CAPÍTULO I

INTRODUCCIÓN

I.1 INTRODUCCIÓN

Las redes neuronales artificiales son una forma de emular ciertas características propias de los humanos, con la capacidad de memorizar y de asociar hechos. Aquellos problemas que no pueden expresarse a través de un algoritmo se observará que todos ellos tienen una característica en común: la experiencia. El hombre es capaz de resolver esas situaciones acudiendo a la experiencia acumulada. Así, parece claro que una forma de aproximarse al problema que consista en la construcción de sistemas que sean capaces de reproducir esta característica humana.

En definitiva, las redes neuronales no son más que un modelo artificial y simplificado del cerebro humano, que es el ejemplo más perfecto del que disponemos para un sistema que es capaz de adquirir conocimiento a través de la experiencia. Una red neuronal es un sistema para el tratamiento de la información, cuya unidad básica de procesamiento está inspirada en la neurona que es el elemento fundamental del sistema nervioso humano.

Todos los procesos del cuerpo humano se relacionan en alguna u otra manera con la actividad de igual manera con la inactividad de estas neuronas. Las mismas son un componente relativamente simple del ser humano, pero cuando millares de ellas se conectan en forma conjunta hacen muy poderosas.

También, es bien conocido que los humanos son capaces de aprender. Aprendizaje significa que aquellos problemas que inicialmente no pueden resolverse, pueden ser resueltos después de obtener más información acerca del problema.

Por lo tanto, las Redes Neuronales:

- Consiste de unidades de procesamiento de intercambian datos o información.
- Se utilizan para reconocer patrones, incluyendo imágenes, caracteres y secuencias de tiempo tendencias financieras.
- Tienen capacidad de aprender y mejorar su funcionamiento.

En este trabajo presentaremos la creación de reconocimiento de caracteres de manera automática. El reconocimiento de caracteres engloba un conjunto de métodos y algoritmos que se describirán más adelante, los cuales permiten realizar una fase de entrenamiento que a la final permitirán reconocer de forma automática caracteres. Se debe mencionar que el reconocimiento de caracteres no solo se utiliza para

reconocimiento de textos escritos sino que tiene muchas aplicaciones que mencionaremos a lo largo del presente trabajo.

El reconocimiento de caracteres es una de las áreas de aplicación más efectiva de las redes neuronales, al punto de que ya es común el diseño de sistema de reconocimiento de caracteres basado en redes neuronales.

I.2 DEFINICIÓN DE LOS OBJETIVOS

I.2.1 OBJETIVO GENERAL

Seleccionar y probar cuatro diferentes redes neuronales artificiales: Perceptrón Multicapa, Madaline, Backpropagation y Hopfield para el reconocimiento de caracteres del idioma español para determinar la eficiencia y la efectividad de los mismos.

I.2.2 OBJETIVOS ESPECÍFICOS

- Seleccionar y probar las redes neuronales artificiales más apropiadas para el reconocimiento de caracteres del idioma español.
- Implementar las redes neuronales artificiales en tres lenguajes de programación diferentes: Java, C#, Visual Basic 6.
- Determinar cual es la red neuronal artificial más eficiente para el reconocimiento de caracteres del alfabeto español.

I.3 ALCANCES

El trabajo permite la comparación de las redes neuronales artificiales: Backpropagation, Perceptrón Multicapa, Madaline, Hopfield; para realizar esta comparación, la red neuronal artificial será capaz de *"aprender"* la forma de escritura aumentando con ello las capacidades de identificación.

Luego de la identificación de los caracteres ingresado por el desarrollador, procederá a la verificación de los valores obtenidos de cada red neuronal artificial y así poder determinar cual es la red neuronal artificial con más grado de eficiencia en cuanto al reconocimiento de caracteres del alfabeto español.

I.4 LIMITACIONES

- No contará con una interfaz gráfica de usuario para ingresar datos. Ya que no es una aplicación para una empresa o persona.

- Solo se podrá ingresar diez caracteres. Puesto que la red neuronal sería demasiado extensa y la programación sería compleja.
- No se podrá ingresar números.
- No se podrá ingresar símbolos ni fórmulas.

En el presente trabajo no esta dirigido a un usuario especifico porque se analizó un conjunto de redes neuronales artificial para determinar qué red neuronal cuenta con mayor grado de eficiencia en cuanto al reconocimiento de caracteres.

I.5 ANTECEDENTES HISTÓRICOS

La consecución, el diseño y la construcción de máquinas capaces de realizar proceso con cierta inteligencia ha sido uno de los principales objetivos científicos a lo largo de la historia, ya que en un principio los esfuerzos estuvieron dirigidos a la obtención de autómatas, cuya función era la realización de algunas funciones típicas de los sensores humanos, con alta posibilidad de éxito. De los intentos realizados en este sentido se ha llegado a definir las líneas fundamentales para la obtención de máquinas inteligentes y ese así como, actualmente los estudios siguen esta misma línea, con resultados sorprendentes, evidenciados en la existencia de diferentes maneras de realizar procesos similares a los inteligentes y que podemos clasificar dentro de la llamada Inteligencia Artificial(I.A).

Sin embargo existe un enorme problema que limita los resultados que se pueden obtener a pesar de disponer de herramientas y lenguajes de programación diseñados expresamente para el desarrollo de máquinas inteligentes, ya que estas máquinas se implementan sobre computadoras basadas en la filosofía de Von Neumann, y se apoyan en una descripción secuencial del proceso de tratamiento de la información. Si bien el desarrollo de estas computadoras es espectacular, no deja de seguir la línea antes expuesta como en el caso de una máquina que es capaz de realizar tareas mecánicas de forma increíblemente rápida, como cálculo, como ordenación o control, pero a su vez es incapaz de obtener resultados aceptable cuando se trata de tareas como reconocimiento de voz, formas, etc.

Por otra parte, otra línea de la investigación ha tratado de aplicar principios físicos que rigen en la naturaleza para obtener máquinas que realicen trabajos complejos en nuestro lugar. De igual manera se puede pensar respecto a la forma y capacidad de

razonamiento humano; se puede intentar obtener máquinas con esa capacidad basadas en el mismo principio de funcionamiento.

No se trata de construir máquinas que compitan con los seres humanos, sino que realicen ciertas tareas de rango intelectual para ayudarle, principio básico de la Inteligencia Artificial.

Biológicamente una neurona según María Villanueva en su monografía titulada "Las Redes Neuronales Artificiales y su importancia como herramienta para tomar decisiones", comenta que el sistema nervioso esta conformado por una red de células (neuronas), ampliamente interconectadas entre sí. En las neuronas, la información fluye desde las dendritas hasta el axón atravesando el soma. Se estima que el sistema nervioso tiene cien mil millones de neuronas.

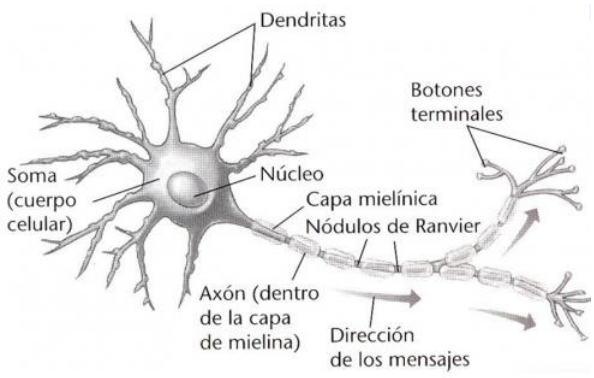


Figura 1: Neurona Biológica

Entorno al funcionamiento se describe los canales de entrada o dendritas, el órgano de cómputo o soma y el canal de salida o axón. Existen otras neuronas especializadas que son las interneuronas que llevan la información que cumplen la información que cumplen funciones de comunicación entre otras neuronas, existen las que llevan la información directamente al músculo llamadas neuronas motoras, y están las receptoras – sensoras que son las que reciben la información directamente del exterior. Se calcula aproximadamente que una neurona se intercomunica con otras 1000 neuronas y envía impulsos a varios cientos de ellas.

A nivel cerebral se observa una organización horizontal en capas y una vertical en forma de columnas, además hay unas funciones específicas para cada grupo de ellas y están en zonas particulares del cerebro y todos los subsistemas conforman el encéfalo.

La unión de dos neuronas o más recibe el nombre de sinapsis, que es direccional, es decir en un solo sentido. Cada neurona recibe impulsos eléctricos de otras a través de las dendritas y estas se conectan a la salidas de otras para producir la sinapsis, esta altera la efectividad de la señal transmitida debido a un parámetro, el peso. El aprendizaje es el resultado de la modificación de estos pesos y junto con el procedimiento de la información se genera el mecanismo básico de la memoria.

Esta explicación biológica fue la que sirvió de inspiración a muchos científicos a lo largo de la historia como lo menciona Ulises Román Concha en su artículo "Redes Neuronales y lo Autorreferente al ser humano".

Los principales acontecimientos históricos relacionados con las redes neuronales son:

- Walter Pitts y Warren McCulloch intentaron explicar en 1943 el funcionamiento del cerebro humano, por medio de una red de células conectadas entre sí que podían ejecutar operaciones lógicas[1].
- En 1949, el fisiólogo Donald Hebb expuso en su libro The Organization of Behavior la conocida regla de aprendizaje del mismo nombre. Su propuesta tenía que ver con la conectividad de la sinapsis, es decir, con las conexiones de las neuronas[2].
- En 1951, Minsky y Edmons montaron la primera máquina de redes neuronales, compuesta básicamente de 100 tubos de vacío y un piloto automático de un bombardero B – 24 (en desuso). Llamaron a su creación "Sharc", se trataba nada menos que de una red neuronal de 40 neuronas artificiales que imitaban el cerebro de una rata[3].
- En 1957, Frank Rosenblatt presentó al Perceptrón, una red con aprendizaje supervisado cuya regla de aprendizaje era una regla planteada por Hebb[4].
- En 1969, Marvin Minsky y Seymour Paper escribieron un libro llamado Perceptrons, en donde definían a estos como caminos sin salida[5].
- En los años 60 se propusieron otros dos modelos también supervisados, basados en el Perceptrón de Rosenblatt denominados Adaline y Madaline. En estos, la

[1] Gutiérrez, J. M. (9 de Marzo de 2000). Introducción a las Redes Neuronales. Retrieved 10 de Mayo de 2009 from José Manuel Gutiérrez. Pág. 29.

[2] Gutiérrez, J. M. (9 de Marzo de 2000). Introducción a las Redes Neuronales. Retrieved 10 de Mayo de 2009 from José Manuel Gutiérrez. Pág.2 9.

[3] José Manuel Gutiérrez (Universidad de Cantabria), Introducción a las redes Neuronales. Pág. 31

[4] José Manuel Gutiérrez (Universidad de Cantabria), Introducción a las redes Neuronales. Pág. 31

[5] José Manuel Gutiérrez (Universidad de Cantabria), Introducción a las redes Neuronales. Pág. 32

adaptación de los pesos se realiza teniendo en cuenta el error, calculando como la diferencia entre la salida deseada y la dada por la red, al igual que el Perceptrón. Sin embargo, la regla de aprendizaje empleada es distinta[6].

- En los años 70 las redes neuronales artificiales surgen con la técnica de aprendizaje de propagación hacia atrás o Backpropagation[7].
- En 1977, James Anderson desarrolló un modelo lineal, llamado Asociador lineal, que consistía en unos elementos integradores lineales (neuronas) que sumaban sus entradas[8].
- En 1982, John Hopfield presentó un trabajo sobre las redes neuronales a la Academia Nacional de las Ciencias, en el cual se describe con claridad y rigor matemático una red a la que ha dado su nombre, que viene a ser una variación de Asociador Lineal. En ese mismo año la empresa Fujitsu comenzó el desarrollo de computadoras pensantes para aplicaciones en robótica[9].
- En 1985, el Instituto Americano de Física comenzó la "Anual Networks for Computing"[10].
- En 1987, la IEEE celebró su primera conferencia internacional sobre las redes neuronales artificiales. En ese mismo año se formó la International Neural Network Society (INNS) bajo la iniciativa y dirección de Grossberg en USA, Kohonen en Finlandia y Amari en Japón[11].
- En 1987 se crea la Sociedad Europea de Redes Neuronales y la "Neural Information Processing Systems" reunión que se celebra anualmente en Denver Colorado[12].
- En 1988, resultó la unión de la IEEE y la INNS que produjo la "International Joint Conference on Neural Network" que realizó 430 artículos de los cuales 63 estaban enfocados a una aplicación[13].

---

[6] Alfonso Ballesteros (Málaga, España). Neural Network Framework. Computering Engineer by the University of Malaga. Pag. 3.

[7] Alfonso Ballesteros (Malaga, España). Neural Network Framework. Computering Engineer by the University of Malaga. Pag. 4.

[8] Basogain Olabe, X. (Diciembre de 2008). Redes Neuronales Artificiales y sus Aplicaciones. Retrieved 3 de Mayo de 2009 from Enseñanzas Técnicas. Pag. 54.

[9] Basogain Olabe, X. (Diciembre de 2008). Redes Neuronales Artificiales y sus Aplicaciones. Retrieved 3 de Mayo de 2009 from Enseñanzas Técnicas. Pag. 55.

[10] Basogain Olabe, X. (Diciembre de 2008). Redes Neuronales Artificiales y sus Aplicaciones. Retrieved 3 de Mayo de 2009 from Enseñanzas Técnicas. Pag. 55.

[11] Basogain Olabe, X. (Diciembre de 2008). Redes Neuronales Artificiales y sus Aplicaciones. Retrieved 3 de Mayo de 2009 from Enseñanzas Técnicas. Pag. 56.

[12] Basogain Olabe, X. (Diciembre de 2008). Redes Neuronales Artificiales y sus Aplicaciones. Retrieved 3 de Mayo de 2009 from Enseñanzas Técnicas. Pag. 56.

I.6 ESTADO DEL ARTE

En un principio los controladores de procesos eran diseñados mediante la obtención de modelos matemáticos, para posteriormente ser implementados al proceso para el cual fueron diseñados, pero a medida que los procesos eran más complejos, el modelo para su control también lo era, es por ello que surgió la necesidad de crear o desarrollar nuevas herramientas que permitieran reducir el tiempo de elaboración del controlador. Una de estas herramientas son las redes neuronales, las cuales permiten sustituir los modelos matemáticos por algoritmos de aprendizaje.

A continuación se describe algunos trabajos de redes neuronales artificiales para el reconocimiento de caracteres:

Trabajo Nº 1

Uso de una red neuronal multicapa para el reconocimiento de caracteres griegos.

En este trabajo realizado en el 2008 se propone la creación de una red multicapa con aprendizaje supervisado. La arquitectura de esta red es como se detalla a continuación:

- Alfabeto de 24 caracteres.
- Imágenes de 16x16 pixeles.
- Capa de entrada 256 x pixel.
- Capa de salida de 24 neuronas, 1 x letra.
- 2 capas ocultas.
- Todas las neuronas están conectadas con la capa superior.

Pre procesamiento

- Primero se carga las imágenes.
- Se procesa la imagen y se la ubica en una matriz dando 2 valores posibles 1 y -1.

Post procesamiento

Para conocer la respuesta de la red basta con tomar el carácter correspondiente a la neurona de salida con valor máximo.

Algoritmo utilizado: Backpropagation con mejoras. Estas mejoras se muestran a continuación:

---

[13] Basogain Olabe, X. (Diciembre de 2008). Redes Neuronales Artificiales y sus Aplicaciones. Retrieved 3 de Mayo de 2009 from Enseñanzas Técnicas. Pag. 57.

- Adaptative Learning Rate.

Adapta el parámetro según la evolución del error de aprendizaje. Si el error disminuye consistentemente, se le suma α. Cuando el error aumenta, se le resta b.

- Momentum.

La actualización de pesos toma en cuenta el valor del cambio producido en la época anterior.

P. García H. Rajchert I. Scena (4 de junio).

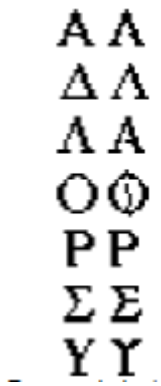


Figura 2: Reconocimiento de Caracteres Griegos

Trabajo Nº2

Procesado de encuestas

Otro trabajo similar es el reconocimiento de caracteres y marcas orientado al procesado de encuestas, detallado en **De Armas Domínguez & Bautista Rodríguez,** donde también se utiliza redes neuronales para el propósito mencionado. Se describe que para la clasificación de descriptores dentro del módulo de reconocimiento de caracteres se ha realizado una selección de redes neuronales para cada tipo de caracteres: alfabéticos, numéricos y alfanuméricos; todas las redes tienen 42 neuronas en la capa de entrada, se entrenaron con 19 tipos de caracteres y luego se validaron con otros 11 tipos.

La arquitectura de cada una de las redes neuronales queda como sigue:

- Caracteres alfabéticos, 100 neuronas en la capa de entrada, 50 en la capa oculta y 26 en la capa de salida.
- Caracteres numéricos, 85 neuronas en la capa de entrada, 45 en la capa oculta y 13 en la capa de salida.
- Caracteres alfanuméricos, 160 neuronas en la capa de entrada, 140 en la capa oculta y 39 en la capa de salida.

Además se hace mención que el reconocimiento de caracteres generó el 10,08% de errores probando las 15 hojas de encuestas y generó una precisión del 100% con otras pruebas realizadas con 50 hojas de encuesta.

Trabajo Nº3

Reconocimiento de imágenes con algoritmo Hopfield en Salta – Argentina

Se planteó el aprendizaje de tres patrones de 10x10 ingresados por diferentes métodos, para luego determinar el reconocimiento de un patrón de entrada ingresado con niveles de corrupción o falencia de datos variables, para así poder determinar el comportamiento de la red neuronal ante distintas contingencias y su porcentaje de reconocimiento de patrones.

Este trabajo esta compuesto por cuatro aspectos básicos:

Topología: Es una red monocapa con conexiones laterales no recurrentes, esto quiere decir que solo se tiene una capa de neuronas que se encuentran conectadas entre sí pero no con ellas mismas.

Aprendizaje: No supervisado.

Tipo de asociación entre la información de entrada y la salida: La función de activación de cada neurona (i) de la red (f (x)) es de tipo escalón, que trabaja con los valores binarios -1 y +1.

Representación de la información: Previamente codificada y representada en forma de vector como una configuración binaria “red discreta”.

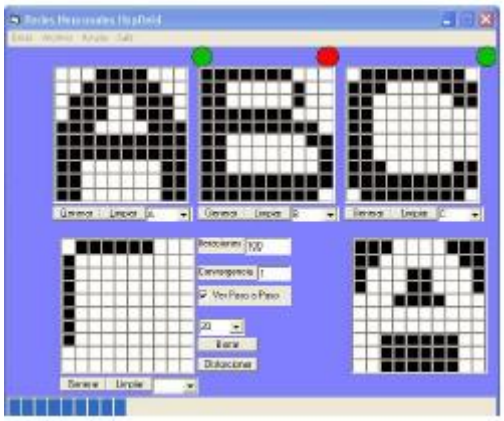


Figura 3: Aprendizaje con Redes de Hopfield

Trabajo Nº4

Lectura automática de cheques

En Palacios & Gupta, 2003 se muestra el desarrollo de un sistema de reconocimiento de caracteres para la lectura automática de cheques, en el cual se describe la utilización de técnicas basadas en redes neuronales artificiales para la lectura del valor del cheque. El módulo de reconocimiento de dígitos utiliza una serie de algoritmos de normalización y un clasificador. El clasificador que se utiliza esta basado en redes neuronales, por su rapidez y porque ofrece buenos resultados en el reconocimiento de caracteres.

La red neuronal artificial utilizada es un Perceptrón Multicapa (MLP), la estructura es multiconectada y tiene 117 neuronas, una capa oculta con 50 neuronas y 10 neuronas de salida, estas salidas corresponden a los 10 posibles dígitos. El nivel de precisión que se obtiene con este tipo de red es muy alto (92,2% de aciertos, con sólo 7,8% de fallos) cuando se entrena con una conjunto de ejemplos grande, tal como la base de datos de caracteres manuscritos de National Institute of Standars and Techonology.

CAPÍTULO II

DESARROLLO TÉCNICO Y RESULTADOS EXPERIMENTALES

I.1 MODELO BIOLÓGICO

La teoría de las redes neuronales artificiales esta inspirada en la estructura y el funcionamiento de los sistemas nerviosos, donde la neurona es el elemento fundamental. Existen neuronas de diferentes formas, tamaños. Estos atributos son importantes para determinar la función y utilidad de la neurona. La clasificación de estas células en tipo estándar ha sido realizada por muchos neuroanatomistas.

I.1.1 ESTRUCTURA DE LA NEURONA

Una neurona es la célula viva, y como tal, contiene los mismos elementos que forman parte de todas las células biológicas. Además contienen elementos característicos que las diferencian.

En general, una neurona consta de un cuerpo celular más o menos esférico de 5 – 10 micras de diámetro, del que salen una rama principal, el axón y varias ramas más cortas llamadas dendritas. A su vez el axón puede tener ramas entorno a su punto de arranque, y con frecuencia se ramifica extensamente cerca de su extremo.

Una de las características que diferencian a las neuronas del resto de las células, es la capacidad de comunicarse entre sí. En términos generales las dendritas y el cuerpo celular reciben señales de entrada; el cuerpo celular combina e integra y emite señales de salida. El axón transporta esas señales a las terminales axónicos que se encargan de distribuir información a un nuevo conjunto de neuronas. Por lo general, una neurona recibe información de miles de otras neuronas, y a su vez, envía información a miles de neuronas más.

I.1.2 NATURALEZA BIOLÓGICA DE UNA RED NEURONAL

Las señales que se utilizan son de dos tipos distintos de naturaleza: electricidad y química, la señal generada por la neurona y transportada a lo largo del axón es un impulso eléctrico, mientras que la señal que se transmite entre las terminales axónicos de una neurona y las dendritas de las neuronas siguientes es de origen químico; concretamente se realiza mediante moléculas de sustancias transmisoras (neurotransmisoras), que fluyen a través de contactos especiales, llamados sinapsis, que tienen la función de receptor y están entre las terminales axónicos y las dendritas de la neurona siguiente.

La generación de las señales eléctricas esta íntimamente relacionada con la composición de la membrana celular. La neurona como todas las células, es capaz de mantener en su interior un líquido cuya composición difiere principalmente en la composición de líquido exterior. La diferencia se da en la relación en la concentración de los iones de potasio y sodio. El medio externo es de 10 veces más rico en potasio que el medio interno mientras que el medio interno es 10 veces más rico en sodio que el medio externo.

La diferencia de concentración de iones de sodio y potasio en cada lado de la membrana, producen un diferencia de potencial aproximadamente de 70mV negativos en el interior de la célula. A ese potencial se lo conoce con el nombre potencial de reposo de la célula nerviosa, cuando la neurona esta en ese estado, se dice que esta en estado reposo.

La llegada de señales procedentes de otras neuronas a través de las dendritas (recepción de neurotransmisoras) actúa acumulativamente bajando ligeramente el valor del potencial de reposo. Dicho potencial modifica la permeabilidad de la membrana de manera que cuando llega a cierto valor crítico comienza una entrada masiva de iones de sodio que invierten la polaridad de la membrana.

La inversión del voltaje de la cara interior de la membrana cierra el paso de los iones de sodio y abre paso de los iones de potasio hasta que se restablece el equilibrio del reposo. La inversión del voltaje, es conocida como potencial de acción, se propaga a lo largo del axón, y su vez, provoca la emisión de los neurotransmisores en las terminales axónicos.

Después de un pequeño periodo refractorio, puede seguir un segundo impulso. El resultado de todo eso es la misión por parte de la neurona de trenes de impulso cuya frecuencia varía en función de la cantidad de las neurotransmisoras recibidas.

Existen dos etapas de sinapsis:

- Sinapsis excitadoras: cuyos neurotransmisoras provocan disminuciones de potencial en la membrana de la célula post – sináptica, facilitando la generación de los impulsos a mayor velocidad.
- Sinapsis inhibridas: cuyos neurotransmisoras tienden a estabilizar el potencial de la membrana, dificultando la emisión de los impulsos.

I.2 REDES NEURONALES ARTIFICIALES

El cerebro humano es un computador notable. Gran cantidad de investigaciones acerca de su funcionamiento, se realiza a cada momento. Es así como desde hace más de 30 años se desarrolla la teoría de las redes neuronales artificiales. Las cuales emulan las redes neuronales biológicas y que han sido utilizadas para resolver diversos problemas, aprendiendo estrategias de solución, basadas en ejemplos de comportamiento típico de patrones.

I.2.1 INTRODUCCIÓN A LAS REDES NEURONALES ARTIFICIALES

Una red neuronal artificial es un modelo de procesamiento de información que es inspirado por el modelo de un sistema nervioso biológico, tal como el cerebro procesa información.

Las redes neuronales artificiales, como la gente aprenden con ejemplos. Una red neuronal artificial es configurada para una aplicación específica, tal como reconocimiento de patrones o clasificación de datos, a través de un proceso de aprendizaje. Aprender en sistemas biológicos implica ajustes en las conexiones sinópticas que existen entre las neuronas, de igual manera lo hace una red neuronal artificial. Las redes neuronales artificiales se han aplicado a un gran número de problemas que son demasiados complejos para las tecnologías convencionales.

En general, a causa de su abstracción del cerebro biológico, las redes neuronales artificiales son aptas para resolver problemas que la gente puede resolver, pero las computadoras no pueden. Estos problemas incluyen reconocimiento de patrones y pronósticos (los cuales requieren el reconocimiento de tendencia de datos).

El modelo de una red neuronal artificial es construido averiguando cuales son los elementos relevantes del sistema simplificándolos, debido a la cantidad de información de la que dispone puede ser excesiva o redundante. La elección adecuada de una red se basa en sus características, una estructura conveniente, el tipo de aprendizaje, etc., es el procedimiento convencional.

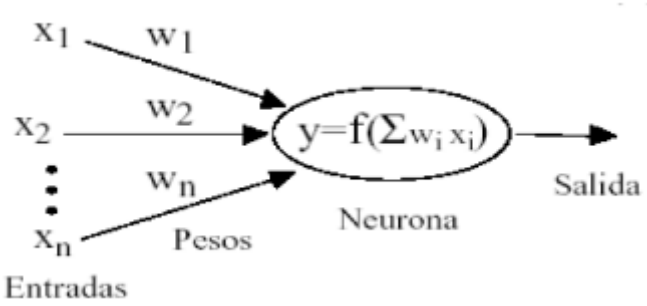


Figura 1: Red Neuronal Artificial

I.2.2 ELEMENTOS DE UNA RED NEURONAL ARTIFICIAL

Las redes neuronales artificiales son modelos que intentan reproducir el comportamiento del cerebro. Cualquier modelo de red neuronal artificial consta de dispositivos elementales de procesos: las neuronas, a partir de ellas, se pueden generar representaciones específicas, de tal forma que un conjunto de estado de ellas puede significar una letra, número o cualquier otro objeto. General se puede encontrar tres tipos de neuronas:

1. Aquellas que reciben estímulos externos, relacionados con el aparato sensorial, que tomaría la información de entrada.
2. Dicha información se transmite a ciertos internos que se ocupa de su procesado. Es en la sinapsis y las neuronas correspondientes a este nivel donde se genera cualquier tipo de representación interna de la información. Puesto que no tiene relación directa con la información de entrada ni con la de salida, estos elementos se denominan unidades ocultas.
3. Una vez finalizado el período de procesado, la información llega a las unidades de salida, cuya misión es dar la respuesta al sistema.

La red neuronal artificial pretende mimetizar las características más importantes de las neuronas biológicas. Cada neurona i – ésima está caracterizada en cualquier instante por un valor numérico denominado valor o estado de activación a_i (t); asociado a cada unidad, existe una función de salida f_i que transforma el estado actual de la activación de una señal de salida y_i. Dicha señal es enviada a través de los canales de comunicación unidireccionales a otras unidades de la red; en estos canales la señal se modifica de acuerdo a la sinapsis (el peso w_{ij}) asociada a cada uno de ellos según una determinada

regla. Las señales moduladas que han llegado a la unidad j – ésima se combinan entre ellos, generando así la entrada total net_j.

$$Net_j = \sum_i y_i w_{ji}$$

Una función de activación *f*, determina el nuevo estado de activación aj (t+1) de la neurona, teniendo en cuenta la entrada total calculada al anterior estado de activación a_j (t).

La dinámica que rige la actualización de los estados de las unidades (evolución de la red neuronal artificial), puede ser de dos tipos: modo asíncrono y modo síncrono. En primer caso, las neuronas evalúan su estado continuamente, según van llegando la información, y lo hacen de forma independiente. En el caso síncrono, la información también llega de forma continua, pero los cambios se realizan simultáneamente, como si existiera un reloj interno que decidiera cuándo deben cambiar de estado. Los sistemas biológicos quedan probablemente entre ambas posibilidades.

I.2.2.1 ESTADO DE ACTIVACIÓN

Adicionalmente al conjunto de unidades, la representación necesita los estados del sistema en un tiempo t, esto específica por un vector N, números reales a (t), que representa el *estado de activación* del conjunto de unidades de almacenamiento. Cada elemento del vector representa la activación de una unidad de tiempo t, la activación de una unidad U_i en el tiempo t se designa por a_i (t) es decir:

a (t) = (a_1 (t), a_2 (t), … a_i (t), … a_N (t))

El proceso que realiza la red se ve como la evolución de un patrón de activación en el conjunto de unidades que lo componen a través del tiempo.

Todas las neuronas que componen la red se hallan en cierto estado. En una visión simplificada, se puede decir que hay dos posibles estados, reposo y excitado, a los que denominamos globalmente *función de activación*, y a cada uno de los cuales se le asigna un valor. Los valores de activación pueden ser continuos o discretos. Además puede ser limitado o ilimitado. Si son discretos, suelen tomar un conjunto pequeño de valores o bien valores binarios.

En notación binaria, un estado activo se indicaría por un 1, y se caracteriza por la emisión de un impulso por parte de la neurona (potencial de acción), mientras que un

estado pasivo se indicaría por un 0, y significaría que la neurona esta en reposo. En otros modelos se considera un conjunto continuo de estados de activación, en lugar de solo dos estados, en cuyo caso se le asigna un valor entre [0,1] o en el intervalo [-1,1], generalmente siguiendo una función sigmoidal.

I.2.2.2 FUNCIÓN DE SALIDA O DE TRANSFERENCIA

Entre las unidades o neuronas que forma una red neuronal artificial existe un conjunto de conexiones que unen con las otras. Cada unidad transmite señales a aquellas que están conectadas con su salida. Asociada a cada unidad U, hay una función de salida $f_i(a_i(t))$, que transforma el estado actual de activación $a_i(t)$ en una señal de salida $y_i(t)$, es decir:

$y_i(t)= (f_1(a_1(t)), f_2(a_2(t)), \ldots\ f_i(a_i(t)), \ldots\ f_N(a_N(t)))$

Existen cuatro funciones de transferencia típicas que determinan distintos tipos de neuronas:

- **FUNCIÓN ESCALÓN**

La forma más fácil de definir la activación de una neurona es considerar que esta es binaria. La función de transferencia escalón se asocia a neuronas binarias en las cuales cuando la suma de las entradas es mayor o igual al umbral de la neurona, la activación es 1; si es menor la activación es 0 (ó -1).

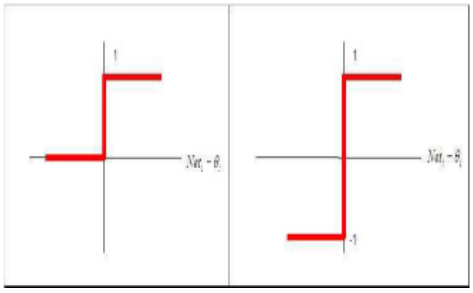


Figura 2: Función Escalón

- **FUNCIÓN LINEAL Y MIXTA**

La función lineal o identidad responde a la expresión f(x)=x. En las neuronas con función mixta, si la suma de las señales de entrada es menor que el límite inferior, la activación se define como 0 (ó -1). Si dicha suma es mayor o igual al límite superior,

entonces la activación se define como la función lineal de la suma de las señales de entrada.

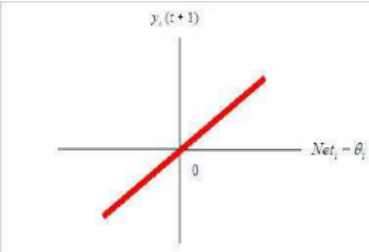


Figura 3: Función Lineal

- **FUNCIÓN SIGMOIDAL**

Cualquier función definida en un intervalo de posible valores de entrada, con un incremento monotónico y que tenga ambos limites inferiores (por ejemplo, las funciones sigmoidal o arco tangente), podrá realizar la función de activación o de transferencia de forma satisfactoria.

Con la función sigmoidal para la mayoría de los vectores del estímulo de entrada (variable independiente), el valor dado de la función es cercano a unos intervalos asintótico. Esto hace que la mayoría de los casos, el valor de la salida esta comprendido en la zona alta o baja de la sigmoide. De hecho cuando la pendiente es elevada, esta función tiende a la función escalón. Sin embargo la importancia de la función sigmoidal (o cualquier otra función similar), es que su derivada siempre es positiva y cercana al cero para los valores grandes positivos o negativos; además toma su valor máximo cuando x es 0. Esto hace que se puedan utilizar reglas de aprendizaje definidas por las funciones escalón con la ventaja de que la derivada en el punto de transición y esto no ayuda a los métodos de aprendizaje en los cuales se utilizan derivadas.

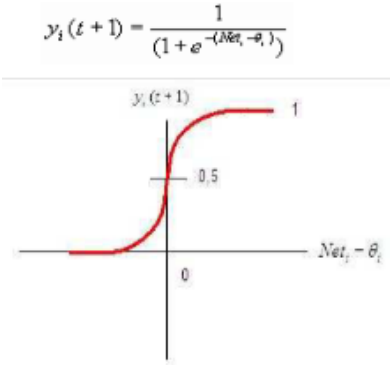


Figura 4: Función Sigmoidal

- **FUNCIÓN GAUSSIANA**

Los centros y anchura de estas funciones pueden ser adaptadas, lo cual las hace más adaptativa que la función sigmoidal. Mapeos que suelen requerir dos niveles ocultos (neuronas en la red que se encuentran entre la entrada y la salida), utilizando neuronas con funciones de transferencias sigmoidal algunas veces se puede realizar con un solo nivel en redes con neuronas con función gaussiana.

I.2.3 ESTRUCTURA DE UNA RED NEURONAL ARTIFICIAL

Los componentes más importantes de la red neuronal artificial:

- Unidades de procesamiento (la neurona artificial).
- Estado de activación de cada neurona.
- Patrón de conectividad entre neuronas.
- Regla de propagación.
- Función de transferencia.
- Regla de aprendizaje.
- Regla de activación.

Ajustados en las características de cada nodo de la red (microestructura), la estructura (mesoestructura) de la red neuronal artificial esta diseñada en función de:

- Número de niveles o capas.
- Número de neuronas por nivel.
- Patrones de conexión.
- Flujo de información.

I.2.3.1 NIVELES O CAPAS DE NEURONAS

La distribución de neuronas dentro de la red se realiza formando niveles o capas de un número determinado de neuronas cada una. A partir de su situación dentro de la red, se pueden distinguir tres tipos de capas:

- De entrada: Es la capa que recibe directamente la información proveniente de las fuentes externas a la red.
- De oculta: Son internas a la red y no tienen contacto directo con el entorno externo. El número de niveles ocultos pueden estar entre cero y un número grande. Las neuronas de las capas ocultas pueden interconectarse de distintas maneras lo que determina junto con su número las distintas topologías de la red hacia el exterior.
- De salida: Transfieren la información de la red hacia el exterior.

I.2.3.2 FORMAS DE CONEXIÓN ENTRE NEURONAS

La conectividad entre los nodos de una red neuronal esta relacionada con la forma en que las salidas de las neuronas están canalizadas para convertirse en entradas de otras neuronas. Las señal de salida de un nodo puede ser una entrada de otro proceso o incluso ser una entrada de sí mismo (conexión concurrente).

Cuando ninguna salida de las neuronas es entrada de neuronas del mismo nivel o niveles procedentes, la red se describe como *propagación hacia adelante* cuando las salidas pueden ser conectadas como entradas de neuronas de niveles previos o del mismo nivel, incluyéndose ellas mismas, la red es de *propagación hacia atrás*.

I.2.4 CARACTERISTICAS DE LAS REDES NEURONALES ARTIFICIALES

Existen cuatro aspectos que caracterizan una red artificial: su topología, el mecanismo de aprendizaje, tipo de asociación realizada entre la información de entrada y salida, y por último, la forma de representación de estas informaciones.

I.2.4.1 TOPOLOGÍA DE LAS REDES NEURONALES ARTIFICIALES

La topología o arquitectura de las redes neuronales consiste en la organización y disposición de las neuronas en la red formando capas o agrupaciones de neuronas más o menos alejadas de la entrada o salida de la red. Los parámetros fundamentales de la red: el número de capas, el número de neuronas en la capa, el grado de conectividad y el tipo de conexiones.

Cuando se realiza una clasificación de las redes neuronales en términos topológicos, se suelen distinguir entre las redes de una sola capa o nivel de neuronas y las redes con múltiples capas (1, 2, 3,… N).

- **REDES MONOCAPA**

En las redes monocapa, como la del Hopfield y la Brain – State – in – a – Box, se establecen conexiones laterales ente las neuronas que pertenecen a la única capa que constituye a la red. También pueden existir conexiones autorecurrentes (salida de una neurona conectada por su propia entrada), aunque en algún modelo, como el de Kohonen esta recurrencia no se utiliza.

Las redes monocapa se utilizan típicamente en tarea relacionada con lo que se conoce como autoasociación; por ejemplo para regenerar informaciones que se presentan en las redes incompletas o distorsionadas.

- **REDES MULTICAPA**

Las redes multicapa son aquellas que disponen de conjuntos de neuronas agrupadas de varios niveles o capas. En estos casos, una forma para distinguir la capa que pertenece una neurona, consistía en fijarse en el origen de las señales que recibe la entrada y el destino de la señal de salida. Normalmente todas las neuronas de una capa reciben señales de salida a una capa posterior, más cercana a la salida a la red. A estas conexiones se les denominan conexiones hacia adelante feedforward.

Sin embargo en un gran número de estas redes también existe la posibilidad de conectar las salidas de las neuronas posteriores a las entradas de las capas anteriores, a estas conexiones se le denominan hacia atrás feedback.

Estas dos posibilidades permiten distinguir entre dos tipos de redes múltiples capas: las redes con conexiones hacia adelante o redes feedforward y las redes que disponen con conexiones hacia adelante como hacia atrás o redes feedforward/ feedback.

I.2.4.2 MECANISMOS DE APRENDIZAJE DE LAS REDES NEURONALES ARTIFICIALES

El aprendizaje es el proceso por el cual una red modifica sus pesos en respuesta de la información de entrada. Los cambios que se producen durante el proceso de aprendizaje se reducen a la destrucción, modificación y creación de conexiones entre neuronas. En los modelos de las redes neuronales artificiales la creación de la nueva conexión implica

que el peso de la misma pasa para tener un valor distinto de cero. De la misma forma, una conexión se destruye cuando su peso se va a cero.

Durante el proceso de aprendizaje, los pesos de las conexiones de la red sufren modificaciones, por tanto se puede afirmar que este proceso ha terminado (la red ha aprendido), cuando los valores de los pesos permanecen estables ($dw_i/dt=0$).

Es por ello que una de las clasificaciones que se realizan de las redes neuronales obedece al tipo de aprendizaje utilizado por dichas redes. Así se puede distinguir:

- **REDES NEURONALES CON APRENDIZAJE SUPERVISADO**

El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor o maestro), que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor comprueba la salida de la red y en caso de que esta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada.

- **REDES NEURONALES CON APRENDIZAJE NO SUPERVISADO**

El aprendizaje no supervisado se caracteriza (también conocido como auto – supervisado) no requiere de influencia externa para ajustar los pesos de las conexiones entre las neuronas, la red no recibe ninguna determinada entrada para ver si es o no es la correcta; por ello suele decirse que estas redes son capaces de autoorganizarse.

También el aprendizaje sin supervisión permite realizar una codificación de datos de entrada, generando a la salida una versión codificada de la entrada, con menos bits, pero manteniendo la información relevante de los datos.

Finalmente algunas redes con aprendizaje no supervisado lo que realiza es un *mapeo de las características (feature mapping)*, obteniendo en la neurona de salida una disposición geométrica que representa un mapa topográfico de las características de los datos de entrada, de tal forma que se presentan a la red entre sí, en la misma zona del mapa.

- **REDES NEURONALES CON APRENDIZAJE COMPETITIVO**

En dicho aprendizaje suele decirse que las neuronas compiten(cooperan) unas con otras con el fin de llevar una tarea dada.

La competición entre neuronas se realiza en todas las capas de la red, existiendo en estas neuronas conexiones recurrentes de autoexcitación y conexiones de inhibición por parte de las neuronas vecinas. Si el aprendizaje es cooperativo, estas conexiones con las vecinas serán de excitación.

El objetivo de este aprendizaje es *clusterizar* los datos que se introducen a la red. De esta forma, las informaciones similares son clasificadas formando parte de la misma categoría, y por tanto deben activar la misma neurona de salida. Las categorías deben ser creadas por la misma red, puesto que se trata de aprendizaje no supervisado, a través de las correlaciones entre los datos.

En este tipo de redes, cada neurona tiene asignado un peso total, suma de todos los pesos de las conexiones que tiene a su entrada. El aprendizaje afecta sólo a las neuronas ganadoras (activas), redistribuyendo este peso total entre las conexiones que llegan a la neurona vencedora y repartiendo esta cantidad por igual entre las conexiones procedentes de unidades activas. Por tanto, la variación del peso de una conexión entre la unidad *i* y otra *j* será nula si la neurona *j* no recibe excitación por parte de la neurona *i* y otra *j* será nula si la neurona *j* no recibe excitación por parte de la neurona *i*, y se modificará si es excitada por dicha neurona *i*.

Existe otro caso particular de aprendizaje competitivo, denominado teoría de resonancia adaptativa, desarrollado por Carpenter y Grossberg en 1986 y utilizado en la red feedforward/feedback de dos capas conocida como ART. Esta red realiza un prototipado de las informaciones que recibe a la entrada, generando como salida un ejemplar o prototipo que representa a todas las informaciones que podrían considerarse pertenecientes a la misma categoría.

- **REDES HIBRIDAS**

Es un enfoque mixto en el que se utiliza una función de mejora para facilitar la convergencia. Un ejemplo de este último tipo son las redes de base radial.

- **APRENDIZAJE REFORZADO**

Basado en un proceso de prueba y error que busca maximizar el valor esperado de una función criterio conocida como una *señal de refuerzo*. La idea de este paradigma surge

en la psicología en relación con el estudio del aprendizaje en los animales. Si una acción supone una mejora en el comportamiento entonces la tendencia a producir esta acción se refuerza y en caso contrario se debilita.

I.3 RED NEURONAL ARTIFICIAL: PERCEPTRÓN

El Perceptrón es un tipo de red neuronal artificial desarrollado por Frank Rosenblatt, también puede entenderse como Perceptrón la neurona artificial y unidad básica de inferencia en forma de discriminador lineal, que constituye este modelo de red neuronal artificial, esto debido a que el Perceptrón puede usarse como neurona dentro de un Perceptrón más grande y otro tipo de red neuronal.

El concepto más básico que permite comenzar a entender un Perceptrón es asociarlo a un sensor, ya sea de temperatura, humedad, nivel de líquidos, grado de acidez, coloración, densidad, etc. Es, en esencia, un dispositivo que, dada la presencia de uno (o varios) fenómenos(s) de entrada, permite representarlos(s) mediante la señal de salida fácilmente reconocible. Si damos a este simple dispositivo de varios canales de entrada(dos o más), le habremos agregado una notable mejoría ya que podrá discriminar entre fenómenos de entrada variables y entregarnos una salida que representará el criterio discriminador o resultado de la interacción de las entradas.

El modelo biológico más simple de un Perceptrón es una neurona y viceversa. Es decir, el modelo matemáticos más simple de la neurona es un Perceptrón. La neurona es una célula especializada y caracterizada por poseer una cantidad indefinida de canales de entrada llamados dendritas y un canal de salida llamado axón. Las dendritas operan como sensores que recogen información de la región donde se hallan y la derivan hacia el cuerpo de la neurona que reacciona de alguna manera y produce una respuesta entregada por el axón.

Una neurona sola y asilada carece de razón de ser. Su labor especializada se torna valiosa en la medida en que se asocia a otras neuronas, formando una red. Normalmente, el axón de una neurona entrega su información como “señal de entrada” a una dendrita de otra neurona y así sucesivamente. El Perceptrón aludido en adelante se entiende como formando por redes neuronales sean éstas biológicas o de sustratos semiconductor(compuertas lógicas).

El Perceptrón usa una matriz para representar las redes neuronales y es un discriminador terciario que traza se entrada *x*(un vector binario) a un único valor de salida *f(x)* (un solo valor binario) a través de dicha matriz.

$$f(x) = \begin{cases} 1 & \text{si } w \cdot x - u > 0 \\ 0 & \text{en otro caso} \end{cases}$$

Donde *w* es un vector de pesos reales y *w.x* es el producto punto(que computa una suma ponderada), *u* es el "umbral", el cual representa el grado de inhibición de la neurona, es un término constante que no depende del valor que tome la entrada.

El valor de *f(x)* (0 ó 1) se usa para clasificar *x* como un caso positivo o un caso negativo, en el caso de un problema de clasificación binario. El umbral puede pensarse de cómo compensar la función de activación, o dando un nivel bajo de actividad a la neurona del rendimiento. La suma ponderada de las entradas debe producir un valor mayor que *u* para cambiar la neurona de estado 0 a 1.

El Perceptrón se define como una red de dos capas (una de entrada y una salida con conexiones hacia adelante).

El primer nivel esta compuesto por un número de unidades de entrada, denominadas unidades sensoriales. El segundo nivel está compuesto por un número de unidades de salida, denominadas unidades de asociación, cuyas entradas son las salidas de las unidades de entrada ponderada por unos pesos.

Las unidades de entrada tienen una sola entrada correspondiente a una de las entradas de la red, y una sola salida. Estas unidades transmiten la señal que aparecen en su entrada.

I.3.1 REGLA DE APRENDIZAJE DE LA RED NEURONAL PERCEPTRÓN

PASO 1: Inicialización del umbral: estos valores son aleatorios.

PASO 2: Presentación de nuevo par (entrada, salida esperada).

$X_p=X_1, X_2, X_n$ junto a la salida deseada d(t)

PASO 3: Calcular la salida actual

$$y(t) = f[\sum_{i=0}^{n} W(t)\, X(t) - \theta]$$

Donde:

w(t): pesos actuales

x(t): vector de entrada

y(t): salida actual

f(x): función de transferencia

PASO 4: Adaptación de los pesos es necesaria solo si la salida no responde a lo que se esperaba.

$w(t+1)=w_i+\alpha[d(t)+y(t)]x_i(t)$

Donde:

w(t): pesos actuales

x(t): vector de entrada

y(t): salida actual

d(t): salida deseada

α: factor de aprendizaje que puede variar entre 0.0 a 1.0

PASO 5: Volver al PASO 2 hasta que la red proporcione una salida válida para todos los patrones que se le pretende enseñar.

I.3.2 LIMITACIONES DE LA RED PERCEPTRÓN

Se debe tener en cuenta que siempre el algoritmo de entrenamiento de Perceptrón podrá converger hacia un error nulo. De hecho el Perceptrón es incapaz de converger en aquellas funciones que no son linealmente separables, es decir, aquellas cuyos elementos pueden ser separados por una línea recta.

Esto se debe a las propiedades inherentes de las unidades básicas del Perceptrón que son las redes neuronales artificiales cuya limitación reside principalmente en la función de activación.

Visto de otra forma, podríamos decir que el Perceptrón divide en dos grupos las entradas por medio de una línea divisoria de manera que no es posible separar elementos que no se encuentren claramente separados de otros elementos. Es decir que se puede caracterizar elementos no lineales.

I.3.3 REGLA DELTA

- Los patrones de entrenamiento están constituidos por pares de valores(x,d) que son el vector de entrada y su salida deseada.
- La regla Delta utiliza la diferencia entre la salida producida para cada patrón(p) y la salida (d^p-y^p).
- Se calcula una función de error para todo el conjunto de patrones.

$$E = \sum_{p=1}^{m} E^p = \frac{1}{2}\sum_{p=1}^{m}(d^p - y^p)^2$$

Error global — Error cuadrático por patrón

- La regla Delta busca el conjunto de pesos que minimiza la función de error.
- Se hará mediante un proceso iterativo donde se van presentando los patrones uno a uno y se van modificando los parámetros de la red mediante la regla del descenso del gradiente.
- La idea es realizar un cambio en cada peso proporcional a la derivada del error, medida en el patrón actual, respecto del peso

$$\Delta_p w_j = -\gamma \frac{\partial E^p}{\partial w_j}$$

PASO 1: Inicialización con valores aleatorios y pequeños

PASO 2: Aplicar un patrón x^p y calcular la salida y , donde x_i, y d ϵ {-1,1}

PASO 3: Se calcula δ=(d-y) y se utiliza η(coeficiente de aprendizaje):

$wi(t+1)=w_i(t)+\eta\delta x_i$

Sea $\Delta_i= \eta\delta x_i$ => $wi(t+1)=w_i(t)+ \Delta_i$

Paso 4: Volver al paso 1

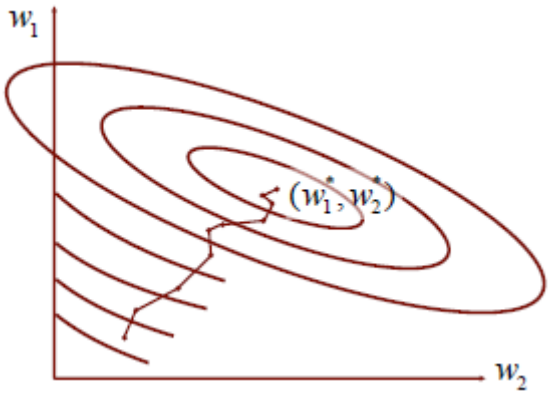


Figura 5: Regla Delta

1.- Se aplica $\vec{x}_p$

2.- Se calcula $E_p(w(t)) = (d_p \quad \vec{w}^T \vec{x}_p)^2$

3.- Se calcula $\Delta E_p(t) = \Delta_i^p = -2\delta_p x_{ip}$

4.- Se actualiza $\vec{w}(t+1) = \vec{w}(t) + \eta \delta_p x_{ip}$

I.3.4 RED NEURONAL: PERCEPTRÓN MULTICAPA

El Perceptrón multicapa es una red neuronal artificial formada por múltiples capas, esto le permite resolver problemas que no son linealmente separables, lo cual es la principal limitación del Perceptrón (también llamado Perceptrón simple). El Perceptrón multicapa puede ser totalmente o localmente conectado. En el primer caso cada salida de una neurona de la capa "i" es entrada de todas las neuronas de la capa "i+1", mientras que en el segundo cada neurona de la capa "i" es entrada de una serie de neuronas (región) de la capa "i+1".

Los modelos que se desprenden del Perceptrón se basan en los principios de corrección de error planteados por el algoritmo de la regla delta para entrenar a estos sistemas. En un principio el desarrollo del Perceptrón llevó a la generación de un nuevo tipo de red cuya modificación principal respecto a la estructura del Perceptrón se basa en el uso de varias capas de neuronas artificiales, en vez de usar una sola capa.

Este hecho significativo no hubiera servido de nada sin el cambio de la función de activación de las neuronales artificiales pasando de una función no diferenciable como era la activación logística a una función diferenciable y no lineal como es la sigmoide que se muestra en la figura 9:

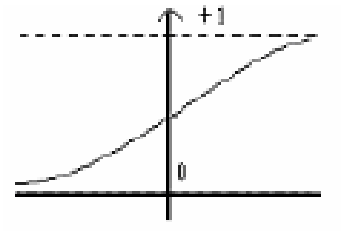


Figura 6: Función Sigmoide

El recurso de este tipo de función de activación introdujo un nuevo paradigma en el procedimiento de los sistemas neuronales permitiendo a las redes neuronales aprender las variaciones no lineales de los distintos tipos de ambientes, que en su mayoría, presentan variaciones de tipo lineal.

Se puede entender la importancia de este trabajo ya que la mayor parte del tiempo el flujo de datos en las redes de comunicaciones sucede aleatorio y discontinuo. Este tipo de características son precisamente las que el nuevo sistema neuronal nos permitirá asimilar. Este sistema neuronal considerado también una red neuronal se como en la literatura como Perceptrón Multicapa debido a que parte del principio del Perceptrón simple. A continuación se muestra una arquitectura de esta red neuronal.

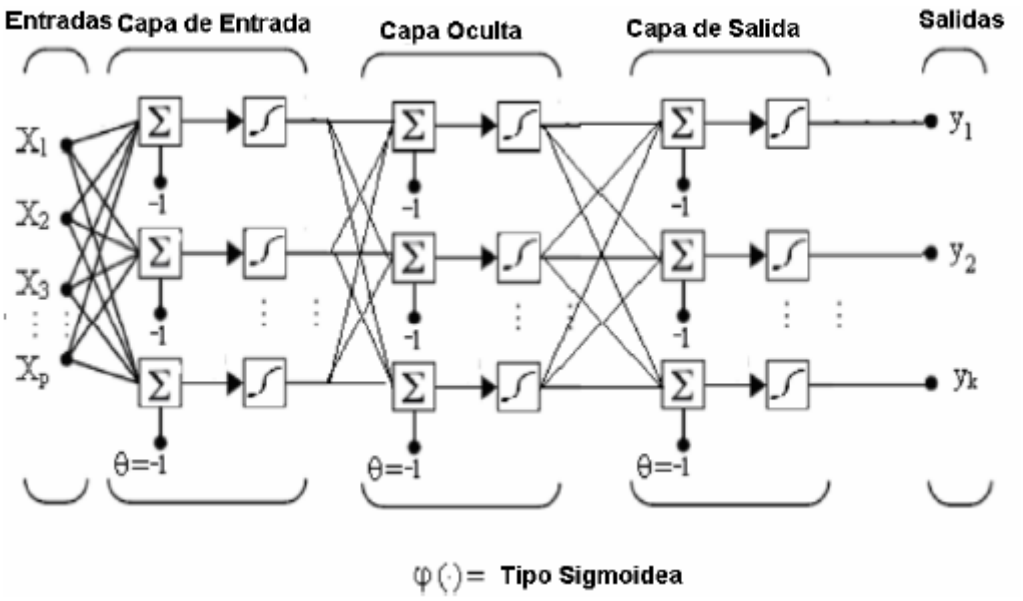


Figura 7: Arquitectura Perceptrón Multicapa

Al observar la arquitectura del Perceptrón Multicapa se puede observar que las múltiples entradas conectadas en la primera capa son mapeadas en las salidas en función de las distintas capas de neuronas intermediarias y de los parámetros libres de la red. Se puede ver una caja negra que realiza una operación sobre las entradas, produciendo un rango de salidas en función de los parámetros libres

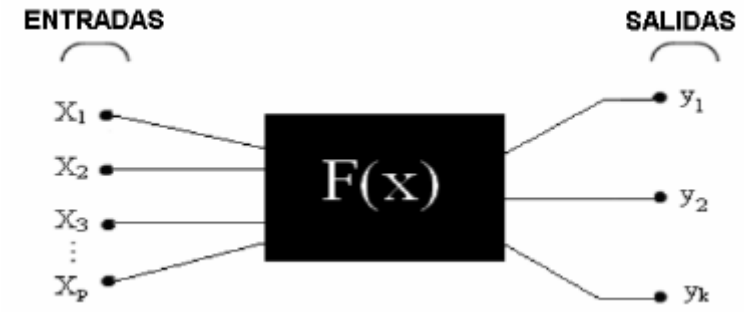


Figura 8: Perceptrón Multicapa visto como una caja negra

Se infiere de la arquitectura que el algoritmo de entrenamiento de una red con tales características deberá ser planeado para que los cambios en los parámetros libres sean tales que el error en las unidades básicas de las estructuras sea mínimo de manera que el conjunto de los cambios produzca un error global que tienda al mínimo. Se buscará entonces el límite en el cual la configuración de los parámetros libres produzca un error mínimo.

Tomando en cuenta este razonamiento, la evolución al Perceptrón Multicapa tuvo que basar su éxito en el diseño del algoritmo de entrenamiento de lograra minimizar el error al modificar adecuadamente los pesos y umbrales. La historia marco como primer paso, estudiar la forma de minimizar el error en una capa de neuronas lineales, que se conoce también como filtro lineal. El análisis de este tipo de red neuronal que posee elementos lineales, nos permite deducir un algoritmo más complejo para entrenar a una red como la Perceptrón Multicapa que posee elementos no lineales.

I.3.4.1 CARACTERISTICAS DEL PEERCEPTRÓN MULTICAPA

- Es una de la arquitectura más utilizada para resolver problemas reales.

- Se evalúa con un conjunto de datos de entrada y se obtienen valores o vectores con valores reales.
- Se diferencia del Perceptrón simple y Adaline en que tiene una capa oculta.
- Todas las neuronas se relacionan con todas incluyendo las neuronas de la capa oculta.
- Se trata de una estructura altamente no lineal.
- Presenta tolerancia a fallos.
- El sistema es capaz de establecer una relación entre dos conjuntos de datos
- Existe la posibilidad de realizar una implementación hardware.
- Puede aproximar cualquier función continua y=f(x) con soporte compacto hasta una precisión dada.
- Generaliza la respuesta frente a nuevos patrones de entrada(basado en aproximación/interpolación).

I.3.4.2 LIMITACIONES DE LA RED NEURONAL PERCEPTRÓN MULTICAPA

- El Perceptrón multicapa no extrapola bien, es decir, si la red se entrena mal o de manera insuficiente, las salidas pueden ser imprecisas.
- La existencia de mínimos locales en la función de error dificulta considerablemente el entrenamiento se detiene aunque no se haya alcanzado la tasa de convergencia fijada.

Cuando baja en un mínimo local sin satisfacer el porcentaje de error permitido se puede considerar: cambiar la topología de la red(número de capas y números de neuronas), comenzar el entrenamiento con unos pesos iniciales diferentes, modificar los parámetros de aprendizaje, modificar el conjunto de entrenamiento o presentar los patrones en otro orden.

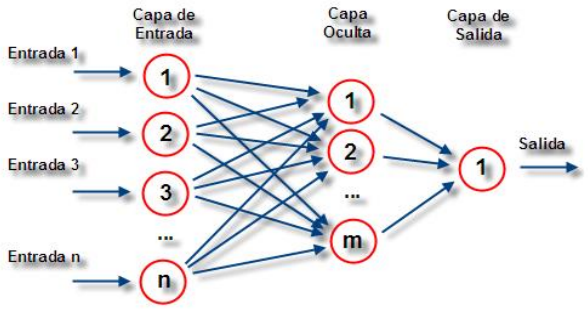


Figura 9: Perceptrón Multicapa

I.4 RED NEURONAL ARTIFICIAL: ADALINE

La red Adaline esta limitado a una única neurona de salida, la diferencia fundamental con respecto al Perceptrón es el mecanismo de aprendizaje.

Adaline utilizada la denominada Regla Delta de Hidrow o regla de mínimo error cuadrático medio(LMS), basada en la búsqueda del mínimo de una expresión del error entre la salida deseada y la salida lineal obtenida antes de aplicarle la función de activación escalón(frente a la salida binaria utilizada en el caso del Perceptrón).

Debido a esta nueva forma de evaluar el error. Estas redes neuronales artificiales pueden procesar información analógica, tanto de entrada como de salida, utilizando la función lineal sigmoidal.

En cuanto a la estructura de Adaline, es casi idéntica a la del Perceptrón elemental, sus autores consideran que esta formada por un elemento denominado *combinador adaptativo lineal*(ALC), que obtiene la salida lineal (*s*) que puede ser aplicada a otro elemento de conmutación bipolar, de forma que si la salida del ALC, es positiva la salida de la red Adaline es +1; si la salida del ALC es negativa entonces la salida de la salida Adaline es -1.

La red Adaline se puede utilizar para generar una salida analógica utilizando un conmutador sigmoidal, en lugar del binario; en tal caso, se obtendrá la salida aplicando la función de tipo sigmoidal como la tangente hiperbólica(tanh(s) o la expresión $(1/1+e^{-s})$).

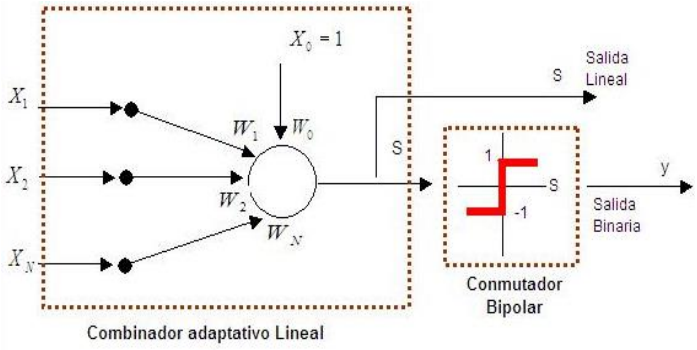
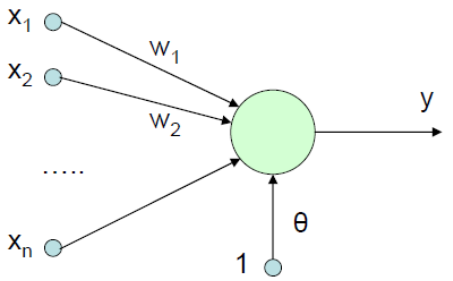


Figura 10: Combinador Adaptativo Lineal

I.4.1 ARQUITECTURA DE LA RED NEURONAL ARTIFICIAL ADALINE

- En Adaline se utiliza directamente la salida de la red teniendo en cuenta cuánto se ha equivocado.

- Se utiliza la diferencia entre el valor real esperado d y la salida producida por la red y. Para un patrón de entrada xp, se tendrá en cuanta el error producido (dy-yp).
- El objetivo es obtener una red tal que yp≈dp para todos los patrones p



$$y = \sum_{i=1}^{n} w_i \cdot x_i + \theta$$

Figura 11: Arquitectura de la Red Adaline

I.4.2 CARACTERISTICAS DE LA RED ADALINE

- Es un tipo de aprendizaje OFF LINE.
- Se enmarca dentro del tipo de aprendizaje por corrección de error.
- Se utiliza para entrenar un elemento simple procesado, con una función de transferencia lineal.
- Se puede combinar un cierto número de patrones de entrada en la capa de salida (estructura con cierto grado de complejidad). La regla sobre cada uno de los patrones de entrada de manera individual.

I.4.3 REGLA DE APRENDIZAJE DE LA RED ADALINE

PASO 1: Se aplica un vector de entrada X_i para la red Adaline.

$$\{X^1, X^2, X^3, \ldots\ldots, X^L\}$$

PASO 2: Se obtiene la salida lineal.

$$S_k = X_k W^T = \sum_{j=0}^{N} w_j x_{k_j}$$

Donde:

w_j: pesos actuales.

x_{kj}: vectores de entrada.

PASO 3: Se calcula la diferencia con respecto a la salida deseada.

$$\varepsilon_k = (x_k - s_k)$$

Donde:

s_k: salida lineal.

x_k: vector de entrada.

PASO 4: Se actualizan los pesos

$$W(t+1) = W(t) + \alpha\varepsilon_k X_k = W(t) + \alpha(d_k - s_k)X_k$$

Donde:

w(t): pesos actuales.

x(t): vectores de entrada.

s_k: salida lineal.

d_k: salida deseada.

α: factor de aprendizaje que puede variar entre 0.0 a 1.0.

PASO 5: Se repite los pasos 1 al 3 con los vectores de entrada.

PASO 6: Se calcula el error cuadrático.

$$< \varepsilon_k^2 >= \frac{1}{2L}\sum_{k=1}^{L}\varepsilon_k^2$$

Donde:

L: número de entradas que se utilizó al inicio de la fase de aprendizaje.

ε_k: salida lineal.

I.4.4 RED NEURONAL ARTIFICIAL: MADALINE

El entrenamiento de esta red no puede ser el mismo que el de la red Adaline, el algoritmo de mínimo error cuadrático medio(LMS) podría aplicarse a la capa de salida, puesto que se conoce el vector de salida deseada para cada una de las tramas de entrada de entrenamiento. Sin embargo, lo que se desconoce es la salida deseada para los nodos de cada una de las capas ocultas. Además, el algoritmo LMS funciona para las salidas lineales(analógicas) del combinador adaptativo y no para las digitales de Adaline, la

forma de aplicar las ideas del algoritmo LMS para entrenar una estructura tipo Madaline pasa por sustituir la función de salida por una función continua derivable(la función umbral es descontinua en 0 y por tanto no derivable es el punto).

Existe otro método, como regla II de Madaline(MRII) parecido a un procedimiento de acierto y error con una inteligencia adicional basada en un principio de mínima perturbación(Widrow). El entrenamiento equivale a hacer una reducción del número de neuronas de salidas incorrectas para cada una de las tramas de entrenamiento que se den como entrada. Este método se puede aplicar mediante el siguiente algoritmo:

1. Se aplica un vector a las entradas de Madaline y se hace que se propague hasta las unidades de salida.
2. Se cuenta el número de valores incorrectos que hay en la capa de salida, denominándose error de dicho número.
3. Para las unidades de capa de salida:
    - Se selecciona la primera neurona que no haya sido seleccionada antes y cuya salida lineal este mas próxima a cero. Esta es la neurona que puede cambiar su salida binaria con el menor cambio de sus pesos, y según el principio de mínima perturbación, debe tener prioridad en el proceso de aprendizaje.
    - Se cambian los pesos de la neurona seleccionada de tal modo que cambie su salida binaria.
    - Se hace que se propague el vector de entrada hacia adelante, partiendo de las entradas y en dirección a las salidas, una vez más.
    - Se emite el cambio de pesos si ha dado lugar a una reducción del error, en caso contrario, se restauran los pesos originales.
4. Se repite el paso 3 para todas las capas, salvo de la salida.
5. Para las unidades de la capa de salida:
    - Se selecciona el par de neuronas que no hayan sido seleccionadas anteriormente y cuya salidas lineales estén más próximas a cero.
    - Se aplica una corrección de pesos a ambas neuronas para modificar el valor de su salida.
    - Se hace que se propague hacia adelante el vector de entradas desde las entradas hasta las salidas.

- Se admite el cambio de pesos si ha dado lugar a una reducción del error; en caso contrario, se restauran los pesos originales.

6. Se repite el paso 5 para todas las capas, salvo de la salida.

Los pasos 5 y 6 se pueden repetir con grupos de tres, cuatro o mayor número de neuronas hasta obtener resultados satisfactorios. Se considera que las parejas son apropiadas para redes que tengan un máximo de 25 neuronas por capa aproximadamente.

I.4.4.1 CARACTERÍSTICA DE LA RED MADALINE

Una de las características más llamativas de la red Madaline es la ausencia de pesos asociados a las conexiones entre las unidades Adaline y Madaline.

I.5 RED NEURONAL ARTIFICIAL: BACKPROPAGATION

En 1986, Rumelhart, Hinton y Williams, formalizaron un método para que una red pueda aprender la asociación que existe entre los patrones de entrada y las clases correspondientes, utilizando varios niveles de neuronas.

El método Backpropagation (propagación del error hacia atrás), basado en la generalización de la Regla Delta, a pesar de sus limitaciones ha ampliado de forma considerable el rango de aplicaciones de las redes neuronales.

El funcionamiento de la red Backpropagation consiste en el aprendizaje de un conjunto predefinido de pares de entradas – salidas dados como ejemplo: primero se aplica un patrón de entrada como estímulo para la primera capa de las neuronas de la red, se va propagando a través de todas las capas superiores hasta generar la salida, se compara el resultado en las neuronas de salida con la salida que se desea obtener y se calcula un valor de error para cada neurona de salida. A continuación, estos errores se transmiten hacia atrás, partiendo de la capa de salida hacia todas las neuronas de la capa intermedia que contribuyan directamente a la salida, recibiendo de error aproximado a la neurona intermedia a la salida original. Este proceso se repite, por cada capa, hasta que todas las neuronas de la red hayan recibido un error que describa su aportación relativa al error total. Basándose en el valor del error recibido, se reajusta los pesos de conexión de cada neurona, de manera que en la siguiente vez que se presente el mismo patrón, la más salida este cercana a la deseada.

La importancia de la red Backpropagation consiste en su capacidad de autoadaptar los pesos de las neuronas de las capas intermedias para aprender la relación que existe entre un conjunto de patrones de entrada y sus salidas correspondiente. Es importante la capacidad de generalización, facilidad de dar salidas satisfactorias a entradas que el sistema no ha visto nunca en su fase de entrenamiento. La red debe encontrar una representación interna que le permita generar las salidas deseadas cuando se le dan entradas de entrenamiento, y que pueda aplicar, además, a entradas no presentadas durante la etapa de aprendizaje para clasificarlas.

Provee salidas satisfactorias a entradas que no ha visto nunca en su etapa de entrenamiento.

I.5.1 REGLA DE APRENDIZAJE DE LA RE D DE BACKPROPAGATION

PASO 1: Inicializar los pesos con valores pequeños aleatorios.

PASO 2: Presentación de patrón de entrada y especificar la salida deseada que debería generar la red.

PASO 3: Calcular la actual red. Para ello se presenta las entradas a la red y vamos calculando la salida que presenta cada capa hasta llegar a la capa de salida, esta será la salida de la red. Los pasos son los siguientes:

- Para una neurona j oculta

$$net_{pj}^{h} = \sum_{i=1}^{M} w_{ji}^{h} x_{pi} + \theta_{j}^{h}$$

En donde el índice *h* se refiere a magnitudes de la capa oculta; el subíndice *p*, al p – ésimo vector de entrenamiento y j a la j – ésima neurona oculta. El término θ puede ser opcional, pues actúa como una entrada más.

- Se calculan las salidas de las neuronas ocultas.

$$y_{pj} = f_{j}^{h}(net_{pj}^{h})$$

- Se realiza los mismos cálculos para obtener las salidas de las neuronas de salida(capa output).

$$net_{pk}^{o} = \sum_{j=1}^{L} w_{kj}^{o} y_{pj} + \theta_{k}^{0}$$

$$y_{pk} = f_{k}^{0}(net_{pk}^{o})$$

PASO 4: Calcular los términos del error para todas las neuronas

$$\delta_{pk}^{0} = (d_{pk} - y_{pk}) f_k^{o'}(net_{pk}^{o})$$

La función *f*debe ser derivable. En general se dispone de dos formas de función de salida:

La función lineal:

$$f_k(net_{jk}) = net_{jk}$$

La función sigmoidal:

$$f_k(net_{jk}) = \frac{1}{1 + e^{-net_{jk}}}$$

La selección de la función depende de la forma que se decida representar la salida: si se desea que las neuronas de salida sean binarias, se utilizan la función sigmoidal, en otros casos, la lineal.

Para una función lineal, tenemos: $f_k^{o\,'} = 1$, mientras que la derivada de una función sigmoidal es: $f_k^{o\,'} = f_k^{o}(1 - f_k^{o}) = y_{pk}(1 - y_{pk})$ por lo que los términos de error para las neuronas de salida quedan:

$\delta_{pk}^{o} = (d_{pk} - y_{pk})$ para la salida lineal.

$\delta_{pk}^{o} = (d_{pk} - y_{pk}) y_{pk}(1 - y_{pk})$ para la salida sigmoidal.

Si la neurona j no es de la salida, entonces la derivada parcial del error no puede ser evaluada directamente, por tanto se obtiene el desarrollo a partir de valores que son conocidos y otros que pueden ser evaluados.

La expresión obtenida en este caso es: $\delta_{pj}^{h} = f_j^{h\,'}(net_{pj}^{h}) \sum_{k} \delta_{pk}^{0} w_{kj}^{0}$ donde podemos observar que el error en las capas ocultas depende de todos los términos de error de la capa de salida. De aquí surge el término *propagación hacia atrás*.

PASO 5: Actualizar los pesos

Para ello se utiliza un algoritmo recursivo, comenzando por las neuronas de salida y trabajando hacia atrás hasta llegar a la capa de entrada, ajustando los pesos de la siguiente forma:

- Salida

$$\Delta w_{ij} = y_i * y_j$$

$$\Delta w_{kj}^{o}(t+1) = \alpha \delta_{pk}^{o} y_{pj}$$

- Oculta

$$w_{ji}^{h}(t+1) = w_{ji}^{h}(t) + \Delta w_{ji}^{h}(t+1)$$

$$\Delta w_{ji}^{h}(t+1) = \alpha \delta_{pj}^{h} x_{pi}$$

En ambos casos, para acelerar el proceso de aprendizaje se puede añadir un término *momento*.

PASO 6: El proceso se repite hasta que el término de error $E_p = \frac{1}{2}\sum_{k=1}^{M} \delta_{pk}^{2}$ resulta aceptablemente pequeños para cada uno de los patrones aprendidos.

I.5.2 CARACTERÍSTICAS DE LA RED BACKPROPAGATION

- El algoritmo busca el mínimo de la función error a partir de un conjunto de patrones de entrenamiento.
- El algoritmo precisa que la función de activación sea diferenciable (fácilmente).
- Los pesos se modifican hacia la dirección descendente de la función error.
- La red entrenada es capaz de generalizar, clasificando correctamente patrones ruidosos o incompletos.

I.5.3 VENTAJAS E INCOVENIENTES DE LA RED DE BACKPROPAGATION

La principal ventaja de la red Backpropagation es su capacidad genérica de mapeo de patrones. La red es capaz de aprender una gran variedad de relaciones de mapeo de patrones. No requiere un conocimiento matemático de la función que relaciona los ejemplos de mapeo para aprender. La flexibilidad de esta red es aumentada con la posibilidad de elegir número de capas, interconexiones, unidades procesadores, constante aprendizaje y representación de datos. Como resultado de estas características

la red Backpropagation es capaz de participar con éxito en una amplia gama de aplicaciones.

El mayor inconveniente es el tiempo de convergencia. Las aplicaciones reales pueden llegar a tener miles de ejemplos en el conjunto de entrenamiento y ello requiere días de tiempo de cálculo. Además la red Backpropagation es susceptible a fallar en el entrenamiento, es decir, la red puede que nunca llegue a converger.

Existe una variedad de técnicas desarrolladas para disminuir el tiempo de convergencia y evitar los mínimos locales. El término de "momentum" se utiliza para aumentar la velocidad del proceso de convergencia. Otra forma de mejorar la convergencia se basa en la variación del parámetro de aprendizaje η comenzando con valores altos y adquiriendo progresivamente valores más pequeños. Entre las técnicas utilizadas para evitar los mínimos locales destacan cambiar la red, cambiar el conjunto de entrenamiento y añadir ruido aleatorio a los pesos.

I.6 RED NEURONAL: HOPFIELD

Uno de los principales responsables del desarrollo que ha experimentado el campo de la computación neuronal ha sido J. Hopfield, quien construyó un modelo de red con el número suficiente de simplificaciones como para poder extraer analíticamente información sobre las características relevantes del sistema, conservando las ideas fundamentales de las redes construidas en el pasado y presentando una serie de funciones básicas de sistemas neuronales básicas.

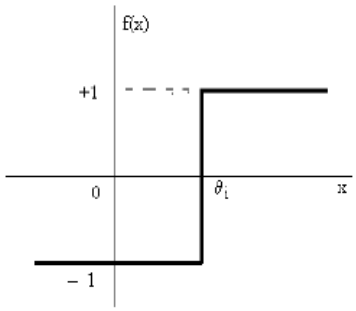
Con su aportación, Hopfield redescubrió el mundo caso olvidado de las redes autoasociativas, caracterizadas por una nueva arquitectura y un nuevo funcionamiento, a las que tuvo que añadir otro tipo de reglas de aprendizaje.

I.6.1 ARQUITECTURA DE LA RED DE HOPFIELD

El modelo de Hopfield consiste en una red monocapa con N neuronas cuyos valores de salida son binarios: 0/1 ó -1/+1. En la versión original del modelo(Hopfield Discreto) las funciones de activación de las neuronas eran de tipo escalón. Se trataba, por tanto, de una red discreta, con entradas y salidas binarias; sin embargo, posteriormente Hopfield desarrolló una versión continua con entradas y salidas analógicas utilizando neuronas con funciones de activación tipo sigmoidal (Hopfield Continuo).

Cada neurona de la red se encuentra conectada a todas las demás(conexiones laterales), pero no consigo misma(no existen conexiones autorecurrentes). Además, los pesos asociados a las conexiones entre pares de neuronas son simétricos. Esto significa que el peso de la conexión de una neurona i con otra j es de igual valor que el de la conexión de la neurona j con la i($w_{ij}=w_{ji}$).

La versión discreta de esta red fue ideada para trabajar con valores binarios -1 y +1 (aunque mediante un ajuste de los pesos pueden utilizarse en su lugar los valores 1 y 0). Por tanto, la función de activación de cada neurona(i) de la red(f(x)) es de tipo escalón.



$$f(x)=\begin{Bmatrix}+1\ \mathbf{x}>\boldsymbol{\theta}_i\\-1\ \mathbf{x}<\boldsymbol{\theta}_i\end{Bmatrix}$$

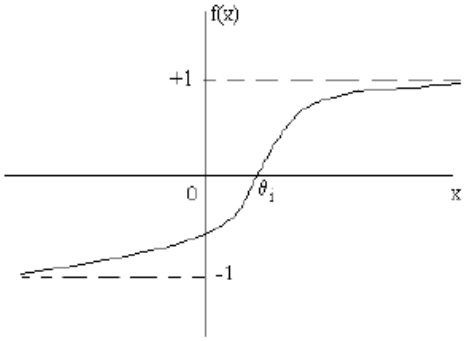
Cuando el valor de x coincide exactamente con Θ_i, la salida de la neurona i permanece son valor anterior. Θ_i es el umbral de disparo de la neurona i, que representa el desplazamiento de la función de transferencia a lo largo del eje de ordenadas(x). En el modelo de Hopfield discreto suele adoptarse un valor proporcional a la suma de los pesos de las conexiones de cada neurona con el resto:

$$\boldsymbol{\theta}\, i = k\sum_{j=1}^{N} w_{ij}$$

Si se trataba con los valores -1 y +1, suele considerarse el valor nulo para Θ_i. Si los valores binarios son 0 y 1, se toma un valor de ½ para k.

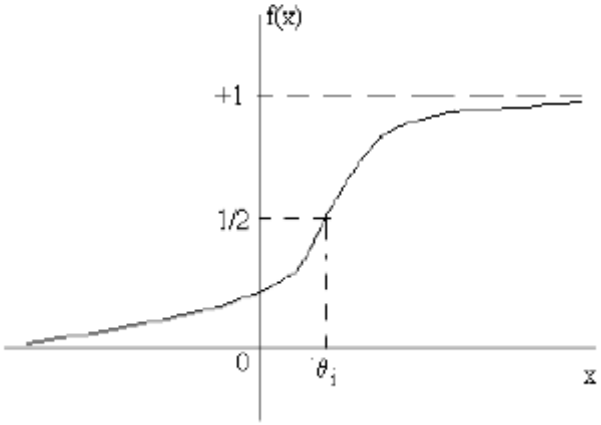
En el caso de las redes de Hopfield continuas, se trabaja con valores reales en los rangos [-1,+1] ó [0,1]. En ambos casos, la función de activación de las neuronas es de tipo sigmoidal. Si se trabaja con valores entre -1 y +1, la función que se utiliza es la tangente hiperbólica:

$$f(x-\theta i) = tgh(\alpha(x-\theta i)) = \frac{e^{\alpha(x-\theta i)} - e^{-\alpha(x-\theta i)}}{e^{\alpha(x-\theta i)} + e^{-\alpha(x-\theta i)}}$$



Si el rango es [0,1], se utiliza la misma función que para la red Backpropagation:

$$f(x-\theta i) = \frac{1}{1+e^{-\alpha(x-\theta i)}}$$



En este caso, α es un parámetro que determina la pendiente de la función sigmoidal.

I.6.2 FUNCIONAMIENTO DE LA RED DE HOPFIELD

Una de las características del modelo de Hopfield, es que se trata de una red autoasociativa. Así, varias formaciones(patrones) diferentes pueden ser almacenadas en la red, como si de una memoria se tratase, durante la etapa de aprendizaje. Posteriormente, si se presenta a la entrada alguna de las informaciones almacenadas, la red evoluciona hasta estabilizarse, ofreciendo entonces en la salida la información almacenada, que coincide con la presentada en la entrada. Si, por el contrario, la

información de entrada no coincide con ninguna de las almacenadas, por estar distorsionada, o incompleta, la red evoluciona generando como salida la más parecida.

La información que recibe esta red debe haber sido previamente codificada y representada en forma de vector(como una configuración binaria si la red es discreta y como conjunto de valores reales si es continua) con tantos componentes como neuronas(N) tenga la red. Esa información es aplicada directamente a la única capa de que consta la red, siendo recibida por las neuronas de dicha capa(cada neurona recibe una parte de la información, un elemento del vector que representa dicha información). Si SE considera en principios el caso de una neurona concreta de la red, esta neurona recibiría como entrada las salidas de cada una de las otras neuronas, valores que inicialmente coincidirán con los de entrada, multiplicada por los pesos de las conexiones correspondiente. La suma de todos estos valores constituirá el valor de entrada neta de la neurona, al que le será aplicada la función de transferencia, obteniéndose el valor de salida correspondiente, 0/1 ó -1/+1 si la red es discreta, y un número real en el intervalo [0,1] ó [-1,+1] si es continua.

La descripción anterior correspondía a un primer paso en el procesamiento realizado por la red. Este proceso continúa hasta que las salidas de las neuronas se estabilizan, lo cual ocurrirá cuando dejen de cambiar de valor. Entonces, el conjunto de estos(N) valores de salida de todas las neuronas constituye la información de salida que ha generado la red que se corresponderá con alguna de las informaciones durante la etapa de aprendizaje fueron almacenada en la misma.

Este funcionamiento puedes expresarse matemáticamente de la siguiente forma:

1º En el instante inicial(t=0) se aplica la información de entrada (valores e_1, e_2, … ,e_N).

$$s_i(t=0)=e_i \quad 1 \le i \le N$$

Inicialmente, la salida de las neuronas coincide con la información aplicada a la entrada.

2º la red realiza iteraciones hasta alcanzar la convergencia(hasta que $s_i(t+1)$ sea igual a $s_i(t)$).

$$s_i(t+1) = f(\sum_{j=1}^{N} w_{ij} s_j(t) - \theta_i) \qquad 1 \le i \le N$$

Donde f es la función de transferencia(activación) de las neuronas de la red. En el caso del modelo discreto, si se trabaja con valores binarios -1 y +1, la salida se obtendría según la función escalón:

$$si(t+1) = \begin{cases} +1 & \sum_{j-1}^{N} w_{ij} s_j(t) > \theta i \\ s_i(t) & \sum_{j-1}^{N} w_{ij} s_j(t) = \theta i \\ -1 & \sum_{j-1}^{N} w_{ij} s_j(t) < \theta i \end{cases}$$

El proceso se repite hasta que las salidas de las neuronas permanecen sin cambios durante algunas iteraciones. En ese instante, la salida(S_1,S_2,…, S_N) representa la información almacenada por la red que más parece a la información presentada en la entrada(e_1,e_2, … ,e_N).

El funcionamiento descrito corresponde a una red de Hopfield discreta clásica, ya que trabaja con informaciones binarias. Sin embargo, existe una variación del modelo desarrollado también por Hopfield que pretende parecerse un poco más al funcionamiento de las neuronas reales, se trata de la red de Hopfield continua, con funciones de activación de las neuronas de tipo sigmoidal, que ofrece más posibilidades que la anterior, ya que permite almacenar patrones formados por valores reales(por ejemplo, imágenes con color o en blanco y negro con diferentes tonalidades de grises) y además facilita la resolución de determinados problemas generales de optimización.

Tanto en este caso como en el de la red discreta, se pueden distinguir diferentes versiones del modelo en función de la forma temporal en que se leva a cabo la generación o actualización de las salidas de las neuronas de la red. Si esta actualización se realiza de forma simultanea en todas las neuronas, se trata de una red de Hopfield con funcionamiento paralelo o síncrono, ya que supone que todas las neuronas son capaces de operar sincronizadas y que, por tanto, la salida es generada al mismo tiempo por todas aquellas en cada iteración, de tal forma que en la próxima iteración(t+1) todas van a utilizar como entradas las salidas generadas por las otras en el instante anterior(t). Si, por el contrario, las neuronas trabajan de forma secuencial, actualizándose solo la salida de una neurona en cada iteración, se tratará de una red Hopfield con funcionamiento

secuencial o asíncrono. En este caso, ocurre que la salida a la que converge la red puede ser diferente en función del orden de activación de las neuronas.

También existe la posibilidad, sólo utilizada en las redes Hopfield continuas, que trabajan con valores no binarios, de que la generación de la salida de todas las neuronas se realice de forma simultánea(síncrona) y continua en el tiempo. En este caso, el funcionamiento de la red tendría que ser representado en forma de una ecuación diferencial:

$$\tau_i \frac{\partial s_i}{\partial t} = f\left(\sum_{j=1}^{N} w_{ij} s_j - \theta_i\right) - s_i$$

Donde la f en la función de activación de las neuronas, que será de tipo sigmoidal y γ_i es un parámetro, denominado tasa de retardo, que pretende representar una característica biológica de las neuronas, como es el retraso que se produce en la recepción por parte de una neurona i de los valores de las salidas generadas por las otras salidas a las que está conectada.

Si se pretende simular una red Hopfield continua como la anterior, puede hacerse resolviendo la ecuación diferencial utilizando métodos numéricos, aunque también, y sobre todo, puede implementarse físicamente de forma directa mediante un circuito analógico que tenga un comportamiento gobernado por una ecuación similar.

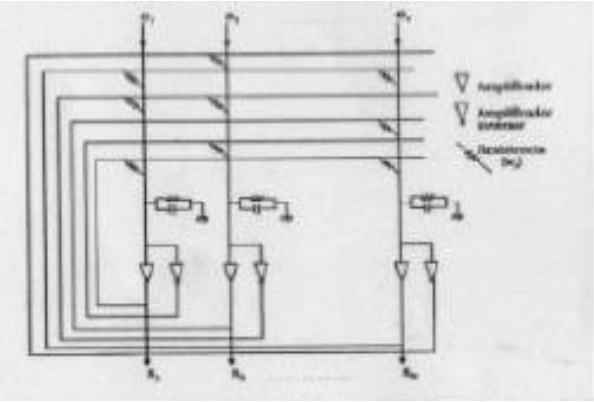


Figura 12: Red Neuronal de Hopfield

En estos circuitos, los pesos entre neuronas se implementan como resistencia, las funciones de activación mediante amplificadores operacionales, y la tasa de retardo de cada neurona se consigue colocando en paralelo una resistencia y un condensador a la entrada del amplificador correspondiente.

I.6.3 REGLA DE APRENDIZAJE DE LA RED DE HOPFIELD

La red de Hopfield tiene un mecanismo de aprendizaje OFF LINE. Por tanto, existe una etapa de aprendizaje y otra de funcionamiento de la red. En la etapa de aprendizaje se fijan los valores de los pesos en función de las informaciones que se pretende que memorice o almacene la red.

Esta red utiliza un aprendizaje no supervisado de tipo hebbiano, de tal forma que el peso de una conexión entre una neurona i y otra j se obtiene mediante el producto de los componentes i – ésimo y j – ésimo del vector que representa la información o patrón que debe almacenar. Si el número de patrones a aprender es M, el valor definitivo de cada uno de los pesos se obtiene mediante la suma de los M productos obtenidos por el procedimiento anterior, un producto por información a almacenar.

En el caso de la red Hopfield discreta, que trabaja con valores -1 y +1, este algoritmo de aprendizaje puede expresarse de la siguiente forma:

$$w_{ij} = \begin{cases} \sum_{k=1}^{M} e_i^{(k)} e_{ij}^{(k)} & 1 \le i, j \le N ; \mathrm{i} \neq \mathrm{j} \\ 0 & 1 \le i, j \le N ; \mathrm{i} = \mathrm{j} \end{cases}$$

wij: Peso asociado a la conexión entre la neurona j y la neurona i, que coincide con wji

$e_j^{(k)}$: Valor de la componente i –ésima del vector correspondiente a la información k – ésima que debe aprender la red.

N: Número de neuronas de la red, y por tanto, tamaño de los vectores de aprendizaje.

M: Número de informaciones que debe aprender la red

Si la red trabajase con valores discretos 0/1 en lugar de -1/+1, entonces los pesos se calculan según la expresión:

$$w_{ij} = \begin{cases} \sum_{k=1}^{M} (2e_i^{(k)} - 1)(2e_{ij}^{(k)} - 1) & 1 \le i, j \le N ; \mathrm{i} \neq \mathrm{j} \\ 0 & 1 \le i, j \le N ; \mathrm{i} = \mathrm{j} \end{cases}$$

El algoritmo de aprendizaje también suele expresar utilizando una notación matricial. En tal caso se podría considerar una matriz W de dimensiones NxN que representase todos los pesos de la red:

$$W = \begin{bmatrix} w_{11}\ w_{21}\ w_{31} \ldots\ldots w_{N1} \\ w_{12}\ w_{22}\ w_{32} \ldots\ldots w_{N2} \\ \ldots\ldots \\ w_{1N}\ w_{2N}\ w_{3N} \ldots\ldots w_{NN} \end{bmatrix}$$

Esta matriz es simétrica, al cumplirse que $w_{ij}=w_{ji}$, y tiene una diagonal principal con valores nulos debido a la no existencia de conexiones autorecurrentes($w_{ij}=0$).

También se tendría el conjunto de los M vectores que representa las informaciones que ha de aprender la red:

$$E_1 = [e_1^{(1)}, e_2^{(1)}, \ldots, e_N^{(1)}]$$
$$E_2 = [e_1^{(2)}, e_2^{(2)}, \ldots, e_N^{(2)}]$$
$$\ldots\ldots\ldots\ldots$$
$$E_M = [e_1^{(M)}, e_2^{(M)}, \ldots, e_N^{(M)}]$$

Utilizando esta notación, el aprendizaje consistiría en la creación de la matriz de pesos W a partir de los M vectores o informaciones de entrada(E_1, …, E_M) que se enseña a la red, matemáticamente se expresaría:

$$W = \sum_{k=1}^{M} [E_k^T E_k - I]$$

Donde la matriz E^T_k es la transpuesta de la matriz E_k, e I es la matriz identidad de dimensiones NxN que anula los pesos de las conexiones autorecurrentes(w_{ij}).

I.6.4 LA FUNCIÓN ENERGÍA DE LA RED DE HOPFIELD

Como se mencionó, el aprendizaje de la red Hopfield es de tipo hebbiano. Esta función asegura la estabilidad de la red; es decir, la convergencia hacia una respuesta estable cuando se presenta una información de entrada.

Muchas de las investigaciones acerca de la estabilidad de las redes se basan en el establecimiento de una función, denominada función energía de la red, para representar los posibles estados (puntos de equilibrio) de la red. De hecho, una de las causas por la que se considera a Hopfield responsable de impulsar aplicando modelos matemáticos como éste, lo cual constituyó la base de posteriores trabajos sobre redes neuronales.

La función de energía de una red Hopfield discreta tiene la siguiente forma:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i\neq j}}^{N} w_{ij} s_i s_j + \sum_{i=1}^{N} \theta_i s_i$$

Siendo:

W_{ij}: Peso de la conexión entre la neurona i y j.

S_i: Valor de salida de la neurona i.

S_j: Valor de salida de la neurona j.

Θ_i: Umbral de la función de activación de la neurona i.

Esta expresión guarda una profunda similitud formal con la energía mecánica clásica. Trata de representar la evolución del sistema, considerando cada configurando(vector) de las salidas de las neuronas de la red como puntos en un espacio de dimensión N y relacionando el estado de la red en cada momento con un punto en ese espacio. La función energía puede imaginarse entonces como una superficie que representa determinados valores mínimos, algo semejante a un paisaje montañoso donde los mínimos serían los valles.

Cuando en la red se han almacenado M informaciones o patrones, los posibles estados estables, de la red serán también M (durante su funcionamiento podrá responder ante una entrada con una salida que represente alguno de esos M patrones registrados). Estos M estados corresponden precisamente a los mínimos de la función energía. Cuando se presenta a la entrada de la red una nueva información, esta evoluciona hasta alcanzar un mínimo de la función energía, generando salida estable.

Cuando la red Hopfield se utiliza como memoria asociativa, el objetivo es conseguir que los patrones que debe memorizar se sitúen en los mínimos de la función y, consecuentemente, sean los estados estacionarios(estables) de la red. Puede demostrarse que esto se consigue si se verifica que los pesos de las conexiones autorecurrentes son nulos (w_{ij}=0) y el resto cumple la regla Hebb $(w_{ij} = \sum s_s s_s)$, de ahí que Hopfield eligiera dicho aprendizaje.

En cuanto a las redes de Hopfield continuas, su autor considera que una posible expresión de su función de energía es la siguiente:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i\neq j}}^{N} w_{ij}s_i s_j + \sum_{i=1}^{N}\int_{0}^{s_i} f^{-1}(s)ds$$

Donde f^{-1}es la inversa de la función de activación sigmoidal(f) de una neurona. Si la función f fuese, por ejemplo, $f(x)=1/(1+e^{-\alpha(x-\theta i)})$, entonces la inversa sería $f^{-1}(s)=-(\ ½\ \ln(-1+ 1/s))+\Theta_i$, con lo que la función energía quedaría de la siguiente forma:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i\neq j}}^{N} w_{ij}s_i s_j + \sum_{i=1}^{N}\theta_i - \frac{1}{\alpha}\sum_{i=1}^{N}\int_{0}^{s_i} \ln(\frac{1}{s}-1)ds$$

Se puede observa que coincidiría con la función energía de Hopfield discreta si no existiera el ultimo término, cuyo valor depende del parámetro α de la función de activación de las neuronas, que representa la ganancia o la pendiente de esta función sigmoidal. Por ello, habrá que tener especial cuidado en la elección de su valor, ya que influye directamente en los mínimos de la función energía. Si α tiene un valor infinito(función de activación tipo escalón), este término se hace cero y la red se convierte en una red Hopfield discreta. Si el valor α es finito, pero muy grande (α>1), el término puede despreciarse y los mínimos de la función de energía siguen siendo los mismos. Pero si α tiene valores pequeños, disminuye el número de mínimos de esta función y, como consecuencia, el número de posibles estados estables en la red, reduciéndose a un solo mínimo cuando α=0.

I.6.5 CARACTERISTICAS DE LA RED DE HOPFIELD

- Solo tiene una capa.
- Los valores pueden ser binarios o análogos, dependiendo de la función que se utilice a la salida de las neuronas.
- Cada neurona de la red se encuentra conectada a todas las demás neuronas de la red.
- En algunos casos existen conexiones de una neurona hacia sí misma.
- Los pesos asociados a las conexiones entre pares de neuronas son simétricos.
- Aprendizaje fuera de línea(offline).
- Es un tipo de red asociativa.
- La cantidad de neuronas en la red depende del número de entradas que se requiera.

- El número de entradas es igual al número de salidas.
- Es una red discreta con entradas y salidas binarias 0/1 ó -1/1.
- Esta red consiste en un conjunto de N elementos de procesado interconectadas que actualizan sus valores de activación de forma asíncrona e independiente del resto de los elementos de procesado. Todos los elementos son a la vez de entrada y salida.

I.6.6 LIMITACIONES DEL MODELO DE HOPFIELD

En una red de Hopfield, el número de informaciones que puede ser aprendido(almacenado) está severamente limitado. Si se almacenan demasiadas informaciones, durante su funcionamiento la red puede converger a valores de salida diferentes de los aprendidos durante la etapa de entrenamiento, con lo que la tarea de asociación entre la información presentada y alguna de las almacenadas se realiza incorrectamente, pues se está asociando dicha información de entrada a otra desconocida.

Esta situación no se produce nunca si el número de información almacenada es menor o igual que N/4lnN, siendo N el número de neuronas de la red. Sin embargo, este limite puede suavizarse si se permite la posibilidad de un mínimo error en la recuperación de las informaciones almacenadas, suficientemente pequeño para poder identificar dicha información sin ningún problema; en tal caso, el número de informaciones que se pueden almacenar debe ser menor que el 13,8% del número de neuronas de la red(13.8N), por tanto, la capacidad o cantidad de informaciones que pueden almacenar una red Hopfield de N neuronas es:

$$Capacidad = \begin{cases} 0.138.N & \text{para una recuperación suficientemente buena} \\ \dfrac{N}{4 \ln N} & \text{para una recuperación perfecta} \end{cases}$$

Esta es una seria limitación de la red Hopfield, puesto que para almacenar unas pocas informaciones se precisará una gran cantidad de neuronas, y por tanto, un número muy elevado de conexiones. Por ejemplo, tomando la limitación del 13,8%, en una red de 100 neuronas, que tendría 100x99=9900 conexiones, sólo se podrían almacenar 13 informaciones.

Una segunda limitación del modelo, es que a pesar de cumplirse el requisito anterior, no siempre se puede garantizar que la red realice una asociación correcta entre una neurona

y una de las informaciones almacenadas, si estas últimas no son suficientemente diferentes entre sí, es decir, si no son ortogonales, puede ocurrir que para cada una de ellas no represente un mínimo de la función energía, con la probabilidad de que se generan salidas diferentes a todas ellas. También puede ocurrir que ante una entrada que coincida con una de las informaciones aprendidas, la red converja hacia una salida correspondiente a otra de las informaciones almacenadas que fuese muy parecida.

I.6.7 INTRODUCCIÓN RED BIDIRECTIONAL ASSOCIATIVE MEMORY

El modelo de la red B.A.M (Bidirectional Associative Memory) fue desarrollada por Kosko (1987) aun cuando presenta varios aspectos inspirados en el trabajo de Grossberg(1982).

También se puede considerar esta red como la generalización del modelo de Hopfield en redes heteroasociativas. La red BAM es heteroasociativa en cuanto que acepta un vector de entrada en con conjunto de neuronas y produce otro vector relacionado, un vector de salida en otro en otro conjunto de neuronas. Sin embargo la red Hopfield debido a su única capa de neuronas requiere que el vector de salida aparezca en las mismas neuronas en las que se aplica el vector de entrada, de aquí el carácter de red o memoria autoasociativa.

I.6.8 ARQUITECTURA DE LA RED BAM

La red tiene dos capas centrales de neuronas totalmente interconectadas además de las capas buffer de entrada y salida. En la siguiente figura 16 muestra la estructura de una red BAM siguiendo la red de Hopfield(no se muestran las capas de entrada y salida).

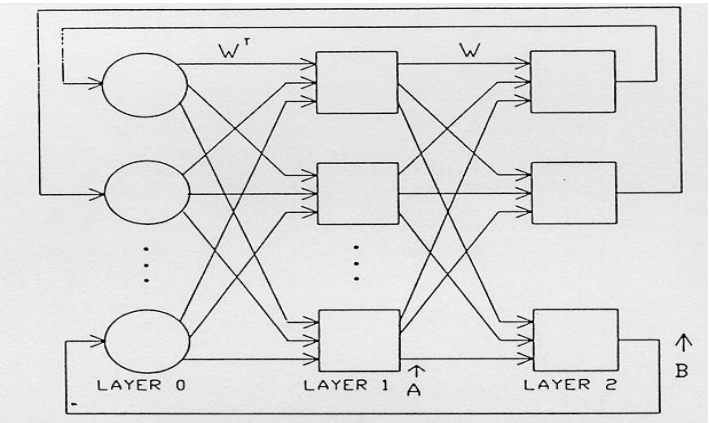


Figura 13: Estructura simplificada de la red BAM

La red se diseña para almacenar parejas asociadas de vectores. Los pesos de las conexiones entre las capas centrales almacenan la información asociativa. Si la primera capa tiene N neuronas y la segunda capa tiene M neuronas los pesos de estas conexiones se almacenan en una matriz W de orden NxM. Esta arquitectura si se aplica un vector de entrada A a los pesos W(es decir a la salida de la primera capa) se produce nuevas salidas para el vector A. Este proceso se repite hasta que la red alcanza un punto estable en el que A y B no cambian.

Recuperación de las Asociaciones Almacenadas

Una vez definida la arquitectura de la red, esta debe ser entrenada para reconocer o recordar una serie de memorias o vectores. El conjunto de entrenamiento está constituido por parejas de vectores A y B. El entrenamiento en el cálculo de la matriz de pesos W según la ecuación

$W = \Sigma i\ Ai^t\ Bi$

En este cálculo lo realizan las redes BAM modificando sus pesos a lo largo de la fase de entrenamiento utilizando la regla Hebb.

Las asociaciones o memorias de la red se almacenan en las matrices W y $W^{t.}$ Para recuperar una asociación basta con presentar parcial o totalmente el vector A en la salida de la primera capa.

La red a través de W produce un vector de salida B en la capa segunda. Este vector opera sobre la matriz transpuesta W^T produciendo una réplica próxima al vector de entrada A en la salida de la capa primera. Cada paso en el lazo de la red produce que los vectores de salida de ambas capas sean cada vez más próxima a la memoria o asociación almacenada. Este proceso se lleva a cabo hasta que se alcance un punto estable llamado resonancia en el que los vectores pasan hacia adelante y hacia atrás reforzando las salidas sin modificaciones.

La relación estrechada entre la BAM y la red de Hopfield se manifiesta en el caso de que la matriz sea cuadrada y simétrica($W=W^T$). Para este caso si las capas primera y segunda tienen el mismo conjunto de neuronas la red BAM queda reducida a la red Autoasociativa de Hopfield.

I.7 SISTEMA DE PRUEBA: CLASIFICACIÓN DE LETRAS

A continuación se llevará a cabo el diseño e implementación en un sistema capaz de reconocer caracteres del alfabeto español.

En un principio se puede decir que una letra es considerada como un modelo ideal de tipo de letra pero con ruido en las mismas letras. Ya que al final de todas las formas se acercan a un modelo ideal que solo que con pequeñas variaciones del modelo original. Partiendo esta premisa se continuará con el proceso de entrenamiento de las redes neuronales artificiales: Perceptrón Multicapa, Madaline, Backpropagation y Hopfield tomando en cuenta que existen por lo general cuatro pasos en este proceso que son:

- Ensamblar el paquete de datos de entrenamiento.
- Crear y definir a la red.
- Entrenar a la red.
- Simular la respuesta de la red con nuevas entradas.

El paquete de datos de entrenamiento con que contamos es un conjunto de 10 vectores binarios que presentan cada una de las 10 letras del alfabeto español en su modelo ideal. Es decir, el modelo ideal con el que se entrenará a la red. En la figura 17 observamos el modelo de la letra A tomando en cuenta que cada letra se represente en un rectángulo de 5 bit de ancho y 7 bits de alto. Todas las letras serán representadas de esta forma por lo que el conjunto total de datos estará definido por una matriz binaria de 10x35 representando a las 10 letras del alfabeto español en 35 bits.

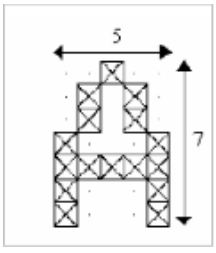


Figura 14: Modelo de la letra A en 35 bits

Esta se encuentra representada por el siguiente vector de 35 bits. [00100 01010 01010 10001 11111 10001 10001], corresponde a un vector de entrada para el entrenamiento de nuestra red neuronal artificial.

A continuación se conocerá como se define y entrena una red neuronal artificial para obtener la mejor solución para este problema de reconocimiento de caracteres.

I.8 DEFINICIÓN Y ENTRENAMIENTO DE LA RED NEURONAL ARTIFICIAL

En esta práctica del proyecto se utilizarán las herramientas que proporcionan los componentes GUI de Netbeans,GUI Visual Basic 6 y por último el GUI de Microsoft Visual Studio. Este paquete de herramientas nos permite definir, entrenar y simular cualquier tipo de red neuronal artificial y es esto que a lo largo del proyecto será utilizado extensivamente para obtener el modelo neuronal más adecuado para solucionar el problema en cuestión.

Para este caso práctico especifico, Java provee un conjunto de datos de entrenamiento que contiene las matrices de 10 caracteres cada uno representando por su vector de 35 valores binarios.

Además esto nos proporciona los destinos necesarios para identificar a que tipo de letra corresponde cada vector de 35 bits. Es decir, que se tendrá una matriz cuya única finalidad será representar 10 salidas posibles que nuestra red neuronal deberá tener.

En este momento se cuenta con 10 entradas y los 26 correspondientes destinos de salida que está listo para entrenar a la red artificial. En cuanto a Visual Basic y C# se definieron las redes neuronales artificiales, la cual será objeto de entrenamiento, por medio del siguiente código:

Visual Basic:

```
Private Sub boton_Click()
cd1.Filter = ("*.txt |*.txt|")
cd1.ShowOpen
texto.LoadFile (patrones.FileName)
End Sub
```

C#:

```
private void button1_Click(object sender, EventArgs e)
{

DialogoTxt.Filter = "Archivos txt|*.txt";
DialogoTxt.FileName = "Seleccione un archivo de Texto";
DialogoTxt.Title = "Lector de Archivo de Texto";

DialogoTxt.InitialDirectory = "C:/";

if (DialogoTxt.ShowDialog() == DialogResult.OK)
{
this.txtdireccion.Text = DialogoTxt.FileName;
}
```

```
            System.IO.StreamReader          sr          =          new
System.IO.StreamReader(@txtdireccion.Text,
System.Text.Encoding.Default);
            string texto;
texto = sr.ReadToEnd();
sr.Close();
txtcontenido.Text = texto;
}
```

Donde definimos que se creará las redes neuronales artificiales (Perceptrón Multicapa, Madaline, Backpropagation y Hopfield).

Luego de entrenar a las redes neuronales artificiales nos devuelve una red entrenada con valores de 35x1 por medio del algoritmo de entrenamiento seleccionado para cada red neuronal artificial.

Al momento que el error alcanza el valor mínimo esperado, se detiene el entrenamiento y los pesos de la red quedan ajustados para obtener este error mínimo cuando se le presenten los datos de entrenamiento.

La importancia de las redes neuronales artificiales no se encuentra únicamente en aproximar los datos con los que ha sido entrenada, sino generar este error mínimo para los datos de la misma naturaleza que nunca haya visto la red neuronal artificial, o a los que nunca haya sido expuesta, de manera que la red neuronal puede efectuar el conocimiento de caracteres que previamente tiene el modelo ideal de datos con el que fue entrenada.

El interés de esta etapa práctica es que una vez que haya sido entrenada con los datos del modelo ideal de las letras, la red pueda a su vez identificar las letras que se acerquen al modelo ideal pero que sean exactamente este modelo. Para esto la red neuronal que entrenamos ha sido probada con letras que tengan ruido desde 5% hasta 50%. Y así se ha evaluado su funcionamiento de la red neuronal artificial.

I.9 PRUEBAS EXPERIMENTALES DE LAS REDES NEURONALES ARTIFICIALES IMPLEMENTADAS EN EL LENGUAJE JAVA

Las redes neuronales artificiales: Perceptrón, Backpropagation, Madaline y Hopfield están programadas en el lenguaje de programación Java

I.9.1 SEUDOCÓDIGO DE LA RED NEURONAL ARTIFICIAL PERCEPTRÓN MULTICAPA

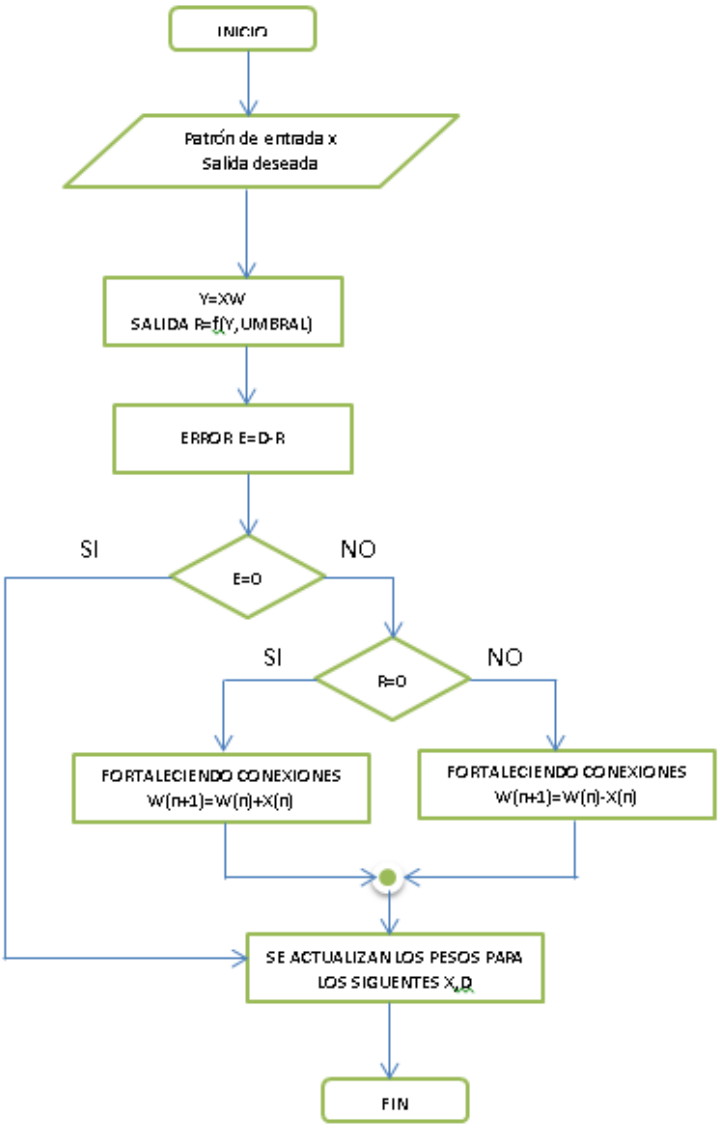


Figura 15**:** ***Diagrama de flujo del aprendizaje de la Red Neuronal Perceptrón Multicapa***

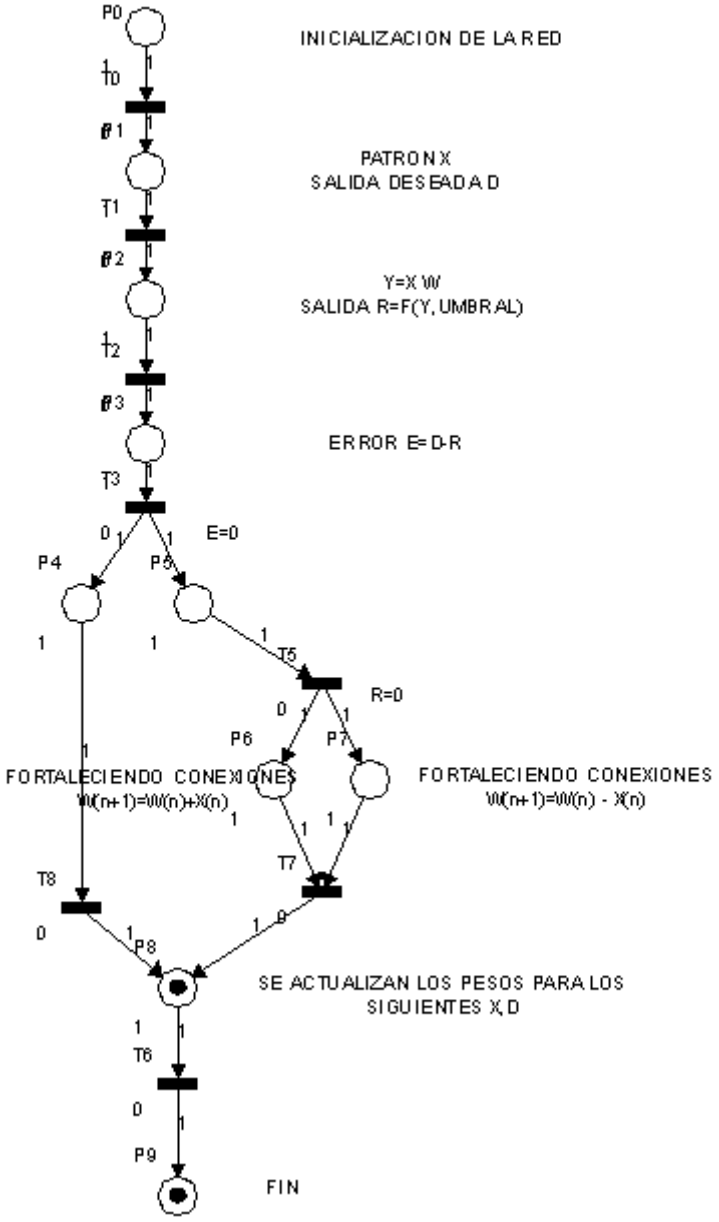


Figura 16: Diagrama de Petri: Fase de Aprendizaje de la Red Neuronal Perceptrón Multicapa

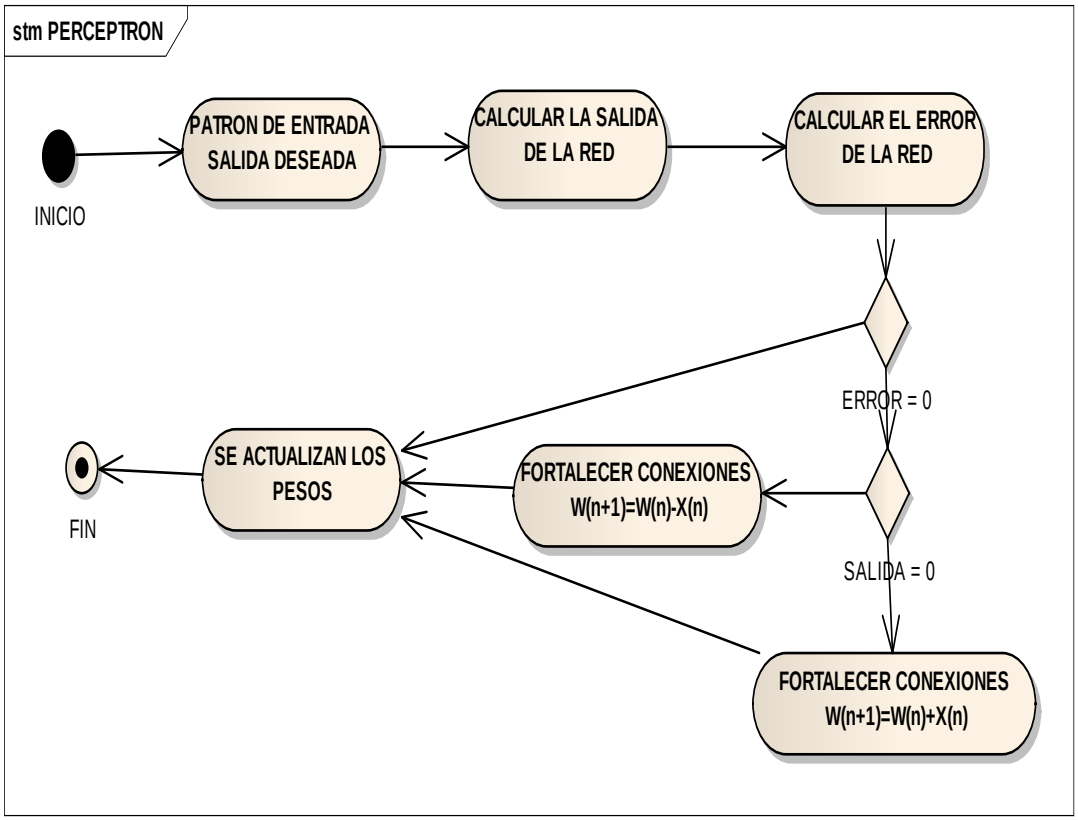
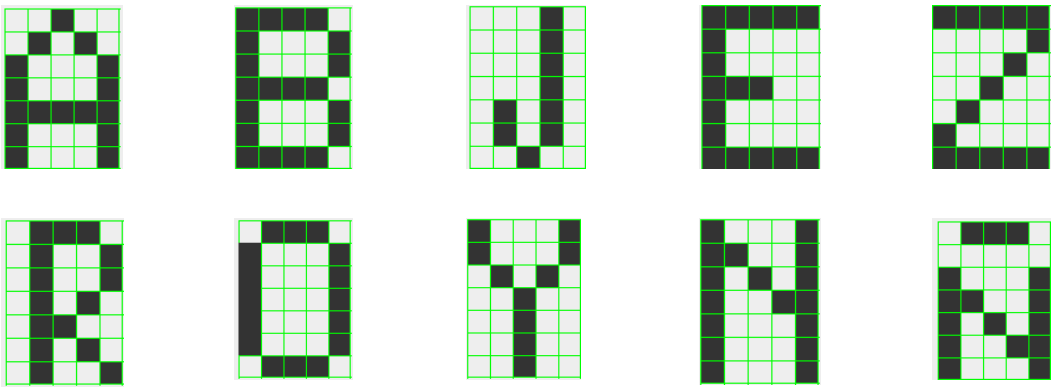


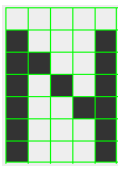
Figura 17: Diagrama De Estado: Fase de Aprendizaje de la Red Neuronal Perceptrón Multicapa

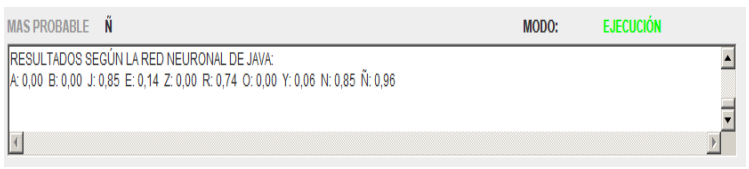
I.9.2 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL PERCEPTRÓN MULTICAPA PARA EL RECONOCIMIENTO DE CARACTERES

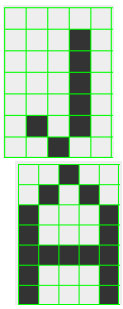
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA

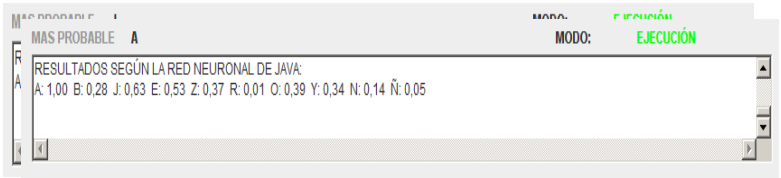


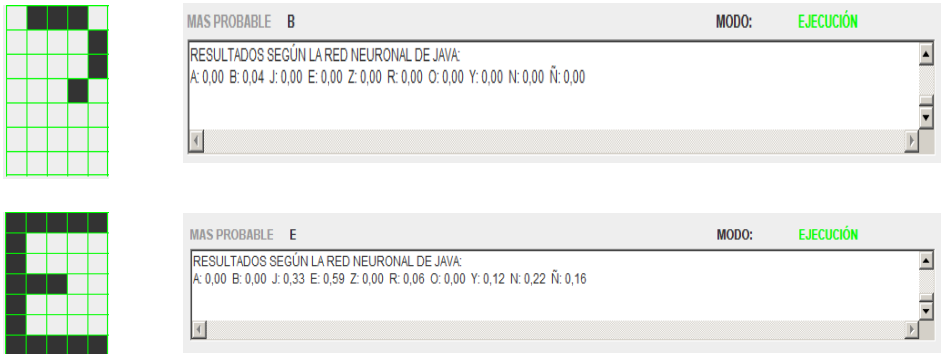
RESULTADOS DE LA PRIMERA FASE DE RECONOCIMIENTO DE CARACTERES











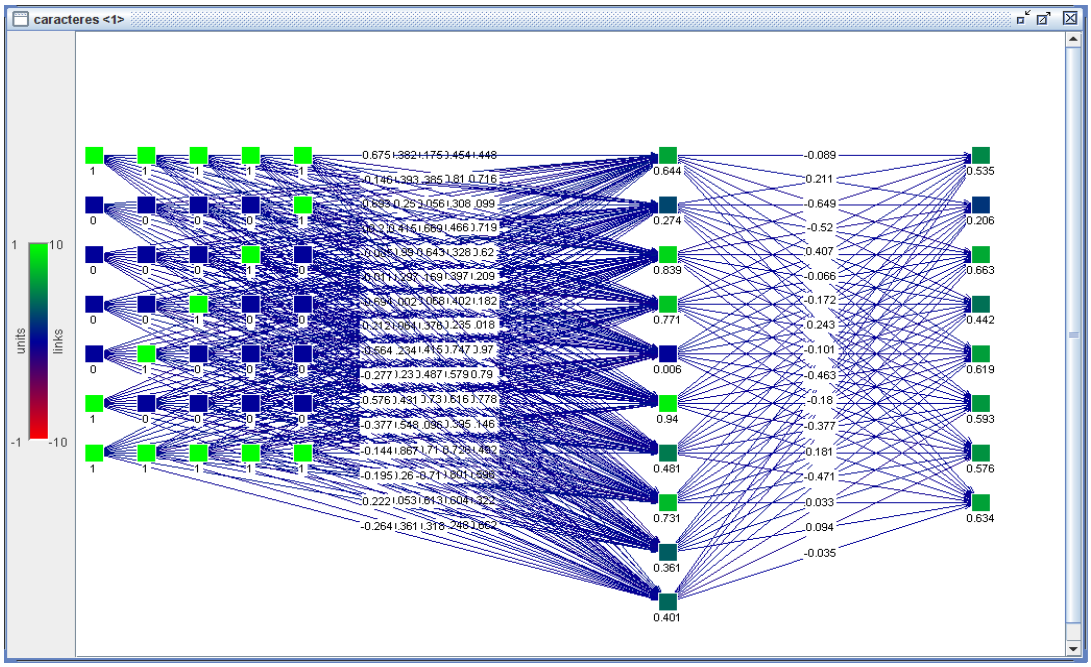
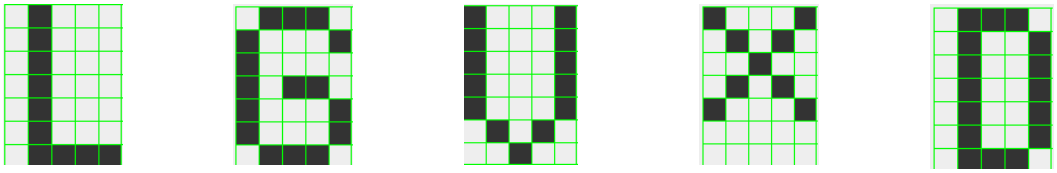
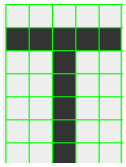
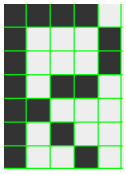
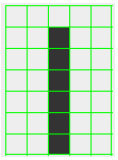
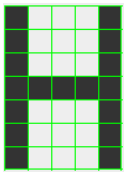
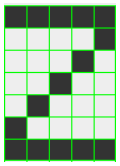


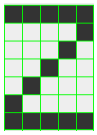
Figura18: Simulación y estructura de la red neuronal Perceptrón Multicapa en primera fase de prueba

2º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRÓN MULTICAPA



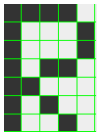






RESULTADOS DE LA SEGUNDA FASE DE RECONOCIMIENTO DE CARACTERES



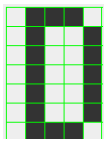
MAS PROBABLE Z MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Z: 0,94 H: 0,53 I: 0,92 R: 0,76 T: 0,90 D: 0,87 X: 0,72 V: 0,92 G: 0,80 L: 0,88



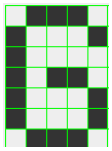
MAS PROBABLE R MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Z: 0,03 H: 0,03 I: 0,02 R: 0,06 T: 0,01 D: 0,00 X: 0,00 V: 0,00 G: 0,05 L: 0,01



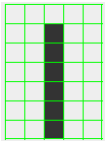
MAS PROBABLE D MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Z: 0,27 H: 0,19 I: 0,09 R: 0,71 T: 0,03 D: 0,72 X: 0,07 V: 0,30 G: 0,41 L: 0,57



MAS PROBABLE G MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Z: 0,19 H: 0,25 I: 0,16 R: 0,36 T: 0,09 D: 0,17 X: 0,06 V: 0,14 G: 0,53 L: 0,35



MAS PROBABLE I MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Z: 0,42 H: 0,51 I: 0,62 R: 0,49 T: 0,45 D: 0,09 X: 0,35 V: 0,12 G: 0,44 L: 0,18

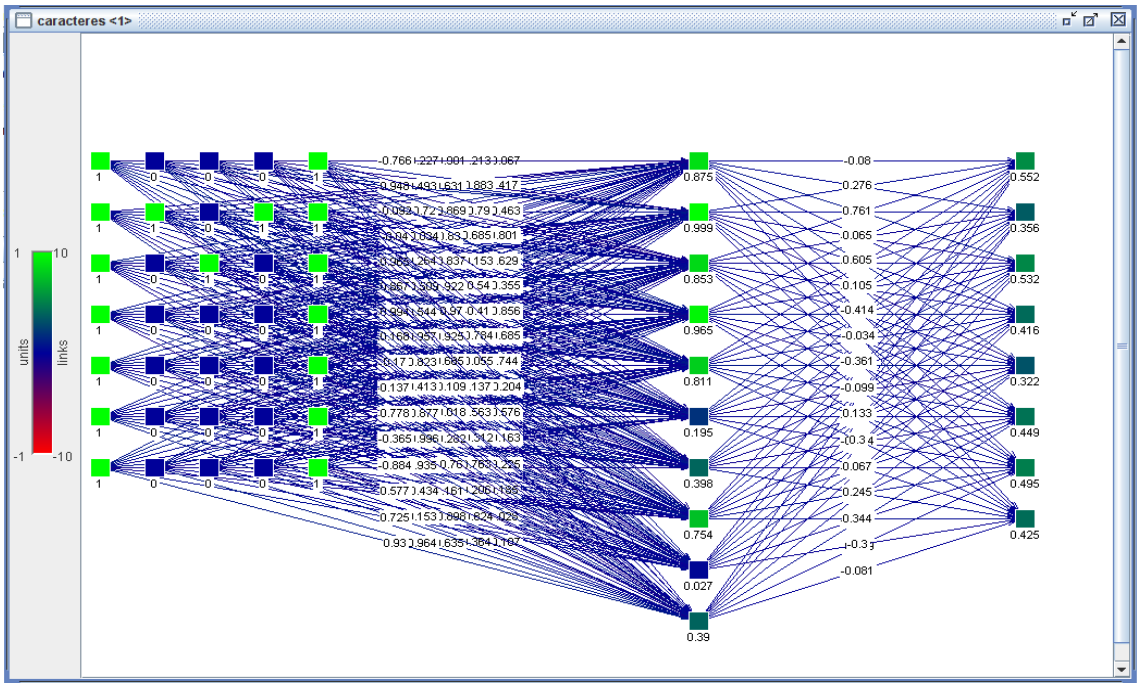
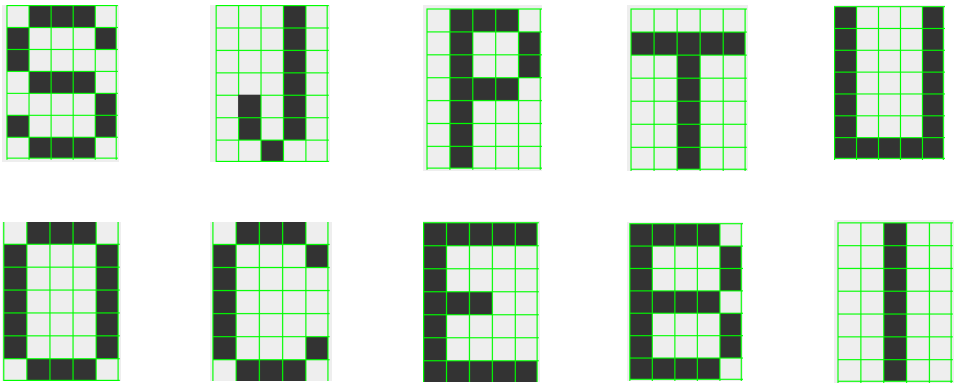
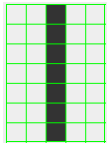


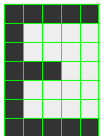
Figura 19: Simulación y estructura de la red neuronal Perceptrón Multicapa en la segunda fase de prueba

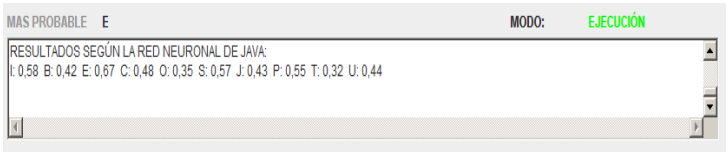
3º PRUEBA DE RED NEURONAL ARTIFICIAL PERCEPTRÓN MULTICAPA

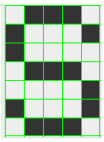


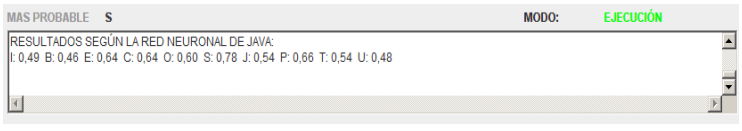
RESULTADOS DE LA TERCERA FASE DE RECONOCIMIENTO DE CARACTERES

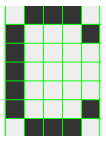


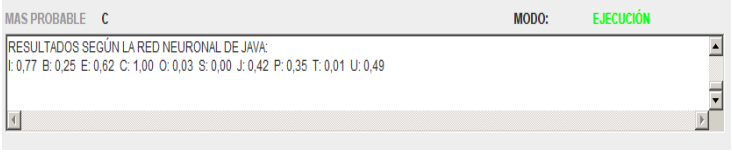


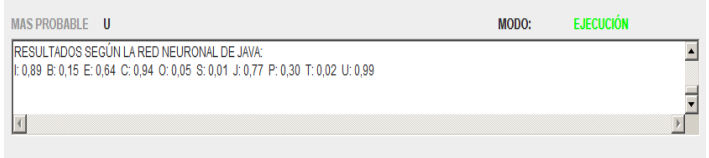


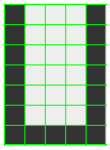












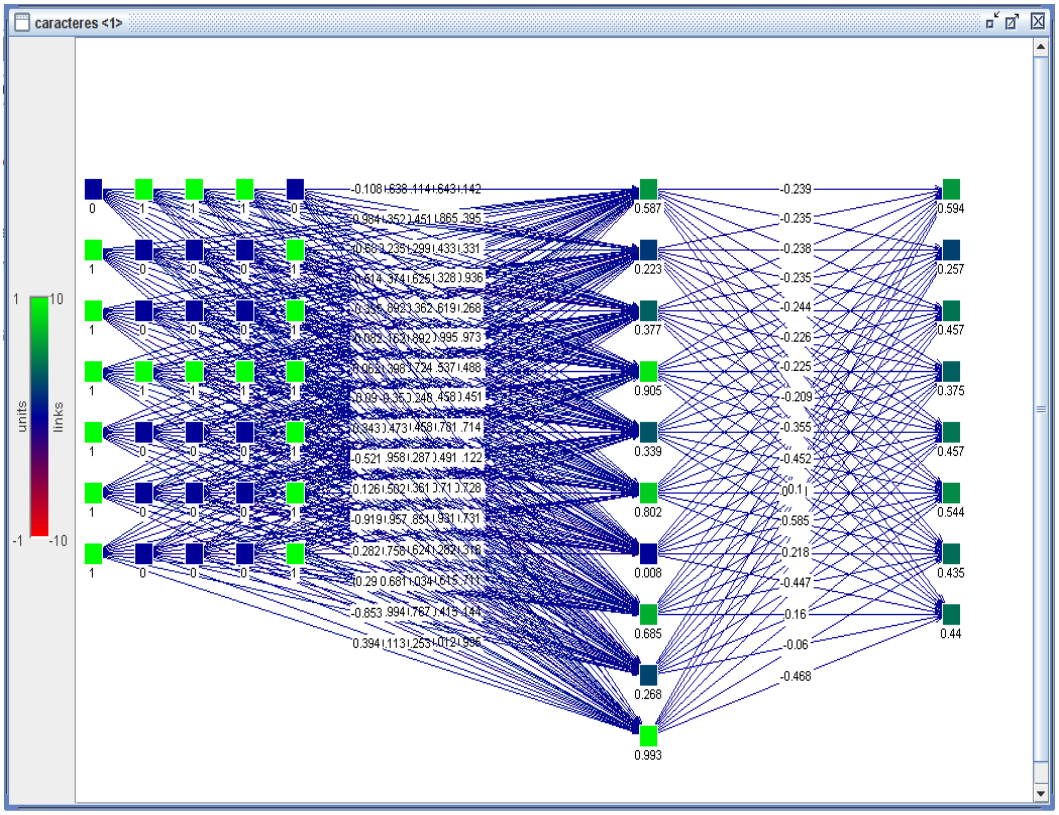


Figura 20: Simulación y estructura de la red neuronal Perceptrón Multicapa en la tercera fase de prueba

I.9.3 SEUDOCÓDIGO DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION

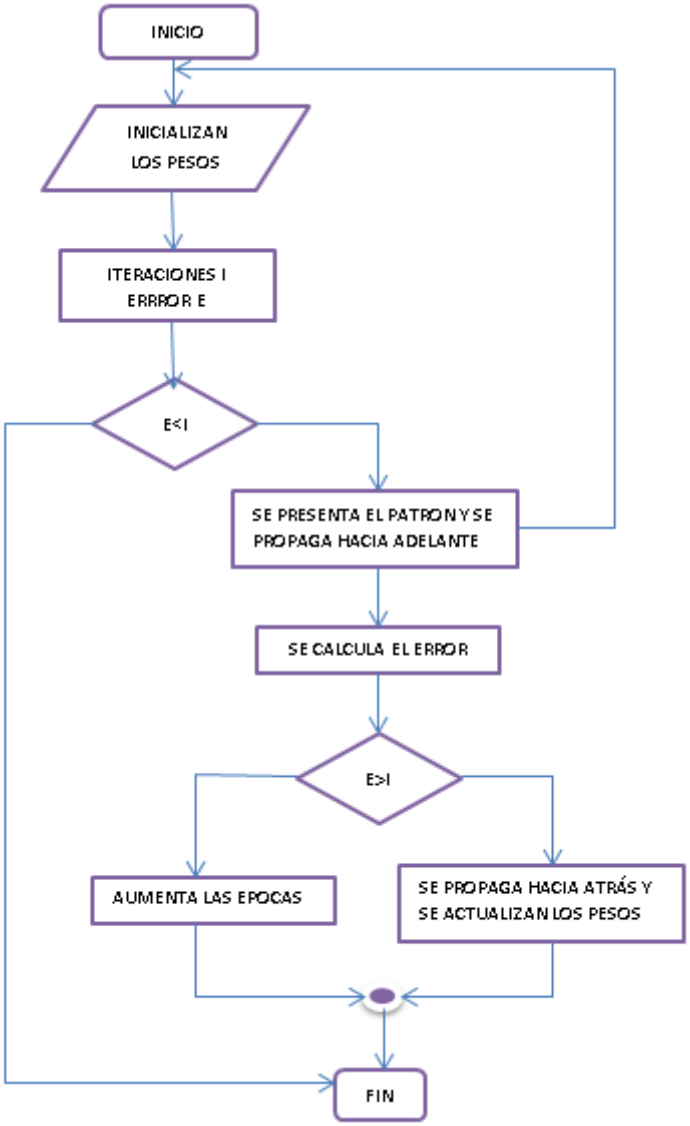


Figura 21: Seudocódigo de la red neuronal artificial Backpropagation

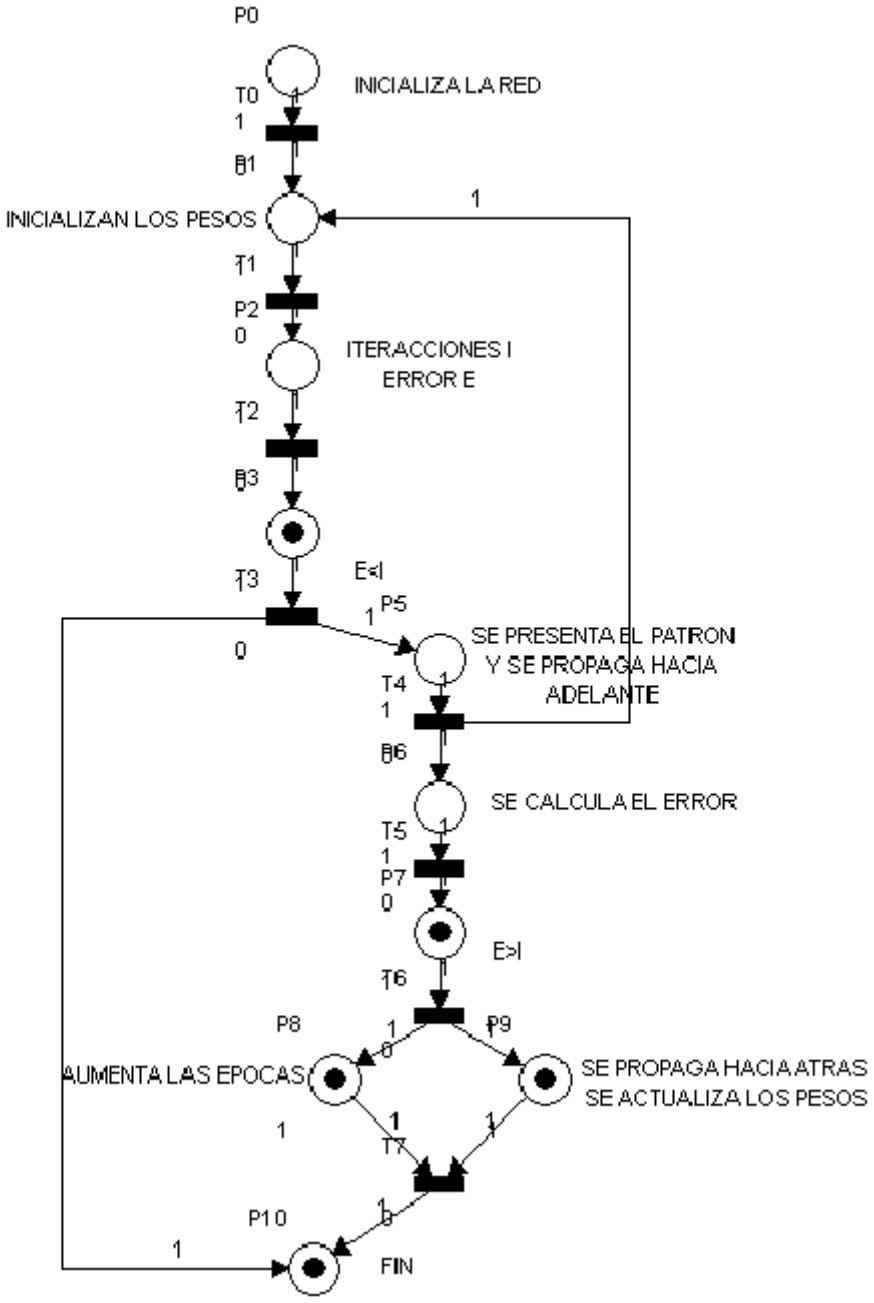


Figura 22: Diagrama de Petri: Fase de Aprendizaje de la Red Neuronal Backpropagation

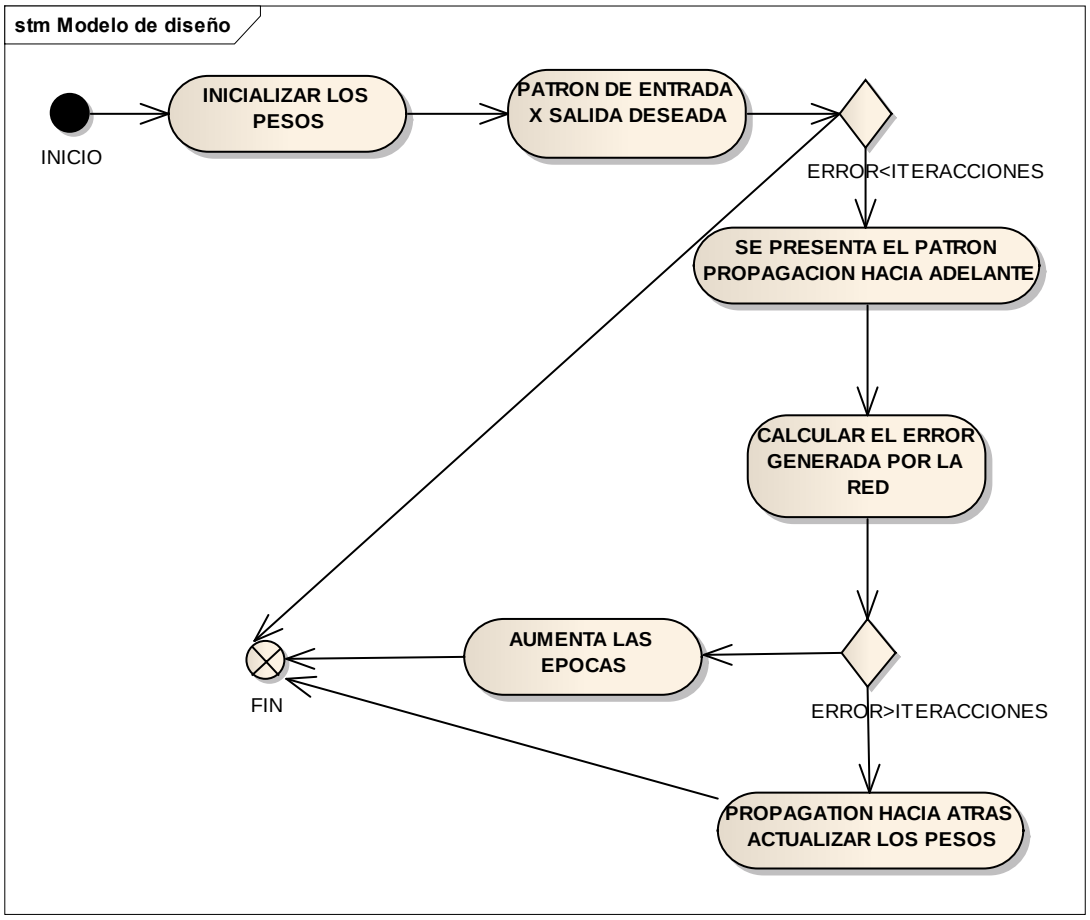
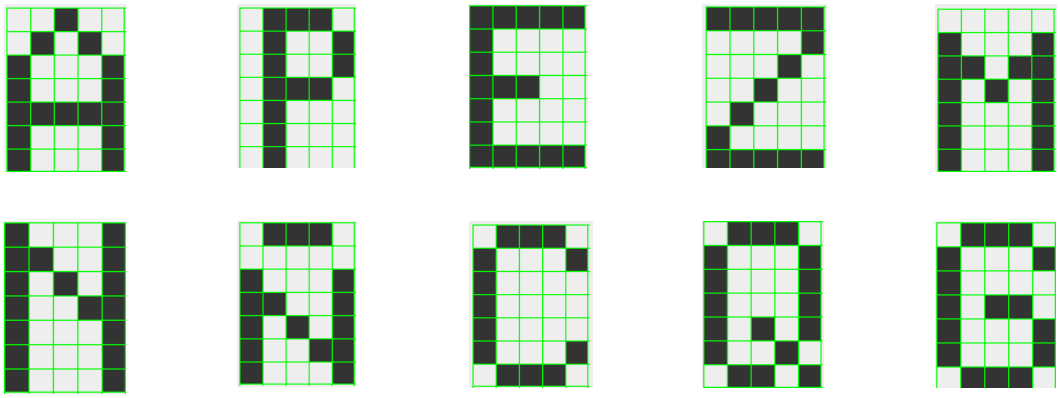


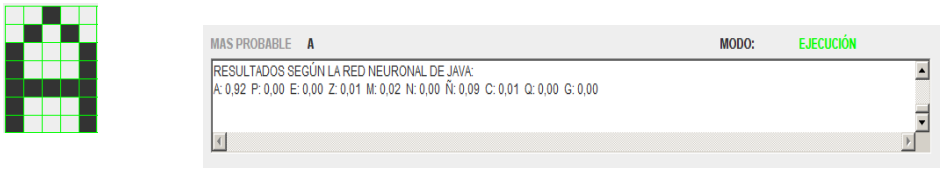
Figura 23: Diagrama de Estado: Fase de Aprendizaje de la Red Neuronal Backpropagation

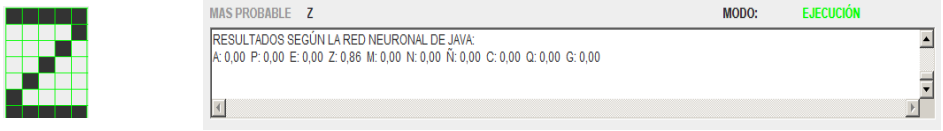
I.9.4 PRUEBAS EXPERIMIENTALES DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION PARA EL RECONOCIMIENTO DE CARACTERES

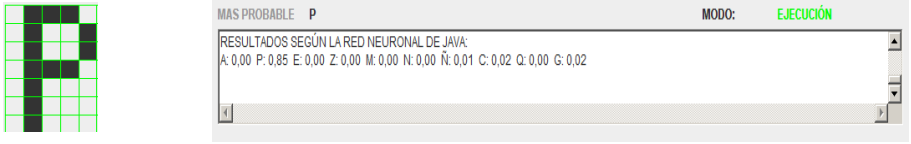
1º PRUEBA DE RED NEURONAL ARTIFICIAL BACKPROPAGATION

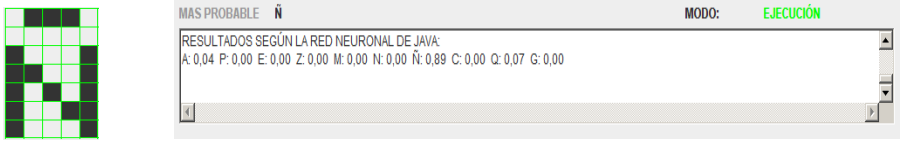


RESULTADOS DE LA PRIMERA FASE DE RECONOCIMIENTO DE CARACTERES









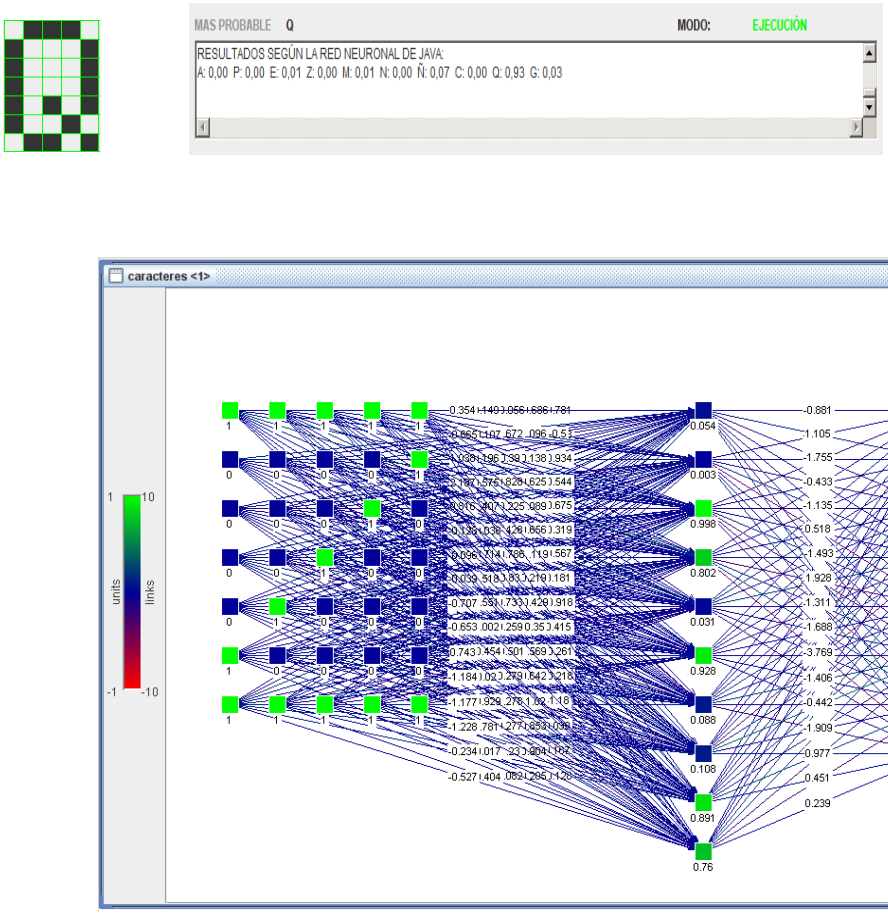
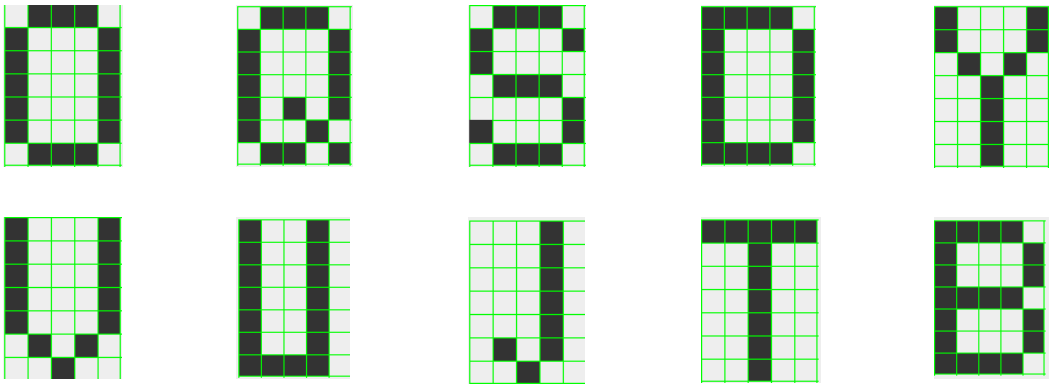
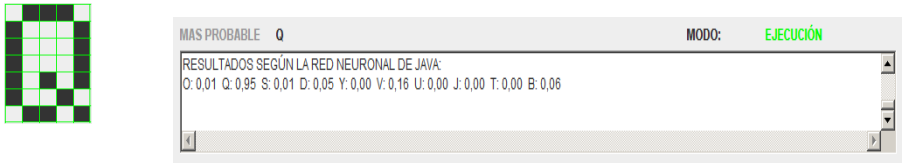


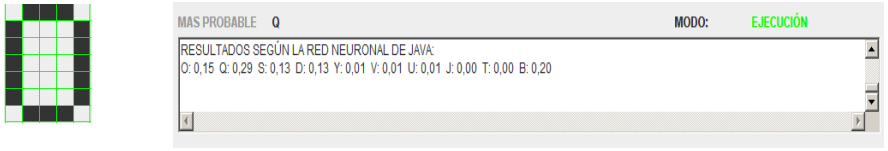
Figura 24: Simulación y estructura de la red neuronal Backpropagation en la primera fase de prueba

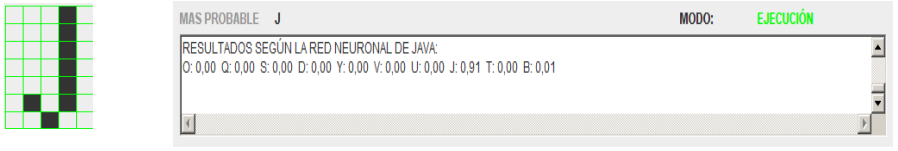
2º PRUEBA DE RED NEURONAL ARTIFICIAL BACKPROPAGATION

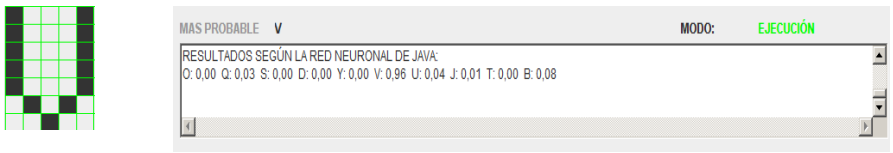


RESULTADOS DE LA SEGUNDA FASE DE RECONOCIMIENTO DE CARACTERES









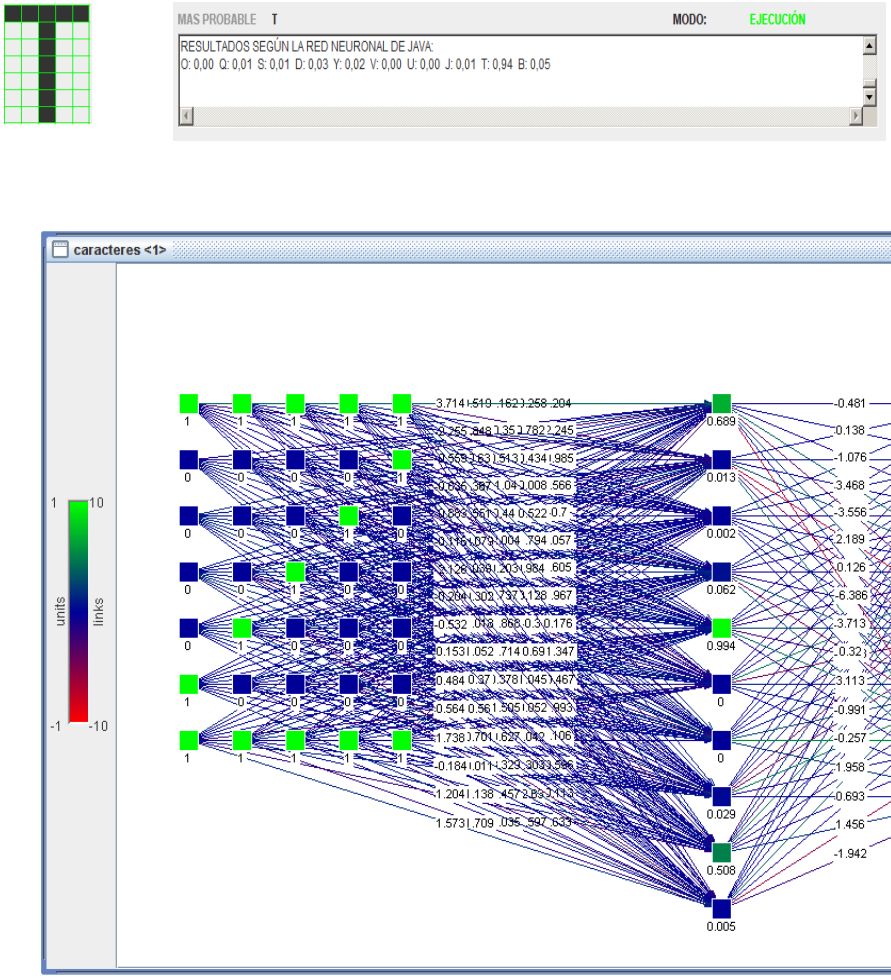
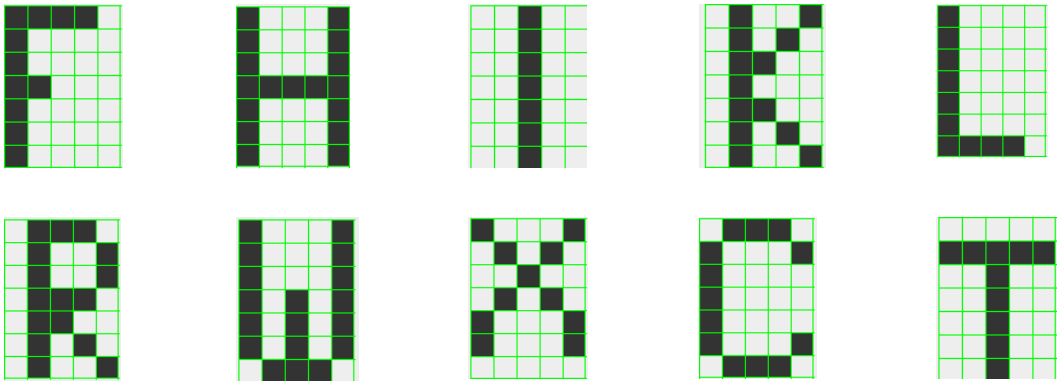
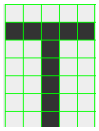


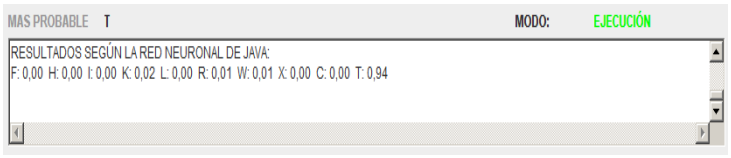
Figura 25: Simulación y estructura de la red neuronal Backpropagation en la segunda fase de prueba

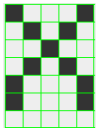
3º PRUEBA DE RED NEURONAL ARTIFICIAL BACPROPAGATION

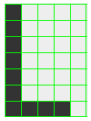


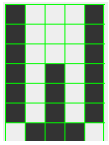
RESULTADOS DE LA TERCERAFASE DE RECONOCIMIENTO DE CARACTERES

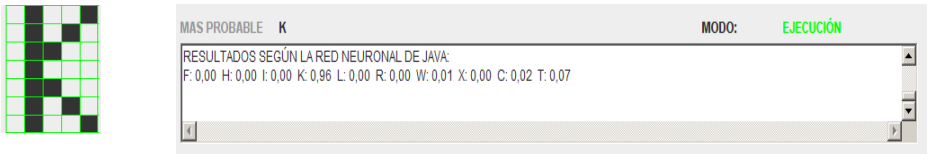












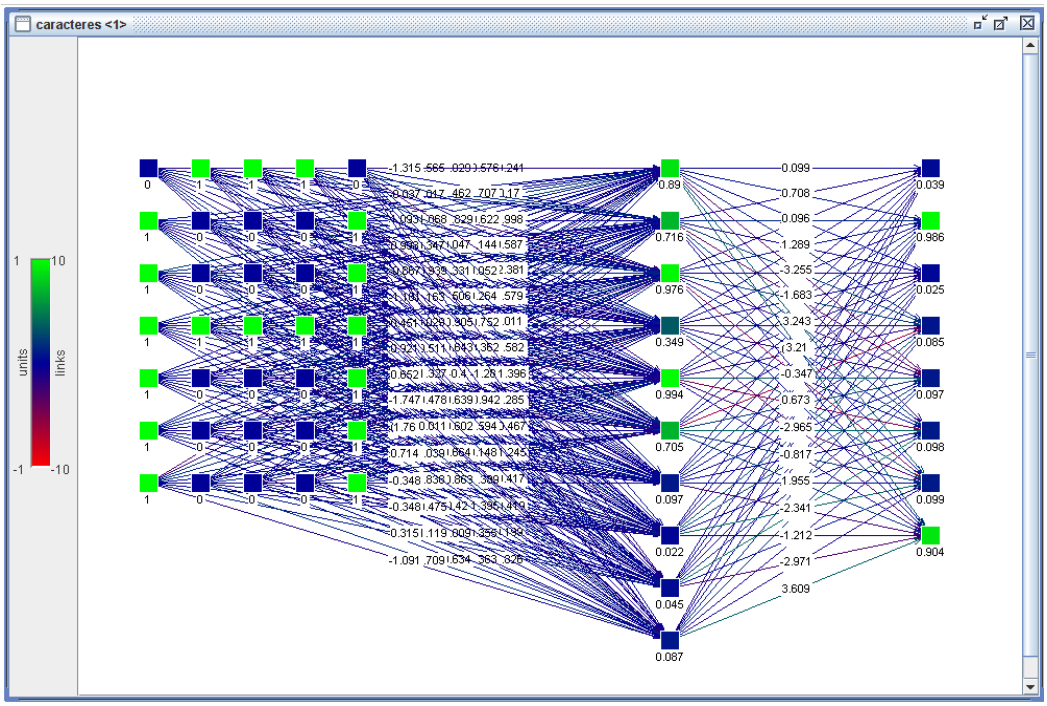


Figura 26: Simulación y estructura de la red neuronal Backpropagation en la tercera fase de prueba

I.9.5 SEUCÓDIGO DE LA RED NEURONAL ARTIFICIAL MADALINE

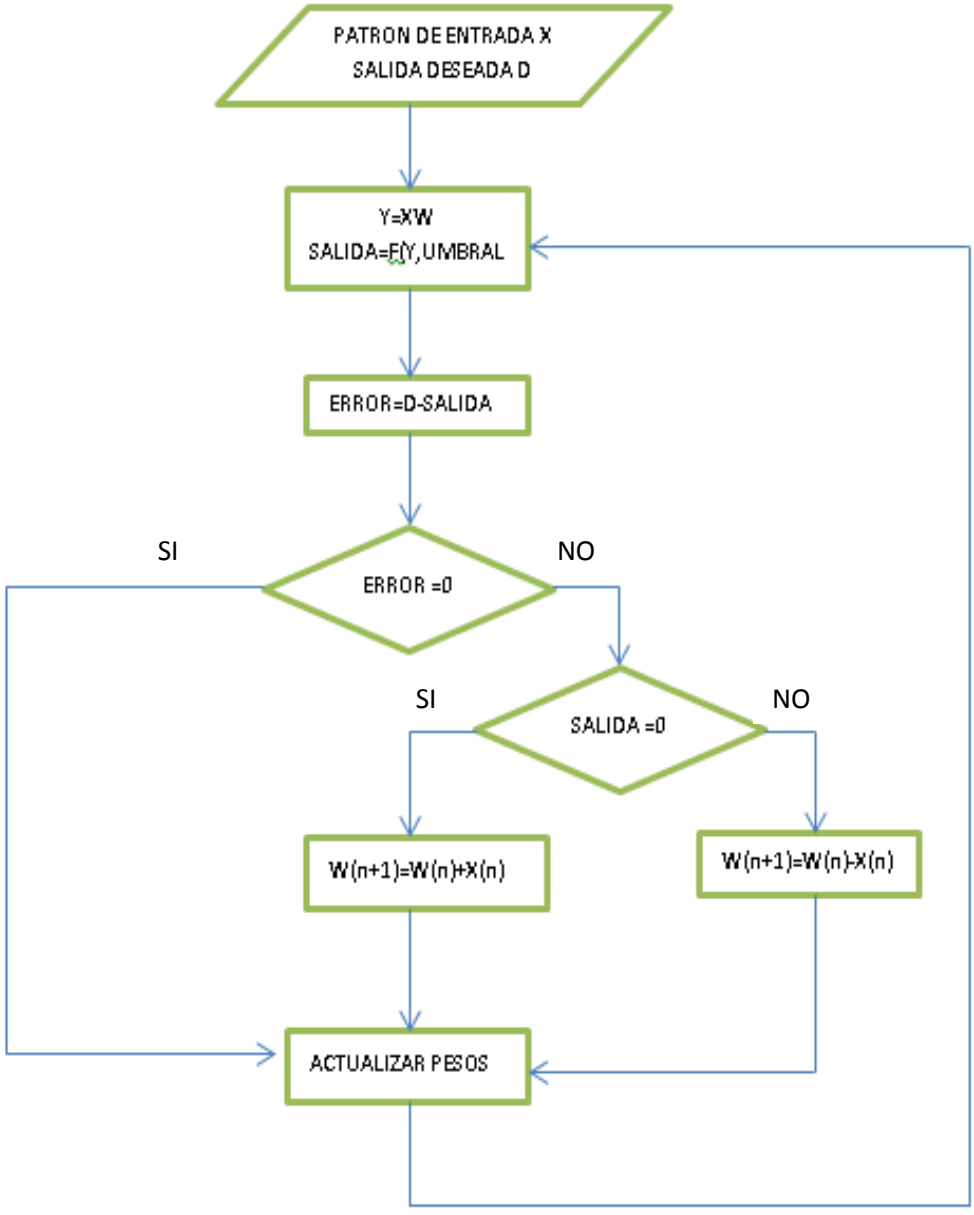


Figura 27: Diagrama de Flujo de Entrenamiento de la Red Neuronal Madaline

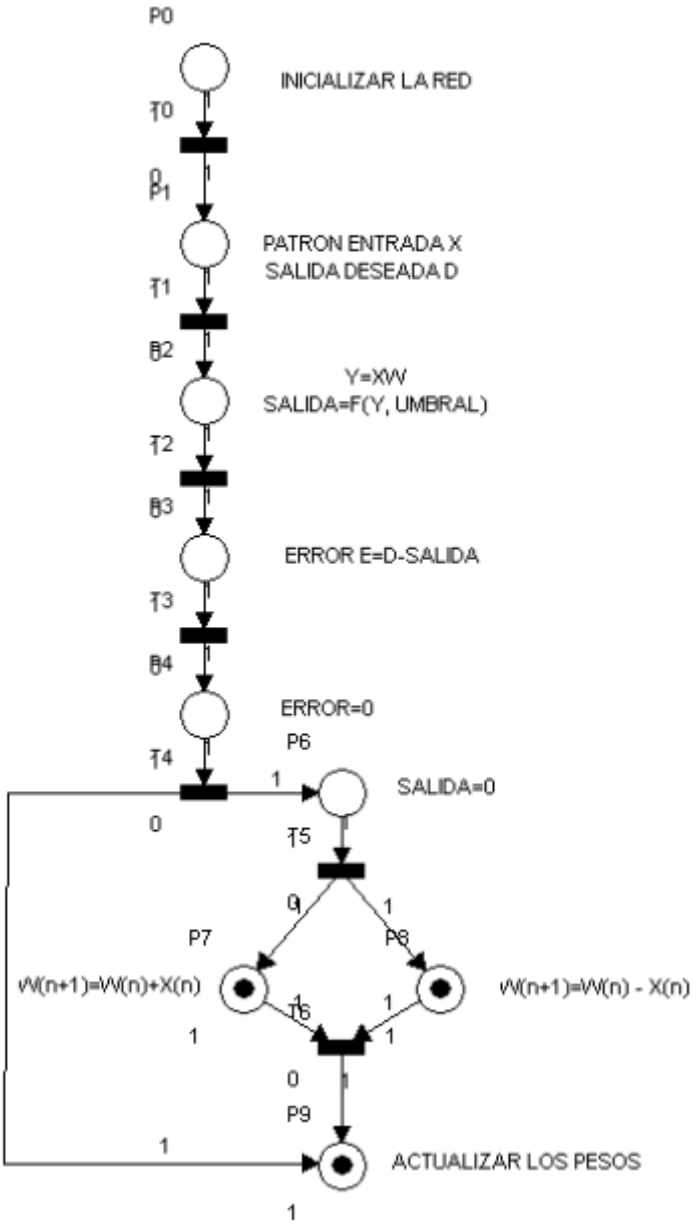


Figura 28: Diagrama de Petri: Fase de Aprendizaje de la Red Neuronal Madaline

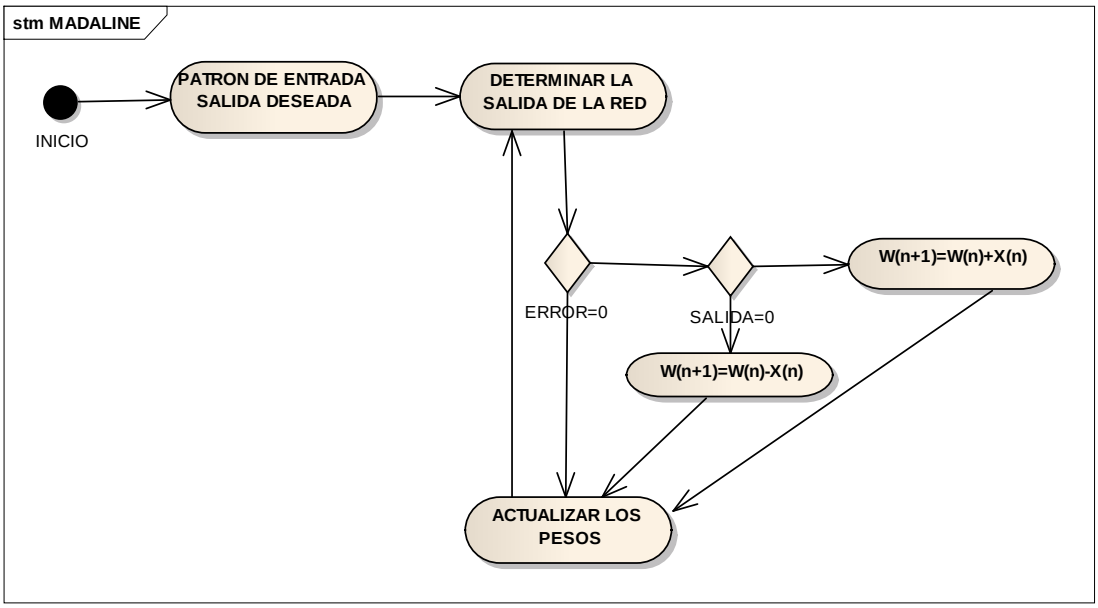
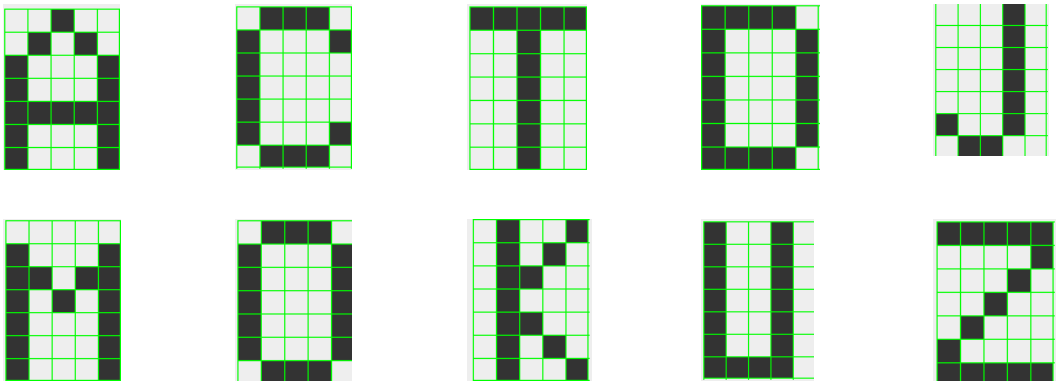


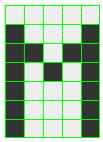
Figura 29: Diagrama de Estados: Fase de Aprendizaje de la Red Neuronal Madaline

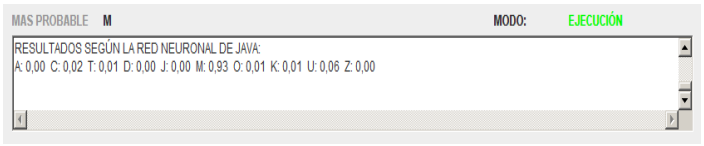
I.9.6 PRUEBAS EXPERIMIENTALES DE LA RED NEURONAL ARTIFICIAL MADALINE PARA EL RECONOCIMIENTO DE CARACTERES

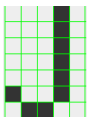
1º PRUEBA DE RED NEURONAL ARTIFICIAL MADALINE

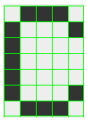


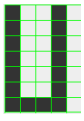
RESULTADOS DE LA PRIMERA FASE DE RECONOCIMIENTO DE CARACTERES











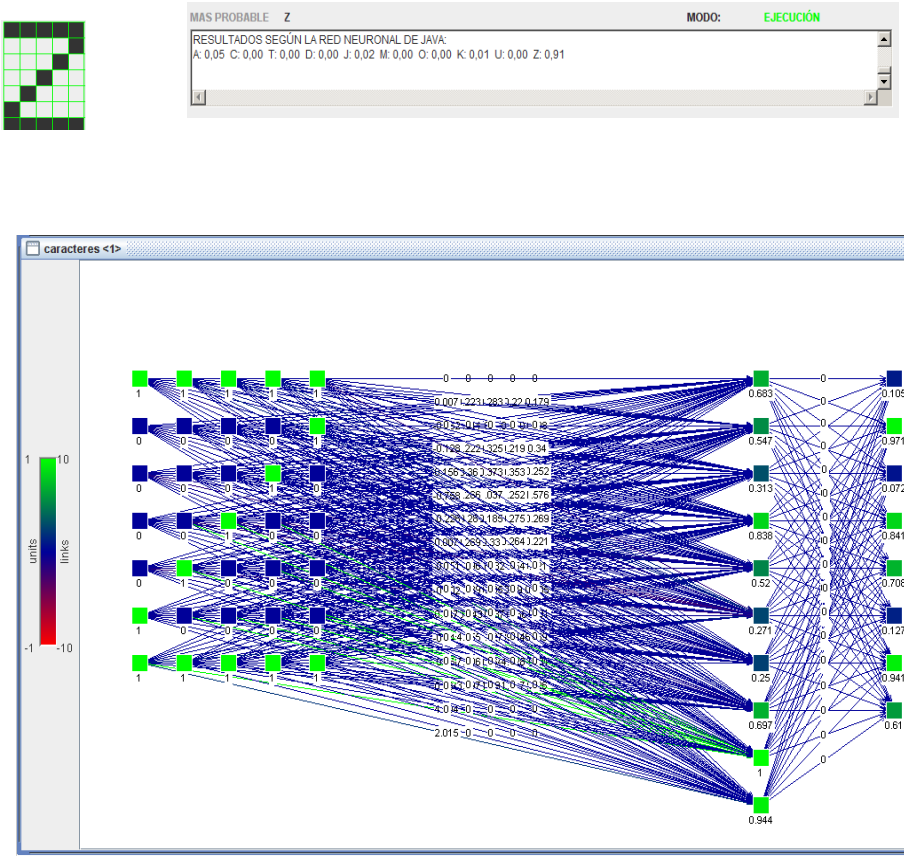
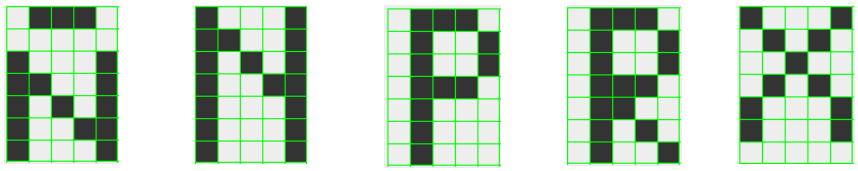
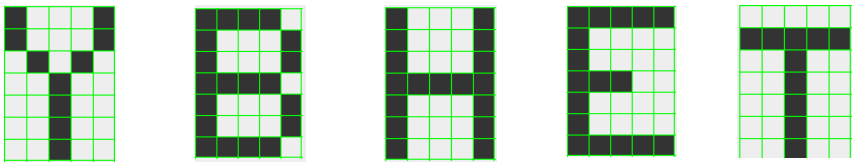


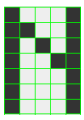
Figura 30: Simulación y estructura de la red neuronal Madaline en la primera fase de prueba

2º PRUEBA DE RED NEURONAL ARTIFICIAL MADALINE



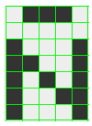


RESULTADOS DE LA SEGUNDA FASE DE RECONOCIMIENTO DE CARACTERES



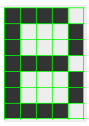
MAS PROBABLE N MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Ñ: 0,18 N: 0,43 P: 0,08 R: 0,30 X: 0,00 B: 0,00 H: 0,00 E: 0,01 T: 0,00 Y: 0,00



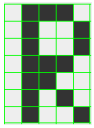
MAS PROBABLE Ñ MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Ñ: 0,83 N: 0,02 P: 0,00 R: 0,01 X: 0,00 B: 0,02 H: 0,00 E: 0,00 T: 0,00 Y: 0,00



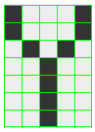
MAS PROBABLE B MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Ñ: 0,01 N: 0,00 P: 0,05 R: 0,02 X: 0,00 B: 0,94 H: 0,01 E: 0,00 T: 0,01 Y: 0,00



MAS PROBABLE R MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Ñ: 0,01 N: 0,05 P: 0,36 R: 0,72 X: 0,02 B: 0,06 H: 0,02 E: 0,02 T: 0,01 Y: 0,03



MAS PROBABLE Y MODO: EJECUCIÓN

RESULTADOS SEGÚN LA RED NEURONAL DE JAVA:
Ñ: 0,00 N: 0,00 P: 0,05 R: 0,01 X: 0,00 B: 0,00 H: 0,00 E: 0,00 T: 0,04 Y: 0,92

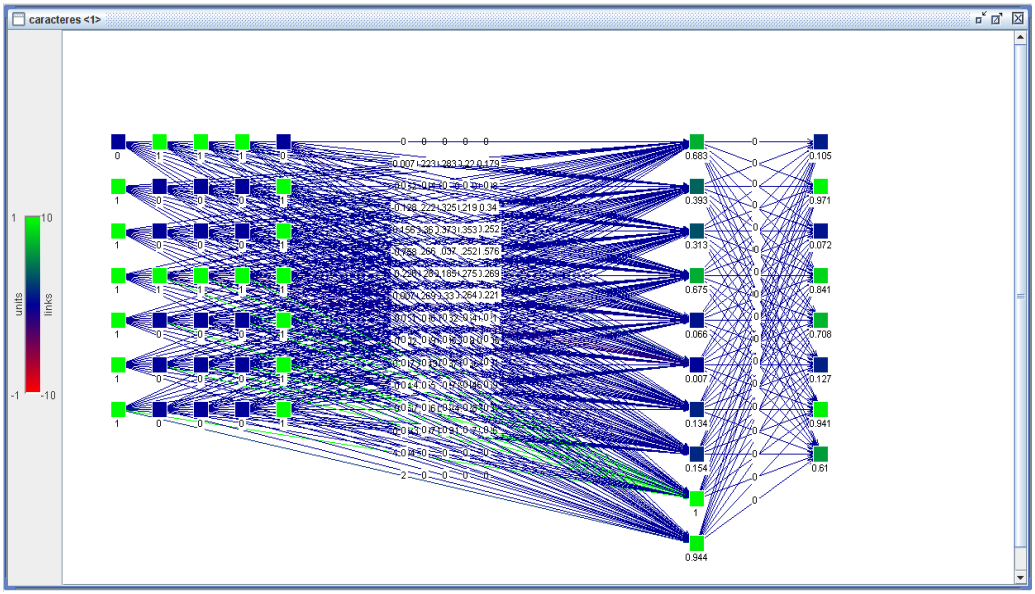
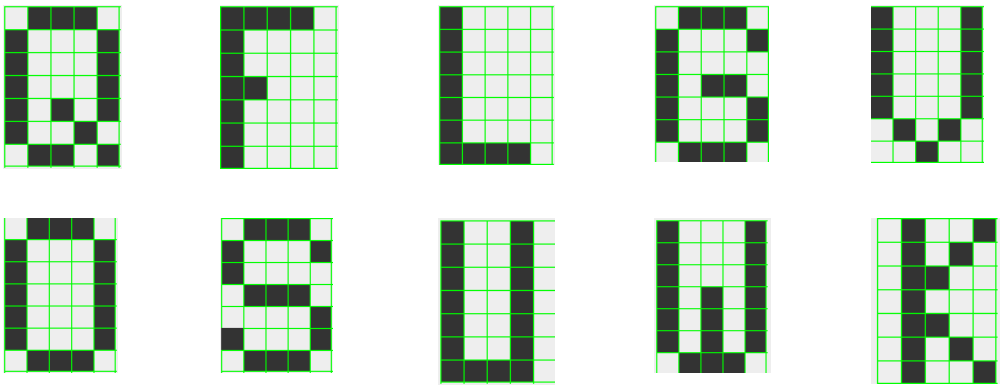
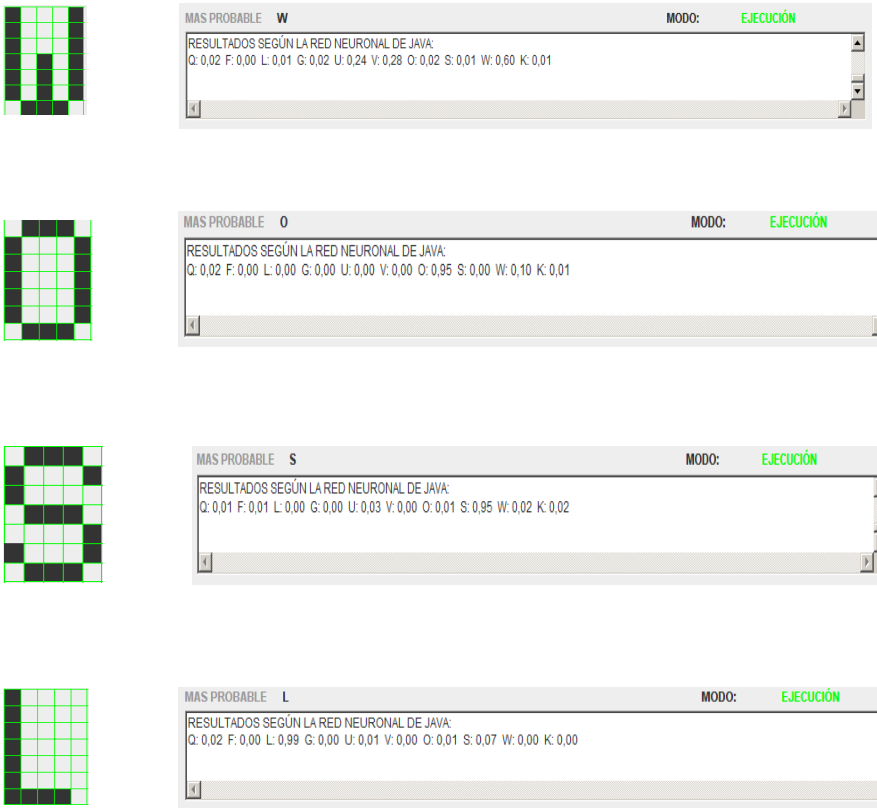


Figura 31: Simulación y estructura de la red neuronal Madaline en la segunda fase de prueba

3º PRUEBA DE RED NEURONAL ARTIFICIAL MADALINE



RESULTADOS DE LA TERCERA FASE DE RECONOCIMIENTO DE CARACTERES



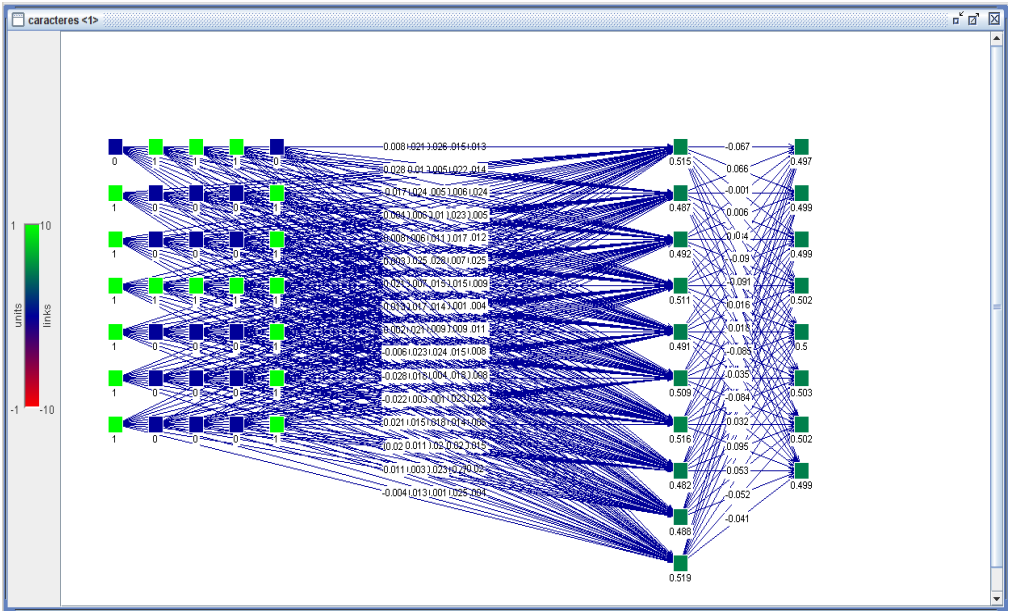


Figura 32: Simulación y estructura de la red neuronal Madaline en la tercera fase de prueba

I.9.7 SEUDOCODIGO DE LA RED NEURONAL ARTIFICIAL HOPFIELD

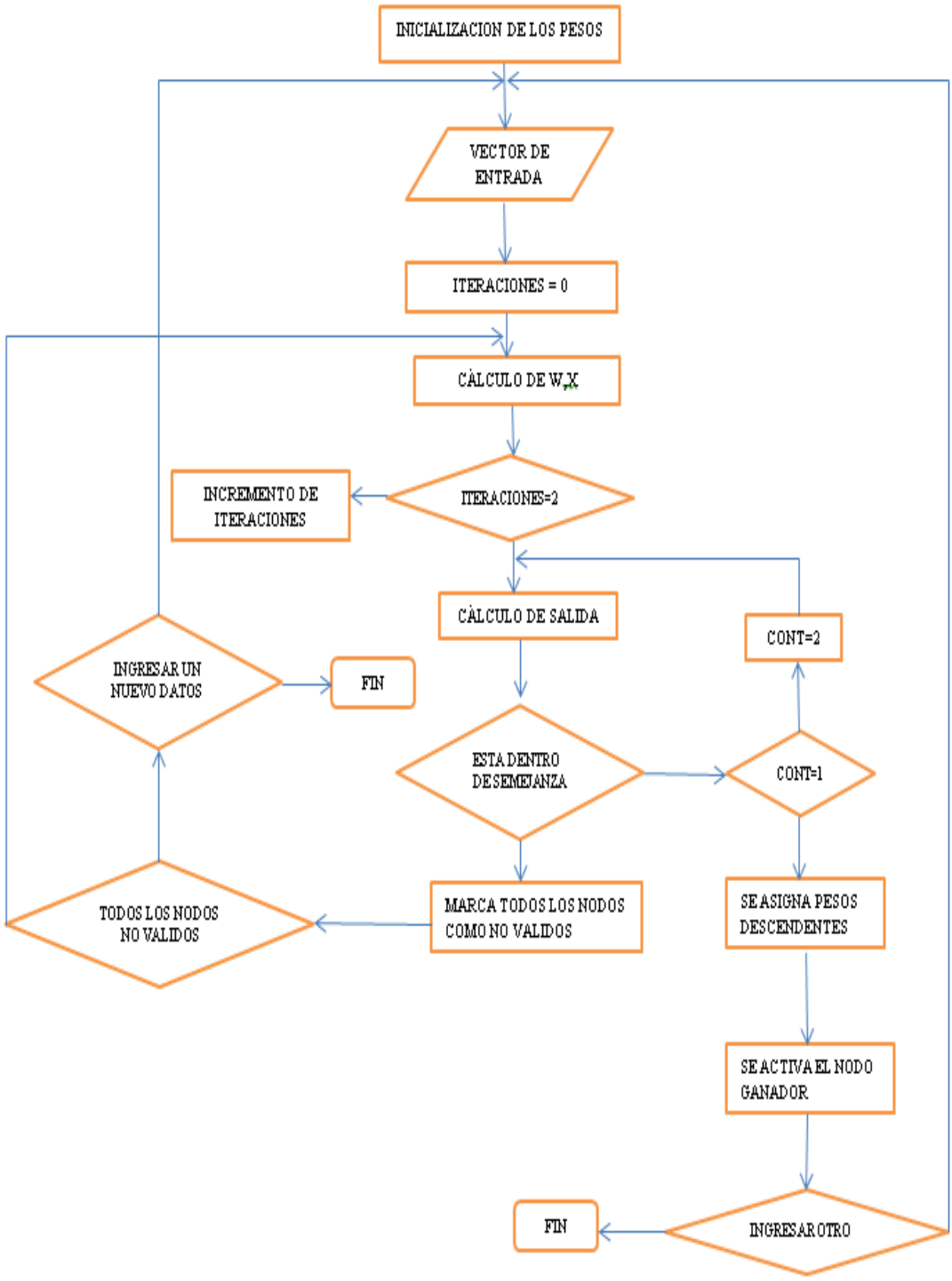


Figura 33: Seudocódigo de la red neuronal artificial Hopfield

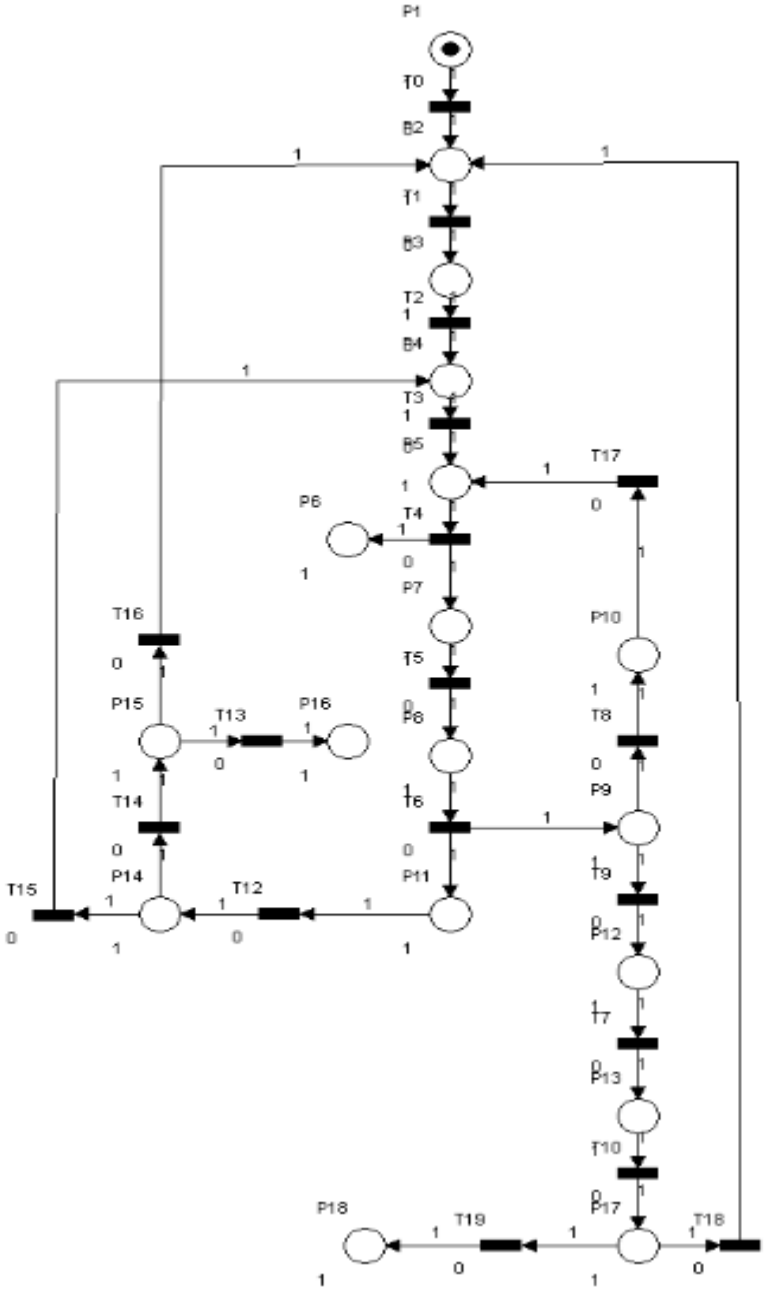


Figura 34: Diagrama de Petri: Fase de Entrenamiento de la Red Neuronal Hopfield

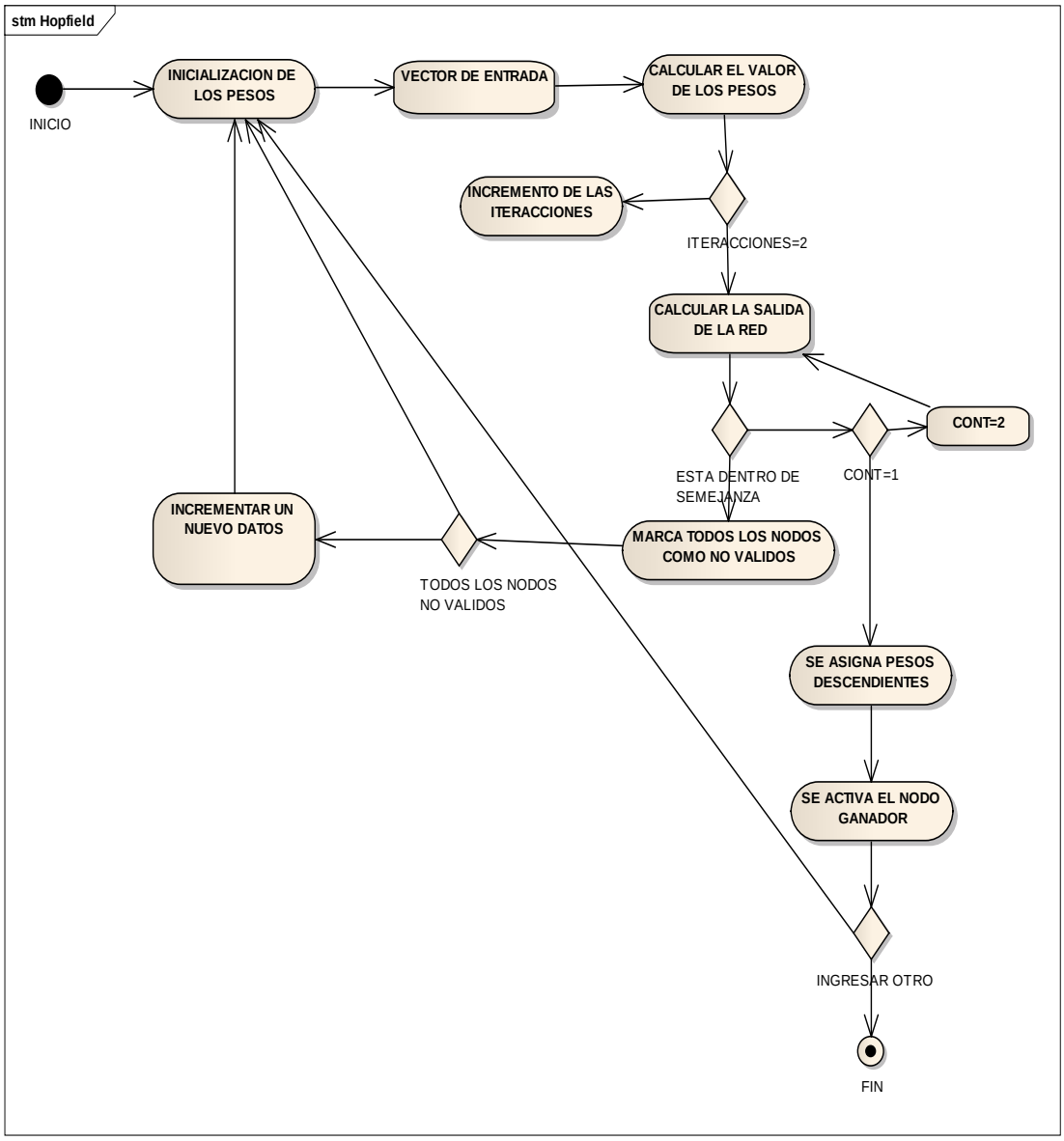
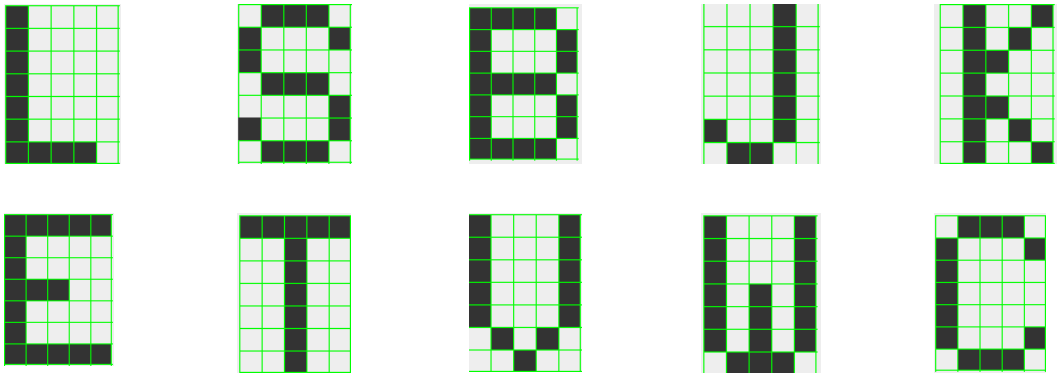


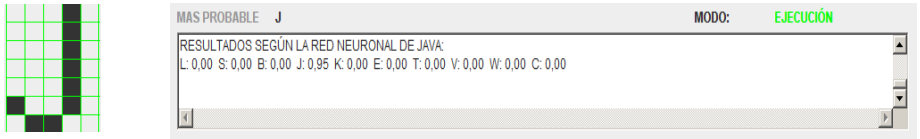
Figura 35: Diagrama de Estados: Fase de Entrenamiento de la Red Neuronal Hopfield

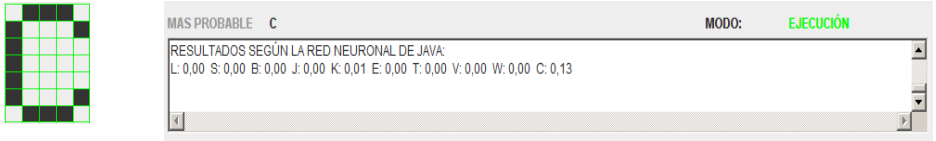
I.9.8 PRUEBAS EXPERIMIENTALES DE LA RED NEURONAL ARTIFICIAL HOPFIELD PARA EL RECONOCIMIENTO DE CARACTERES

1º PRUEBA DE RED NEURONAL ARTIFICIAL HOPFIELD



RESULTADOS DE LA PRIMERA FASE DE RECONOCIMIENTO DE CARACTERES





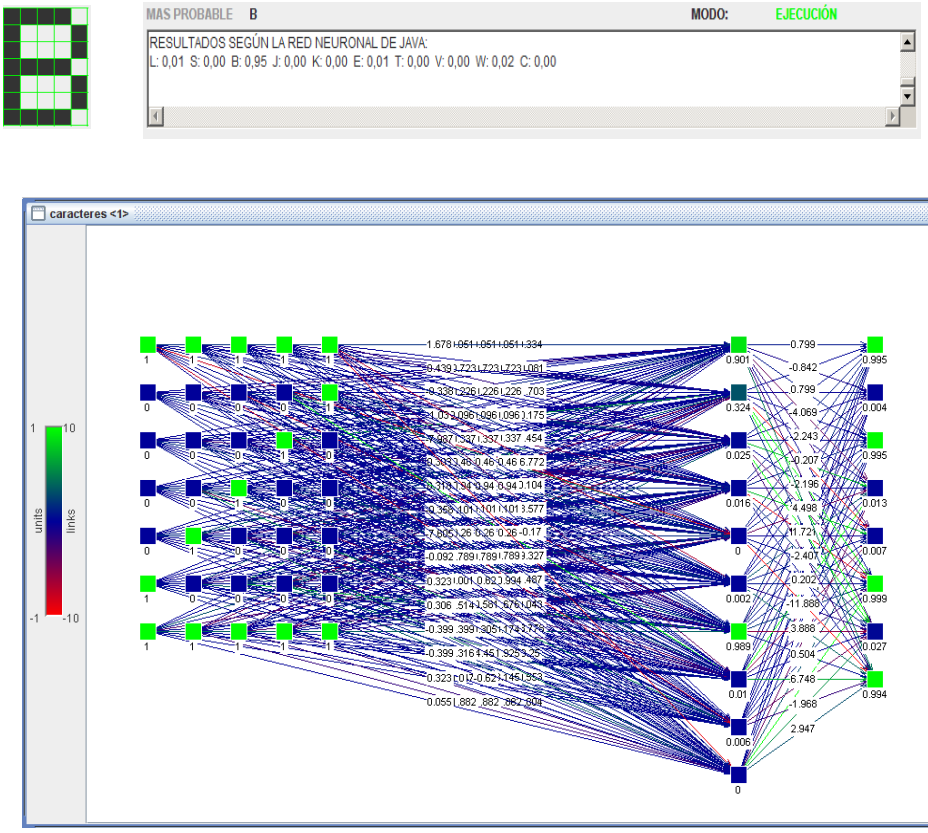
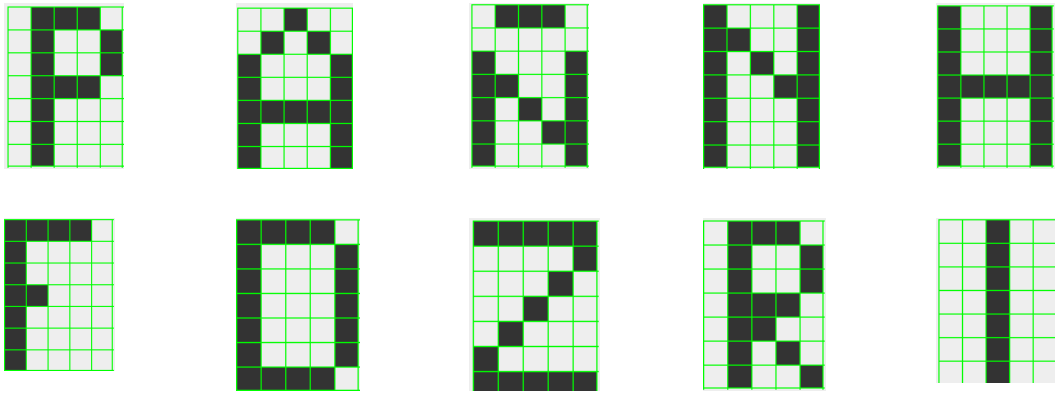
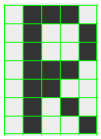


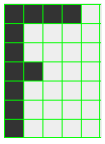
Figura 36: Simulación y estructura de la red neuronal Hopfield en la primera fase de prueba

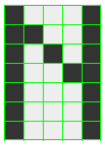
2º PRUEBA DE RED NEURONAL ARTIFICIAL HOPFIELD



RESULTADOS DE LA SEGUNDA FASE DE RECONOCIMIENTO DE CARACTERES







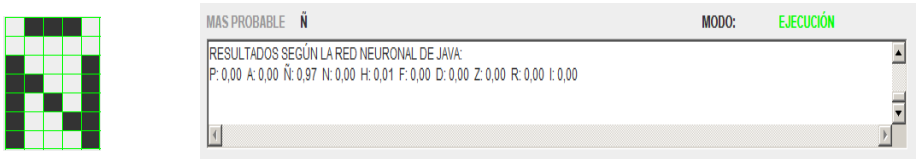
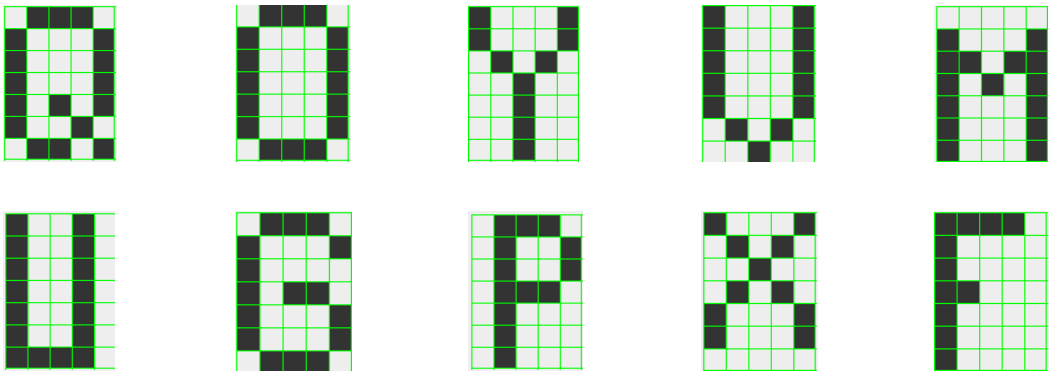
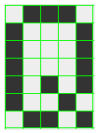


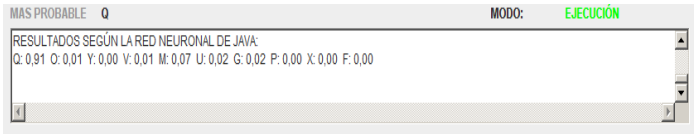
Figura 37: Simulación y estructura de la red neuronal Hopfield en la segunda fase de prueba

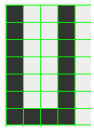
3º PRUEBA DE RED NEURONAL ARTIFICIAL HOPFIELD

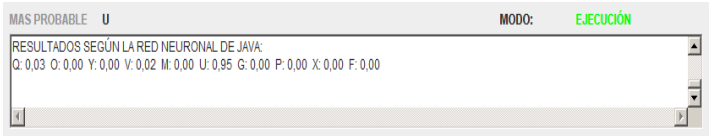


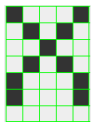
RESULTADOS DE LA TERCERA FASE DE RECONOCIMIENTO DE CARACTERES

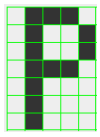












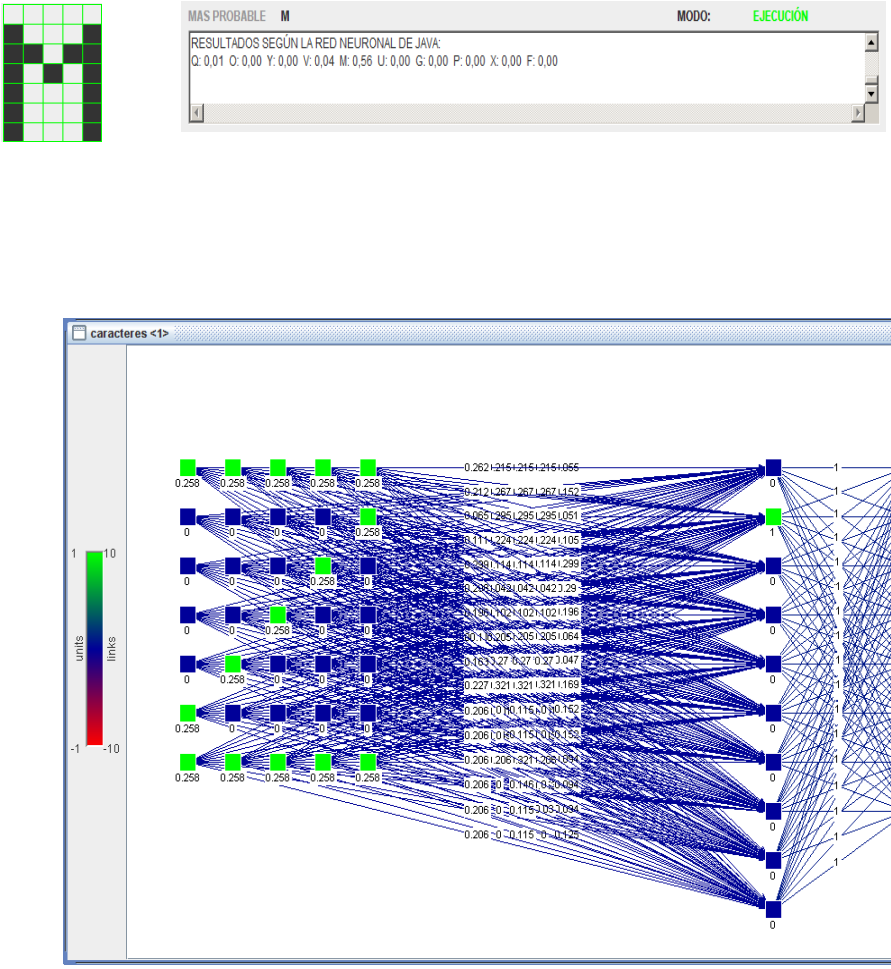
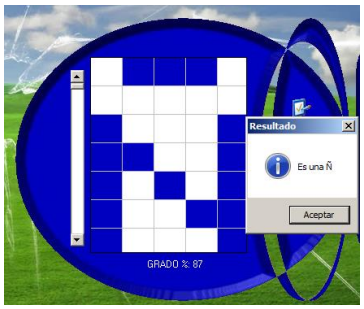


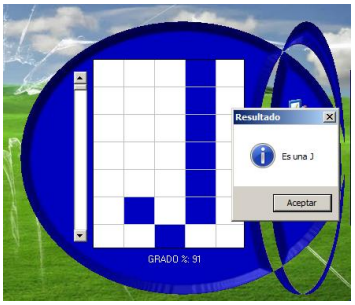
Figura 38: Simulación y estructura de la red neuronal Hopfield en la tercera fase de prueba

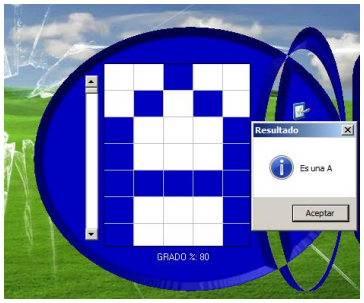
I.10 PRUEBAS EXPERIMENTALES DE LAS REDES NEURONALES ARTIFICIALES PROGRAMADAS EN EL LENGUAJE VISUAL BASIC 6

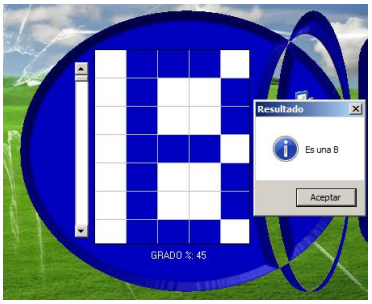
I.10.1 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA PARA EL RECONOCIMIENTO DE CARACTERES

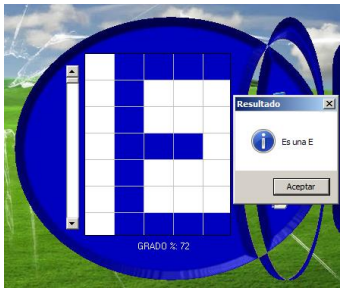
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA



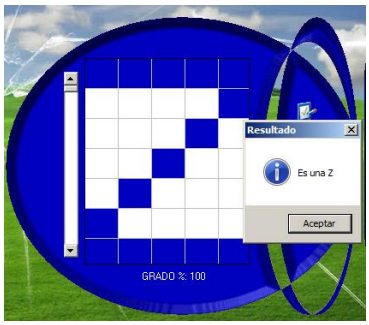


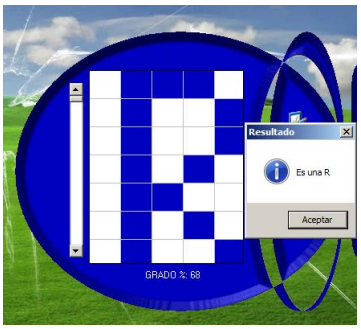


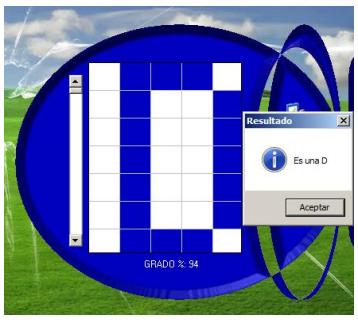


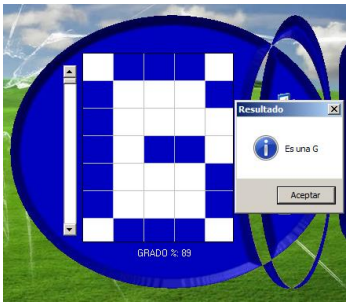


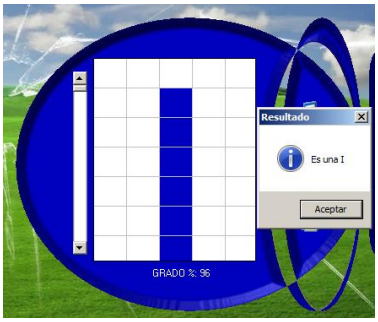
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA



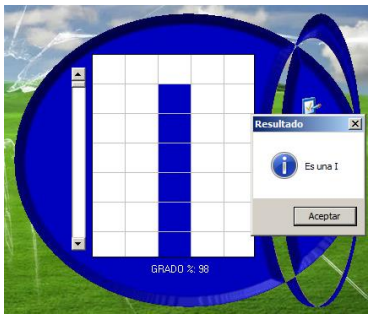


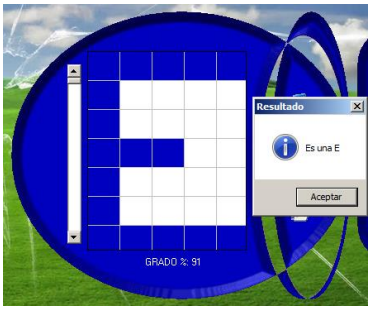


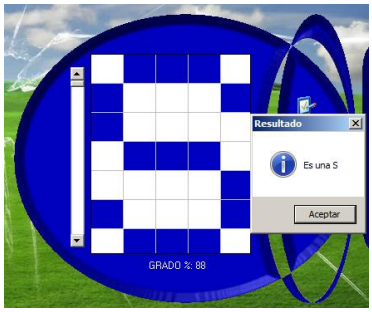


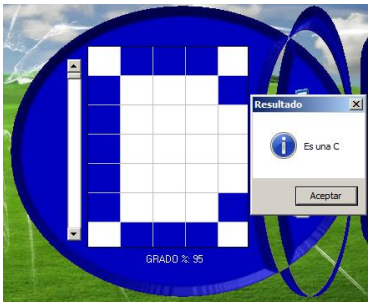


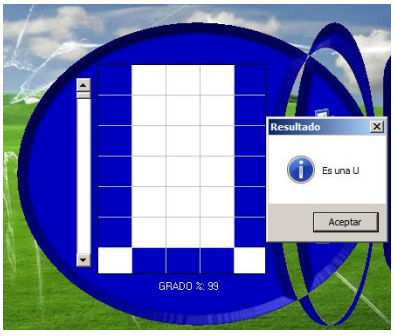
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA





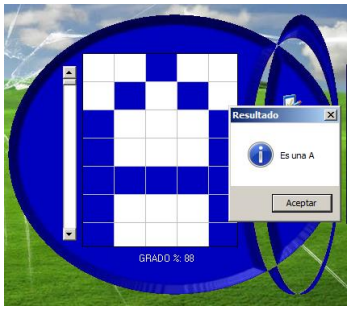


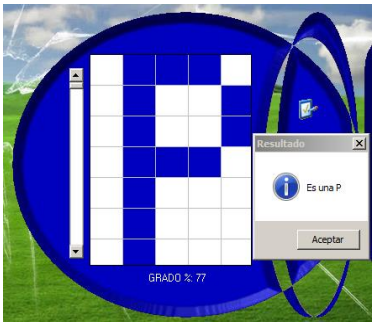


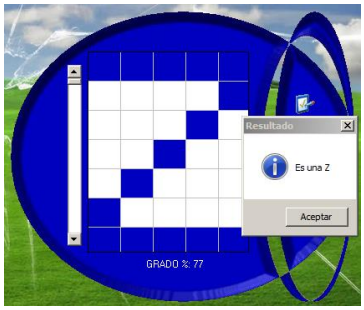


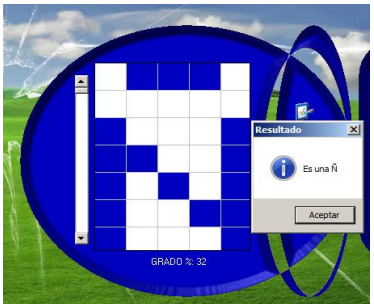
I.10.2 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION PARA EL RECONOCIMIENTO DE CARACTERES

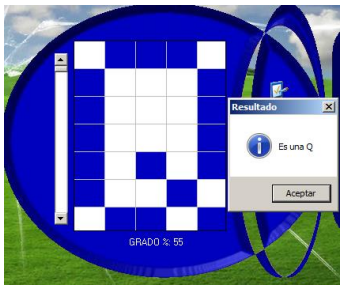
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL BAKPROPAGATION



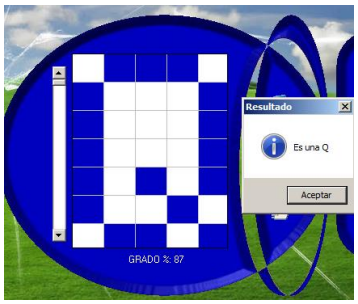


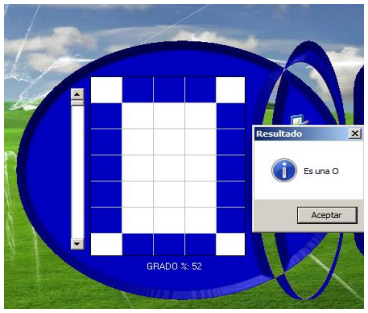


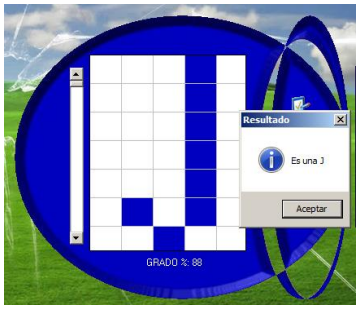


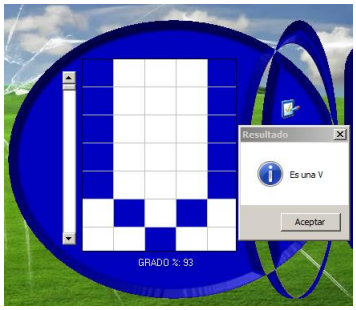


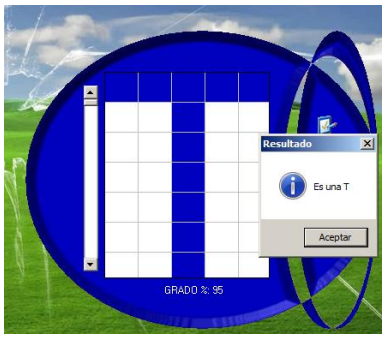
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION



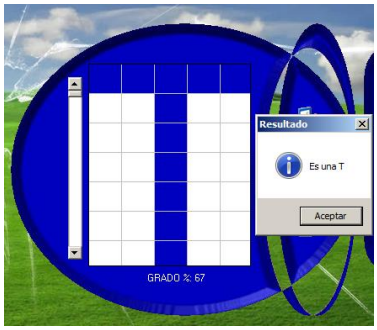


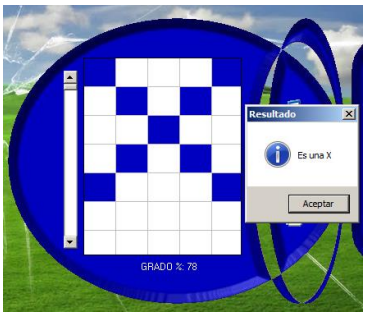


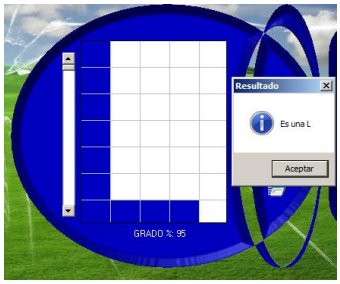


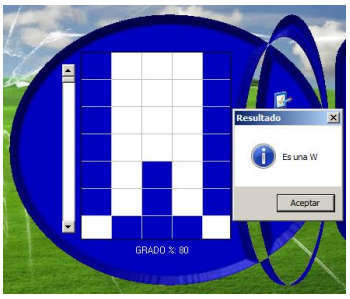


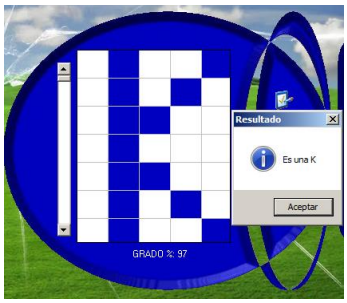
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION





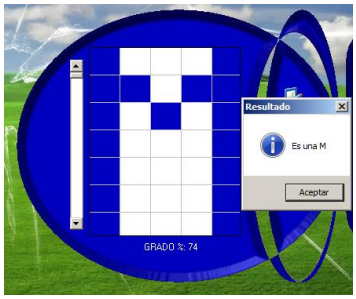


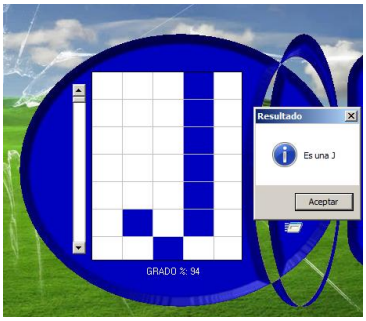


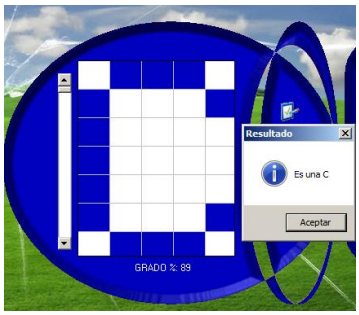


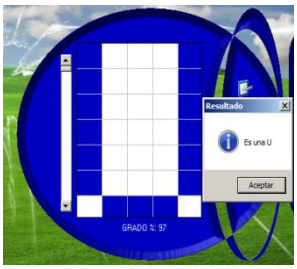
I.10.3 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL MADALINE PARA EL RECONOCIMIENTO DE CARACTERES

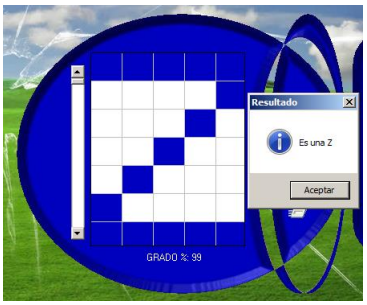
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL MADALINE



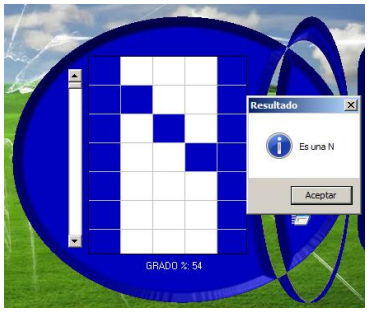


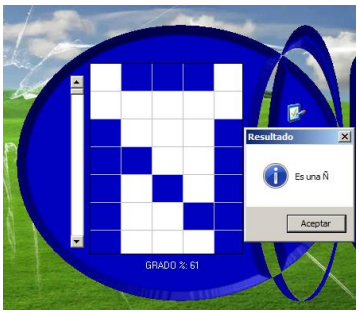


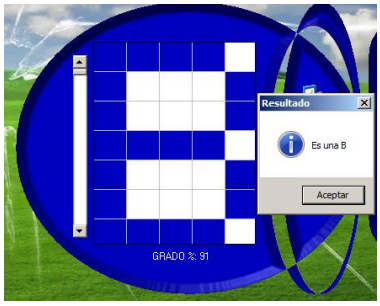


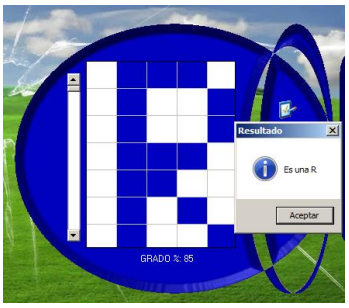


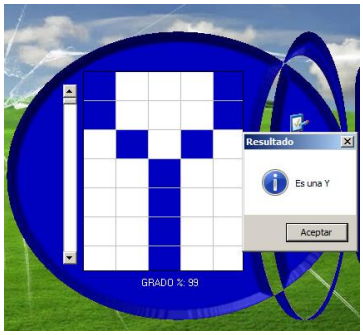
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL MADALINE



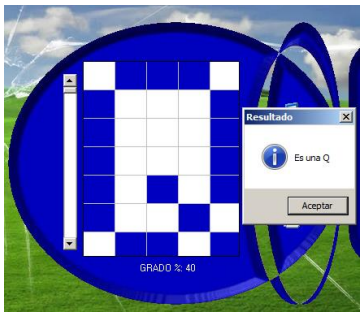


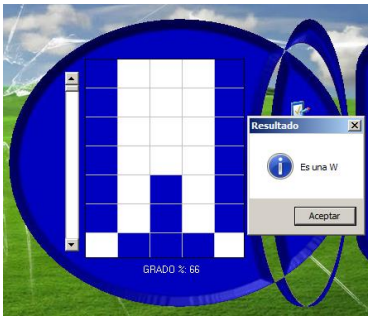


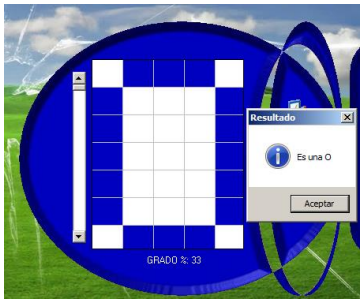


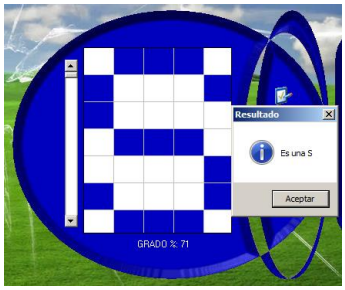


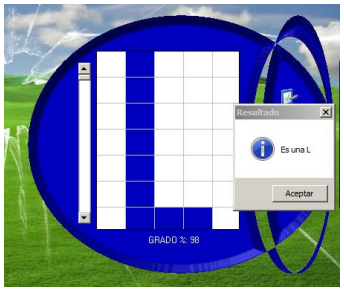
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL MADALINE





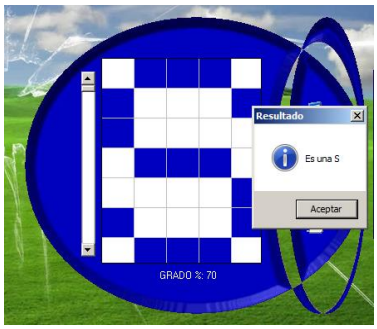


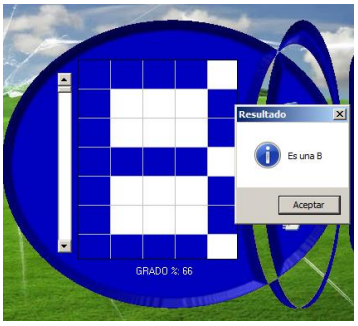


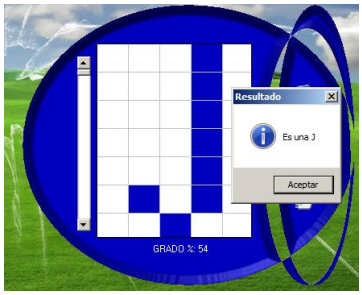


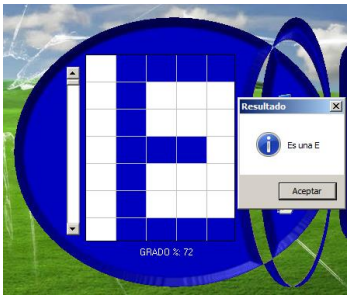
I.10.4 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL HOPFIELD PARA EL RECONOCIMIENTO DE CARACTERES

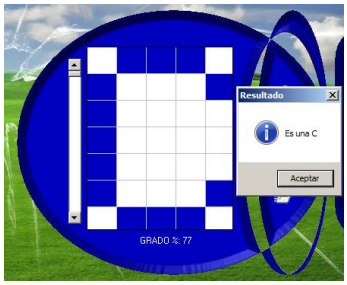
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL HOPFIELD



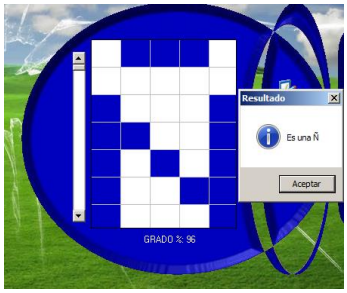


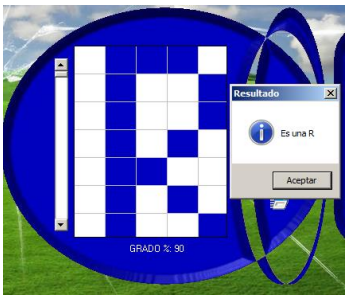


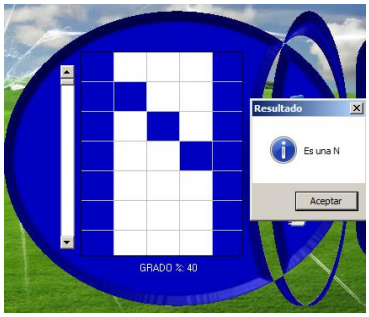


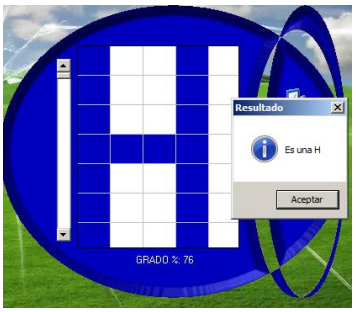


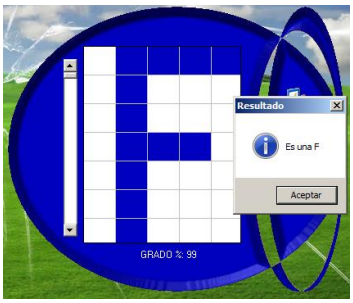
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL HOPFIELD



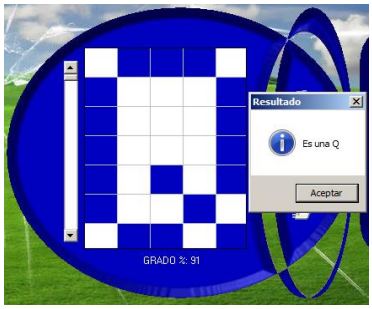


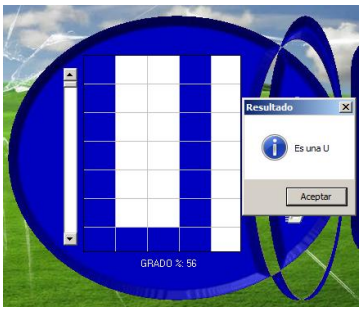


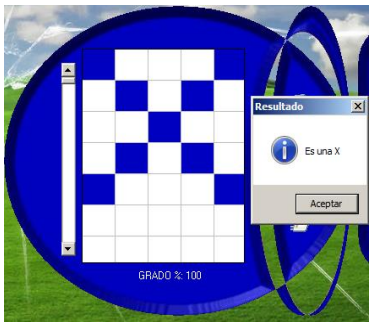


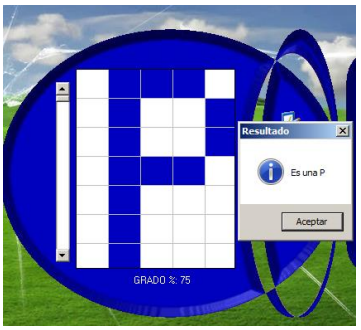


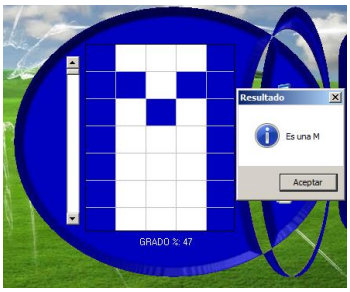
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL HOPFIELD







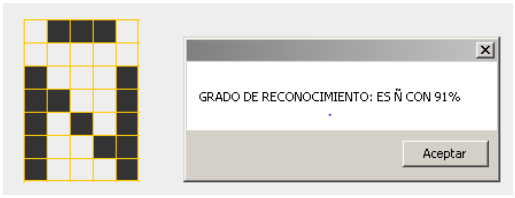


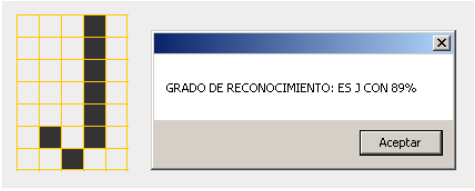


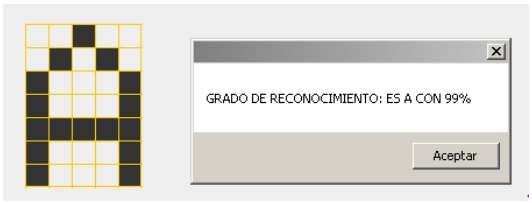
I.11 PRUEBAS EXPERIMENTALES DE LAS REDES NEURONALES ARTIFICIALES PROGRAMADAS EN EL LENGUAJE C#

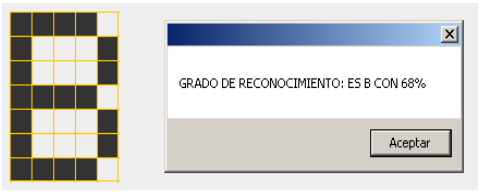
I.11.1 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA PARA EL RECONOCIMIENTO DE CARACTERES

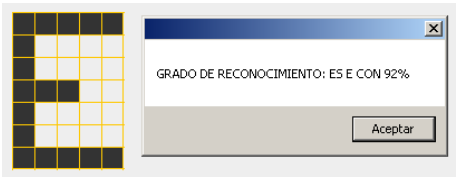
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA



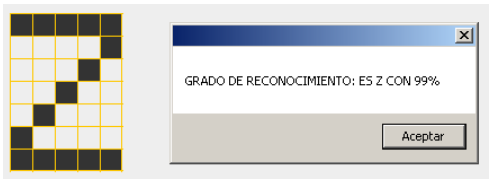


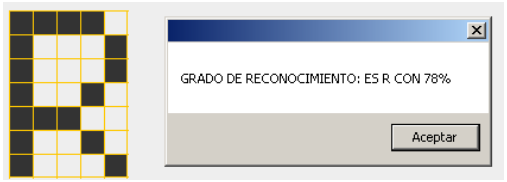


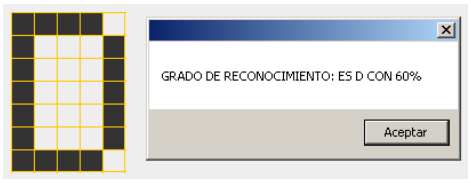


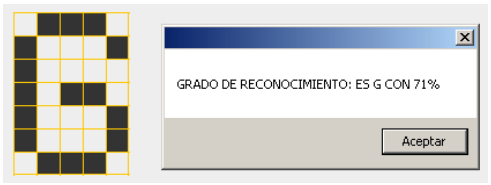


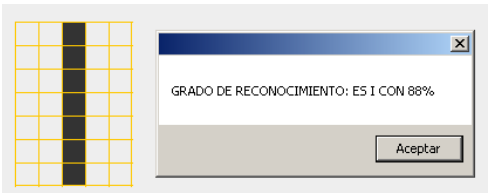
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA



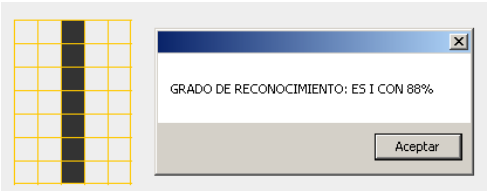


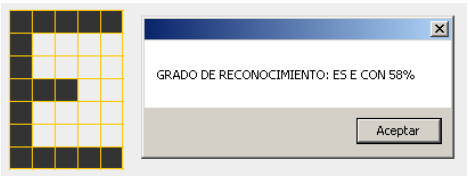


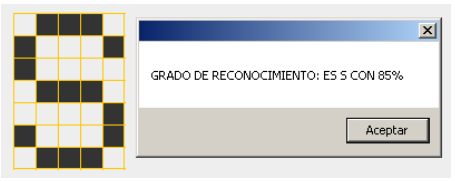


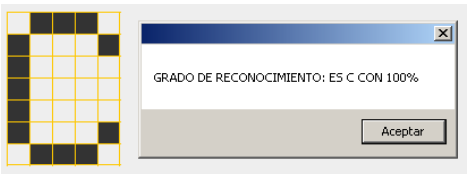


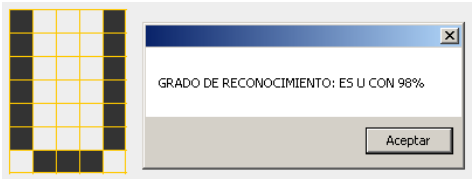
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL PERCEPTRON MULTICAPA





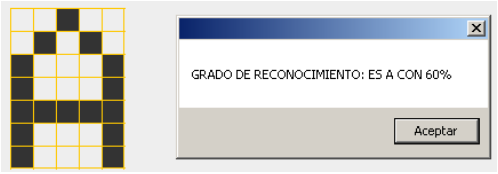


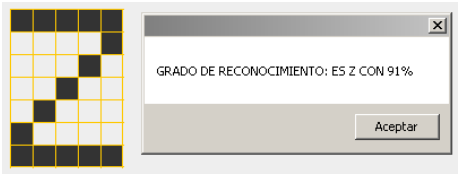


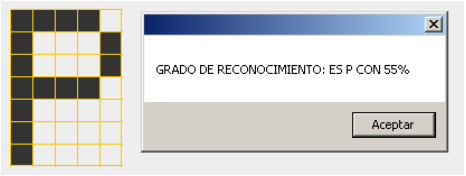


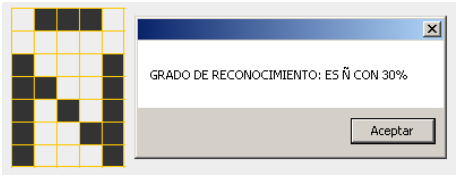
I.11.2 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION PARA EL RECONOCIMIENTO DE CARACTERES

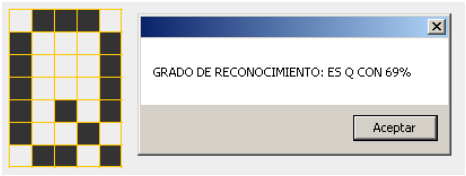
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION



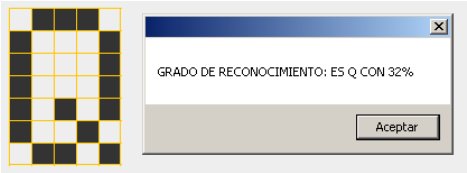


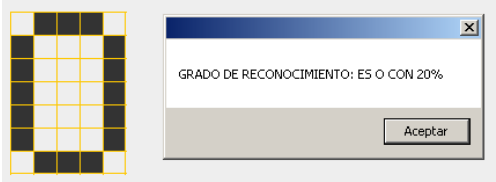


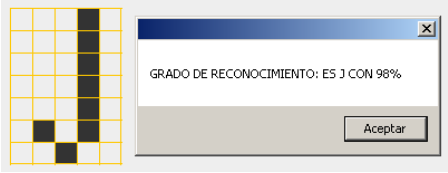


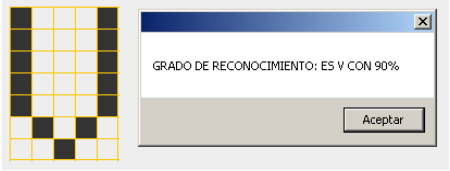


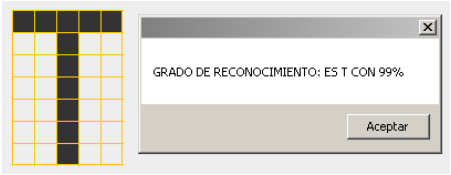
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION



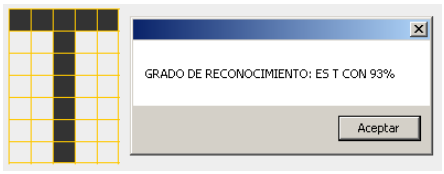


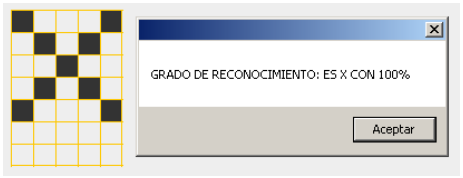


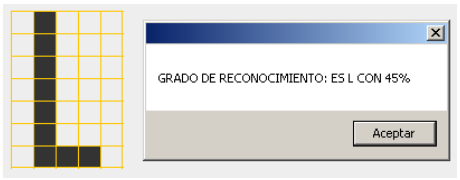


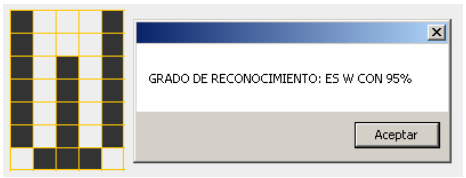


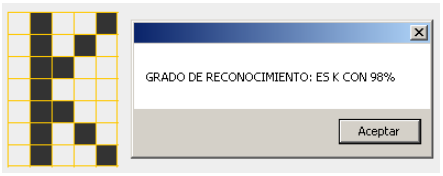
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL BACKPROPAGATION





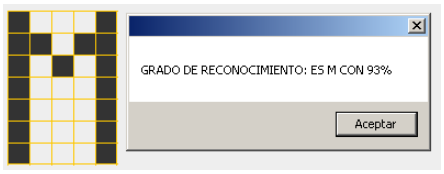


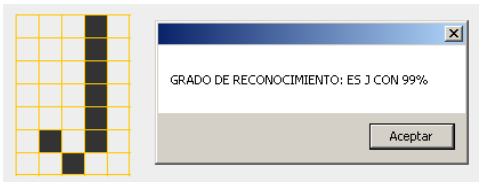


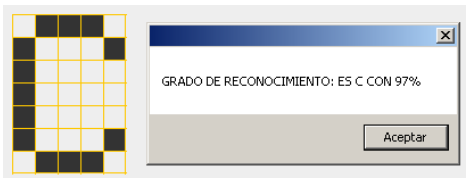


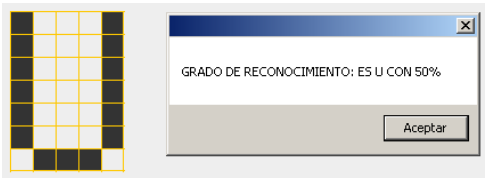
I.11.3 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL MADALINE PARA EL RECONOCIMIENTO DE CARACTERES

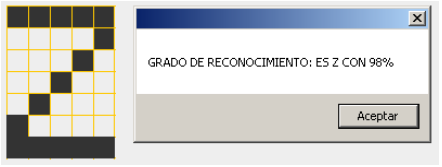
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL MADALINE



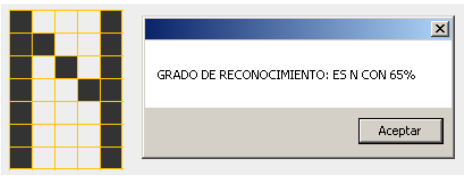


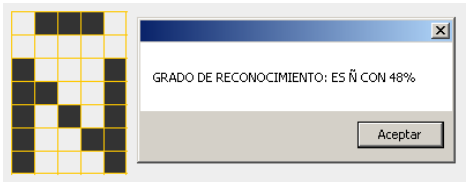


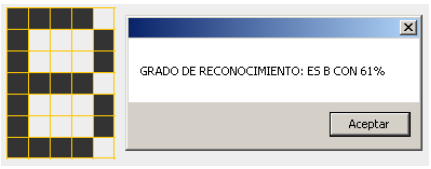


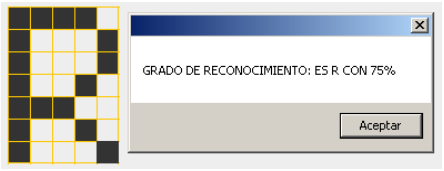


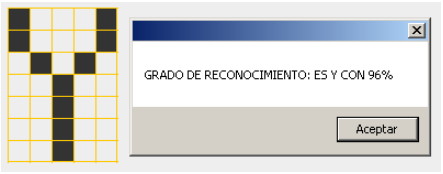
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL MADALINE



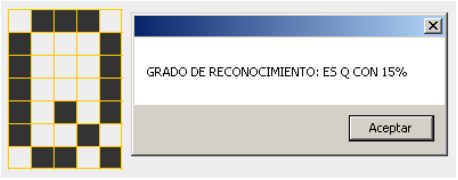


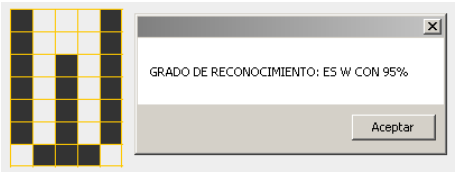


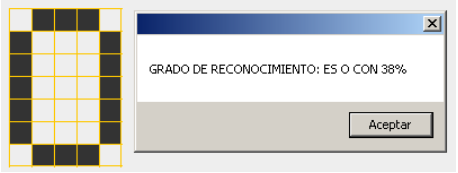


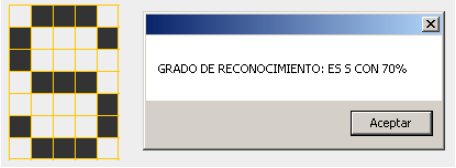


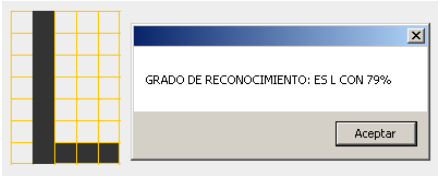
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL MADALINE





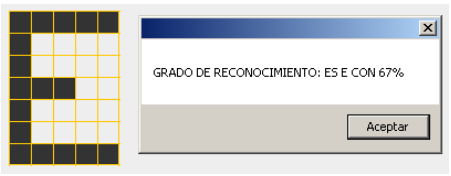


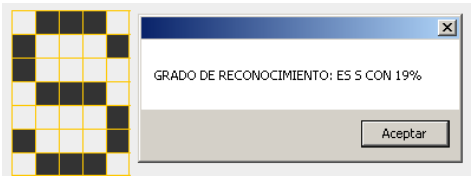


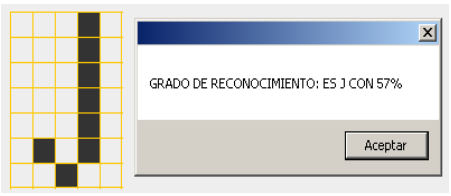


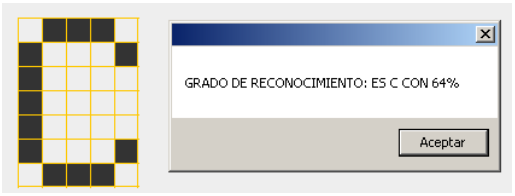
I.11.4 PRUEBAS EXPERIMENTALES DE LA RED NEURONAL ARTIFICIAL HOPFIELD PARA EL RECONOCIMIENTO DE CARACTERES

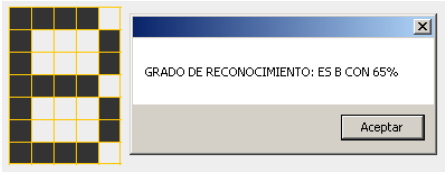
1º PRUEBA DE LA RED NEURONAL ARTIFICIAL HOPFIELD



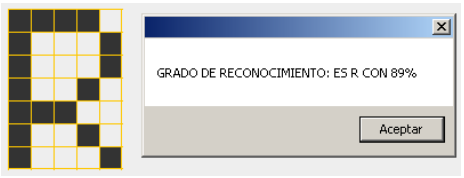


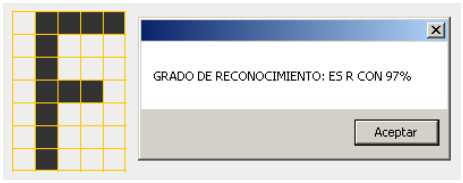


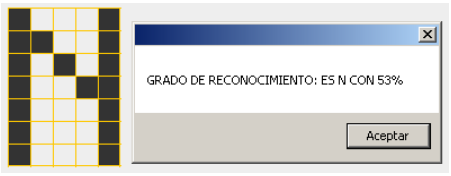


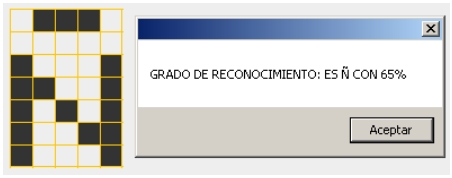


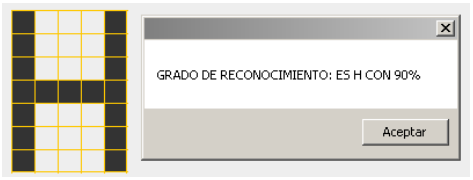
2º PRUEBA DE LA RED NEURONAL ARTIFICIAL HOPFIELD



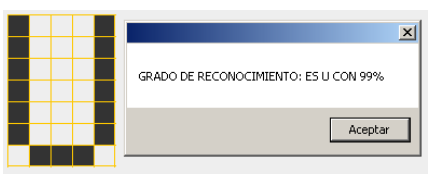


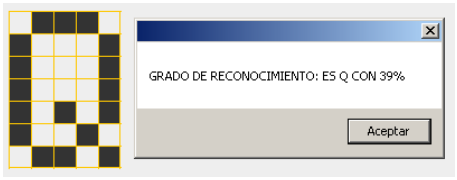


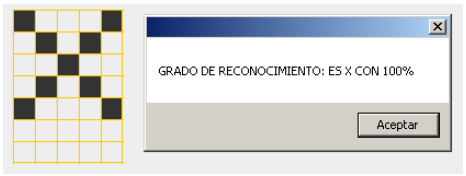


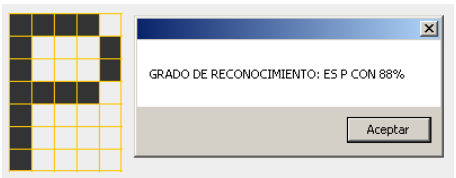


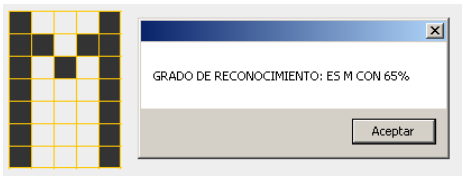
3º PRUEBA DE LA RED NEURONAL ARTIFICIAL HOPFIELD




GRADO DE RECONOCIMIENTO: ES Q CON 39%
Aceptar


GRADO DE RECONOCIMIENTO: ES X CON 100%
Aceptar


GRADO DE RECONOCIMIENTO: ES P CON 88%
Aceptar


GRADO DE RECONOCIMIENTO: ES M CON 65%
Aceptar

CAPÍTULO III

CONCORDANCIA DE RESULTADOS Y CONCLUSIONES

II.1 CORCONDANCIA DE RESULTADOS Y OBJETIVOS

Los algoritmos implementados presentan un grado de similitud considerable para el reconocimiento de caracteres del idioma español quedando el algoritmo Perceptrón Multicapa como el más aplicable en respuesta al objetivo general de este trabajo.

La red neuronal artificial Perceptrón Multicapa alcanzó un promedio de 79,57% en reconocimiento de caracteres del idioma español en comparación a las demás redes neuronales artificiales que fueron de la red neuronal Backpropagation 78,23%, la red neuronal Madaline obtuvo 77,77% y finalmente la red neuronal Hopfield consiguió 72.08%

II.2 CONCLUSIONES Y TRABAJOS FUTUROS

En el análisis obtenido de todos las letras, se pudo dar cuenta que existe un problema N, ya que al momento de entrenar esa letra se obtuvo como resultado de un 30% el cual es bajo y representa un problema de incompatibilidad de caracteres, porque en el panel de resultados existe conflicto con la letra Ñ.

El problema existente se debe a que la estructura de la letra N con la letra Ñ es similar, lo cual presenta problemas al momento de realizar la fase de entrenamiento de la red, porque existe confusión de caracteres y por ende el resultado de probabilidad es bajo lo cual representa un resultado ambiguo.

El resultado es óptimo y sus resultados son precisos, cuando están en el rango de aceptación comprendido entre 43% - 83%.

Otro de los resultados que se analizó es el de la letra Q, puesto que, al igual que el caso anterior, ésta presenta conflicto con la letra O ya que su estructura es similar.

Al momento re alizar la fase de entrenamiento se obtuvo un resultado del 13% siendo un resultado bajo, igual al caso anterior.

Pese a que la letra Q difiere de la O, su estructura es similar y el resultado va a dar lugar a ambigüedades.

En conclusión la red neuronal Perceptrón Multicapa es la red con mayor grado de reconocimiento de caracteres por su capacidad para la generalización es decir su facilidad para las salidas satisfactorias a entradas que el sistema nunca vio en la fase de entrenamiento está característica le da versatilidad para que la red neuronal pueda ser implementada en otras aplicaciones mas complejas.

Se requiere un numero de neuronas que optimicen el tiempo que le toma a la red realizar la clasificación se pude recurrir a una matriz de resultados obtenidos después del entrenamiento.

Todos los demás caracteres analizados al momento de pasar por la fase de entrenamiento y mostrar sus resultados finales la probabilidad del resultado final comprenden un rango de nivel de aceptación comprendido entre 75% - 100% lo cual representa un resultado óptimo de la imagen del carácter que se está analizando.

En cuanto a los resultados obtenidos con relación tiempo de ejecución, el leguaje que obtuvo menor tiempo fue Java con 0,139 milisegundos, Visual Basic alcanzó 0,166 milisegundos y por último C# obtuvo 0,269.

Si se realiza un aprendizaje de patrones de validación sin ruido los resultados nos muestran que el error en el aprendizaje disminuye de manera más rápida.

Si se realiza un aprendizaje con patrones de validación con ruido, el error en el aprendizaje disminuye más lentamente.

A mayor cantidad de ciclos de aprendizaje, el error del mismo es el mínimo.

La utilización de modelos de redes neuronales para soporte a la toma de decisiones es posible.

El algoritmo Perceptrón es más eficiente, pese a que calcula el error cuadrático medio y realiza un proceso de balanceo de carga entre los nodos de la red neuronal, lo que no da un resultado en un mejor tiempo que Hopfield, Madaline y Backpropagation.

Una aplicación de este tipo debe ser considerada como una herramienta más para ser utilizada por la persona encargada de tomar decisiones, no es posible denegar la toma de decisiones completamente a la aplicación ya que existe factores subjetivos e implícitos.

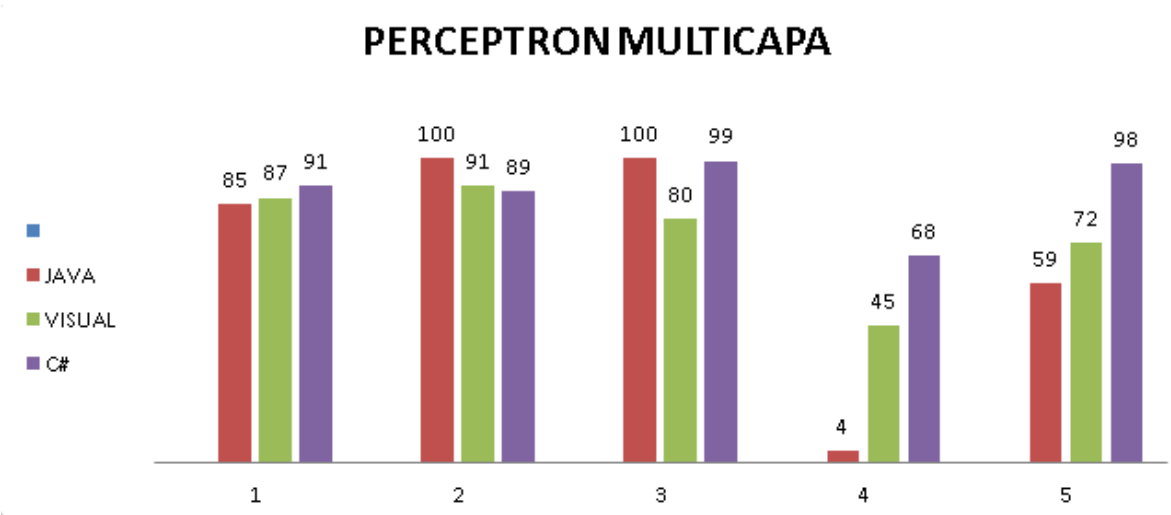
Dado el tiempo relativamente corto que emplea la red para aprender, es recomendable reentrenarla periódicamente.

A continuación, presentan los valores obtenidos en las salidas de las redes neuronales artificiales, para todas las letras mostradas en la simulación.

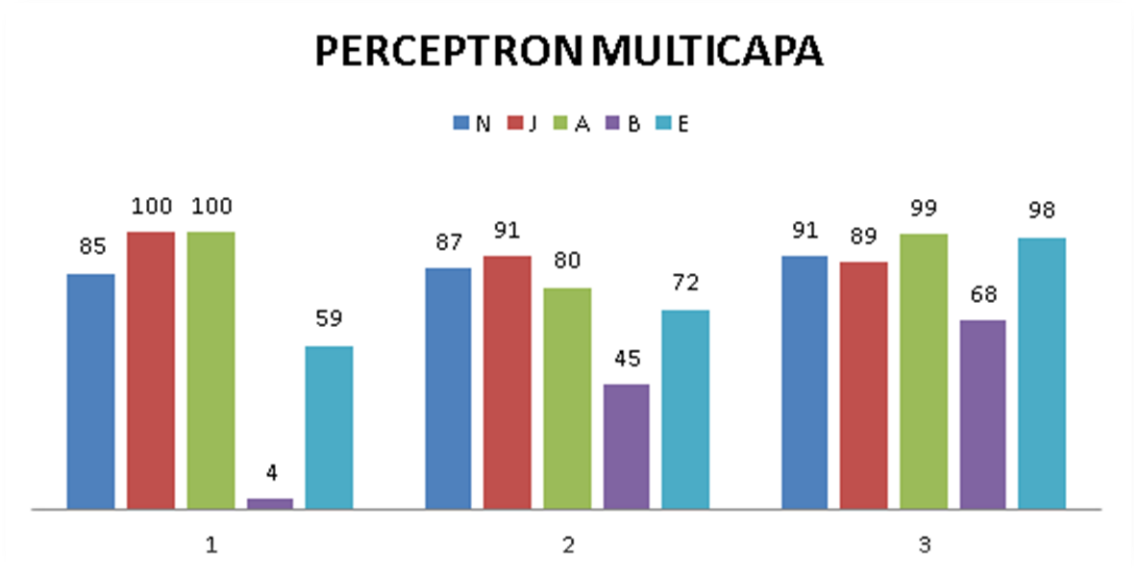
RESULTADOS OBTENIDOS DE LA SIMULACION DE LAS REDES NEURONALES ARTIFICIALES EN EL LEGUAJE JAVA

1º PRUEBA DE RED NEURONAL PERCEPTRÓN MULTICAPA

%	N	J	A	B	E
JAVA	85	100	100	4	59
VISUAL	87	91	80	45	72
C#	91	89	99	68	98



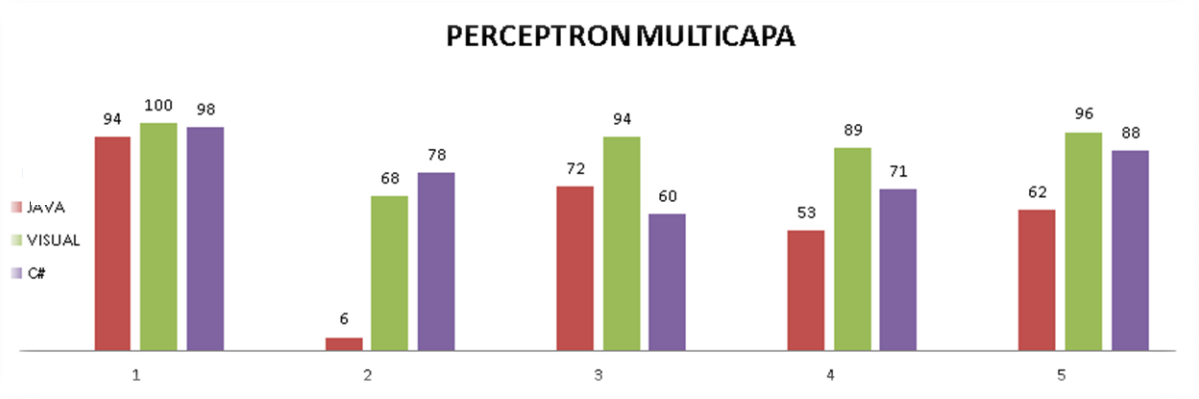
Resultados obtenidos de la primera fase de entrenamiento de la red Perceptrón Multicapa



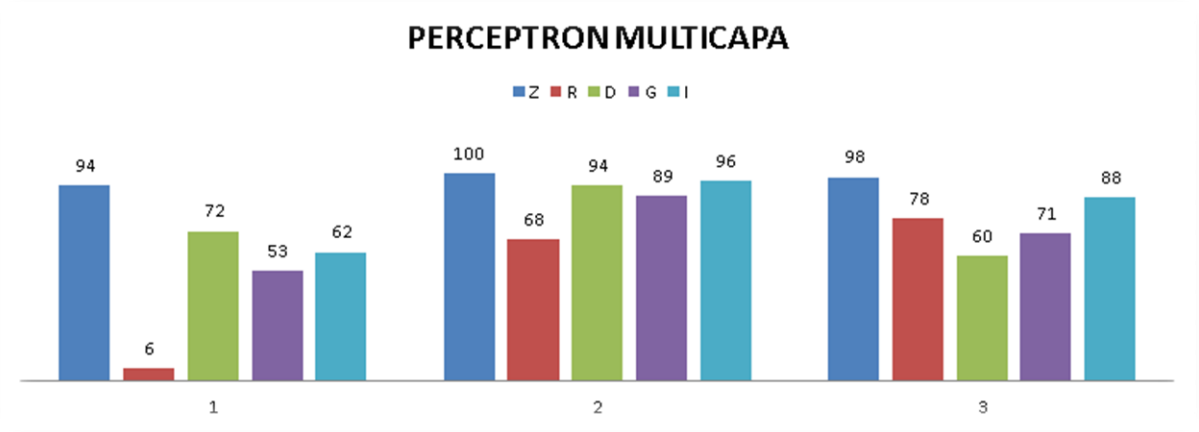
Resultados según la letra con la que se realizo el entrenamiento

2º PRUEBA DE RED NEURONAL PERCEPTRÓN MULTICAPA

	Z	R	D	G	I
JAVA	94	6	72	53	62
VISUAL	100	68	94	89	96
C#	98	78	60	71	88



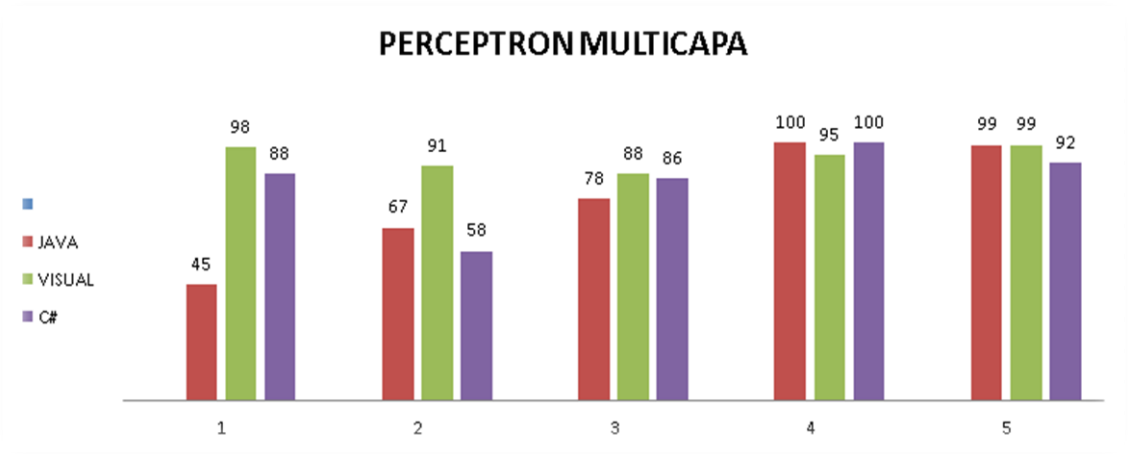
Resultados obtenidos de la segunda fase de entrenamiento de la red Perceptrón Multicapa



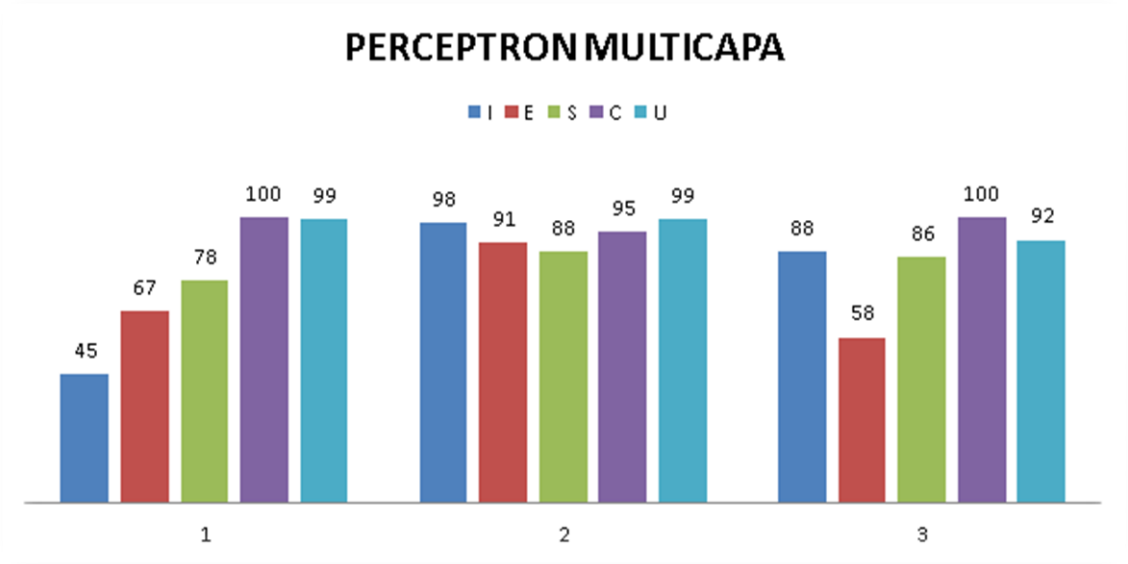
Resultados según la letra con la que se realizo el entrenamiento

3º PRUEBA DE LA RED PERCEPTRÓN MULTICAPA

	I	E	S	C	U
JAVA	45	67	78	100	99
VISUAL	98	91	88	95	99
C#	88	58	86	100	92



Resultados obtenidos de la tercera fase de entrenamiento de la red Perceptrón Multicapa

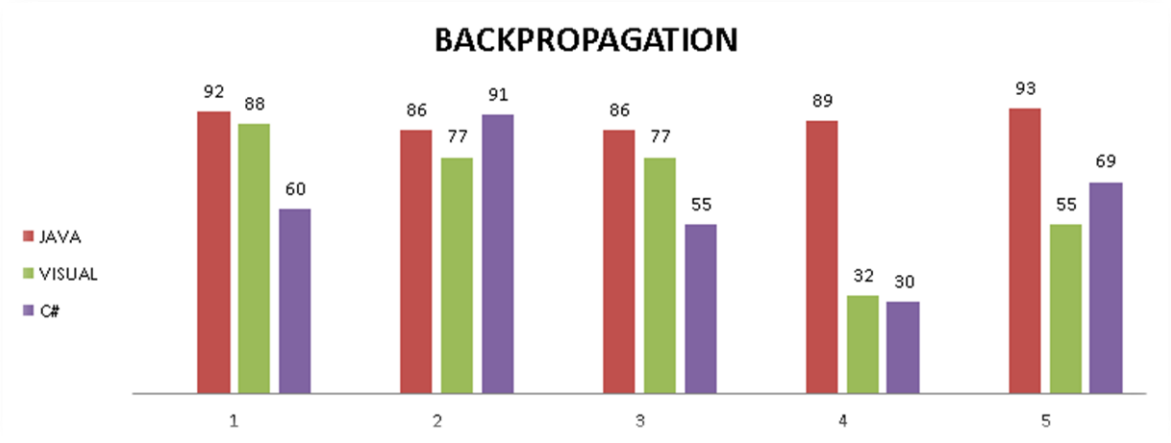


Resultados según la letra con la que se realizo el entrenamiento

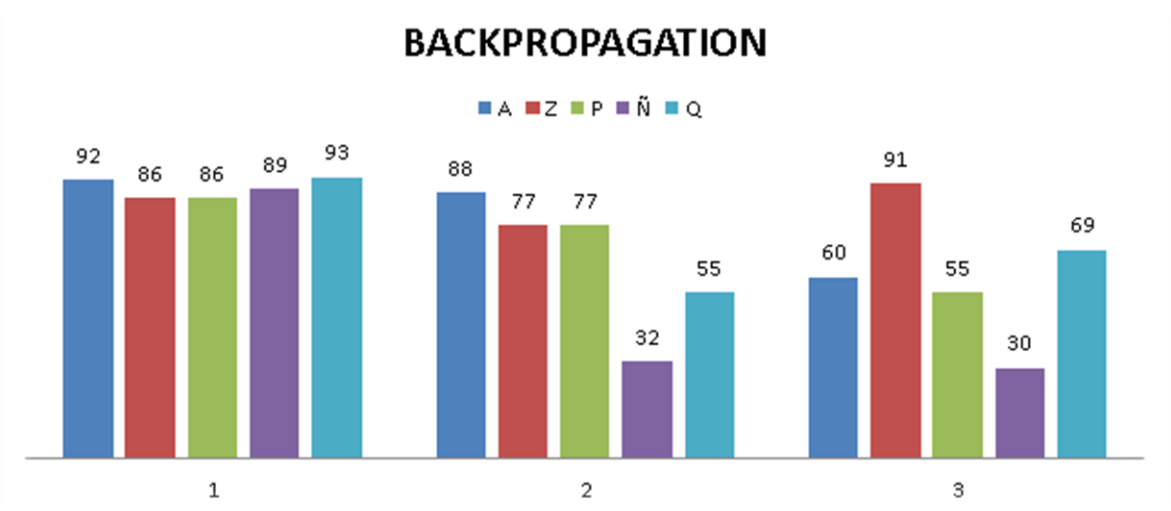
1º PRUEBA DE LA RED BACKPROPAGATION

	A	Z	P	Ñ	Q
JAVA	92	86	86	89	93
VISUAL	88	77	77	32	55

C#	60	91	55	30	69



Resultados obtenidos de la primera fase de entrenamiento de la red Backpropagation

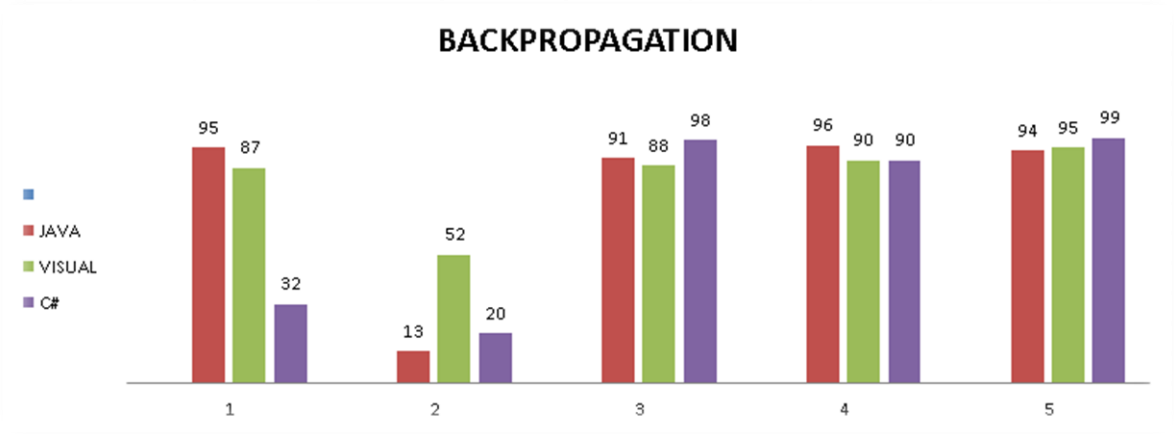


Resultados según la letra con la que se realizo el entrenamiento

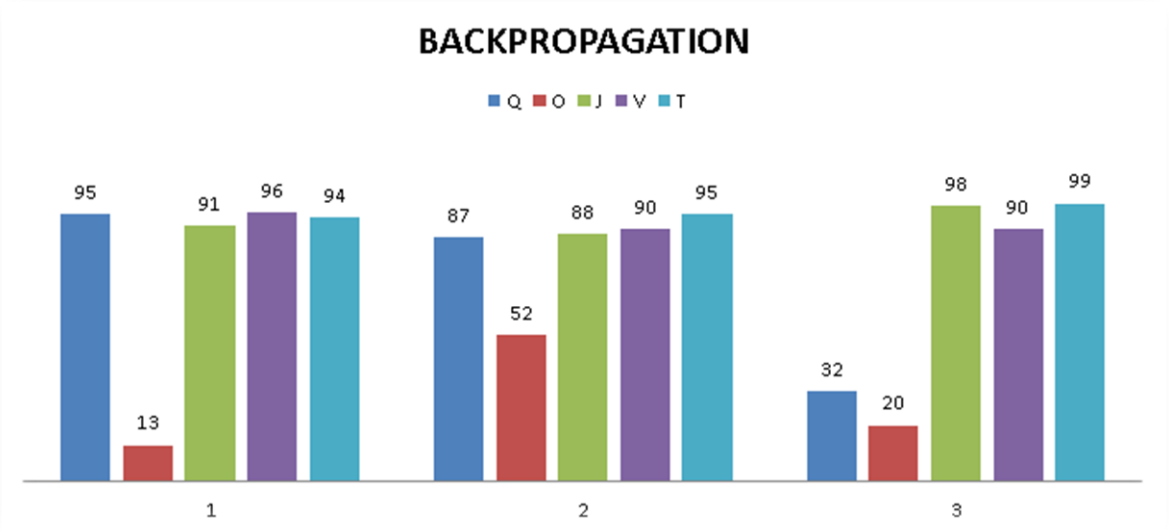
2º PRUEBA DE LA RED BACKPROPAGATION

	Q	O	J	V	T
JAVA	95	13	91	96	94
VISUAL	87	52	88	90	95

C#	32	20	98	90	99



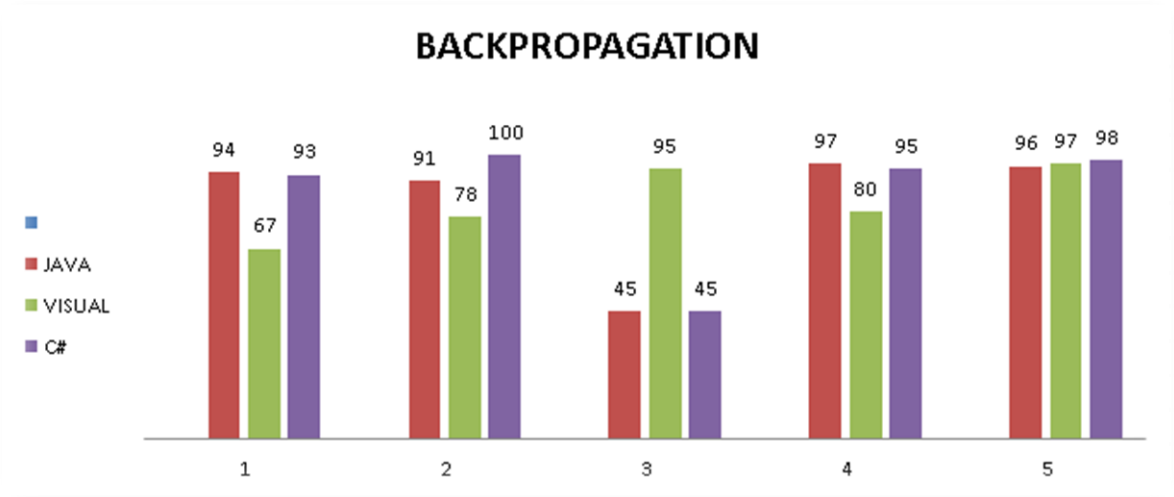
Resultados obtenidos de la segunda fase de entrenamiento de la red Backpropagation



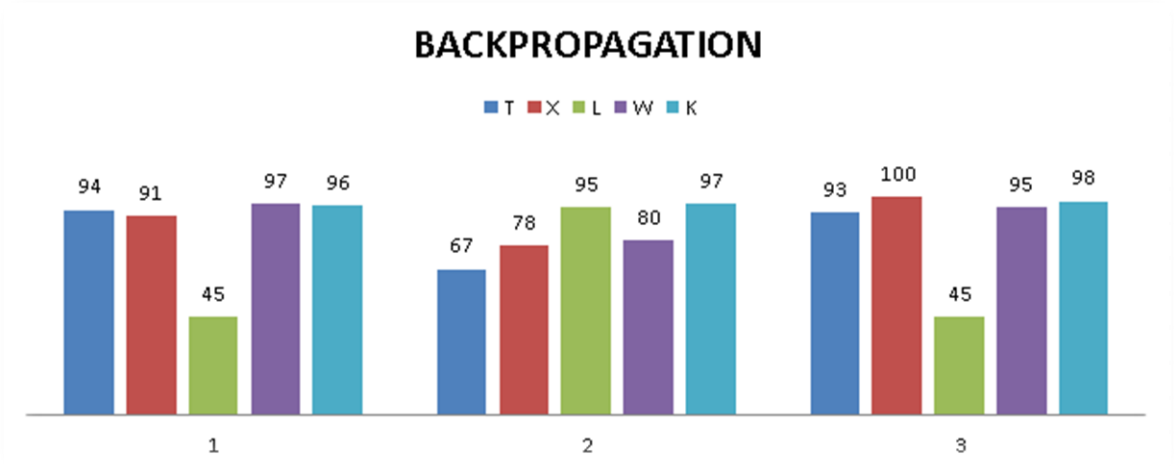
Resultados según la letra con la que se realizo el entrenamiento

3º PRUEBA DE LA RED BACKPROPAGATION

	T	X	L	W	K
JAVA	94	91	45	97	96
VISUAL	67	78	95	80	97
C#	93	100	45	95	98



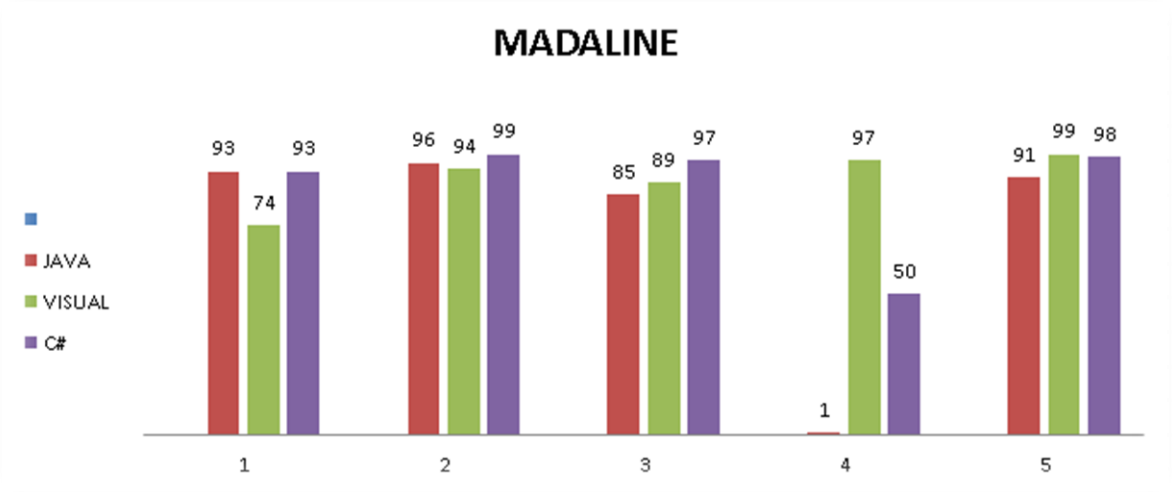
Resultados obtenidos de la tercera fase de entrenamiento de la red Backpropagation



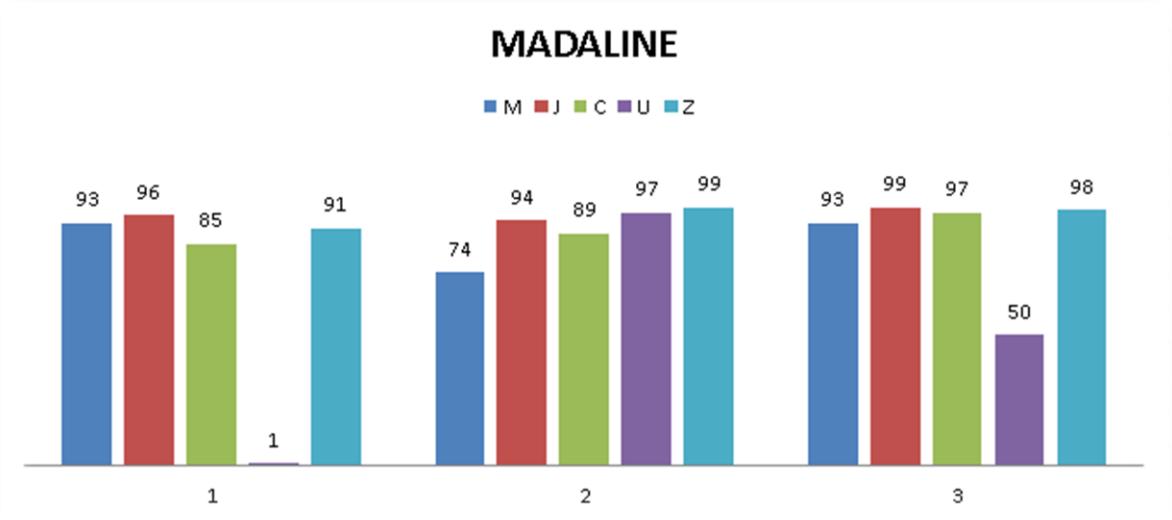
Resultados según la letra con la que se realizo el entrenamiento

1º PRUEBA DE LA RED MADALINE

	M	J	C	U	Z
JAVA	93	96	85	100	91
VISUAL	74	94	89	97	99
C#	93	99	97	50	98



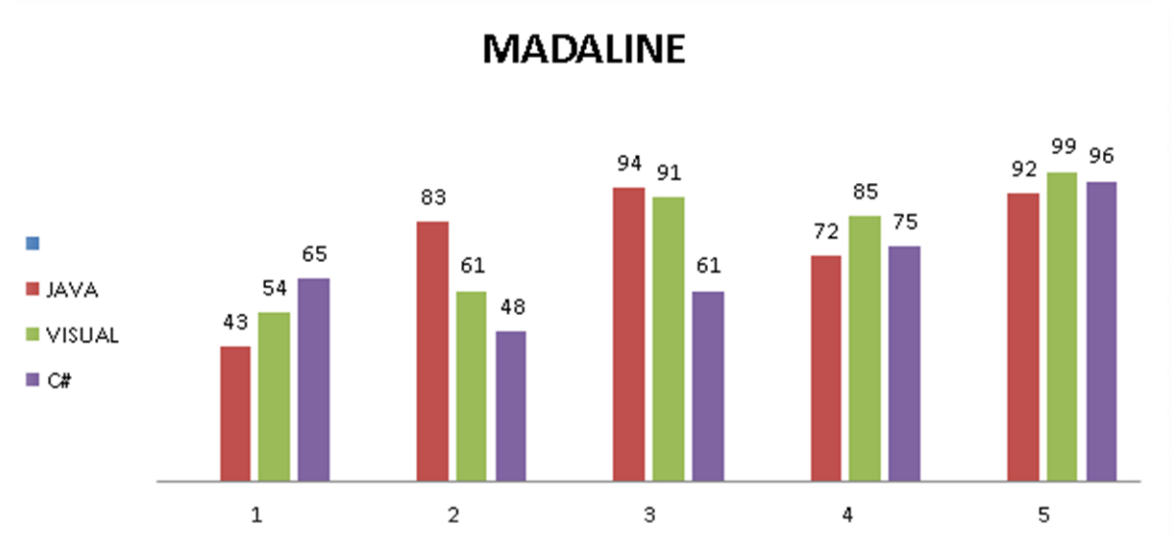
Resultados obtenidos de la primera fase de entrenamiento de la red Madaline



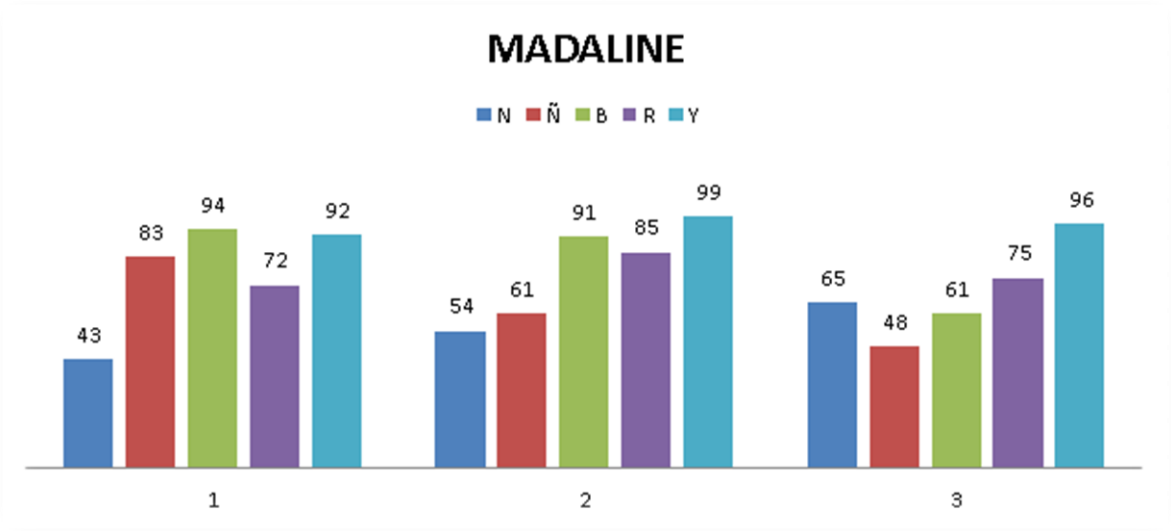
Resultados según la letra con la que se realizo el entrenamiento

2º PRUEBA DE LA RED MADALINE

	N	Ñ	B	R	Y
JAVA	43	83	94	72	92
VISUAL	54	61	91	85	99
C#	65	48	61	75	96



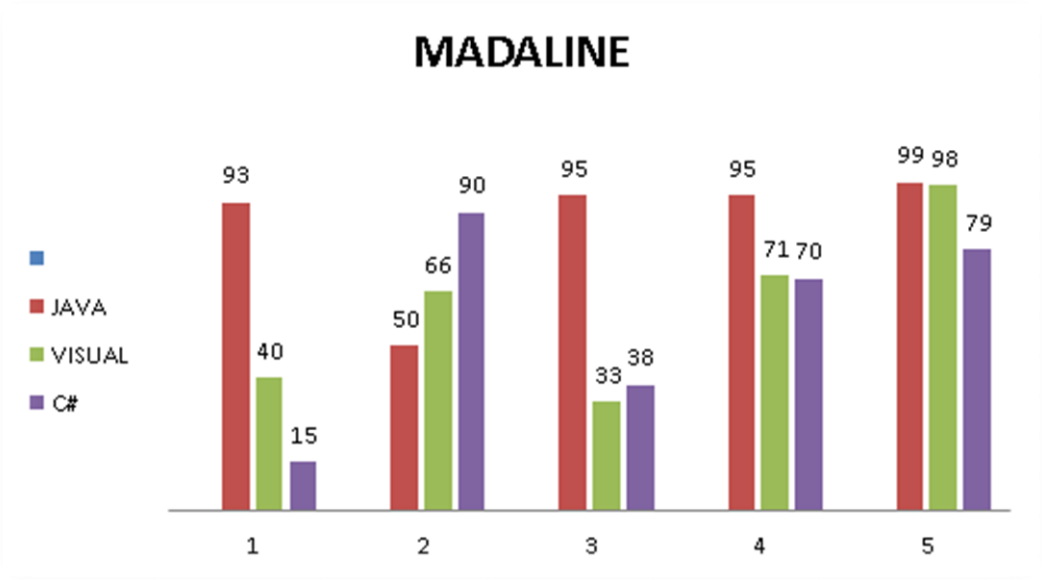
Resultados obtenidos de la segunda fase de entrenamiento de la red Madaline



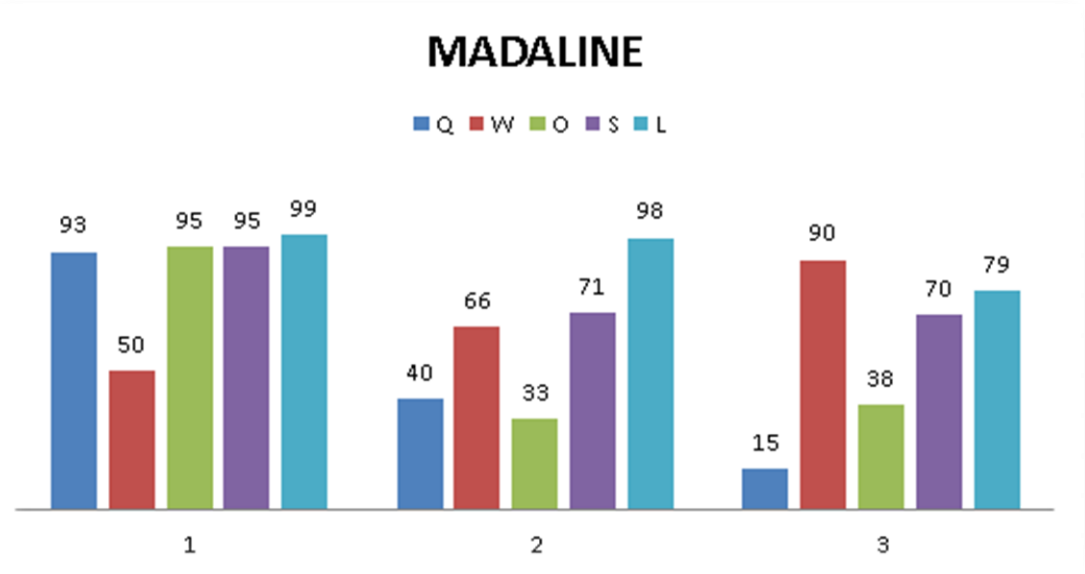
Resultados según la letra con la que se realizo el entrenamiento

3º PRUEBA DE LA RED MADALINE

	Q	W	O	S	L
JAVA	93	50	95	95	99
VISUAL	40	66	33	71	98
C#	15	90	38	70	79



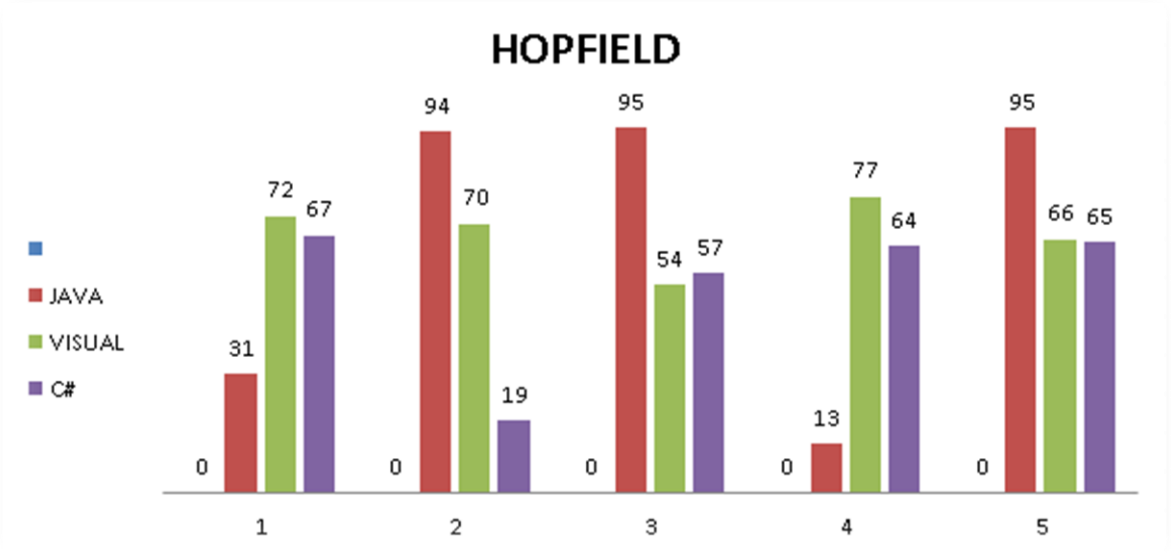
Resultados obtenidos de la tercera fase de entrenamiento de la red Madaline



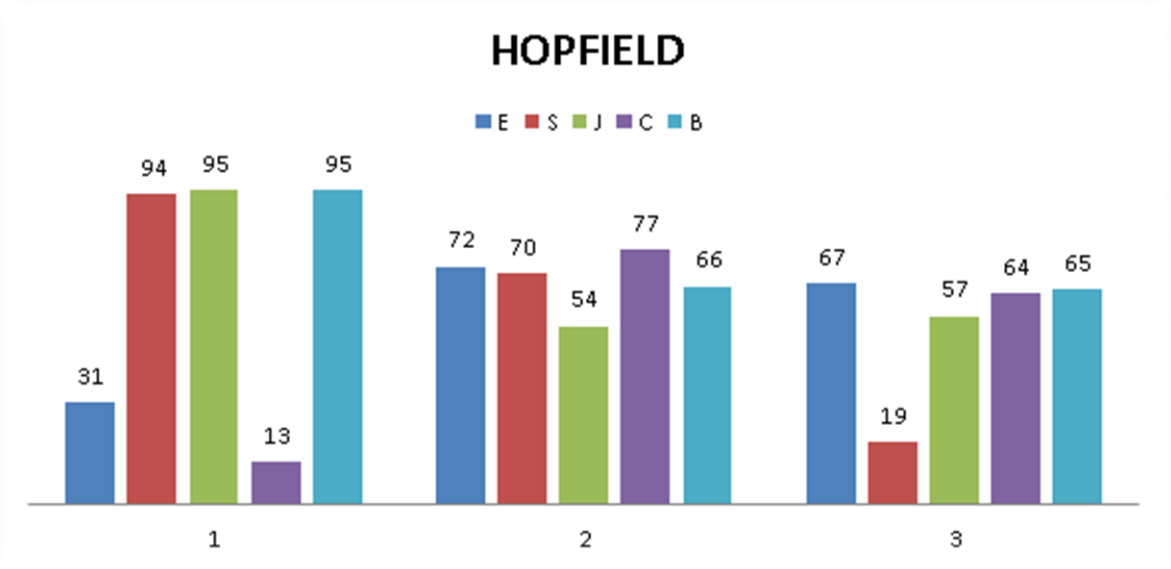
Resultados según la letra con la que se realizo el entrenamiento

1º PRUEBA DE LA RED HOPFIELD

	E	S	J	C	B
JAVA	31	94	95	13	95
VISUAL	72	70	54	77	66
C#	67	19	57	64	65



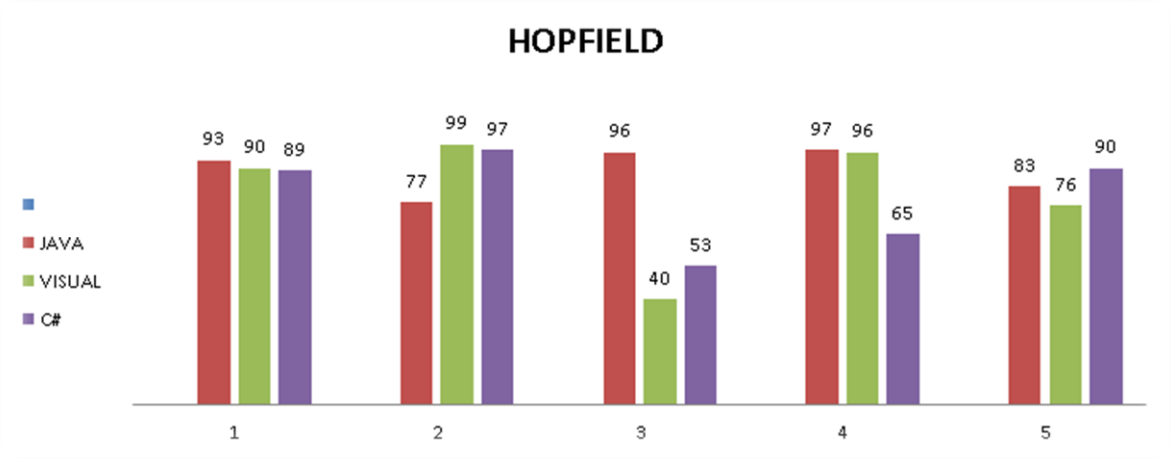
Resultados obtenidos de la primera fase de entrenamiento de la red Hopfield



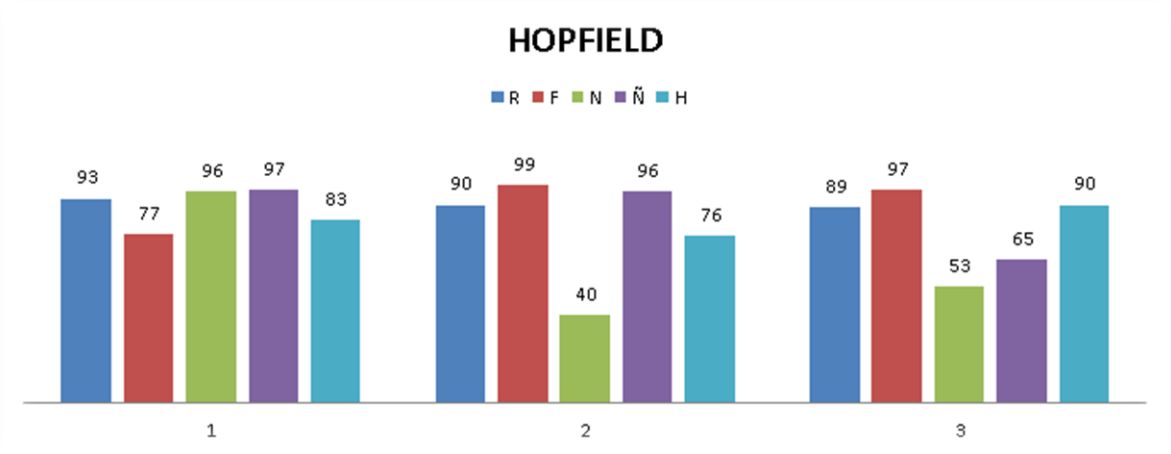
Resultados según la letra con la que se realizo el entrenamiento

2º PRUEBA DE LA RED DE HOPFIELD

	R	F	N	Ñ	H
JAVA	93	77	96	97	83
VISUAL	90	99	40	96	76
C#	89	97	53	65	90



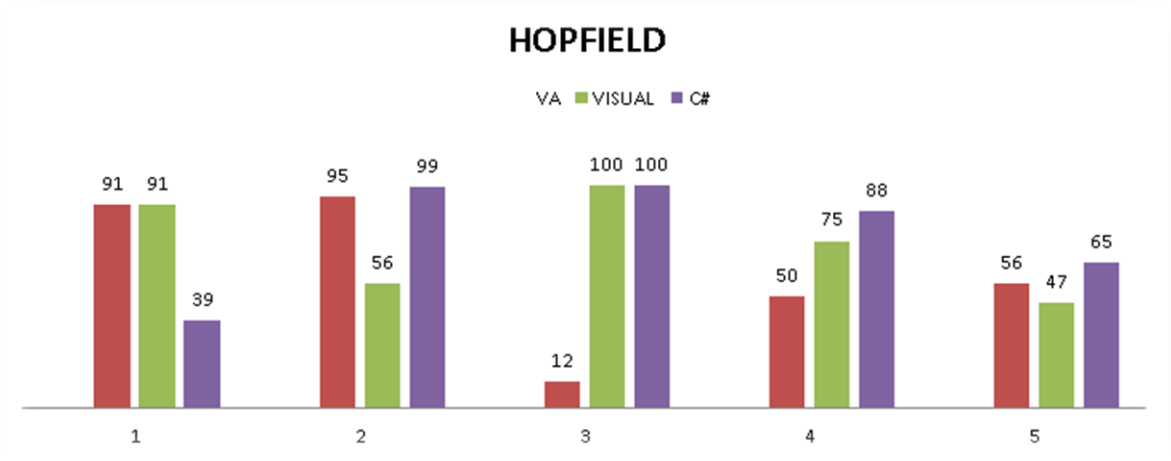
Resultados obtenidos de la segunda fase de entrenamiento de la red Hopfield



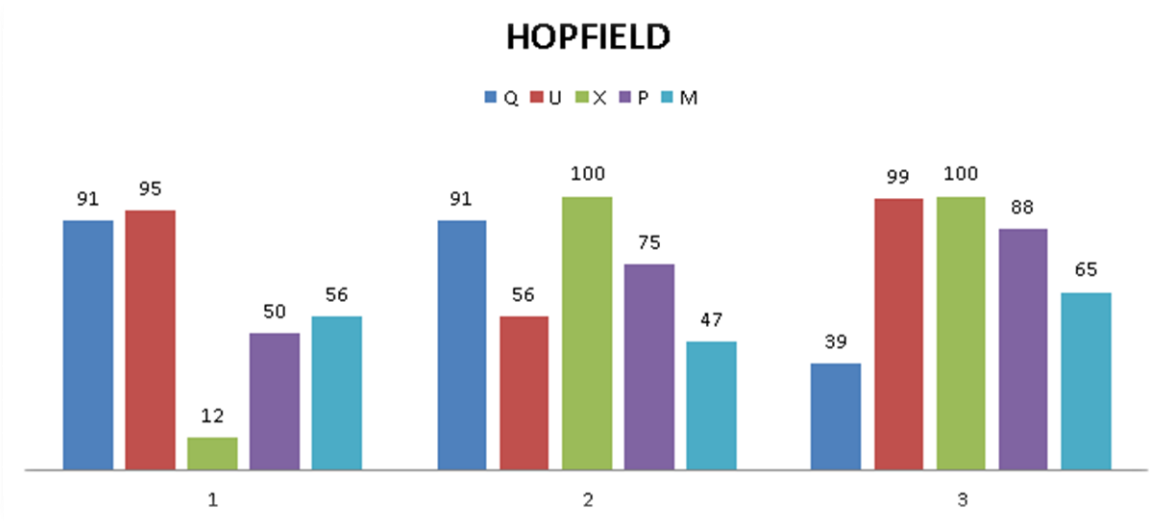
Resultados según la letra con la que se realizo el entrenamiento

3º PRUEBA DE LA RED HOPFIELD

	Q	U	X	P	M
JAVA	91	95	12	50	56
VISUAL	91	56	100	75	47
C#	39	99	100	88	65



Resultados obtenidos de la tercera fase de entrenamiento de la red Hopfield



Resultados según la letra con la que se realizo el entrenamiento

Como trabajos futuros, considero interesante la posibilidad de aplicar la arquitectura de la red neuronal Perceptrón Multicapa a la resolución de otros problemas reconocimiento de patrones, para observar la adaptabilidad de clasificador. Por otro lado, y con el objeto de mejorar el performance del sistema reconocedor de caracteres, planteo la incorporación de nuevos mapas de características que permitan definir los números que presentan mayores problemas de reconocimiento. Sería interesante ver qué ocurre con el comportamiento del sistema si se agregan, por ejemplo asociadas a características tales como cantidad de vértices, curvas, aberturas, entre otras presentes en cada dígito a

tratar. Otra cuestión asociada con esto último y que podría ser investigada, es la técnica que utiliza el módulo analizador para decidir las respuestas, más aún, si éste se ve obligado a manejar un gran número de votos. Considero la incorporación en el diseño de varias características representativas, tratada de forma independiente y luego integradas en la característica global y a través del módulo analizador, es muy beneficioso ya que no permite que la presencia de errores en los patrones de entrada no tenga influencia en la respuesta del sistema.

Y también como trabajo futuro se podría realizar la implementación de una aplicación de reconocimiento de caracteres para las tabletas Wacom. La aplicación de reconocimiento de caracteres se podría partir de la digitalización de la escritura por un explorador óptico después que el usuario termine de escribir.