

Licenciatura em Engenharia Informática

# Índice

[Índice 2](#_Toc105202257)

[Alínea A 3](#_Toc105202258)

[Conversão das imagens para matrizes binárias 3](#_Toc105202259)

[Criação e parametrização da rede neuronal 4](#_Toc105202260)

[Protótipo 5](#_Toc105202261)

[Referências 6](#_Toc105202262)

# Alínea A

## Conversão das imagens para matrizes binárias

De modo a converter as várias imagens para matrizes binárias criámos uma função chamada process\_images que recebe como argumento a pasta onde as imagens se encontram (a pesquisa é efetuada dentro das subpastas também) e retorna uma matriz com as várias matrizes binárias das imagens concatenadas (input) e a quantidade de imagens lidas (size).

Para cada imagem encontrada é efetuado o seguinte processo:

* A imagem é lida utilizando a função imread
* É efetuado um redimensionamento à imagem de 224 píxeis para 32 píxeis (quadrados) com recurso à função imresize
* A imagem redimensionada é então convertida para uma matriz binária com o imbinarize
* A matriz é transposta de modo a que os dados fiquem em colunas.
* A matriz binária da imagem é concatenada à matriz input

function [input, size] = process\_images(folder)

directory = dir(folder);

size = length(directory);

for i = 1 : length(directory)

% Read image

img = imread(append(directory(i).folder, "/", directory(i).name));

img = img(:,:,1);

% Rezise image from 224px to 32px

img = imresize(img, [32, 32]);

% Convert image in binary matrix

img = imbinarize(img);

% Transforma a matriz numa coluna

img = img(:);

% Adiciona a coluna aos dados

input(:, i) = img;

end

end

## Criação e parametrização da rede neuronal

O enunciado pedia-nos que criássemos uma rede neuronal do tipo feedforward com 10 neurónios. Para que isso fosse possível utilizámos a função feedforwardnet(10).

Demos também a possibilidade ao utilizador de escolher outras arquiteturas de redes neuronais como as arquiteturas Pattern e Fitting.

Existem vários parâmetros que podem ser configurados numa rede neuronal, como por exemplo: a função de ativação (transferFcn), a função de treino (trainFcn), a função de divisão (divideFcn) e as iterações (epochs).

% Função de Ativação

act\_func = "hardlim";

% Função de Treino

trn\_func = "trainlm";

% Função de Divisão

dvd\_func = "dividerand";

% Épocas (iterações)

epochs = 1000;

net.layers{1}.transferFcn = act\_func;

net.trainFcn = trn\_func;

net.divideFcn = dvd\_func;

net.trainParam.epochs = epochs;

// FALTA COMPARAR OS RESULTADOS OBTIDOSTreino e simulação da rede neuronal

Depois de termos a rede neuronal configurada a gosto podemos passar à fase de treino da mesma. Nesta altura fornecemos a matriz binária que gerámos anteriormente e que representa todas as imagens existentes e fornecemos também uma matriz alvo (target). O objetivo da rede neuronal é aproximar-se o máximo possível do target.

net = train(net, input, target);

Com a nossa rede treinada chega à altura de simular a rede. Fazemos isto 5 vezes para podermos observar o quão precisa é a rede neuronal que treinámos.

A função sim recebe como argumentos a rede treinada e a matriz de input com as imagens convertidas em binário. O retorno desta função é o output da rede neuronal que corresponde à matriz aproximada ao target.

Para gerarmos valores de desempenho utilizamos um ciclo em que sempre que um valor da matriz de output corresponda ao valor da matriz target adicionamos um valor ao r.

nSim = 5;

accuracyFinal = 0;

for i=1 : nSim

out = sim(net, input);

plotconfusion(target, out);

r = 0;

for j=1 : size(out, 2)

[a, b] = max(out(:,i));

[c, d] = max(target(:,i));

if(b == d)

r = r + 1;

end

end

accuracy = r/size(out,2)\*100;

fprintf('Precisão do teste (%d): %.3f\n', i, accuracy);

accuracyFinal = accuracyFinal + accuracy;

end

A matriz de confusão mostra-nos visualmente o desempenho da simulação da rede neuronal.

Uma imagem com mesa

Descrição gerada automaticamente

# Alínea B

## Segmentação do Dataset

Era pedido no enunciado para segmentar o dataset pelas várias fases do modelo (70% para treino, 15% para validação e 15% para teste).

Para que isto fosse possível bastou-nos configurar os vários parâmetros da estrutura divideParam da variável da rede neuronal (net).

net.divideParam.trainRatio = 0.70;

net.divideParam.valRatio = 0.15;

net.divideParam.testRatio = 0.15;

## Precisão total e de teste

Ao treinar a rede com os parâmetros abaixo apresentados atingimos uma média global dos testes de 75.56% de desempenho. Consideramos que este valor é bastante satisfatório mas mesmo assim, nos próximos passos, vamos parametrizar a rede de maneira diferente de modo a obter um desempenho melhor.

net.layers{1}.transferFcn = "logsig";

net.trainFcn = "trainlm";

net.divideFcn = "dividerand";

Média global final dos [Exemplos] depois de 1 testes: 92.667;

Média global final dos [Testes] depois de 1 testes: 75.556;

## Melhor rede neuronal

Asaasda

# Alínea C