. Sin embargo, si se reduce la probabilidad de que las neuronas estén activadas, la red evoluciona sin problema:

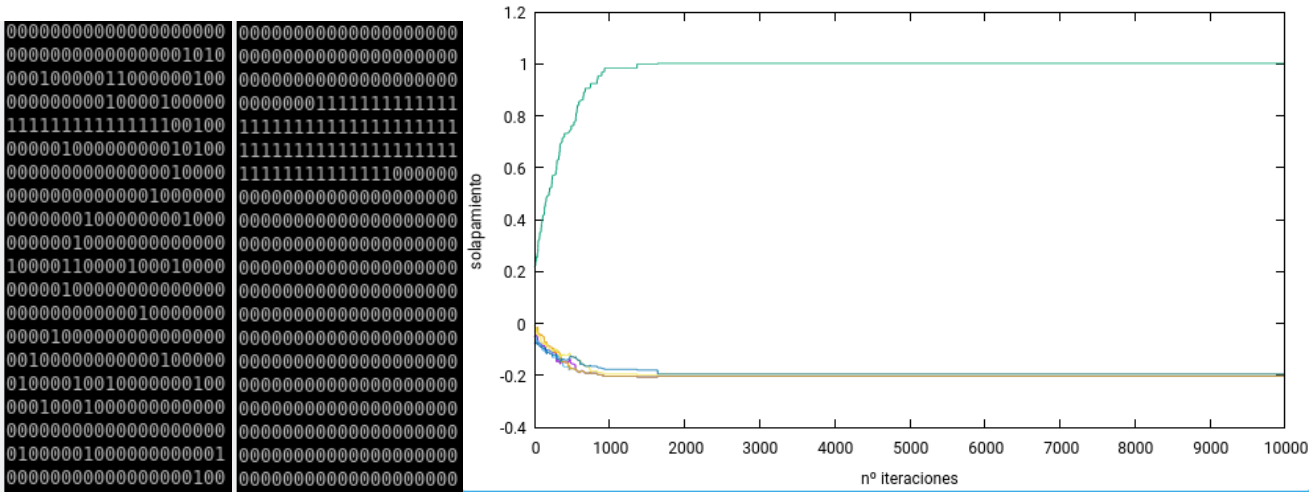

Figura 7: Config. inicial / Config. final / solapamiento, $T = 0{,}01$. 6 memorias almacenadas.

Aquí se ve que introduciendo una memoria 2 deformada, la red evoluciona favorablemente a dicha memoria, pero es capaz de hacerlo porque hay pocas neuronas activadas.

Cabe añadir, que en esta situación, al aumentar prácticamente nada la temperatura, la red ya converge a "suma" de 2 memorias, concretamente lo hace para una temperatura entorno a 0.012, a diferencia de cuando teníamos 4 neuronas, que la red evolucionaba correctamente hasta la temperatura de 0.026. Con 8 memorias guardadas, la diferencia entre temperaturas se hace aún mas sutil. Para temperaturas mayores a 0.105 la red ya es incapaz de converger a un patrón concreto, aún partiendo de una condición inicial muy favorable.

También se tiene ahora una temperatura crítica menor, $T_{crítica} \simeq 0{,}149$, temperatura a la cuál la red empieza a no distinguir bien los patrones. Se expone lo que ocurre a T=0.16.

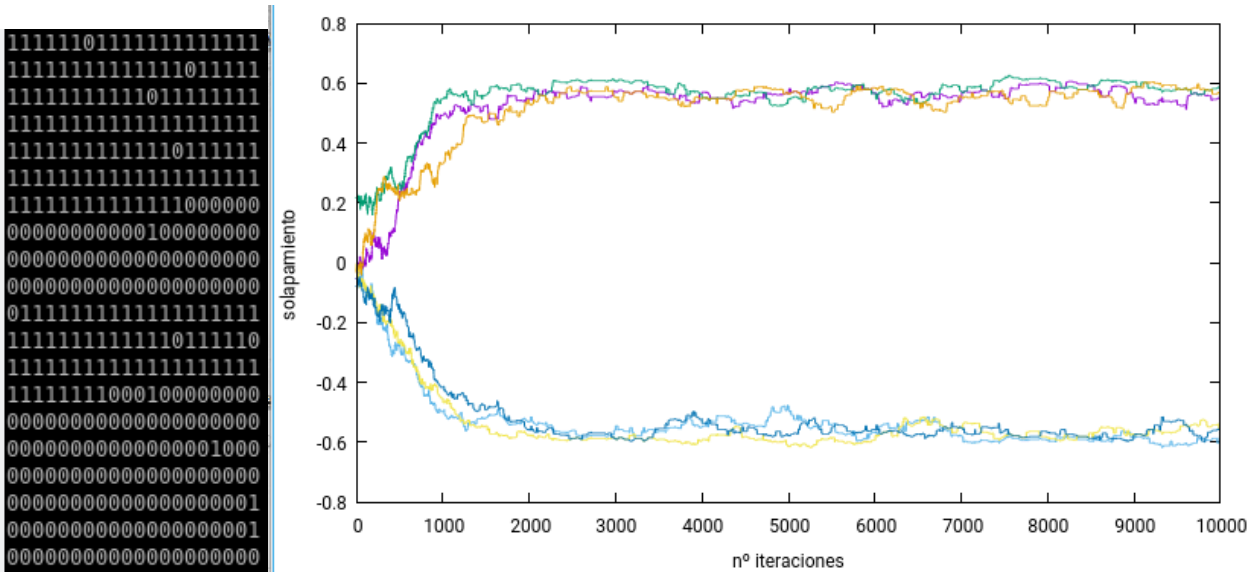

Figura 8: Config. final / solapamiento, $T = 0{,}16$. 6 memorias almacenadas.

Aumentando la temperatura un poco más ocurriría lo mismo que en la figura 4, que es lo que está empezando a ocurrir aquí.

En líneas generales, se ha visto que aumentando el nº de patrones memorizados, al sistema le cuesta más distinguir unos de otros, y le afecta más que la temperatura aumente:

- Con 6 u 8 memorias el sistema necesita que las config. iniciales tengan pocas neuronas activadas fuera del patrón deformado inicial, independientemente de lo baja que sea la temperatura.
  Por tanto, se puede concluir que a la hora de introducir un patrón deformado al sistema para que lo reconozca, es preferible que el nº de neuronas activadas sea bajo

- Suponiendo que la cond. inicial lo permita, con 4 memorias, el sistema es capaz de converger a un patrón mientras la temperatura sea menor de 0.026. Con 6 memorias esto ocurre a 0.012. Con 8 a 0.0105. A temperaturas mayores que esta, el sistema activa las neuronas de 2 o más memorias, no solo la que debiera.

- A partir de una determinada temperatura crítica, el sistema empieza a ser incapaz de distinguir unos patrones de otros. Esta temperatura es 0.3 si tenemos 4 memorias almacenadas, 0.149 si se tienen 6, y 0.095 si se tienen 8. A temperaturas lo suficientemente por encima de estas (entorno al doble) la red neuronal se comporta de forma totalmente aleatoria, tal y como en la figura 4, independientemente del nº de patrones que haya guardados y de las condiciones iniciales que tenga la red.

Reconocimiento de patrones con letras.

Se estudia ahora un caso más interesante y realista en cuanto a aplicaciones:

Las memorias son ahora imágenes donde las 400 neuronas forman las letras A, B, C y D. Se analiza como se comporta la red neuronal cuando se parte de una letra difuminada.

Para difuminar, en el programa se ha utilizado una función que en cada uso cambia el estado de 50 neuronas aleatorias. Se aplica tantas veces cuanto más difuminada se quiera que esté la condición inicial.

Para la letra A por ejemplo, se parte de la A perfecta, se difumina tanto como queramos y se introduce en la red. Se muestra como ha actuado la red:

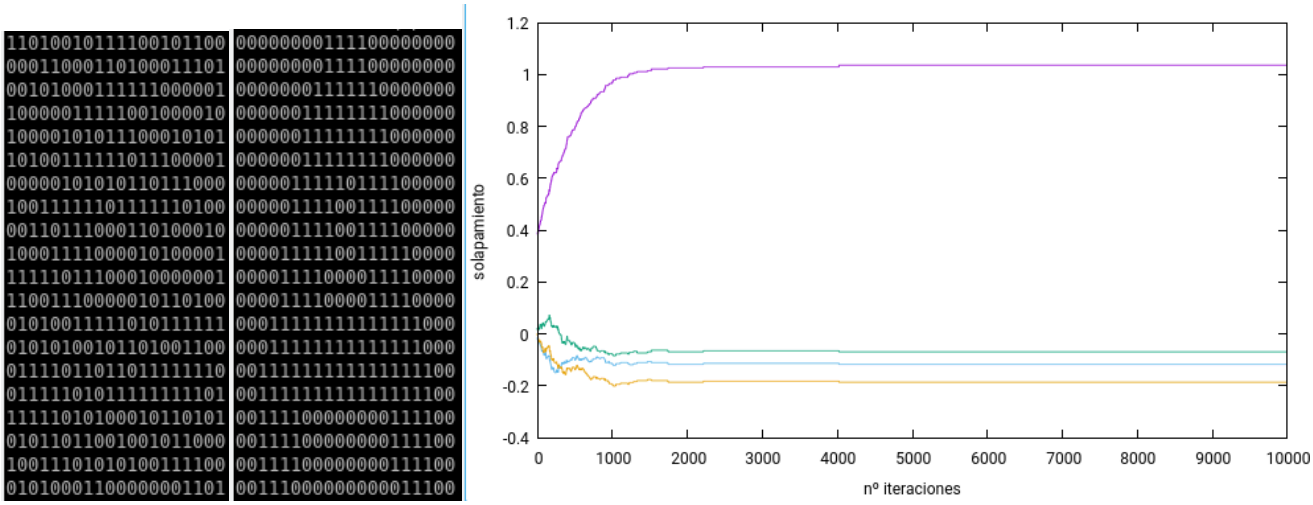

Figura 9: Config. inicial / Config. final / solapamiento, $T = 0{,}01$

Se puede apreciar el muy buen funcionamiento de la red neuronal ante casos como estos. La configuración inicial es la memoria de la A difuminada 4 veces. Aunque a simple vista no se parezca en nada a la A, la red es capaz de evolucionar favorablemente hacia ella.

Se analiza ahora que ocurre si se difumina aún más (6 veces)

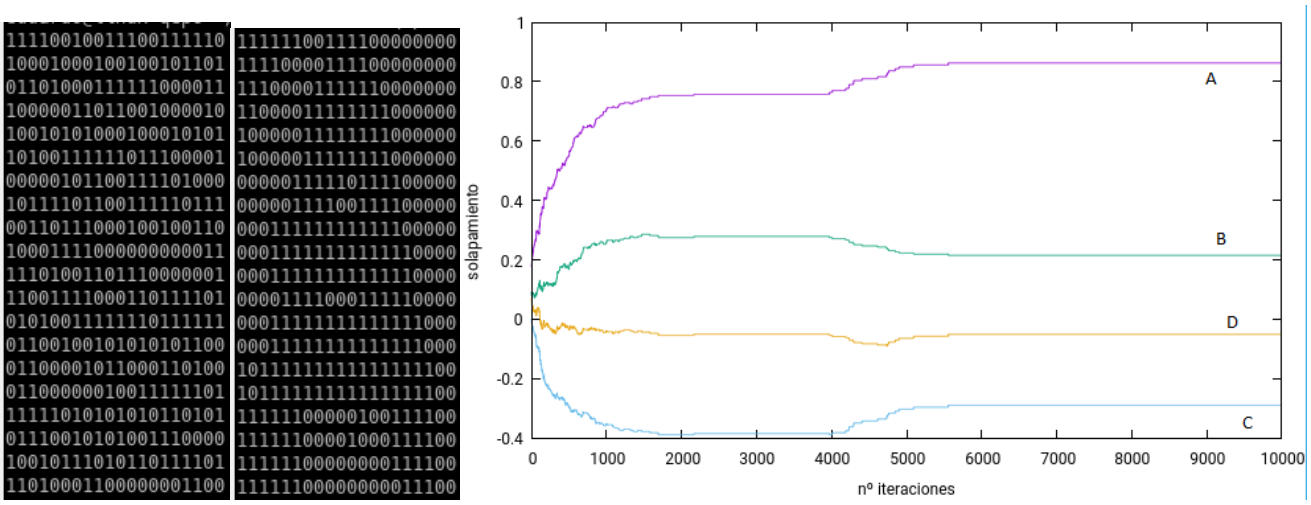

Figura 10: Config. inicial / Config. final / solapamiento, $T = 0{,}01$

Se ve ahora que partiendo de una condición inicial más complicada, la red no es capaz de converger perfectamente a la memoria, el solapamiento no llega a ser 1.

No obstante, el valor de este está por encima de 0.75 y es capaz de mostrar una configuración muy similar a la A.

Con este tipo de patrones no ocurre como con los anteriores, en los que con poco que el parámetro de ruido (la temperatura) aumentaba, la red convergía a la "suma" de 2 patrones.

En este caso la red sigue comportándose igual hasta la temperatura de aproximadamente 0.08, donde toma una forma que en nada se parece a ninguna de las letras:

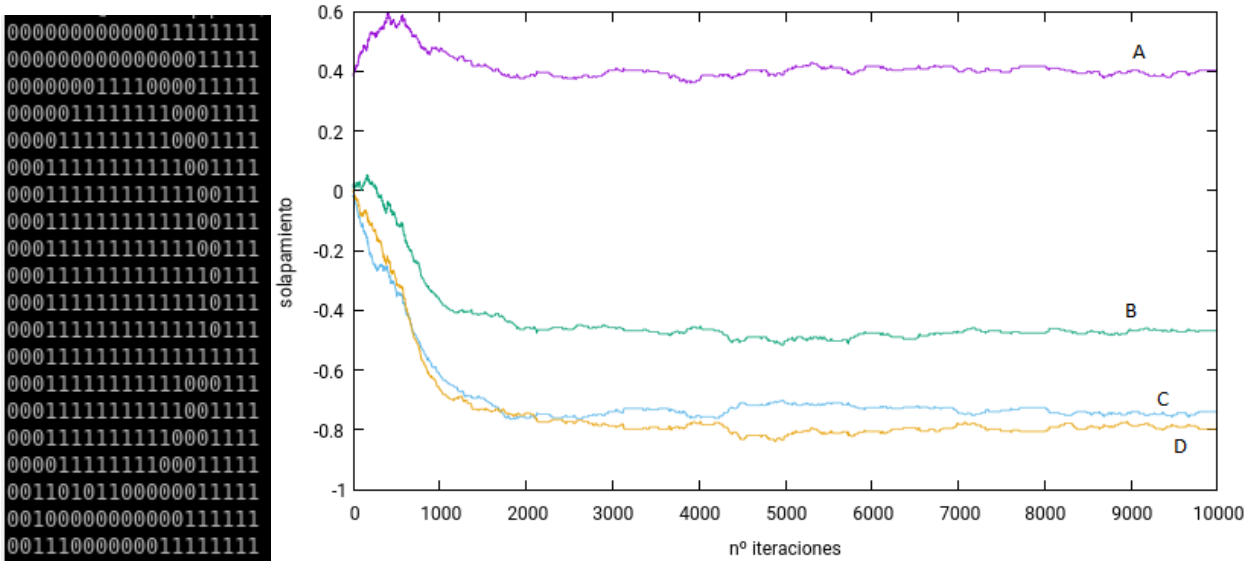

Figura 11: Config. final / solapamiento, $T = 0{,}08$

Esta forma es también la adoptada cuando se parte de una configuración inicial totalmente aleatoria, o que en nada se parezca a ninguno de los patrones guardados.

Es importante remarcar que esto ocurre siempre y cuando se esté por debajo de la temperatura crítica correspondiente a 4 patrones, 0.3. A la cuál empieza a ocurrir lo correspondiente a la figura 3. Suceso que, como se dijo antes es independiente a lo demás: a partir de la temperatura crítica la red deja de distinguir patrones y se comporta de un modo totalmente aleatorio. Esta temperatura crítica es menor cuantas más memorias haya almacenada, y es independiente de la forma de estas y de la configuración inicial.

Si se introduce de condición inicial un patrón difuminado de la letra C, el resultado es el mismo que para la A.

El problema aparece para las letras B y D. Cómo ambas son muy parecidas, la red no es capaz de converger a la D ni aún partiendo de una configuración inicial que sea una D perfecta.

El resultado que se obtiene partiendo de una B difuminada:

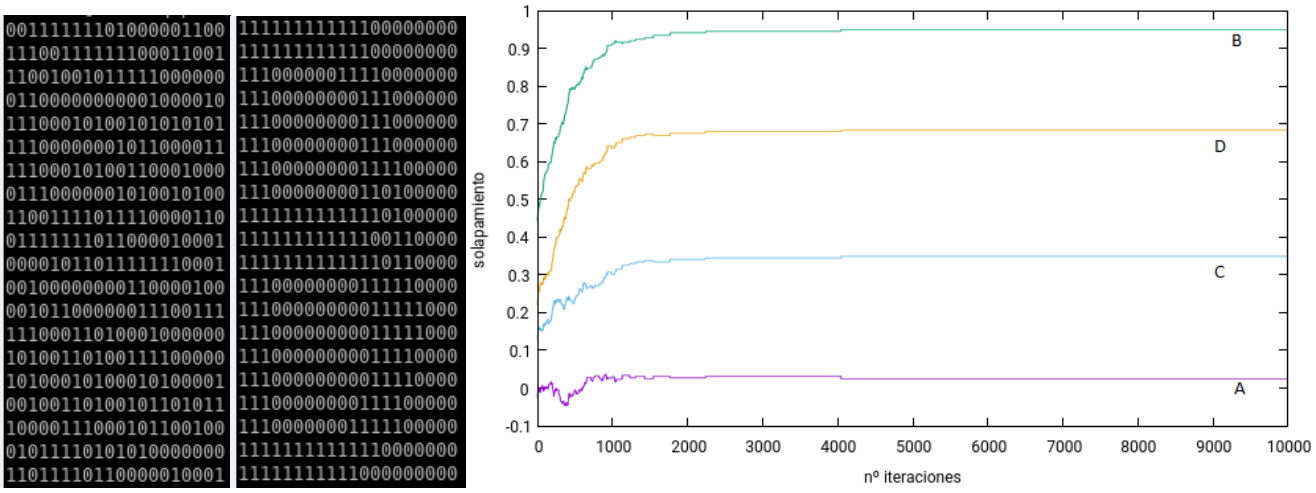

Figura 12: Config. inicial / Config. final / solapamiento, $T = 0{,}01$

Se ve que la red no converge perfectamente a la B (el solapamiento no es 1, y observando la imagen se aprecia que no es una B perfecta).

Se obtiene el mismo resultado si se parte de una configuración inicial que es una D perfecta:

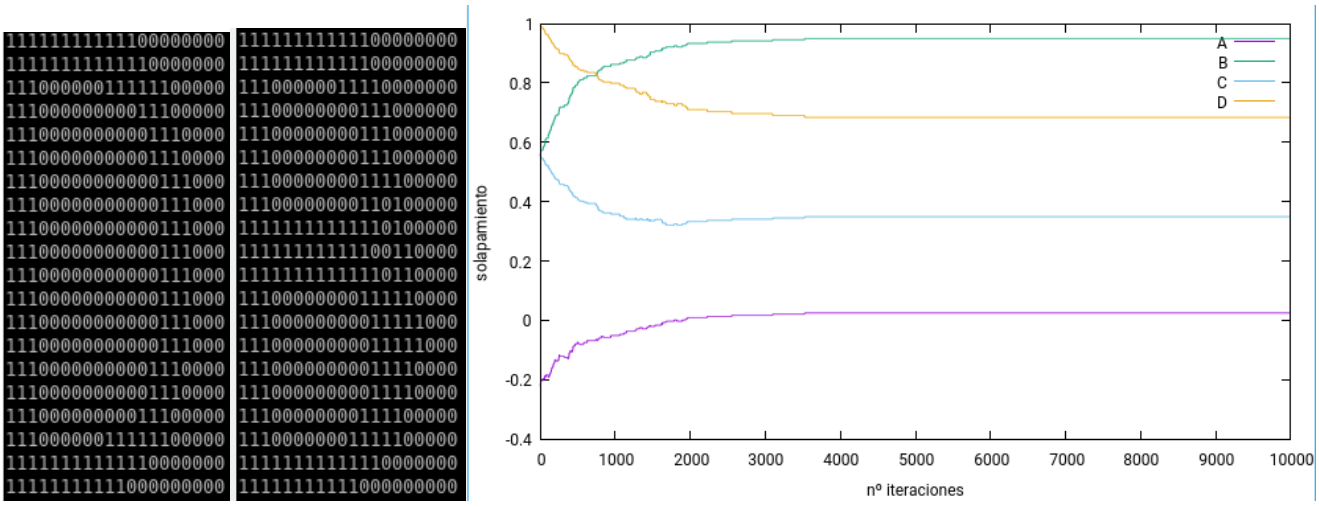

Figura 13: Config. inicial / Config. final / solapamiento, $T = 0{,}01$

En tiempo 0, es decir, nº iteraciones 0, la red es la D que se le ha introducido como config. inicial. No obstante, cuando el sistema empieza a funcionar evoluciona hasta el caso anterior.

Si sustituimos la memoria que forma la letra D por una más distinta a cualquiera de las demás, X por ejemplo,

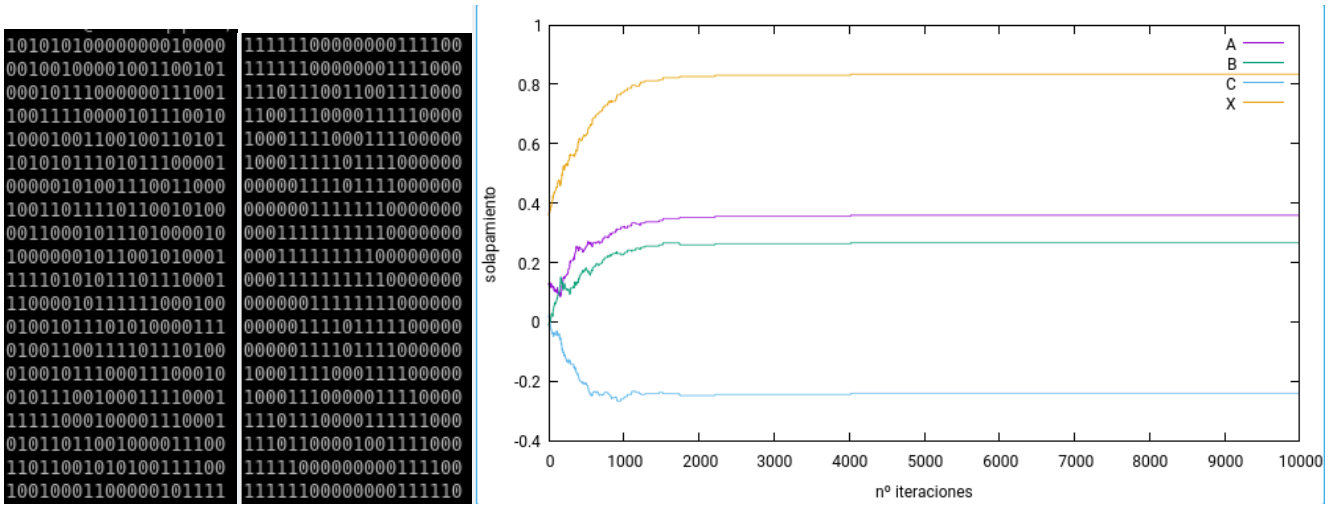

Figura 14: Config. inicial / Config. final / solapamiento, $T = 0{,}01$

El sistema converge a la X, aunque no perfectamente, pero si bastante bien.
Además, ahora la red converge perfectamente a la B, a diferencia de como lo hacía antes.

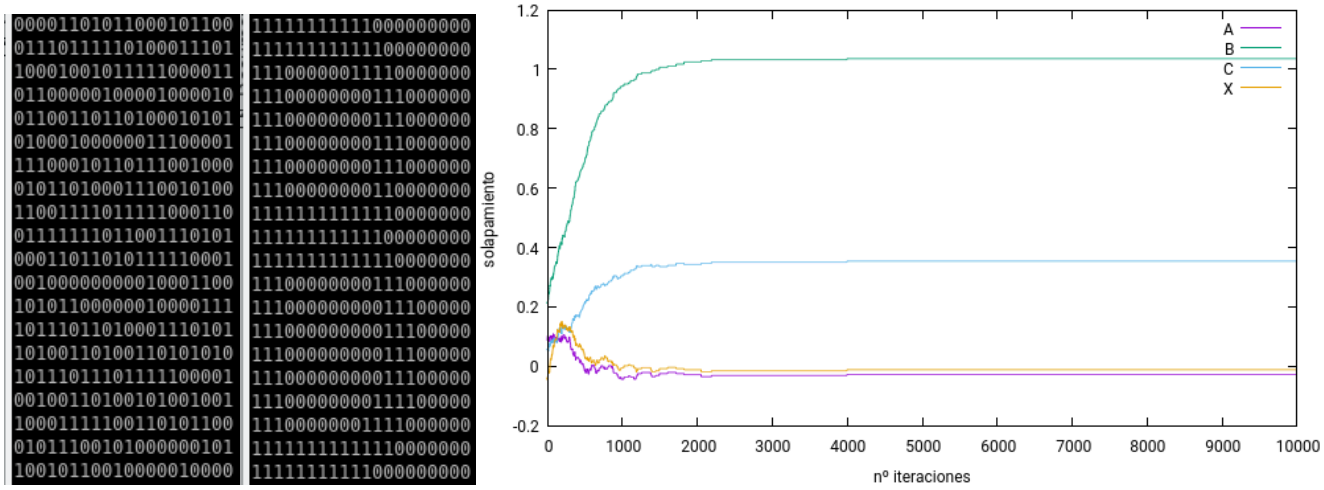

Figura 15: Config. inicial / Config. final / solapamiento, $T = 0{,}01$

Por tanto, la red neuronal trabaja bastante bien con patrones almacenados que formen letras, siempre y cuando estos no se *parezcan* demasiado entre ellos. Incluso partiendo de config. iniciales muy desfavorables, en las que el ojo humano es incapaz de ver parecido alguno con la letra concreta, la red es capaz de evolucionar favorablemente hacia una configuración muy parecida a la buscada. (figura 10).

La temperatura a la que empieza a comportarse peor (figura 11) es más alta que para los patrones que forman bloques.

Si se aumenta el nº de memorias almacenadas, ocurre algo similar al primer caso en el que aumentábamos el nº de memorias que formaban bloques: es necesario que las condiciones iniciales sean más favorables y que las temperaturas sean más bajas.

Se omite este estudio en el informe porque las conclusiones serían las mismas, con números un poco más bajos para la temperatura a la que se empiezan a mezclar patrones y los mismos para la temperatura crítica en la que ya no los reconoce.

También se ha llevado a cabo un análisis en el que en lugar de letras, las memorias formaban dígitos. Los resultados han sido similares a las letras: si los números eran lo suficientemente distintos (1, 2, 6 y 0 por ejemplo, aunque hay muchas más combinaciones posibles, con más patrones incluso) la red evoluciona favorablemente para todas las memorias, de forma aproximadamente parecida al caso estudiado cuando hemos introducido la X. No obstante, cuando las memorias forman números entre los que se encuentran el 4 y el 9 o el 6 y el 8 por ejemplo, la red los mezcla o no consigue converger a uno de ellos.

Se estudia ahora un caso distinto a los anteriores: las memorias almacenadas son patrones aleatorios.

Reconocimiento de patrones aleatorios.

En esta última parte, se ha estudiado a partir de memorias aleatorias cuántos patrones es capaz de recordar la red a temperatura nula.

Se omite en las figuras la imagen de la red neuronal inicial y final, ya que al ser patrones aleatorios apenas aportan información visual.

Se ve en primer lugar que con 8 memorias almacenadas, la red es capaz de reconocer cualquiera de ellas, a diferencia de cómo pasaba en casos anteriores. Por tanto, en primer lugar hay un mejor comportamiento de la red cuando las memorias son aleatorias.

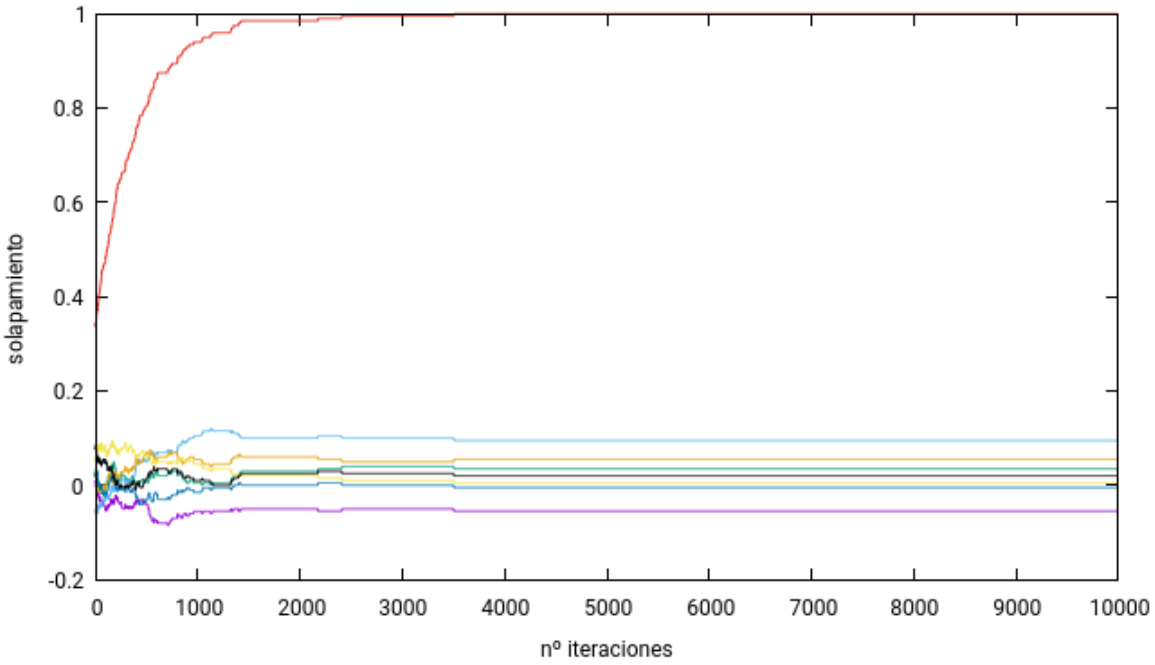

Figura 16: Evolución de la red con patrones aleatorios / 8 memorias almacenadas

La configuración inical de la red ha sido la memoria correspondiente al rojo difuminada 5 veces.
Lo mismo ocurre para cualquiera de las otras siete si se parte de un patrón difuminado de la misma.

Cuando se introducen 9 memorias:

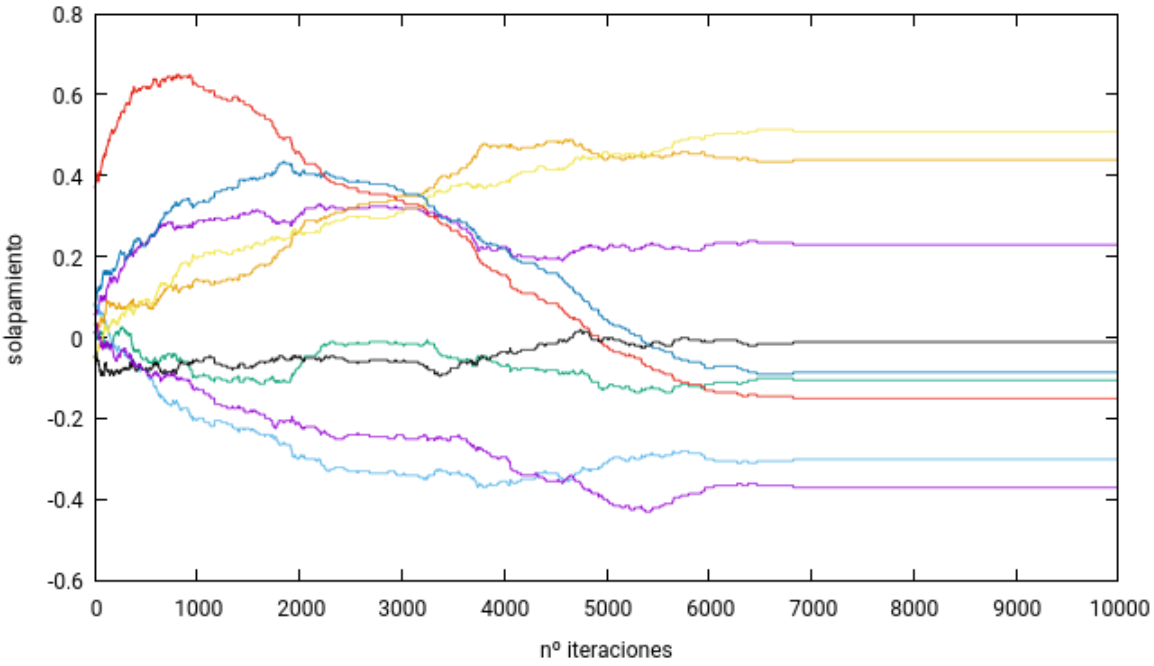

Figura 17: Evolución de la red con patrones aleatorios / 9 memorias almacenadas

La red falla partiendo de las mismas condiciones iniciales.

Cabe añadir que aunque se tengan 9 memorias, la red sí es capaz de converger a algunas de ellos, pero ya no a todos, como si es posible con 8.

La fracción máxima de patrones que la red puede recordar queda entonces:

$$\alpha_c = \frac{P_c}{N} = \frac{8}{400} = 0{,}02$$

Se ha analizado ahora como cambia el comportamiento si en lugar de 400 tenemos 900 neuronas. Siendo 9 las memorias almacenadas:

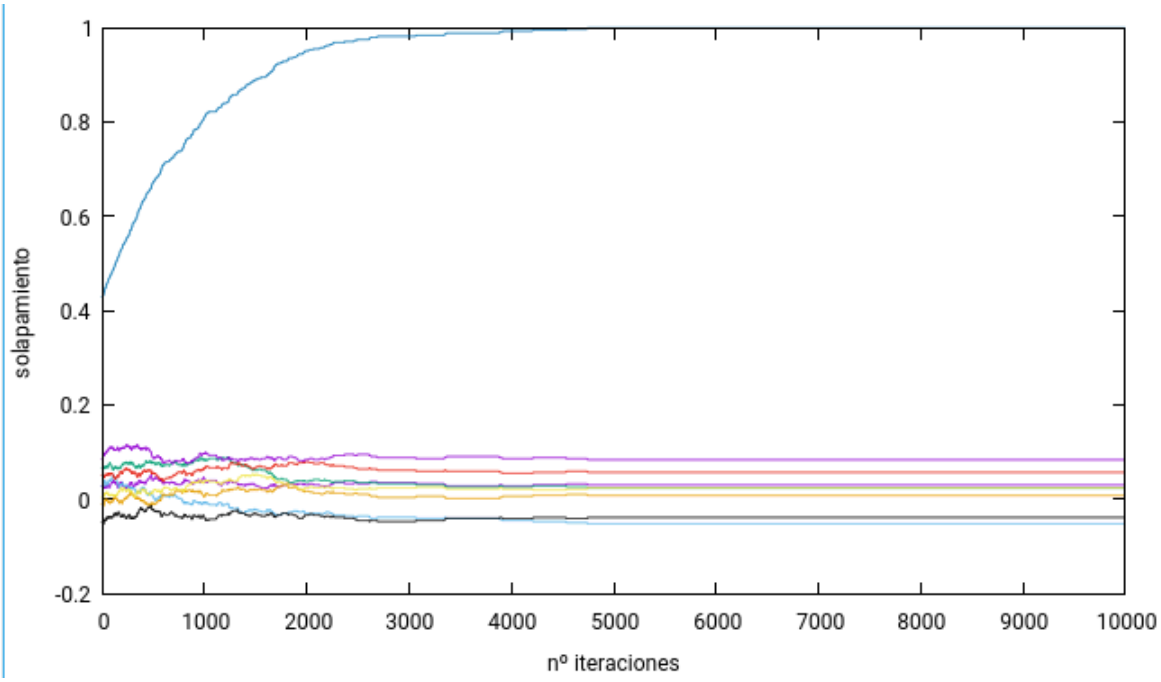

Figura 18: Evolución de la red con patrones aleatorios / 9 memorias almacenadas. Red de 900 neuronas.

Partiendo de una config. inicial similar a la anterior (claramante ahora se ha difuminado más veces que antes, ya que hay más del doble de neuronas) la red ha sido capaz de converger perfectamente.

Lo mismo ocurre para los otros patrones si se parte de una config. difuminada de cualquiera de ellos.

La figura 17 se repite en este caso a partir de las 14 memorias almacenadas, se omite la imagen porque es prácticamente la misma pero con 14 funciones en lugar de 9.

La fracción máxima de patrones queda ahora

$$\alpha_c = \frac{P_c}{N} = \frac{14}{900} = 0{,}016$$

Por tanto, mientras más neuronas tenga la red, mejor trabaja esta y más patrones puede recordar. No obstante, la fracción maxima va disminuyendo, lo que conduce a que por mucho que se aumente el nº de neuronas habrá un tope en el nº de patrones que se puedan almacenar. Dado que el programa no soporta más de 1020 neuronas, esas 14 memorias son el máximo que el sistema puede memorizar.

5. Conclusión general.

Todo lo estudiado permite concluir que el buen funcionamiento de la red neuronal de Hopfield desarrollada depende de 5 parámetros, que en orden de importancia son:

- Temperatura: Siempre hay una temperatura a partir de la cuál la red neuronal empieza a fallar, esta es menor cuánto mayor sea el número de patrones, pero siempre existe y es muy baja (menor que 0.5 por poner una cota superior genérica).
- Número de patrones almacenados: Si el nº de patrones almacenados es grande, la red los mezcla. Este número puede ser mayor cuánto mayor sea el nº de neuronas. El tope está en 14 memorias.
- Configuración inicial: Es necesario que inicialmente la red tenga una forma que favorezca a alguno de los patrones, así podrá evolucionar hasta el mismo. El solapamiento inicial debe ser como mínimo 0.1 (el 10 % de la memoria), sino dificilmente podrá converger. Además, cuanto más desfavorable sea la condición inical más favorables tendrán que ser la temperatura, el nº de patrones, etc.
- Forma de los patrones: Si 2 o más memorias guardadas son parecidas entre ellas es posible que la red solo converja a una de ellas o a una mezcla de ambas. También se ha observado que el sistema comete menos fallos cuando trabaja con patrones aleatorios.
- Número de neuronas: Cuantas más neuronas, mejor funciona la red, ya que los patrones se pueden distinguir más entre ellos. Además, si se forman figuras concretas, como letras, las imágenes formadas quedan más detalladas.

Cabe añadir, a modo de conclusión personal, que la realización de este trabajo me ha propocionado importantes conocimientos sobre informática, programación, redes neuronales y números aleatorios, que seguro que me serán útiles en mi carrera como físico. Además, gracias a ello he descubierto una parte de la física y de la programación que me resulta realmente interesante, y a la que no me importaría dedicarme en un futuro.

Referencias

[https://www.xataka.com/robotica-e-ia/las-redes-neuronales-que-son-y-por-que-estan-volviendo]

[http://avellano.fis.usal.es/ lalonso/RNA/index.htm]

[Memoria asociativa en redes tipo Hopfield - TFG Ibon Recio (Univ.del País Vasco)]

[https://www.dcode.fr/binary-image]