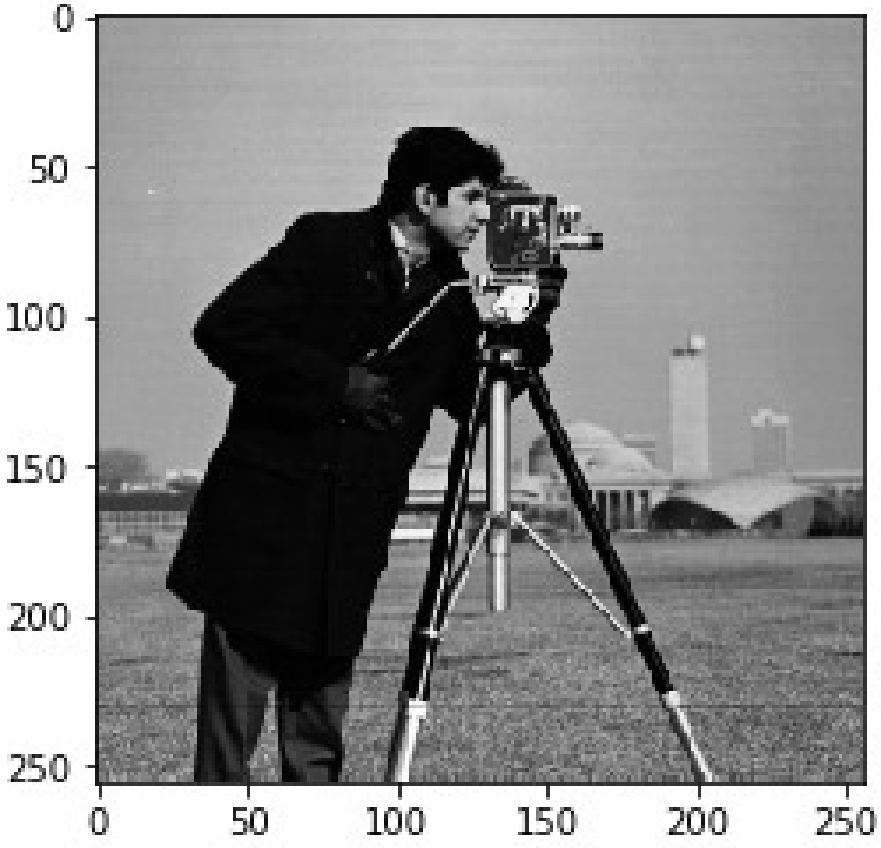
Código para ler e mostrar uma imagem

- import matplotlib.pyplot as plt
- image = plt.imread('cameraman.jpg')
- plt.imshow(image, cmap='gray')

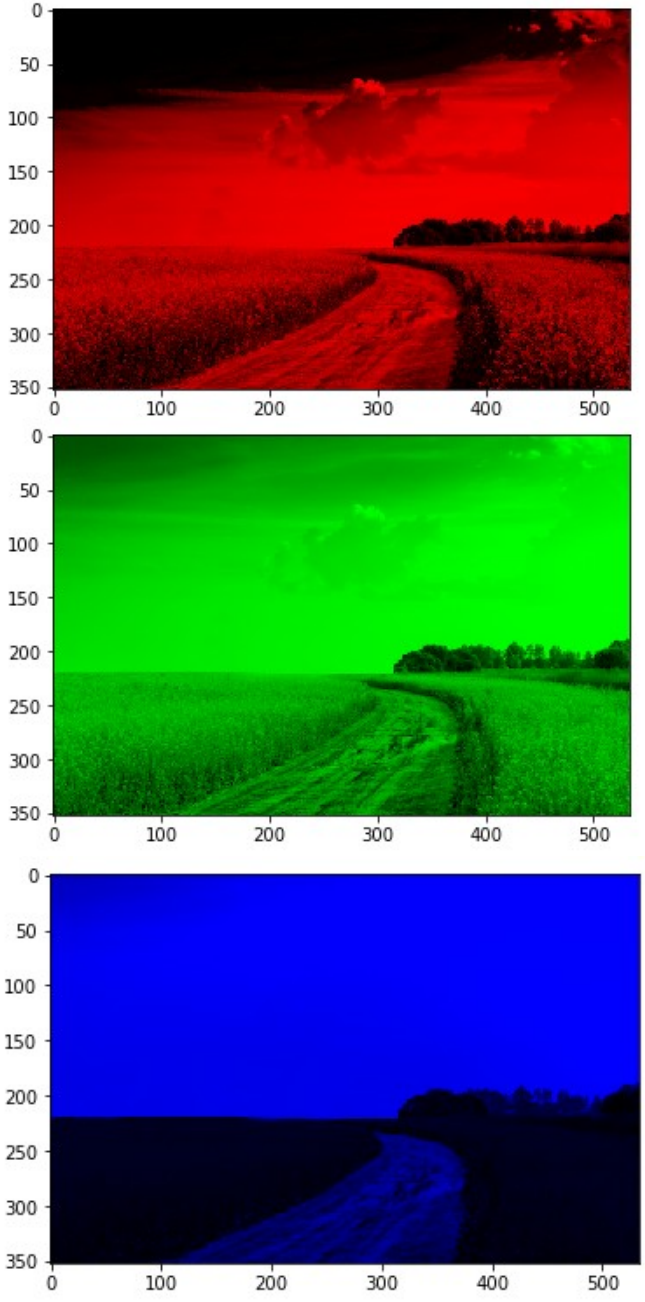


Outras funções

- image.shape → retorna o tamanho da matriz
- Se digitar o nome da imagem ‘image’ no console, mostra-se o conteúdo da variável
- A função ‘imsave’ grava a variável imagem em arquivo.

Formato RGB

- f[:,:,0] → Camada R (red)
  - r = f.copy()
  - r[:,:,1] = 0
  - r[:,:,2] = 0
  - plt.imshow(r)
- f[:,:,1] → Camada G (green)
-
-
- f[:,:,2] → Camada B (blue)



Apenas com estes conceitos, o que é possível se fazer?

Tarefa 1

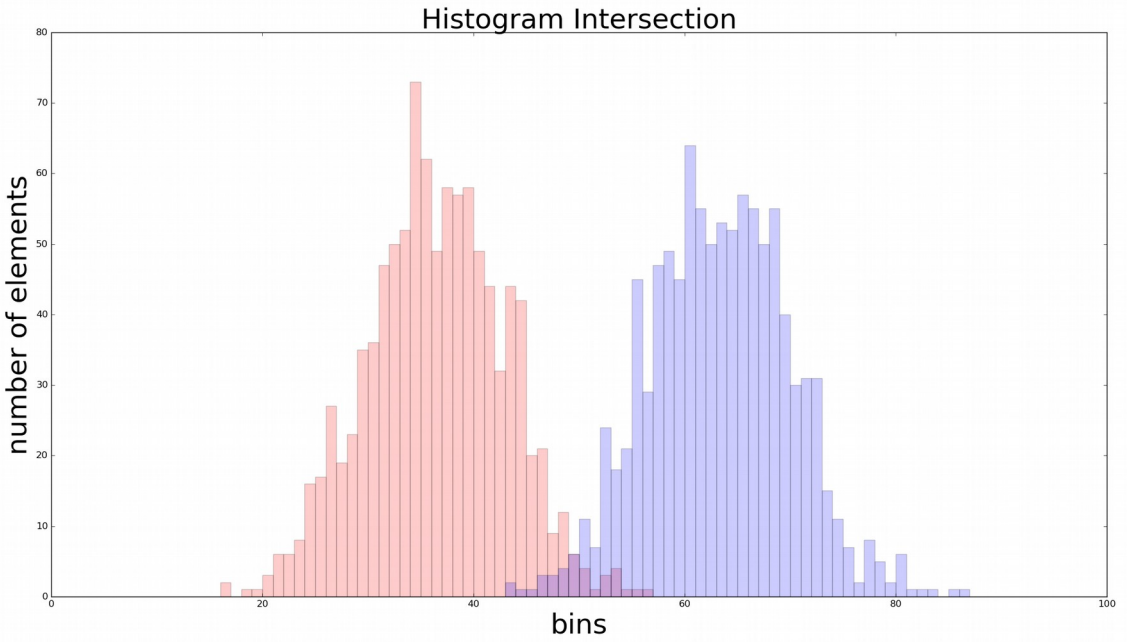
É possível comparar imagens

Como?

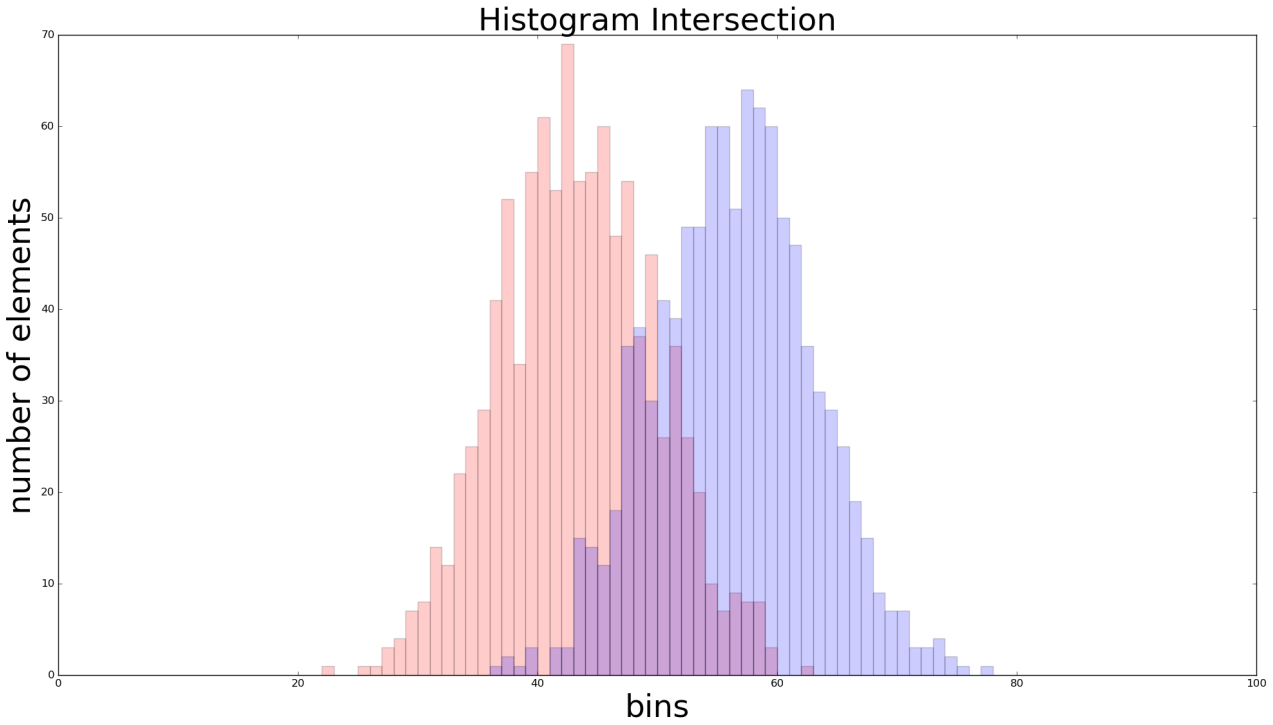
- Comparando histogramas!
- Digamos que temos duas imagens em escala de cinza, uma com histograma *I* e outra com histograma *M*, cada uma com *n* bins

$$\sum_{j=1}^{n} min(I_j, M_j)$$

$$\frac{\sum_{j=1}^{n} min(I_j, M_j)}{\sum_{j=1}^{n} M_j}$$

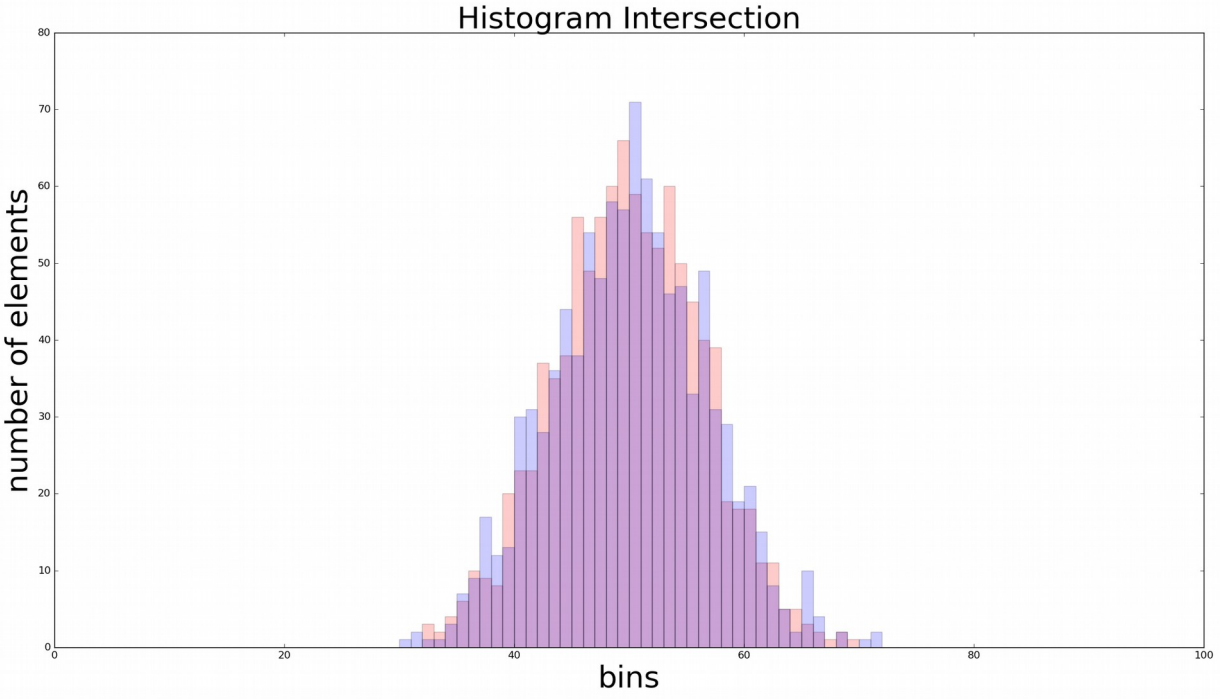


0.035


Histogram Intersection
number of elements
bins
0
10
20
30
40
50
60
70
0
20
40
60
80
100

0.329

0.89


Histogram Intersection
number of elements
bins
0
10
20
30
40
50
60
70
80
0
20
40
60
80
100

Código que faz a comparação

```
def return_intersection(hist_1, hist_2):
    minima = np.minimum(hist_1, hist_2)
    intersection = np.true_divide(np.sum(minima), np.sum(hist_2))
    return intersection
```

Comparando

- r = return_intersection(hr, hr2)
- g = return_intersection(hg, hg2)
- b = return_intersection(hb, hb2)
- compatibilidade = r + g + b

- O resultado do histograma com ele mesmo, resulta no máximo, que é 3,0

1ª parte da Tarefa

- Sua base de dados (Base1) tem 8 imagens-modelo:
  1) America
  2) Batman
  3) Ferro
  4) Flash
  5) Hulk
  6) Maravilha
  7) Super
  8) Wolverine

1ª parte da Tarefa

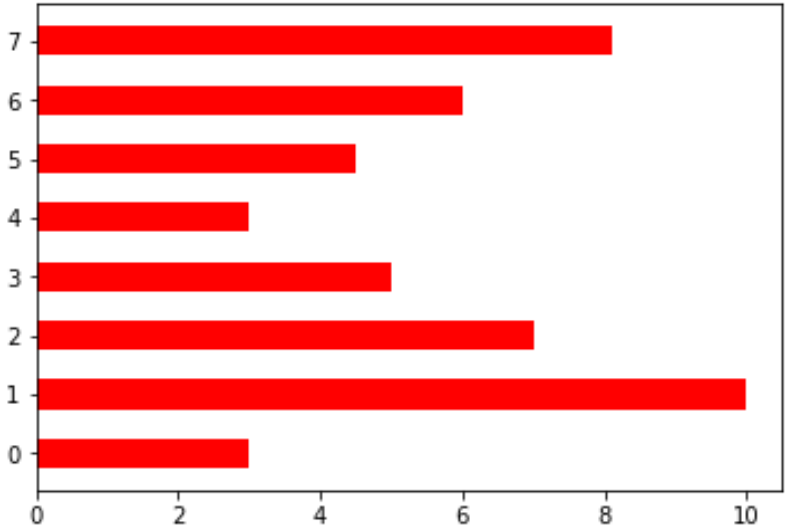
- E tem 5 imagens-teste:
  1) QUEM1
  2) QUEM2
  3) QUEM3
  4) QUEM4
  5) QUEM5

1ª parte da Tarefa

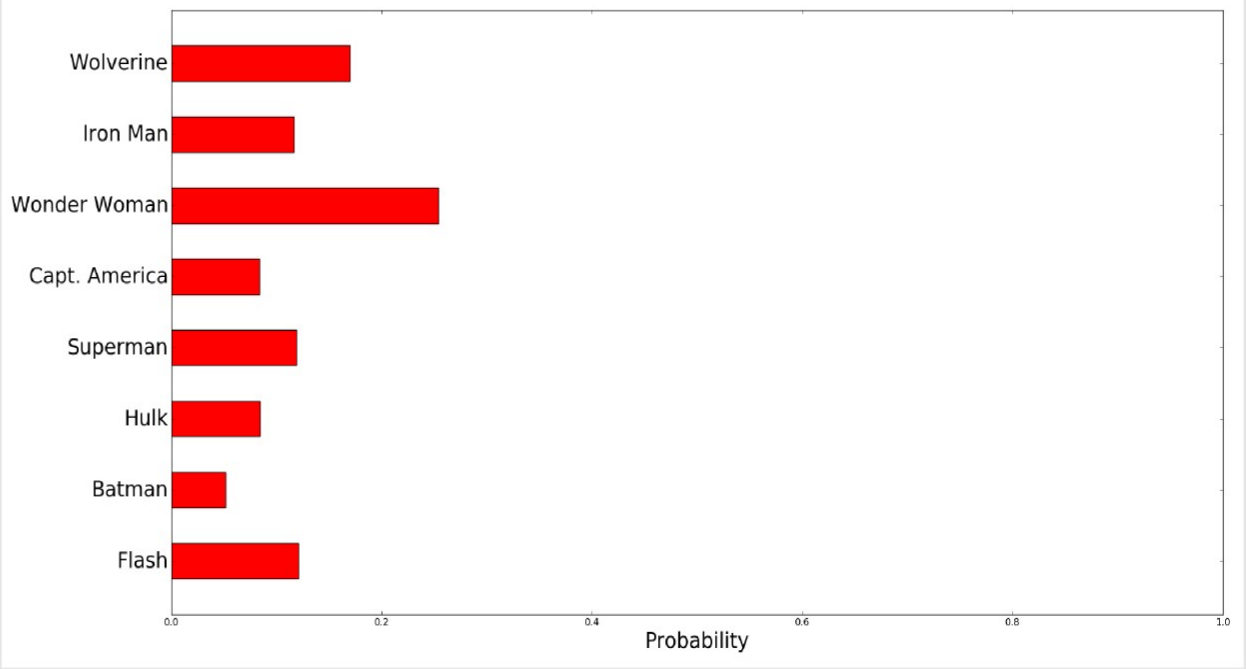
- Compare os histogramas das imagens-teste com as imagens-modelo
- Use diferentes formas de cálculo de histograma (via matplotlib e via OpenCV)
- Analise os erros e acertos. Faça uma análise comparativa.

2ª parte da Tarefa

- Armazene o resultado em um array, e depois apresente-os em um gráfico de barras. Veja um exemplo
- y = [3, 10, 7, 5, 3, 4.5, 6, 8.1]
- N = len(y)
- x = range(N)
- width = 0.5
- plt.barh(x, y, width, color="red")
- plt.show()



Com o gráfico é possível avaliar quem é o mais parecido



3ª parte da Tarefa

- Mantenha as 8 imagens-modelo e faça os testes usando a Base2 como testes.
- Use outras formas de comparação de histograma, que são descritas em: http://www.pyimagesearch.com/2014/07/14/3-ways-compare-histograms-using-opencv-python/
- Com as notas formas de comparação de histogramas, refaça os testes com todas as imagens iniciadas com “quem”.
-

4ª parte da Tarefa

- Monte **sua própria base** de dados, em que a cor seja o elemento principal para a classificação das imagens:
  - Monte uma base de imagens com os modelos
  - Monte uma base de imagens de testes

- Exemplo de base de dados **não** válido: https://www.pyimagesearch.com/2014/01/27/hobbits-and-histograms-a-how-to-guide-to-building-your-first-image-search-engine-in-python/

5ª parte da Tarefa

- O que pode ser melhorado na solução?
- Faça 1 proposta de melhoria.
- Na apresentação da tarefa, deve evidenciar esta modificação.
- Refaça os testes nas duas bases de dados.

Produtos da Tarefa

- Produto 1 – Solução inicial da 1ª parte com as duas formas de histograma
- Produto 2 – Solução com resultado gráfico
- Produto 3 – Solução com diferentes formas de comparação de histogramas
- Produto 4 – Resultados com a sua base de dados
- Produto 5 – Solução com proposta de melhoria
- Todos os códigos devem estar no formato Jupyter Notebook, em que sejam incluídos textos explicando o que foi feito e os resultados obtidos
- Slides da apresentação