|  |  |  |
| --- | --- | --- |
| **ipn** | **INSTITUTO POLITÉCNICO NACIONAL**  **ESCUELA SUPERIOR DE CÓMPUTO** |  |

**Neural Networks**

**“Red Hamming”**

Resumen

El abstracto es un reporte preciso de todo el reporte completo. Su función es indicar los contenidos del reporte para que el lector pueda ver si vale la pena leerlo completo o no.

**Por:**

**Joel Mauricio Romero Gamarra**

Profesor:

MARCO ANTONIO MORENO ARMENDÁRIZ

Noviembre 2017

**Índice**

Contenido

[Introducción: 1](#_Toc476313175)

[Análisis Teórico: 2](#_Toc476313176)

[Software (librarias, paquetes, herramientas): 2](#_Toc476313177)

[Procedimiento: 3](#_Toc476313178)

[Resultados 3](#_Toc476313179)

[Discusión: 1](#_Toc476313180)

[Conclusiones: 2](#_Toc476313181)

[Referencias: 2](#_Toc476313182)

[Código 2](#_Toc476313183)

# Introducción:

La red Hamming es una red que fue diseñada para resolver problemas de reconocimiento y clasificación de patrones binarios (1 o -1). Esta RNA es bastante interesante ya que cuenta con 2 capas distintas, en la primera se encuentra la capa FeedForward y en la segunda se encuentra la capa Recurrente.1

En la Figura 1 se muestra la arquitectura en forma matricial que representa a la red Hamming.

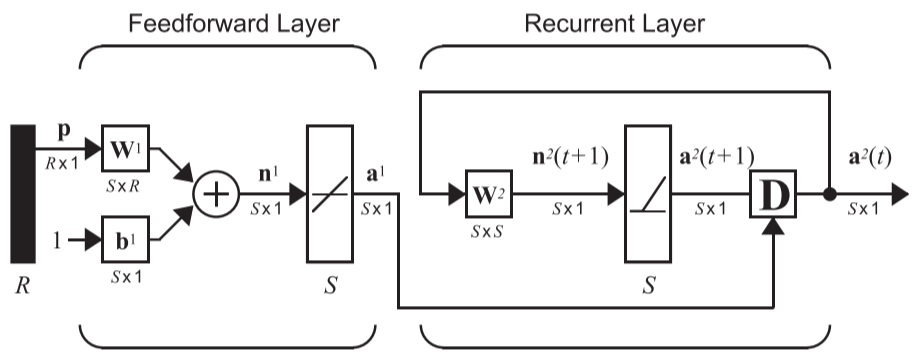


Figura . Arquitectura de la Red Hamming

Como se observa en la figura anterior, se ve claramente que la RNA cuenta con 2 capas distintas, además, cada una tiene una salida distinta descrita a continuación en su modelo matemático:

**a1 = purelin (W1 · p + b1) y a2 (0) = a1; a2 (t + 1) = poslin (W2 · a2 (t))**

Donde:

* a1: Salida de la capa FeedForward
* a2: Salida de la capa recurrente
* purelin y poslin: Funciones de Activación de la RNA
* W1 y W2: Matrices de pesos de la capa 1 y 2 respectivamente
* b1 y b2: Bias de la capa 1 y 2 respectivamente
* S: Número de neuronas
* R: Dimensión del vector de entrada

Cabe resaltar que el número de neuronas en las 2 capas es el mismo, además algo interesante es que en la capa recurrente no existe un bias como en la primera y la primera señal de salida para la segunda capa es la salida de la capa 1.

# Análisis Teórico:

Para el caso de la capa FeedForward se calcula el producto interno entre cada uno de los vectores prototipo y el vector de entrada (p). Por lo tanto, cada fila de la matriz de pesos W será cada uno de los vectores prototipo, con la restricción que todos los vectores prototipo deben tener el mismo número de rasgos.1

Para calcular el bias, es una matriz como se observa en la Figura 1 de Sx1, donde S es el número de neuronas, que ya sabemos que es el número de filas de la matriz de pesos W, y cada uno de esos elementos serán iguales, con el valor de R, que es la dimensión del vector de entrada, o sea que es el número de columnas de la matriz de pesos W.

Como ya se mencionó, la red Hamming es para patrones binarios (1 o -1), por lo tanto, al asignarle al bias los valores de R, nos aseguramos de que las salidas de la capa no sean negativas, y esto provoca que la capa recurrente funcione de manera correcta.

En el caso de la capa recurrente se le conoce como competitiva, las neuronas de la misma se inicializan con la salida de la capa FeedForward como ya se mencionó previamente, en esta capa las neuronas compiten entre ellas para determinar a un ganador, esto quiere decir que solo 1 neurona está activa a la vez y solo 1 contendrá el resultado correcto de la clasificación del vector de entrada. Después de un número de iteraciones, 1 de las neuronas tendrá 1 valor distinto de cero y todas las demás serán 0, esto nos indicará a que clase pertenece el vector de entrada p.

En el caso de la capa recurrente, la matriz de pesos se tiene que calcular a partir de S y nunca cambia, W2 se define como:

Donde ε es un valor menor a

La capa recurrente realizará iteraciones hasta que converja a una clase donde solamente 1 de las neuronas (o sea el valor de salida a) sea distinto de 0 por 2 iteraciones consecutivas.

En cada iteración, el valor de la matriz de pesos seguirá siendo el mismo, sin embargo, lo que se va a estar actualizando cada vez es el vector de entrada p, que será igual a la salida en un instante de tiempo anterior, comenzando con t = 0 que es el valor de salida de la capa FeedForward.

# Software (librarías, paquetes, herramientas):

* MATLAB R2016a
* Sublime Text 3

# Procedimiento:

Como se puede observar en la arquitectura, primero debemos hacer el cálculo de la salida de la primera capa (FeedForward), ya que nos servirá para comenzar a hacer las iteraciones de la segunda capa. Para realizar el programa en MATLAB, se escribirá en 1 archivo de texto la matriz de pesos W y en otro archivo de texto el vector de entrada p.

El algoritmo por realizar es descrito ahora:

* Leer la matriz de pesos W1 del archivo de texto correspondiente.
* Leer el vector de entrada p del archivo de texto correspondiente.
* Calcular el valor de salida a1 de la capa FeedForward.
* Calcular el valor de ε aleatoriamente únicamente cuidando que se encuentre en el rango establecido.
* Utilizar el valor de salida de la capa 1 como la primera salida de la capa 2.
* Actualizar el valor de entrada p como el valor de salida en la iteración anterior.
* Escribir cada uno de los valores de salida en un archivo de texto.
* Realizar iteraciones hasta que el valor de salida a2 sea el mismo en 2 iteraciones consecutivas y que además solo tenga 1 valor distinto de 0.
* Mostrar al usuario en que iteración la red convergió a una clase.
* Leer el archivo que contiene los valores de salida de la RNA.
* Graficar cada uno de los valores leídos para mostrar la evolución de la salida de la RNA.

Ya que contamos con el algoritmo, el paso siguiente es realizar la codificación. Los resultados obtenidos son mostrados en la siguiente sección y se hablará de ellos a profundidad en la sección de discusión más adelante.

# Resultados

Las Figuras 2 a \_\_ muestran el funcionamiento de la red Hamming para los siguientes valores de la matriz de pesos y el vector de entrada p.

y

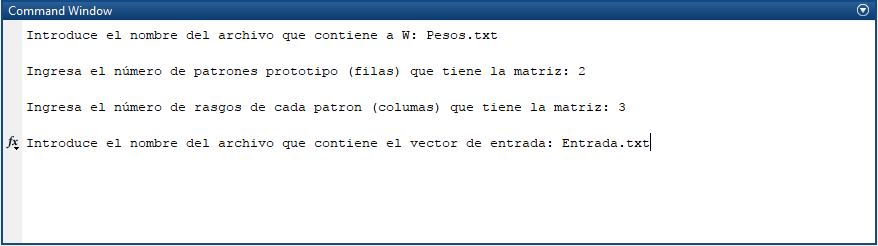


Figura . Valores iniciales pedidos al usuario

A continuación, se calcula el valor de épsilon aleatoriamente ya que conocemos el valor de S y podemos calcularlo para no hacerlo estático y lo mostramos en el Command Window para saber cuál es.

Además, se mostrará únicamente un mensaje que nos diga en que número de iteración se logró una correcta clasificación del vector de entrada (como ya se mencionó, la condición de paro es que 2 iteraciones consecutivas se tenga el mismo valor y que además solamente 1 de los valores de a2 sea distinto de cero.

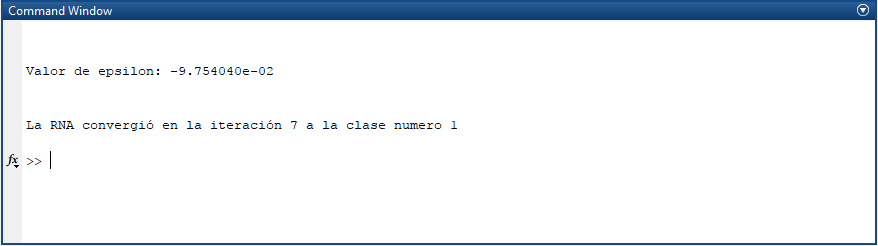


Figura . Impresiones finales ya que se clasificó correctamente el vector p

AQUÍ SIGUE WE

# Discusión:

La sección de discusión tiene 2 objetivos principales:

* Interpretar y explicar los resultados del estudio.
* Explorar la importancia del estudio, encontrando, calificando y explorando la importancia teórica de los resultados.

La discusión es también un espacio en el reporte donde cualquier calificación o reservación que se tiene sobre la investigación debe ser mencionada.

# Conclusiones:

* Lista una cosa que hayas aprendido y describe como lo aplicarías a una situación de la vida real.
* Discute los posibles errores que podrían haber ocurrido en la colección de los datos (errores experimentales).
* ¿Cómo se aplicarían los resultados obtenidos generalmente?
* ¿Hubo algún defecto en el diseño experimental o en el procedimiento?

# Referencias:

**[1]** “Red Hamming”, class notes for Neural Networks, Department of Engineering in Computer Systems, Escuela Superior de Cómputo, 2017.

# Código

Incluir todo el código fuente, comentar todo el código reutilizado y mostrar referencias.

Se debe usar el siguiente link para darle formato al código.

<https://tohtml.com/c/>

\*Acerca de las tablas y/o figuras\*

Utilizar figuras como diagramas, tablas, gráficos, gráficos o mapas puede ser una forma muy útil de mostrar y enfatizar la información en el reporte.

Las figuras esenciales para el informe deben integrarse sin problemas y correctamente, además deben explicarse y mencionarse en el cuerpo principal del reporte.

Ejemplo:

