Organização Automática de Fotografias de Eventos Usando K-Means

Bárbara Zamperete, Mateus Moura

Resumo - Este trabalho mostra uma solução para organização de coleções de fotos usando o algoritmo de clusterização, como K-means. O objetivo é agrupar imagens com base em características visuais parecidas, facilitando o agrupamento de fotos de eventos como aniversários, casamentos e formaturas. O trabalho utiliza funções para obter a cor dominante, o histograma de cores e HOG features para criar uma matriz de características utilizada para a clusterização. Em seguida analisa a complexidade do algoritmo desenvolvido e realiza testes com conjuntos de dados de diferentes tamanho visando entender o seu funcionamento e os seus resultados conforme o tamanho da entrada varia. Como resultado, percebeu-se que essa ferramenta funciona melhor conforme a quantidade de imagens aumenta, no entanto os resultados ainda são imprecisos.

Palavras-chave: Clusterização, Organização de Imagens, K-Means.

I. INTRODUÇÃO

Com o aumento do uso de smartphones e câmeras digitais, a tarefa de organizar grandes volumes de fotos tornou-se cada vez mais desafiadora. Eventos como casamentos, formaturas e aniversários geram centenas ou até milhares de imagens, e categorizá-las manualmente é um processo exaustivo e demorado. Para abordar esse problema, este trabalho aplica o algoritmo de clusterização K-Means, que organiza as imagens em grupos com base em características visuais, como histogramas de cores, HOG features e cores dominantes.

Na prática, o K-Means pode ser utilizado por fotógrafos profissionais que precisam organizar fotos rapidamente após eventos ou por empresas de gerenciamento de imagens que oferecem serviços de curadoria automática de grandes coleções. Por exemplo, ao término de um casamento, as fotos podem ser automaticamente agrupadas em categorias como "cerimônia", "recepção" e "fotos com os noivos", otimizando o fluxo de trabalho. Além disso, algoritmos com esse mesmo fim têm sido amplamente utilizado em sistemas de gestão de bibliotecas de imagens, como o Google Photos, que utiliza clusterização para agrupar imagens semelhantes.

O presente estudo explora também o custo de complexidade do algoritmo K-Means, testando a clusterização em conjuntos de dados de diferentes tamanhos. A análise permite identificar como o desempenho do algoritmo varia conforme o volume de dados, fornecendo insights sobre sua escalabilidade em aplicações práticas. O objetivo é oferecer uma solução simples e satisfatória para a organização automática de fotos em categorias, reduzindo o esforço manual envolvido.

II. O ALGORITMO

O K-Means é um algoritmo de aprendizado de máquina não supervisionado usado para clusterizar dados em K grupos, onde K é um parâmetro definido pelo usuário. No contexto de imagens, ele agrupa imagens com base em características visuais semelhantes. O processo do K-Means pode ser descrito em três etapas principais:

- Inicialização: Seleciona-se aleatoriamente K centroides que representam os centros iniciais dos clusters.
- Atribuição: Cada ponto de dados (neste caso, as imagens representadas pelas features extraídas) é atribuído ao cluster cujo centróide está mais próximo.

A métrica de proximidade mais comum é a distância euclidiana.

- Atualização: O centróide de cada cluster é recalculado como a média dos pontos de dados atribuídos a ele.
- Convergência: O processo de atribuição e atualização se repete até que não haja mudanças significativas nos centróides ou um número máximo de iterações seja atingido.

O código fornecido utiliza o K-Means para encontrar agrupamentos baseados nas características extraídas das imagens (histograma de cores, HOG features, e cores dominantes). A função k-means aplicada no código realiza essa tarefa.

A. Histograma de Cores

Um histograma de cores é uma representação da distribuição de cores em uma imagem. No código, as imagens são convertidas para o espaço de cores HSV (Hue, Saturation, Value), e o histograma é calculado com base em três canais (matiz, saturação e valor). A função "calcHist()" é usada para gerar o histograma, que, em seguida, é normalizado para valores entre 0 e 1.

Os histogramas de cores são úteis para descrever o conteúdo visual de uma imagem, pois capturam a distribuição de cores de maneira robusta a pequenas mudanças na posição ou orientação dos objetos.

B. HOG Features (Histogram of Oriented Gradients)

As HOG features (Histogramas de Gradientes Orientados) são uma técnica que descreve a forma e a estrutura dos objetos em uma imagem com base nos gradientes de intensidade. Esses gradientes são agrupados em bins que representam diferentes direções de borda na imagem, o que torna essa técnica eficaz para capturar a textura e as bordas dos objetos.

No código, a função HOGDescriptor cria um descritor HOG com parâmetros como tamanho da janela, blocos, e células. O método hog.compute() é então utilizado para calcular as características HOG de uma imagem.

Essa técnica é particularmente eficiente em tarefas de reconhecimento de objetos e é frequentemente usada em combinações com algoritmos de classificação ou clusterização, como o K-Means.

C. Extração de Cores Dominantes

A extração de cores dominantes utiliza o próprio K-Means para identificar as cores mais frequentes em uma imagem. No código, a imagem é convertida para o espaço RGB, redimensionada para uma matriz de pixels, e o K-Means é aplicado para identificar K cores predominantes. As cores dominantes podem ser usadas para descrever a paleta de cores de uma imagem, facilitando sua classificação ou busca.

No código, a função kmeans() é utilizada novamente para agrupar os pixels em K clusters, e as cores centrais desses clusters são consideradas as cores dominantes da imagem.

III. METODOLOGIA

Neste estudo, utilizamos o algoritmo de clusterização K-Means para agrupar automaticamente imagens de eventos, como casamentos, formaturas e aniversários, com base em características visuais extraídas de cada imagem. As features selecionadas incluem histogramas de cores, HOG features e cores dominantes. O objetivo é analisar a complexidade do algoritmo e o seu funcionamento para diferentes tamanhos de entrada.

Foram utilizados conjuntos de imagens de eventos contendo 30, 60, 90, 120, 150 e 210 fotos. As imagens foram armazenadas em diretórios específicos e carregadas para o ambiente de execução. Passaram também por um processo de redimensionamento para uma resolução fixa de 128x128 pixels, a fim de padronizar o tamanho dos dados e acelerar o processamento.

Em seguida, para cada imagens foram extraídas as suas características: cor dominante, HOG features e histograma de cores.

Essas três características foram combinadas em vetores descritivos para cada imagem, resultando em uma representação numérica que alimenta o processo de clusterização.

Após a extração das características, o k-means foi aplicado sobre esses vetores para agrupar as imagens em clusters, representando categorias de eventos visuais semelhantes. O número de clusters foi definido empiricamente (k=3) devido ao testes se limitarem a uma variação pré-definida de tipos de imagens (aniversários, casamentos e formaturas).

A função de clusterização, implementada em C++ e utilizando as bibliotecas OpenCV e Eigen, recebe os vetores de características como entrada. A normalização dos dados foi realizada para garantir uma distribuição consistente dos valores antes de serem passados ao K-Means. A execução do algoritmo gerou agrupamentos que categorizam as fotos em diferentes eventos de acordo com suas características visuais.

Por fim, foram realizados testes com diferentes quantidades de imagens (30, 60, 90, 120, 150 e 210), de forma a avaliar o desempenho e a escalabilidade do algoritmo à medida que o volume de dados aumenta. Em cada teste, as imagens foram agrupadas em clusters com base nas características previamente extraídas. Os resultados mostraram como o aumento do número de imagens impacta a eficiência e a precisão da clusterização, permitindo uma análise do custo computacional do algoritmo.

O código completo da implementação está disponível no repositório GitHub, através do link: https://github.com/mateusmoraes99/MateusBarbara_FinalProject_AA_RR_2024.

IV. COMPLEXIDADE

O algoritmo desenvolvido foi analisado assintoticamente, a fim de compreender e prever o seu funcionamento conforme a quantidade de imagens aumenta.

A. Funções principais

O algoritmo inicia na função “main()”, onde é inicializado o caminho para o diretório contendo as imagens e o diretório de saída (onde as imagens serão divididas em clusters).

Essas operações de atribuição tem complexidade constante O(1).

Algoritmo 1 Função Main()

```
int main(){

    string dir_path = "imagens/img210";
    vector<string> extensions = { ".jpg", ".jpeg" };
     string output_folder = "clusterizacao_resultados";

    load_images_and_extract_features(
    dir_path, extensions, output_folder);
```

A função “load_images_and_extract_features” realiza e chama todas as operações necessárias para o cálculo do algoritmo, a sua complexidade define a complexidade geral do código.

Algoritmo 2 Resumo da função load_images_and_extract_features ()

```
void load_images_and_extract_features ( const string&
dir_path, const vector<string>& extensions, const string&
output_folder
)

    Inicialização de variáveis

    Para cada imagem faça {
       Leitura da imagem
       Redimensionamento
       Extração de características
       Concatenação das características extraídas
    }

    Matriz de características
    Clusterização
    Copiar e separar as imagens
```

Essa função pode ser dividida em partes.

a) Primeira parte:

Inicialização de variáveis e atribuição de valores, custo de complexidade O(1).

b) Segunda para:

Para cada imagem são feitas operações de:

- Redimensionamento: O(WxH),
- Histograma de cores: O(128x128),
- Cor dominante: O(128x128x5),
- HOG: O(128x128)
- Concatenação das características: O(F) onde F é a quantidade de características extraídas.

Sendo assim, para cada N imagem, as operações feitas dependem do tamanho da imagem de entrada e da quantidade F de características extraídas, resultando em uma complexidade O(N x (H x W + F).

c) Terceira parte:

Com as características extraídas de todas as imagens, é formado então uma matriz de características $\left[M_{NxF}\right]$. onde N é a quantidade de imagens e F as features extraídas de cada imagem. Sendo assim, o custo dessa operação depende de N e de F, resultando então na complexidade O(NxF).

d) Quarta parte:

Nessa etapa a matriz da fase anterior é passada na forma de uma array unidimensional para o algoritmo k-means, que faz a clusterização das imagens com base em suas características e a quantidade k de clusters passado como parâmetro. No cenário deste trabalho, k é um valor constante definido como k=3.

Sendo assim, a complexidade dessa parte pode ser expressa como O(NxF).

e) Quinta parte:

Por fim, os resultados obtidos pelo k-means são utilizados para copiar e organizar as imagens em subdiretórios.

A função “copy()” tem o custo dependente do tamanho da imagem O(W x H) e é chamada para cada N imagem, totalizando O(N x W x H)

A complexidade total da função “load_images_and_extract_features()” pode ser expressa pela soma da complexidade das operações que a compõem, considerando os termos de maior influência.

Total = O(1) + O(N x H x W + N x F)+ O(N x F) + O(N x F) + O(N x H x W)

Total = O(N (H x W + F))

Esse resultado mostra como a complexidade do algoritmo depende da quantidade de imagens, o tamanho delas e a quantidade de características extraídas de cada uma.

V. TESTES

Nessa seção será exposto os resultados da distribuição das imagens nos clusters, para testes realizados com diferentes tamanhos de dados.

Distribuição para o total de 30, 60 e 90 imagens respectivamente:

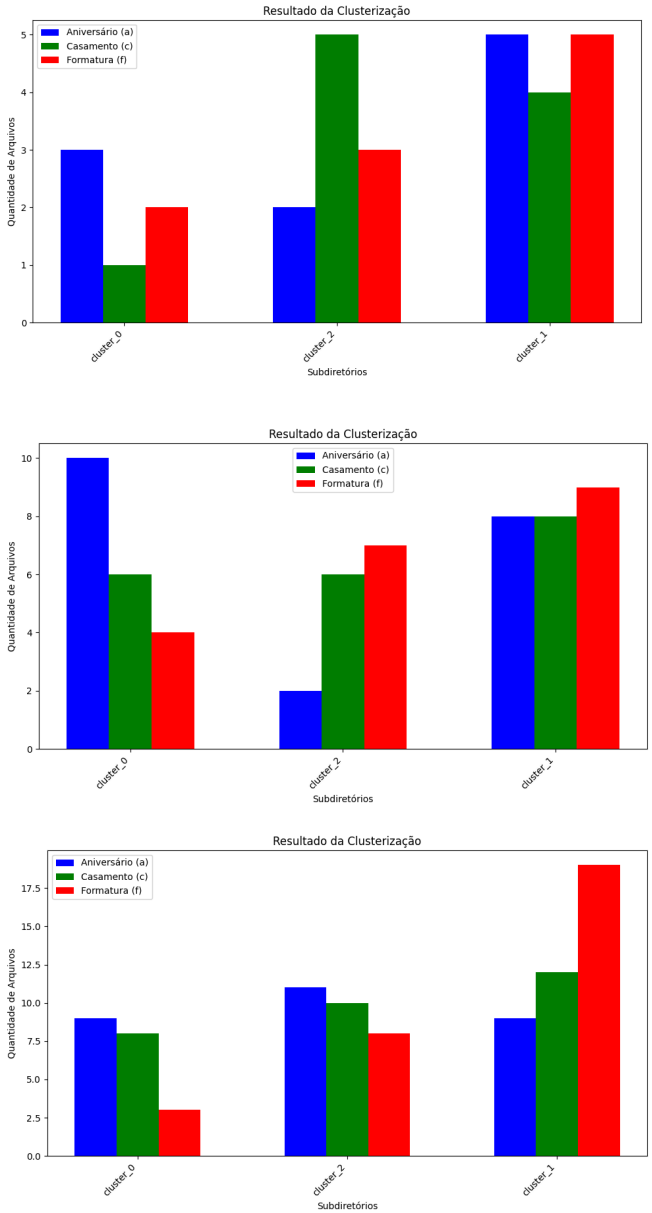

Fig 1. Distribuição para o total de 30, 60 e 90 imagens respectivamente

Percebe-se que para os conjuntos pequenos de dados, as divisões não são satisfatórias. O esperado é que cada categoria de imagem (aniversário, formatura ou casamento) se sobressaia em cada cluster. Nos testes que foram realizados com poucas imagens, isso não ocorreu. Muitas vezes imagens da mesma categoria foram predominantes em mais de um cluster.

Já quando a quantidade de imagens passadas como entrada aumenta, sutilmente essa predominância única de cada categoria passa a aparecer, como pode ser observado nas imagens (Fig 2).

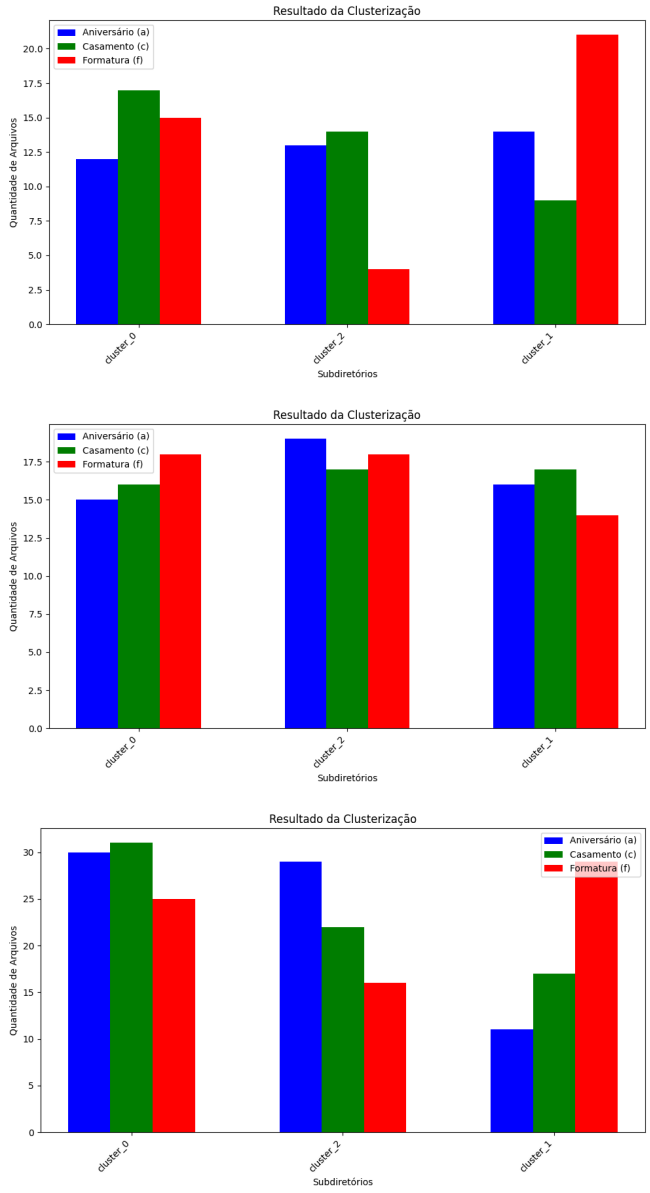

Fig 2. Distribuição para o total de 120, 180 e 210 imagens respectivamente

Essa leve melhora nos resultados leva a crer que em conjuntos muito maiores, como por exemplo com mais de 1.000 imagens, a divisão será mais precisa e útil em um cenário real. De forma a facilitar o trabalho fotógrafos profissionais que precisam organizar fotos rapidamente após eventos.

No entanto, é importante lembrar que a complexidade O(N (H x W + F)) faz com que o custo desse algoritmo aumente linearmente com o aumento do conjunto de dados, além do tamanho das imagens também impactar no custo (imagens profissionais costumam ter resolução mais alta).

VI. CONCLUSÃO

Foi apresentada uma solução prática para a organização automática de grandes coleções de fotografias de eventos, utilizando o algoritmo de clusterização K-Means. A aplicação de técnicas de extração de características visuais, como histograma de cores, Histogram of Oriented Gradients (HOG) e extração de cores dominantes, permitiu uma representação otimizada das imagens para o processo de agrupamento. Os resultados mostraram que o algoritmo apresentou bom desempenho ao agrupar imagens em categorias visualmente semelhantes, sendo que conjuntos de dados maiores tendem a produzir agrupamentos mais precisos e úteis, especialmente em contextos práticos.

A análise da complexidade demonstrou que o algoritmo é escalável, porém o custo computacional aumenta linearmente com o número e o tamanho das imagens. Para trabalhos futuros, recomenda-se explorar técnicas mais avançadas de extração de características ou ajustar os parâmetros do K-Means, visando melhorar a precisão e reduzir o custo computacional em ambientes com grande volume de imagens.

VII. REFERÊNCIAS

[1] T. N. Pappas and N. S. Jayant, "An adaptive clustering algorithm for image segmentation," International Conference on Acoustics, Speech, and Signal Processing, Glasgow, UK, 1989, pp. 1667-1670, vol. 3, doi: 10.1109/ICASSP.1989.266767.

[2] O. Marques, *Practical Image and Video Processing Using MATLAB*, Belmont, CA, USA: Wiley-IEEE Press, 2011, pp. 35–47.

[3] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed., Prentice Hall, 2007, pp. 225–265.

[4] D. Lowe, "Object recognition from local scale-invariant features," in *Proc. Int. Conf. Computer Vision*, 1999, pp. 1150-1157.