UNIVERSIDADE D COIMBRA FACULDADE DE CIÊNCIAS E TECNOLOGIA

RELATÓRIO TEORIA DA INFORMAÇÃO

Entropia, redundância e informação mútua

Gabriel Mendes Fernandes Nº 2018288117

Maria da Conceição Silva Dias Nº 2018274188

Pedro Miguel Duque Rodrigues Nº 2018283166

Introdução

Neste primeiro trabalho prático da disciplina de Teoria da Informação pretende-se não só desenvolver a programação em MATLAB, como também aplicar os conceitos teóricos abordados nas aulas. Nomeadamente o conceito de entropia, redundância e ainda informação mútua.

Para o efeito, foi-nos proposta a realização de diversos exercícios que nos permitem colocar estes conceitos em prática e observar como os podemos aplicar a situações um pouco mais reais. Inerente à realização destes exercícios temos também a análise de gráficos, comparação de valores obtidos e ainda o consequente comentário dos mesmos resultados, permitindo-nos assim aprofundar as nossas competências críticas.

Um dos tópicos centrais deste trabalho prático é a entropia, que se pode definir como o limite mínimo teórico para o número médio de bits por símbolo. A entropia pode ser calculada segundo a seguinte fórmula:

$$\text{Entropia} = \sum_{i=1}^{n} pi * \log_2 \frac{1}{pi}$$

Um outro conceito importante é a informação mútua que se pode definir como a quantidade de informação que uma variável contém sobre a outra. A fórmula que permite o cálculo da informação mútua é a seguinte:

$$I(X,Y) = H(X) + H(Y) - H(X,Y)$$

Exercício 1

Para a resolução do exercício número 1, recorremos a uma função chamada histPlot (que recebe como parâmetros o ficheiro, o alfabeto e o número que determina os agrupamentos).

Esta função retorna o histograma de ocorrência de cada símbolo num alfabeto. Se o alfabeto não for fornecido a função irá analisar a fonte e construir um alfabeto com todos os símbolos (únicos) no ficheiro fonte, recorrendo a uma função denominada getAlphabet (que recebe como argumentos a matriz de símbolos da fonte, o tipo do ficheiro e ainda o ficheiro de símbolos da fonte (opcional)). Se o valor dos agrupamentos não for fornecido o valor default é de 1 (criando assim agrupamentos de apenas um símbolo). A função histPlot pode receber fontes de um ficheiro de imagem (.bmp), um ficheiro áudio (.wav) ou um ficheiro de texto (.txt).

Exercício 2

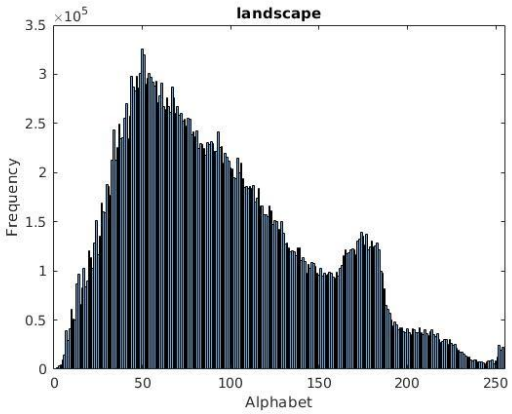
Para a resolução do exercício 2, recorremos a uma função denominada entropy.

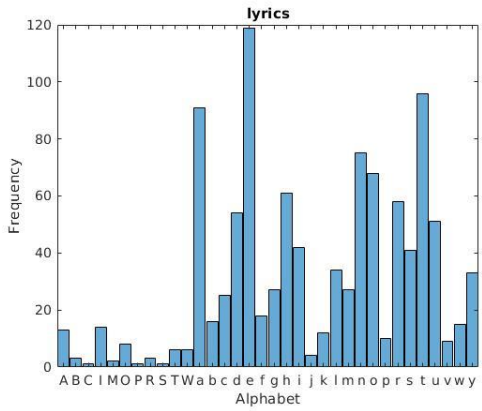
Esta função vai buscar os dados do ficheiro fonte e calcula a entropia da fonte usando para o efeito um alfabeto dado. Se o alfabeto não for fornecido a função construirá um, recorrendo à função getAlphabet anteriormente referida.

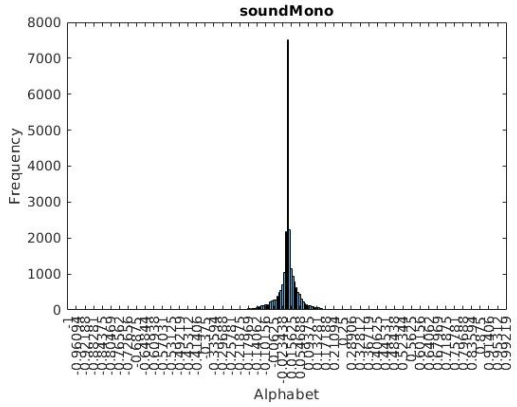
Exercício 3

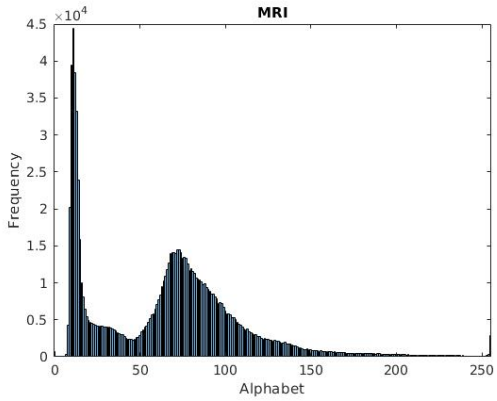
No exercício 3 usámos as funções criadas quer no exercício 1 quer no exercício 2.

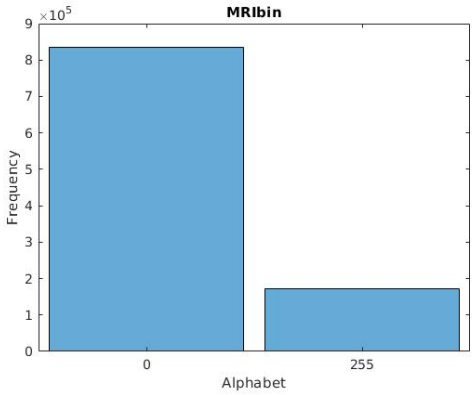
Apresente os resultados.











Analise e comente os resultados.

Previamente sabíamos que a entropia é tanto maior quanto maior a dispersão dos símbolos dessa mesma fonte.

Conseguimos perceber pela análise do histograma da imagem landscape.bmp que os valores em decimal das diferentes cores (numa imagem de 256 tons de cinza) se encontram bastante dispersos ao longo de todo o intervalo de valores possíveis que estas podem tomar. Como tal podemos concluir que a entropia desta imagem terá em observação esta dispersão de valores e será maior em comparação com uma imagem como o MRI.bmp.

No MRI.bmp podemos verificar que a frequência de ocorrência de certas cores é mais preponderante que outras. Por exemplo no intervalo de valores para as cores de valores numéricos entre [0 25] e [50 100] conseguimos verificar que estas ocorrem com muita frequência ao contrário de outras que praticamente não ocorrem (por exemplo no intervalo [150 250]). Com esta concentração e tendo em atenção a fórmula da entropia conseguimos provar que a entropia desta imagem será mais baixa devido a esta concentração de valores.

No MRIBin.bmp as únicas cores que ocorrem em toda a imagem são o preto de valor numérico 0 e o branco de valor numérico 255. Como só existem dois valores possíveis para as cores nesta imagem é espectável que a entropia desta seja a menor

relativamente as duas anteriores visto que não só o alfabeto associado a esta imagem é mais pequeno como a dispersão dos elementos deste é muito baixa.

Analisando o histograma da fonte lyrics.txt e sabendo que na língua inglesa (língua em que o texto presente nesta fonte foi escrito) conseguimos confirmar que as vogais 'a' e 'e' e consoantes como por exemplo o 't' e o 's' ocorrem com muita frequência. Algo que já tínhamos como intuição antes de fazermos o histograma e que ficou confirmado após a visualização deste gráfico. São estes padrões nas línguas que nos permitem identificar linguagens (sendo esta técnica utilizada por muitos linguistas).

No histograma do áudio conseguimos perceber que a maioria dos valores se localiza junto do zero e se distribui de forma simétrica para ambos os lados do histograma. Se repararmos no sinal da onda sonora conseguimos perceber que esta tem sempre uma periocidade e que passa praticamente sempre nas zonas com amplitudes compreendidas entre -0.17 e 0.17. Chegamos assim à conclusão de que os dados do histograma coincidem com a nossa observação da onda sonora.

$$\text{Taxa (x)} = \frac{EntropiaMax - Entropia}{EntropiaMax}$$

Nome Ficheiro	Entropia	Taxa de compressão (x100)	Entropia Máxima
landscape	7.606914	4.91%	8
lyrics	4.410705	26.49%	6
soundMono	4.065729	49.18%	8
MRI	6.860542	14.24%	8
MRIbin	0.661080	33.89%	1

Será possível comprimir cada uma das fontes de forma não destrutiva? Se sim, qual a compressão que se pode alcançar? Justifique.

Através da compressão conseguimos representar a mesma informação veiculada por uma dada fonte de uma forma mais compacta e que nos economiza o espaço que essa mesma informação irá ocupar. Para comprimirmos uma fonte temos de ter em atenção se a compressão da mesma é destrutiva (o que pretendemos evitar), para isso

temos que garantir que os dados que obtemos após a compressão nos permitem reconstruir facilmente os dados que tínhamos inicialmente.

Assim, é possível e a compressão máxima que se pode alcançar coincide com limite mínimo teórico de bits por símbolo.

Exercício 4:

Para a resolução deste exercício usámos a rotina hufflen fornecida com o auxílio de uma função chamada HufflenStatistic.

A função HufflenStatistic recebe os dados do ficheiro fonte e o alfabeto da fonte e calcula a média e a variância do número de bits por símbolo usando códigos Huffman (recorrendo à rotina hufflen).

Analise e comente os resultados.

Os valores obtidos estão um pouco acima dos valores anteriormente obtidos para a entropia. Contudo, verificam (tal como expectável) a seguinte condição:

$$H(s) \leq l \leq H(s) + 1$$

Como os valores obtidos usando códigos de Huffman são superiores aos valores anteriormente calculados para a entropia, podemos concluir que estes códigos não permitem atingir o limite mínimo teórico.

Nome Ficheiro	Entropia	Entropia Huffman	Variância
landscape	7.606914	7.629301	0.751616
lyrics	4.410705	4.443487	1.082055
soundMono	4.065729	4.110714	4.333505
MRI	6.860542	6.890996	2.193081
MRIbin	0.661080	1.000000	0.000000

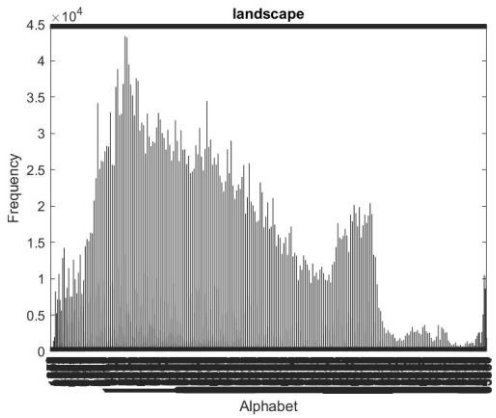
Será possível reduzir-se a variância? Se sim, como pode ser feito e em que circunstância será útil?

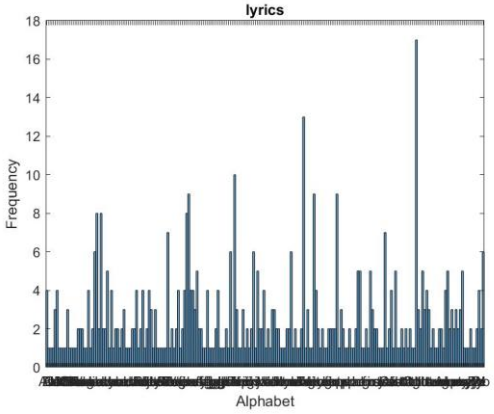
Sim, de facto é possível reduzir a variância. Para isso podemos criar árvores de Huffman com menor profundidade através de um agrupamento de símbolos, colocando o resultado da associação de dois símbolos o mais acima possível na árvore.

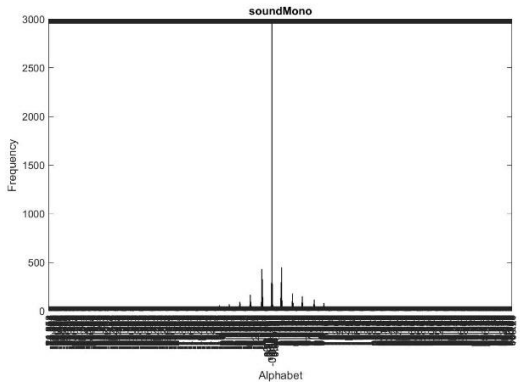
Exercício 5:

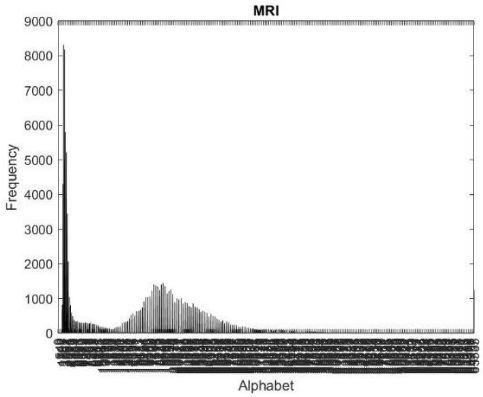
Para este a resolução deste exercício usámos a função histPlot anteriormente criada, passando no último argumento o valor dois (para obtermos agrupamentos de dois a dois).

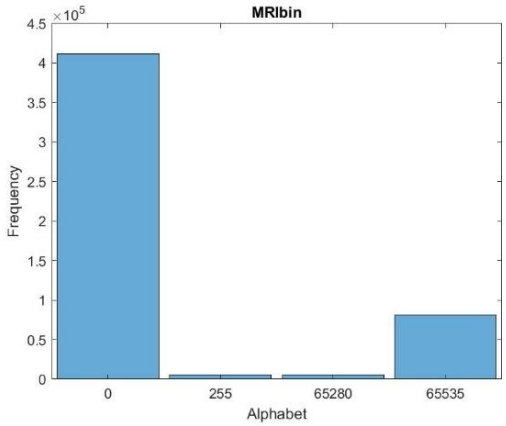
Analise e comente os resultados.











Antes de analisar os dados obtidos é de esperar que estes apresentem uma entropia menor comparativamente com os valores obtidos para o valor da mesma no exercício número 3, uma vez que desta vez foram usados agrupamentos de símbolos.

Nome ficheiro	Entropia	Entropia Conjunta
landscape	7.606914	6.206420
lyrics	4.410705	3.652180
soundMono	4.065729	3.310996
MRI	6.860542	5.226929
MRIbin	0.661080	0.400694

Analisando os resultados obtidos podemos concluir que foram de encontro ao expectável. Ao agruparmos símbolos estamos a reduzir a redundância da fonte através da procura de padrões e de sequências que naturalmente acontecem. Tirando proveito destes padrões conseguimos eficazmente reduzir a entropia da fonte original.

Contudo, é também preciso realçar o facto de o agrupamento de símbolos implicar um alfabeto com um comprimento bastante superior ao do exercício 3. Assim, a execução do programa é mais lenta e para agrupamentos superiores (para tentar diminuir a entropia ainda mais) esta solução tornar-se-á ineficiente devido à complexidade algorítmica. Assim, não podemos utilizar os agrupamentos e esta modelação de contexto como melhor forma de reduzir o número de bits necessários para codificar a fonte visto que ao agrupar símbolos dois a dois e por aí adiante aumenta exponencialmente o tamanho do nosso alfabeto, o que se torna também ineficiente em termos de memória chegando até mesmo a ser impossível armazenar o alfabeto (com o

hardware que temos disponível hoje em dia) quando considerarmos agrupamentos de ordens cada vez maiores.

O valor da entropia obtido agrupando os símbolos do alfabeto foi inferior ao número de bits necessários para codificar cada símbolo não considerando estes agrupamentos. Logo, conseguimos concluir que é possível reduzir o número de bits necessários para codificar um símbolo agrupando os símbolos dois a dois.

Exercício 6:

Para a resolução do exercício 6 e das suas diversas alíneas foram usadas 4 funções: imgMutualInformation, maxMutualInfo, mutualInfo e varMutualInfo.

A função mutualInfo calcula a informação mútua entre a query e o target. Para o efeito, a função calcula a entropia da query e do target e a entropia conjunta entre a fonte e o target. A fórmula para a informação mútua (que é extremamente útil) é a seguinte:

$$I(query; target) = H(query) + H(target) - H(query, target)$$

Esta função retorna a informação mútua obtida pela fonte e pelo target.

A função imgMutualInformation obtém a matriz mutualInfo que corresponde ao cálculo da informação mútua entre a nossa query e as múltiplas partes da janela de target que são fatiadas de acordo com o passo dado.

Esta função retorna a mutualMatrix, que é uma matriz contendo a informação mútua de todas as partes que obtemos do target.

A função maxMutalInfo obtém a matriz mutualInfo e retorna o valor máximo para essa matriz e as coordenadas do canto superior direito da janela onde obteve essa informação. As coordenadas representam a localização do retângulo.

A função varMutualInfo obtém a matriz mutualInfo e retorna a variância entre os valores dessa mesma matriz.

Usando o ficheiro "query" como query, determine a variação da informação mútua entre este e os ficheiros "target_original.bmp", "target_noise.bmp", "target_inverted.bmp" e "target_lightning_contrast.bmp". Defina um passo com valor de 15 pixeis.

Calcule o valor máximo de informação mútua da query com cada imagem de target, e as coordenadas 2d da janela em que esse valor é máximo.

Apresente estes valores no relatório.

Target_inverted

Informação mútua máxima = 1.350032

variância = 0.035988

(x, y) = (421, 316)

Target_lightning_contrast

Informação mútua máxima = 1.224019

variância = 0.022746

(x, y) = (421, 316)

Target_noise

Informação mútua máxima = 1.196346

variância = 0.065021

(x, y) = (421, 316)

Target_original

Informação mútua máxima = 1.350032

variância = 0.035988

(x, y) = (421, 316)

Usando essas coordenadas, desenhe um retângulo sobre cada imagem target na localização da janela com maior valor de informação mútua.

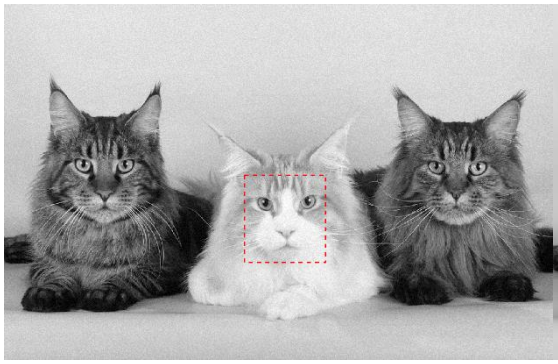


FIGURA 1 TARGET_NOISE

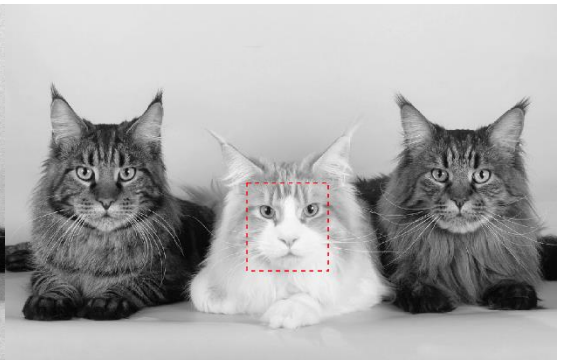


FIGURA 2 TARGET_ORIGINAL

FIGURA 3 TARGET_INVERTEDFIGURA

FIGURA 4 TARGET_LIGHTNING_CONTRAST

Analise e comente os resultados. Foi possível localizar a imagem nas várias distorções do target? Qual o efeito na informação mútua obtida?

Sim, neste caso o maior valor de infomação mútua é comum a duas imagens, sendo elas o target_original e o target_inverted.

A distorção de uma imagem não altera a relação que esta tem com a query uma vez que o nosso objetivo é encontrar a melhor relação entre a query e o target.

Pretende-se agora simular um pequeno simulador de identificação de faces de gatos. Usando o ficheiro "query.bmp" como query e os ficheiros "target*.bmp" como target (mantenha o passo de 15 pixeis da alínea anterior):

calcule a informação mútua máxima para cada target

verifique se corresponde à face do gato na imagem

e, finalmente, apresente os resultados da pesquisa, seriados por ordem decrescente de informação mútua. O sistema consegue encontrar o animal correto?

Apresente os resultados (informação mútua em cada caso).

Target1

Informação mútua máxima = 1.700786

(x, y) = (121, 196)

Target2

Informação mútua máxima = 1.200681

(x, y) = (76, 151)

Target3

Informação mútua máxima = 1.170559

(x, y) = (166, 91)

Target4

Informação mútua máxima = 1.112329

(x, y) = (256, 196)

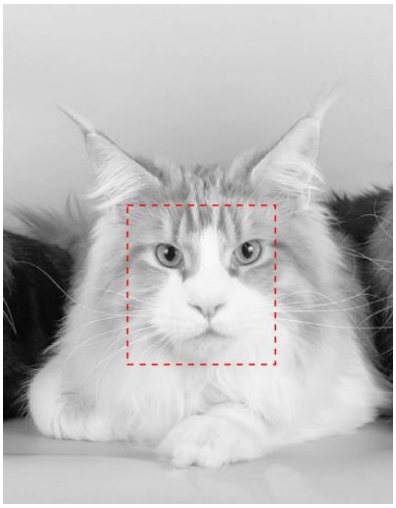


FIGURA 5 TARGET 1

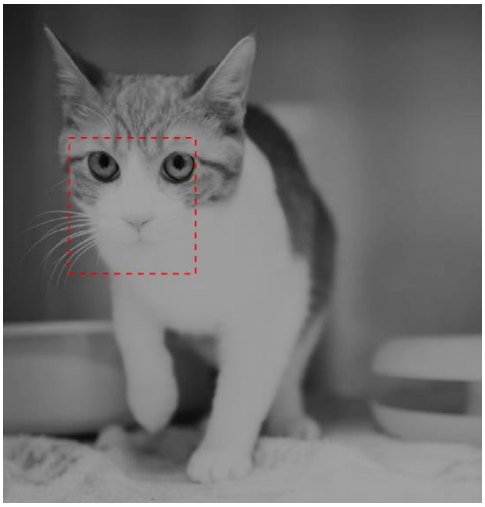


FIGURA 6 TARGET 2

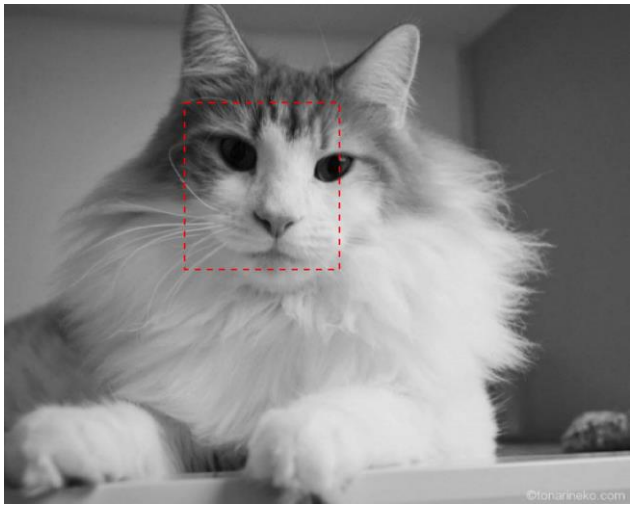


FIGURA 7 TARGET 3

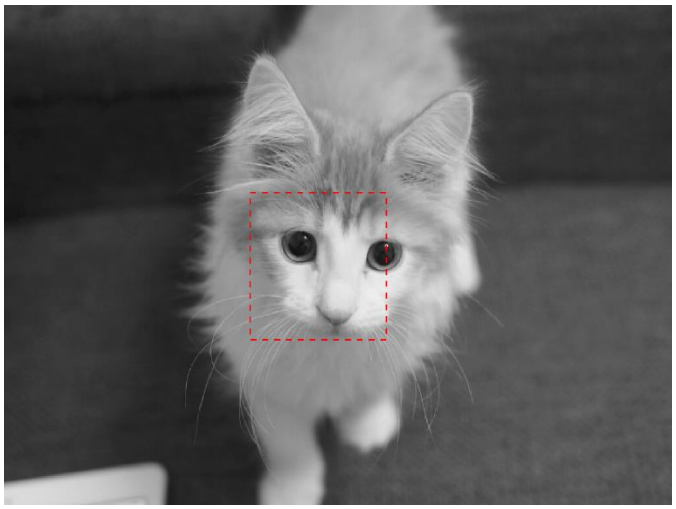


FIGURA 8 TARGET 4

Analise e comente os resultados.

Neste caso podemos concluir que a imagem mais parecida é o target 1, porque segundo os valores obtidos (entre todas as imagens) é aquela que apresenta um valor de informação mútua maior. Verificando, podemos concluir que foi identificada a imagem correta.

Conclusão

Concluída a realização deste trabalho podemos dizer que de facto nos foi possível aprofundar e aplicar os conceitos propostos pelo docente com o auxílio dos diversos exercícios e com a análise dos resultados obtidos.

Chegámos ainda a diversas conclusões, a realçar o facto de os agrupamentos de símbolos nos permitem reduzir a entropia da fonte original. Contudo estes agrupamentos tornam-se ineficientes quando fazemos o agrupamento de diversos símbolos, uma vez que aumentam imenso a complexidade algorítmica (devido ao tamanho do alfabeto que aumenta exponencialmente com o aumento dos agrupamentos).

Contudo a realização deste trabalho não foi perfeita na medida em que fomos encontrando algumas dificuldades ao longo da mesma. A realçar o facto de termos tido alguma dificuldade em encontrar algoritmos que nos permitissem diminuir os tempos de execução do programa. Na questão número cinco também tentámos fazer plot dos gráficos em 3D, contudo como não conseguimos fazer o plot de todos optámos por os apresentar todos em 2D.