Inteligencia Artificial

Redes Neuronales – TP3.Punto3-Falken

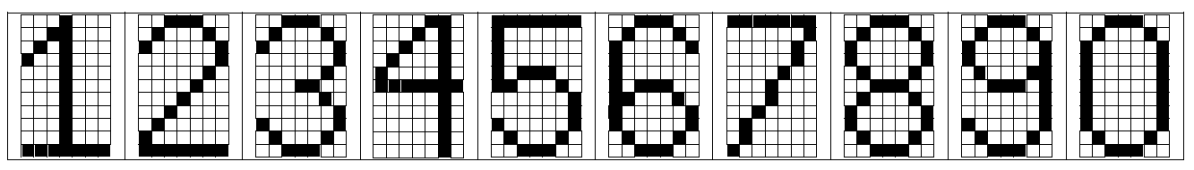
Se considera a las redes neuronales como modelos en hardware o en software que intentan reproducir el comportamiento del cerebro humano, con el propósito de resolver problemas. Desde otro punto de vista Es la interconexión en paralelo, de elementos de procesamiento unitarios, capaces de resolver problemas, luego de un proceso de aprendizaje.

	FACULTAD DE INGENIERÍA – UNJu – INTELIGENCIA ARTIFICIAL 2016		Inicio
	T.P. Nº 3 – Redes Neuronales		25/05/16
		Grupo: *Falken*	
Calificación	Nombre: *Tejerina, Guillermo Fernando*		Entrega
			01/06/16

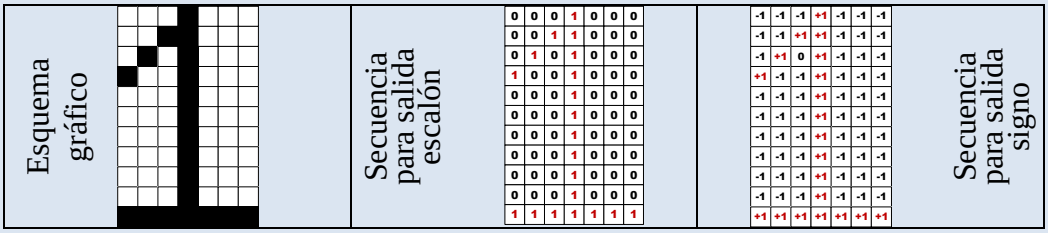
Problema 3 – Reconocimiento de patrones (3 ptos.)

Crear una red de Hopfield discreta, para trabajar como OCR, bajo las siguientes consideraciones:

- Utilizar como patrones de entrenamiento las secuencias siguientes, generadas sobre una plantilla de 11x7 pixeles.



- Puede elegir la salida discreta escalón o signo, de modo que los patrones de entrenamiento se configuran como secuencias de acuerdo a las figuras siguientes:



- El diseño e implementación puede ser realizado desde la línea de comando, con un script o utilizando la GUI *nntool*.
- Comprobar la operación de la red alimentando patrones trazados "a mano" sobre la misma plantilla.
- Describir los resultados obtenidos y de acuerdo a ellos, emitir conclusiones.

Nota: *puede facilitar la generación de las secuencias, utilizando Excel® que luego se pueden copiar y pegar en el editor de variables de Matlab, o pasarse como un archivo .csv, o copiar directamente en la línea de comandos sobre una variable (ej. >> var=[pegar]).*

Desarrollo del Ejercicio:

Red de Hopfield discreta para trabajar como OCR: El ejercicio 3 muestra la forma de usar Matlab y funciones de su toolbox de procesamiento de imágenes para reconocer en una imagen un numero o números.

Se usa la correlación para determinar la semenjanza de numeros de entrada con las plantilla. El tamaño de las letras debe ser mayor o igual a 11x7 pixeles, de tal forma que se ajuste al tamaño de la plantilla.

El Archivo interface.m dentro de la carpeta TP3-Punto3 preferentemente ubicada en el disco D://, es el encargado de desplegar la GUI del Ejercicio completo.

```
% Esta funcion interface es la encargada de desplegar la GUI del ejercicio completo.
function varargout = interface(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
```

```
                   'gui_OpeningFcn', @interface_OpeningFcn, ...
                   'gui_OutputFcn',  @interface_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function interface_OpeningFcn(hObject, ~, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
screen=imread('screen.jpg');
axes(handles.axes15);
imshow(screen);

% --- Outputs from this function are returned to the command line.
function varargout = interface_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton2.
function pushbutton11_Callback(hObject, eventdata, handles)
imagen=getimage;
figure(1)
imshow(imagen);
title('INPUT IMAGE WITH NOISE')

%% Convert to gray scale
if size(imagen,3)==3 % RGB image
    imagen=rgb2gray(imagen);
end

%% Convert to binary image
threshold = graythresh(imagen);
imagen =~im2bw(imagen,threshold);



%% Remove all object containing fewer than 30 pixels
imagen =bwareaopen(imagen,15);
pause(1)

%% Show image binary image
figure(2)
imshow(imagen);
title('INPUT IMAGE WITHOUT NOISE')

%% Edge detection
Iedge = edge(uint8(imagen));
imshow(~Iedge)
```

```
%% Morphology
% * *Image Dilation*
se = strel('square',3);
Iedge2 = imdilate(Iedge, se);
figure(3)
imshow(~Iedge2);
title('IMAGE DILATION')

% * *Image Filling*
Ifill= imfill(Iedge2,'holes');
figure(4)
imshow(~Ifill)
title('IMAGE FILLING')

Ifill=Ifill & imagen;
figure(5)
imshow(~Ifill);
re=Ifill;

while 1
    %Fcn 'lines' separate lines in text
    [fl re]=lines(re);
    imgn=fl;
    % Label and count connected components
    [L Ne] = bwlabel(imgn);
    set(handles.text11, 'String',Ne);

%% Objects extraction
axes(handles.axes5);
for n=1:Ne
    [r,c] = find(L==n);
    n1=imgn(min(r):max(r),min(c):max(c));
   %imshow(~n1);
    BW2 = bwmorph(n1,'thin',Inf);
    imrotate(BW2,0);
    imshow(~BW2);
    z=imresize(BW2,[50 50]);
    contents = get(handles.popupmenu5,'String');
  popupmenu5value = contents{get(handles.popupmenu5,'Value')};
  switch popupmenu5value
      case 'Salida discreta escalon'
        z=feature_extract(z);
      case 'Salida discreta signo'
        z=feature_extractor(z);
   end
    load ('D:\TP3-Punto3\featureout.mat');
    featureout=z;
    %disp(z);
    save ('D:\TP3-Punto3\featureout.mat','featureout');
    test
    pause(1.5);
end
if isempty(re)  %See variable 're' in Fcn 'lines'
        break
    end
end
clear all
winopen('D:\TP3-Punto3\output.txt');
close (gcbf)
interface
%set(handles.pushbutton9,'Enable','on')
```

```
% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --------------------------------------------------------------------
function Menu_Callback(hObject, eventdata, handles)
% hObject    handle to Menu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton6.
function pushbutton12_Callback(hObject, eventdata, handles)
%% reading the image from the user
[filename, pathname] = ...
     uigetfile({'*.jpg';'*.jpeg';'*.png';'*.*'},'Seleccione Imagen de Prueba');
 I=strcat(pathname,filename);
axes(handles.axes6);
imshow(I);
set(handles.pushbutton13,'Enable','on')
helpdlg('La Imagen fue cargada exitosamente. Proceda a entrenar la Red  ',...
        'Seleccionar Imagen');

% --- Executes during object creation, after setting all properties.
function axes3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
axis on
% Hint: place code in OpeningFcn to populate axes3

% --- Executes during object creation, after setting all properties.
function axes4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
axis on
% Hint: place code in OpeningFcn to populate axes4

function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double
disp(Ne);

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a double

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function text8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% --- Executes on button press in pushbutton10.
function pushbutton13_Callback(hObject, eventdata, handles)
contents = get(handles.popupmenu5,'String');
  popupmenu5value = contents{get(handles.popupmenu5,'Value')};
  switch popupmenu5value
      case 'Salida discreta escalon'
        train
        helpdlg('La Red fue entrenada exitosamente. Click en "Extraer Texto" para
procesar la imagen',...
        'Entrenamiento Exitoso');
      case 'Salida discreta signo'
        strain
        helpdlg('La Red fue entrenada exitosamente. Click en "Extraer Texto" para
procesar la imagen',...
        'Entrenamiento Exitoso');
  end
    set(handles.pushbutton11,'Enable','on')



% --- Executes on selection change in popupmenu4.
function popupmenu4_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu4 contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from popupmenu4

% --- Executes during object creation, after setting all properties.
function popupmenu4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --------------------------------------------------------------------
function Exit_Callback(hObject, eventdata, handles)
% hObject    handle to Exit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
conf=questdlg('Esta seguro que quiere salir','Exit Image','Si','No','No');
switch conf
    case 'Si'
        close(gcf)
    case 'No'
        return
end

% --------------------------------------------------------------------
function Help_Callback(hObject, eventdata, handles)
% hObject    handle to Help (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
open ReadMe.pdf

% --------------------------------------------------------------------
function About_us_Callback(hObject, eventdata, handles)
open aboutus.fig

% --- Executes on selection change in popupmenu5.
function popupmenu5_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu5 contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from popupmenu5

% --- Executes during object creation, after setting all properties.
function popupmenu5_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes when uipanel6 is resized.
function uipanel6_ResizeFcn(hObject, eventdata, handles)
% hObject    handle to uipanel6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes when figure1 is resized.
function figure1_ResizeFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on mouse press over figure background.
function figure1_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%//*****************************************************************************************************

function [features]=feature_extract(image);
original_image=image; % copia de seguridad de la imagen original
row=size(image,1);
column=size(image,2);
add_rows=0;
add_columns=0;
if row<9
    add_rows=9-row;
end
if column<9
    add_columns=9-column;
end

if mod(add_rows,2)==0
    image=[zeros(add_rows/2,column);image;zeros(add_rows/2,column)];
else
    image=[zeros((add_rows-1)/2,column);image;zeros((add_rows+1)/2,column)];
end
row=size(image,1);
if mod(add_columns,2)==0
    image=[zeros(row,(add_columns)/2),image,zeros(row,(add_columns)/2)];
else
    image=[zeros(row,(add_columns-1)/2),image,zeros(row,(add_columns+1)/2)];
end
column=size(image,2); % actualizar el valor de la columna
n_rows=ceil(row/3)*3-row;
n_columns=ceil(column/3)*3-column;

if mod(n_rows,2)==0
    image=[zeros(n_rows/2,column);image;zeros(n_rows/2,column)];
else
    image=[zeros((n_rows-1)/2,column);image;zeros((n_rows+1)/2,column)];
end

row=size(image,1);
if mod(n_columns,2)==0
    image=[zeros(row,(n_columns)/2),image,zeros(row,(n_columns)/2)];
else
    image=[zeros(row,(n_columns-1)/2),image,zeros(row,(n_columns+1)/2)];
end
column=size(image,2); % actualizar el valor de la columna
zone_height=row/3;
zone_width=column/3;

% esta imagen es punto en 11x7 pixeles, por las filas en cada zona deben ser 4, mientras
que las columnas deben ser 3.
% Esto se almacena en altura de la zona de variables y el ancho.
zone11=image(1:zone_height,1:zone_width);
zone12=image(1:zone_height,(zone_width+1):2*zone_width);
zone13=image(1:zone_height,(2*zone_width+1):end);

zone21=image((zone_height+1):2*zone_height,1:zone_width);
zone22=image((zone_height+1):2*zone_height,(zone_width+1):2*zone_width);
zone23=image((zone_height+1):2*zone_height,(2*zone_width+1):end);
```

```
zone31=image((2*zone_height+1):end,1:zone_width);
zone32=image((2*zone_height+1):end,(zone_width+1):2*zone_width);
zone33=image((2*zone_height+1):end,(2*zone_width+1):end);

% caracteristicas de los vectores
zone11_features=line_classifier(zone11);
zone12_features=line_classifier(zone12);
zone13_features=line_classifier(zone13);

zone21_features=line_classifier(zone21);
zone22_features=line_classifier(zone22);
zone23_features=line_classifier(zone23);

zone31_features=line_classifier(zone31);
zone32_features=line_classifier(zone32);
zone33_features=line_classifier(zone33);

% Euler que no se differencia entre ninguno de los objetos y los agujeros en la imagen.
euler=bweuler(image);
features=[zone11_features;zone12_features;zone13_features;zone21_features;zone22_features
;zone23_features;zone31_features;zone32_features;zone33_features];

%//*****************************************************************************************

x(:,1)= (feature_extract(~im2bw(a1)));
 x(:,2)= (feature_extract(~im2bw(a2)));
 x(:,3)= (feature_extract(~im2bw(a3)));
 x(:,4)= (feature_extract(~im2bw(a4)));
 x(:,5)= (feature_extract(~im2bw(a5)));
 x(:,6)= (feature_extract(~im2bw(a6)));
 x(:,7)= (feature_extract(~im2bw(a7)));
 x(:,8)= (feature_extract(~im2bw(a8)));
 x(:,9)= (feature_extract(~im2bw(a9)));
 x(:,10)= (feature_extract(~im2bw(a10)));
 x(:,11)= (feature_extract(~im2bw(a11)));
 x(:,12)= (feature_extract(~im2bw(a12)));
 x(:,13)= (feature_extract(~im2bw(a13)));
 x(:,14)= (feature_extract(~im2bw(a14)));
 x(:,15)= (feature_extract(~im2bw(a15)));
 x(:,16)= (feature_extract(~im2bw(a16)));
 x(:,17)= (feature_extract(~im2bw(a17)));
 x(:,18)= (feature_extract(~im2bw(a18)));
 x(:,19)= (feature_extract(~im2bw(a19)));
 x(:,20)= (feature_extract(~im2bw(a20)));
 x(:,21)= (feature_extract(~im2bw(a21)));
 x(:,22)= (feature_extract(~im2bw(a22)));
 x(:,23)= (feature_extract(~im2bw(a23)));
 x(:,24)= (feature_extract(~im2bw(a24)));
 x(:,25)= (feature_extract(~im2bw(a25)));
 x(:,26)= (feature_extract(~im2bw(b1)));
 x(:,27)= (feature_extract(~im2bw(b2)));
 x(:,28)= (feature_extract(~im2bw(b3)));
 x(:,29)= (feature_extract(~im2bw(b4)));
 x(:,30)= (feature_extract(~im2bw(b5)));
 x(:,31)= (feature_extract(~im2bw(b6)));
 x(:,32)= (feature_extract(~im2bw(b7)));
 x(:,33)= (feature_extract(~im2bw(b8)));
 x(:,34)= (feature_extract(~im2bw(b9)));
 x(:,35)= (feature_extract(~im2bw(b10)));
 x(:,36)= (feature_extract(~im2bw(b11)));
 x(:,37)= (feature_extract(~im2bw(b12)));
```

```
x(:,38)= (feature_extract(~im2bw(b13)));
x(:,39)= (feature_extract(~im2bw(b14)));
x(:,40)= (feature_extract(~im2bw(b15)));
x(:,41)= (feature_extract(~im2bw(b16)));
x(:,42)= (feature_extract(~im2bw(b17)));
x(:,43)= (feature_extract(~im2bw(b18)));
x(:,44)= (feature_extract(~im2bw(b19)));
x(:,45)= (feature_extract(~im2bw(b20)));
x(:,46)= (feature_extract(~im2bw(b21)));
x(:,47)= (feature_extract(~im2bw(b22)));
x(:,48)= (feature_extract(~im2bw(b23)));
x(:,49)= (feature_extract(~im2bw(b24)));
x(:,50)= (feature_extract(~im2bw(b25)));
x(:,51)= (feature_extract(~im2bw(c1)));
x(:,52)= (feature_extract(~im2bw(c2)));
x(:,53)= (feature_extract(~im2bw(c3)));
x(:,54)= (feature_extract(~im2bw(c4)));
x(:,55)= (feature_extract(~im2bw(c5)));
x(:,56)= (feature_extract(~im2bw(c6)));
x(:,57)= (feature_extract(~im2bw(c7)));
x(:,58)= (feature_extract(~im2bw(c8)));
x(:,59)= (feature_extract(~im2bw(c9)));
x(:,60)= (feature_extract(~im2bw(c10)));
x(:,61)= (feature_extract(~im2bw(c11)));
x(:,62)= (feature_extract(~im2bw(c12)));
x(:,63)= (feature_extract(~im2bw(c13)));
x(:,64)= (feature_extract(~im2bw(c14)));
x(:,65)= (feature_extract(~im2bw(c15)));
x(:,66)= (feature_extract(~im2bw(c16)));
x(:,67)= (feature_extract(~im2bw(c17)));
x(:,68)= (feature_extract(~im2bw(c18)));
x(:,69)= (feature_extract(~im2bw(c19)));
x(:,70)= (feature_extract(~im2bw(c20)));
x(:,71)= (feature_extract(~im2bw(c21)));
x(:,72)= (feature_extract(~im2bw(c22)));
x(:,73)= (feature_extract(~im2bw(c23)));
x(:,74)= (feature_extract(~im2bw(c24)));
x(:,75)= (feature_extract(~im2bw(c25)));
x(:,76)= (feature_extract(~im2bw(d1)));
x(:,77)= (feature_extract(~im2bw(d2)));
x(:,78)= (feature_extract(~im2bw(d3)));
x(:,79)= (feature_extract(~im2bw(d4)));
x(:,80)= (feature_extract(~im2bw(d5)));
x(:,81)= (feature_extract(~im2bw(d6)));
x(:,82)= (feature_extract(~im2bw(d7)));
x(:,83)= (feature_extract(~im2bw(d8)));
x(:,84)= (feature_extract(~im2bw(d9)));
x(:,85)= (feature_extract(~im2bw(d10)));
x(:,86)= (feature_extract(~im2bw(d11)));
x(:,87)= (feature_extract(~im2bw(d12)));
x(:,88)= (feature_extract(~im2bw(d13)));
x(:,89)= (feature_extract(~im2bw(d14)));
x(:,90)= (feature_extract(~im2bw(d15)));
x(:,91)= (feature_extract(~im2bw(d16)));
x(:,92)= (feature_extract(~im2bw(d17)));
x(:,93)= (feature_extract(~im2bw(d18)));
x(:,94)= (feature_extract(~im2bw(d19)));
x(:,95)= (feature_extract(~im2bw(d20)));
x(:,96)= (feature_extract(~im2bw(d21)));
x(:,97)= (feature_extract(~im2bw(d22)));
x(:,98)= (feature_extract(~im2bw(d23)));
x(:,99)= (feature_extract(~im2bw(d24)));
x(:,100)= (feature_extract(~im2bw(d25)));
```

```
 x(:,101)= (feature_extract(~im2bw(e1)));
x(:,102)= (feature_extract(~im2bw(e2)));
x(:,103)= (feature_extract(~im2bw(e3)));
x(:,104)= (feature_extract(~im2bw(e4)));
x(:,105)= (feature_extract(~im2bw(e5)));
x(:,106)= (feature_extract(~im2bw(e6)));
x(:,107)= (feature_extract(~im2bw(e7)));
x(:,108)= (feature_extract(~im2bw(e8)));
x(:,109)= (feature_extract(~im2bw(e9)));
x(:,110)= (feature_extract(~im2bw(e10)));
x(:,111)= (feature_extract(~im2bw(e11)));
x(:,112)= (feature_extract(~im2bw(e12)));
x(:,113)= (feature_extract(~im2bw(e13)));
x(:,114)= (feature_extract(~im2bw(e14)));
x(:,115)= (feature_extract(~im2bw(e15)));
x(:,116)= (feature_extract(~im2bw(e16)));
x(:,117)= (feature_extract(~im2bw(e17)));
x(:,118)= (feature_extract(~im2bw(e18)));
x(:,119)= (feature_extract(~im2bw(e19)));
x(:,120)= (feature_extract(~im2bw(e20)));
x(:,121)= (feature_extract(~im2bw(e21)));
x(:,122)= (feature_extract(~im2bw(e22)));
x(:,123)= (feature_extract(~im2bw(e23)));
x(:,124)= (feature_extract(~im2bw(e24)));
x(:,125)= (feature_extract(~im2bw(e25)));
 x(:,126)= (feature_extract(~im2bw(f1)));
x(:,127)= (feature_extract(~im2bw(f2)));
x(:,128)= (feature_extract(~im2bw(f3)));
x(:,129)= (feature_extract(~im2bw(f4)));
x(:,130)= (feature_extract(~im2bw(f5)));
x(:,131)= (feature_extract(~im2bw(f6)));
x(:,132)= (feature_extract(~im2bw(f7)));
x(:,133)= (feature_extract(~im2bw(f8)));
x(:,134)= (feature_extract(~im2bw(f9)));
x(:,135)= (feature_extract(~im2bw(f10)));
x(:,136)= (feature_extract(~im2bw(f11)));
x(:,137)= (feature_extract(~im2bw(f12)));
x(:,138)= (feature_extract(~im2bw(f13)));
x(:,139)= (feature_extract(~im2bw(f14)));
x(:,140)= (feature_extract(~im2bw(f15)));
x(:,141)= (feature_extract(~im2bw(f16)));
x(:,142)= (feature_extract(~im2bw(f17)));
x(:,143)= (feature_extract(~im2bw(f18)));
x(:,144)= (feature_extract(~im2bw(f19)));
x(:,145)= (feature_extract(~im2bw(f20)));
x(:,146)= (feature_extract(~im2bw(f21)));
x(:,147)= (feature_extract(~im2bw(f22)));
x(:,148)= (feature_extract(~im2bw(f23)));
x(:,149)= (feature_extract(~im2bw(f24)));
x(:,150)= (feature_extract(~im2bw(f25)));
 x(:,151)= (feature_extract(~im2bw(g1)));
x(:,152)= (feature_extract(~im2bw(g2)));
x(:,153)= (feature_extract(~im2bw(g3)));
x(:,154)= (feature_extract(~im2bw(g4)));
x(:,155)= (feature_extract(~im2bw(g5)));
x(:,156)= (feature_extract(~im2bw(g6)));
x(:,157)= (feature_extract(~im2bw(g7)));
x(:,158)= (feature_extract(~im2bw(g8)));
x(:,159)= (feature_extract(~im2bw(g9)));
x(:,160)= (feature_extract(~im2bw(g10)));
x(:,161)= (feature_extract(~im2bw(g11)));
x(:,162)= (feature_extract(~im2bw(g12)));
x(:,163)= (feature_extract(~im2bw(g13)));
```

```
x(:,164)= (feature_extract(~im2bw(g14)));
x(:,165)= (feature_extract(~im2bw(g15)));
x(:,166)= (feature_extract(~im2bw(g16)));
x(:,167)= (feature_extract(~im2bw(g17)));
x(:,168)= (feature_extract(~im2bw(g18)));
x(:,169)= (feature_extract(~im2bw(g19)));
x(:,170)= (feature_extract(~im2bw(g20)));
x(:,171)= (feature_extract(~im2bw(g21)));
x(:,172)= (feature_extract(~im2bw(g22)));
x(:,173)= (feature_extract(~im2bw(g23)));
x(:,174)= (feature_extract(~im2bw(g24)));
x(:,175)= (feature_extract(~im2bw(g25)));
 x(:,176)= (feature_extract(~im2bw(h1)));
x(:,177)= (feature_extract(~im2bw(h2)));
x(:,178)= (feature_extract(~im2bw(h3)));
x(:,179)= (feature_extract(~im2bw(h4)));
x(:,180)= (feature_extract(~im2bw(h5)));
x(:,181)= (feature_extract(~im2bw(h6)));
x(:,182)= (feature_extract(~im2bw(h7)));
x(:,183)= (feature_extract(~im2bw(h8)));
x(:,184)= (feature_extract(~im2bw(h9)));
x(:,185)= (feature_extract(~im2bw(h10)));
x(:,186)= (feature_extract(~im2bw(h11)));
x(:,187)= (feature_extract(~im2bw(h12)));
x(:,188)= (feature_extract(~im2bw(h13)));
x(:,189)= (feature_extract(~im2bw(h14)));
x(:,190)= (feature_extract(~im2bw(h15)));
x(:,191)= (feature_extract(~im2bw(h16)));
x(:,192)= (feature_extract(~im2bw(h17)));
x(:,193)= (feature_extract(~im2bw(h18)));
x(:,194)= (feature_extract(~im2bw(h19)));
x(:,195)= (feature_extract(~im2bw(h20)));
x(:,196)= (feature_extract(~im2bw(h21)));
x(:,197)= (feature_extract(~im2bw(h22)));
x(:,198)= (feature_extract(~im2bw(h23)));
x(:,199)= (feature_extract(~im2bw(h24)));
x(:,200)= (feature_extract(~im2bw(h25)));
x(:,201)= (feature_extract(~im2bw(i1)));
x(:,202)= (feature_extract(~im2bw(i2)));
x(:,203)= (feature_extract(~im2bw(i3)));
x(:,204)= (feature_extract(~im2bw(i4)));
x(:,205)= (feature_extract(~im2bw(i5)));
x(:,206)= (feature_extract(~im2bw(i6)));
x(:,207)= (feature_extract(~im2bw(i7)));
x(:,208)= (feature_extract(~im2bw(i8)));
x(:,209)= (feature_extract(~im2bw(i9)));
x(:,210)= (feature_extract(~im2bw(i10)));
x(:,211)= (feature_extract(~im2bw(i11)));
x(:,212)= (feature_extract(~im2bw(i12)));
x(:,213)= (feature_extract(~im2bw(i13)));
x(:,214)= (feature_extract(~im2bw(i14)));
x(:,215)= (feature_extract(~im2bw(i15)));
x(:,216)= (feature_extract(~im2bw(i16)));
x(:,217)= (feature_extract(~im2bw(i17)));
x(:,218)= (feature_extract(~im2bw(i18)));
x(:,219)= (feature_extract(~im2bw(i19)));
x(:,220)= (feature_extract(~im2bw(i20)));
x(:,221)= (feature_extract(~im2bw(i21)));
x(:,222)= (feature_extract(~im2bw(i22)));
x(:,223)= (feature_extract(~im2bw(i23)));
x(:,224)= (feature_extract(~im2bw(i24)));
x(:,225)= (feature_extract(~im2bw(i25)));
x(:,226)= (feature_extract(~im2bw(j1)));
```

```
 x(:,227)= (feature_extract(~im2bw(j2)));
 x(:,228)= (feature_extract(~im2bw(j3)));
 x(:,229)= (feature_extract(~im2bw(j4)));
 x(:,230)= (feature_extract(~im2bw(j5)));
 x(:,231)= (feature_extract(~im2bw(j6)));
 x(:,232)= (feature_extract(~im2bw(j7)));
 x(:,233)= (feature_extract(~im2bw(j8)));
 x(:,234)= (feature_extract(~im2bw(j9)));
 x(:,235)= (feature_extract(~im2bw(j10)));
 x(:,236)= (feature_extract(~im2bw(j11)));
 x(:,237)= (feature_extract(~im2bw(j12)));
 x(:,238)= (feature_extract(~im2bw(j13)));
 x(:,239)= (feature_extract(~im2bw(j14)));
 x(:,240)= (feature_extract(~im2bw(j15)));
 x(:,241)= (feature_extract(~im2bw(j16)));
 x(:,242)= (feature_extract(~im2bw(j17)));
 x(:,243)= (feature_extract(~im2bw(j18)));
 x(:,244)= (feature_extract(~im2bw(j19)));
 x(:,245)= (feature_extract(~im2bw(j20)));
 x(:,246)= (feature_extract(~im2bw(j21)));
 x(:,247)= (feature_extract(~im2bw(j22)));
 x(:,248)= (feature_extract(~im2bw(j23)));
 x(:,249)= (feature_extract(~im2bw(j24)));
 x(:,250)= (feature_extract(~im2bw(j25)));


%//*************************************************************************************
*************************************************************************************
*************************************************************************************//%

function [featurevector]=line_classifier(image)
row=size(image,1);
column=size(image,2);
[Gmag, Gdir]=imgradient(image);
code0=0;
code1=0;
code2=0;
code3=0;
code3=0;
code4=0;
code5=0;
code6=0;
code7=0;
code8=0;
code9=0;
code10=0;
code11=0;
for r = 1:row
    for c = 1:column
        if Gdir(r,c) >= 0 && Gdir(r,c) < 30
           Code(r,c) = 0;
           code0=code0+1;
        elseif Gdir(r,c) >= 30 && Gdir(r,c) < 60
                Code(r,c) = 1;
                code1=code1+1;
        elseif Gdir(r,c) >= 60 && Gdir(r,c) < 90
                Code(r,c) = 2;
                code2=code2+1;
        elseif Gdir(r,c) >= 90 && Gdir(r,c) < 120
                Code(r,c) = 3;
                code3=code3+1;
        elseif Gdir(r,c) >= 120 && Gdir(r,c) < 150
                Code(r,c) = 4;
```

```
                code4=code4+1;
        elseif Gdir(r,c) >= 150 && Gdir(r,c) <180
                Code(r,c) = 5;
                code5=code5+1;
        elseif Gdir(r,c) >= -180 && Gdir(r,c) < -150
                Code(r,c) = 6;
                code6=code6+1;
        elseif Gdir(r,c) >= -150 && Gdir(r,c) < -120
                Code(r,c) = 7;
                code7=code7+1;
        elseif Gdir(r,c) >= -120 && Gdir(r,c) < -90
                Code(r,c) = 8;
                code8=code8+1;
        elseif Gdir(r,c) >= -90 && Gdir(r,c) < -60
                Code(r,c) = 9;
                code9=code9+1;
        elseif Gdir(r,c) >= -60 && Gdir(r,c) < -30
                Code(r,c) = 10;
                code10=code10+1;
        elseif Gdir(r,c) >= -30 && Gdir(r,c) < 0
                Code(r,c) = 11;
                code11=code11+1;
        end
    end
end
featurevector=[code0;code1;code2;code3;code4;code5;code6;code7;code8;code9;code10;code11]

%//*****************************************************************************

 function [fl re]=lines(im_texto)
% Funcion que divide el texto en lineas
im_texto=clip(im_texto);
num_filas=size(im_texto,1);
for s=1:num_filas
    if sum(im_texto(s,:))==0
        nm=im_texto(1:s-1, :); % Primera línea de la matris
        rm=im_texto(s:end, :);% Remanente de linas en la matris
        fl = clip(nm);
        re=clip(rm);
        break
    else
        fl=im_texto; %Solo una linea
        re=[ ];
    end
end

function img_out=clip(img_in)
[f c]=find(img_in);
img_out=img_in(min(f):max(f),min(c):max(c)); % Imagenes de cosechas

%//*****************************************************************************

disp('testing')
 load ('D:\TP3-Punto3\featureout.mat');
 p=featureout;
 net.inputs{1}.processFcns = {'removeconstantrows','mapminmax'};
 load d:\TP3-Punto3\net.mat;
 load net;
 y5=sim(net,p);
 disp(y5);
 [C I]=max(y5);
 disp(I)
```

```
 disp(C)
 fid = fopen('D:\TP3-Punto3\output.txt','a');
if (I==1)
    fprintf(fid,'1');
fclose(fid);
elseif (I==2)
    fprintf(fid,'2');
fclose(fid);
elseif (I==3)
    fprintf(fid,'3');
fclose(fid);
elseif (I==4)
    fprintf(fid,'4');
fclose(fid);
elseif (I==5)
    fprintf(fid,'5');
fclose(fid);
elseif (I==6)
    fprintf(fid,'6');
fclose(fid);
elseif (I==7)
    fprintf(fid,'7');
fclose(fid);
elseif (I==8)
    fprintf(fid,'8');
fclose(fid);
elseif (I==9)
    fprintf(fid,'9');
fclose(fid);
elseif (I==10)
    fprintf(fid,'0');
fclose(fid);
elseif (I==11)
    disp(' not Found');
clear
end
```

%//**

% El archivo train.m es el que resuelve el problema de reconocimiento de patrones con una red neuronal.

```
% Se Resolvio un problema de reconocimiento de patrones con una red neuronal
rng('default');
load('input108.mat');
load('target650.mat');
inputs = input108';
targets = target650';

% Crear una Red de Reconocimiento de Patrones
hiddenLayerSize = 39;
net = patternnet(hiddenLayerSize);

% Seleccione entrada y salida
net.inputs{1}.processFcns = {'removeconstantrows','mapminmax'};
net.outputs{2}.processFcns = {'removeconstantrows','mapminmax'};

% Establece divicion de datos para el entrenamiento, validación y prueba
net.divideFcn = 'dividerand';  % Divide los datos aleatoriamente
net.divideMode = 'sample';  % Dividir cada muestra
net.divideParam.trainRatio = 80/100;
net.divideParam.testRatio = 20/100;
net.trainFcn = 'trainscg';
```

```
% Elija una función de rendimiento
net.performFcn = 'mse';

% Elija Funcion Plot
net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
  'plotregression', 'plotfit'};

net.efficiency.memoryReduction = 100;
net.trainParam.max_fail = 6;
net.trainParam.min_grad=1e-5;
net.trainParam.show=10;
net.trainParam.lr=0.9;
net.trainParam.epochs=1000;
net.trainParam.goal=0.00;

% Entrenamiento de Red
[net,tr] = train(net,inputs,targets);

% Prueba de Red
outputs = net(inputs);
errors = gsubtract(targets,outputs);
performance = perform(net,targets,outputs)

% Recalcular Entrenamiento, Validación y Prueba
trainTargets = targets .* tr.trainMask{1};
valTargets = targets  .* tr.valMask{1};
testTargets = targets  .* tr.testMask{1};
trainPerformance = perform(net,trainTargets,outputs)
valPerformance = perform(net,valTargets,outputs)
testPerformance = perform(net,testTargets,outputs)

% Ver la Red
view(net)
disp('after training')
y1=sim(net,inputs);
y1=abs(y1);
y1=round(y1);
save   d:\TP3-Punto3\net net;

%Plots
%Uncomment these lines to enable various plots.
%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, plotconfusion(targets,outputs)
%figure, plotroc(targets,outputs)
%figure, ploterrhist(errors)
```

Concluciones:

Un programa para el OCR es un sistema experto en condición de convertir una imagen textual digitalizada de un documento de texto, sea él en un formato digital o manuscrito, reconociendo la disposición en líneas de caracteres alfabéticos y señas diacríticas, para producir un fichero, en qué ellos son traducios en un formato ASCII o Unicode que cualquier ordenador puede editar. Es justo a través de algoritmos de inteligencia artificial que se volvió posible este diálogo entre las dos líneas de búsqueda, implementando programas para los más variados usos o como este ejercicio lo demuestra el de Reconocimiento de Digitos.

Un procedimiento muy eficaz que se empleo es el de las redes neuronales, por la capacidad que estas tienen en probar simultáneamente muchas alternativas de soluciones y también por la poca interferencia que el ruido genera en estas funciones.

Las redes de Hopfield pueden reconstruir perfectamente imágenes a partir de versiones distorsionadas, con ruido o incompletas, siempre que no se hayan producido traslaciones o rotaciones en el patrón.

Cuando las operaciones de reconocimiento de caracteres se completaron, puede intervenir manualmente sobre las informaciones extraídas por el sistema, o para corregir los eventuales errores generados durante el proceso o para la manipulación y la adaptación de del texto extraído.

Además, puesto que una sola búsqueda de semejanzas no es bastante, se potencia el algoritmo sin analizar cada carácter, sino para las palabras enteras como ser una plantilla de digitos del 0 al 9 y poner más inmediata la elección de algunos términos, descartando los más improbables.

Pequeño Manual de Usuario para el Uso de la Aplicación:

Pasos para el correcto funcionamiento de la aplicación OCR:

1. El Archivo interface.m dentro de la carpeta TP3-Punto3 preferentemente ubicada en el disco D://, es el encargado de desplegar la GUI del Ejercicio.
2. Correr la interfaz grafica presionando Run interface (F5).
3. Abierta la interfaz de la aplicación Redes neuronales de Hopfield discreta, para trabajar como OCR proceda a presionar el botón ***Seleccionar Imagen*** luego abrir la carpeta test y seleccionar una de las imágenes de prueba.
4. Cargada exitosamente la imagen se procede entrenar la Red presionando ***Entrenar Red***.
5. Entrenada la red Presione ***Extraer Texto*** para procesar la Imagen y hacer el Reconocimiento del Número correspondiente, se mostrara el valor aproximado de salida y el valor correspondiente de salida en un archivo de texto.