

# Processamento e Análise de Imagens Digitais

Exercício 03 - Segmentação usando  $K$ -médias

**Artur Rodrigues Rocha Neto - 431951**

Mestrando em Engenharia de Teleinformática

`artur.rodrigues26@gmail.com`

**André Washington Moraes de Freitas - 407343**

Doutorando em Engenharia de Teleinformática

`andre.was12@gmail.com`

26 de Março de 2019

## 1 Introdução

Clusterização faz parte de diversos *pipelines* de predição. Ela envolve o conjunto de algoritmos responsáveis por agrupar objetos/entidades/amostras a partir de medidas de semelhança, ou seja, amostras de um dado grupo são mais parecidas entre si que entre amostras de um outro grupo. É uma técnica dita não-supervisionado. As técnicas de clusterização e validação ajudam a revelar padrões entre as observações.

No contexto de análise de imagens digitais, a clusterização é usada na tarefa de segmentação. Podemos enxergar os diferentes níveis de intensidades como as possíveis classes a serem agrupadas. A clusterização também é usada em compressão; agrupando um conjunto de *pixels* em uma imagem pela sua similaridade, assim reduzindo a quantidade de memória necessária para representá-la.

Abordaremos o uso do algoritmo  $k$ -Médias na tarefa de segmentação. Diferentes métricas de distância foram usadas e comparadas em imagens de *benchmark*. Códigos escritos em Python<sup>1</sup> foram implementados fazendo uso da biblioteca OpenCV [1].

## 2 $K$ -médias

O  $K$ -médias é um método de clusterização (ou agrupamento) que arranja massas de dados em  $n$  conjuntos bem separados e de igual variância. Seu funcionamento baseia-se na minimização de um critério chamado *soma das distâncias quadráticas*, do inglês *Squared Sum Distance* ou apenas SSD. O índice SSD é conhecido também como inércia: quanto menor a inércia de um ponto, menos esse ponto "se moveu" de uma interação a outra. A inércia, portanto, é o critério de convergência do  $K$ -médias. O algoritmo pode ser descrito como [2]:

**Passo 1:** Definir um valor para  $K$ .

**Passo 2:** Atribuir valores iniciais aos  $K$  protótipos.

---

<sup>1</sup><https://github.com/keizerzilla/pdi-pos/blob/master/T03.py>

**Passo 3:** Determinar a partição  $V_i$  do protótipo  $w_i$ ,  $i = 1, 2, \dots, K$ , usando a Eq.1:

$$V_i = \{x \in \mathbb{R}^p \mid \|x - w_i\| < \|x - w_j\|, \forall j \neq i\} \quad (1)$$

**Passo 4:** Calcular a nova posição do protótipo  $w_i$  como a média dos  $N_i$  objetos da partição  $V_i$ :

$$w_i = \frac{1}{N_i} \sum_{x \in V_i} x \quad (2)$$

**Passo 5:** Repetir os Passos 3 e 4 até a convergência do algoritmo.

A implementação do  $K$ -médias no pacote ***scikit-learn*** [3] encontra-se na classe `sklearn.cluster.KMeans`.

### 3 Métricas de Distância

Quatro métricas de distância foram escolhidas para avaliar o resultado da clusterização: euclidiana (3), *manhattan* (4), cosseno (5) e chebyshev (6).

$$d_e = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2} \quad (3)$$

$$d_m = |x_b - x_a| + |y_b - y_a| \quad (4)$$

$$d_{cos} = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|} \quad (5)$$

$$d_{chbv} = \max(|x_b - x_a|, |y_b - y_a|) \quad (6)$$

### 4 Resultados

Para mostrar os efeitos de direntes distâncias diferentes valores de agrupamento  $K$ , dois cenários de experimentação foram escolhidos: imagem em tons de cinza e imagem colorida.

A Figura 1 mostra o resultado do  $K$ -médias usando distância euclidiana em uma imagem em escala de cinza. O resultado se assemelha ao de uma quantização. A Figura 2 traz uma comparação das distâncias escolhidas com quatro escolhas de agrupamento: 4, 8, 16 e 32 grupos. Diferentes distâncias resultam em diferentes componentes de cor e contrast mantidas.



Figura 1: Métrica euclidiana em imagem tom de cinza



Figura 2: Resultado comparativo

## Referências

- [1] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
- [2] Guilherme de Alencar Barreto. Introdução à clusterização de dados. Slides da disciplina TIP8311 - Reconhecimento de Padrões, 2018.
- [3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.