

Facultad: Ingeniería.

Escuela: Electrónica

Asignatura: Redes Neuronales

## Tema: Reconocimiento de caracteres.

### Contenidos

Aplicación de la red neuronal backpropagation al reconocimiento de caracteres.

### Objetivo Específico

Ser capaz de realizar el reconocimiento de caracteres usando una red neuronal estudiada antes.

### Materiales y Equipo

| No | Cantidad | Descripción  |
|----|----------|--|
| 1  | 1        | Computadora con sistema operativo Windows 95 o superior y el programa MATLAB 5.3 o superior. |

### Introducción Teórica

Se desea poder reconocer los 27 caracteres del alfabeto, definidos en una imagen de 7x5 píxeles. Para ello nos valemos de una red backpropagation de dos capas, la red tiene 35 entradas una por cada píxel; 10 neuronas en la segunda capa y 27 salidas una por cada letra. Esta red se entrena con las letras sin ruido y con ruido para lograr una buena clasificación. Se escoge la salida que genere el mayor valor como la que indica cual letra es la más parecida a la de entrada.

### Procedimiento

#### PARTE I. Definición del problema.

4. Encienda la computadora y corra el programa MATLAB.
5. Cree un archivo-m con el nombre ejemplo1.m que plantee el problema de reconocimiento de caracteres y entrene la red con el alfabeto sin ruido y con ruido.

```

% EJEMPLO1: ENTRENAMIENTO DE LA RED PARA EL RECONOCIMIENTO DE
CARACTERES
% Guía 9 de Redes Neuronales.
%
% Este archivo entrena una red de dos capas
%   sigmoidal-logarítmica/sigmoidal-logarítmica
%   para clasificar las letras en el mapa de bits. La
%   red tiene 35 entradas, 10 neuronas en la primera capa,
%   y 27 salidas.
% La red primero se entrena con las letras sin ruido
%   luego con las letras con ruido. Finalmente la red se
%   entrena de nuevo con las letras sin ruido.
% El resultado es una red que puede clasificar apropiadamente
%   las letras sin ruido y hacer un buen trabajo en la
%   clasificación de las letras con ruido.
% CARGA DEL PROBLEMA
%=====
defcarac;
% INICIALIZACIÓN DE LA ARQUITECTURA DE LA RED
%=====
[R,Q] = size(alfabeto); S1 = 10; [S2,Q] = size(deseadas);
red = newff(minmax(alfabeto),[S1,S2],{'logsig' 'logsig'},'traingdx');
red.inputweights{1,1}.initFcn = 'rands';
red.biases{1}.initFcn = 'rands';
red=init(red);
red.LW{2,1} = 0.01*red.LW{2,1}; red.b{2} = 0.01*red.b{2};
% ENTRENAMIENTO DE LA FUNCIÓN
%=====
% PROBLEMA
P = alfabeto;
T = deseadas;
% PARÁMETROS DE ENTRENAMIENTO
frec_present = 50;
max_iteracion = 5000;
meta_error = 0.01;
lr = 0.01; %Razón de aprendizaje
lr_inc = 1.05;
lr_dec = 0.7;
razon_error = 1.04;
red.performFcn = 'sse';
red.trainParam.show = frec_present;
red.trainParam.goal = meta_error;
red.trainParam.epochs = max_iteracion;
red.trainParam.lr = lr;
red.trainParam.lr_inc = lr_inc;
red.trainParam.lr_dec = lr_dec;
red.trainParam.max_perf_inc = razon_error;
red.trainParam.min_grad = 1e-20; % gradiente minimo
% ENTRENEAMIENTO SIN RUIDO
[red, tr]=train(red,P,T);
W1 = red.IW{1,1}; B1 = red.b{1};
W2 = red.LW{2,1}; B2 = red.b{2};
% GUARDADO DE LA RED ENTRENADA SIN RUIDO
%=====

```

## 128 Redes Neuronales, Guía 7

```
save ejel_w1.dat W1 /ascii /double
save ejel_b1.dat B1 /ascii /double
save ejel_w2.dat W2 /ascii /double
save ejel_b2.dat B2 /ascii /double
disp('pausa')
% PARÁMETROS DE ENTRENAMIENTO
max_iteracion = 300;
meta_error = 0.06;
red.trainParam.goal = meta_error;
red.trainParam.epochs = max_iteracion;
% ENTRENAMIENTO CON RUIDO
for paso = 1:10
    fprintf('Paso = %.0f\n',paso);
    P = [alfabeto, alfabeto, ...
        (alfabeto + randn(R,Q)*0.1), ...
        (alfabeto + randn(R,Q)*0.2)];
    T = [deseadas deseadas deseadas deseadas];
    [red, tr]=train(red,P,T);
end
% PROBLEMA
P = alfabeto;
T = deseadas;
% PARÁMETROS DE ENTRENAMIENTO
max_iteracion = 500;
meta_error = 0.01;
red.trainParam.goal = meta_error;
red.trainParam.epochs = max_iteracion;
% DE NUEVO ENTRENEAMIENTO SIN RUIDO
[red, tr]=train(red,P,T);
W1 = red.IW{1,1}; B1 = red.b{1};
W2 = red.LW{2,1}; B2 = red.b{2};
% GUARDADO DE LA RED ENTRENADA SIN RUIDO
%=====
save eje2_w1.dat W1 /ascii /double
save eje2_b1.dat B1 /ascii /double
save eje2_w2.dat W2 /ascii /double
save eje2_b2.dat B2 /ascii /double
% RESUMEN DE RESULTADOS
%=====
A = sim(red,P);
SSE = sumsqr(A-T);
fprintf('Error cuadrado medio final sin ruido: %g.\n',SSE);
```

Guarde el archivo con el nombre EJEMPLO1.M.

3. Para que el archivo anterior funcione se deben definir los caracteres del alfabeto. Escriba el siguiente archivo-m que define las imágenes de las 27 letras del alfabeto.

```
% DEFCARAC: DEFINICIÓN DEL PROBLEMA DE RECONOCIMIENTO DE
CARACTERES
% Guía 9 de Redes Neuronales.
%
% Este archivo define el mapa de bits para las 27 letras
del
```

```

% alfabeto. Cada letra se representac como un vector de
35-elementos
% de valores booleanos.
% También se definen 27 vectores de salida deseada.
% Cada vector de salida deseada tiene 27 elementos,
donde
% el vector de salida deseado que corresponden a la i-
ésima
% letra tiene un 1 en su i-ésimo elemento y ceros en sus
% otros 26 elementos.
letraA = [0 0 1 0 0 ...
          0 1 0 1 0 ...
          0 1 0 1 0 ...
          1 0 0 0 1 ...
          1 1 1 1 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ]';
letraB = [1 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 1 1 1 0 ]';
letraC = [0 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 0 ...
          1 0 0 0 0 ...
          1 0 0 0 0 ...
          1 0 0 0 1 ...
          0 1 1 1 0 ]';
letraD = [1 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 1 1 1 0 ]';
letraE = [1 1 1 1 1 ...
          1 0 0 0 0 ...
          1 0 0 0 0 ...
          1 1 1 1 0 ...
          1 0 0 0 0 ...
          1 0 0 0 0 ...
          1 1 1 1 1 ]';
letraF = [1 1 1 1 1 ...
          1 0 0 0 0 ...

```

```

1 0 0 0 0 ...
1 1 1 1 0 ...
1 0 0 0 0 ...
1 0 0 0 0 ...
1 0 0 0 0 ...
letraG = [0 1 1 1 0 ...
1 0 0 0 1 ...
1 0 0 0 0 ...
1 0 0 0 0 ...
1 0 0 1 1 ...
1 0 0 0 1 ...
0 1 1 1 0 ]';
letraH = [1 0 0 0 1 ...
1 0 0 0 1 ...
1 0 0 0 1 ...
1 1 1 1 1 ...
1 0 0 0 1 ...
1 0 0 0 1 ...
1 0 0 0 1 ]';
letraI = [0 1 1 1 0 ...
0 0 1 0 0 ...
0 0 1 0 0 ...
0 0 1 0 0 ...
0 0 1 0 0 ...
0 0 1 0 0 ...
0 1 1 1 0 ]';
letraJ = [1 1 1 1 1 ...
0 0 1 0 0 ...
0 0 1 0 0 ...
0 0 1 0 0 ...
0 0 1 0 0 ...
1 0 1 0 0 ...
0 1 0 0 0 ]';
letraK = [1 0 0 0 1 ...
1 0 0 1 0 ...
1 0 1 0 0 ...
1 1 0 0 0 ...
1 0 1 0 0 ...
1 0 0 1 0 ...
1 0 0 0 1 ]';
letraL = [1 0 0 0 0 ...
1 0 0 0 0 ...
1 0 0 0 0 ...
1 0 0 0 0 ...
1 0 0 0 0 ...
1 0 0 0 0 ...
1 1 1 1 1 ]';

```

```

letraM = [1 0 0 0 1 ...
          1 1 0 1 1 ...
          1 0 1 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ]';
letraN = [1 0 0 0 1 ...
          1 1 0 0 1 ...
          1 1 0 0 1 ...
          1 0 1 0 1 ...
          1 0 0 1 1 ...
          1 0 0 1 1 ...
          1 0 0 0 1 ]';
letraN2 = [1 1 1 1 1 ...
           0 0 0 0 0 ...
           1 0 0 0 1 ...
           1 1 0 0 1 ...
           1 0 1 0 1 ...
           1 0 0 1 1 ...
           1 0 0 0 1 ]';
letraO = [0 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          0 1 1 1 0 ]';
letraP = [1 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 1 1 1 0 ...
          1 0 0 0 0 ...
          1 0 0 0 0 ...
          1 0 0 0 0 ]';
letraQ = [0 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 1 0 1 ...
          1 0 0 1 0 ...
          0 1 1 0 1 ]';
letraR = [1 1 1 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 1 1 1 0 ...
          1 0 1 0 0 ...

```

```

      1 0 0 1 0 ...
      1 0 0 0 1 ]';
letraS = [0 1 1 1 0 ...
          1 0 0 0 1 ...
          0 1 0 0 0 ...
          0 0 1 0 0 ...
          0 0 0 1 0 ...
          1 0 0 0 1 ...
          0 1 1 1 0 ]';
letraT = [1 1 1 1 1 ...
          0 0 1 0 0 ...
          0 0 1 0 0 ...
          0 0 1 0 0 ...
          0 0 1 0 0 ...
          0 0 1 0 0 ...
          0 0 1 0 0 ]';
letraU = [1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          0 1 1 1 0 ]';
letraV = [1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          0 1 0 1 0 ...
          0 0 1 0 0 ]';
letraW = [1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ...
          1 0 1 0 1 ...
          1 1 0 1 1 ...
          1 0 0 0 1 ]';
letraX = [1 0 0 0 1 ...
          1 0 0 0 1 ...
          0 1 0 1 0 ...
          0 0 1 0 0 ...
          0 1 0 1 0 ...
          1 0 0 0 1 ...
          1 0 0 0 1 ]';
letraY = [1 0 0 0 1 ...
          1 0 0 0 1 ...
          0 1 0 1 0 ...

```

```

        0 0 1 0 0 ...
        0 0 1 0 0 ...
        0 0 1 0 0 ...
        0 0 1 0 0 ]';
letraZ = [1 1 1 1 1 ...
        0 0 0 0 1 ...
        0 0 0 1 0 ...
        0 0 1 0 0 ...
        0 1 0 0 0 ...
        1 0 0 0 0 ...
        1 1 1 1 1 ]';
alfabeto =
[letraA, letraB, letraC, letraD, letraE, letraF, letraG, letraH, .
..
letraI, letraJ, letraK, letraL, letraM, letraN, letraN2, letraO, .
..
letraP, letraQ, letraR, letraS, letraT, letraU, letraV, letraW, .
.
        letraX, letraY, letraZ];
deseadas = eye(27);

```

Guarde el archivo anterior con el nombre DEFCARAC.M.

## PARTE II. Prueba de la red.

4. Cree el siguiente archivo-m que prueba la red entrenada por EJEMPLO1.M para la clasificación apropiada de las letras.

```

% PRUEBEJ1: PRUEBA DE LA RED DE RECONOCIMIENTO DE
CARACTERES
% Guía 9 de Redes Neuronales.
%
% Este archivo prueba la habilidad de la red entrenada
% por EJEMPLO1 para una clasificación apropiada de las
letras con
% una cantidad de ruido variante.
% Los resultados se grafican también con los resultados
obtenidos
% al probar la red cuando sólo se ha entrenado con
letras con ruido.
% La red entrenada con letras con ruido, de hecho, no es
mucho mejor.
% CARGADO DEL PROBLEMA: alfabeto, deseadas
defcarac
[R,Q] = size(alfabeto); S1 = 10; [S2,Q] = size(deseadas);

```



**134 Redes Neuronales, Guía 7**

```
red = newff(minmax(alfabeto), [S1,S2], {'logsig'
'logsig'}, 'traingdx');
% CARGA LA RED 1: RED ENTRENADA CON LETRAS LIBRES DE
RUIDO.
load eje1_w1.dat
load eje1_b1.dat
load eje1_w2.dat
load eje1_b2.dat
% CARGA LA RED 2: RED ENTRENADA CON RUIDO Y LETRAS LIBRES
DE RUIDO.
load eje2_w1.dat
load eje2_b1.dat
load eje2_w2.dat
load eje2_b2.dat
% INICIALIZACIÓN DE VARIABLES DE PRUEBA
red1 = [];
red2 = [];
% COLOCACIÓN DE LOS PARÁMETROS DE PRUEBA
prueba_max = 100;
rango_ruido = 0:.05:.5;
% REALIZA LA PRUEBA
for nivelruido = rango_ruido
    fprintf('Prueba de la red con un nivel de ruido de
%.2f.\n', nivelruido);
    errores1 = 0;
    errores2 = 0;
    for i=1:prueba_max
        P = alfabeto + randn(R,Q)*nivelruido;
        % PRUEBA DE LA RED 1
        red.IW{1,1} = eje1_w1; red.b{1} = eje1_b1;
        red.LW{2,1} = eje1_w2; red.b{2} = eje1_b2;
        A = sim(red,P);
        AA = compet(A);
        errores1 = errores1 + sum(sum(abs(AA-deseadas)))/2;
        % PRUEBA DE LA RED 2
        red.IW{1,1} = eje2_w1; red.b{1} = eje2_b1;
        red.LW{2,1} = eje2_w2; red.b{2} = eje2_b2;
        A = sim(red,P);
        AA = compet(A);
        errores2 = errores2 + sum(sum(abs(AA-deseadas)))/2;
    end
    % RESULTADOS ALMACENADOS
    red1 = [red1 errores1/prueba_max/Q];
    red2 = [red2 errores2/prueba_max/Q];
end
% PRESENTACIÓN DE RESULTADOS
plot(rango_ruido, red1*100, '--', rango_ruido, red2*100);
```

```

title('Porcentaje de Errores de reconocimiento');
xlabel('Nivel de Ruido');
ylabel('Red 1 \_ \_ (entrenada sin ruido)      Red 2 \_ \_ \_
(entrenada con ruido)');

```

### PARTE III. Visualización de la prueba.

5. Cree un archivo-m llamado EJEMPLO2.M que pruebe una letra cualquiera del alfabeto con ruido y presente gráficamente el resultado de la letra seleccionada por la red neuronal.

```

% EJEMPLO2.M Prueba con un carácter de EJEMPLO1

defcarac;
[R,Q] = size(alfabeto); S1 = 10; [S2,Q] = size(deseadas);
red      =      newff(minmax(alfabeto), [S1,S2], {'logsig'
'logsig'}, 'traingdx');
% LETRAS ENTRENADAS CON RUIDO
load eje2_w1.dat
load eje2_b1.dat
load eje2_w2.dat
load eje2_b2.dat
c=clock;
semilla=c(1)*c(2)*c(3)*c(4)*c(5);
rand('seed',semilla)
red.IW{1,1} = eje2_w1; red.b{1} = eje2_b1;
red.LW{2,1} = eje2_w2; red.b{2} = eje2_b2;
for i=1:10
    selector=compet(rand(27,1));
    letra=alfabeto*selector;
    plotchar(letra)
    title('letra deseada')
    disp('Presione cualquier tecla para continuar')
    pause
    P=letra+randn(35,1)*.3;
    plotchar(P)
    title('letra con ruido')
    disp('Presione cualquier tecla para continuar')
    pause
    A = sim(red,P);
    AA = compet(A);
    letra2=alfabeto*AA;
    plotchar(letra2)
    if letra2==letra
        title('letra reconocida correctamente')
    else
        title('letra reconocida incorrectamente')
    end
end



```

### 136 Redes Neuronales, Guía 7

```
end
disp('Presione cualquier tecla para continuar')
pause
end
```

6. Salga del programa y apague el equipo.

### Bibliografía

-  Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones, José Ramón Hilera González, Víctor José Martínez Hernando. Biblioteca UDB.
-  Redes neuronales. Algoritmos, aplicaciones y técnicas de programación. James A. Freeman, David M. Skapura. Biblioteca UDB.

Guía 7: Reconocimiento de caracteres.

Estudiante:

Máquina No:

Docente:

GL:

Fecha:

| EVALUACION                         |      |   |   |   |      |
|------------------------------------|------|---|---|---|------|
|                                    | %    | 1-4   | 5-7   | 8-10  | Nota |
| <b>CONOCIMIENTO</b>                | 10%  | Conocimiento deficiente de los fundamentos teóricos | Conocimiento y explicación incompleta de los fundamentos teóricos           | Conocimiento completo y explicación clara de los fundamentos teóricos               |      |
| <b>APLICACIÓN DEL CONOCIMIENTO</b> | 60%  | No puede reconocer cualquier número de caracteres.  | Puede reconocer el abecedario y los dígitos del 0 al 10.                    | Puede reconocer el abecedario en mayúsculas y minúsculas y los dígitos del 0 al 10. |      |
|                                    |      |   |   |   |      |
| <b>ACTITUD</b>                     | 30%  | No tiene actitud proactiva.                         | Actitud propositiva y con propuestas no aplicables al contenido de la guía. | Tiene actitud proactiva y sus propuestas son concretas.                             |      |
|                                    |      |   |   |   |      |
| <b>TOTAL</b>                       | 100% |   |   |   |      |