Michel André Lima Vinagreiro

TUTORIAL RÁPIDO DE RECONHECIMENTO FACIAL VEICULAR UTILIZANDO *FACE RECOGNITION FRAMEWORK*

Fatec Santo André

Novembro de 2019

O objetivo deste estudo é desenvolver um sistema de reconhecimento facial veicular embarcado que possua a função de localizar e classificar faces como pertencentes, ou não, a determinados usuários do veículo.

A primeira parte deste tutorial aborda os problemas que ocorridos e algoritmos utilizados na localização de faces em ambientes não controlados. A segunda parte, relata o problema de classificação de identidade da face localizada e propõe uma abordagem que visa o melhor o desempenho geral do sistema. A terceira parte apresenta todos os processos necessários que possibilitam o embarque de todos os algoritmos envolvidos em plataformas de hardware embarcado.

1. Algoritmo de detecção de faces.

Os algoritmos de detecção enfrentam o desafio de encontrar grupos de *pixels* que correspondem a faces em meio objetos e condições variados.

O objetivo principal é localizar grupos de *pixels,* cujas características de gradientes de mudanças de intensidade assemelham-se aos padrões de características referentes as faces utilizadas como exemplos de comparação.

Imagens capturas em ambientes não controlados tornam-se desafiadoras para os algoritmos em função das condições, como a baixa luminosidade, devido a dificuldade de computação dos gradientes, oclusões parciais, que mudam o padrão de dos grupos de pontos que compõem a face. Outro ponto desafiador que demanda a utilização de recursos computacionais adicionais é a variação de escalas, ou seja, a variação do tamanho das faces, dado o tamanho da imagem. Normalmente, este problema é tratado utilizando a técnica de pirâmide.

O algoritmo *HOG* (histograma de gradientes orientados) computa os gradientes de um determinado grupo de pontos. Após computado, o vetor de gradientes é classificado, utilizando o critério de distancia, com os vetores obtidos a partir das faces exemplos.

Após localizadas as faces na imagem, o algoritmo retorna as coordenadas iniciais e finais da caixa que delimita a face na imagem, estas coordenadas serão utilizadas posteriormente para recortar a imagem da face *(croping)*.

O algoritmo *HOG* utilizado é um dos componentes da biblioteca *face_recognition*, utilizada para aplicações de reconhecimento de face, desenvolvida para a linguagem *python*.

1.1. Recorte da região contendo a face, extração de características e classificação utilizando algoritmo KNN.

A biblioteca computação visual científica denominada *OpenCv* é utilizada para recortar a a parte da imagem referente a região onde a face foi localizada, posteriormente esta imagem recortada é aplicada a um descritor, que consiste em uma rede CNN utilizada para classificação de faces, pré-treinada em milhões de faces, contida no pacote da biblioteca de reconhecimento facial. Como resultado de predição, o descritor gera um vetor de características de *128* elementos.

O vetor resultante é comparado com um número *N* de amostras de pessoas distintas contidas na base de dados e classificada pelo critério de menor distancia euclidiana, sendo que, o vetor que representa a face desconhecida é classificada como pertencente a classe da amostra com a qual possui a maior similaridade, ou seja, menor distancia.

1.2. Montagem da base de dados.

A base de dados é composta por 412 imagens de faces recortadas, sendo 400 pertencentes a classe intruso e 12 imagens diferentes pertencentes a cada uma das 3 classes que representam os usuários autorizados.

O banco de imagens contendo as faces pertencentes a classe intruso é o *dataset* aberto denominado *Fei face database*, contendo 400 imagens referentes a rostos masculinos e femininos, apresentando as expressões neutras e sorridentes. Os rostos são isolados e recortados pelo algoritmo *HOG* e são aplicadas ao descritor.

As imagens referentes aos usuários são capturas dos rostos em posição neutra.

Durante os testes realizados, foi concluído que, um número maior que 4 imagens por usuário facilita o surgimento de falsos positivos e um número menor implica em

dificuldade de localização, mesmo em condições consideradas boas (boa iluminação, ausência de oclusões e etc).

Todas as imagens pertencentes a base de dados são submetidas ao processo de localização de face e extração de características, os vetores resultantes, juntamente acompanhados com seus respectivos nomes, são armazenadas em uma matriz de dimensões 412 linhas por 128 colunas e um vetor de 412 posições contendo as *strings* dos nomes, que é utilizada como base de comparação no processo de predição, dada uma imagem recortada que se deseja classificar.

2. Classificação final baseada em características profundas utilizando *DeepCNN*

Para melhorar a acurácia e diminuir a taxa de falsos positivos, foi empregada, além da classificação baseada em distancia, uma rede neural artificial convolucional que possui a finalidade de classificar as imagens de faces já recortadas. Após recortadas, as imagens são convertidas do padrão de cores *RGB* para níveis de cinza. Após o processo de conversão de padrão de cores, é aplicado o processo de equalização de histograma, melhorando imagens captadas em com condições de iluminação inadequadas. Para tornar as imagens adequadas ao tamanho padrão da camada de entrada, as imagens são redimensionadas para a resolução 224 por 224 *pixels.*

A saída final predita pela rede é um valor numérico binário, que assume o valor 0, se a imagem for classificada como intruso ou 1 se for classificada como o usuário discriminado.

A rede CNN é treinada 3 vezes, gerando uma matriz de parâmetros para cada um dos 3 usuários aos quais se deseja discriminar. O treinamento é realizado utilizando-se dois *datasets*, um contendo 400 faces intrusas provenientes do *dataset Labelded Faces in the Wild* e outro, contendo o mesmo número de faces, do usuário e se discriminar.

A rede CNN utilizada foi confeccionada utilizando a biblioteca *keras python*

2.1. Saída final do sistema.

A decisão final gerada pelo sistema, ou seja, a concessão deliberação ou não de acesso, se dá por uma decisão composta, baseada em lógica “AND”, onde, a pessoa cuja a face analisada somente obtém acesso se a distancia euclidiana em relação a uma face de usuário permitido for inferior a 0.45 e se for classificada pela rede CNN como pertencente a este mesmo usuário.

3. Processo de embarque da arquitetura na plataforma de *hardware Raspberry pi model B*.

Todo processo de treinamento é realizado de modo *off-line* ou seja, todo treinamento, processamento de imagens, confecção dos programas e ajustes de parâmetros são realizados utilizando um computador pessoal, e computação em nuvem, pois, a plataforma de *hardware* embarcado utilizada não possui poder computacional que possibilite tais operações.

A manipulação das saídas físicas da plataforma da *hardware* foram possibilitadas utilizando a biblioteca *Rpi.GPIO*, utilizando a linguagem *python*.

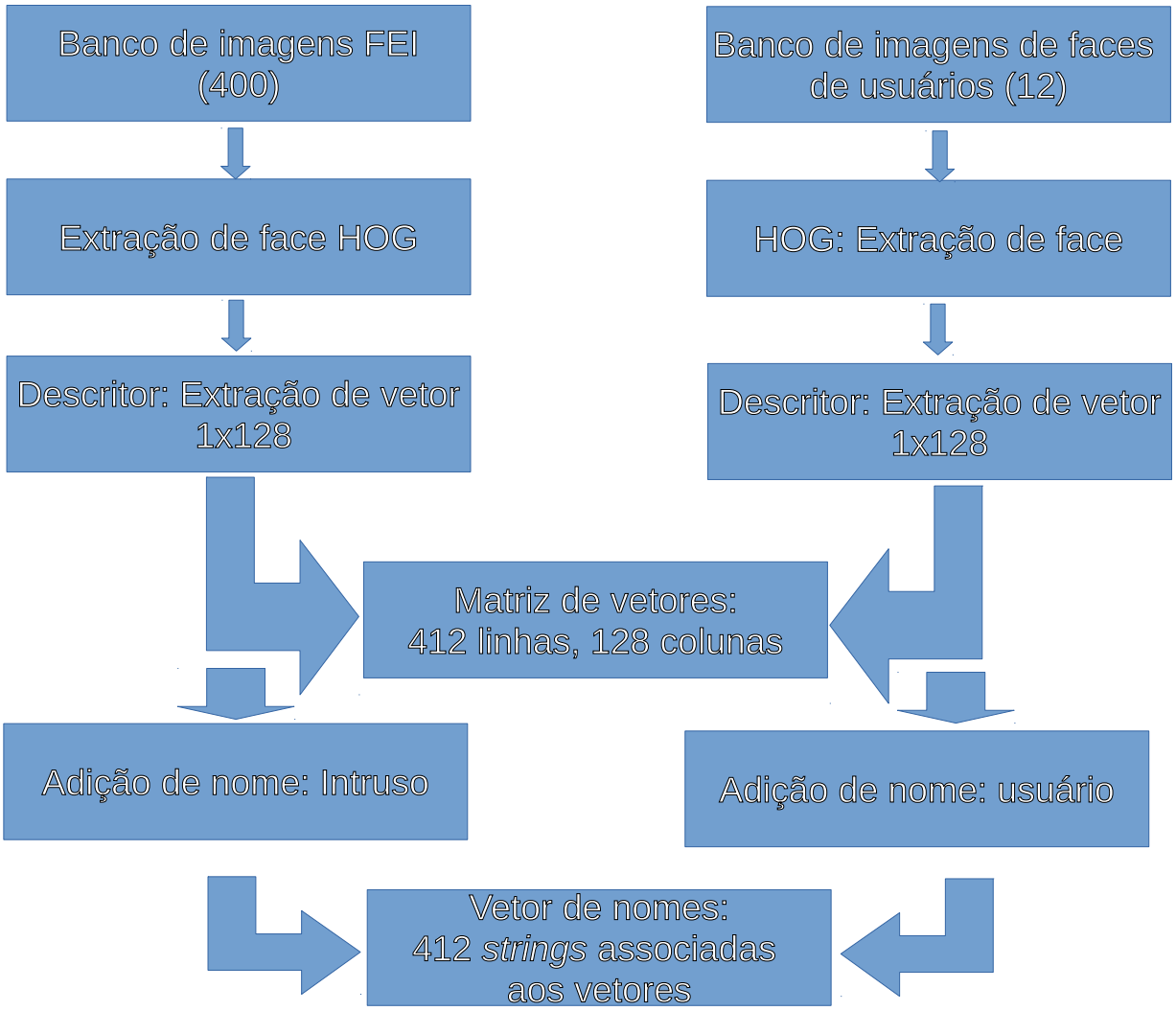

Figura 1: Fluxograma do processo de obtenção da matriz de vetores.

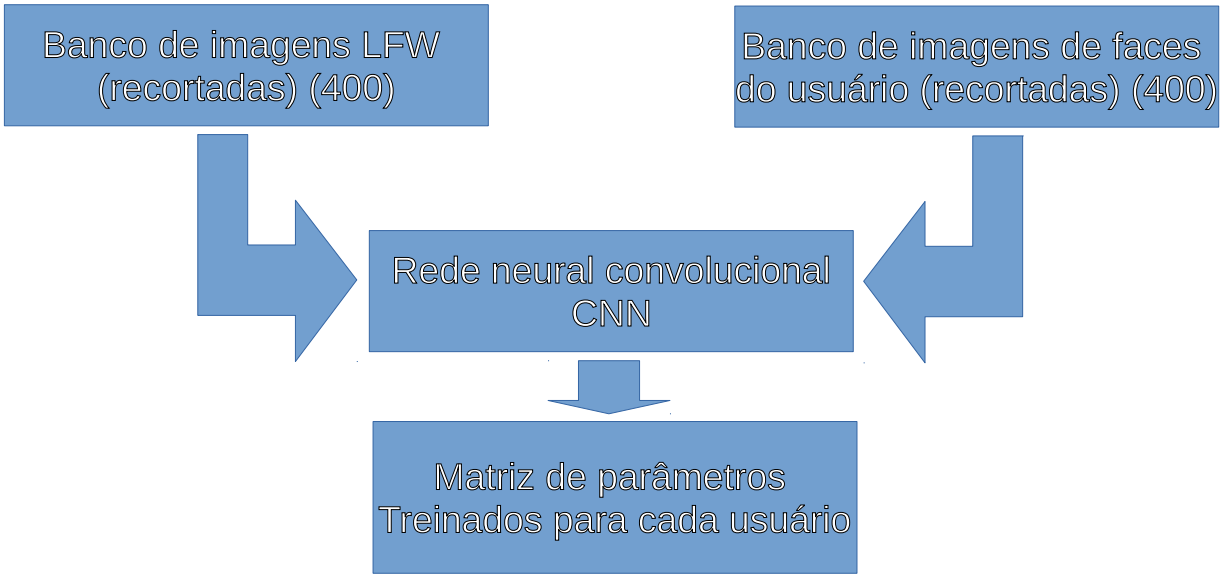

Figura 2: Fluxograma do processo de treinamento da rede CNN.

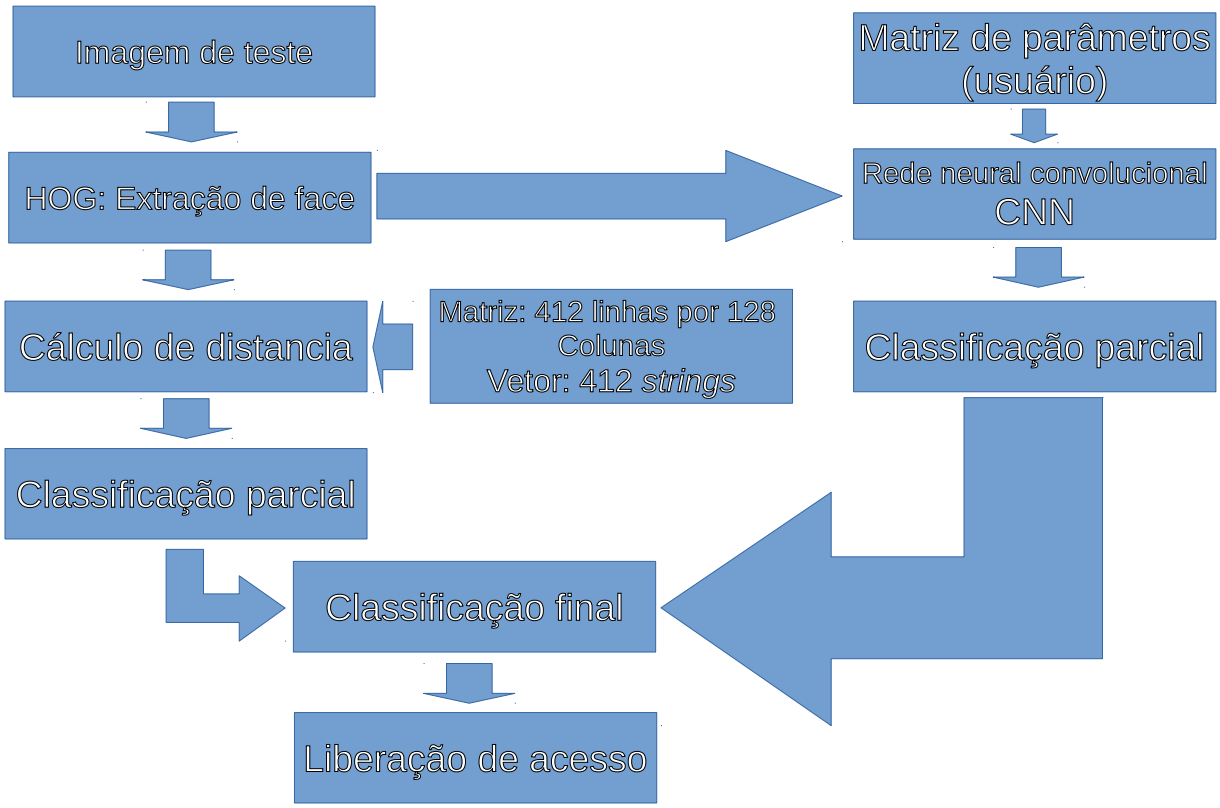

Figura 3: Fluxograma de controle de acesso do sistema dada uma imagem de teste.