

Comparação de imagens por similaridade¹

Descrição

O programa utiliza redes neurais convolucionais e cálculos de similaridade (otimizados para execução em GPU) para comparar dois grupos de imagens com base no conteúdo.

A implementação foi realizada em Python 3, com bibliotecas Keras e Tensorflow, dentre outras.

Casos de Uso

- 1) O principal caso de uso nas perícias de informática ocorre quando o perito recebe um CD com arquivos que precisam ser pesquisados em um HD examinado. A abordagem tradicional é a de pesquisa por identidade (Hashs) ou pesquisa manual/visual.
- 2) Em um exame de HD relacionado a uma investigação de falsificação de moeda, é possível colocar diversas imagens de cédulas em um diretório e comparar com imagens de um HD questionado.
- 3) Em um exame de HD relacionado a uma investigação de falsificação de documentos, é possível colocar diversas imagens de documentos de segurança em um diretório e comparar com imagens de um HD questionado.

Objetivo

O objetivo do projeto é fazer a comparação automatizada com base nas características visuais da imagem, extraídas através de uma rede convolucional pre-treinada (arquitetura VGG-16² - mas pode ser alterada para qualquer outra).

Os passos para instalar e executar o programa são descritos a seguir.

1. Instalar o docker para CPU

Instruções para instalação do docker para CPU em <https://docs.docker.com/docker-for-windows/> ou <https://www.docker.com/docker-ubuntu>.

Se desejar utilizar uma GPU (sugiro, **no mínimo**, usar uma **GTX 1050 Ti** e, **preferencialmente**, uma **GTX 1080 Ti**), a versão do docker deve ser para GPU - consulte o passo 10.

2. Obter programa e container docker

Obter programa:

- fazer download em <https://github.com/joaomacedo/image-similarity> (neste tutorial, assumimos que os arquivos do programa foram colocadas no diretório **/home/jm/report_similaridade**)

¹ Esse projeto de busca de imagens por similaridade foi efetuado como atividade de pesquisa individual e não foi desenvolvido sob supervisão ou qualquer tipo de vínculo institucional.

² Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.2.

Para obter o container docker, na linha de comando digite:

```
docker pull ufoym/deepo:all-jupyter-py36-cpu
```

3. Iniciar container

Comando para iniciar ativando o notebook:

```
sudo docker run -it -p 8888:8888 -v /home/jm:/home_docker ufoym/deepo:all-jupyter-py36-cpu jupyter notebook --no-browser --ip=0.0.0.0 --allow-root --NotebookApp.token=--notebook-dir='/home_docker/report_similaridade'
```

O docker inicia no diretório '/home_docker/report_similaridade'.

O arquivo runcpu.sh contem esse comando. Sugiro alterar lá para ficar mais fácil (ou criar um arquivo .bat).

Observação:

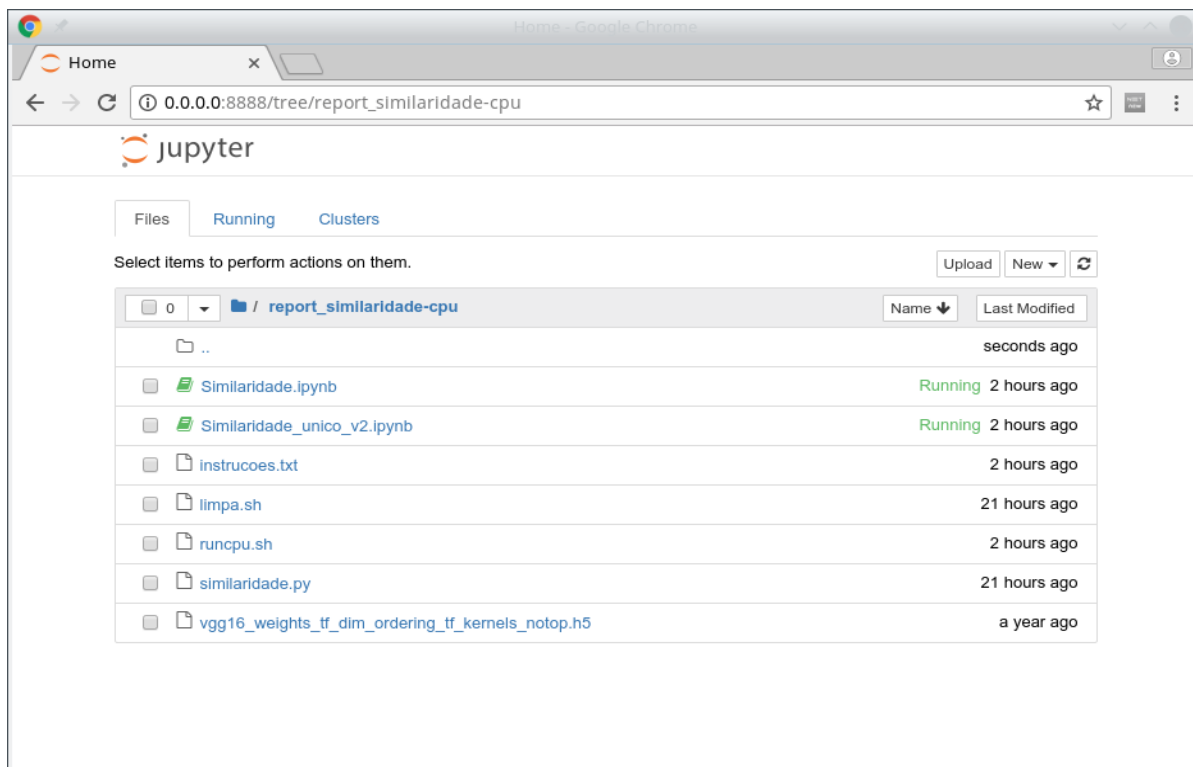
"/home/jm" pode ser substituído por qualquer diretório existente da sua máquina - não precisa necessariamente ser seu "home", basta que seja qualquer diretório e que você coloque a pasta report_similaridade nele.

Quando o docker iniciar, esse diretório pode ser acessado como /home_docker

os diretórios dos arquivos questionado e padrão têm que ser acessíveis a partir do diretório report_similaridade.

4. Abrir página do jupyter

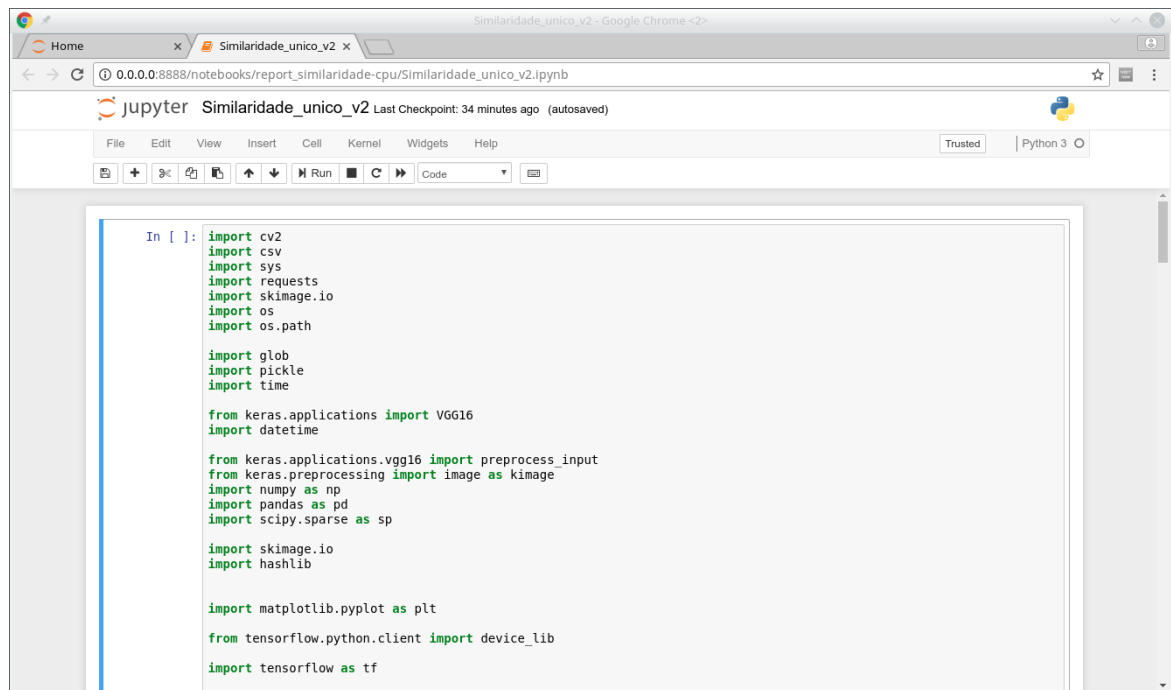
Abrir um navegador e digitar: <http://0.0.0.0:8888>



5. Selecionar um notebook

Existem 2 notebooks: Similaridade_unico_v2.ipynb e notebooks Similaridade.ipynb

Sugiro que inicie o **Similaridade_unico_v2.ipynb**



Esse notebook contém apenas uma célula com as partes principais do código para geração dos relatórios (o resto foi colocado no arquivo similaridade.py para simplificar).

O outro notebook (Similaridade.ipynb) tem várias células com todo o código. A funcionalidade é idêntica, mas permite uma execução passo a passo (útil para depurar ou ver as etapas do trabalho).

6. Ajustar caminho dos arquivos e "nome de exibição" dos conjuntos de dados

Ajuste os quatro valores a seguir:

```
arqs["busca"]["raiz"] = "./img_busca/"
arqs["busca"]["nome_exibicao"] = "CD Anexo"
```

```
arqs["material"]["raiz"] = "./img_quest/"
arqs["material"]["nome_exibicao"] = "HD Examinado"
```

7. Ajuste de parâmetros de busca e geração de relatórios

Por fim, você também pode alterar os parâmetros das buscas por similaridade dos relatórios gerados:

```
gera_html_busca(dados, arqs, excl, N=3, minimoinicial=0.20, minimofinal=0.15)
gera_html_material(dados, arqs, excl, N=3, minimoinicial=0.20, minimofinal=0.15)
```

N=3 - para cada arquivo analisado (no CD, por exemplo), reporta até 3 semelhantes (no HD), de acordo com os índices de similaridade minimoinicial e minimofinal.

minimoinicial=0.20 - o arquivo mais semelhante tem que ter similaridade de no mínimo 0.20. Você pode pensar esse número como 20% de similaridade. Dessa forma, ao analisar um arquivo do CD, se o arquivo com maior similaridade do HD tiver índice maior ou igual a 0.2 ele será selecionado, caso contrário não será reportado nenhum semelhante.

minimofinal=0.15 - se foi encontrado pelo menos um semelhante com índice maior que o mínimo inicial, continua selecionando os semelhantes em índice de semelhança decrescente, desde que seja maior que o mínimo final e que respeite o número de semelhanças desejado ("N=3", no caso).

O relatório gerado vai “crescer” ou “diminuir”, de acordo com esses 3 parâmetros. Se os 2 últimos forem muito altos, pode não encontrar nenhuma semelhança.

8. Executar

Clique em um ponto do código e digite CTRL+Enter

São gerados dois relatórios.

- ◆ **gera_html_busca:** para cada arquivo do diretório arqs["busca"]["raiz"], procura semelhantes no diretório arqs["material"]["raiz"], conforme parâmetros especificados.
- ◆ **gera_html_material:** para cada arquivo do diretório arqs["material"]["raiz"], procura semelhantes no diretório arqs["busca"]["raiz"], conforme parâmetros especificados.

Os relatórios serão gerados na pasta /home_docker/report_similaridade/relatorios

9. Se não ficou satisfeito e quer reajustar os parâmetros dos relatórios:

Todos os dados de predições, cálculos de similaridade, etc, são guardados em arquivos na pasta do notebook.

Para não ter que refazer toda a leitura de arquivos e cálculos, basta alterar as seguintes variáveis para False (inicialmente estão True):

```
gera_arqs = False  
gera_dados = False
```

(não esqueça de retornar para True em um novo projeto ou caso os dados sejam alterados)

Volte ao passo 7.

Utilização com GPU

A utilização com GPU pode acelerar a execução em 10x ou mais, utilizando uma placa Nvidia GTX 1080-Ti, por exemplo. Com exceção dos passos 1 ao 3, os demais são idênticos.

Para usar GPU:

- ◆ No passo 1 tem que instalar o docker-nvidia (e antes disso pode ser necessário instalar os drives CUDA 9.0 e CUDNN 7.0 da nvidia) em vez do docker.
- ◆ Nos passos 2 e 3: usar o comando **docker-nvidia** em vez de docker e a imagem **ufoym/deepo:all-jupyter-py36** em vez de ufoym/deepo:all-jupyter-py36-cpu.