



GARBAGE CLASSIFICATION

Bruno Oliveira nº2019136478

Micael Eid nº2019112744

Índice

- Descrição e objetivos do problema _____pg 3
- Descrição dos Algoritmos_____pg 4 e 5
- Análise de Resultados _____pg 6 a 16
- Conclusões_____pg 17

Descrição e objetivos do problema

- “O desenvolvimento que procura satisfazer as necessidades da geração atual, sem comprometer a capacidade das gerações futuras de satisfazerem as suas próprias necessidades, significa possibilitar que as pessoas, agora e no futuro, atinjam um nível satisfatório de desenvolvimento social, econômico e de realização humana e cultural, fazendo, ao mesmo tempo, um uso razoável dos recursos da terra e preservando as espécies e os habitats naturais.” - Relatório Brundtland
- Este dataset (Garbage Classification) é composto por:
 - Imagens RGB.
 - Tamanho das imagens são aproximadamente 200x200.
 - Classes: 10 (metal, glass, biological, paper, battery, trash, cardboard, shoes, clothes, and plastic).
- Problema de classificação de 5 diferentes tipos de lixo para reciclagem (Papel, Vidro, Cartão, Plástico, Metal).

Descrição dos Algoritmos

Scripts Auxiliares:

- ResizeImg.m
- DeleteImg
- DeleteCMYK.m
- DividirDataSet.m

Divisão do Dataset:

- 70% para treino
- 15% para teste
- 15% para validação

```
Editor - C:\Users\mikae\OneDrive\Públicos\Imagens\Documentos\MATLAB\IC\Project_1\TreinarDataSet.m
TreinarDataSet.m  ValidarDataSet.m  +
1  clear all;
2
3  % path para o dir das imagens
4  FolderPath = 'C:\ISEC\1Sem\IC\TP\DataSet\Garbage_Classification_Resize\TrainSet';
5  dataFiles = dir(fullfile(FolderPath, '*.jpg'));
6
7  % arrays vazios para armazenar as imagens
8  imagens_em_escala_de_cinza = cell(1, numel(dataFiles));
9  labels = zeros(1, numel(dataFiles));
10
11 % Loop atraves das imagens, converter RGB para tons de cinza e retirar as labels pelo nome dos ficheiros
12 for i = 1:numel(dataFiles)
13
14     nome_arquivo = fullfile(FolderPath, dataFiles(i).name);
15     imagem = imread(nome_arquivo);
16
17     % converter a imagem para tons de cinza
18     imagens_em_escala_de_cinza{i} = im2gray(imagem);
19
20     %definição das labels para o treino, através dos nomes dos
21     %arquivos/imagens
22
23     if contains(dataFiles(i).name, "cardboard")
24         labels(i) = 1;
25     elseif contains(dataFiles(i).name, "glass")
26         labels(i) = 2;
27     elseif contains(dataFiles(i).name, "metal")
28         labels(i) = 3;
29     elseif contains(dataFiles(i).name, "paper")
30         labels(i) = 4;
31     elseif contains(dataFiles(i).name, "plastic")
32         labels(i) = 5;
33     end
34 end
35
36 % apanhe o tamanho de uma imagem em grayscale
37 imgSize = size(imagens_em_escala_de_cinza{1});
38
39 % numero total de imagens
40 numImagens = numel(imagens_em_escala_de_cinza);
41
42 % matriz 2D para armazenar as imagens
43 x = zeros(imgSize(1) * imgSize(2), numImagens);
44
45 % matriz x com as imagens
46 for i = 1:numImagens
47     x(:, i) = reshape(imagens_em_escala_de_cinza{i}, [], 1);
48 end
49
50 % labels em vetores one-hot
```

Treino, Validação e Teste

Devido à divisão das imagens do Dataset tivemos que atribuir os labels a cada uma das imagens, convertendoas imagens RGB para imagens com tom cinzento.

Análise de Resultados

- Realizámos vários tipos de testes com diferentes funções, camadas e número de neurónios, de maneira a obter resultados distintos, verificando o melhor.
- Estes testes tiveram recurso a Código dado nas aulas práticas.
- Todas as imagens foram enviadas em anexo no ficheiro .zip

Análise de Resultados

```
% rede neural e suas configurações para o treino

% camadas e neuronios e função de treino
net = patternnet(500);

% funções de ativação das camadas
net.layers{1}.transferFcn = 'tansig';
net.layers{2}.transferFcn = 'tansig';
net.layers{2}.transferFcn = 'tansig';

% função de ativação da camada output
net.layers{end}.transferFcn = 'logsig';

% função de perda
net.performFcn = 'crossentropy';

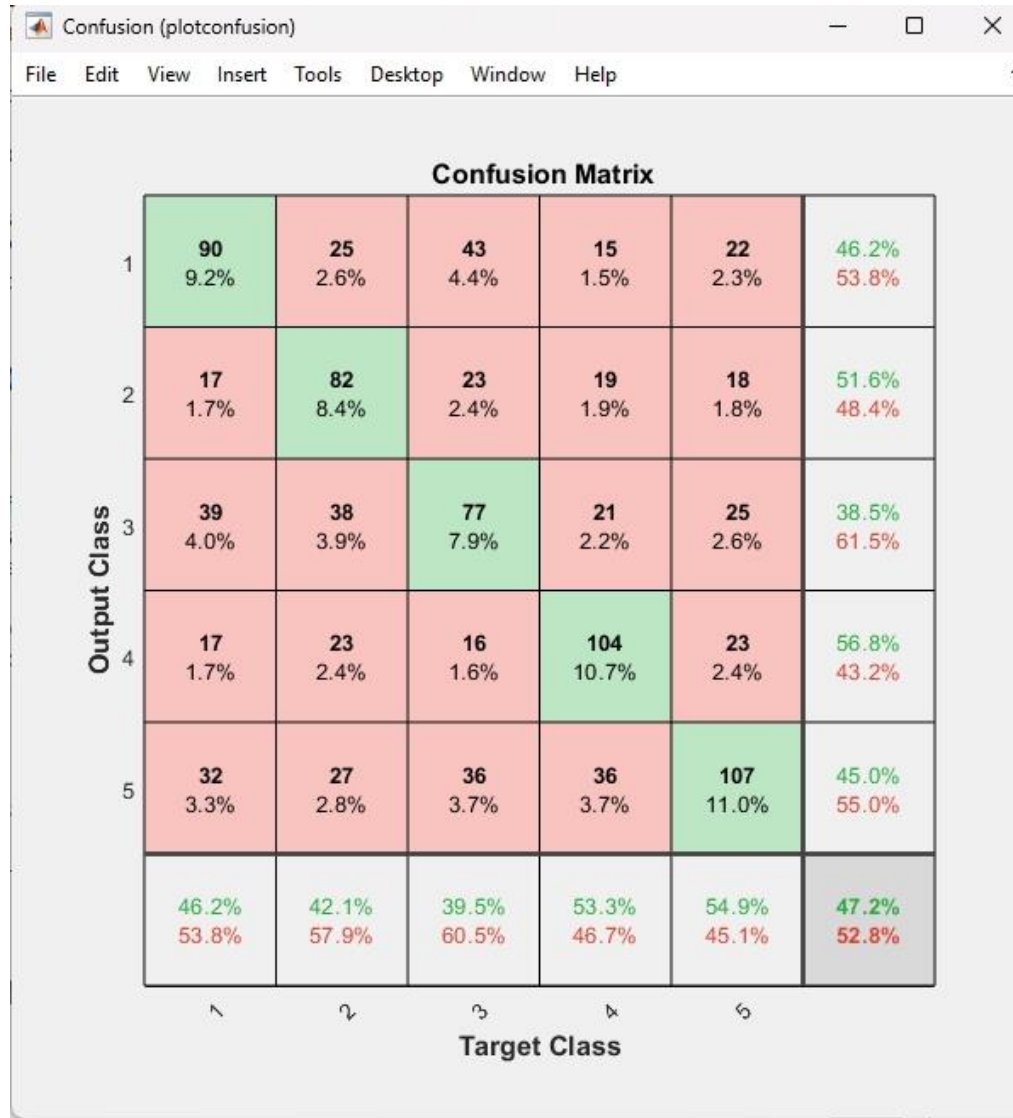
% taxa de aprendizado
net.trainParam.lr = 1;

% numero max de epochs
net.trainParam.epochs = 1000;

% treino
net = train(net, x, t);

% avaliação da rede neural
```

- 1 camada com 500 neurónios, 'trainfcg'
- Accuracy = 47.18%
- Sensibilidade = 46.15%
- Especificidade = 42.02%
- F-measure = 0.46
- AUC = 0.76



Análise de Resultados

Análise de Resultados

```
% rede neural e suas configurações para o treino

% camadas e neuronios e função de treino
net = patternnet([250 250]);

% funções de ativação das camadas
net.layers{1}.transferFcn = 'tansig';
net.layers{2}.transferFcn = 'tansig';
%net.layers{2}.transferFcn = 'tansig';

% função de ativação da camada output
%net.layers{end}.transferFcn = 'logsig';

% função de perda
net.performFcn = 'crossentropy';

% taxa de aprendizado
net.trainParam.lr = 1;

% numero max de epochs
net.trainParam.epochs = 1000;
|
% treino
net = train(net, x, t);
```

- 2 camadas com 250 neurónios, 'trainfcg'
- Accuracy 48.51%
- Sensibilidade = 48.21%
- Especificidade = 49.23%
- F-measure = 0.50
- AUC = 0.78



Análise de Resultados

Análise de Resultados

```
% camadas e neuronios e função de treino
net = patternnet([500 500 500]);

% funções de ativação das camadas
net.layers{1}.transferFcn = 'tansig';
net.layers{2}.transferFcn = 'tansig';
net.layers{3}.transferFcn = 'tansig';
%net.layers{4}.transferFcn = 'tansig';

% função de ativação da camada output
%net.layers{end}.transferFcn = 'logsig';

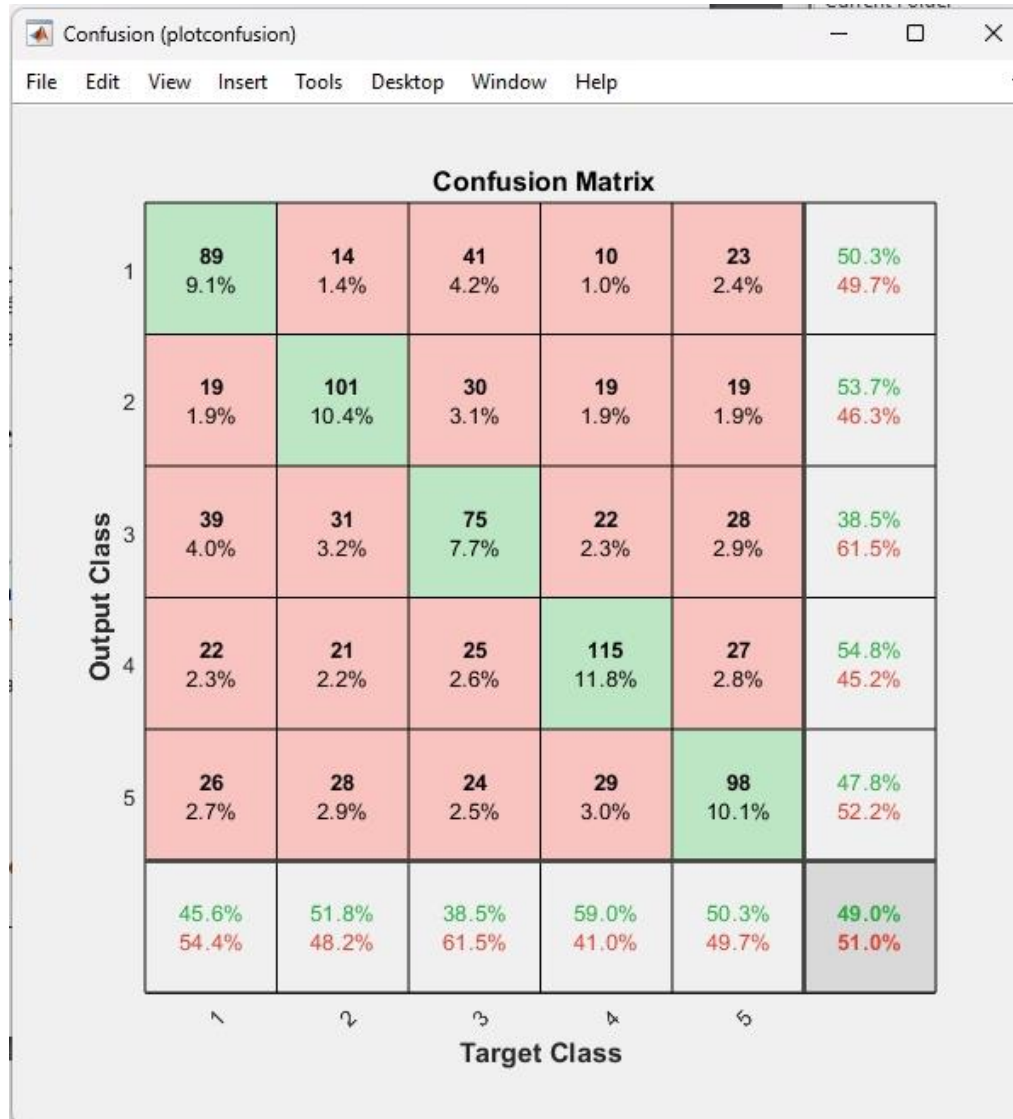
% função de perda
net.performFcn = 'crossentropy';

% taxa de aprendizado
net.trainParam.lr = 1;

% numero max de epochs
net.trainParam.epochs = 1000;

% treino
net = train(net, x, t);
```

- 3 camadas com 500 neurónios, 'trainfcg'
- Accuracy 49.03%
- Sensibilidade = 45.64%
- Especificidade = 51.79%
- F-measure = 0.48
- AUC = 0.77



Análise de Resultados

Análise de Resultados

```
% rede neural e suas configurações para o treino

% camadas e neurônios e função de treino
net = patternnet([200 200 100 50], 'traincgp');

% funções de ativação das camadas
net.layers{1}.transferFcn = 'tansig';
net.layers{2}.transferFcn = 'tansig';
net.layers{3}.transferFcn = 'tansig';
net.layers{4}.transferFcn = 'tansig';

% função de ativação da camada output
net.layers{end}.transferFcn = 'logsig';

% função de perda
net.performFcn = 'crossentropy';

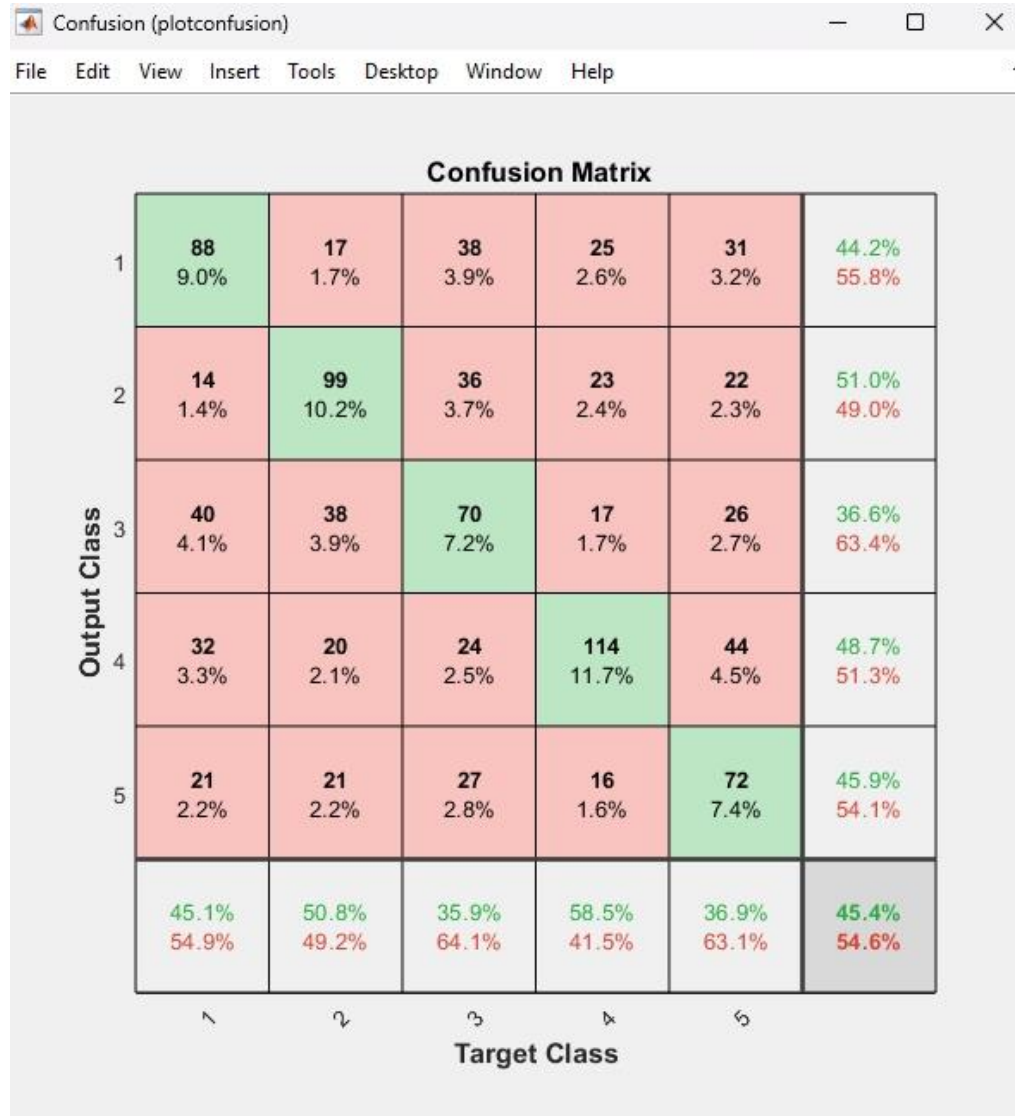
% valor de regularização
net.performParam.regularization = 0.1;

% taxa de aprendizado
net.trainParam.lr = 1;

% número max de epochs
net.trainParam.epochs = 1000;

% treino
net = train(net, x, t);
```

- 4 camadas com 200 200 100 50 neurónios, 'traincgp' e adicionado o valor de regularização
net.performParam.regularization = 0.1 (Overfitting)
- Accuracy = 45.44%
- Sensibilidade = 45.13%
- Especificidade = 50.77%
- F-measure = 0.45
- AUC = 0.77



Análise de Resultados

Este foi o pior teste realizado com accuracy de 45.4%.

Análise de Resultados

```
% rede neural e suas configurações para o treino

% camadas e neurônios e função de treino
net = patternnet([500 500 250 100], 'traincgp');

% funções de ativação das camadas
net.layers{1}.transferFcn = 'tansig';
net.layers{2}.transferFcn = 'tansig';
net.layers{3}.transferFcn = 'tansig';
net.layers{4}.transferFcn = 'tansig';

% função de ativação da camada output
net.layers{end}.transferFcn = 'logsig';

% função de perda
net.performFcn = 'crossentropy';

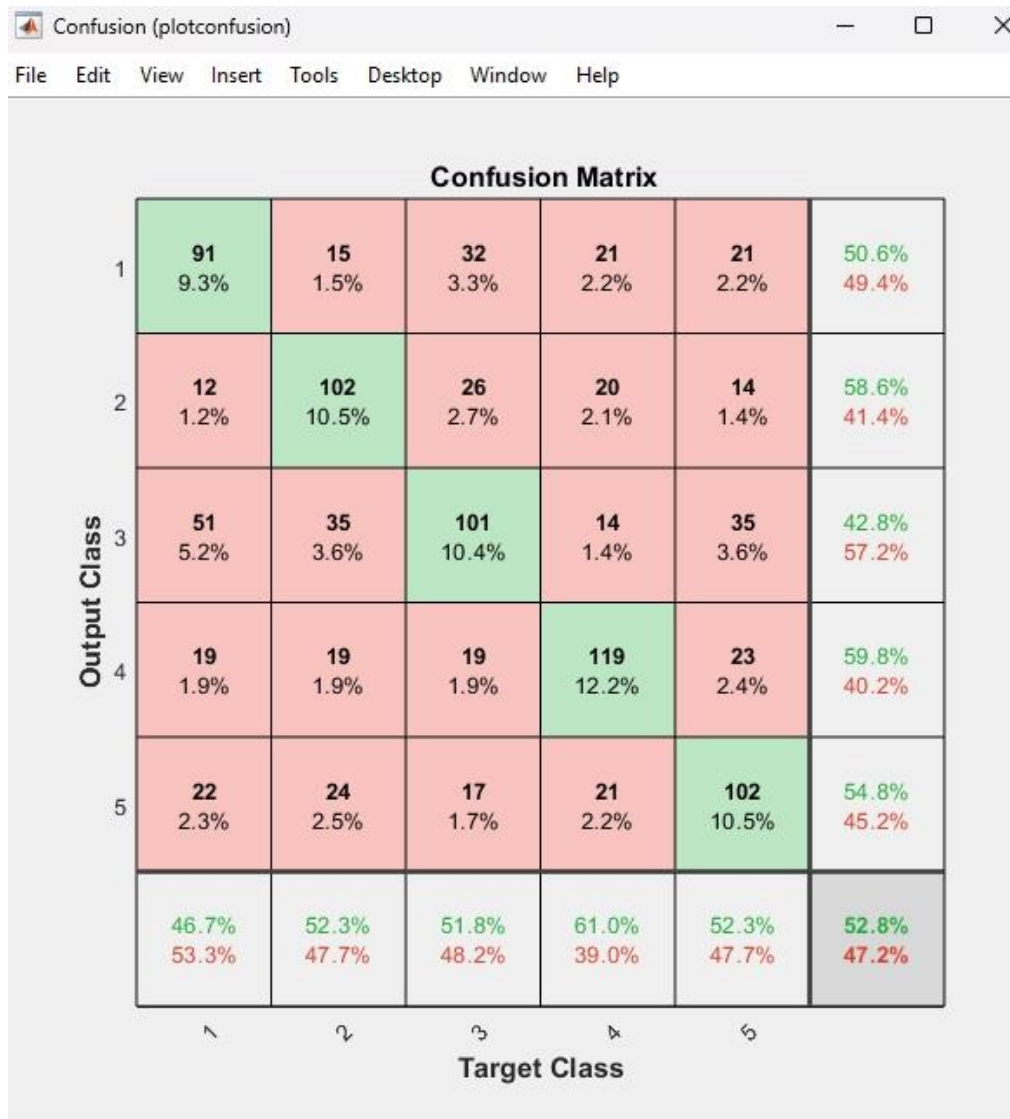
% Valor de regularização
net.performParam.regularization = 0.2;

% taxa de aprendizado
net.trainParam.lr = 1;

% numero max de epochs
net.trainParam.epochs = 1000;

% treino
net = train(net, x, t);
```

- 4 camadas com 500 500 250 100 neurónios, 'traincgp' e adicionado o valor de regularização
net.performParam.regularization = 0.2 (Overfitting)
- Accuracy = 52.82%
- Sensibilidade = 46.67%
- Especificidade = 52.31%
- F-measure = 0.49
- AUC = 0.79



Análise de Resultados

Este foi o melhor teste realizado com accuracy de 52.8%.

Conclusões

- Com a realização desta meta foi-nos possível adquirir novos conhecimentos sobre Matlab e a sua utilização para testes e treino, problemas de classificação com recurso a dataset retirados online e criação de scripts para facilitar o processo de escrita e organização de Código.
- Tivemos algumas dificuldades com o tamanho do dataset e com a utilização das imagens RGB.
- Analisando os resultados obtidos, conseguimos verificar que estão baixos relativamente à Accuracy pretendida (acima de 90%).
- Na meta 2 pretendemos melhorar o código com recurso à linguagem Python, aumentando os parâmetros pretendidos de Accuracy, Sensibilidade, Especificidade, F-Measure e AUC, tanto no teste como em treino.