Universidade Federal Rural de Pernambuco
Unidade Acadêmica de Garanhuns

Amanda Colatino
Gustavo Fernandes

Relatório de Projeto de Disciplina

Garanhuns-Pe
2019

Universidade Federal Rural de Pernambuco
Unidade Acadêmica de Garanhuns

Relatório de Projeto de Disciplina

Projeto desenvolvido para fins de avaliação
da disciplina de Reconhecimento de Padrões,
2° Verificação de Aprendizagem,
sob a orientação do Profº Luis Felipe

Garanhuns-Pe
2019

Descrição do Projeto

Image retrieval refere-se ao problema em que, a partir de uma imagem fornecida como entrada, imagens similares são encontradas em uma base de dados.

Nesse projeto, trabalharemos com uma base de dados de tomografias industriais com imagens de alta e baixa qualidade. As imagens de baixa qualidade foram geradas usando níveis reduzidos de radiação. A Figura 1 ilustra exemplos de tomografias de alta qualidade contidas na base de dados.

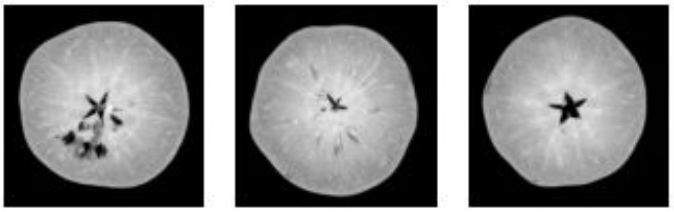

Figura 1: Exemplos de imagens contidas na base de dados do projeto.

Clusters com K-means

Utilizamos a biblioteca Scikit-Learn para a execução do algoritmo de cluster. Importando a biblioteca e o objeto KMeans.Para definir e criar os k clusters usamos os seguintes métodos:

```
kmeans = KMeans(n_clusters = k, init = 'random')
kmeans.fit(lista_imagens)
```

Onde lista_imagens são as imagens da base de dados em forma de array.

O parâmetro init se refere ao modo como o algoritmo será inicializado dando o valor random significa que o modo de inicialização será de forma aleatória, ou seja, os centróides iniciais serão gerados de forma totalmente aleatória sem um critério para seleção.

Para calcular o k ideal usamos o método Elbow. Basicamente o que o método faz é testar a variância dos dados em relação ao número de clusters. É considerado um valor ideal de k quando o aumento no número de clusters não representa um valor significativo de ganho.

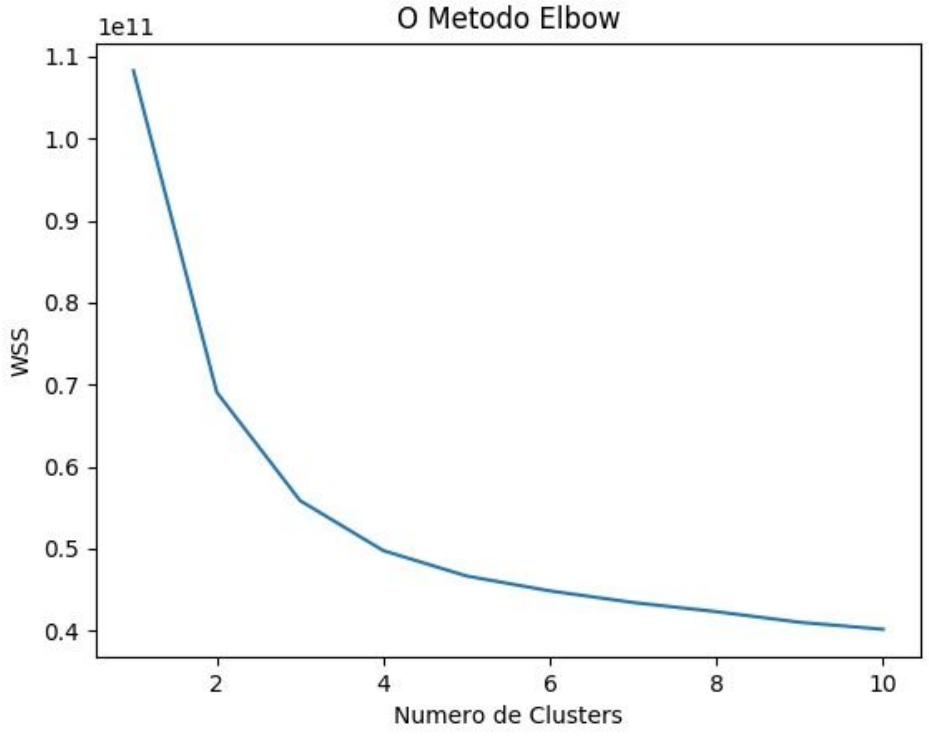

Figura 2: Gráfico do Método de Elbow

Conforme a figura 2, plotamos o somatório da variância dos dados em relação ao número de clusters para conseguir verificar até que ponto com o aumento do número de clusters não existe ganho. Observamos que a partir do número de três clusters as distâncias dos erros quadráticos praticamente se estabiliza. Neste ponto que seria o “cotovelo”, ou seja, a partir desse ponto que não existe uma discrepância tão significativa em termos de variância.

Classe Util

A classe util possui as seguintes funções auxiliares:

- distancia_euclidiana
- distancia_hamming
- media_lista
- desvio_padrao
- ordenar_lista_maior
- print_lista
- get_k_proximos
- get_proximos_limiar

CNN - Rede Neural Convolucional

A class Net implementa a rede neural, que possui 9 camadas, dispostas na seguinte ordem:

1. Camada convolucional (filtro 4x4, 3 neurônios)
2. Camada Max Pooling (tamanho 2x2)
3. Camada convolucional (filtro 4x4, 7 neurônios)
4. Camada Max Pooling (tamanho 2x2)
5. Camada convolucional (filtro 4x4, 10 neurônios)
6. Camada Max Pooling (tamanho 2x2)
7. Camada totalmente conectada (200 neurônios)
8. Camada totalmente conectada (20 neurônios)
9. Camada totalmente conectada (3 neurônios)

As camadas de 1 a 6 formam uma CNN.

A classe Imagem é uma estrutura de dados para auxiliar a manipulação das imagens durante as fases de treino e teste.

A base de dados foi dividida em três conjuntos: treinamento, validação e teste, com proporções 80%, 10% e 10%, respectivamente, sendo que as duas versões de cada imagem (baixa qualidade e alta qualidade estão no mesmo conjunto (esta divisão foi implementada na função carregar_bases, no arquivo cnn.py).

A função criar_cnn criar (ou carrega de um arquivo) e a treina, durante o treinamento é armazenada a “melho_cnn”, definida como cnn com menor taxa de erros (no caso de ocorrer um over fitting) e após o treinamento é perguntado se deseja salvar esta rede.

Na função testar_cnn são concatenadas as bases de treino e validação em uma única base, e a base de teste é usado para o Image Retrieval, nesta função é implementado o método proposto por
Lin et al., usando as funções get_proximo_limiar e get_k_proximos do arquivo util.py, são mostradas as 6 imagens mais parecidas com a imagem de busca na base de dados (base de treino + base de validação).

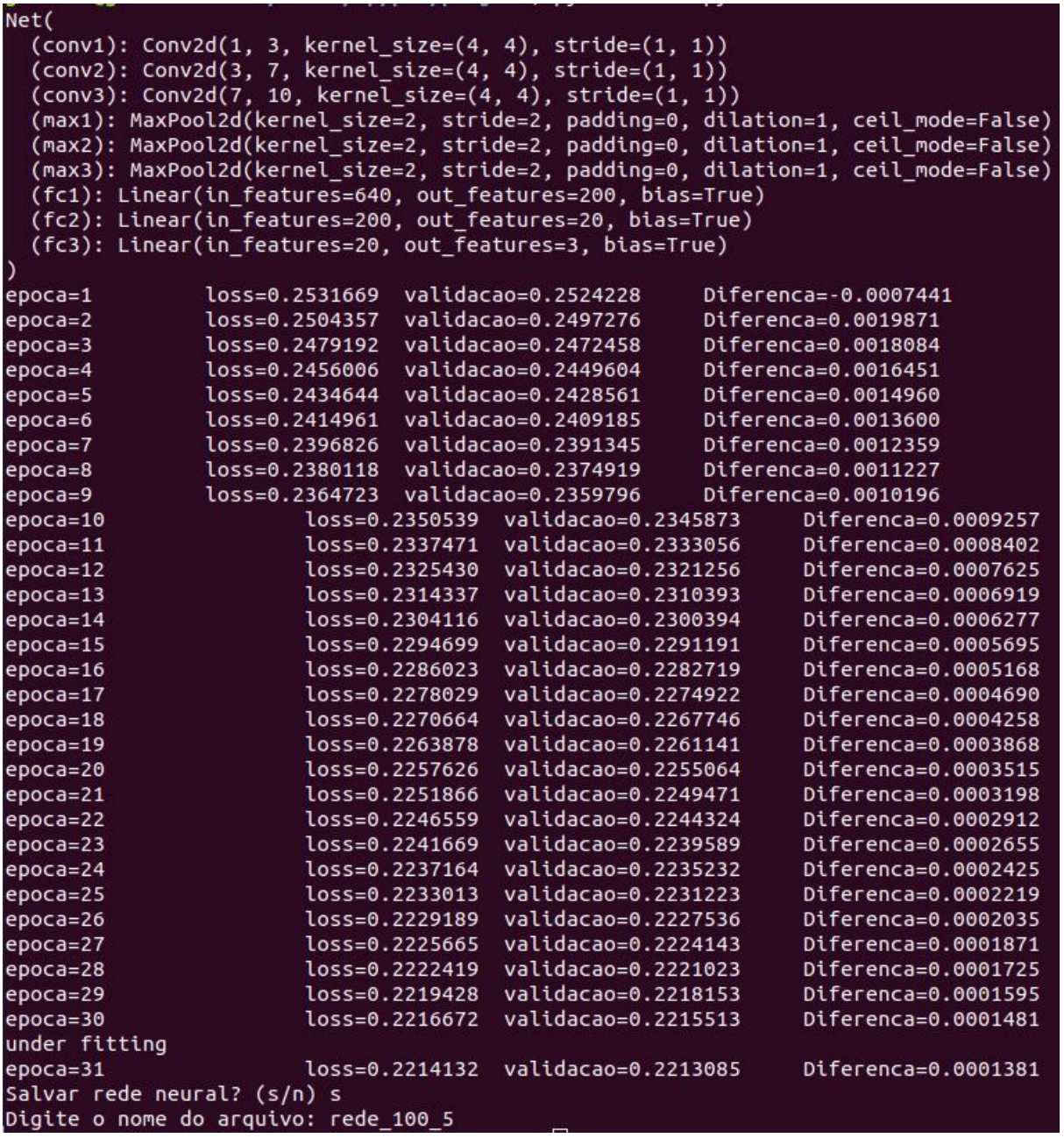

Figura 3: Resultado de uma rede neural

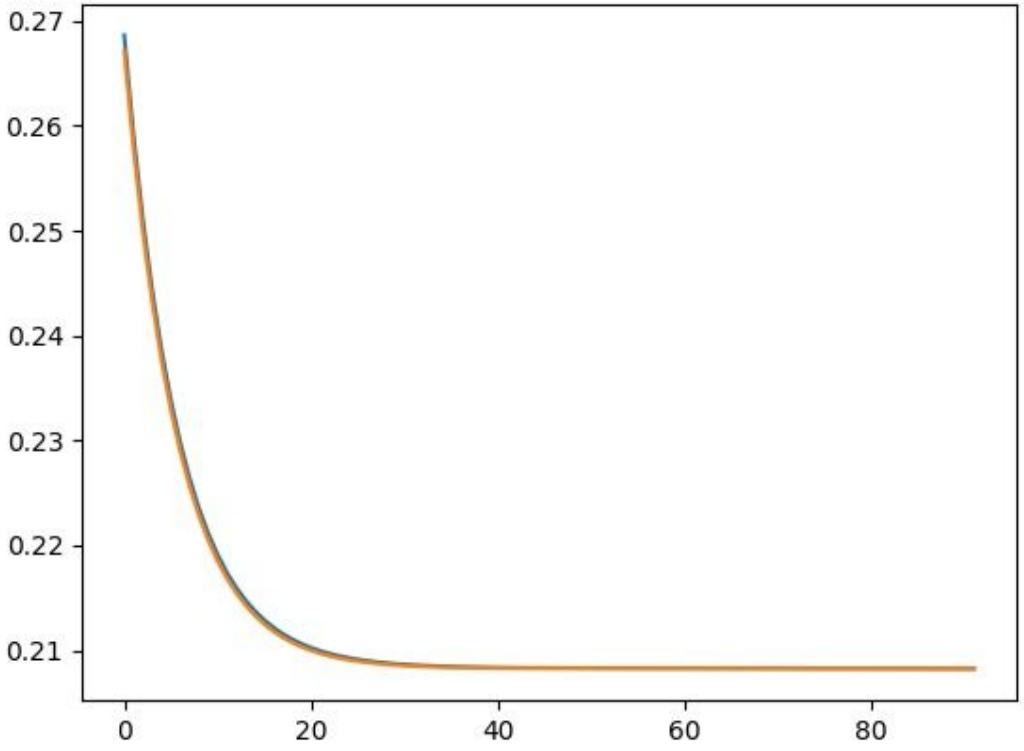

Figura 4: Curva de aprendizado. Treino e validação.

Resultados Obtidos

Segue abaixo resultados da rede para entradas de imagens de baixa qualidade e alta qualidade.

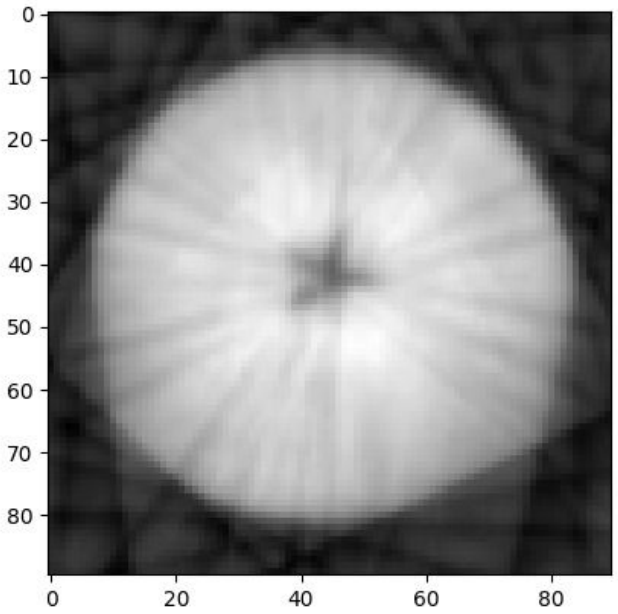

Figura 5: Imagem de baixa qualidade passada como busca.

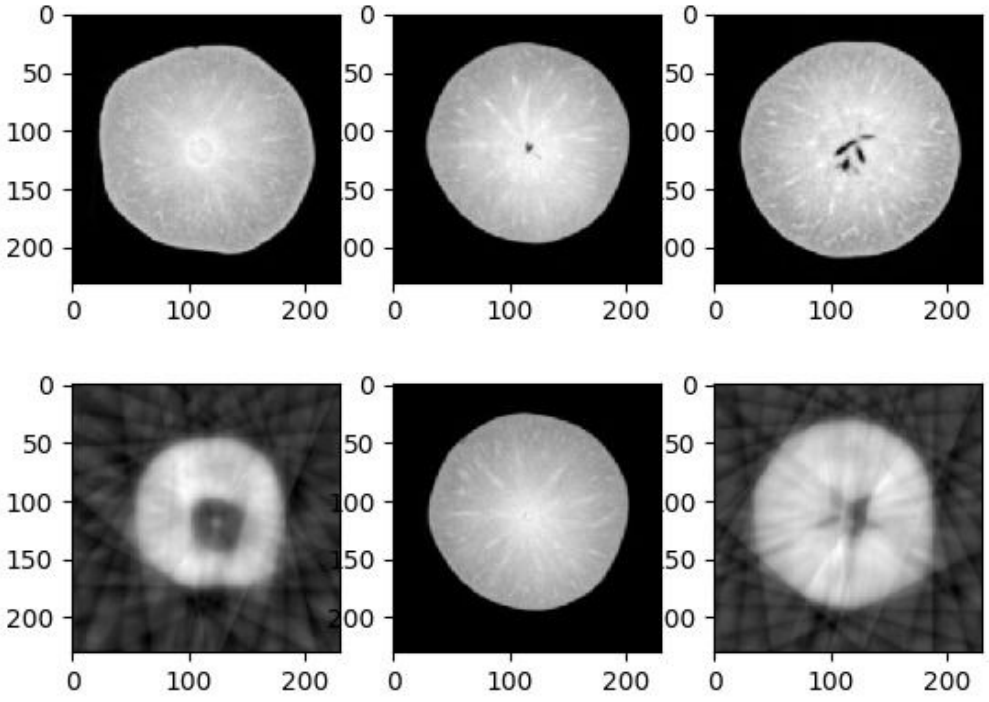

Figura 6: Resultados

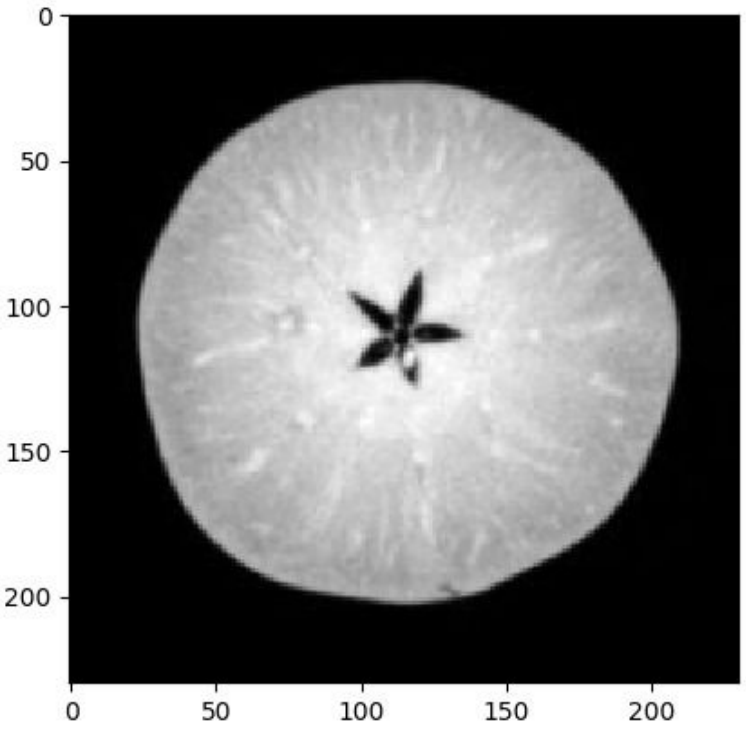

Figura 7: Imagem de alta qualidade passada como busca.

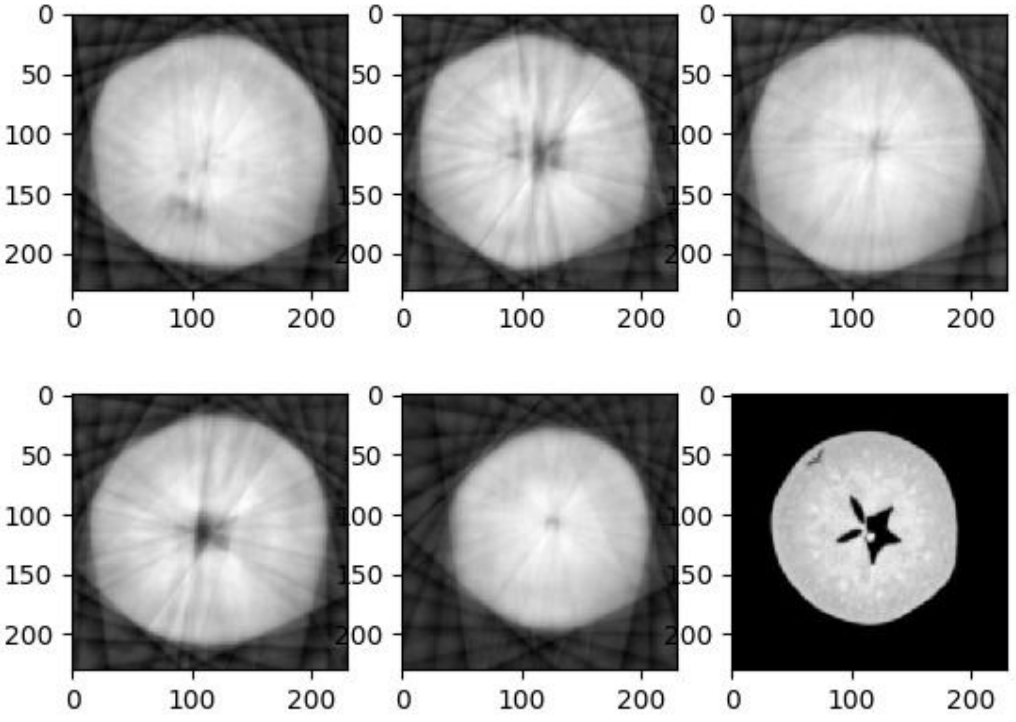

Figura 8: Resultados.