Apuntes Neural Networks

Jorge Gómez Reus

Parte I
Introducción

1. Neural Networks

1.1. Usos principales de las redes neuronales

Se tienen 4 usos principales de las redes neuronales artificiales (RNA)

1. Aproximación de sistemas (Modo regresor)
2. Predicción de series de tiempo (Modo regresor)
3. Control de Sistemas (Modo regresor)
4. Clasificación de objetos (Modo Clasificador)

1.2. Aproximación de Sistemas

Dado un sistema $f(t_i)$ del cuál se desconoce su modelo matemático, pero se cuenta con un conjunto de datos entrada-salida (p, t) que representa su comportamiento. Se puede entrenar una RNA para que se comporte de manera similar a $f(t_i)$ en donde:

t_i : Es la variable de tiempo
p : Es la entrada (input)
t : El valor deseado (target)

1.3. Modelo Matemático

Es una representación abstracta que aproxima al comportamiento de un fenómeno real, normalmente mediante un conjunto de ecuaciones.

1.4. Ecuación

Igualdad

1.5. Datos input-output

Es un conjunto de valores que muestrea mediante sensores el comportamiento dinámico del sistema en todo su rango de funcionamiento

1.6. Buena Interpolación

Se llama generación de conocimiento

1.7. Mala Interpolación

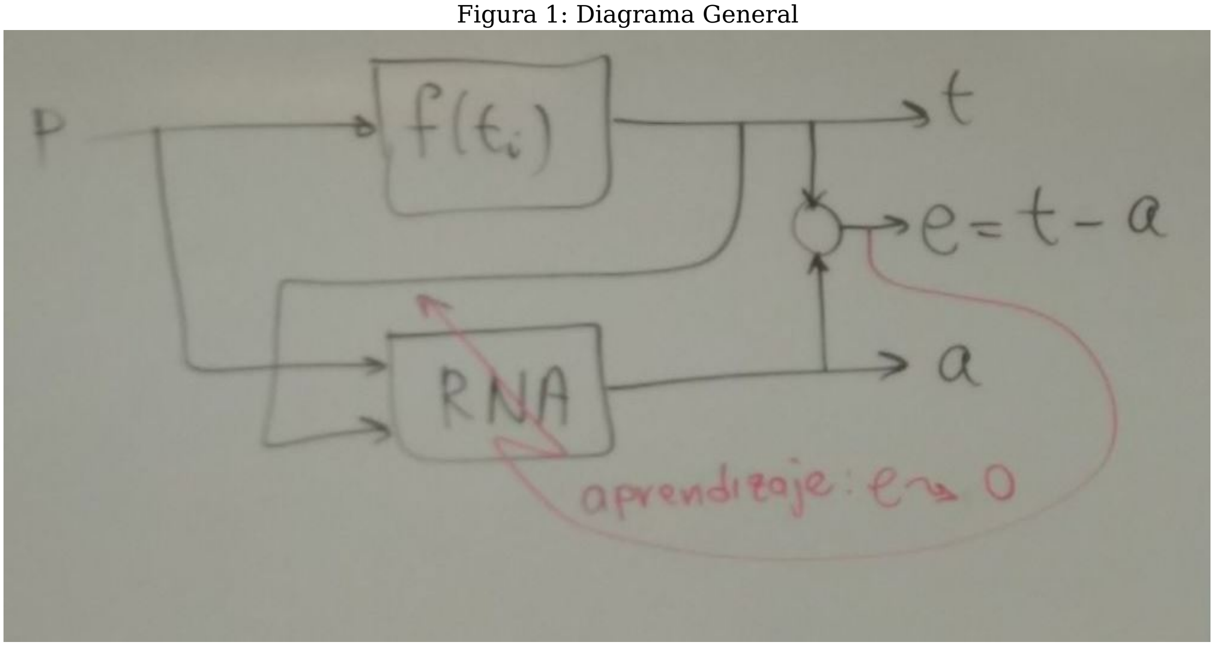
Se le llama sobrentendimiento
LA RED EN MODO REGRESIÓN FUNCIONA COMO UN INTERPOLADOR

1.8. Extrapolar

Pronosticar o predecir, se requieren RNA's recurrentes.

1.9. Diagrama General

Figura 1: Diagrama General



2. Tipos de Aprendizaje

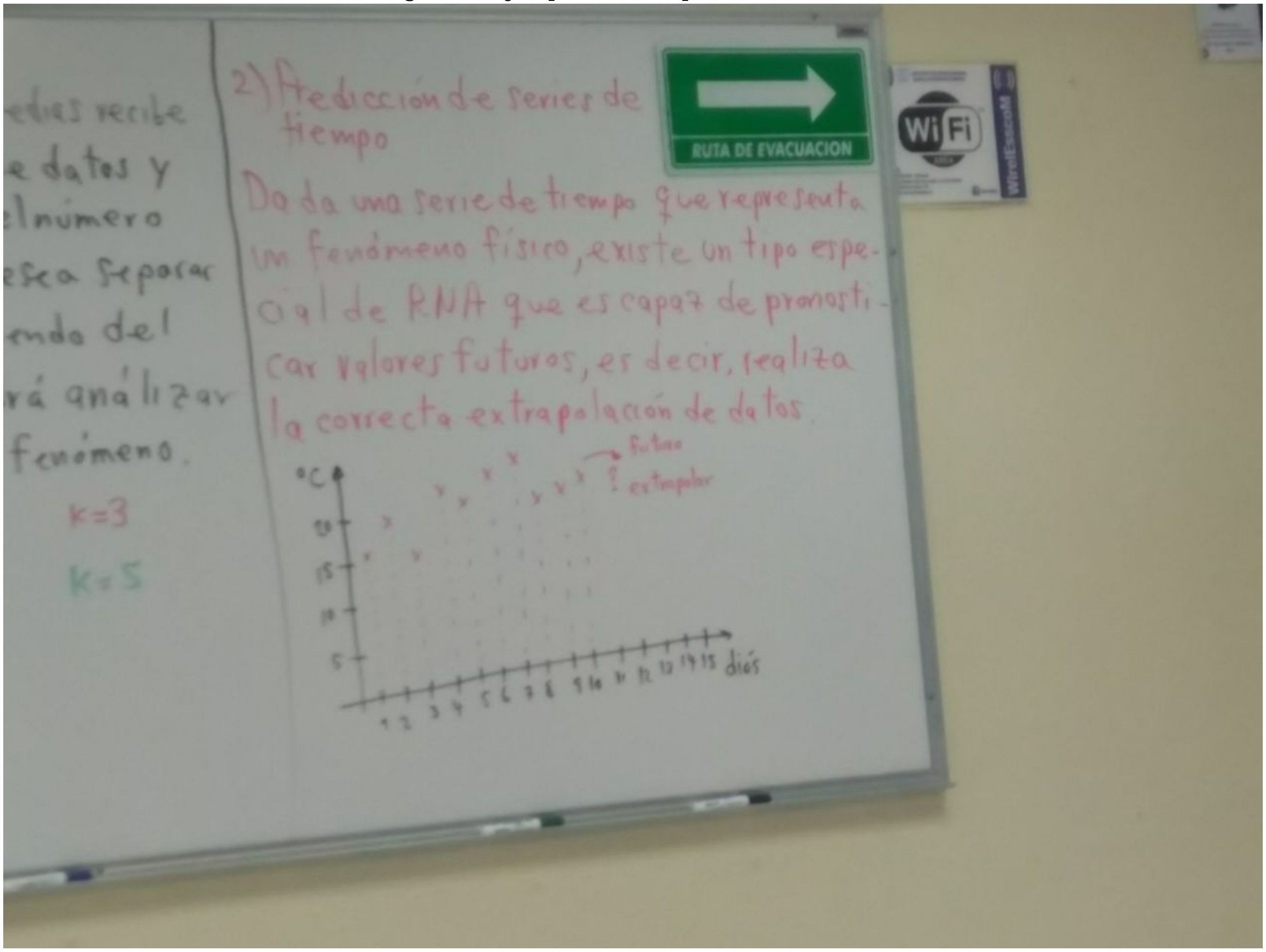
2.1. Aprendizaje Supervisado

Es aquel en el que se cuenta con ejemplos que contienen valores deseados(target) para llevar a cabo el ajuste de los parámetros de la RNA. En este caso se cuenta con un conjunto de datos(p, t);

2.2. Aprendizaje no Supervisado

No se cuenta con ejemplos que contengan valores deseados. Sin embargo, existen diversos algoritmos en esta clasificación que se usan para analizar y extraer información valiosa de una fenómeno, por ejemplo, el algoritmo $k - medias$ recibe como entrada un conjunto de datos y un valor k que representa el número de clases en los que se desea separar a dicho conjunto. Dependiendo del valor de k el usuario podrá analizar de diferentes maneras el fenómeno.

Figura 2: Ejemplo de extrapolación



3. Predicción de Series de tiempo

Dada una serie de tiempo que representa un fenómeno físico, existe un tiempo especial de RNA que es capaz de pronosticar valores futuros, es decir, realiza la correcta extrapolación de datos. Para esta tarea se hace uso de las RNA's recurrrentes.

4. Control de Sistemas

Dado un sistema $f(t)$ se requiere modificar su comportamiento para que sea estable. En esta aplicación se usa una RNA para aproximar a $f(t)$ y una RNA para diseñar un controlador.

Figura 3: Red Feed Foward

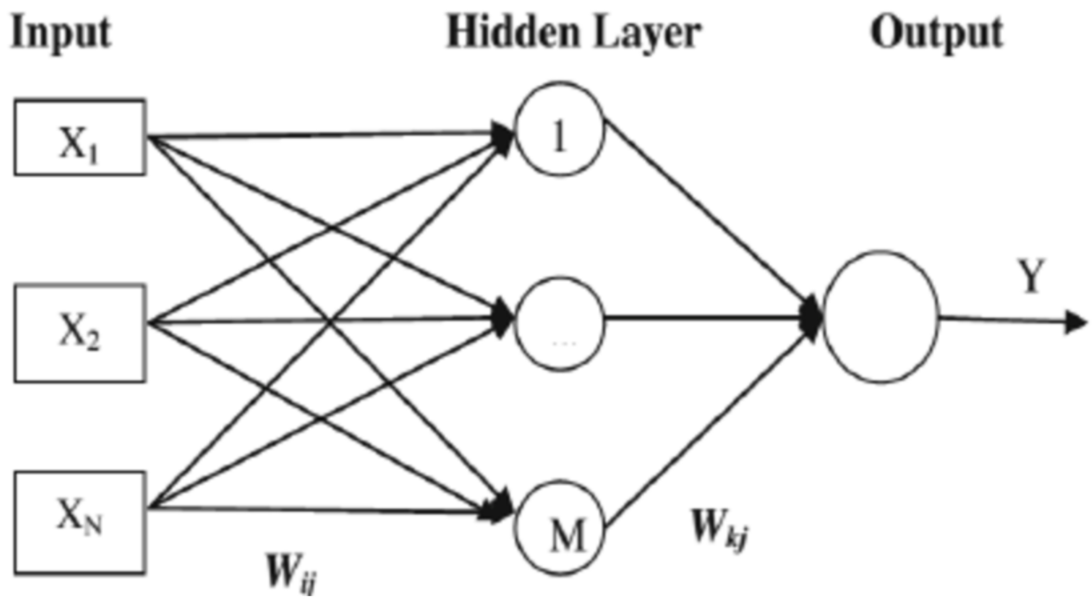


Figura 4: Red Recurrente

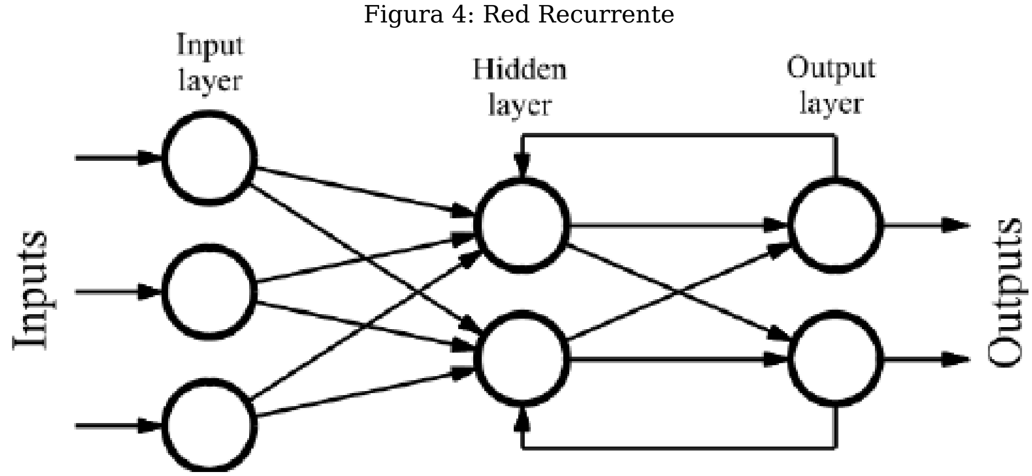


Figura 5: Red Recurrente con bloques recurrentes

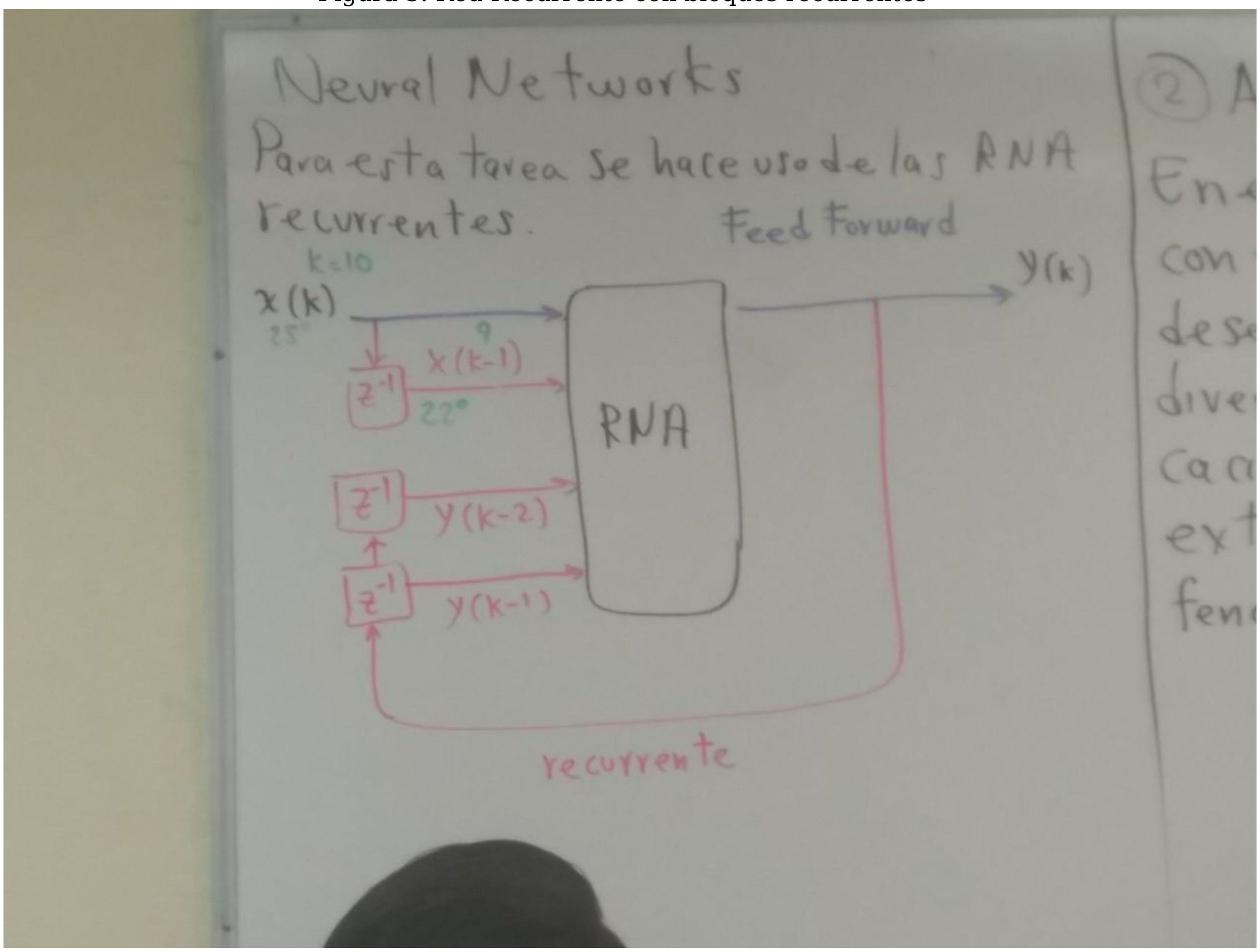
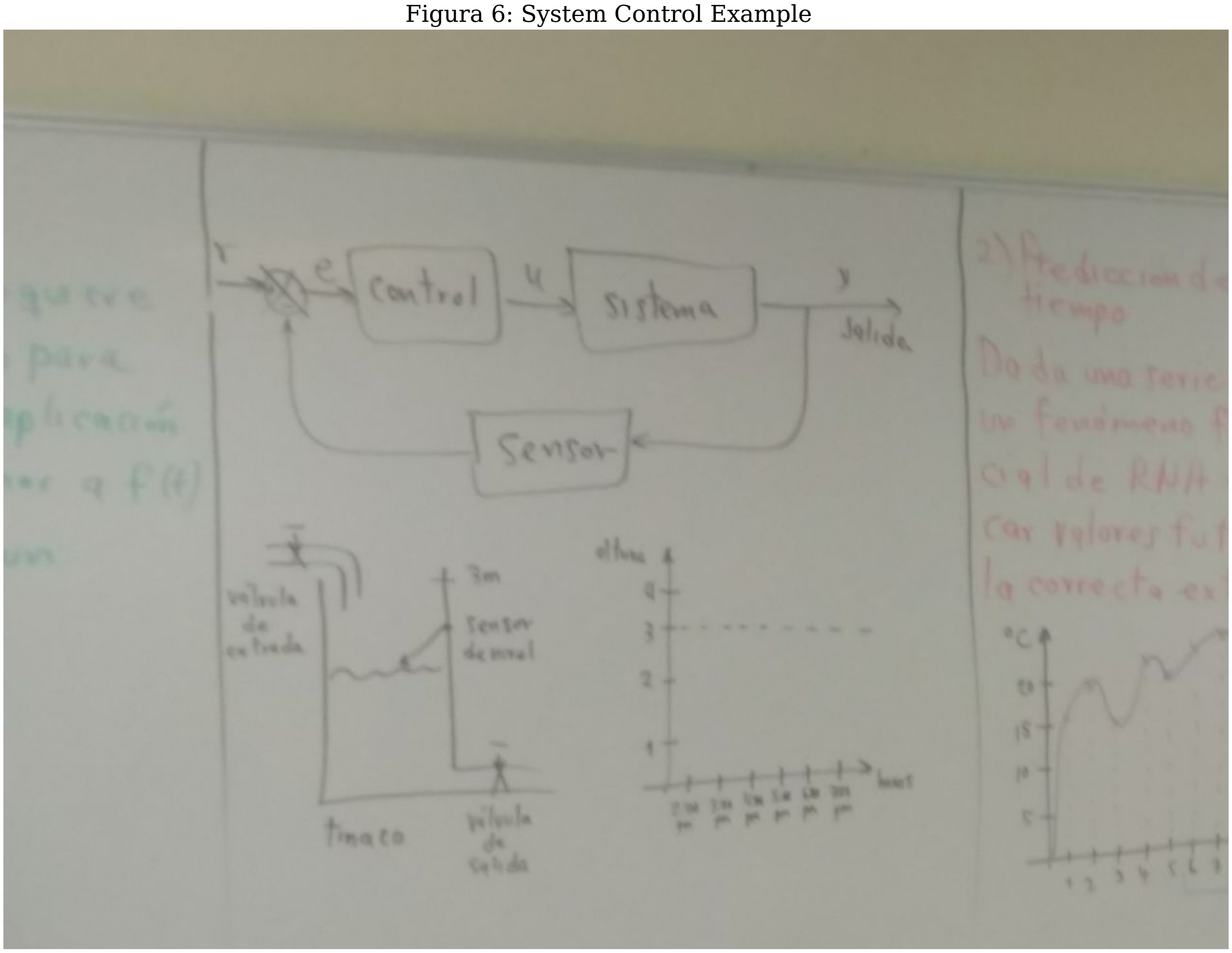


Figura 6: System Control Example



Parte II
Aplicaciones de la RNA

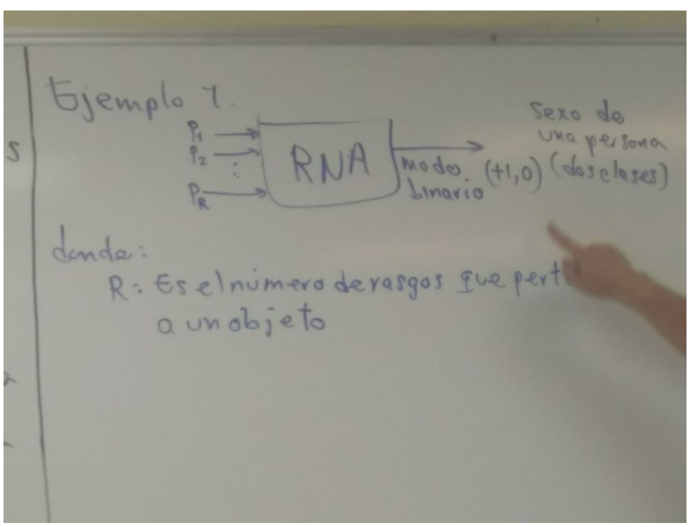
5. Reporte de Práctica

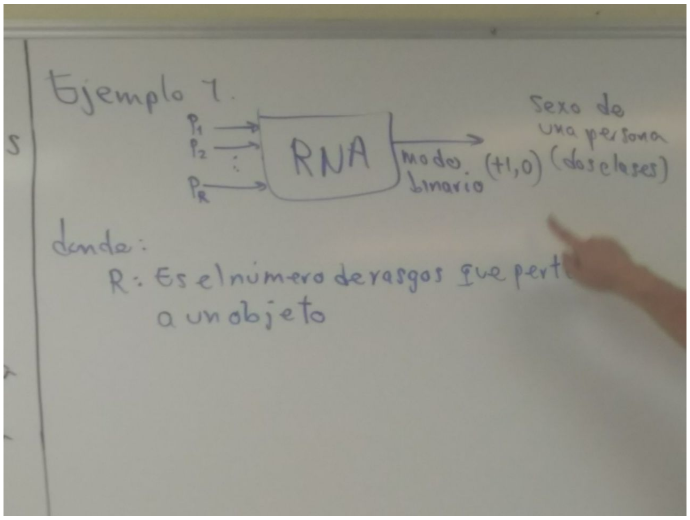
1. Intro
2. Metodología
3. Pseudo código
4. Resultados(Discusión)
5. Conclusiones
6. Referencias(mínimo 2)

6. Clasificación de Objetos

Consiste en tener una adecuada separación de un conjunto de objetos mediante una función de semejanza.

1. Clasificación supervisada:
   Aquí se cuenta con un conjunto de objetos y se conoce a que clase pertenence cada uno de ellos. Una RNA en modo clasificador puede usar una o más salidas para represtnar a que calse considera que un dato de entrada pertenece.
   Aquí tenemos targets ya que tenemos un conjunto de clases a las que este se puede etiquetar.
   Las redes pueden ser entrenadad no supervisadamente y supervisadamente
   Ejemplos:



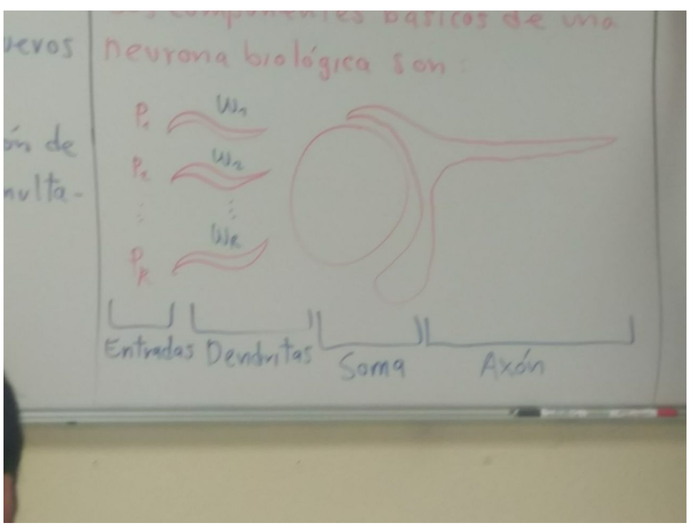


7. Neurona Biológica

Los sistemas biolgcos son tan complejos que se han convertido en una excelente fuente de inspiración para diseñar sistemas artificiales, que emulen algunas de sus características, entre ellas se encuentras las siguientes

1. No siempre requieren de módulos de referencia (targets)
2. Se desempeñan exitosamente ante incertidumbres
3. Se adaptan fácilmente a nuevos ambientes
4. Pueden Procesar información de diversas fuentes en forma simultánea

Dado que las RNA nacieron de los elementos básicos de una neurona biológica, es bioinspirada.
Los componentes básicos de una neurona biológica son:



donde:
$P_1, \ldots, P_r$: Son Entradas

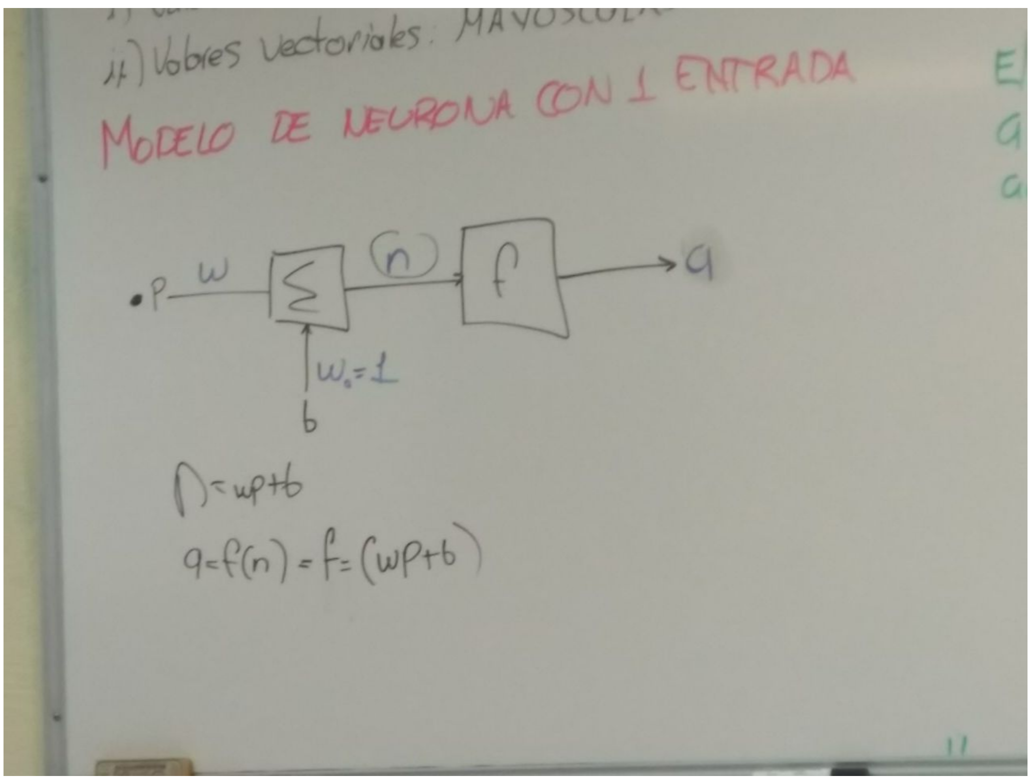
$W_1, \ldots, W_r$: Son los pesos sinápticos
Los pesos sinápticos se usan para indicar el nivel de importancia de cada una de las conexiones para una tarea particular.
El soma se encarga de acumular energía y determinar cuando se generará una señal de salida.
Nota: Para un target o tarea deseada, aprendizaje es buescar un conjutno de W_1 a W_r tales que la salida de la red desea igual al target, para todos los datos del conjutno de entramiento

Parte III
Arquitecturas de Redes

8. Notación

- Valores escalaras: minúsculas
- Valores vectoriales: Mayúsculas

9. Modelo de neurona con una entrada

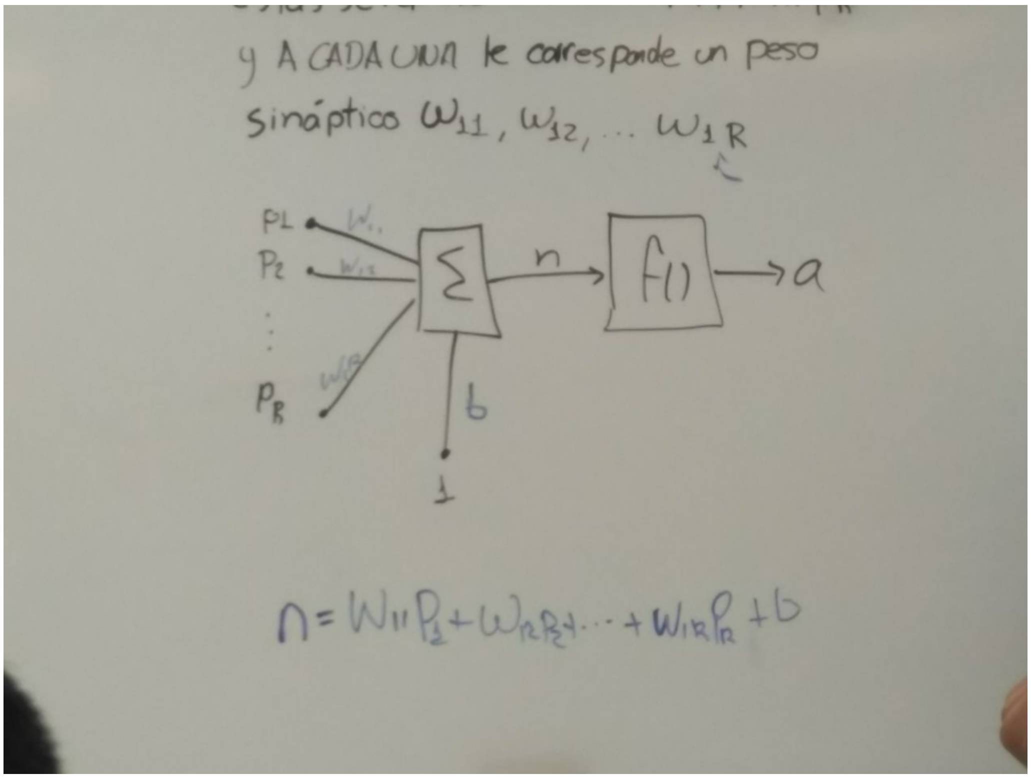


El bias es una entrada artificial que por definición tendrá siempre un peso sináptico de 1.
El bias como parámetro extra, le permite a la red resolver problemas más complejos, aunque existen RNA sin bias.

Ejemplo:
Siendo p=2, w=3 y b=-1.5, sustituya en el modelo de la neurona con 1 entrada.
$a = f(4{,}5)$

10. Neurona con múltiples entradas

Típicamente, para resolver un problema, una neurona tiene más de una entrada. Estas se representan como $p_1, p_2, p_3, \ldots p_r$ y a cada una le corresponde un peso sináptico $w_{11}, w_{12}, \ldots, w_{1r}$



En forma matricial, se puede expresar:
$n = W * P + b$
Donde n es un escalar
W es una matriz de [1Xr]
P es una matriz de [rx1]
b es una escalar
R es el número de entradas
1 es el número de neuronas

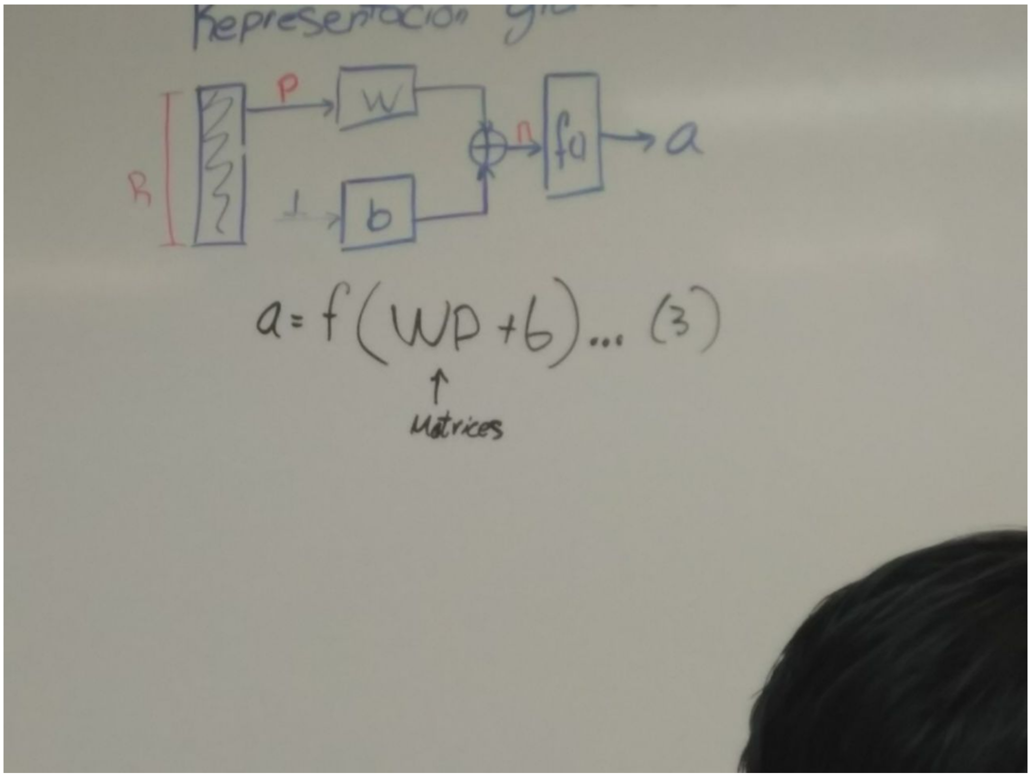
Finalmente, la salida de la red es:
$a = f(WP + b)$
Nota:
Para este caso particular, la matriz de pesos W es un vecotr fila, ya que representa el número de neuronas en cada capa de la RNA

10.1. Índices de la matriz W

Durante el curso, se utilizó la siguiente convención para los índices de la matriz de pesos:

1. El primer índice, se refiere a la Neurona a la que llega a la conexión.
2. El segundo índice indica el número de la **Fuente** de la que proviene el dato

10.2. Representación Gráfica matricial



11. Múltiples neuronas en Paralelo con múltiples entradas(Una capa)

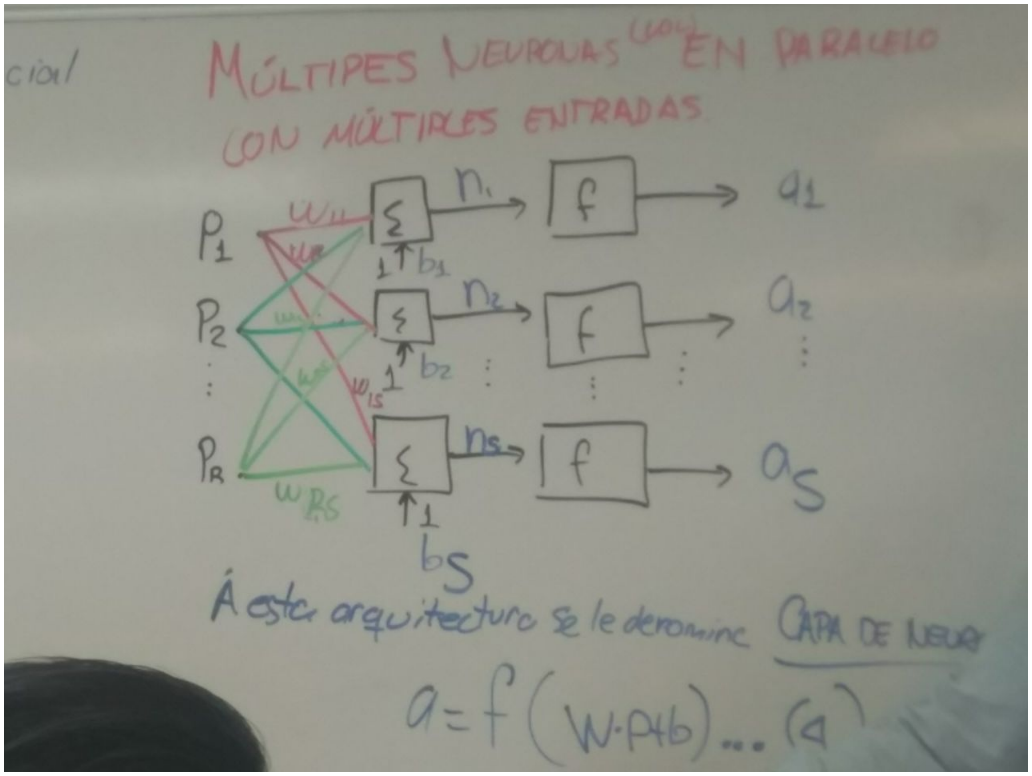


Figura 7: Arquitectura "Capa de neuronas"

$a = f(w * p + b)$
donde las dimensiones de w son: SxR
las dimensiones de P son: RxS
las dimensiones de b son Sx1

11.1. Diagrama Simplificado

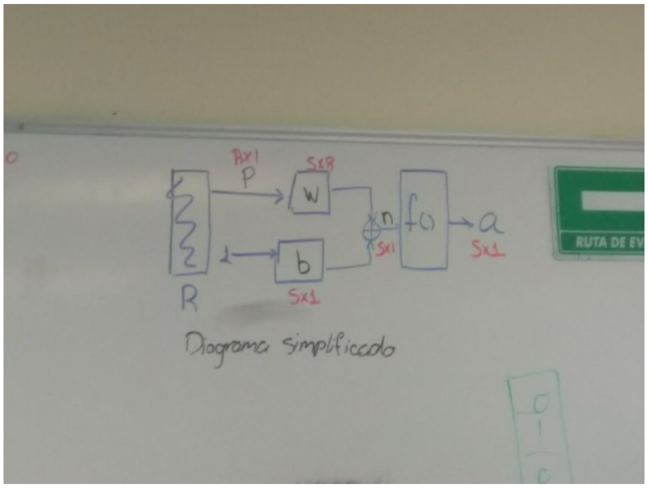


Figura 8: Arquitectura “Capa de neuronas” en diagrama simplificado

11.2. Ejemplo con s = 1

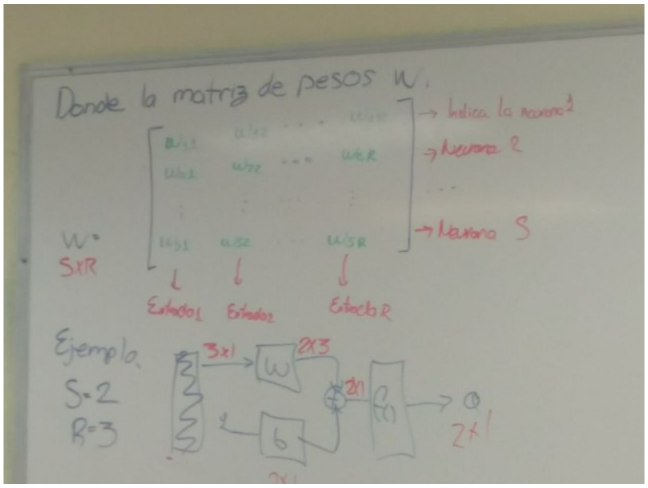


Figura 9: Matriz de pesos W_i

11.2.1. Nota acerca de las capas

Algunos autores llaman al vector de entrada P “Capa de entrada”.
Sin embargo, como en ella no se llevan a cabo operaciones, durante el curso **NO LO HAREMOS**

- Capas Ocultas: Se les llama así porque no tiene contacto con el exterior(Hidden Layers)
- Capa de Salida: Es aquella que entrega el resultado

12. Multiples Capas de Neuronas

Ahora se considerará una RNA con varias entradas y múltiples capas. Cada capa tiene su propia matriz de pesos w, vector de bias b y vector de salida a. Para distinguir entre cada capa, agregamos un superíndice o la variable.

$$a^1 = f^1(w_s^1 p_r^1 b_s^1)$$

$$a^2 = f^2(w^2 a^1 + b_s^2)$$

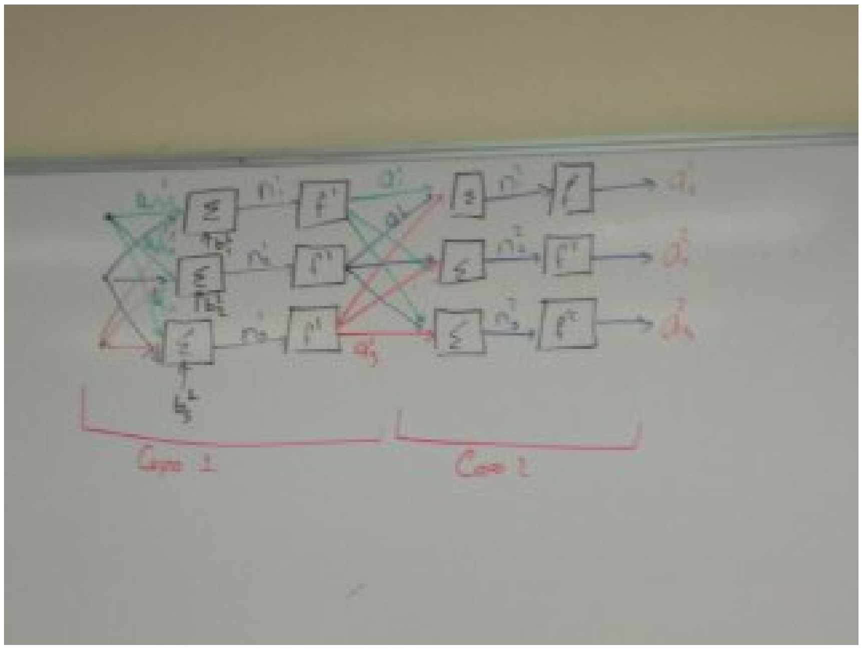


Figura 10: Modelo Arcoiris

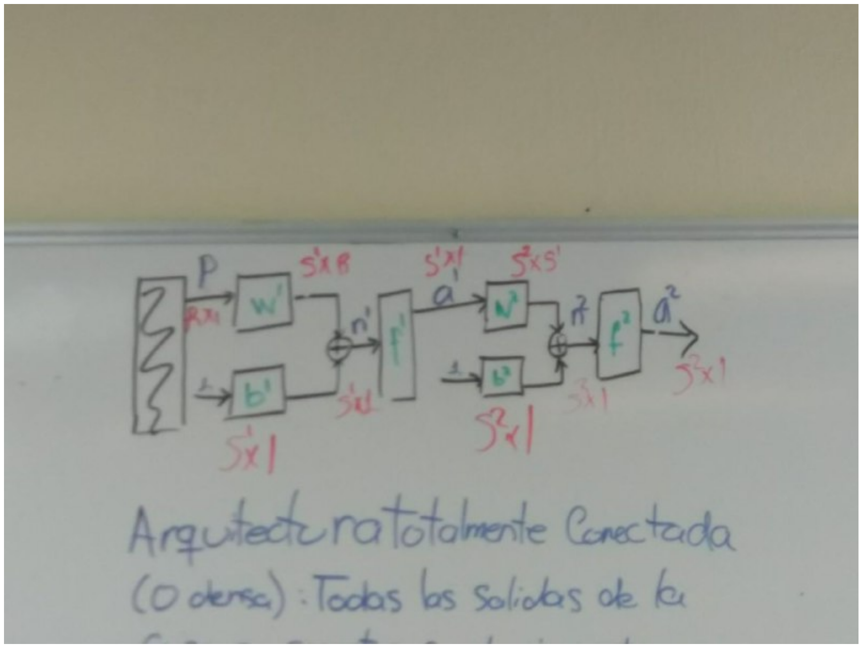


Figura 11: Capa del modelo arcoiris

13. Arquitectura totalmente conectada (O densa)

$$a^2 = f^2[w^2 f^1(w^1 p + b^1) + b^2]$$

Para representar una arquitectura multicapa, (MLP-Multilayer-Perceptron) se utiliza notación:

$$[R\ S^1\ S^2\ S^3 \ldots\ S^m]$$

Donde:
R: Es el número de entradas
M: Es el número de capas.
$S^1 \ldots\ S^m$: El número de neuronas en cada capa

13.1. Ejemplo

[4 2 5 1]
$[P]$ = 4x1
$[w^1]$ = 2x4
$[w^2]$ = 5x2
$[w^3]$ = 1x5
$[b^1]$ = 2x1
$[b^2]$ = 5x1
$[b^3]$ = 1x1

14. Multilayer Perceptron (MLP)

La arquitectura mínima de un MLP es $[R\ S^1\ S^2]$, pero ¿Cómo seleccionas la arquitectura?
Debemos partir de las especificaciones del problema, ello nos permite definir lo siguiente:

1. El número de entradas (R) al MLP es igual al número de rasgos o variables usados en el problema
2. El número de neuronas en la capa de salida es igual al número de clases definida en el problema
3. El tipo de función de activación depende de las consideraciones del problema
4. En cuanto al número de capas ocultas y al número de neuronas en cada una de ellas, sigue siendo un problema abierto

14.1. Ejemplo

Determinemos la arquitectura que clasifique los siguientes objetos en 2 clases

$$O_1 = \begin{cases} R_1 \\ R_2 \\ R_3 \end{cases} O_2 = \begin{cases} R_1 \\ R_2 \\ R_3 \end{cases} O_3 = \begin{cases} R_1 \\ R_2 \\ R_3 \end{cases} O_4 = \begin{cases} R_1 \\ R_2 \\ R_3 \end{cases}$$

[3 2]

14.2. Uso del MLP

- Para problemas de aproximación de señales, es común pensar que la dimensión de la señal de entrada Sea R=1. Se usan 1 o 2 capas ocultas y una neurona de la capa de salida

15. Funciones de transferencia

15.1. Función Hardlim

Esta función genera un cero como salida si n es menor a cero y uno si el valor de n es mayor o igual a cero($n \geq 0$).
Esta función se usa para crear neuronas capaces de clasificarlas entradas en dos clases.

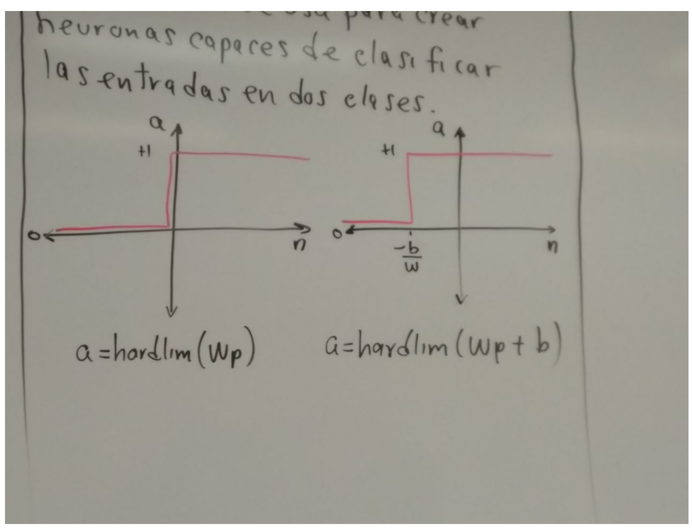


Figura 12: Función HardLim

15.2. Función Lineal

En esta función la salida es igual a la entrada

$$a = n$$

Este tipo de función se usa para trareas de regresión(aproximación de señales), por ejemplo la red ADALINE

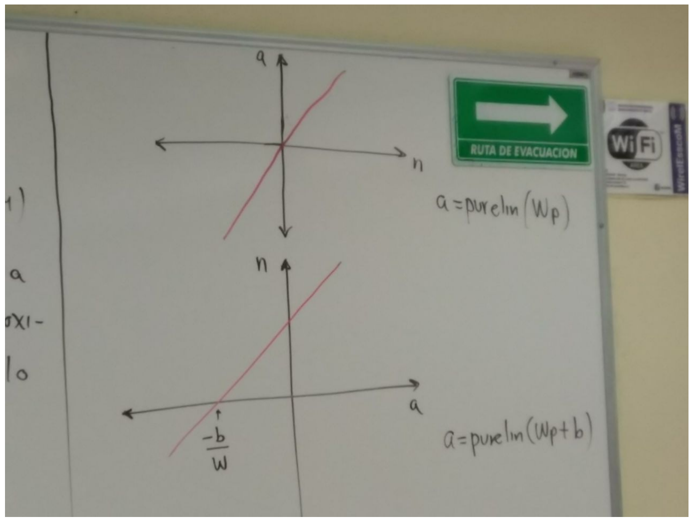


Figura 13: Función PureLim

15.2.1. Ejemplo

Aproximación de señales
Conjuntos: entrenamiento, aprendizaje, validación y prueba

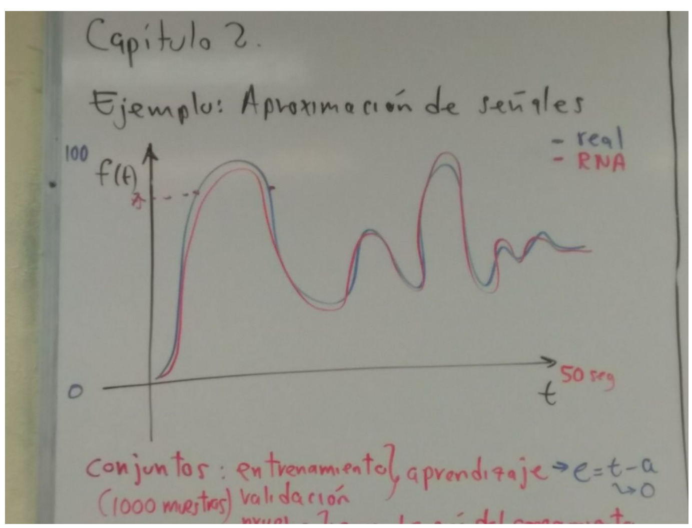


Figura 14: Ejemplo de aproximación de señales

Si logra interpolar bien, se dice que logró la generalización del conocimiento ya que error=t-a→ 0

15.3. Función logsigmoid

Esta función toma como entrada n que puede tener valores entre $-\text{ínf}$ y $+\text{ínf}$; y la reduce a una salida en el rando de 0 a 1, de acuerdo a la siguiente expresión

$$a = \frac{1}{1 + e^{-n}}$$

Esta función es usada comúnmente en redes neuronales multicapa que son entrenadas mediante el

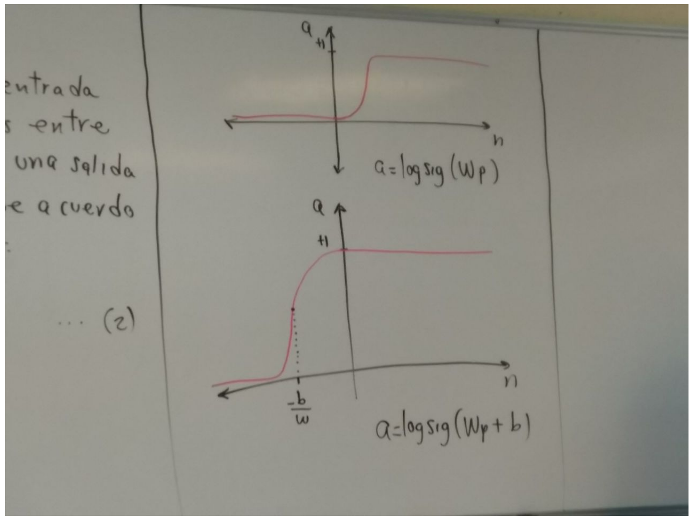


Figura 15: Función logsig

algoritmo backpropagation debdo a que esta técnica requiere que las funciones de transferencia de las capas ocultas sean no lineales, continuas y diferenciables.
La linea que separa las clases se llama frontera de desición

16. Red Hamming

Esta RNA fue diseñada explicitamente para resolver problemas de reconocimiento de patrones binarios (+1 o -1). Esta RNA es interesante, debido a que usa dos tipos de capas, una feed foward y otra recurrente. En hamming el mismo número de neuronas **AQUI VA UNA FOTO**

16.1. Observaciones

1. El número de neuronas en ambas capas, es el mismo, s
2. La capa recurrente no tiene bias
3. La recurrencia se lleva a cabo usando la señal de salida $a^2(t)$ como entrada a los pesos W^2

16.2. Capa FeedForward

Esta capa calcula la correlación o producto interno entre cada unos de los vecgores protitipo y e patrón de entrada. Con este objetivo, as filas de la matriz de pesos W_1, serán cada uno de los patrones

prototipo.
Para el ejemplo de clasificar una fruta en manzanas y naranjas, la matriz queda como:

$$W^1 = \begin{bmatrix} p_1^t \\ p_2^t \end{bmatrix} = \begin{bmatrix} +1 & -1 & -1 \\ +1 & +1 & -1 \end{bmatrix}$$

Nota: Naraja es el primer renglón y manzana el segundo
El valor del bias,b^1 es igual a R. Para nuestro ejemplo:
donde $S = 2$, ya que es igual al número de patrones prototipo. **Nota: Al agregar el valor R, se garantiza que las salidas de esta capa no sean negativas, lo cual es necesario para el correcto funcionamiento de la capa recurrente**

16.3. Capa Recurrente

Las neuronas de esta capa se inicializan con las salidas de la capa feed foward. En esta capa las neuronas compiten entre ellas para determinar a la ganadora.
Al final de la competencia, solo una neuronas de esta capa tendrás un valor diferente de cero.

La neurona ganadora indica a que clase pertence el vector de entrada. Las ecuaciones que describen esta capa son:

$$a^2(0) = a^1$$

$$a^2(t+1) = poslin(W^2 a^2(t))$$

La matriz de pesos tiene la siguiente forma para el ejemplo:

$$W^2 = \begin{bmatrix} 1 & -\epsilon \\ -\epsilon & 1 \end{bmatrix}$$

donde:
ϵ es un valor menor a $\frac{1}{S-1}$
S: es el número de neuronas en la capa recurrente

Para nuestro ejemplo, una iteración de la capa recurrente, se lleva a cabo con la siguiente ecuación:

$$a^2(t+1) = poslin(\begin{bmatrix} 1 & -\epsilon \\ -\epsilon & 1 \end{bmatrix} \begin{bmatrix} a_1^2(t) \\ a_2^2(t) \end{bmatrix})$$

Esta capa seguirá realizando iteraciones hasta que converja a una solución donde una sola de las neuronas tenga un valor diferente de cero y se reputa en dos iteraciones consecutivas.

16.4. Solución al problema de clasificación

Indique a que clase pertenece el siguiente vector de entrada usando la red de hamming

$$p = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Comenzamos calculando la capa feed foward esto se hace sollo una vez:

$$a^1 = W^1 p + b^1$$

$$a^1 = \begin{bmatrix} +1 & -1 & -1 \\ +1 & +1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

Ahora se pasa a la capa de recurrente

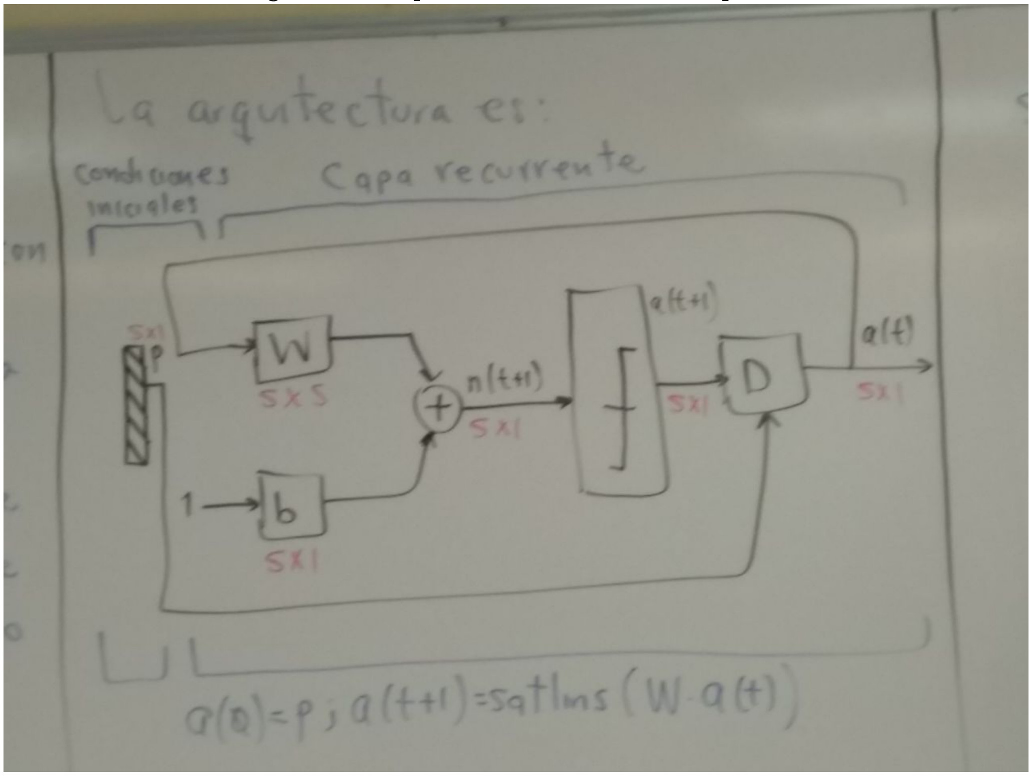
$$0 < \epsilon < \frac{1}{S-1} = 1$$

Se propone ϵ = 0.5
Se realza la primera iteración t = 0 $\rightarrow a^2(0) = a^1 =$

17. Red de Holfield

Esta RNA es recurrente y con una sola capa es capaz de realizar las operaciones de la red de Hamming. La red de Hamming no siempre va a converger.Esto sucede si dos o más vectores prototipo son equidistantes.

Figura 16: Arquitectura de la red de hopfiled



$$satlints = \begin{cases} a = -1 & n < -1 \\ a = n & -1 \leq n \leq +1 \\ a = +1 & n > +1 \end{cases}$$

Para el ejemplo de clasificación, se propone usar los siguientes valores:

$$W = \begin{bmatrix} 2. & 0 & 0 \\ 0 & 1{,}2 & 0 \\ 0 & 0 & 0{,}2 \end{bmatrix} ; b = \begin{bmatrix} 0{,}9 \\ 0 \\ -0{,}9 \end{bmatrix}$$

A continuación,se sustituyen estos valores en el modelo:

$$\begin{bmatrix} a_1(t+1) \\ a_2(t+1) \\ a_3(t+1) \end{bmatrix} = satlins(\begin{bmatrix} 2. & 0 & 0 \\ 0 & 1{,}2 & 0 \\ 0 & 0 & 0{,}2 \end{bmatrix} \begin{bmatrix} a_1(t) \\ a_2(t) \\ a_3(t) \end{bmatrix} + \begin{bmatrix} 0{,}9 \\ 0 \\ -0{,}9 \end{bmatrix})$$

Nota: Esta RNA realiza iteraciones hasta converger a uno de los vectores prototipo
Ejemplo: Diga a que calse pertenece el siguiente vector de entrada: $p^T = [-1 - 1 - 1]$ Solución: Se realiza la primera iteración
t = 0

$$\begin{bmatrix} a_1(1) \\ a_2(1) \\ a_3(1) \end{bmatrix} = satlins(\begin{bmatrix} 2. & 0 & 0 \\ 0 & 1{,}2 & 0 \\ 0 & 0 & 0{,}2 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 0{,}9 \\ 0 \\ -0{,}9 \end{bmatrix} = \begin{bmatrix} 0{,}7 \\ -1 \\ -1 \end{bmatrix})$$

t = 1

$$\begin{bmatrix} a_1(2) \\ a_2(2) \\ a_3(2) \end{bmatrix} = satlins(\begin{bmatrix} 2. & 0 & 0 \\ 0 & 1{,}2 & 0 \\ 0 & 0 & 0{,}2 \end{bmatrix} \begin{bmatrix} 0{,}7 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 0{,}9 \\ 0 \\ -0{,}9 \end{bmatrix} = \begin{bmatrix} +1 \\ -1 \\ -1 \end{bmatrix})$$

Converge al patrón prototipo de la naranja

Parte IV

Capítulo 4

18. Regla de Aprendizaje del perceptrón

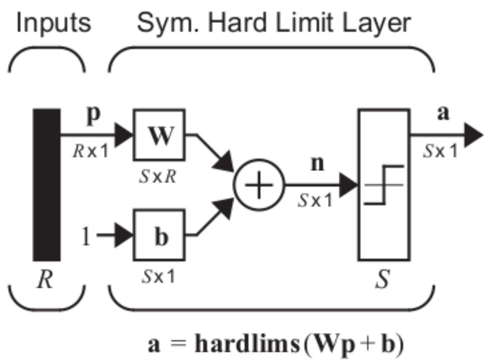
Una regla de aprendizaje es un procedimiento que de manera automática modifica los valores de los pesos y bias de una RNA. A este procedimiento también se conoce como algoritmo de entrenamiento.
Tomamos el caso de un aprendizaje supervisado por lo que se cuenta con un conjunto de entrenamiento qe indica cuál debe ser el comportamiento correcto de la RNA:

$$\{P_1, T_1\}, \{P_2, T_2\}, \ldots, \{P_q, T_q\}$$

19. Arquitectura del perceptrón

La forma general de un perceptrón es:

Figura 17: Arquitectura del perceptrón



Nota: el función de trasnferencia tambien puede ser HARDLIM()
Será útil paa el desarrollo de la regla de aprendizaje del perceptrón tener una referencia individual a cada uno de los elementos de la RNA. Primero, consideremos la matriz de pesos:

$$W = \begin{bmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,R} \\ w_{2,1} & w_{2,2} & \dots & w_{2,R} \\ \vdots & \vdots & \vdots & \vdots \\ w_{S,1} & w_{S,2} & \dots & w_{S,R} \end{bmatrix}$$

Las filas son las neuronas y las columnas las entradas, por lo tanto los elementos son los pesos entre ellas.
Se puede definir los elementos de un vector compuesto de los elementos de la i-ésima fila de W:

$${}_iW = \begin{bmatrix} w_{i,1} \\ w_{i,2} \\ \vdots \\ w_{i,R} \end{bmatrix}$$

Ahora se pueden representar la matriz de pesos de la siguiente manera:

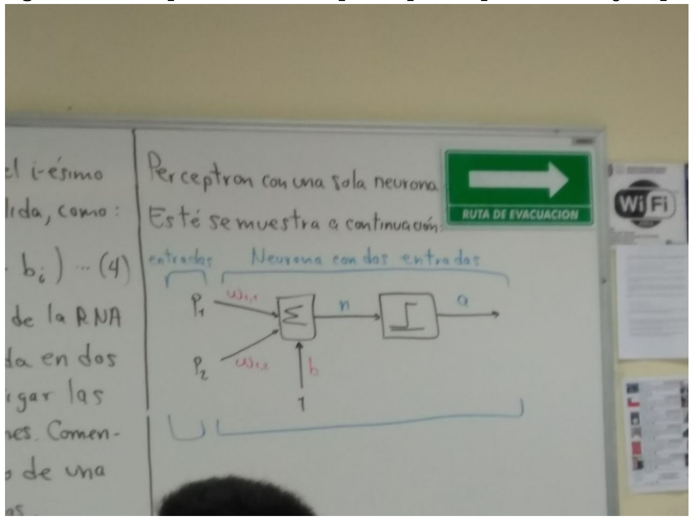
$$W = \begin{bmatrix} {}_1W^T \\ {}_2W^T \\ \vdots \\ {}_SW^T \end{bmatrix}$$

Esto nos permite escribir el i-ésimo elemento del vector de salida, como:

$$a_i = hardlim({}_iW^T p + b_i)$$

Recuerde que cada neurona de la RNA divide el espacio de entrada en dos regiones. Es valioso investigar las fornteras entre estas regiones. Comenzaremos con el sencillo caso de una sola neurona de dos

Figura 18: Arquitectura del perceptrón para este ejemplo



entradas.

El modelo matemático es:

$$a = hardlim(n)$$

$$a = hardlim(Wp + b)$$

$$a = hardlim({}_1W^T p + b)$$

$$a = hardlim(W_{1,1}P1 + W_{1,2}P2 + b)$$

20. Análisis de frontera de desición

Ésta frontera se determina por os vectores de entrada para los cuáles la señal n es igual a 0:

$$n = W_{1,1}P_1 + W_{1,2}P_2 + b = 0$$

20.1. Ejemplo

$$W_{1,1} = W_{1,2} = 1, b = -1$$

Sustituyendo en la fórmula:

$$n = P_1 + P_2 - 1 = 0$$

La ecuación anterior define una linea en el espacio. En un lado de la linea de salida de la RNA será cero y en otro lado será 1. Para dibujar la línea, se obtiene los puntos donde esta intersecta los ejes P_1 y P_2

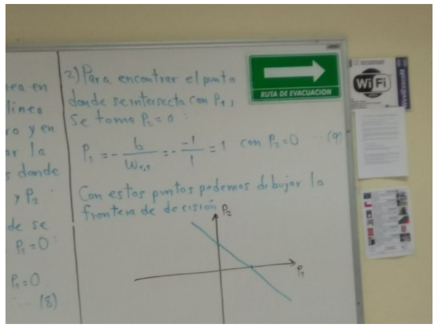
1. Para encontrar el putno donde intersecta con P_2, se toma $P_1 = 0$:

$$P_2 = -\frac{b}{W_{1,2}} = -\frac{-1}{+1} = 1, \text{con } P_1 = 0$$

2. Para encontrar el punto donde se intersecta con P_1 se toma $P_2 = 0$

$$P_1 = -\frac{b}{W_{1,1}} = -\frac{-1}{1} = 1 \text{ con } P_2 = 0$$

Con estos puntos podemos dibujar la frontera de desición



Para encontrar de que lado de la frontera corresponde a la salida 1, solo es necesario aplicar una entrada a la RNA, por ejemplo, $p = [2\ 0]^T$:

$$W = [1\ 1] \begin{bmatrix} 2 \\ 0 \end{bmatrix} - 1 = 1$$

Nota: El vector de pesos es perpendicular a la frontera de desición y apunta a la clase +1

Parte V

Capítulo 4

Ejercicio: Dado el siguiente problema de clasificación con cuatro clases, diseñe un perceptron que resuelve este problema.

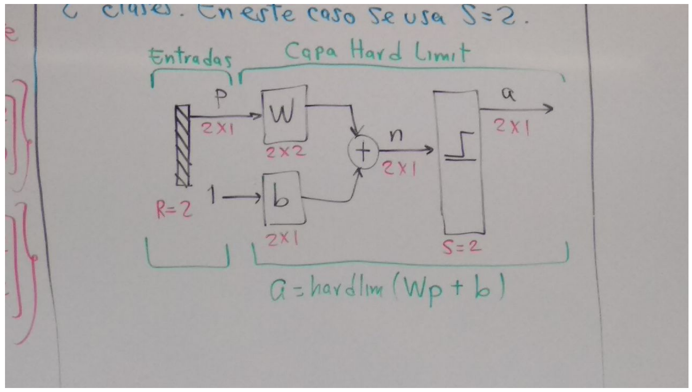
$$Clase1 = \{P1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}; P2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}\}$$

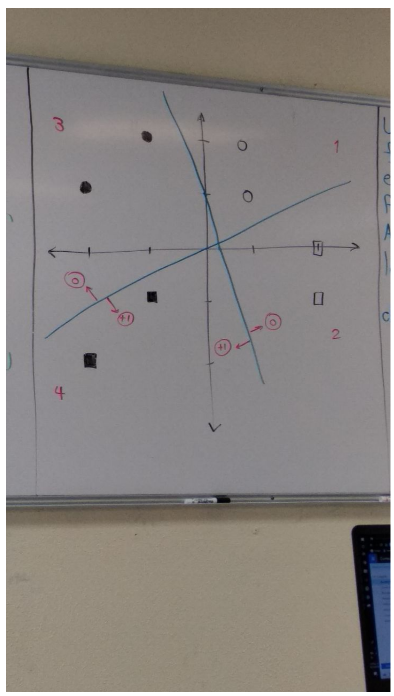
$$Clase2 = \{P3 = \begin{bmatrix} 2 \\ -1 \end{bmatrix}; P4 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}\}$$

$$Clase3 = \{P5 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}; P6 = \begin{bmatrix} -2 \\ 1 \end{bmatrix}\}$$

$$Clase4 = \{P1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}; P2 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}\}$$

Solución: Recordando, un perceptron con S neuronas, puede clasificar los datos en 2^S. En este caso se usa S=2





Una vez que se dibujan las fronteras de desición, se elije de que cada frontera de desición tendrá la clase +1. Así, ya es posible definir el valor de los valores deseadas para cada clase:

$$Clase1 = \{t1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}; t2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}\}$$

$$Clase2 = \{t3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; t4 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}\}$$

$$Clase3 = \{t5 = \begin{bmatrix}1\\0\end{bmatrix}; t6 = \begin{bmatrix}1\\0\end{bmatrix}\}$$

$$Clase4 = \{t7 = \begin{bmatrix}1\\1\end{bmatrix}; t8 = \begin{bmatrix}1\\1\end{bmatrix}\}$$

Ahora se proponene los vectores de pesos que cumplan las propiedades antes mecionadas, por ejemplo:

$$_1W = \begin{bmatrix}-3\\-11\end{bmatrix}$$

y

$$_1W = \begin{bmatrix}1\\-2\end{bmatrix}$$

Finalmente, se obtiene el valor de los bías:

$$_1b = -_1W^T p = [-3 - 1]\begin{bmatrix}0\\1\end{bmatrix} = 1$$

$$_2b = -_2W^T p = [1 - 2]\begin{bmatrix}0\\0\end{bmatrix} = 0$$