Trabalho 5

MO443 - Introdução ao Processamento de Imagem Digital

Thales Eduardo Nazatto
074388
tenazatto@gmail.com

I. Introdução

Clusterização ou agrupamento, conforme mencionado em [1], são técnicas de aprendizado não-supervisionado utilizadas para descobrir padrões não descobertos anteriormente sem depender de dados ou padrões anteriores. Dentre elas, a mais comum é o *K-Means*, que consiste em achar agrupamentos em áreas definindo um número k de centróides. Um algoritmo *K-Means* se define da seguinte maneira:

1) Definir quais os k centróides.
2) Encontrar o centróide mais próximo e atualizar os agrupamentos com base nesses centróides.
3) Mover os centróides para o centro dos seus agrupamentos.
4) Repetir os passos 2 e 3 até a convergência do algoritmo.

Em processamento de imagens, é possível executar técnicas de aprendizado não supervisionado para realizar a quantização de uma imagem colorida, regerando uma imagem com um menor número de cores e afetando a qualidade da aparência da mesma o mínimo possível. Neste trabalho, o objetivo é utilizar o algoritmo *K-Means* para realizar quantizações com determinados números de cores para verificar sua possibilidade e sua eficiência.

II. Metodologia

A. Execução e utilização do programa

Para quantizar e recriar a imagem de entrada, foi criado um programa na linguagem *Python*. É feito o agrupamento utilizando o *K-Means* para obter os centróides e estes são utilizados para colorizar a nova imagem.

Para executar o programa, é necessário o *Python 3* e são necessárias as seguintes bibliotecas instaladas:

- *NumPy*: Utilizada para realizar os cálculos matriciais necessários para manipular as imagens.
- *OpenCV*: Utilizado para manipular as imagens de entrada e gerar as imagens de saída.
- *Scikit-Learn*: Utilizada para aplicar o algoritmo *K-Means* no espaço de cores RGB.
- *MatPlotLib*: Utilizada para processar os gráficos dos agrupamentos resultantes das imagens.

Para utilizar o programa, *flags* foram configuradas para realizar experimentos de forma separada:

- **-images**: Utilizada para listar as imagens que serão combinadas para criar a nova imagem panorâmica. Parâmetro obrigatório.
- **-ncolors**: Utilizada para enumerar o número de cores resultantes da quantização para a nova imagem. Parâmetro obrigatório.
- **-show-groups**: Utilizada para exibir um gráfico do agrupamento resultante das imagens.

B. Entrada e Saída de dados

Como entrada de dados, foi utilizado um banco de imagens presente em [3], que possui 4 imagens no formato PNG. A saída de dados serão as imagens quantizadas resultantes também no formato PNG, disponibilizados em uma pasta **result_images** que o programa também criará caso não exista.

Caso uma imagem no formato JPEG seja passada como entrada, a saída também será no formato JPEG. Caso a flag **-show-groups** seja passada, também serão geradas imagens a respeito dos agrupamentos das cores de cada imagem, colocadas em um gráfico no espaço de cores RGB.

C. Abordagens tratadas nos algoritmos

O algoritmo de quantização, conforme detalhado em [2], consiste em 4 passos:

1) Ler a imagem colorida de entrada.
2) Aplicar o algoritmo *K-Means* para encontrar grupos de cores mais representativas.
3) Salvar dicionário (*codebook*) gerado pelo *K-Means*, contendo os centróides dos grupos e os rótulos correspondentes a cada pixel da imagem.
4) Reconstruir a imagem com cores reduzidas a partir do dicionário armazenado.

Para o passo 1, utilizamos o *OpenCV* para ler a imagem, transformando a imagem em uma matriz de cores compatíveis com o *NumPy*. Para os passos 2 e 3, normalizamos as cores no intervalo $[\ 0\ ,\ 1\ ]$ dividindo a matriz por 255 e transformamos a matriz em um vetor de cores, aplicando a função **shuffle** do *Scikit-Learn* para embaralhá-las uniformemente, e em seguida utilizamos o objeto **KMeans** da mesma biblioteca, executando o algoritmo e obtendo os centróides com a função **fit** e obtendo os rótulos correspondentes a cada pixel com a função **predict**. Para o passo 4, uma nova matriz foi construída e preenchida com os dados obtidos no passo 3, gerando uma nova imagem colorizada em seguida.

III. Experimentos e Discussão

O único experimento realizado para este trabalho foi a execução do programa presentes no banco de imagens presente em [3] na quantidade de cores de 16, 32, 64, 128 e 256 cores.

Duas coisas foram notadas: A diferença na qualidade entre as imagens e a compressão entre as diferentes quantizações. Quanto a compressão, os dados estão na Tabela I informada abaixo.

Tabela I
TAMANHO E COMPRESSÃO DAS IMAGENS QUANTIZADAS

	Imagens							
	baboon.png		monalisa.png		peppers.png		watch.png	
Cores	Tamanho(KB)	Compressão(%)	Tamanho(KB)	Compressão(%)	Tamanho(KB)	Compressão(%)	Tamanho(KB)	Compressão(%)
24 bits	622,26	0	108,17	0	526,12	0	680,72	0
256	601,66	3,31	119,42	-10,4	478,96	8,96	640,44	5,91
128	576,08	7,42	113,33	-4,77	433,75	17,56	572,87	15,84
64	529,39	14,92	105,53	2,44	368,71	29,92	508,77	25,26
32	460,22	26,04	89,64	17,13	295,43	43,85	421,57	38,07
16	369,17	40,67	65,9	39,08	206,09	60,83	341,61	49,82

Percebemos que quanto menor o número de cores presente na quantização, maior é a compressão de tamanho realizada pela mesma. Entretanto, apareceram resultados curiosos de que o tamanho da imagem quantizada foi maior que o tamanho da imagem original. O que se pode concluir disso é que o método da compressão PNG realizada pela imagem original é mais eficiente do que o utilizado pelo *OpenCV* em alguma maneira, uma vez que existem diferentes versões do formato.

Quanto a qualidade, todas sofreram perdas perceptíveis mas o formato da imagem permaneceu inalterado, comprovando a eficiência do *K-Means* para realizar tal feito. Contudo, uma análise feita na quantização em 16 cores mostrada na Figura 1, as imagens *monalisa.png* e *watch.png* foram as que tiveram perdas mais perceptíveis, com ondulações presentes devido a iluminação e sombreamento da imagem.

Para descobrir algo melhor a respeito das razões de acontecerem essas ondulações foram feitos gráficos a respeito dos agrupamentos e dos centróides das imagens, presentes na Figura 2. Nele, pode-se perceber que, no espaço de cores RGB, as distribuições de cores das imagens *monalisa.png* e *watch.png* são mais densas e parecem estar em um plano dentro do espaço, o que pode ocasionar em centróides menos próximos, cores mais evidentes e causar as ondulações em uma quantização menor. Já as distribuições de cores das imagens *baboon.png* e *peppers.png* são mais dispersas e presentes em todo o espaço, o que pode ocasionar em diferenças de cores menos evidentes apesar da perda de qualidade ser teoricamente a mesma.

Tabela II
CORES PRESENTES NAS IMAGENS

Imagem	Número de cores
baboon.png	230427
monalisa.png	28059
peppers.png	183525
watch.png	45493

Um outro ponto que pode ter acontecido se deve ao número de cores. Conforme mostrado na Tabela II, as imagens *monalisa.png* e *watch.png* tem um menor número de cores, o que pode ocasionar em centróides mais dispersos ao longo do espaço de cores RGB.

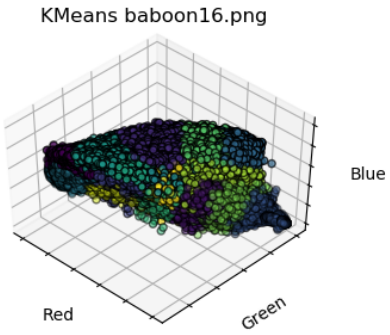
(a) Quantização de *baboon.png*

(b) Quantização de *monalisa.png*

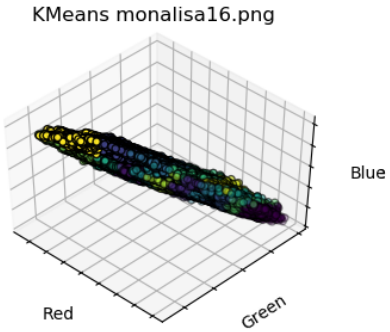
(c) Quantização de *peppers.png*

(d) Quantização de *watch.png*

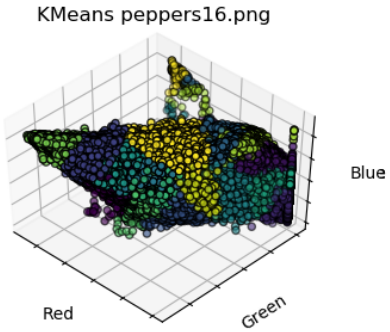
Figura 1. Quantização em 16 cores.



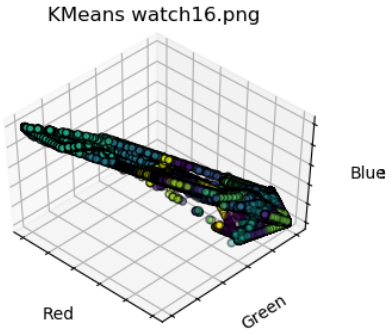
(a) Distribuição de *baboon.png*



(b) Distribuição de *monalisa.png*



(c) Distribuição de *peppers.png*



(d) Distribuição de *watch.png*

Figura 2. Distribuição de cores das imagens na quantização em 16 cores.

IV. Conclusões e Trabalhos Futuros

Podemos concluir que algoritmos de aprendizado não supervisionado com base na obtenção de centróides como o *K-Means* podem ser extremamente úteis para processamento de imagens, uma vez que os agrupamentos de cores podem ser substituídos por um único centróide e não afetam a qualidade de sua aparência. Na maioria dos casos, este algoritmo também consegue efetuar uma compressão relativamente eficiente uma vez que menos cores serão utilizadas, podendo ser utilizado para economizar dados. A qualidade acaba sendo sacrificada, mas dependendo da imagem este sacrifício não é percebido por um olhar humano menos técnico. Tais características também implicam também que ele pode ser utilizado para aumentação de dados em aprendizado de máquina, uma vez que o conteúdo da imagem mudou sem afetar muito suas características para o olho humano.

Como trabalhos futuros, podemos realizar implementações com outros algoritmos de aprendizado não-supervisionado baseados na descoberta de centróides, como o *K-Medians*, o *K-Medoids* ou o *Fuzzy C-Means*.

Referências

[1] http://www.ic.unicamp.br/~helio/disciplinas/MO443/aula_agrupamentos.pdf
[2] http://www.ic.unicamp.br/~helio/disciplinas/MO443/trabalho5.pdf
[3] http://www.ic.unicamp.br/~helio/imagens_coloridas/