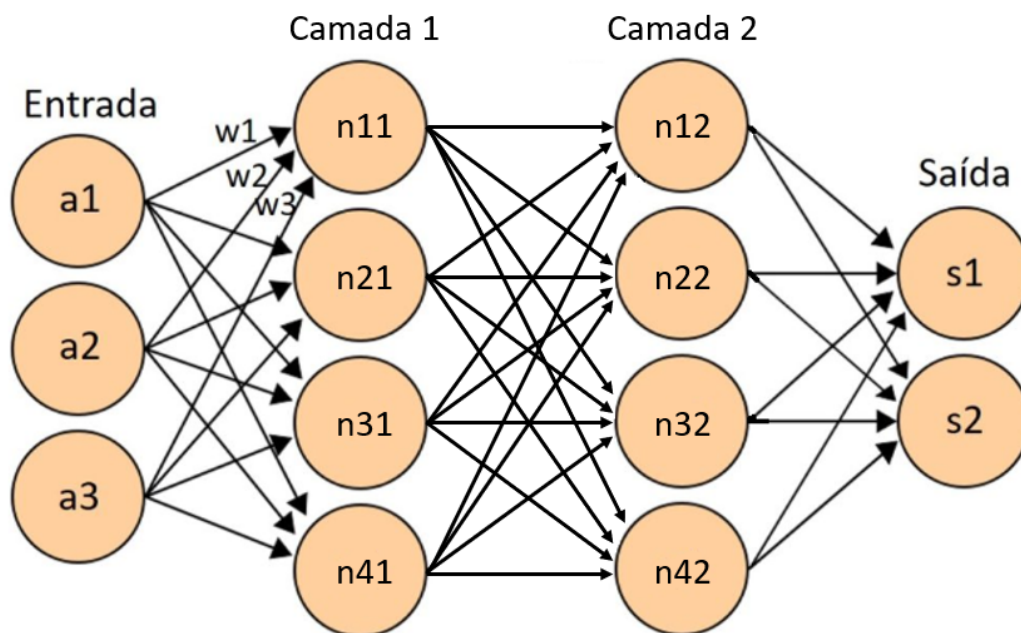


# RELATÓRIO DO TRABALHO PRÁTICO

## Tema 1: Rede Neural



# Índice

Índice.....	1
<b>Alinha a).....</b>	<b>2</b>
preporcess.m.....	2
getTarget.m.....	3
A.m.....	4
<b>Alinha b).....</b>	<b>6</b>
<b>Alinha c).....</b>	<b>8</b>
<b>Alinha d).....</b>	<b>9</b>

## Alinha a)

### preprocess.m

Esta função faz a conversão das imagens que estão nas pastas em matrizes binárias para a rede neural. Ela recebe como parâmetros a diretoria da pasta e o número da resolução que será convertida na matriz binária, depois começa por ver a quantidade e imagens são, vai ler as imagens, mudar a sua resolução para o valor introduzido no input, converte para uma matriz binária e inseres um array que devolve a lista das matrizes binárias produzidas

```
function [input, size] = preprocess(folder, px)
    directory = dir(folder);
    size = length(directory);
    input = [];

    for i = 1 : size
        img = imread(append(directory(i).folder, "/", directory(i).name));
        img = img(:,:,1);

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

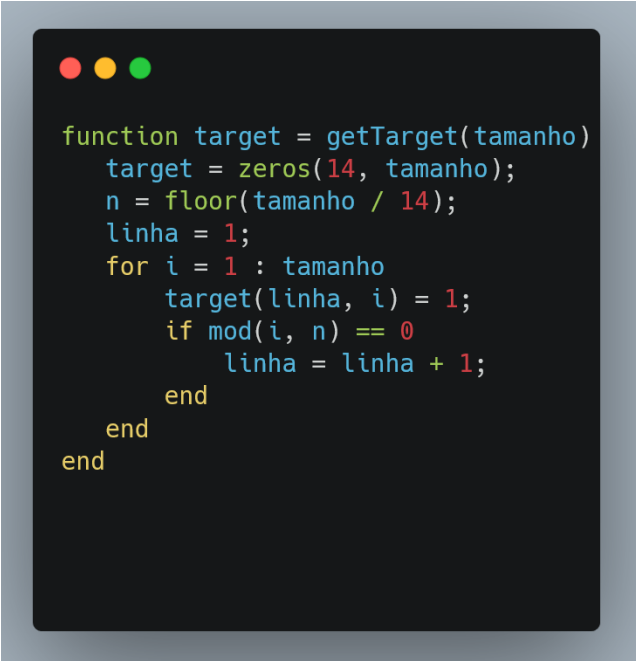
        img = imbinarize(img);

        img = img(:);

        input(:, i) = img;
    end
end
```

## getTarget.m

Esta função faz uma matriz alvo para a rede neural que será treinada. Ela recebe o tamanho das imagens que foram convertidas para matriz binária e devolve a matriz alvo da rede neural.

A screenshot of a MATLAB code editor window. The window has a dark background with three colored window control buttons (red, yellow, green) in the top-left corner. The code is written in a light-colored font. The function is named 'getTarget' and takes 'tamanho' as an input argument. It initializes a 'target' matrix of zeros with dimensions 14 by 'tamanho'. It then calculates 'n' as the floor of 'tamanho' divided by 14. A loop starts at 'linha = 1' and iterates from 'i = 1' to 'tamanho'. Inside the loop, it sets 'target(linha, i) = 1' and checks if 'mod(i, n) == 0'. If true, it increments 'linha' by 1. The loop ends with 'end' and the function ends with 'end'.

```
function target = getTarget(tamanho)
    target = zeros(14, tamanho);
    n = floor(tamanho / 14);
    linha = 1;
    for i = 1 : tamanho
        target(linha, i) = 1;
        if mod(i, n) == 0
            linha = linha + 1;
        end
    end
end
```

## A.m

Neste ficheiro será feito a configuração da rede neural, treina 10 vezes e ver a precisão média das 10 rede neurais treinadas. Começamos por limpar as variáveis na memória e limpar a consola, depois processamos as imagens usando a função preprocesses e fazer a matriz alvo para os 10 treinos. Com o final de cada treino fazemos a simulação de cada rede neural para fazer a avaliação da precisão.

```
clear all
close all
clc

folder = "../data/start/*//*.png";
[input, tamanho] = preprocess(folder, 28);
target = getTarget(tamanho);

epochs = 10;

accuracyFinal = 0;
nSim = 10;

for sim = 1 : nSim
    net = feedforwardnet(10);
    net.trainParam.epochs = epochs;

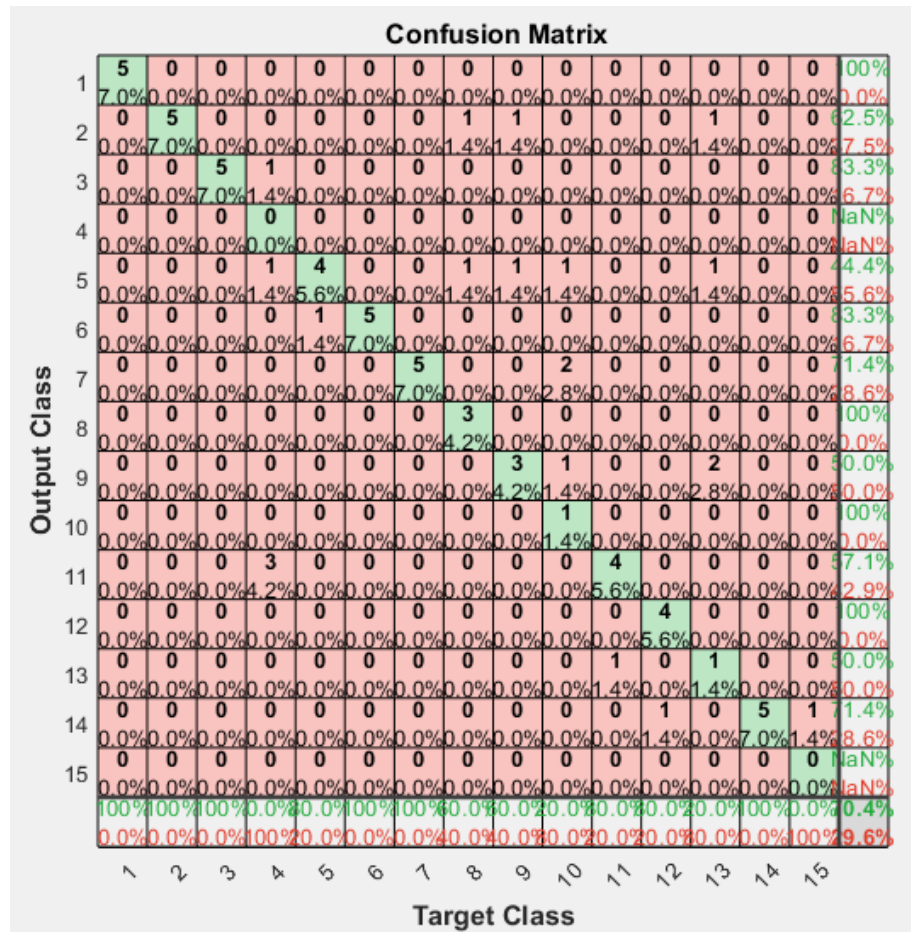
    [net, tr] = train(net, input, target);

    out = net(input);
    plotconfusion(target, out);

    r = 0;
    for i = 1:size(out, 2)
        [~, b] = max(out(:, i));
        [~, d] = max(target(:, i));
        if b == d
            r = r + 1;
        end
    end
    accuracy = r / size(target, 2) * 100;
    fprintf('Precisão na iteração %d: %.3f\n', sim, accuracy)
    accuracyFinal = accuracyFinal + accuracy;
end

fprintf('\nMédia da precisão depois de %i iterações: %.3f\n', nSim, accuracyFinal/nSim);
```

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



## Alinha b)

Nesta alinha será feito várias configurações para várias redes neurais para ver qual seria a melhor rede neural para o nosso objetivo do trabalho. Com isso, optamos por modificar as funções de ativação, a função de treino e a divisão de exemplos.

As funções de ativação escolhidas foram *purelin*, *logsig* e *tansig*.

As funções de treino escolhidas foram *trainlm*, *trainbfg* e *traingd*. Por motivos de escolha, decidimos que a *traingd* não era uma boa opção de configuração, pois demorava muito no treino de uma só rede e os seus valores eram muitos baixos.

A divisão de exemplos seria 70% para treino, 15% para validação e 15% para teste.

Também a opção de separa as redes neurais em duas parte, uma para os números e outra para as operações era pedia, mas os valores finais não faziam muitos sentidos, pois a matriz alvo estava feita para treinos com ambas as partes e não separadas. Mas como só reparamos nesse problema no final e o tempo era pouco, optamos por não fazer essa correção e temos noção que a diferença entres a rede neural para números e operações seria menor que as rede neural para números e a rede neural para operações.

Rede Neural Números + Opeções							
	Número de camadas	Número de neurónios	Funções de ativação	Função de treino	Divisão de exemplos	Média de precisão	Média do teste
Config1	10	10	purelin	trainlm	dividerand = { 70, 15, 15 }	70,914	54,19
Config2	10	10	logsig	trainlm	dividerand = { 70, 15, 15 }	20,486	18,095
Config3	10	10	tansig	trainlm	dividerand = { 70, 15, 15 }	76,157	46
Config4	10	10	purelin	trainbfg	dividerand = { 70, 15, 15 }	6,429	6,667
Config5	10	10	purelin	traingd	dividerand = { 70, 15, 15 }	8,286	5,714
Config6	10	10	logsig	traingd	dividerand = { 70, 15, 15 }	5,571	4,762
Config7	10	10	tansig	traingd	dividerand = { 70, 15, 15 }	5	2,857

Rede Neural Números							
	Número de camadas	Número de neurónios	Funções de ativação	Função de treino	Divisão de exemplos	Média de precisão	Média do teste
Config1	10	10	purelin	trainlm	dividerand = { 70, 15, 15 }	54,32	32,8
Config2	10	10	logsig	trainlm	dividerand = { 70, 15, 15 }	15,56	10,267
Config3	10	10	tansig	trainlm	dividerand = { 70, 15, 15 }	67,94	27,867
Config4	10	10	purelin	trainbfg	dividerand = { 70, 15, 15 }	7,2	8,5
Config5	10	10	purelin	traingd	dividerand = { 70, 15, 15 }	6,8	6,267
Config6	10	10	logsig	traingd	dividerand = { 70, 15, 15 }	6,28	6,933
Config7	10	10	tansig	traingd	dividerand = { 70, 15, 15 }	5,32	5,733

Rede Neural Opeções							
	Número de camadas	Número de neurónios	Funções de ativação	Função de treino	Divisão de exemplos	Média de precisão	Média do teste
Config1	10	10	purelin	trainlm	dividerand = { 70, 15, 15 }	46,25	4,133
Config2	10	10	logsig	trainlm	dividerand = { 70, 15, 15 }	8,7	2,133
Config3	10	10	tansig	trainlm	dividerand = { 70, 15, 15 }	60,6	5,333
Config4	10	10	purelin	trainbfg	dividerand = { 70, 15, 15 }	6,667	2,667
Config5	10	10	purelin	traingd	dividerand = { 70, 15, 15 }	7,5	2,8
Config6	10	10	logsig	traingd	dividerand = { 70, 15, 15 }	6,25	2,533
Config7	10	10	tansig	traingd	dividerand = { 70, 15, 15 }	6,35	2,933

No final, escolhemos a *Config1* para todas as 3 redes neurais, pois tinham os melhores valores de precisão média e de teste médio.



## Alinha c)

Nesta alinha foi pedido fazer as nossas imagens para fazer o teste. Com isso, fizemos 5 imagens para cada número e operação.

Os valores resultantes foram os seguintes:

- Média da precisão [da rede num + op] depois de 10 iterações: **11.940**
- Média da precisão [da rede num] depois de 10 iterações: **10.638**
- Média da precisão [da rede op] depois de 10 iterações: **5.000**

Os valores que tivemos não fizeram sentido, mais para a rede neural números + operações, pois as outras duas foram comprometidas pela nossa falha de configuração na parte de treino das redes neurais. Mas os valores que recebemos da rede neural números + operações não fazem sentido, pois seria aquela que tínhamos mais certeza que teria melhores resultados, mas não foram os valores que recebemos.

## Alinha d)

Por fim, era pedido fazer um interface para desenhar uma conta matemática e para calcular o valor que a rede(s) neural(is) leu(ram).

Com isso a interface por se feita com duas opções: com a rede neural com treino para números e operações ou para a rede neural dedicada para números e a rede neural dedicada para operações.

Mas devido a problemas, não foi concluído a interface por parte de desenhar os números/símbolos das operações. Com isso, a interface fica por fazer neste trabalho

**Redes Neurais**  
Trabalho Prático de CR

Quantas redes

Carregar

Testar