Estimando Áreas em Imagens Aplicando Kmeans

A. T. B. Couto – Inteligência Artificial, UFES

Abstract— The ability to perceive the area in an image is a matter of economy and practicality. A system for analyzing color patterns in an image and from a cluster analysis uses standard statistical algorithms to return the quantized area according to the number of clusters that are described. In this paper, we also implemented the Kmean clustering algorithm to analyze the images and visualize their importance in the image. The software described abuses the use of scientific libraries for best results and performance, leaving a door open for future updates.

Index Terms— IA, PDI, Processamento, Imagens, Geoprocessamento, Kmeans, Clustering, Área, Aproximação, Estimação)

1 Panorama Sobre Processamento De Imagens

Estamos durante um mundo visualmente encantador, que se manifesta com uma variedade de formas e formas, cores e texturas, movimento e tranquilidade. A percepção humana tem a capacidade de adquirir, integrar, interpretar toda esta informação visual abundante ao nosso redor. É um desafio para transmitir tais recursos a uma máquina para interpretar as informações visuais embutido em imagens, gráficos e imagens de vídeo ou em movimento mundo sensorial. É, portanto, importante compreender as técnicas de armazenamento, processamento, transmissão, reconhecimento e, finalmente, interpretação de tais visuais Cenas.

O primeiro passo é a aquisição da imagem digital utilizando sensores em comprimentos de onda ópticos ou térmicos. Uma imagem bidimensional que é gravada por esses sensores é o mapeamento do mundo visual tridimensional. Os sinais bidimensionais capturados são amostrados e quantificados para produzir imagens digitais.

Às vezes, recebemos imagens com ruído que são degradadas por algum mecanismo. Uma fonte comum de degradação de imagem é a lente óptica sistema em uma câmera digital que adquire a informação visual. Se a câmera não é focada adequadamente, então temos imagens turvas. Muitas vezes, pode-se vir através de imagens de cenas ao ar livre que foram adquiridos em um ambiente nebuloso. Assim, qualquer cena ao ar livre capturado em uma manhã de inverno nebuloso poderia resultar em uma imagem borrada. Em alguns outros casos, pode haver um movimento relativo entre o objeto e a câmera. Assim, se a câmera é deslocada impulsivamente durante o intervalo de captura de imagem, enquanto o objeto está estático, a imagem resultante será borrada e com ruído. Nos casos acima, precisamos técnicas apropriadas de refinar as imagens de modo que as imagens resultantes são de melhor qualidade visual, livre de borrões ou ruídos. A filtragem e a restauração foram algumas das aplicações importantes de processamento de imagens desde o princípio do campo.

Segmentação é o processo que subdivide uma imagem em algumas regiões uniformemente. A segmentação de uma imagem é definida por um conjunto de regiões que são conectados e sobrepostas, de modo que cada pixel em um segmento na imagem adquire um rótulo de região exclusiva que indica a região a que pertence. A segmentação é um dos elementos mais essenciais em análise automatizada de imagem, principalmente porque neste passo os objetos ou outros as entidades de interesse são extraídas de uma imagem para processamento subsequente, como descrição e reconhecimento.

Após a extração de cada segmento; a próxima tarefa é extrair um conjunto de características como textura, cor e forma. Estas são importantes entidades mensuráveis que dão medidas de várias propriedades de segmentos de imagem. Algumas das propriedades da textura são aspereza, lisura, regularidade, etc., quando os descritores comuns da forma são comprimento, largura, relação de aspecto, área, posição, perímetro, compactação, etc. Cada região segmentada em uma cena pode ser caracterizada por um conjunto de tais características.

Finalmente baseado no conjunto desses recursos extraídos, cada objeto segmentado é classificado para um de um conjunto de classes significativas. Em uma imagem digital do oceano, estas classes podem ser navios ou barcos pequenos ou mesmo navios navais e uma classe grande no corpo d'água. Os problemas de segmentação de cenas e classificação de objetos são duas áreas integradas de estudos em visão de computacional. Sistemas especializados, semântica redes, e sistemas de redes neurais foram encontrados para executar tais tarefas de visão de nível superior com bastante eficiência.

1.1 aplicações de processamento de imagem

Há muitas aplicações de processamento de imagem em diversos aspectos de atividades humanas a partir de interpretação de cena remotamente a interpretação da imagem biomédica. A seguir apenas um olhar superficial em algumas dessas aplicações:

1.1.1 Sistema automático de inspeção visual

Os sistemas automatizados de inspeção visual são essenciais para melhorar a produtividade e a qualidade do

• *A. T. B. Couto graduando em Engenharia da Computação na Universidade Federal do Espírito Santo, E-mail: andreww.max@hotmail.com.*

produto na indústria.

Identificação defeituosa do componente: a inspeção visual automatizada pode igualmente ser usada para identificar componentes defeituosos em um sistema eletrônico ou eletromecânico. Os componentes defeituosos criam mais energia térmica. As imagens de infravermelhos (IR) podem ser geradas a partir da distribuição de energias térmicas. Analisando estas imagens de infravermelhos, podemos identificar os componentes defeituosos na montagem.

Sistemas de inspeção automática de superfície: a detecção das falhas nas superfícies é exigência importante em muitas indústrias do metal. Por exemplo, nas usinas de laminadoras quentes ou frias em uma usina de aço, é necessário detectar qualquer aberração na superfície metálica. Isso pode ser realizado usando técnicas de processamento de imagem como detecção de borda, identificação de textura, análise fractal, e assim por diante.

1.1.2 Interpretação de cena remotamente

As informações sobre os recursos naturais, tais como agricultura, hidrologia, mineração, florestamento, recursos geológicos, etc., podem ser extraídas com base em análise de imagem remotamente. Para uma análise de cena detectada remotamente, imagens da superfície da terra são capturadas por sensores em satélites de sensoriamento remoto ou por um scanner Multiespectral alojado em uma aeronave e depois transmitido para a estação terrestre para processamento adicional.

Técnicas de interpretação das regiões e objetos em imagens de satélite são usados no planejamento urbano, mobilização de recursos, controle de inundações, monitoramento de produção agrícola, etc.

1.1.3 Técnicas de imagem biomédica

Diversos tipos de dispositivos de imagem como raio X, imagens auxiliadas por tomográfica computadorizada, ultrassom, etc., são amplamente utilizados para diagnóstico médico.

1. Localizar os objetos de interesse, ou seja, órgãos diferentes.
2. Tomar as medidas dos objetos extraídos, por exemplo, tumores na imagem
3. Interpretar os objetos para diagnóstico.

1.1.4 Vigilância

Aplicação de técnicas de processamento de imagem na defesa, vigilância ou segurança pública área focal de estudo. Há uma necessidade contínua de monitorar a terra e os oceanos usando técnicas de vigilância.

Suponha que estamos interessados em localizar os tipos e formação de navios navais em uma imagem aérea da superfície do oceano. A tarefa principal aqui é segmentar objetos diferentes na parte do corpo da água da imagem. Depois de extrair os segmentos, os parâmetros como área, localização, perímetro, compactação, forma, comprimento, largura e proporção são encontrados, para classificar cada um dos objetos segmentados. Estes objetos podem variar de pequenos barcos a enormes navios navais. Usando os recursos acima, é possível reconhecer e localizar esses objetos. Para descrever todas as formações possíveis dos navios, é necessário que sejamos capazes de identificar a distribuição destes objetos nas oito direções possíveis, nomeadamente, norte, Sul, leste, oeste, Nordeste, noroeste, sudeste e sudoeste. A partir da distribuição espacial desses objetos é possível interpretar toda a cena, que é importante para a defesa em certas situações.

1.1.5 Rastreamento de objeto

Rastreamento de objetos em movimento, para medir os parâmetros de movimento e obter um registro visual do objeto em movimento, é uma área crítica de aplicação em processamento de imagem. Em geral, há duas abordagens diferentes para o controle de objeto:

1. Acompanhamento baseado em reconhecimento.
2. Acompanhamento em movimento.

Um sistema de rastreamento de alvos rápidos (por exemplo, um avião militar, míssil, etc.) é desenvolvido com base em técnicas preditivas baseadas em movimento, como filtragem de Kalman, filtragem de Kalman estendida, filtragem de partículas, etc. Nos sistemas de rastreamento de objetos baseados em processamento de imagens automatizados, os objetos de destino que entram no campo de visão do sensor são adquiridos automaticamente sem intervenção humana. No rastreamento baseado em reconhecimento, o padrão de objeto é reconhecido em quadros de imagens sucessivos e o acompanhamento é realizado usando suas informações posicionais.

2 PANORAMA DO KMEANS

O Kmeans é uma das questões mais antigas e mais importantes em toda a geometria computacional. Dado um inteiro k e um conjunto de n pontos de dados em R^d, o objetivo é escolher k centros para minimizar o erro, a distância total ao quadrado entre cada ponto e seu centro mais próximo.

Resolver esse problema que é difícil classificado como NP-hard, mas vinte e cinco anos atrás, Lloyd propôs uma solução de pesquisa local para este problema que ainda é muito amplamente utilizado hoje. Na verdade, um levantamento de 2002 de técnicas de mineração de dados afirma que *"é de longe o algoritmo mais popular utilizado em aplicações científicas e industriais"*.

Normalmente referida simplesmente como "Kmeans", algoritmo Lloyd começa com k "centros", tipicamente escolhidos aleatoriamente a partir dos pontos de dados. Cada ponto é então atribuído ao centro mais próximo, e cada centro é recalculado como o centro de massa de todos os pontos atribuídos a ele.

2.1 Kmeans Clustering

Clustering é um método para dividir um conjunto de dados em um número específico de grupos. É um dos métodos populares é clustering Kmeans. No Kmeans, ele particiona uma coleção de dados em um grupo de números k de dados. Ele classifica um determinado conjunto de dados em k número de clusters. O algoritmo Kmeans consiste em duas fases separadas. Na primeira fase calcula o centroide k e na segunda fase mede cada ponto do cluster que está mais próximo centroide. Existem diferentes métodos para definir a distância do centroide

mais próximo, e um dos métodos mais utilizados é a distância euclidiana. Uma vez que o agrupamento é feito ele recalcula o novo centroide de cada cluster com base nesse centroide, uma nova distância euclidiana é calculada entre cada centro e cada ponto, então atribui os pontos no cluster que têm distância euclidiana mínima. Cada cluster na partição é definido por seus objetos de membro e por seu centroide. O centroide para cada cluster é o ponto para o qual a soma de distâncias de todos os objetos nesse cluster são minimizados. Assim, Kmeans é um algoritmo iterativo em que minimiza a soma de distâncias de cada objeto para seu centroide, sobre todos os clusters.

Considerando uma imagem com definição de X x Y e a imagem deve ser um cluster com k clusters. Então p(x, y) é um pixel de entrada para o cluster e Ck são os centros de agrupamento. O algoritmo para Kmeans é sugerido como:

1. Inicializar o número de k clusters e seus centros.
2. Para cada pixel de uma imagem, calcule a distância euclidiana d, entre o centro e cada pixel de uma imagem usando a relação dada abaixo.

$$d = \|p(x, y) - c_k\|$$

3. Atribua todos os pixels ao centro mais próximo com base na distância d.
4. Depois de todos os pixels terem sido atribuídos, recalcule a nova posição do centro usando a relação abaixo.

$$c_k = \frac{1}{k} \sum_{y \in c_k} \sum_{x \in c_k} p(x, y)$$

5. Repita o processo até que satisfaça a tolerância ou o parâmetro de erro.
6. Reconstrua os pixels do cluster na imagem.

Embora Kmeans tem uma grande vantagem de ser fácil de implementar, ele tem algumas desvantagens. A qualidade dos resultados de um cluster final é dependente da seleção de centroide inicial. Assim, se o centroide inicial é aleatoriamente escolhido, ele vai ter resultado diferente para diferentes centros iniciais. Assim, o centro inicial será cuidadosamente escolhido para que obter o nosso desejo segmentação. E, a complexidade computacional é outro termo que precisamos considerar enquanto projetando o Kmeans. Baseia-se no número de elementos de dados, número de clusters e número de iteração.

3 O Problema;

3.1 Cálculo de área

No meio computacional temos os desafios, que são propostos ou criados a partir de uma ideia, nesse caso, o trabalho gerou a oportunidade de aprofundar em uma nova área, onde o processamento de imagens poderia ser aplicado como entrada, e poderia ser obtido um resultado dessa técnica aplicada à uma imagem, como análise e exibindo resultados satisfatórios.

A matemática tem seus métodos para a resolução deste problema com precisão e sem ela não seria possível aplicar nenhuma outra técnica, mas pode-se alcançar a otimização de fórmulas e métodos eficientes ou aproximados, com um menor tempo, custo ou mesmo para uso de prévias.

O problema em si é obter a área de formas em uma imagem, baseando-se em uma escala dada como parâmetro para o tamanho da imagem, onde o resultado é a porcentagem entre cada elemento identificado (cluster), que também é um parâmetro. A forma comum de abordagem é com a vetorização da imagem, e utilização de um software de processamento de imagens específico, mas neste trabalho, usaremos o conceito de inteligência artificial para otimizar a resolução do problema com Kmeans, aproximando o resultado em um tempo hábil, com precisão satisfatória, e abrangência, sem o uso de softwares complementares.

Tendo em vista que este trabalho tem o foco na aplicação do conceito de IA, as restrições entre a entrada e saída são:

1. A imagem deve estar discreta em cores, ou escala de cinza, para a definição da quantidade de cluster (para maior precisão);
2. A imagem deve estar totalmente preenchida, pois o algoritmo leva em conta todos os elementos da imagem.
3. O retorno do programa é apenas visual, por ter melhor visualização.

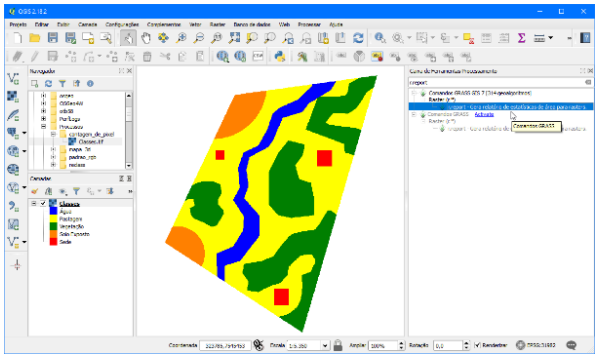
Estas restrições são de construção, pelo escopo da disciplina e podem afetar os resultados negativamente, mas o problema descrito, pode entrar como uma atualização para o código no futuro.

3.2 Método Atual

Um dos exemplos para a produção deste trabalho, foram imagens geográficas, que são amplamente utilizadas para obtenção de dados, há um grande gama de opções para o uso de técnicas de IA nesta área, então podemos considerar que é uma opção interessante de se trabalhar, finalmente podemos mostrar o problema, que consiste em um resumo para identificar o problema, segue o passo a passo comum para obter a área de uma imagem com cores discretas.

Tutorial

No QGIS, abra sua imagem classificada ou use esse demo para testar a ferramenta. No painel **processar**, faça uma pesquisa pelo algoritmo ***r.report***:

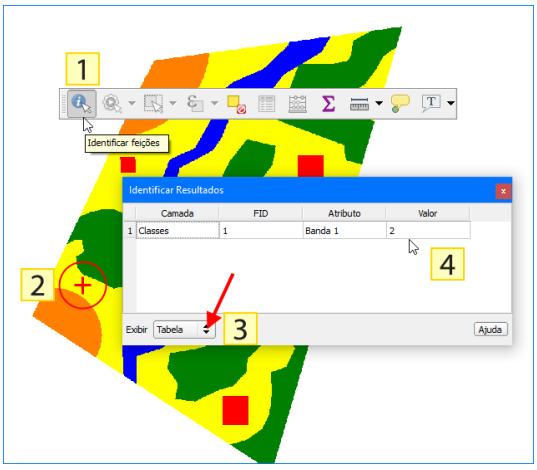


Identificando o Valor do Pixel

Para este trabalho, vamos obter o **cálculo de área por hectare**. No nosso raster modelo, para as classes Água, Pastagem, Vegetação, Solo Exposto e Sede, temos a seguinte

configuração de pixel: 1, 2, 3, 4 e 5.

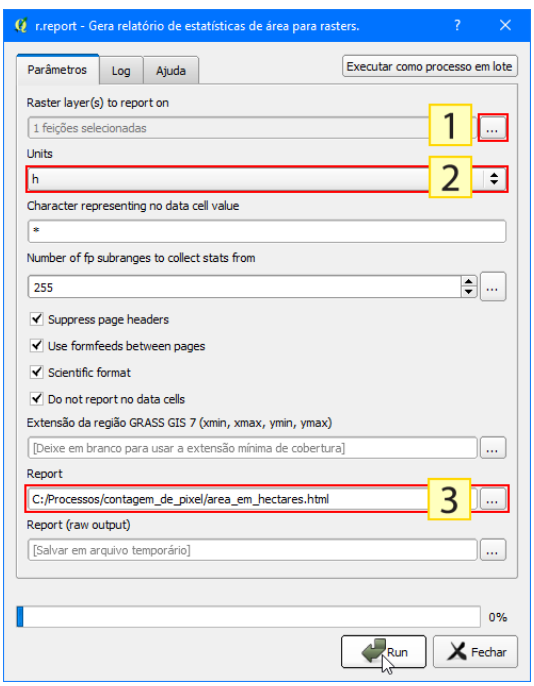
Se você não sabe como identificar o valor de um pixel na imagem, use a ferramenta **Identificar Feições**(1) e clique sobre uma área qualquer (2) do raster. No modo **Tabela** (3), visualize e anote o valor do pixel (4).



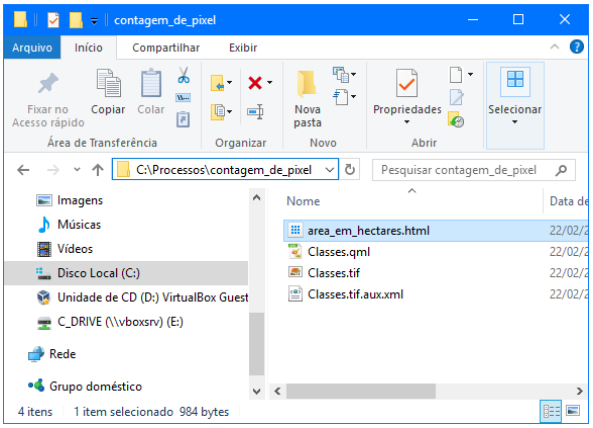
Chegou o momento de definir os parâmetros para execução do algoritmo. Marque o raster alvo do relatório. Nosso objetivo consiste em mapear a área de todas as classes com resultados em hectares. Selecione um dos itens da lista abaixo:

- **mi**: área em milhas quadradas
- **me**: área em metros quadrados
- **k**: área em quilômetros quadrados
- **a**: área em acres
- **h**: área em hectares
- **c**: número de pixels
- **p**: percentual de cobertura

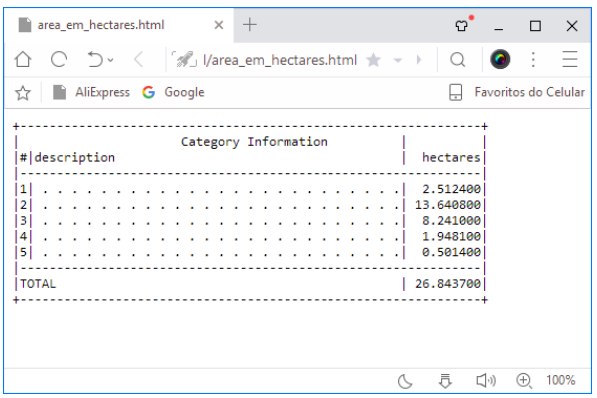
O algoritmo *r.report* gera um relatório no formato **HTML** que pode ser aberto no Navegador Web. Informe um local para o novo arquivo e pressione o botão **Run** para gerar o relatório.



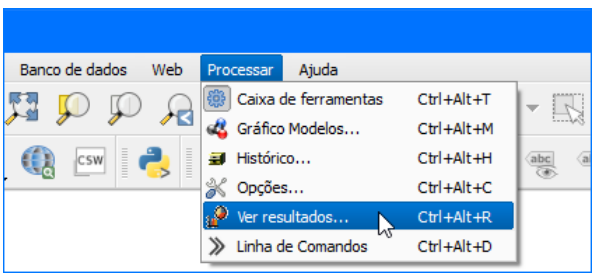
Após criar o arquivo, acesse o diretório e execute o arquivo:

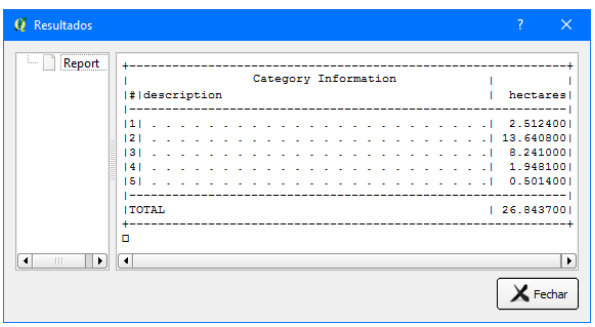


No navegador, copie as informações que são do seu interesse.

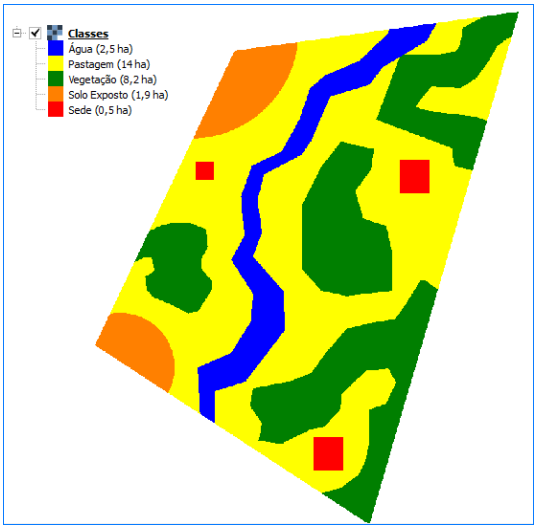


Para obter as informações em tela, clique no menu **Processar – Resultados**:





Este é um recurso muito importante para as atividades de classificação de imagens, embora não seja necessário classificar um raster para obter os resultados. Ao gerar o *layout*, faça a configuração da legenda para tornar o mapa mais intuitivo e informativo.



Créditos por este tutorial ao site Processamento Digital, especializado em materiais do campo de geoprocessamento de imagens.

O tutorial pode ser encontrado no link http://www.processamentodigital.com.br/2017/02/22/qgis-calculo-de-area-ou-contagem-de-pixels-para-uma-imagem-classificada/

Finalizado a vista do tutorial, podemos concluir que para um usuário inexperiente, seria provavelmente difícil de ter êxito levando em consideração o tempo, então entramos com a prática para este trabalho.

Outro trabalho interessante, encontrado e utilizado como comparativo para a situação, está disponível apenas no Youtube no https://youtu.be/weE0IjpuIL8, com mesmo objetivo, porém com outros meios para o resultado. Vale ressaltar que este é um exemplo real para comparativo de desempenho, muito mais complexo que o anterior.

4 Experimentos Computacionais

IEEE Tendo em vista que o método atual tem complexidade alta, podemos dizer que qualquer usuário pode obter um resultado satisfatório com este trabalho, definido como principal ponto na aplicação à simplicidade.

Começamos com a implementação, em que o código foi construído em Python, utilizando as bibliotecas OpenCV, Scikit-learn, Tkinter, Matplotlib e PIL que são descritas abaixo:

OpenCV (Open Source Computer Vision Library). Originalmente, desenvolvida pela Intel, em 2000, é uma biblioteca multiplataforma, totalmente livre ao uso acadêmico e comercial, para o desenvolvimento de aplicativos na área de Visão computacional, bastando seguir o modelo de licença BSD Intel. O OpenCV possui módulos de Processamento de Imagens e Vídeo I/O, Estrutura de dados, Álgebra Linear, GUI (Interface Gráfica do Usuário) básica com sistema de janelas independentes, Controle de mouse e teclado, além de mais de 350 algoritmos de Visão computacional como: Filtros de imagem, calibração de câmera, reconhecimento de objetos, análise estrutural e outros. Esta biblioteca foi desenvolvida nas linguagens de programação C/C++. Também, dá suporte a programadores que utilizem Java, Python e Visual Basic e desejam incorporar a biblioteca a seus aplicativos.

A Scikit-learn é uma biblioteca de aprendizado de máquina de código aberto para a linguagem de programação Python. Ela inclui vários algoritmos de classificação, regressão e agrupamento incluindo máquinas de vetores de suporte, florestas aleatórias, gradiente boosting, Kmeans e DBSCAN, e é projetada para interagir com as bibliotecas Python numéricas e científicas NumPy e SciPy.

Tkinter é uma biblioteca da linguagem Python que acompanha a instalação padrão e permite desenvolver interfaces gráficas. Isso significa que qualquer computador que tenha o interpretador Python instalado é capaz de criar interfaces gráficas usando o Tkinter.

Matplotlib é uma biblioteca de plotagem para a linguagem de programação Python e sua extensão de matemática numérica NumPy. Ele fornece uma API orientada a objetos para incorporar gráficos em aplicativos usando kits de ferramentas GUI de uso geral, como Tkinter, wxPython, Qt ou GTK +.

O NumPy é uma biblioteca para a linguagem de programação Python, que adiciona suporte a matrizes e matrizes grandes e multidimensionais, além de uma grande coleção de funções matemáticas de alto nível para operar nesses arrays.

Python Imaging Library (PIL) é uma biblioteca da linguagem de programação Python que adiciona suporte à abertura e gravação de muitos formatos de imagem diferentes.

4.1 O Código

Código feito em Python com as importações das bibliotecas já citadas, então cria-se uma janela utilizando recursos do Tkinter, após selecionamos o tamanho da janela, para ser definida, logo apresenta-se a janela de seleção do arquivo de entrada, para escolher a imagem, então é definida a imagem escolhida dentro da janela, apresentamos os botões de seleção da quantidade de clusters e o "Run!!!" que executa o algoritmo, além do caminho do arquivo (veremos na descrição do experimento, imagens com a representação).

O próximo passo é clicar no botão para iniciar o procedimento, quando clicado faz a chamada da função "clicked", lá encontra-se as definições de cada elemento do código, começando pela quantidade de clusters, seguindo do carregamento da imagem através da biblioteca OpenCv e a conversão para RGB (evita diferenças nas cores finais), então plota-se a imagem original, converte a imagem em array de pixels, e então executa-se a função Kmeans passando os parâmetros de quantidade de clusters, e o parâmetro "n_jobs", que representa a quantidade de threads que devem ser utilizadas, por opção 1 thread para cada cluster (otimização do processo) a implementação da função Kmeans é extensa e pode ser vista no diretório, caso esteja com a biblioteca Scikit-learn instalada ("Path-python"\Lib\site-packages\sklearn\cluster), mas em resumo segue o algoritmo:

- Defina o número de testes para semente local, se

nenhum for dado
- Escolha primeiro centro aleatoriamente
- Inicialize a lista das distâncias mais próximas e calcule o potencial atual
- Escolha os restantes n_clusters-1 pontos
    - Escolha candidatos do centro por amostragem com probabilidade proporcional à distância ao quadrado para o centro existente mais próximo
    - Calcular distâncias para centrar candidatos
    - Decida qual candidato é o melhor
        - Calcular potencial quando incluir o candidato ao centro
        - Armazene o resultado se for o melhor teste local até agora
    - Adicionar permanentemente o melhor candidato ao centro encontrado em tentativas locais
- Retorna os centros.

Após a execução do algoritmo o programa apresenta a imagem original e o resultado em outra janela com a porcentagem em histograma, junto com o resultado é salvo no disco o histograma em imagem. Então para finalizar, basta fechar as janelas.

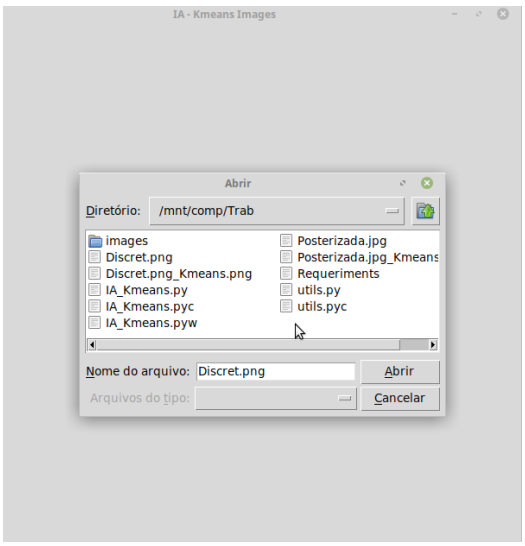
Assim, que temos o resultado podemos ver a eficiência, dependendo da quantidade de clusters, e outros fatores, como podemos ver nos exemplos:

6.3 Descrição do Experimento

Após a apresentação do funcionamento do algoritmo, podemos vê-lo em pleno funcionamento, com testes simples, e ajustes para melhor uso.
Abrindo o Terminal, entramos com:

➢ python IA_Kmeans.py

Após a execução, abrirá a seguinte tela, que serve para selecionar a imagem.



Após selecionar uma imagem, podemos alterar a quantidade de clusters, por padrão já está selecionado 5, após selecionar a quantidade de clusters, clicamos no botão "Run!!!"

No Terminal irá aparecer o tempo gasto para a execução, e em seguida as imagens serão exibidas na tela:

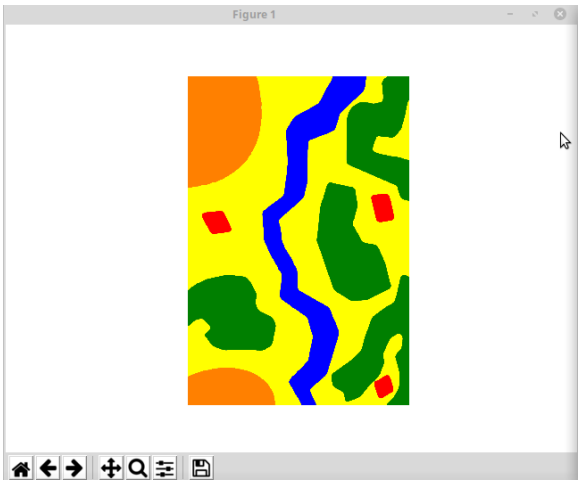
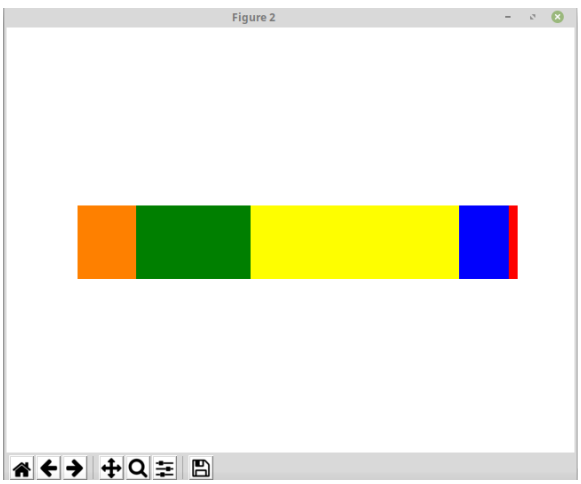
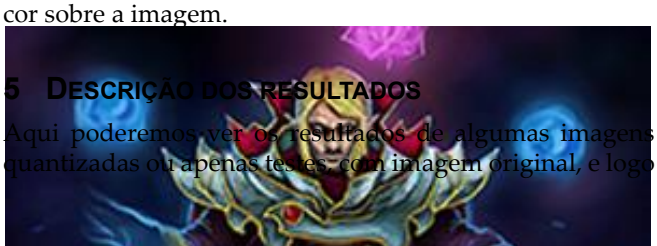


Imagem original, acima.



Histograma com a porcentagem estimada de área de cada cor sobre a imagem.

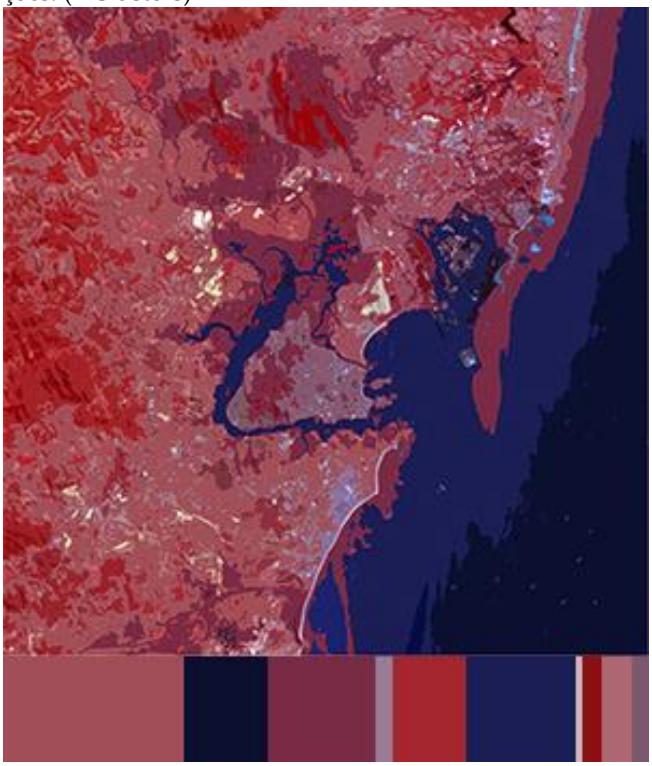


5 Descrição dos resultados

Aqui poderemos ver os resultados de algumas imagens quantizadas ou apenas testes com imagem original, e logo

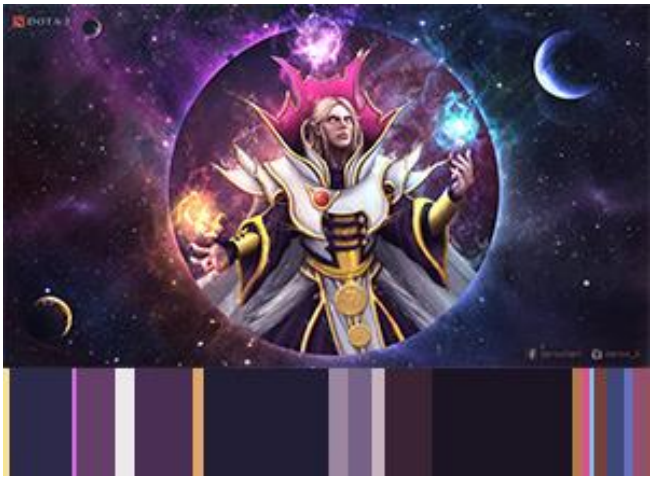
abaixo o histograma de resultado:
Cada imagem com a quantidade de Clusters.

Um teste simples, no qual prova a quantização na imagem, conhcecendo a quantidade de cores e suas devidas proporções. (4 Clusters)



(10 Clusters)

(10 Clusters)



(20 Clusters)

(20 Clusters)

REFERENCES

[1] K. Alsabti, S. Ranka, and V. Singh. An Efficient K-Means Clustering Algorithm, 1997.

[2] C. Atkeson, A. Moore, and S. Schaal. Locally weighted learning. AI Review, 1996.
[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. Introduction to Algorithms. McGraw-Hill Book Company, 1990.
[4] R. C. Dubes and A. K. Jain. Algorithms for Clustering Data. Prentice Hall, 1988.
[5] M. Ester, H. Kriegel, J. Sander, and X. Xu. A Density- Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Proc. of the 2nd Int'l Conf. on Knowledge Discovery and Data Mining, August 1996.
[6] M. Ester, H. Kriegel, and X. Xu. Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification. Proc. of the Fourth Int'l. Symposium on Large Spatial Databases, 1995.
[7] B. Scholkopf and A. Smola. Learning with Kernels. In Press, 2001.
[8] D. Judd, P. McKinley, and A. Jain. Large-Scale Parallel Data Clustering. Proc. Int'l Conference on Pattern Recognition, August 1996.
[9] Dhillon, Inderjit S., Guan, Yuqiang, Kulis, Brian, 2004. Kernel k-means: Spectral clustering and normalized cuts. In: Proc. 10th KDD, pp. 551–556.
[10] Andrews, Nicholas, O., Fox, Edward A., 2007. Recent developments in document clustering. Technical report TR-07-35. Deparment of Computer Science, Virginia Tech.
[11] Backstrom, L., Huttenlocher, D., Kleinberg, J., Lan, X., 2006. Group formation in large social networks: Membership, growth, and evolution. In: Proc. 12th KDD.
[12] Baldi, P., Hatfield, G., 2002. DNA Microarrays and Gene Expression. Cambridge University Press.
[13] Banerjee, Arindam, Merugu, Srujana, Dhillon, Inderjit, Ghosh, Joydeep. 2004. Clustering with bregman divergences. J. Machine Learn. Res., 234–245.
[14] Basu, Sugato, Banerjee, Arindam, Mooney, Raymond, 2002. Semi-supervised clustering by seeding. In: Proc. 19th Internat. Conf. on Machine Learning.
[15] Ben-David, S., Ackerman, M., 2008. Measures of clustering quality: A working set of axioms for clustering. Advances in Neural Information Processing Systems.
[16] Busse, Ludwig M., Orbanz, Peter, Buhmann, Joachim M., 2007. Cluster analysis of heterogeneous rank data. In: Proc. 24th Internat. Conf. on Machine Learning, pp. 113–120.
[17] Chapelle, O., Schölkopf, B., Zien, A. (Eds.), 2006. Semi-Supervised Learning. MIT Press, Cambridge, MA.
[18] Scikit-Learn Documentation, avaliable in http://scikit-learn.org/stable/_downloads/scikit-learn-docs.pdf, acess in 15/05/2018.

Bibliografia

André Thiago Borghi Couto, atualmente cursa graduação em Engenharia da Computação (2016/1) na Universidade Federal do Espírito Santo. Possui conhecimento em desenvolvimento de software, banco de dados, projetos em Arduino, design gráfico e tem interesse nos temas de automação em Arduino, inteligência artificial, redes de computadores e processamento de imagens.