Omar Baruch Morón López
Smart Systems

lunes, 6 de abril de 2020
Robotics student of the University of Guadalajara

Universidad de Guadalajara

Centro universitario de ciencias exactas e ingenierías

Sistemas inteligentes

Impartida por: *M.C* JOSE DE JESUS HERNANDEZ BARRAGAN

Actividad 5

Reconocimientos de números

Realizada por: OMAR BARUCH MORON LOPEZ

Omar Baruch Morón López
Smart Systems

lunes, 6 de abril de 2020

Robotics student of the University of Guadalajara

OBJETIVOS

El objetivo de esta práctica poder identificar números por medio de la cámara web de la computadora y así mismo te diga que numero es con lo complejidad que debe de ser un número no estandarizado y escrito a mano.

Desarrollo:

Primero que nada, se tiene que programar el algoritmo del MLP para n número de entradas m número de neuronas en la capa ocultad y l neuronas en la capa de salida.

Una vez que se tenga el algoritmo se descarga de internet la base de datos MINST que contiene 50,000 datos de entrenamiento de numero del 0 al 9 en una resolución del 28x28 pixeles.

Se pasan los datos por el entrenamiento de la MLP (código anexo [1]) es importante tener en cuenta que las filas corresponden a cada pixel, que a su vez corresponde a una entrada de la red neuronal, es decir las imágenes estarán en formato 784x1 y este será un dato de entrada de enteramiento, al finalizar el entrenamiento con solamente 40,000 datos de los 50,000 se guardan los pesos de la capa oculta y la capa de salida.

Posteriormente se pasa la generalización (Código anexo [2]) donde los 10,00 datos restantes se calculan con la red neuronal y se comparan con el que sabemos es el número real y así se obtiene un porcentaje total de asertividad.

Por último, se crea una aplicación en línea (Código anexo [3]) donde se activa la cámara web y se toman capturas de lo que hay en la cámara web, se prepara la imagen y se compacta a una imagen de 28x28 se pone todo en una columna y se mete a la generalización de la red neuronal y el resultado será el numero que la red neuronal detecta.

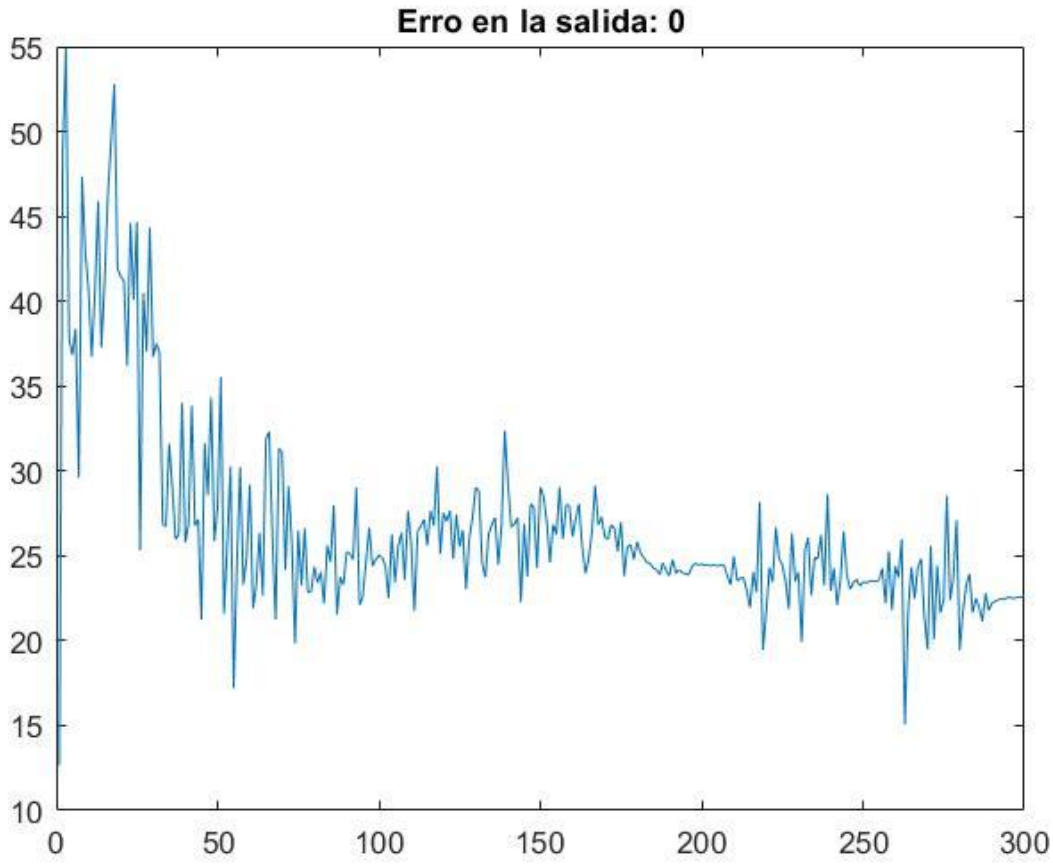
Omar Baruch Morón López
Smart Systems

Resultados

El entrenamiento que mejor rendimiento dio con un asertividad del 92.55% es el que cuenta con las siguientes características:

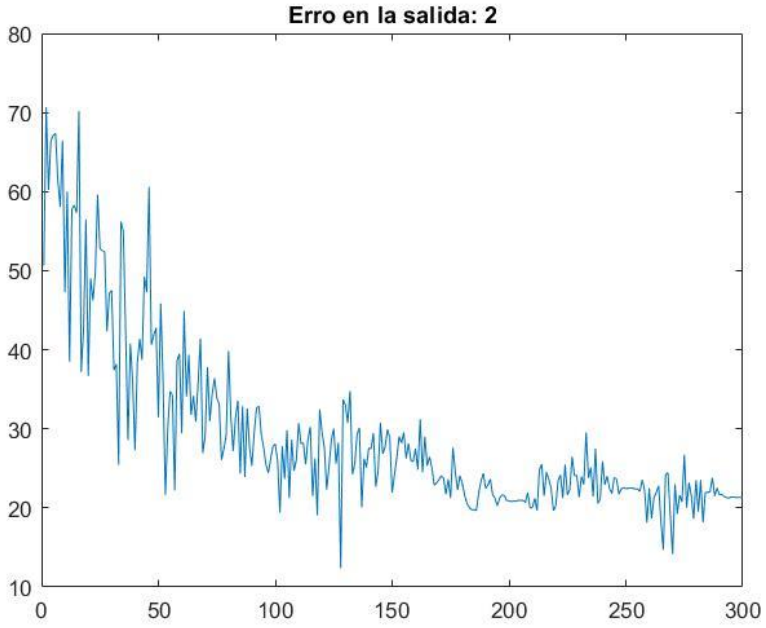
- 50 neuronas en la capa oculta
- 300 interacciones
- Inteligencia del 0.2
- Pesos iniciales entre los valores de 0.001
- Neuronas de la capa oculta con función tnah
- Neuronas de la capa de salida con la función sigmoide

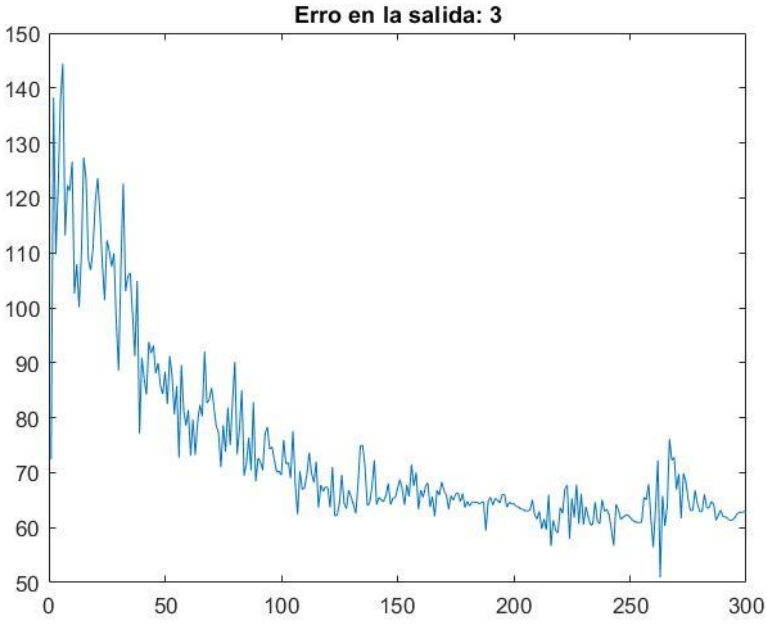
Con las siguientes graficas de error en el entrenamiento para cada número:



Omar Baruch Morón López
Smart Systems

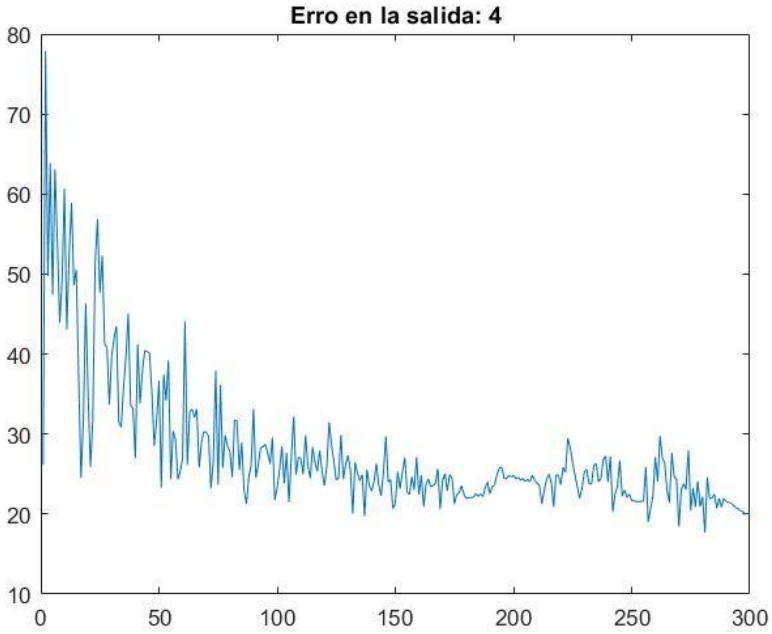
lunes, 6 de abril de 2020
Robotics student of the University of Guadalajara

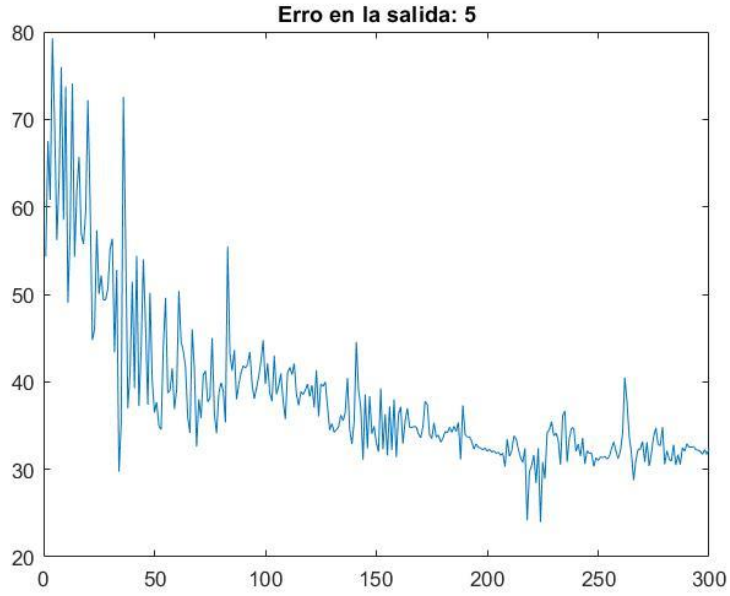




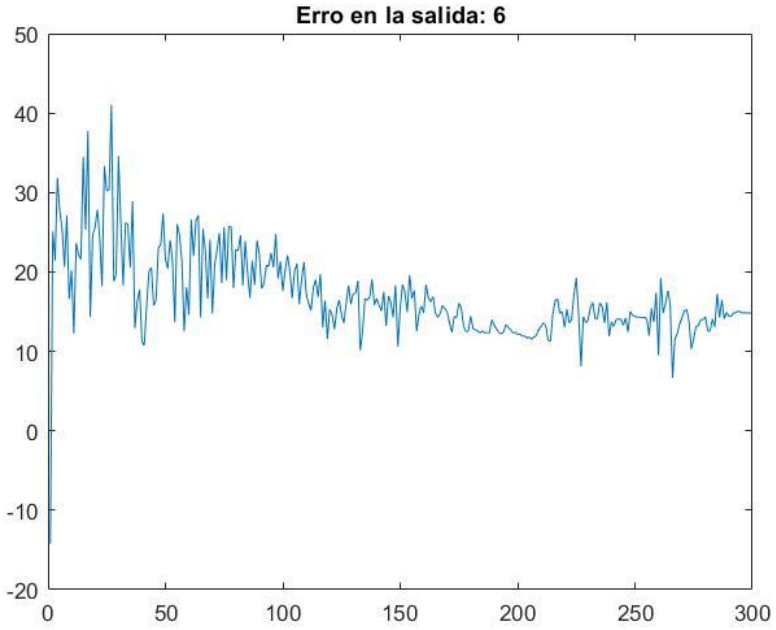
Omar Baruch Morón López
Smart Systems

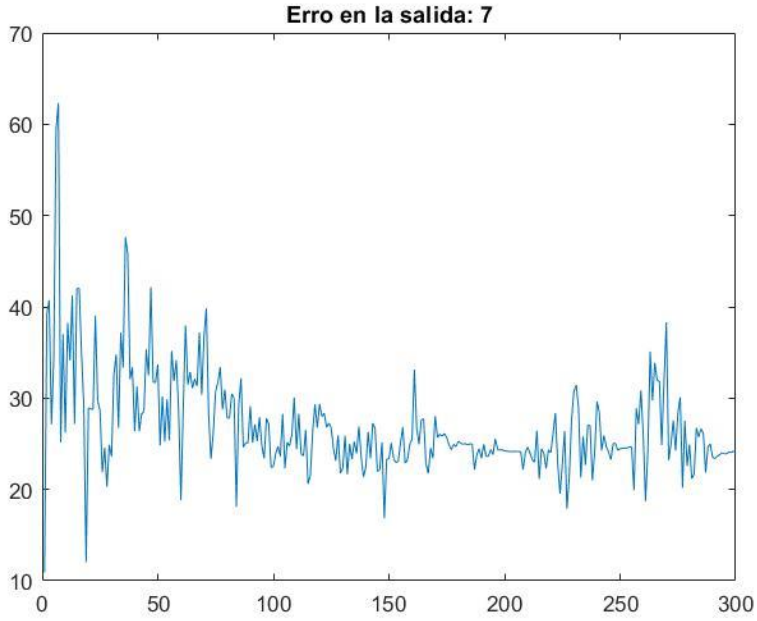
lunes, 6 de abril de 2020
Robotics student of the University of Guadalajara



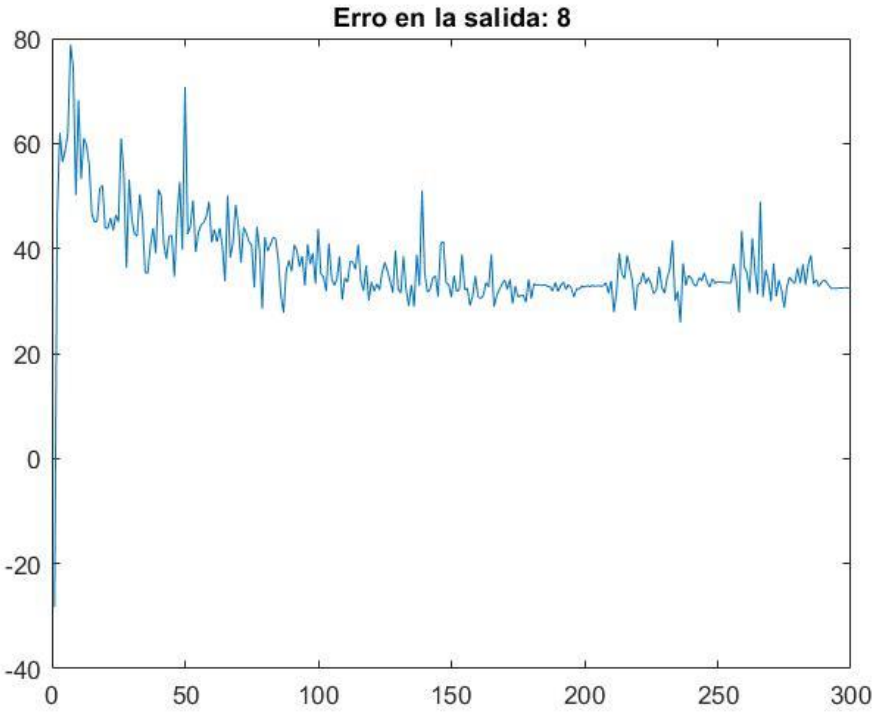


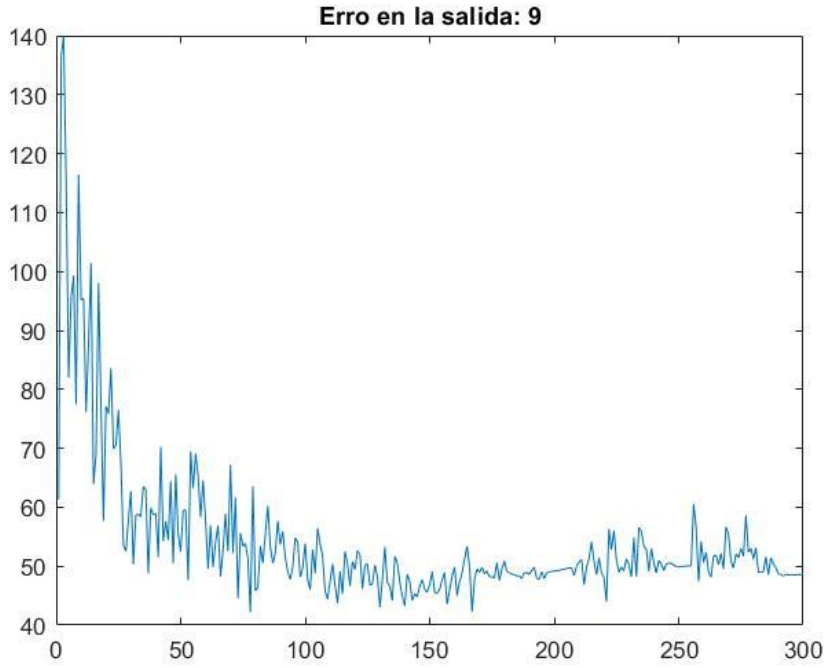
Omar Baruch Morón López
Robotics student of the University of Guadalajara
Smart Systems





Omar Baruch Morón López
Robotics student of the University of Guadalajara
Smart Systems





Omar Baruch Morón López
Smart Systems

lunes, 6 de abril de 2020
Robotics student of the University of Guadalajara

Análisis y conclusión:

Se puede analizar a partir de los diversos entrenamientos y pruebas que realice que es importante la estandarización primero que nada ya que me tope con problemas a la hora de convertir la imagen de nxm a nx1 esto debido a que las funciones incluidas en matlaba no lo organizan como nosotros esperaríamos que se organizaran, sin embargo existe el modo que no esta por default de realizarlo en el acomodo que nosotros queramos, es muy importante la estandarización debido a como ya vimos es posible entrenar una MLP con datos bidimensional y de la dimensión que sea siempre y cuando podemos primero estandarizar y organizar las información de la forma correcta para alimentar a la red neuronal, se puede concluir con que también es importante la selección de los parámetros y esto es una conclusión a la que siempre llego no importa cual sea la red neuronal o que implantación tenga siempre hay que cuidar los parámetros de ajuste de la red que son: eta, valor inicial de los pesos, iteraciones y neuronas de la capa oculta.

Video de YouTube de la implantación del algoritmo:

https://youtu.be/ikjpB5z8S34

Código [Anexo 1] Entrenamiento

```
clearvars -except data;
close all; clc;
%% leer muestras
data = dlmread('entrenamientoTodo.txt');
%% MLP
%% Entrenamiento
sigmoide = @(v) 1./(1+exp(-v)); %Para poder hacer calculos matriciales con ella
xd = data(1:40000,1:end-10)';%%--------------originalmente va tranpuesto
d=data(1:40000,end-9:end)';
[nP,nK]=size(xd); %numero de entreadas y muestras _____cambie
[nS,~]=size(d); %numero de salidas
nO=50;%nP-1%(nP+10)/2; %capa oculta ------------------------EDITABLE-------------------
eta=.2; %aprendisaje
wO=rand(nP+1,nO)*.001; %Inicialisamos pesos aleatorios de capa oculta
wS=rand(nO+1,nS)*.001; %Inicialisamos pesos aleatorios de capa de salida
eVector=[];
```

```
for i=1:300 %iteraciones del MLP
    eA=0;
    i
    for k=1:nK  %para recopilacion de datos de entreniamiento
        %% Porpagacion hacia delante
        xO=[1; xd(:,k)]; % agregamos al dato k el bias
        vO=wO'*xO;       % se calcula la salida de las neuronas de la clapa oculta
        yO= tanh(vO);    % se calucla la funcion de activacion de las salida de la capa oculta
        xS=[1; yO];      % ahora las salida son la entrada de la capa de salida
        vS=wS'*xS;       % se calucla la salida de las neuronas  de la capa de salida
        yS=sigmoide(vS); % se calulca la funcion de activacion de la capa de salida
        e=d(:,k)-yS;     %Se calucla el error al deseado
        eA=eA+e;
        %% Propagacion hacia atras
        deltaS=e.*(yS.*(1-yS));
        deltaO=((ones(nO,1)-yO).*(ones(nO,1)+yO)).*(wS(2:end,:)*deltaS);
        %% Reecalculo de los pesos
        wS=wS+eta*(deltaS*xS')';
        wO=wO+eta*(deltaO*xO')';

    end
    eVector=[eVector eA];
end
% Plotear evoluvion el error por iteracion
for sal=1:nS
figure
plot(eVector(sal,:));
str = sprintf('Erro en la salida: %d', sal-1); %MENOS UNOS SOLO EN ESTE ALGROITMO QUE  QUEREMOS
VER LOS NUMEROS DEL 0 AL 9
title(str);
end
```

Código [Anexo 2] Generalización

```
%data = dlmread('entrenamientoTodo.txt');
sigmoide = @(v) 1./(1+exp(-v)); %Para poder hacer calculos matriciales con ella
xd = data(40001:end,1:end-10)';%%--------------originalmente va tranpuesto
d=data(40001:end,end-9:end)';
correctos=0;
[nP,nK]=size(xd);
for i=1:length(xd)
        x0=[1;xd(:,i)]; %entrada con bias
        v0=wO'*x0; %applicacion
        y0=tanh(v0);
        xS=[1; y0]; % salida
        vS=wS'*xS;
        yS=sigmoide(vS)';
        [~,ic]=max(yS);
        if round(yS(:,:)') == d(:,i)
            correctos=correctos+1;
        end

end
porcentaje=(100*correctos)/nK;
disp('El procentaje de aciertos es:');
disp(porcentaje);
```

Código [Anexo 4] Aplicación en línea con la cámara web

```
 %closepreview(cam);
clearvars -except wS wO data %Borrar todo expeto por los pesos
%% Modificable para Cambiar resolucion de la imagen obtenida de la cam
taFilPix=28; %Pxeles
taColuPix=28; %Pixeles
%% Acceder a la camara
inf=imaqhwinfo; infCam=imaqhwinfo(string(inf.InstalledAdaptors));%? tomar infroamcion de la
camara
cam=videoinput(string(inf.InstalledAdaptors)); %seleccionar camara
preview(cam) %mostrar video
sigmoide = @(v) 1./(1+exp(-v)); %Para poder hacer calculos matriciales con ella
for iter=1:10000
im=getsnapshot(cam); %tomar captura de la camara
im=PrepararImagen(im,taFilPix,taColuPix);
imshow(im)
%x=reshape(im,taFilPix*taColuPix,1); %Se tiene la entrasa de [784,1]
x=reshape(im', taFilPix*taColuPix, []);
%%---------------------------------Generalizacion
pause(.01)
        x0=[1;x]; %entrada con bias
        v0=wO'*x0; %applicacion
        y0=tanh(v0);
        xS=[1; y0]; % salida
        vS=wS'*xS;
        yS=sigmoide(vS)';
        [~,ic]=max(yS);
        if ic==1
                disp('CERO');
            elseif ic==2
                disp('UNO');
            elseif ic==3
                disp('DOS');
            elseif ic==4
                disp('TRES');
            elseif ic==5
                disp('CUATRO');
            elseif ic==6
                disp('CINCO');
            elseif ic==7
                disp('SEIS');
            elseif ic==8
                disp('SIETE');
            elseif ic==9
                disp('OCHO');
            elseif ic==10
                disp('NUEVE');
         end
%%-------------------------------------------------------
end



%% Funciones
function a = PrepararImagen(imagen, tamanioColumnasEnPixeles,tamanioFilasEnPixeles)
% im=rgb2gray(imagen); %convertir a escala de grises
% im=imcomplement(im);
% im=normalize(double(im),'range'); %binarizar
img_gray = rgb2gray(imagen);
[n,m] = size(img_gray);
for i=1:n
    for j=1:m
        if img_gray(i,j)>100
            img_gray(i,j) = 0;
        else
            img_gray(i,j) = 255;
```

```
        end
    end
end
img_gray = imresize(img_gray,[tamanioColumnasEnPixeles tamanioFilasEnPixeles]);
a = (double(img_gray)/255);
end
```