| **escudocolor** | **UNIVERSIDAD DE COSTA RICA**  **FACULTAD DE INGENIERÍA**  **ESCUELA DE CIENCIAS DE LA**  **COMPUTACION E INFORMÁTICA** |
| --- | --- |

**CI1441 – PARADIGMAS COMPUTACIONALES**

**Grupo 2**

**Prof. Yadira Solano Sabatier**

**INFORME 1 PROYECTO PRÁCTICO**

**TEMA:**

**COMPARACIÓN DE LAS REDES NEURONALES DE TIPO RETROPROPAGACIÓN Y CONTRAPROPAGACIÓN EN TÉRMINOS DE EFICIENCIA Y RENDIMIENTO*, AL SER UTILIZADO EN EL RECONOCIMIENTO DE FIGURAS GEOMETRICAS SIMPLES.***

**Elaborado por:**

**Emerson Alvarado Matamoros – A40235**

**Ricardo Alvarado Villalobos – A60289**

**II-2010**

Tabla de Contenidos

[Introducción 4](#_Toc277020506)

[Objetivos del proyecto 4](#_Toc277020507)

[Objetivo general 4](#_Toc277020508)

[Objetivos específicos 4](#_Toc277020509)

[Marco Teórico 5](#_Toc277020510)

[Redes Neuronales Artificiales: 5](#_Toc277020511)

[Origen: 5](#_Toc277020512)

[Elementos de una Red Neuronal Artificial: 5](#_Toc277020513)

[Estructura de una Red Neuronal Artificial 6](#_Toc277020514)

[Justificación del proyecto 7](#_Toc277020515)

[Resumen de los paradigmas a utilizar 8](#_Toc277020516)

[Descripción de la aplicación para la que se desarrollara el proyecto 9](#_Toc277020517)

[Estructura del sistema 10](#_Toc277020518)

[Red neuronal de retropropagación: 11](#_Toc277020519)

[Red neuronal de contrapropagación: 12](#_Toc277020520)

[Metodología 13](#_Toc277020521)

[Evaluación de los resultados 15](#_Toc277020522)

[Pruebas para la figura circulo: 16](#_Toc277020523)

[Pruebas para la figura triangulo: 18](#_Toc277020524)

[Pruebas para la figura cuadrado: 20](#_Toc277020525)

[Pruebas para la figura pentágono: 22](#_Toc277020526)

[Pruebas para la figura hexágono: 24](#_Toc277020527)

[Conclusiones 26](#_Toc277020528)

[Manual de usuario 28](#_Toc277020529)

[Entrenamiento de la red de Retropropagacion 28](#_Toc277020530)

[Entrenamiento de la red de Contrapropagacion 34](#_Toc277020531)

[Creación de muestras 40](#_Toc277020532)

[Ejecución 44](#_Toc277020533)

[Salir 47](#_Toc277020534)

[Acerca de 48](#_Toc277020535)

[Observaciones 49](#_Toc277020536)

[Formato de los archivos 49](#_Toc277020537)

[Referencias bibliográficas 51](#_Toc277020538)

[Apéndice 1: 54](#_Toc277020539)

[Apéndice 2: 55](#_Toc277020540)

[Apéndice 3: 56](#_Toc277020541)

# Introducción

Las redes neuronales son una de las aéreas de aplicación de la Inteligencia Artificial que más se ha desarrollado en los últimos tiempos. Es gracias a ellas que muchos problemas han llegado a soluciones cuando antes eran considerados intratables, abriendo todo un nuevo paradigma y una visión de los problemas distinta a la que se tenía antes de su aparición.

Dado que la IA es un área que está en constante evolución y desarrollo, es común ver como surgen nuevas y diferentes estructuras y modelos para la resolución de problemas, y en el caso de las redes neuronales nuevos tipos de redes, con algoritmos y modos de aprendizaje diferentes a los existentes.

Todo esto nos lleva a pensar: ¿cuáles son las ventajas de unas redes sobre otras? ¿existen realmente grandes diferencias a la hora de enfrentar un problema específico? Esta y otras interrogantes son las que nos motivan a realizar esta investigación, en la cual pondremos a prueba una red “tradicional” como lo es la red de retropropagación, con un entrenamiento supervisado, y una red de contrapropagación, de entrenamiento hibrido, en un problema de reconocimiento de patrones, y analizaremos su rendimiento, tanto en su fase de entrenamiento como en el reconocimiento de los mismos, en aéreas como el tiempo y la eficiencia en la solución del problema de las mismas.

# Objetivos del proyecto

### Objetivo general

* Realizar un análisis de rendimiento y eficiencia entre las redes neuronales artificiales de tipo retropropagación (backpropagation) y contrapropagación (counterpropagation).

### Objetivos específicos

* Comprender la importancia de la utilización de un paradigma alterno, diferente a los tradicionales, para la resolución de un problema especifico.
* Implementar y comprender el funcionamiento tanto de una red neuronal de tipo retropropagación como el de una de tipo contrapropagación, sus algoritmos, diferencias y similitudes en implementación y diseño de capas.
* Realizar el entrenamiento de las redes neuronales para que sean capaces de reconocer figuras geométricas simples.
* Analizar la eficacia y rendimiento de ambas redes para la resolución del problema específico.

# Marco Teórico

### Redes Neuronales Artificiales:

### Origen:

Las redes neuronales artificiales, como su nombre lo indica pretenden imitar la forma de funcionamiento de las neuronas biológicas que constituyen el sistema nervioso humano. Todo el desarrollo de las redes neuronales tiene mucho que ver con la neurofisiología: no es por casualidad que se trata de imitar a una neurona humana con la mayor exactitud posible.

Entre los impulsores del modelado de neuronas se encuentra Warren McCulloch y Walter Pitts. Estos dos investigadores propusieron un modelo matemático de neurona artificial, en el que cada neurona estaba dotada de un conjunto de entradas y salidas, cada entrada estaba afectada por un peso, la activación de la neurona se calcula mediante la suma de los productos de cada entrada y la salida resultante es una función de activación. Sin embargo, no fue hasta que Rosenblatt propuso su modelo el perceptrón que las redes se tornaron interesantes, pues la principal clave de este sistema se encuentra en los pesos de las diferentes entradas: Las entradas son modificadas por el peso y las salidas se dan en función de estas modificaciones, por lo que podemos ver que los pesos influyen de forma decisiva en la salida y por lo tanto pueden ser utilizados para controlar la salida que se desea, logrando que la red “aprenda”.

### Elementos de una Red Neuronal Artificial:

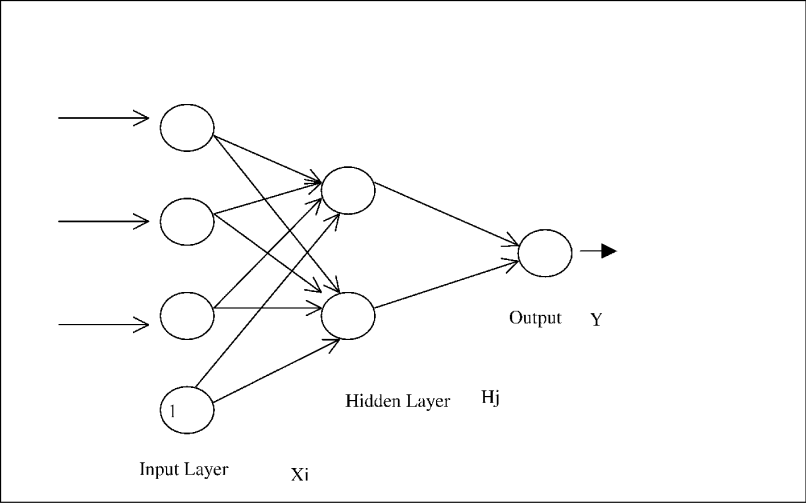
Las RNA actuales constan de tres elementos básicos:

1. Unidad de proceso: La neurona Artificial. Existen tres tipos de neuronas en cualquier sistema: neuronas de entrada, neuronas de salida y neuronas ocultas. Las unidades de entrada reciben señales desde el entorno; las de salida envían la señal fuera de la red, y las unidades ocultas son aquellas cuyas entradas y salidas se encuentran dentro del sistema.
2. Estado de Activación: Los estados del sistema en un [tiempo](http://www.monografias.com/trabajos901/evolucion-historica-concepciones-tiempo/evolucion-historica-concepciones-tiempo.shtml) t se representan por un vector A(t). [Los valores](http://www.monografias.com/trabajos14/nuevmicro/nuevmicro.shtml) de activación pueden ser continuos o discretos, limitados o ilimitados. Si son discretos, suelen tomar un conjunto discreto de [valores](http://www.monografias.com/trabajos14/nuevmicro/nuevmicro.shtml) binarios, así un [estado](http://www.monografias.com/trabajos12/elorigest/elorigest.shtml) activo se indicaría con un 1 y un estado pasivo se representaría por un cero. En otros modelos se considera un conjunto de estados de activación, en cuyo [valor](http://www.monografias.com/trabajos14/nuevmicro/nuevmicro.shtml) entre [0,1], o en el intervalo [-1,1], siendo una [función](http://www.monografias.com/trabajos7/mafu/mafu.shtml) sigmoidal.
3. Función de Salida o de Transferencia: Es una función asociada la salida de cada unidad, que transforma [el estado](http://www.monografias.com/trabajos12/elorigest/elorigest.shtml) actual de activación en una señal de salida. Las más conocidas son:
   1. Función Escalón
   2. Función Lineal y Mixta
   3. Sigmoidal
   4. Función Gaussiana

### Estructura de una Red Neuronal Artificial

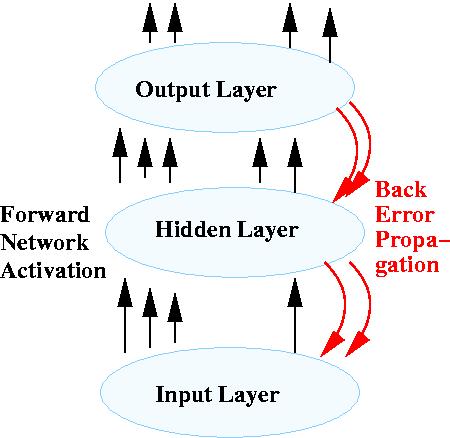
Como fue mencionado anteriormente, la distribución de neuronas dentro de la red se realiza formando niveles o capas de un número determinado de neuronas cada una, y existen capas de entrada, de salida, y ocultas. Sabiendo esto, podemos analizar las dos formas básicas de conexión entre neuronas:

Cuando ninguna salida de las neuronas es entrada de neuronas de su mismo nivel o de niveles anteriores, la red se describe como de propagación hacia adelante:



Ejemplo de 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 se considera como de propagación hacia atrás.



Ejemplo de propagación hacia atrás

# Justificación del proyecto

Las redes neuronales representan uno de los paradigmas más importantes de la inteligencia artificial. En la actualidad podemos ver como en muchos y diferentes campos se aplican para resolver gran variedad de problemas que de otra manera seria mucho más difícil resolver, y tal vez no con la misma eficiencia ni el mismo grado de satisfacción de la solución.

Al mismo tiempo, al ser un paradigma que simula el funcionamiento de las redes neuronales biológicas, con los nuevos descubrimientos médicos también surgen nuevas formas de este paradigma, lo que lleva a tener una gran cantidad de implementaciones, que funcionan de formas diferentes, y tienen diferentes rendimientos según su uso.

Es debido a esta cantidad de implementaciones del paradigma que consideramos interesante la comparación de dos de ellos en la resolución de un problema especifico, de forma que sea posible comparar su rendimiento, eficiencia y resultados, pudiendo tener estadísticas útiles para la consideración en el momento de decidir qué tipo de red usar para ciertos problemas, y observar el comportamiento y las diferencias y similitudes que se puedan presentar entre ambas redes.

Creemos que siempre es bueno tener referencias al momento de realizar una selección, y consideramos importante crear una referencia para estas dos redes, que puede resultar muy útil para otras personas en el futuro, y aumenta el conocimiento de un tema tan amplio como lo es la inteligencia artificial.

# Hipótesis

Según lo leído en la teoría de redes neuronales, tenemos las siguientes hipótesis:

# Resumen de los paradigmas a utilizar

Los paradigmas que usaremos para el proyecto serán una red neuronal artificial de contrapropagación, y una red neuronal de retro propagación.

Una red neuronal es una simulación de las propiedades que existen en los sistemas neuronales biológicos, recreándolos de forma artificial, en este caso en un programa computacional. Como toda red neuronal biológica, su unidad central es la neurona, que en este caso se llama perceptrón. Cada uno de ellos recibe una serie de entradas a través de interconexiones y emite una salida. Esta salida viene dada por tres funciones: Una función de propagación, que consiste en la sumatoria de cada entrada del perceptrón multiplicada por el peso de su interconexión. Una función de activación, que modifica a la de propagación, y una función de transferencia, que se encarga de acotar la salida de la neurona, y viene dada por la interpretación que se le quiera dar a las salidas.

Al unir varios perceptrones en capas y sumar los estímulos de entrada es que tenemos una red neuronal. En el caso específico de una red de propagación hacia atrás, es necesario aclarar más a fondo este concepto: En este tipo de red, se interconectan varios perceptrones en capas, pero los perceptrones de cada capa no se interconectan entre sí. Sin embargo, cada perceptrón de una capa proporciona una entrada a cada uno de los perceptrones de la siguiente capa, Esto quiere decir que cada neurona transmitirá su señal de salida a cada neurona de la capa siguiente, lo cual es la arquitectura fundamental de una red de propagación hacia atrás.

En el caso de la red de contrapropagación, se interconectan varios perceptrones en capas, de forma que se forman 3 capas: una capa de entrada, una capa de salida y una capa oculta. La capa de entrada está constituida por 225 neuronas, que representan los 225 pixeles de la cuadricula de dibujo. En esta capa se realiza una normalización de forma que los datos de entrada estén en un rango de 0 a 1, utilizando la formula , resolviendo problemas como la saturación de las actividades neuronales frente a grandes señales de entrada. En cuanto a la capa oculta de la red, esta capa está compuesta por un conjunto de elementos de proceso conocidos como instars. Los cuales permiten a una neurona aprender un nuevo vector mediante el ajuste de sus pesos equivalente a la entrada que recibe en el momento con una razón proporcional a la salida de la neurona. El ajuste se realiza mediante la formula . De esta forma, después la presentación de la misma entrada causara que la instar tenga una salida significativa, lo que hará que el aprendizaje se incremente de nuevo, brindando una minimización del olvido. Por último, las capas de salida de esta red se llaman outstar, ya que están únicas con la capa oculta. Esta capa lo que hace es que identifica la clase de entrada que clasifico la capa de entrada, asociándole algún valor o identidad. Dado que para este proyecto solo hay 5 conjuntos de figuras diferentes, utilizaremos 5 neuronas instar y 5 outsar. De esta forma vemos como es muy diferente la implementación de ambas redes neuronales.

# Descripción de la aplicación para la que se desarrollara el proyecto

Se desarrollará una aplicación con la cual se reconocerán patrones figuras geométricas simples. Esto mediante el uso de dos redes neuronales diferentes como son la de tipo retropropagación y la de tipo contrapropagación.

Esto se logrará mediante la implementación de una interfaz en la cual el usuario disponga de un panel de dibujo en el cual podrá trazar una silueta que corresponda a una figura geométrica y se preguntará a la aplicación mediante algún elemento de interfaz, probablemente un botón, de que figura es la silueta que se ha trazado. Al final la aplicación mostrará en pantalla la solución, tanto de forma gráfica como de texto, según sea el reconocimiento realizado. Se planea trabajar con las figuras geométricas más básicas, analizando en el caso de los pentágonos, su número de lados. Deberá ser capaz de reconocer figuras geométricas como lo son triángulos, cuadriláteros, pentágonos, hexágonos y círculos.

Luego de ser dibujada la silueta por el usuario, la encargada de realizar el análisis sobre la figura será la red neuronal. Para lograr esto la red debe de ser entrenada de manera se le mostrarán diferentes maneras de dibujar cada una de las figuras geométricas para que esta logre determinar un patrón sobre la misma y poder reconocerla posteriormente. Para este entrenamiento se implementará una interfaz en la cual se utilizara un panel de dibujo en el cual se dibujaran figuras geométricas simples de varias maneras e indicándole el tipo al cual pertenece.

Este trabajo mencionado anteriormente se realizará de manera similar para ambas redes neuronales. Ya que además se pretende desarrollar una estructura dentro de la aplicación con la cual se logren calcular tanto los tiempos de aprendizaje como de ejecución, esto con la finalidad de obtener estadísticas necesarias para medir el rendimiento y de esta manera realizar una comparación entre las redes neuronales.

# Estructura del sistema

La estructura del sistema está basada en una interfaz principal desde la cual se accede a dos elementos importantes: la interfaz de aprendizaje y la interfaz para la utilización de la aplicación. La interfaz de aprendizaje nos proporciona las herramientas con las cuales entrenamos la red de forma supervisada, en la cual dibujamos una silueta de una figura geométrica y le especificamos de cual se trata. Esta será controlada por un objeto encargado de controlar el aprendizaje siendo intermediario entre la interacción en la interfaz y la red neuronal.

En cuanto a la interfaz para la ejecución principal de la aplicación por parte del usuario, presenta elementos en los cuales se puede dibujar una silueta y preguntar a la aplicación de qué tipo de figura se trata, para luego presentarnos la solución. Esta interfaz será controlada por un objeto intermedio de control de ejecución el cual se conecta con la red neuronal para obtener la solución de la ejecución.

Para una mayor comprensión de la estructura general del sistema se tiene el siguiente diagrama:

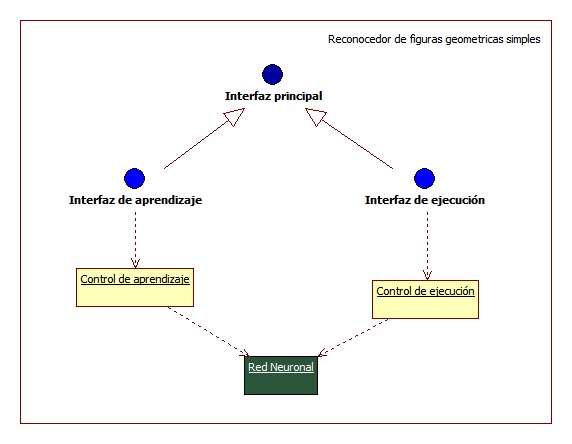


Fig. 1. Estructura general del sistema.

En cuanto a las redes neuronales describimos la arquitectura de ambas:

### Red neuronal de retropropagación:

La arquitectura de esta red neuronal artificial se encontrará compuesta por un capa de entrada la cual estará compuesta por una cantidad a definir de nodos, también tendremos una capa de salida la cual se compone de la misma cantidad de nodos. Y finalmente tenemos una cantidad de capas ocultas las cuales se encargan de procesar los datos mediante la utilización de nodos.

El siguiente diagrama nos muestra la estructura general de cómo se constituye una red neuronal de este tipo:

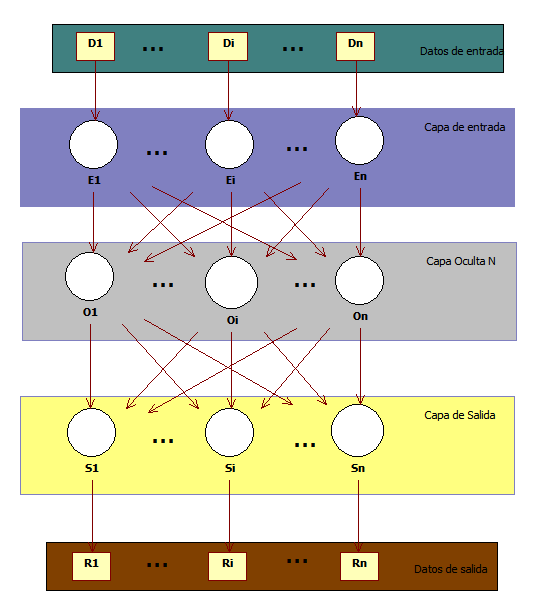


Fig. 2. Estructura general de una red de retropropagación

### Red neuronal de contrapropagación:

La arquitectura de esta red neuronal artificial se encontrará compuesta por 5 capas y 3 niveles de estas. En el primer nivel tenemos dos capas de entrada, en éste nivel ser normaliza el vector de entrada. De la misma manera en otro de los niveles tenemos dos capas de salida llamadas outstar, esta recupera el vector representativo basado en la clasificación de la capa oculta. Finalmente en el nivel medio tenemos la capa oculta la cual contiene un conjunto de elementos de proceso llamados instars.

El siguiente diagrama nos muestra la estructura general de cómo se constituye una red neuronal de este tipo:

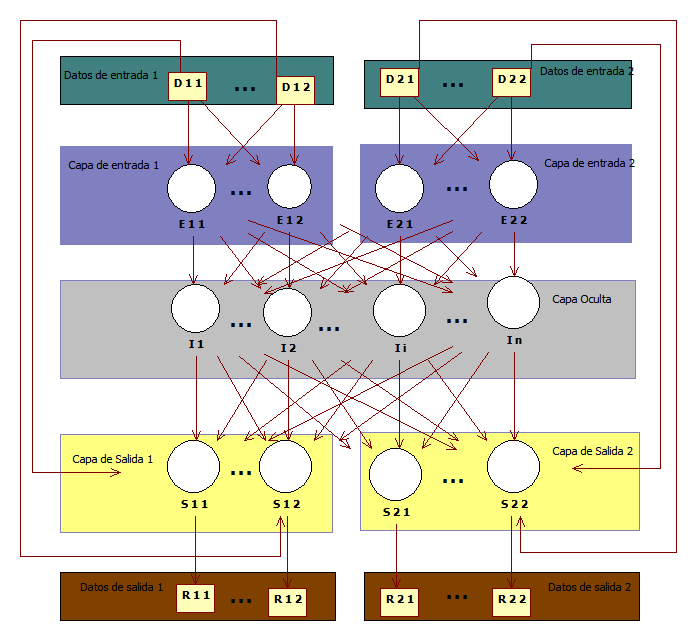


Fig. 3. Estructura general de una red de contrapropagación

# Metodología

Se trabajara en el proyecto de forma progresiva por etapas. En un inicio se realizara una etapa de investigación, ya que el paradigma con el que se va a trabajar es nuevo para ambos miembros del grupo de trabajo.

Una vez comprendido el paradigma, se procederá a trabajar en el diseño del producto, mediante iteraciones semanales de trabajo, en las cuales se procederá a la planificación y diseño de los componentes en una semana, y a las siguientes dos a su implementación y pruebas, mientras que de forma paralela se trabajara en el diseño e implementación de la red neuronal a utilizar. Dado que este es el paradigma nuevo, se le concederá un mayor lapso de tiempo para su implementación y pruebas, de modo que su acoplamiento al resto del sistema derive en la menor cantidad de errores posible.

Finalmente en una última etapa, se trabajara en los detalles de interfaces de usuario y en la depuración y corrección de errores del sistema completo, para poder proceder a realizar las pruebas y análisis correspondientes.

pruebas y conclusiones

# Evaluación de los resultados

La ejecución se realizó de la siguiente manera:

Para la red de retropropagación con un entrenamiento con 225 neuronas en la capa de entrada, 169 neuronas en la capa oculta y 5 neuronas en la capa de salida. Con un error del 10%, un coeficiente de aprendizaje de 0,80 y con el rango de inicialización entre -0.3 y 0.3. Utilizando 400 muestras. El tiempo total de entrenamiento fue de aproximadamente 10462 segundos realizando 4494 iteraciones. El resultado de pesos de este entrenamiento se encuentra en el archivo “Pesos1.ap” que se encuentra en la carpeta “Pesos”.

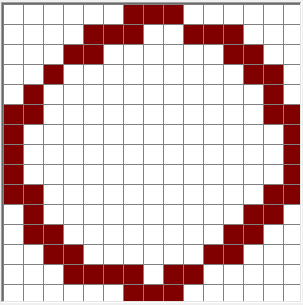
Para la red de contrapropagación con un entrenamiento con 225 neuronas en la capa de entrada, 5 neuronas en la capa oculta y 5 neuronas en la capa de salida. Con un coeficiente de aprendizaje de 0,80 y un error permitido para la capa oculta de 0,05. Utilizando 400 muestras. El tiempo total de entrenamiento fue de aproximadamente 10462 segundos realizando 4494 iteraciones. El resultado de pesos de este entrenamiento se encuentra en el archivo “Pesos1.ap” que se encuentra en la carpeta “Pesos”.

Los resultados obtenidos para cada una de las muestras son los siguientes:

### Pruebas para la figura circulo:

Prueba 1:

Con la muestra:

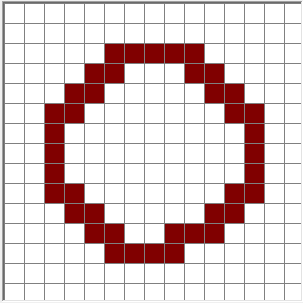


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 2:

Con la muestra:

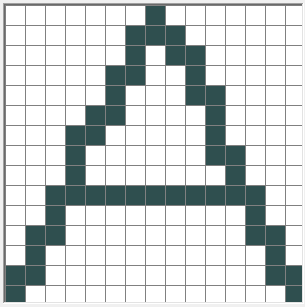


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 3:

Con la muestra:

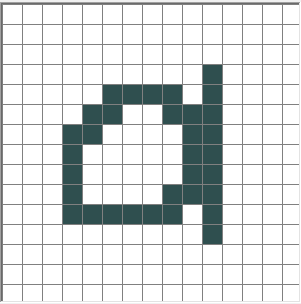


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 4:

Con la muestra:

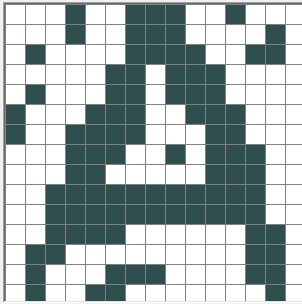


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 5:

Con la muestra:



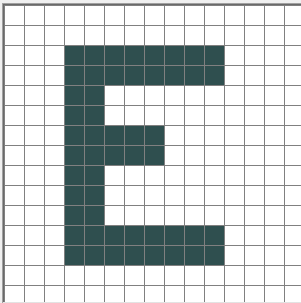
Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

### Pruebas para la figura triangulo:

Prueba 1:

Con la muestra:

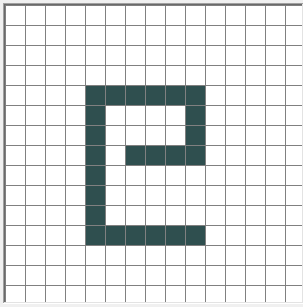


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 2:

Con la muestra:

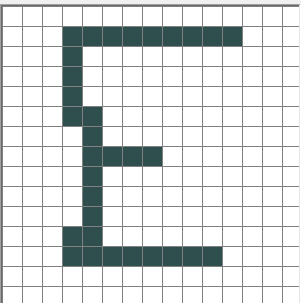


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 3:

Con la muestra:

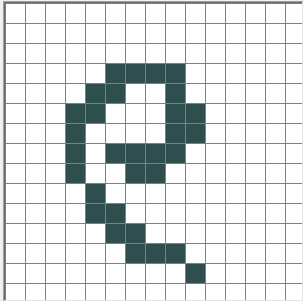


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 4:

Con la muestra:

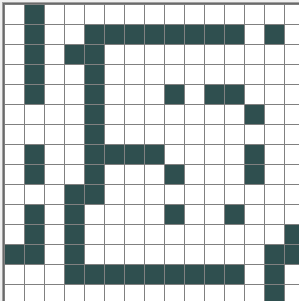


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 5:

Con la muestra:



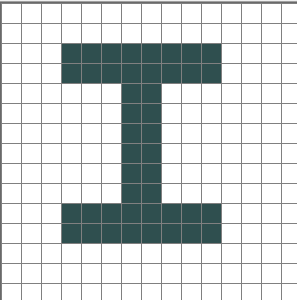
Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

### Pruebas para la figura cuadrado:

Prueba 1:

Con la muestra:

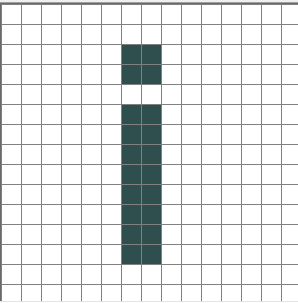


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 2:

Con la muestra:

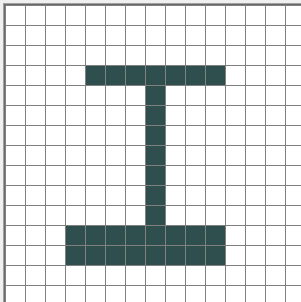


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 3:

Con la muestra:

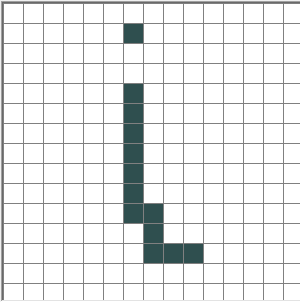


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 4:

Con la muestra:

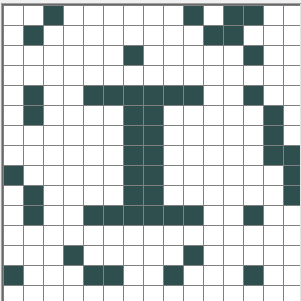


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 5:

Con la muestra:



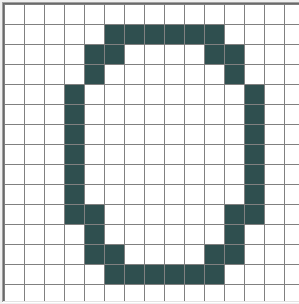
Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

### Pruebas para la figura pentágono:

Prueba 1:

Con la muestra:

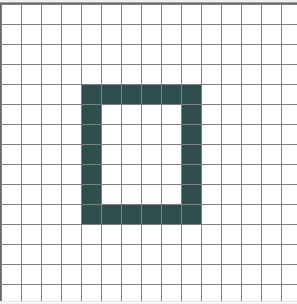


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 2:

Con la muestra:

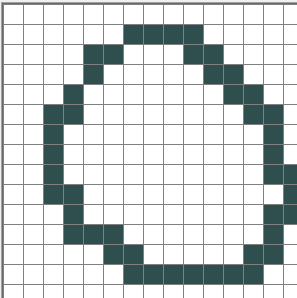


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 3:

Con la muestra:

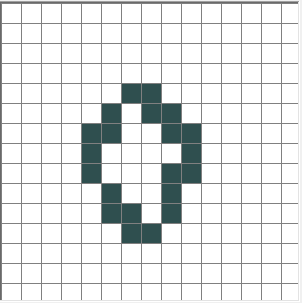


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 4:

Con la muestra:

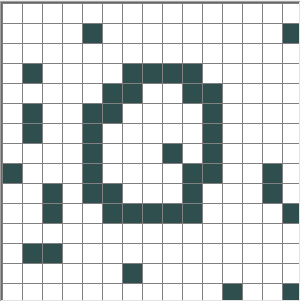


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 5:

Con la muestra:



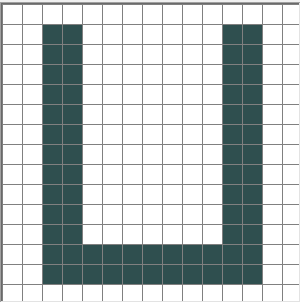
Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

### Pruebas para la figura hexágono:

Prueba 1:

Con la muestra:

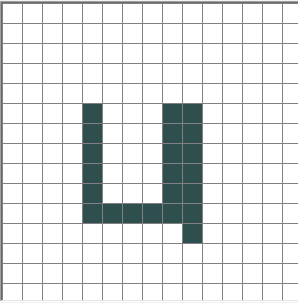


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 2:

Con la muestra:

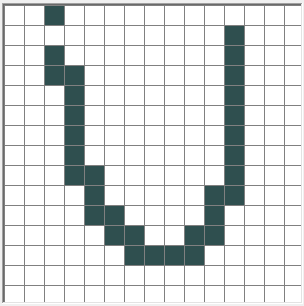


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 3:

Con la muestra:

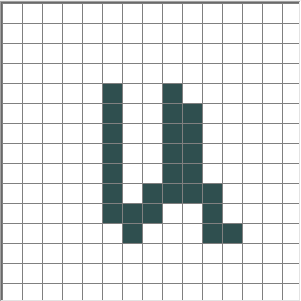


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 4:

Con la muestra:

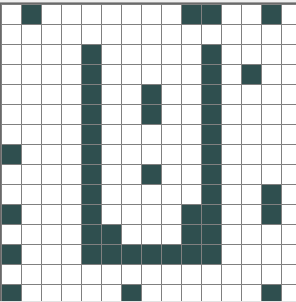


Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

Prueba 5:

Con la muestra:



Con BPN se obtuvo el resultado:

Con CPN se obtuvo el resultado:

# Conclusiones

Con estas pruebas se obtuvieron 18 correctas de 25 en total para un porcentaje de resultados correctos del 72%.

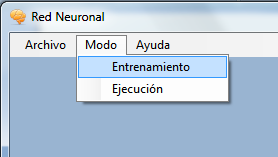
Manual de Usuario

# Manual de usuario

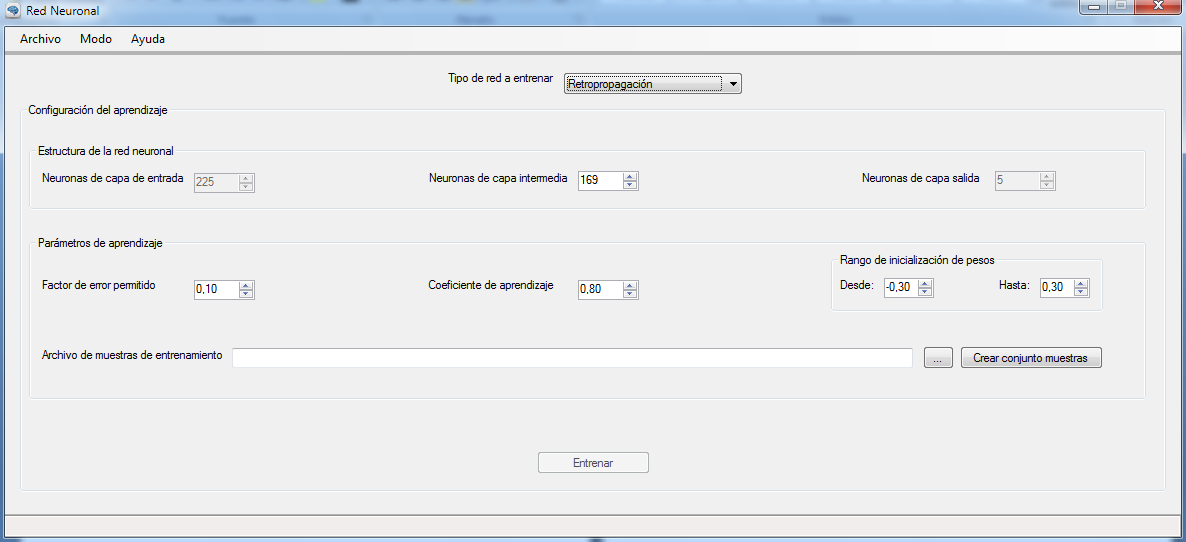
### Entrenamiento de la red de Retropropagacion

Para realizar el entrenamiento de la red debemos de realizar los siguientes pasos:

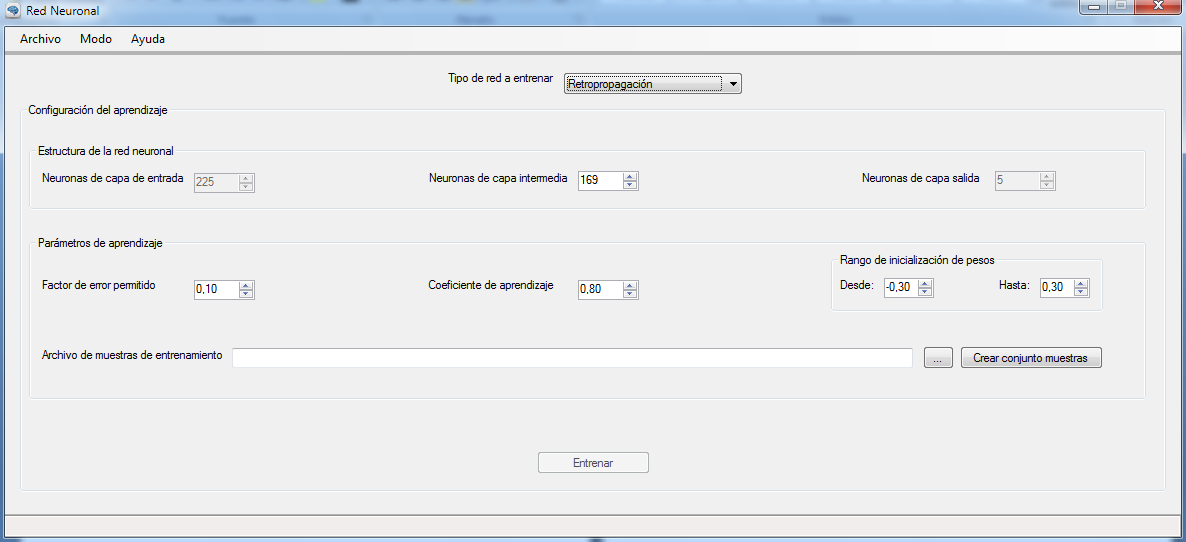
1. Elegimos la opción del menú principal: Modo-> Entrenamiento.



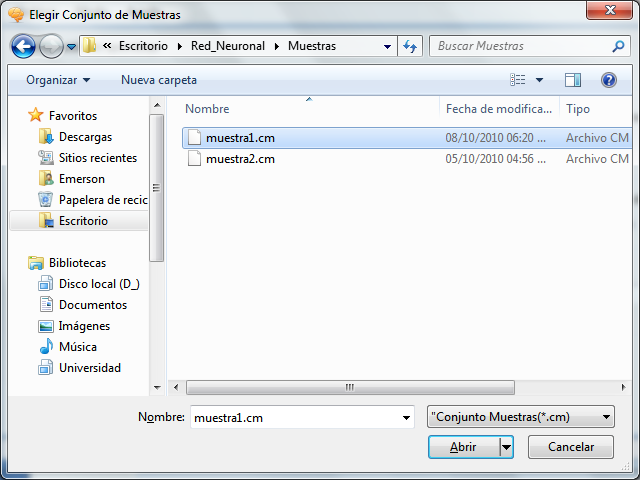
1. Aquí debemos de escoger que tipo de red queremos entrenar. seleccionamos retropropagación:



1. Se nos despliega una pantalla en la cual podemos elegir las opciones de configuración del entrenamiento.

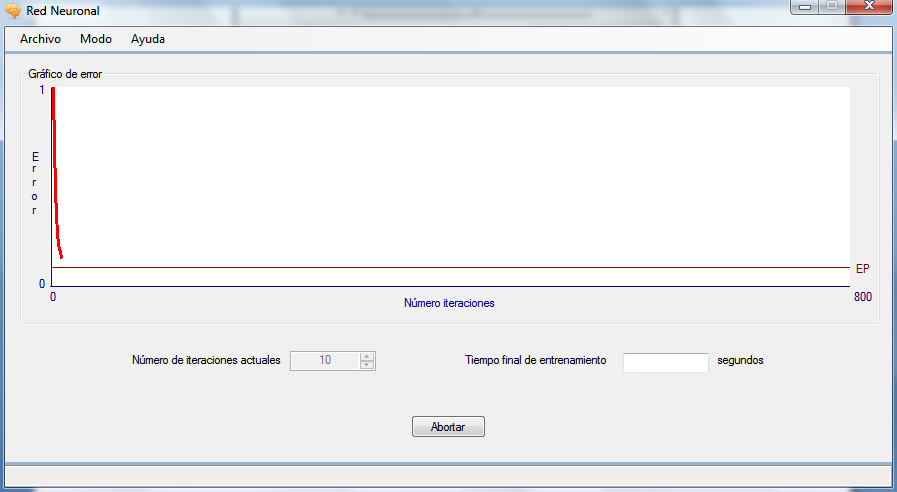


* 1. **Estructura de la red neuronal:** Podemos elegir los parámetros que contienen la estructura de la red como lo son la cantidad de neuronas para cada una de las 3 capas de la red. Para esta implementación solamente se pueden elegir la cantidad de neuronas en la capa intermedia.
  2. **Parámetros de aprendizaje:** Podemos cambiar aspectos como el error permitido durante el entrenamiento, el coeficiente de aprendizaje con el que se ajustan los pesos de la red y el rango dentro del cual se inicializaran los pesos de manera aleatoria.
  3. **Muestras de entrenamiento:** Elegimos el archivo que contiene las muestras para el entrenamiento, esto lo realizamos realizando click en  el cual nos despliega una ventana de navegación en la cual elegimos algún archivo de muestras el cual contenga la extensión “.CM”. En nuestro caso podemos encontrar archivos con muestras en la carpeta “Muestras” que se encuentra en la raíz.



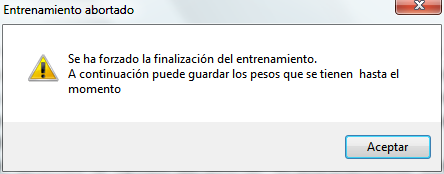
* 1. **Creación de muestras**: además de elegir entre muestras ya creadas, podemos crear nuestras propias muestras, esto se especifica en la sección [Creación de muestras](#_Creación_de_muestras) más adelante.

1. Una vez configurado el entrenamiento realizamos click en el botón “Entrenar”. Esto nos lleva a la pantalla en la cual se muestra el proceso del entrenamiento.



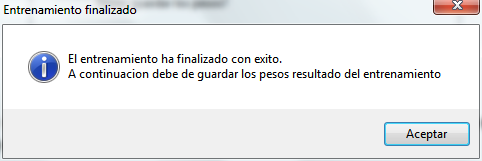
En esta pantalla encontramos un gráfico en el cual se muestra el error que se va obteniendo durante el entrenamiento. En el campo “Número de iteraciones actuales” se nos muestra la iteración por la cual nos encontramos en el entrenamiento y finalmente el campo “Tiempo final de entrenamiento” nos presenta el tiempo, en segundos, que duró el entrenamiento.

Además tenemos la opción de abortar el entrenamiento durante el transcurso de éste, para ello realizamos click en el botón “Abortar”, esto nos despliega un mensaje informándonos de esta situación e informándonos de que podemos guardar los pesos que llevamos hasta el momento en el entrenamiento.

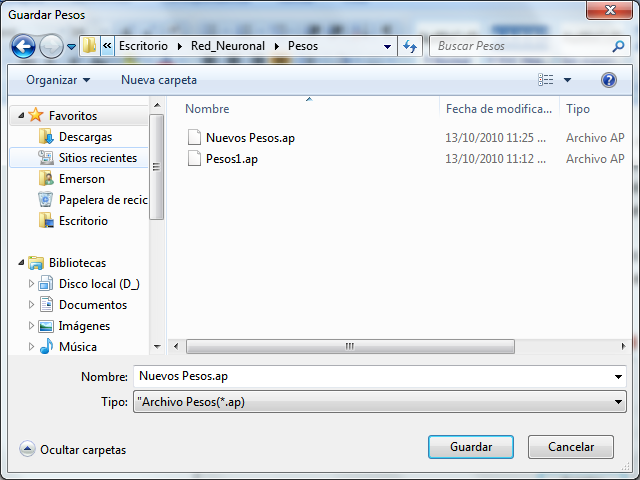


Para guardar los pesos nos enviará al proceso descrito en [guardar pesos de entrenamiento](#guardarPesos).

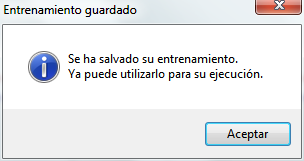
1. Una vez finalizado el entrenamiento se nos despliega un mensaje informándonos este hecho y que seguidamente debemos de guardar los pesos generados por el entrenamiento.



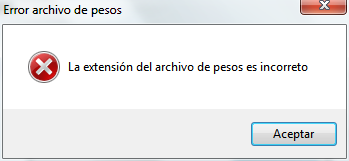
1. **Guardar los pesos del entrenamiento**: Se nos despliega una ventana de navegación en la cual podemos guardar los pesos en la ruta que elijamos.



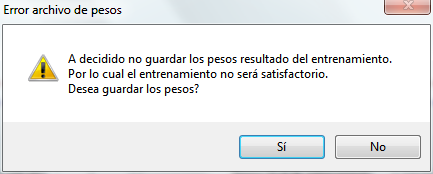
Si guardamos los pesos de manera exitosa se nos despliega un mensaje informándonos.



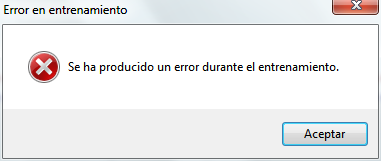
Si ocurriera un error a la hora de guardar los pesos el sistema nos despliega una ventana informándonos según sea el tipo del error, y además nos da la oportunidad de realizarlo de nuevo.



Si decidimos no guardar los pesos se nos despliega una ventana la cual nos informa de esto y nos da la opción de volver a realizarlo. Si se elige realizarlo de nuevo volveremos al mismo proceso anterior, de lo contrario no se salvarán los pesos.



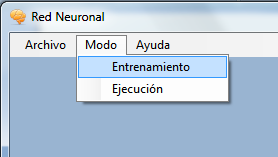
1. Si se presenta algún problema durante el entrenamiento se despliega un mensaje y se direcciona de nuevo a la pantalla de configuración del entrenamiento.



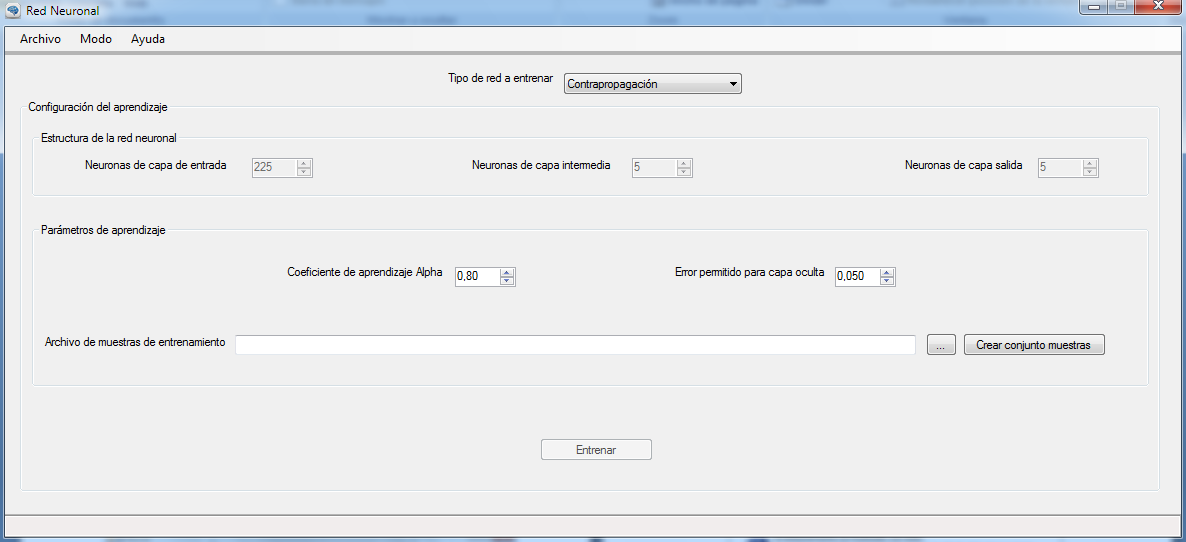
### Entrenamiento de la red de Contrapropagacion

Para realizar el entrenamiento de la red debemos de realizar los siguientes pasos:

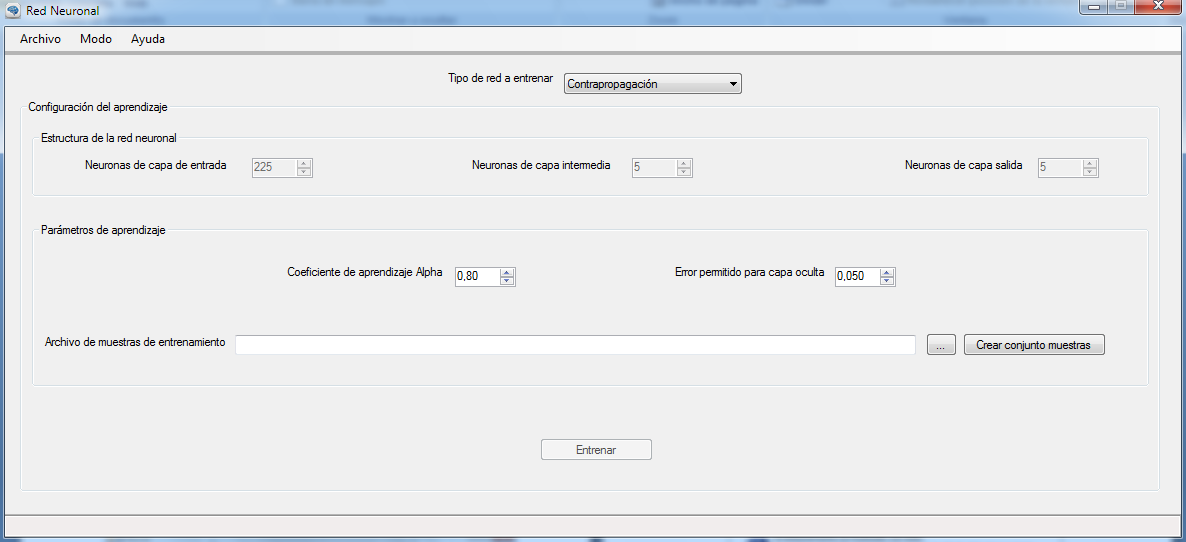
1. Elegimos la opción del menú principal: Modo-> Entrenamiento.



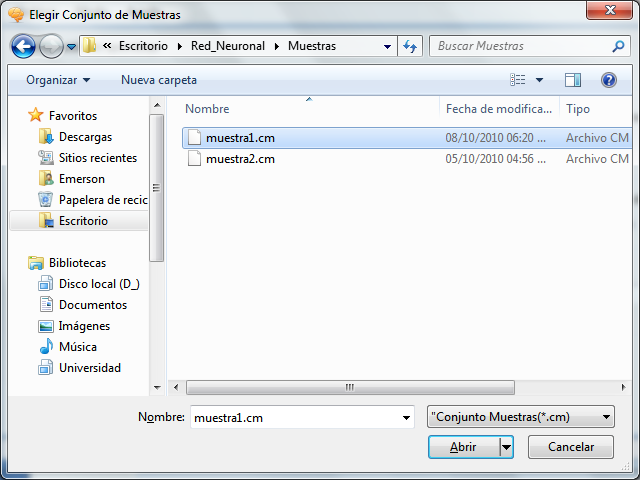
1. Aquí debemos de escoger que tipo de red queremos entrenar. seleccionamos contrapropagación:



1. Se nos despliega una pantalla en la cual podemos elegir las opciones de configuración del entrenamiento.

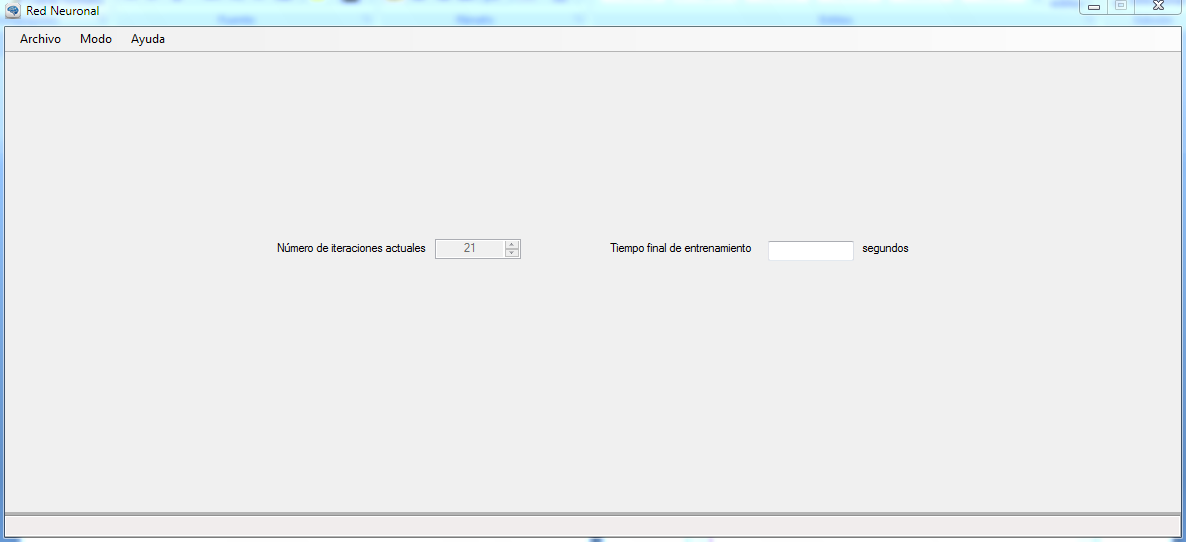


* 1. **Estructura de la red neuronal:** Podemos elegir los parámetros que contienen la estructura de la red como lo son la cantidad de neuronas para cada una de las 3 capas de la red. Para esta implementación ya las cantidades están predefinidas y no se pueden cambiar.
  2. **Parámetros de aprendizaje:** Podemos cambiar aspectos como el error permitido durante el entrenamiento y el coeficiente de aprendizaje con el que se ajustan los pesos de la red.
  3. **Muestras de entrenamiento:** Elegimos el archivo que contiene las muestras para el entrenamiento, esto lo realizamos realizando click en  el cual nos despliega una ventana de navegación en la cual elegimos algún archivo de muestras el cual contenga la extensión “.CM”. En nuestro caso podemos encontrar archivos con muestras en la carpeta “Muestras” que se encuentra en la raíz.



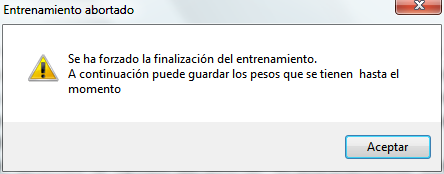
* 1. **Creación de muestras**: además de elegir entre muestras ya creadas, podemos crear nuestras propias muestras, esto se especifica en la sección [Creación de muestras](#_Creación_de_muestras) más adelante.

1. Una vez configurado el entrenamiento realizamos click en el botón “Entrenar”. Esto nos lleva a la pantalla en la cual se muestra el proceso del entrenamiento.



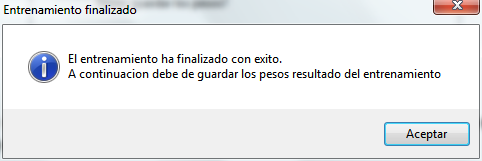
En esta pantalla encontramos el campo “Número de iteraciones actuales” se nos muestra la iteración por la cual nos encontramos en el entrenamiento y finalmente el campo “Tiempo final de entrenamiento” nos presenta el tiempo, en segundos, que duró el entrenamiento.

Además tenemos la opción de abortar el entrenamiento durante el transcurso de éste, para ello realizamos click en el botón “Abortar”, esto nos despliega un mensaje informándonos de esta situación e informándonos de que podemos guardar los pesos que llevamos hasta el momento en el entrenamiento.

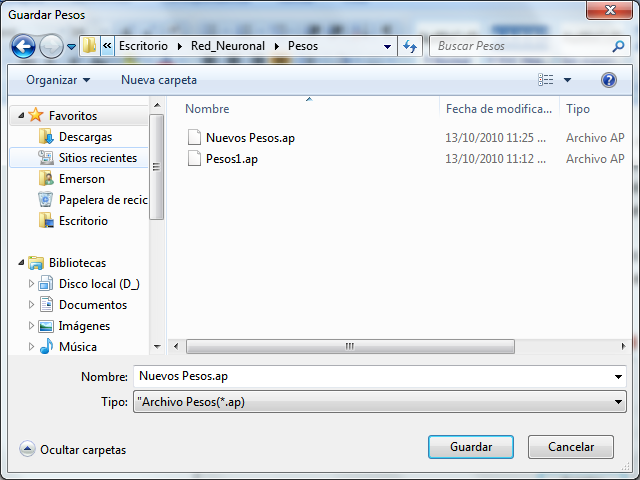


Para guardar los pesos nos enviará al proceso descrito en [guardar pesos de entrenamiento](#guardarPesos).

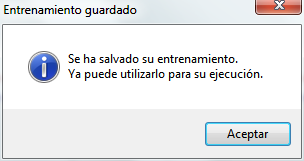
1. Una vez finalizado el entrenamiento se nos despliega un mensaje informándonos este hecho y que seguidamente debemos de guardar los pesos generados por el entrenamiento.



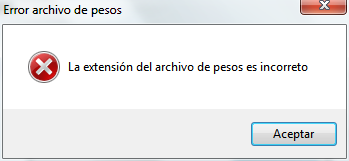
1. **Guardar los pesos del entrenamiento**: Se nos despliega una ventana de navegación en la cual podemos guardar los pesos en la ruta que elijamos.



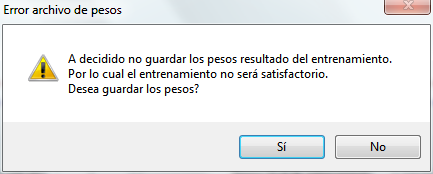
Si guardamos los pesos de manera exitosa se nos despliega un mensaje informándonos.



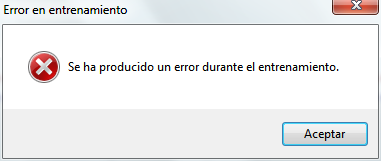
Si ocurriera un error a la hora de guardar los pesos el sistema nos despliega una ventana informándonos según sea el tipo del error, y además nos da la oportunidad de realizarlo de nuevo.



Si decidimos no guardar los pesos se nos despliega una ventana la cual nos informa de esto y nos da la opción de volver a realizarlo. Si se elige realizarlo de nuevo volveremos al mismo proceso anterior, de lo contrario no se salvarán los pesos.



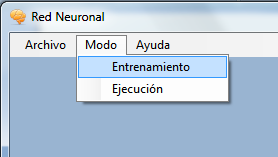
1. Si se presenta algún problema durante el entrenamiento se despliega un mensaje y se direcciona de nuevo a la pantalla de configuración del entrenamiento.



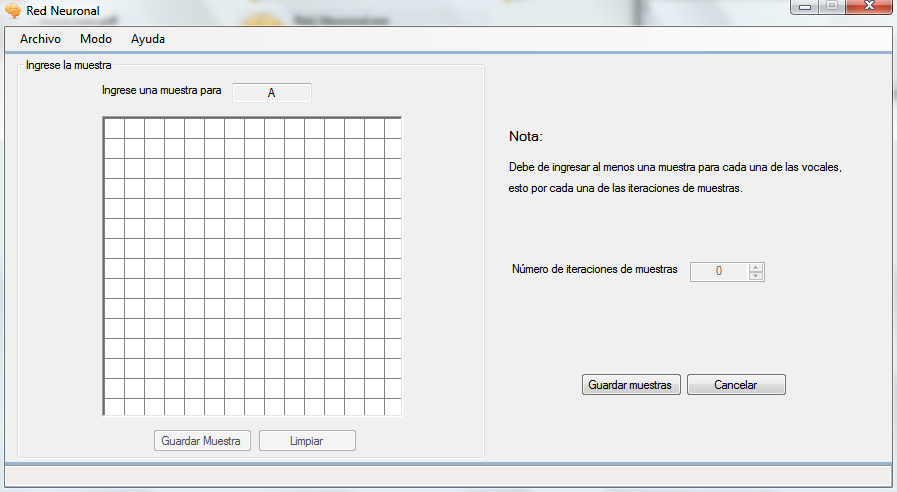
### Creación de muestras

Para la creación de muestras debemos de realizar los siguientes pasos:

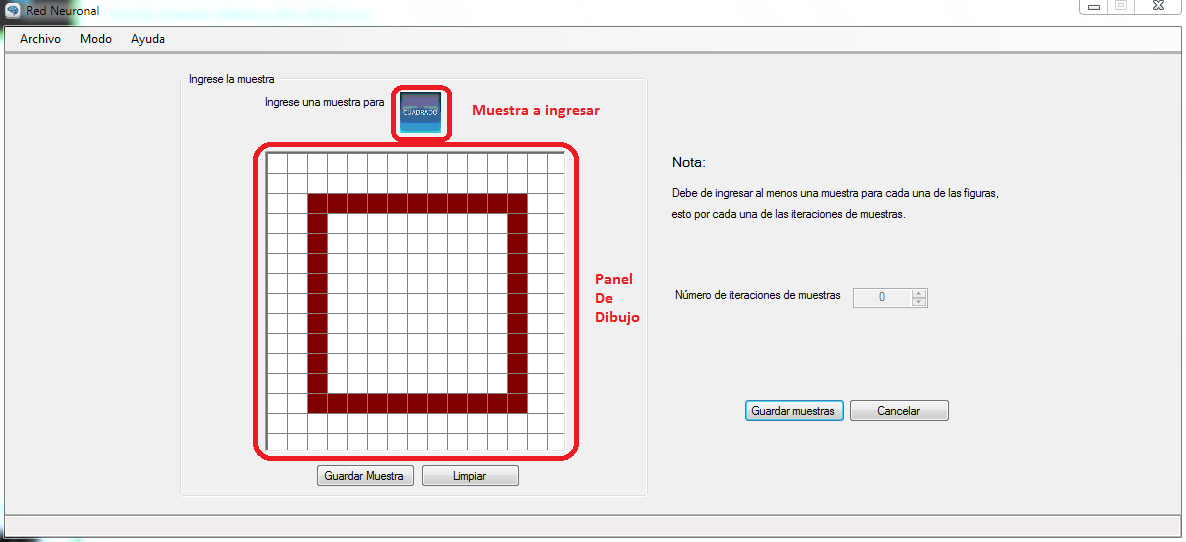
1. Ingresamos en modo de entrenamiento eligiendo la opción del menú principal: Modo-> Entrenamiento.



1. Se nos despliega una pantalla en la cual realizamos click en el botón 
2. Esto nos despliega la pantalla en la cual podemos crear nuestras muestras.

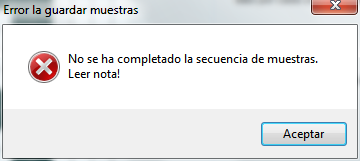


1. Para la creación de las muestras se nos muestra en la parte superior la muestra que debemos ingresar y en el panel de dibujo realizamos el dibujo de dicha muestra. Para dibujar realizamos click izquierdo sobre el panel y arrastramos el mouse llenando las casillas. Si deseamos borrar alguna de las casillas realizamos click derecho sobre la casilla y podemos arrastrarla dentro del panel para borrar más casillas.

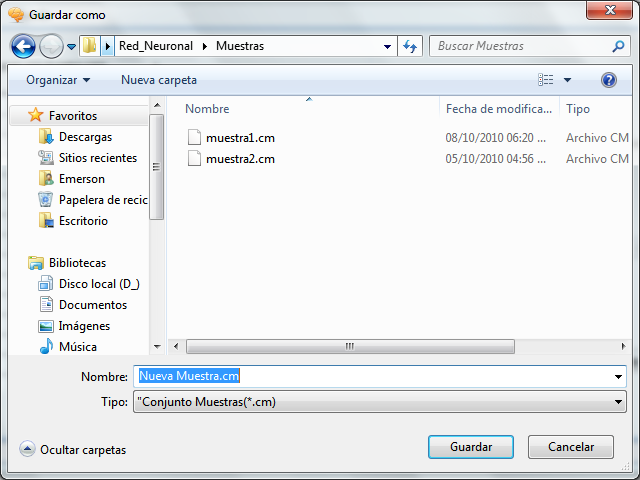


Además podemos limpiar el panel de dibujo, para ingresar una muestra desde cero, realizando click en el botón.

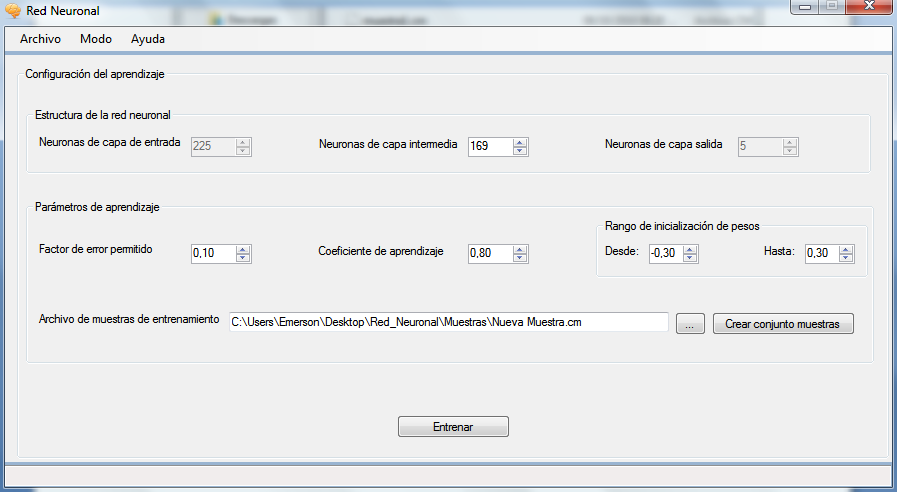
1. Una vez ingresada la muestra realizamos click en el botón, el cual nos guardará la muestra ingresada, nos limpiará el panel y nos pedirá la siguiente muestra.
2. En el campo “Numero de iteración de muestras” se nos muestra la cantidad de iteraciones de muestras que hemos ingresado. Una iteración de muestras está compuesta por una muestra de cada una de las vocales.
3. Una vez ingresado un conjunto de iteración de muestras podemos guardarlas realizando click en el botón. Si no hemos completado una iteración completa de muestras se nos despliega un mensaje con el error y nos deja en terminar de ingresar el conjunto de muestras.



Si hemos completado al menos una iteración se nos despliega una ventana de navegación en la cual elegimos la ruta y el nombre del archivo con el que vamos a guardarlo.



Una vez guardadas las muestras, el sistema nos devuelve a la pantalla de configuración del entrenamiento y en el campo “Archivo de muestras de entrenamiento” se nos muestra la dirección de las muestras que acabamos de crear.

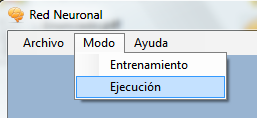


1. Para cancelar la creación de muestras realizamos click en el botón el cual nos envía de regreso a la ventana de configuración del entrenamiento.

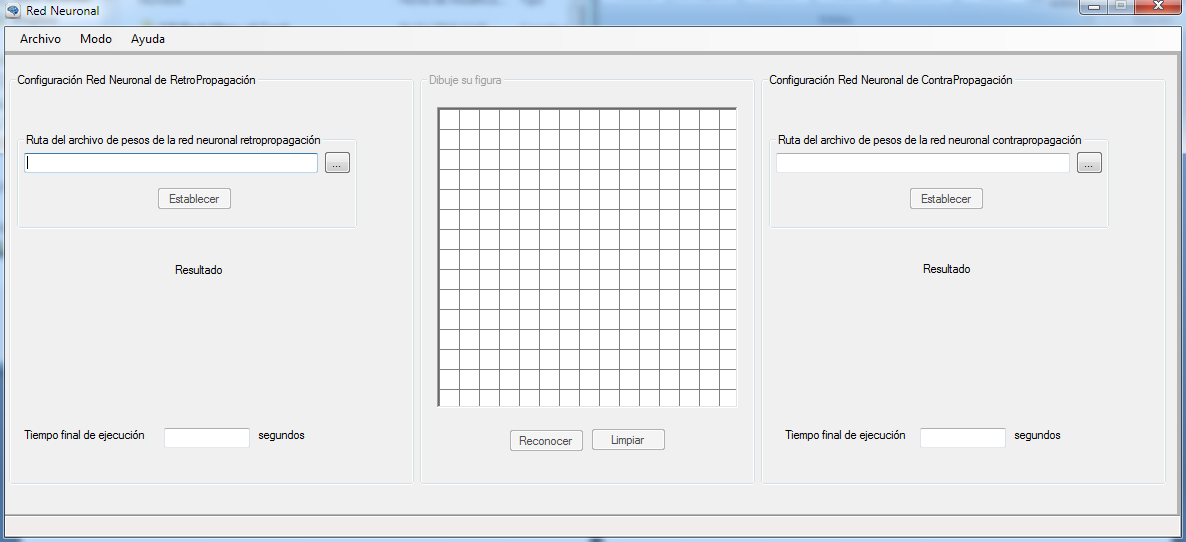
### Ejecución

Para realizar la ejecución del reconocimiento debemos de realizar los siguientes pasos:

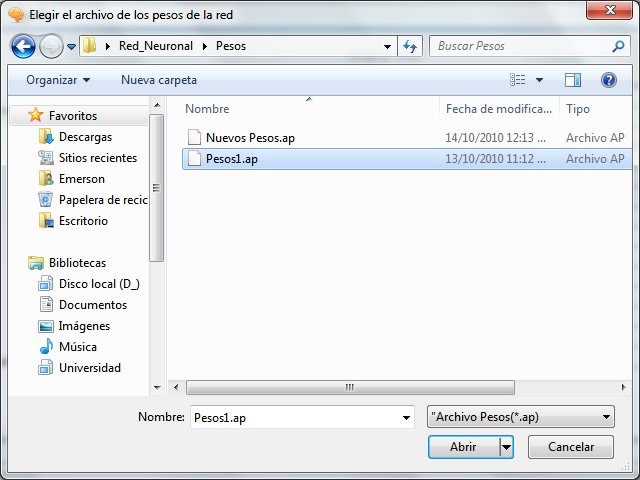
1. Elegimos la opción del menú principal: Modo-> Ejecución.



1. Se nos despliega una pantalla para la ejecución del reconocimiento.

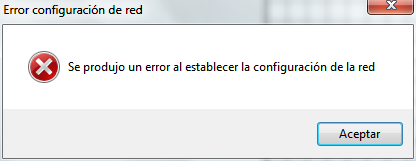


1. Primero que nada debemos de establecer la ruta del archivo de pesos con el cual se configurará cada red neuronal. Para ello realizamos click en el botón de cada red. Esto nos despliega una ventana de navegación en la cual elegimos algún archivo de pesos el cual contenga la extensión “.AP”. En nuestro caso podemos encontrar archivos con muestras en la carpeta “Pesos” que se encuentra en la raíz.

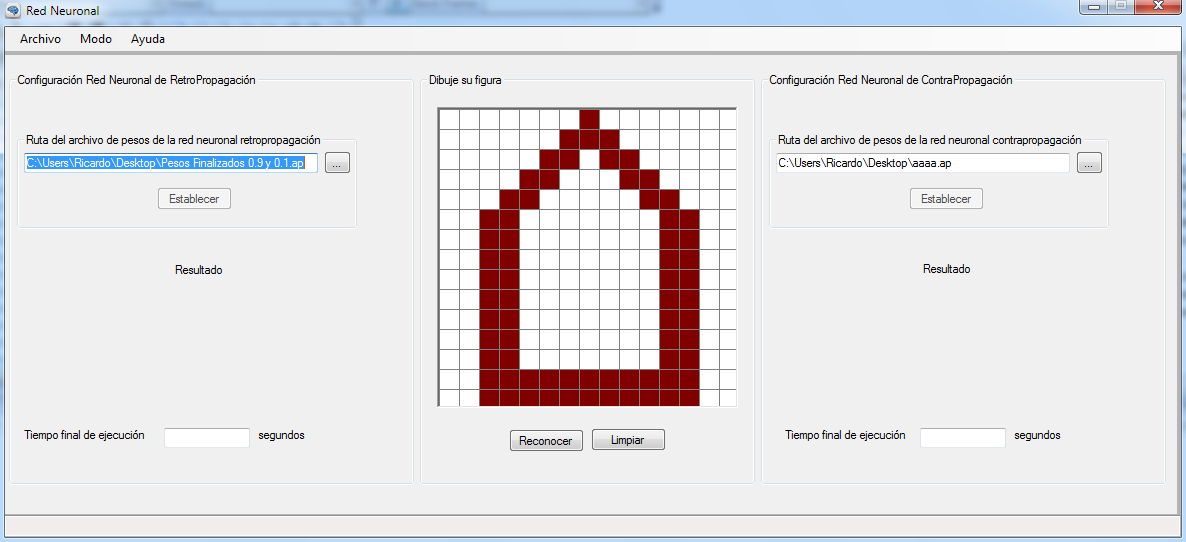


1. Una vez elegidos lol archivo de pesos, realizamos click en el botón de cada red para que se configuren nuestras redes neuronales.

Si ocurre algún error durante el establecimiento de la configuración se despliega un mensaje de error y se permite volver a intentar otra configuración.

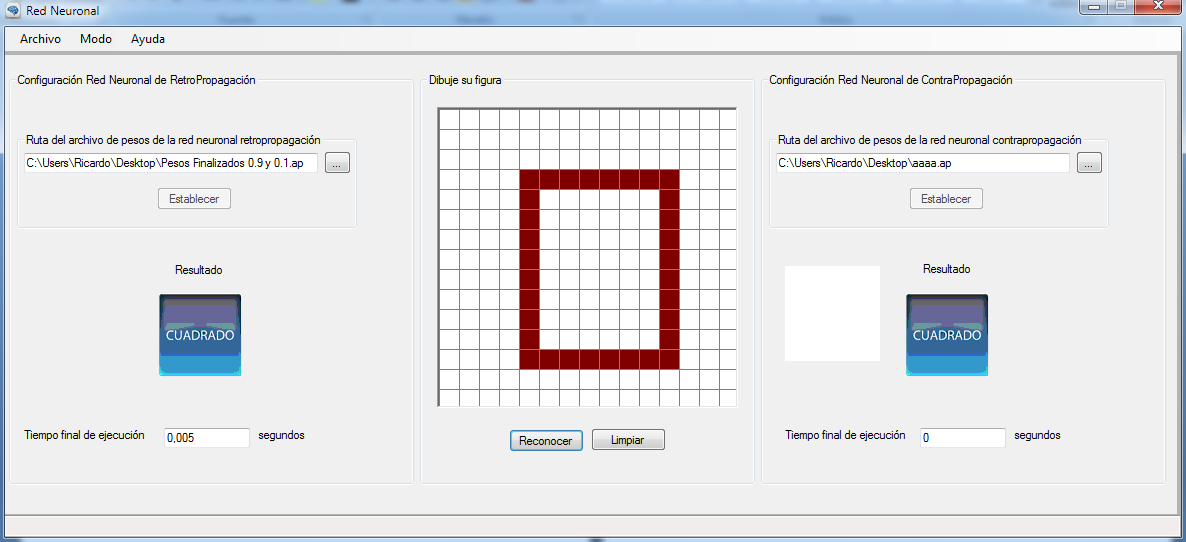


1. Una vez configuradas las redes se nos habilita el panel de dibujo en el cual podemos dibujar una letra para ser reconocida. Para dibujar realizamos click izquierdo sobre el panel y arrastramos el mouse llenando las casillas. Si deseamos borrar alguna de las casillas realizamos click derecho sobre la casilla y podemos arrastrarla dentro del panel para borrar más casillas.



Además podemos limpiar el panel de dibujo, para ingresar una vocal desde cero, realizando click en el botón.

1. Una vez ingresada la vocal realizamos click en el botón, el cual nos hace el reconocimiento de la vocal.



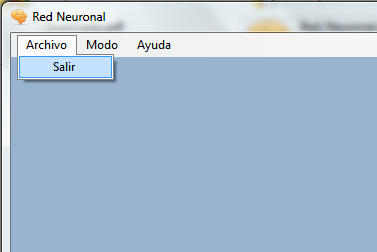
En los campos “Resultado” se nos muestra el resultado del reconocimiento de cada red, en caso de que no se reconozca la letra nos despliega el resultado “X”. En el campo Tiempo final de ejecución se nos muestra el tiempo que duró cada red neuronal en reconocer la letra.

Este proceso podemos repetirlo las veces que deseamos.

### Salir

Para salir del programa podemos realizarlo de dos maneras:

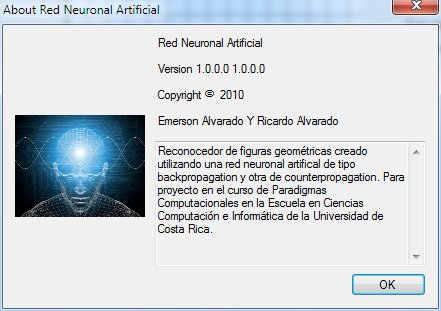
1. Elegimos la opción del menú principal: Archivo -> Salir.



1. Damos click en el botón  de la ventana principal.

### Acerca de

Para encontrar información acerca del programa podemos realizarlo eligiendo la opción en el menú principal: Ayuda -> Acerca De.



# Observaciones

### Formato de los archivos

Archivo de muestras

El formato del archivo de muestras contiene la extensión “.CM” y está estructurado de la siguiente manera:

* Cada una de las líneas del archivo está compuesto por una muestra completa.
* Los primeros 5 dígitos de la muestra son para la entrada, esto es la codificación de cada una de las vocales.
  + Para el circulo la codificación correspondiente es “10000”
  + Para el triangulo la codificación correspondiente es “01000”
  + Para el cuadrado la codificación correspondiente es “00100”
  + Para el pentágono la codificación correspondiente es “00010”
  + Para el hexágono la codificación correspondiente es “00001”
* Después de la entrada se encuentra el separador “;”.
* Los siguientes 225 dígitos son la matriz de entrada en el dibujo realizado. Un “1” significa casilla pintada y un “0” significa casilla en blanco.

Si a la hora de realizar el entrenamiento se elige un archivo con una extensión diferente o si el archivo no contiene la estructura mencionada anteriormente, el sistema lanzará un error y no lo procesará.

Ejemplo de muestra:

|  |
| --- |
| 10000;000000000000000000000000001000000000000010000000000010000000000011101…  01000;000000000000000000000000000000000000000000000001111111111000001000100…  00100;000000000000000000000000000000000000000000000000000000000000000000000…  00010;000000000000000000000000000000000000011110000000111100011000001000000…  00001;000000000000000000000000000000000000000000000011101010011111000000000… |

Archivos de pesos

El formato del archivo de pesos contiene la extensión “.AP” y está estructurado de la siguiente manera:

* Se encuentra separado por etiquetas: la primera etiqueta es “Configuración”, seguido de “Oculta”, “Salida” y finalmente “Fin”
* En configuración tenemos 4 líneas: en la primera el tipo de red, retropropagación (BP) o contrapropagación (CP), luego el numero de neuronas en la capa de entrada, seguido del numero de neuronas en la capa oculta y finalmente el numero de neuronas en la capa de salida.
* En “Oculta” tenemos todos los pesos correspondientes a la capa oculta.
* En “Salida” tenemos todos los pesos correspondientes a la capa de salida.

Si a la hora de realizar la ejecución se elige un archivo con una extensión diferente o si el archivo no contiene la estructura mencionada anteriormente, el sistema lanzará un error y no lo procesará.

Ejemplo de pesos de contrapropagación:

|  |
| --- |
| Configuracion  CP  225  169  5  Oculta  -0,103318809417566  0,0736489179689587  -0,0687190778398462  -0,159105756321885  …  Salida  -0,295134013065907  0,142928800861309  -0,388595099124425  0,146818893835928  …  Fin |

# Referencias bibliográficas

1. Redes neuronales : algoritmos, aplicaciones y técnicas de programación, James A Freeman, David M. Skapura. Editorial Addison Wesley, 1993.
2. Redes Neuronales Artificiales, José R. Hilera y Victor J Martinez. Editorial Alfaomega. Madrid. España, 2000
3. Digital Neural Networks, S. Y. Kung. Editorial Prentice Hall, 1993
4. Redes neuronales artificiales: Fundamentos, modelos y aplicaciones, Sandra Daza P. Universidad Militar Nueva Granada. Facultad de Ingeniería Mecatrónica, Colombia
5. Backpropagation, Sección de redes neurales artificiales de Electrónica México. Consultado el 10 de setiembre del 2010. <http://www.electronica.com.mx/neural/informacion/backpropagation.html>
6. Redes Neuronales Artificiales, Grupo de Circuitos: Departamento de señales, sistemas y radiocomunicaciones de la Universidad Politécnica de Madrid. Consultado el 10 de setiembre del 2010. <http://www.gc.ssr.upm.es/inves/neural/ann2/anntutorial.html>
7. Redes Neuronales Artificiales, Sección de redes neurales artificiales de Electrónica México. Consultado el 10 de setiembre del 2010. <http://electronica.com.mx/neural/>
8. Redes de Contrapropagación: Creación de subredes mediante memorias heteroasociativas. Dr. Héctor Allende, Universidad Técnica Federico Santamaría, Chile. <http://www.inf.utfsm.cl/~hallende/download/capitulo_5.ppt>
9. Redes Neuronales, Sección de redes neuronales artificiales de la Universidad de Antioquia. Consultado el 30 de octubre del 2010. <http://ingenieria.udea.edu.co/investigacion/mecatronica/mectronics/redes.htm>
10. Sección de Ciencias de la Computación de la Universidad de Birmingham. Consultado el 30 de octubre del 2010. <http://www.cs.bham.ac.uk/~mmk/Teaching/AI/figures/backpropagation.jpg>
11. Pagina de investigaciones emerald. Seccion de Inteligencia Artificial. Consultado el 30 de octubre del 2010. <http://www.emeraldinsight.com/content_images/fig/3330070302015.png>

Apéndices

### Apéndice 1:

Diagrama de la Red Neuronal de retropropagación utilizada en el proyecto:

La red que utilizamos en el proyecto consiste de 225 neuronas en la capa de entrada, correspondientes a las 225 celdas que contiene la cuadricula de dibujo presentada en la interfaz del sistema.

En la capa oculta, la red consiste por defecto de 169 neuronas, aunque el usuario puede definir a la hora de entrenar la red la cantidad de neuronas que desea en esta capa.

En la capa de salida se manejan 5 neuronas, dado que se manejan 5 tipos diferentes de figuras que la red puede reconocer.

…

…

### Apéndice 2:

Diagrama de la Red Neuronal de contrapropagación utilizada en el proyecto:

Para la red de contrapropagación manejamos de igual forma 5 neuronas en la capa de entrada, correspondientes a las 225 celdas de la cuadricula de entrada.

Se manejan 5 instars en la capa oculta, que son las que clasifican las figuras geométricas en las 5 categorías posibles que puede reconocer la red.

En la capa de salida se manejan 5 outsars, las cuales mapean el resultado de la instar ganadora a la salida de la red.

…

### Apéndice 3:

Cronograma de trabajo realizado:

|  |  |  |
| --- | --- | --- |
| Semana | Emerson | Ricardo |
| 1 (12 al 18 de Setiembre) | Diseño del componente de control de ejecución del sistema. Definición de clases, atributos y componentes del mismo | Diseño de la red neuronal. Investigación y definición de los elementos a utilizar en la red. |
| 2 (19 al 25 de Setiembre) | Inicio de la Implementación del diseño planificado en la etapa anterior. Al no tener una red neuronal creada, es necesario crear un simulador de red que retorne valores fijos dependiendo de la entrara para probar el componente de control | Comienzo de la implementación de las redes neuronales. Trabajo en conjunto para el simulador temporal de las redes. |
| 3 (26 de Setiembre al 2 de Octubre) | Continuación de la implementación del componente de control de ejecución del sistema. Pruebas al componente | Continuación en el trabajo de redes neuronales. Trabajo en conjunto para las pruebas del control de ejecución del sistema |
| 4 (3 al 9 de Octubre) | Diseño de los componentes de control de aprendizaje del sistema. Definición de clases, atributos y componentes del mismo | Ultima semana para las redes neuronales. Deben de estar listas para poder realizarles pruebas la siguiente semana |
| 5 (10 al 16 de Octubre) | Inicio de la Implementación del diseño planificado en la etapa anterior. Al no tener una red neuronal creada, es necesario crear un simulador de red que retorne valores fijos dependiendo de la entrara para probar el componente de control | Pruebas y depuración de las redes neuronales. Diseño de las interfaces tanto de ejecución como de aprendizaje del sistema |
| 6 (17 al 23 de Octubre) | Continuación de la implementación del componente de control de aprendizaje del sistema. Pruebas al componente | Trabajo en el acoplamiento de la red con ambos componentes, tanto el de ejecución como el de aprendizaje del sistema |
| 7 (24 al 30 de Octubre) | Implementación de la interfaz de ejecución del sistema | Implementación de la interfaz de aprendizaje del sistema y de la interfaz de control del sistema |
| 8 (12 al 18 de Octubre) | Acoplamiento de todos los componentes del sistema y corrección de errores. | Acoplamiento de todos los componentes del sistema y corrección de errores. |
| 9 (31 de Octubre al 6 de Noviembre) | Experimentación y resultados con ambos tipos de redes | Experimentación y resultados con ambos tipos de redes |