DCC
DEPARTAMENTO DE
CIÊNCIA DA COMPUTAÇÃO

UFMG

RECFACE - Desenvolvimento de módulo de reconhecimento facial por imagens

Manual de Uso
9 de abril de 2021

DADOS DO PROJETO

Título do Projeto:	RECFACE - Desenvolvimento de módulo de reconhecimento facial por imagens
Áreas de Interesse:	Processamento de Imagens; Sensoriamento Remoto; Aprendizado de Máquina
Instituição Executora:	Laboratório de Reconhecimento de Padrões e Observação da Terra (PATREO) Departamento de Ciência da Computação Instituto de Ciências Exatas Universidade Federal de Minas Gerais Av. Antônio Carlos, 6627, Pampulha Belo Horizonte - MG, 31270-010
Coordenador:	Jefersson Alex dos Santos Webpage: http://www.dcc.ufmg.br/~jefersson E-mail: jefersson@dcc.ufmg.br Tel.: (31) 3409-1469 Fax: (31) 3409-5858

Sumário

Capítulo 1

Introdução

Esse projeto visa desenvolver um módulo para detecção e reconhecimento facial a partir de imagens. O módulo desenvolvido deve receber uma imagem de entrada, detectar a face (e, idealmente, seus pontos fiduciais), e extrair as características visuais (usando técnicas de *Deep Learning* – aprendizado profundo). Posteriormente, esse módulo deve utilizar tais características para gerar um ranking de similaridade entre a face detectada na imagem de entrada e outras faces de um banco de dados.

Este documento apresenta um manual de uso deste módulo desenvolvido com objetivo de auxiliar usuários a entender como usá-lo. Precisamente, o manual inclui:

- requerimento mínimos (em termos de software) para executar o módulo,
- execução do módulo, onde cada parâmetro de entrada é explicado genericamente, e
- casos de uso, onde são explicados, genericamente e com exemplos, os possíveis casos de uso do módulo.

Capítulo 2

Manual de uso

2.1 Requerimentos mínimos

Os requerimentos mínimos, em termos de software, para execução do módulo são:

- Python 3.x
- PyTorch
- Scipy
- ImageIO
- SKLearn
- OpenCV Python
- dlib
- requests
- py7zr
- gzip

2.2 Execução

O módulo desenvolvido tem diversas opções de execução, entre eles o processamento de uma imagem específica ou de um dataset, o ranqueamento para uma imagem específica ou para um dataset, a geração de resultados visuais para um dataset, etc. Todas essas operações podem ser inicializadas através do arquivo **main.py**, cuja execução segue o padrão abaixo:

```
python main.py [--operation OPERATION]
               [--feature_file FEATURE_FILE]
               [--result_sample_path RESULT_SAMPLE_PATH]
               [--image_query IMAGE_QUERY]
               [--query_label QUERY_LABEL]
               [--dataset DATASET_NAME]
               [--specific_dataset_folder_name DATASET_FOLDER_NAME]
               [--img_extension EXTENSION]
               [--preprocessing_method PRE_PROCESS_NAME]
               [--model_name MODEL_NAME]
               [--batch_size BATCH_SIZE]
```

onde,

- `--operation [OPERATION]`

  representa a operação que será realizada pelo algoritmo.

  - Este é um parâmetro **requerido**.
  - As possíveis opções são:
    * **extract_features** para extrair features de uma imagem ou dataset,
    * **generate_rank** para gerar o ranking de uma imagem ou dataset, e
    * **extract_generate_rank** para extrair features e avaliar o dataset todo [opção válida somente para processamento de datasets].

- `--feature_file [FEATURE_FILE]`

  representa o nome do arquivo (com caminho e extensão) onde as features serão salvas ou carregadas.

  - Este é um parâmetro **opcional**.
  - Por padrão, este parâmetro é definido como “None”.
  - A extensão do arquivo requerido deve ser “.mat”.
  - Um exemplo válidos seria: result_features/features.mat

- `--result_sample_path [RESULT_SAMPLE_PATH]`

  representa um caminho para salvar as imagens com os exemplos visuais dos resultados [opção válida somente para processamento de datasets].

  - Este é um parâmetro **opcional**.
  - Por padrão, este parâmetro é definido como “None”.
  - Se um caminho válido for passado, ao processar um dataset, imagens de exemplo serão salvas nesse caminho.

- `--image_query [IMAGE_QUERY]`

  representa o caminho ou a string para a imagem de entrada (*query*).

  - Este é um parâmetro **opcional**.
  - Por padrão, este parâmetro é definido como “None”.

- `--query_label [QUERY_LABEL]`

  representa a label ou ID da imagem de entrada (*query*).

  - Este é um parâmetro **opcional**.

  - Por padrão, este parâmetro é definido como “None”.
  - Passa a ser **requerido** caso se deseja adicionar uma nova imagem ao database, como mostrado no 2.3.1.

- `--dataset [DATASET_NAME]`

  representa o nome do dataset que terá as características extraídas.

  - Este é um parâmetro **opcional**.
  - Por padrão, este parâmetro é definido como “None”.
  - As possíveis opções são:
    * LFW
    * YALEB

- `--specific_dataset_folder_name [DATASET_FOLDER_NAME]`

  representa o caminho para a pasta específica do dataset.

  - Este é um parâmetro **opcional**.
  - Por padrão, este parâmetro é definido como “None”.
  - Exemplos de parâmetros a serem passados: lfw ou ExtendedYaleB.
  - Se a flag –dataset for usada, esse parâmetro passa a ser **requerido**.

- `--img_extension [EXTENSION]`

  representa a extensão das imagens do dataset a ser processado.

  - Este é um parâmetro **opcional**.
  - Por padrão, este parâmetro é definido como “jpg”.
  - As opções comuns incluem:
    * jpg (para LFW)
    * pgm (para YALEB)

- `--preprocessing_method [PRE_PROCESS_NAME]`

  representa método de pré-processamento a ser utilizado nas imagens.

  - Este é um parâmetro **opcional**.
  - Por padrão, este parâmetro é definido como “None”.
  - As possíveis opções são:
    * MTCNN
    * SphereFace

        * OpenFace
        * None (nessa opção, é feito um crop no centro da imagem por padrão)
    - Importante notar que, se algum método de pré-processamento falhar, o método None é executado por padrão.

- `--model_name [MODEL_NAME]`

  representa o nome do modelo que será usado na extração de características.

    - Este é um parâmetro **requerido**.
    - As possíveis opções são:
        * MobileFaceNet
        * MobiFace
        * SphereFace
        * OpenFace
        * FaceNet
        * ShuffleFaceNet

- `--batch_size [BATCH_SIZE]`

  representa o tamanho do *batch* usado para processar as imagens.

    - Este é um parâmetro **opcional**.
    - Por padrão, este parâmetro é definido como 32.
    - Para dataset YALEB, por motivos de memória de GPU, o padrão é 8.

2.3 Casos de uso

Como introduzido, neste seção explicaremos e exemplificaremos cada possível caso de uso do sistema. Cada caso de uso possui parâmetros e finalidade particulares, cujas especificidades serão minuciosamente detalhadas a seguir.

2.3.1 Extrair características de uma imagem específica

Objetivo: Gerar vetor de características de uma imagem e salvá-lo, acompanhado de um ID, em um arquivo de características (“.mat”) para uso futuro.

Execução:

```
python3 main.py --operation extract_features \
                --feature_file [FEATURE_FILE_PATH] \
                --image_query [IMAGE_QUERY] \
                --query_label [ID] \
                --model_name [MODEL_NAME] \
                --preprocessing_method [PREPROCESSING_METHOD]
```

Parâmetros:

```
--operation extract_features
```

deve ser mantido obrigatoriamente como **extract_features**.

```
--feature_file [FEATURE_FILE_PATH]
```

deve conter obrigatoriamente o caminho para o arquivo em que será armazenado o vetor de características juntamente com o ID.

Note que, se o arquivo informado não existir, um novo será criado e conterá apenas o novo vetor e o ID correspondente. Caso o arquivo já exista, ou seja, possua vetores de características de outras imagens, o vetor recém-extraído será concatenado aos demais.

```
--image_query [IMAGE_QUERY]
```

recebe, obrigatoriamente, o caminho para a imagem cujas características serão extraídas e salvas.

```
--query_label [ID]
```

recebe o ID da imagem da qual se informou o caminho no parâmetro *image_query*.

```
--model_name [MODEL_NAME]
```

deve conter o nome da rede a ser utilizada para a extração das características. Opções disponíveis: MobileFaceNet, MobiFace, SphereFace, OpenFace, FaceNet, e ShuffleFaceNet.

```
--preprocessing_method [PREPROCESSING_METHOD]
```

define o tipo de pré-processamento à ser aplicada à imagem de entrada. Deverá conter obrigatoriamente uma das opções seguintes de pré-processamento: MTCNN, SphereFace, OpenFace, None.

Exemplo:

```
python3 main.py --operation extract_features \
                --feature_file root/features_.mat \
                --image_query root/datasets/LFW/lfw/Winona_Ryder/
                Winona_Ryder_0001.jpg \
                --query_label Winona_Ryder \
                --model_name mobilefacenet \
                --preprocessing_method sphereface
```

Neste caso, o algoritmo extrai as características da imagem *Winona_Ryder_0001.jpg* com o pré-processamento SphereFace e a rede MobileFaceNet. O vetor de features é salvo no arquivo "features.mat"com o ID informado "Winona_Ryder".

2.3.2 Criar ranking de similaridade de uma imagem específica

Objetivo: Gerar ranking com os 10 indivíduos mais similares ao indivíduo da imagem informada. Para tal são extraídas as características da imagem em questão que são, então, comparadas àquelas salvas no arquivo com extensão de características (“.mat”) informado. É Retornado uma lista de 10 tuplas, contendo o score e o ID dos 10 indivíduos mais similares ao indivíduo da imagem.

Execução:

```
python3 main.py --operation generate_rank \
                --feature_file [FEATURE_FILE_PATH] \
                --image_query [IMAGE_QUERY] \
                --model_name [MODEL_NAME] \
                --preprocessing_method [PREPROCESSING_METHOD]
```

Parâmetros:

```
--operation generate_rank
```

deve ser mantido obrigatoriamente como **generate_rank**.

```
--feature_file [FEATURE_FILE_PATH]
```

deve conter, obrigatoriamente, o caminho para o arquivo que possui os vetores de características que serão comparados para gerar o ranking de similaridade.

Vale ressaltar que, se o arquivo informado tiver menos que 10 indivíduos diferentes, então a lista do ranking de similaridade terá tamanho igual ao número de indivíduos presentes.

```
--image_query [IMAGE_QUERY]
```

recebe, obrigatoriamente, o caminho para a imagem cujas características serão extraídas e comparadas àquelas no arquivo informado.

```
--model_name [MODEL_NAME]
```

deve conter o nome da rede a ser utilizada para a extração das características. Opções disponíveis: MobileFaceNet, MobiFace, SphereFace, OpenFace, FaceNet, e ShuffleFaceNet.

```
--preprocessing_method [PREPROCESSING_METHOD]
```

define o tipo de pré-processamento à ser aplicada à imagem de entrada. Deverá conter obrigatoriamente uma das opções seguintes de pré-processamento: MTCNN, SphereFace, OpenFace, None.

Exemplo:

```
python3 main.py --operation generate_rank \
                --feature_file root/features.mat \
                --image_query root/datasets/LFW/lfw/John_Travolta/
                John_Travolta_0004.jpg \
                --model_name mobilefacenet \
                --preprocessing_method sphereface
```

Exemplo de saída da execução:

```
[('John_Travolta', 0.7627566), ('Nelson_Mandela', 0.15842089),
('Tiger_Woods', 0.038725413), ('Will_Smith', 0.018085621),
('Gray_Davis', -0.006768973), ('Jennifer_Aniston', -0.011421391),
('Winona_Ryder', -0.08045954), ('JK_Rowling', -0.09918424),
('Angelina_Jolie', -0.13311145), ('Zico', -0.18437612)]
```

Neste caso, o algoritmo extrai as características da imagem *John_Travolta_0004.jpg* com o pré-processamento SphereFace e a rede MobileFaceNet. Em seguida gera pontuação de similaridade entre o vetor de características recém-criado e cada um dos outros presentes no arquivo “features.mat”. Ao final, retorna lista de 10 tuplas, contendo o score e o ID dos 10 indivíduos mais similares ao John Travolta.

2.3.3 Extrair características de imagens de um dataset específico

Objetivo: Gerar vetor de características para cada uma das imagens de um dataset informado e salvá-los, acompanhados de seus respectivos ID’s, em um arquivo de extensão ".mat"para uso futuro.

Execução:

```
python3 main.py --operation extract_features \
                --feature_file [FILE_PATH] \
                --dataset [DATASET_NAME] \
```

```
                --specific_dataset_folder_name [DATASET_FOLDER_NAME] \
                --model_name [MODEL_NAME] \
                --preprocessing_method [PREPROCESSING_METHOD]
```

Note que, para que essa rotina funcione corretamente, é necessário que os parâmetros *dataset* e *specific_dataset_folder_name* sejam informados. Além disso é importante observar que este caso apenas funciona para datasets pré-selecionados, isto é, para o LFW e o YALEB.

Parâmetros:

```
--operation extract_features
```

deve ser mantido obrigatoriamente como **extract_features**.

```
--feature_file [FEATURE_FILE_PATH]
```

deve conter, obrigatoriamente, o caminho para o arquivo de características (".mat"), onde as *features* extraídas serão salvas.

```
--dataset [DATASET_NAME]
```

deve conter, obrigatoriamente, o nome do dataset pré-selecionado. Opções válidas são: LFW e YALEB.

```
--specific_dataset_folder_name [DATASET_FOLDER_NAME]
```

deve conter obrigatoriamente o nome da pasta com a variação do dataset.

```
 --model_name [MODEL_NAME]
```

deve conter o nome da rede a ser utilizada para a extração das características. Opções disponíveis: MobileFaceNet, MobiFace, SphereFace, OpenFace, FaceNet, e ShuffleFaceNet.

```
 --preprocessing_method [PREPROCESSING_METHOD]
```

define o tipo de pré-processamento à ser aplicada à imagem de entrada. Deverá conter obrigatoriamente uma das opções seguintes de pré-processamento: MTCNN, SphereFace, OpenFace, None.

Exemplo:

```
python3 main.py --operation extract_features \
                --feature_file root/features.mat \
                --dataset LFW \
                --specific_dataset_folder_name lfw \
                --model_name mobilefacenet \
                --preprocessing_method sphereface
```

Neste caso, o algoritmo extrai as características das imagens do dataset LFW. Por meio de um *dataloader* previamente definido o algoritmo itera pelas imagens salvando os vetores em um arquivo de características (".mat"), que precisa obrigatoriamente ser declarado no parâmetro *feature_file*.

2.3.4 Avaliar todo o dataset

Objetivo: Importar as características de um dataset contidas no arquivo de características (".mat") informado e fazer a avaliação das métricas em todo o dataset, reportando o mAP (mean average precision) e o AP (average precision) dos tops 1, 5, 10, 20, 50 e 100.

Execução:

```
python3 main.py --operation generate_rank \
                --feature_file [FEATURE_FILE] \
                --dataset [DATASET_NAME] \
                --specific_dataset_folder_name [DATASET_FOLDER_NAME] \
                --model_name [MODEL_NAME] \
                --preprocessing_method [PREPROCESSING_METHOD]
```

Observação: Para essa rotina é necessário que os parâmetros *features_file*, *dataset* e *specific_dataset_folder_name* sejam declarados.

Parâmetros:

```
--operation generate_rank
```

deve ser mantido obrigatoriamente como **generate_rank**.

```
--feature_file [FEATURE_FILE_PATH]
```

deve conter, obrigatoriamente, o caminho para o arquivo que possui os vetores de características que serão comparados para gerar o ranking de similaridade.

```
--dataset [DATASET_NAME]
```

deve conter, obrigatoriamente, o nome do dataset pré-selecionado. Opções válidas são: LFW e YALEB.

```
--specific_dataset_folder_name [DATASET_FOLDER_NAME]
```

deve conter obrigatoriamente o nome da pasta com a variação do dataset.

```
 --model_name [MODEL_NAME]
```

deve conter o nome da rede a ser utilizada para a extração das características. Opções disponíveis: MobileFaceNet, MobiFace, SphereFace, OpenFace, FaceNet, e ShuffleFaceNet.

```
 --preprocessing_method [PREPROCESSING_METHOD]
```

define o tipo de pré-processamento à ser aplicada à imagem de entrada. Deverá conter obrigatoriamente uma das opções seguintes de pré-processamento: MTCNN, SphereFace, OpenFace, None.

Exemplo:

```
python3 main.py --operation generate_rank \
                --feature_file root/features.mat \
                --dataset LFW \
                --specific_dataset_folder_name lfw \
                --model_name mobilefacenet \
                --preprocessing_method sphereface
```

Exemplos de saída da execução:

```
mAP: 0.915428 top1: 0.967251 top5: 0.964831 top10: 0.963886 top20: 0.956998
top50: 0.926217 top100: 0.920380
Total execution time: 21.914638 seconds. Execution time per query: 0.007249 seconds.
```

Neste caso, o algoritmo importa as características do dataset LFW armazenadas no arquivo "features.mat". Utilizando distância de cossenos, o sistema gera o ranking de similaridade e faz a avaliação com todos os vetores. Em seguida são calculadas as métricas que serão reportadas.

2.3.5 Extrair características das imagens de um dataset específico e avaliá-lo

Objetivo: Gerar um vetor de características para cada uma das imagens de um dataset informado. Em seguida, o avalia calculando a mAP (média da AP de todas as queries). Neste caso, nenhum arquivo contendo os vetores de características é salvo.

Execução:

```
python3 main.py --operation extract_generate_rank \
                --result_sample_path [FILE_PATH] \
                --dataset [DATASET_NAME] \
                --specific_dataset_folder_name [DATASET_FOLDER_NAME] \
                --img_extension [EXTENSION] \
                --model_name [MODEL_NAME] \
                --preprocessing_method [PREPROCESSING_METHOD]
```

Parâmetros:

```
--operation extract_generate_rank
```

deve ser mantido obrigatoriamente como **extract_generate_rank**.

```
--result_sample_path [RESULT_SAMPLE_PATH]
```

deve conter o caminho onde serão salvos os resultados visuais.

Note que, o parâmetro result_sample_path é opcional, e só deve ser utilizado caso se deseje salvar resultados visuais.

```
--dataset [DATASET_NAME]
```

deve conter, obrigatoriamente, o nome do dataset pré-selecionado. Opções válidas são: LFW e YALEB.

```
--specific_dataset_folder_name [DATASET_FOLDER_NAME]
```

deve conter obrigatoriamente o nome da pasta com a variação do dataset.

```
--img_extension [EXTENSION]
```

deve conter, obrigatoriamente, o tipo de extensão usado nas imagens.

```
 --model_name [MODEL_NAME]
```

deve conter o nome da rede a ser utilizada para a extração das características. Opções disponíveis: MobileFaceNet, MobiFace, SphereFace, OpenFace, FaceNet, e ShuffleFaceNet.

```
 --preprocessing_method [PREPROCESSING_METHOD]
```

define o tipo de pré-processamento à ser aplicada à imagem de entrada. Deverá conter obrigatoriamente uma das opções seguintes de pré-processamento: MTCNN, SphereFace, OpenFace, None.

Exemplo:

```
python3 main.py --operation extract_generate_rank \
                --result_sample_path root/results \
                --dataset LFW \
                --specific_dataset_folder_name lfw \
                --img_extension jpg \
                --model_name sphereface \
                --preprocessing_method sphereface
```

Neste caso, o algoritmo extrai as características das imagens do dataset LFW. Por meio de um *dataloader* previamente definido o algoritmo itera pelas imagens e gera um vetor de características para cada uma delas. Utilizando distância de cossenos, o módulog era o ranking de similaridade é faz a avaliação com todos os vetores. Em seguida são calculadas as métricas que serão reportadas.

2.3.6 Criar ranking de similaridade de uma imagem específica no formato base 64

Objetivo: Gerar ranking com os 10 indivíduos mais similares ao indivíduo da imagem em base64 informada. Para tal, são extraídas as características da imagem em questão, que são então comparadas àquelas salvas no arquivo de características (".mat") informado. É importante ressaltar que as características extraídas independem do formato em que a imagem está salva. Posteriormente ele retorna uma lista de 10 tuplas, contendo o score e o ID dos 10 indivíduos mais similares ao indivíduo

da imagem.

Execução:

```
python3 main.py --operation generate_rank \
                --feature_file [FEATURE_FILE_PATH] \
                --image_query [IMAGE_QUERY] \
                --model_name [MODEL_NAME] \
                --preprocessing_method [PREPROCESSING_METHOD]
```

Parâmetros:

```
--operation generate_rank
```

deve ser mantido obrigatoriamente como **generate_rank**.

```
--feature_file [FEATURE_FILE_PATH]
```

deve conter, obrigatoriamente, o caminho para o arquivo que possui os vetores de características que serão comparados para gerar o ranking de similaridade.

Vale ressaltar que, se o arquivo informado tiver menos que 10 indivíduos diferentes, então a lista do ranking de similaridade terá tamanho igual ao número de indivíduos presentes.

```
--image_query [IMAGE_QUERY]
```

recebe, obrigatoriamente, o caminho para a imagem cujas características serão extraídas e comparadas àquelas no arquivo informado. Caso a imagem tenha extensão ".txt" ou ".json", o algoritmo automaticamente lê a imagem como base64.

```
 --model_name [MODEL_NAME]
```

deve conter o nome da rede a ser utilizada para a extração das características. Opções disponíveis: MobileFaceNet, MobiFace, SphereFace, OpenFace, FaceNet, e ShuffleFaceNet.

```
 --preprocessing_method [PREPROCESSING_METHOD]
```

define o tipo de pré-processamento à ser aplicada à imagem de entrada. Deverá conter obrigatoriamente uma das opções seguintes de pré-processamento: MTCNN, SphereFace, OpenFace, None.

Exemplo:

```
python3 main.py --operation generate_rank \
                --feature_file root/features.mat \
                --image_query root/datasets/LFW/lfw/JK_Rowling/
                JK_Rowling_0004.txt \
                --model_name mobilefacenet \
                --preprocessing_method sphereface
```

Exemplo de saída da execução:

```
[('JK_Rowling', 0.8129734), ('Gray_Davis', 0.053280134),
('Zico', 0.043757956), ('Nelson_Mandela', -0.022397166),
('Will_Smith', -0.07271455), ('Angelina_Jolie', -0.0799012),
('Jennifer_Aniston', -0.08473626), ('Tiger_Woods', -0.12978116),
('Winona_Ryder', -0.14375922), ('John_Travolta', -0.19447823)]
```

Neste caso, o algoritmo extrai as características da imagem *JK_Rowling_0004.jpg* com o pré-processamento SphereFace e a rede MobileFaceNet. Em seguida gera pontuação de similaridade entre o vetor de características recém-criado e cada um dos outros presentes no arquivo "features.mat". Então, retorna lista de 10 tuplas, contendo o score e o ID dos 10 indivíduos mais similares à JK_Rowling.

2.3.7 Criar ranking de similaridade usando uma string no formato base 64

Objetivo: Gerar ranking com os 10 indivíduos mais similares ao indivíduo da imagem, neste caso uma string em base64, informada. Para tal, são extraídas as características da imagem em questão, que são então comparadas àquelas salvas no arquivo de características (“.mat”) informado. É importante ressaltar que as características extraídas independem do formato em que a imagem está salva. Posteriormente ele retorna uma lista de 10 tuplas, contendo o score e o ID dos 10 indivíduos mais similares ao indivíduo da imagem.

Execução:

```
python3 main.py --operation generate_rank \
                --feature_file [FEATURE_FILE_PATH] \
                --image_query [IMAGE_QUERY] \
                --model_name [MODEL_NAME] \
                --preprocessing_method [PREPROCESSING_METHOD]
```

Parâmetros:

```
--operation generate_rank
```

deve ser mantido obrigatoriamente como **generate_rank**.

```
--feature_file [FEATURE_FILE_PATH]
```

deve conter, obrigatoriamente, o caminho para o arquivo que possui os vetores de características que serão comparados para gerar o ranking de similaridade.

Note que, se o arquivo informado tiver menos que 10 vetores de características com ID’s distintos (indivíduos diferentes), a lista do ranking de similaridade terá tamanho igual ao número de indivíduos presentes.

```
--image_query [IMAGE_QUERY]
```

recebe obrigatoriamente a string (em base64) da imagem cujas características serão extraídas e comparadas àquelas no arquivo informado.

É importante ressaltar que a string deve ter o formato semelhante ao mostrado no exemplo abaixo. Ou seja, antes dos elementos codificados deve existir base64 e depois uma virgula da seguinte maneira *base64,*.

É imprescindível que a string esteja entre aspas.

```
 --model_name [MODEL_NAME]
```

deve conter o nome da rede a ser utilizada para a extração das características. Opções disponíveis: MobileFaceNet, MobiFace, SphereFace, OpenFace, FaceNet, e ShuffleFaceNet.

```
 --preprocessing_method [PREPROCESSING_METHOD]
```

define o tipo de pré-processamento à ser aplicada à imagem de entrada. Deverá conter obrigatoriamente uma das opções seguintes de pré-processamento: MTCNN, SphereFace, OpenFace, None.

Exemplo:

```
python3 main.py --operation generate_rank \
                --feature_file root/features.mat \
                --image_query "data:JK_Rowling_0004/jpg;base64,iVBORw
                TdAAANrghEUgA...wJvD53ID/AW4uLhm+ZdrgAAAAAElFTkSuQmCC" \
                --model_name mobilefacenet \
                --preprocessing_method sphereface
```

Exemplo de saída da execução:

```
[('JK_Rowling', 0.8129734), ('Gray_Davis', 0.053280134),
('Zico', 0.043757956), ('Nelson_Mandela', -0.022397166),
('Will_Smith', -0.07271455), ('Angelina_Jolie', -0.0799012),
('Jennifer_Aniston', -0.08473626), ('Tiger_Woods', -0.12978116),
('Winona_Ryder', -0.14375922), ('John_Travolta', -0.19447823)]
```

Neste caso, o algoritmo extrai as características da imagem *JK_Rowling_0004.jpg* que foi informada em base64 com o pré-processamento SphereFace e a rede MobileFaceNet. Em seguida gera pontuação de similaridade entre o vetor de features recém-criado e cada um dos outros presentes no arquivo "features.mat". Então, o módulo retorna lista de 10 tuplas, contendo o score e o ID dos 10 indivíduos mais similares à JK_Rowling.