Relatório do Trabalho Prático 1

André Santos
João Botelho
Tiago Martins

Faculdade de Ciências e Tecnologia
da Universidade de Coimbra

30 de Outubro de 2017

Conteúdo

1 Introdução

No âmbito da disciplina de Teoria da Informação, da Licenciatura em Engenharia Informática da Universidade de Coimbra foi proposto a execução de trabalho prático para aprofundar os conhecimentos lecionados nas aulas Teóricas.

A integralidade do projeto foi na no ambiente de Matlab, que proporcionou a aprendizagem de uma nova linguagem. No trabalho foram implementados 3 conceitos, a entropia, códigos de huffman e a informação mútua.

2 Histograma da Ocurrência de Símbolos

Usando as duas funções, histc e bar, é implementada a funcionalidade de construir o histograma com a frequência dos símbolos .

```
function symbols_frequency(source, alphabet)

  bar(histc(source, alphabet));
  xtickangle(90), xlabel('alphabet'), ylabel('frequency');

end
```

3 Calculo da Entropia

Com a função reshape transforma-se a fonte de informação para um vetor. Depois a função tabulate retorna as probabilidades. Assim já é possível calcular a entropia.

$$H(X) = \sum P(X) * \log_2(P(X)) \tag{1}$$

```
function entropy = calc_entropy(source, alphabet)

  new_source = reshape(source, 1, []);
  tbl = tabulate(new_source);
  probability = nonzeros(tbl(:,3)/100);
  entropy = -sum(probability .* log2(probability));

end
```

4 Distribuição estatística e Entropia

Resultados obtidos para a entropia:

```
kid.bmp = 6.954143 bits/simbolo
homer.bmp = 3.465865 bits/simbolo
homerbin.bmp = 0.644781 bits/simbolo
guitarSolo.wav = 7.358020 bits/simbolo
english.txt = 4.228071 bits/simbolo
```

Os resultados obtidos adequam-se ás imagens analisadas, sendo mais evidente na homerBin porque esta é binária.

É possível comprimir as fontes de informação de forma não destrutiva em que o limite mínimo de bits por símbolo é o valor da entropia.

5 Código de Huffman

A frequência é calculada usando a função tabulate, que recebe um vetor, então é utilizada a função reshape. Tendo a frequência, já é possível calcular a variância e o tamanho médio de bits por símbolo.

```
function [bmean, v] = bmean_var(source)

  new_source = [];

  new_source = reshape(source', 1, [])';

  tbl = tabulate(new_source);

  probabilities = tbl(:,3) / 100;
  freq = tbl(:,2);

  hl = hufflen(freq);
  bmean = sum(probabilities .* hl);
  v = var(hl, probabilities);

end
```

6 Entropia para simbolos agrupados

Parâmetros de entrada:
source - fonte de informação
n_column = número de colunas que é necessário agrupar
n_symbols = número de símbolos por linha depois de ser agrupado
Primeiro faz-se reshape da fonte para ter o número de colunas correto. Se a fonte tem 2 elementos por linha então é necessário passar para 1, assim aplica-se:

$$(x, y) = max\_value * x + y \tag{2}$$

Cada par (x,y) vai ter um valor único. De seguida usa-se a função tabulate para o cálculo das probabilidades. Agora já foi calculada toda a informação para a entropia.

```
function entropy = entropy_grouped(source, n_column, n_symbols)

  new_source = [];
  max_value = max(max(source)) + 1;

  new_source = reshape(source', n_column, [])';

  if n_column == 1
    tbl = tabulate(new_source);
  elseif n_column == 2
    mul_matrix = [max_value + 1; 1];
    tbl = tabulate(new_source * mul_matrix);
  end

  probs = nonzeros(tbl(:,3) / 100);

  entropy = -sum(probs .* log2(probs))/n_symbols;
end
```

7 Informação Mútua

Dado que:

$$I(X,Y) = H(X) + H(Y) - H(X,Y) \tag{3}$$

Recorrendo á função já implementada para o cálculo da entropia, podemos obter facilmente a informação mútua.

```
function mut_inf = mutual_inf(query, target, step)

  mut_inf = [];

  h_query = entropy_grouped(query, 1, 1);
  limit = ceil((size(target,2) - size(query, 2) + 1) / step);

  for i = 0:limit-1

    if i == 0
      first_pos = 1;
      last_pos = size(query, 2);
    else
      first_pos = i * step;
      last_pos = i * step + size(query, 2) - 1;
    end

    window = target(first_pos:last_pos);

    h_target = entropy_grouped(window, 1, 1);

    h_grouped = entropy_grouped([query', window'], 2, 1);
    mut_inf(i + 1) = h_query + h_target - h_grouped;
  end
end
```

8 Conclusão

A realização deste projeto, propôs um desafio para cada elemento do grupo de trabalho, sendo assim enriquecedor para a aprendizagem destes conceitos chave para a compressão de dados.

Terminando este projeto, alcançou-se o objetivo final. No decorrer do mesmo surgiram algumas dificuldades, sendo a maior calcular a entropia agrupada dos símbolos. Mas com o auxílio dos professores foi possível ultrapassa-las.

Sugerimos assim, a continuidade da realização destes projetos, como forma a desenvolver espírito criativo, indispensável á prática da nossa profissão.