Faculdade de Ciências da Universidade do Porto
Departamento de Ciência de Computadores
Visão Computacional

Reconhecimento facial para registo de presença em aulas

Pedro Unas
Sara Ferreira

Dezembro
2019

Conteúdo

1 Introdução

No âmbito da cadeira de Visão Computacional foi nos dada a oportunidade de desenvolver um projeto relacionado com a área que a cadeira aborda. Após alguma pesquisa sobre possíveis temas que poderíamos usar, decidimos-nos focar na área do reconhecimento facial, criando um protótipo para resolver um problema comum no ensino superior, as presenças em aulas obrigatórias e não obrigatórias.

Atualmente na Faculdade de Ciências o método mais comum para a marcação de presenças em aulas práticas ou teórico-práticas é a passagem de uma folha de presenças para ser assinada pelos alunos na aula. O principal problema com este método é que alunos que não estejam presentes podem pedir a colegas que assinem por eles de modo a que estes não tenham falta.

A nossa visão para tentar colmatar este problema foi a criação de um sistema para identificar os alunos presentes na aula, no início da mesma ou até a meio. A solução ideal seria, por exemplo, a utilização de uma *webcam* ligada a um computador ou sistema Raspberry Pi que correria automaticamente durante um tempo estipulado e no fim do mesmo enviaria ao professor regente ou responsável pela turma um email com a lista dos alunos presentes na aula.

Devido a questões de tempo e de capacidade técnica, o nosso trabalho apresenta uma solução mais simples em que, usando um computador portátil, corremos o programa que abre a *webcam*, reconhece as pessoas que nela aparecem e no fim da execução enviámos o email com as presenças. Além disso, o programa é capaz de identificar se está no horário da aula de visão computacional, bem como, de forma automática, recolher o nome de todos os estudantes inscritos.

2 Estado da arte

2.1 Python

Esta foi a linguagem escolhida por nós para a realização deste projeto, devido ao nosso conhecimento prévio da mesma e por ser a linguagem que consideramos mais adequada para um projeto deste tipo visto existirem bibliotecas dedicadas à área da Visão Computacional bem como maneiras simples de ler vários ficheiros de maneira rápida e eficiente.

2.2 OpenCV

Uma biblioteca *open source* dedicada a resolver problemas referentes à Visão Computacional. Foi construída de modo a oferecer uma estrutura comum para programas desta área. A biblioteca conta com mais de 2500 algoritmos que vão desde reconhecimento de caras à extração de modelos 3D de objetos. Está disponível em várias linguagens de programação como C++, Java e claro Python.

2.3 Face_recognition

Esta biblioteca para Python permite de maneira *straight forward* o reconhecimento de caras e a manipulação das mesmas. As funções que contem foram extremamente úteis para a realização do projeto pois são fáceis de entender e extremamente eficazes. A utilização desta biblioteca juntamente com a biblioteca OpenCV facilitou o desenvolvimento deste projeto.

2.4 Selenium

Este pacote para Python permite a interação do *browser* com o Python, permitindo ao programa simular a interação que um utilizador humano teria com uma página web. Foi necessária a utilização do Selenium para tentar, automaticamente, retirar do Sigarra a lista de alunos e as suas respetivas fotos, visto que para aceder a lista de alunos de uma cadeira é necessário fazer *login* com um perfil da faculdade. Escolhemos esta ferramenta por ser muito intuitiva, permitir clicar em butões e dar para aceder a qualquer elemento de uma página através do seu xPath ou id.

2.4.1 XPath

XPath, a XML Path Language, é uma linguagem de consulta para selecionar nós de um documento XML. Ademais, XPath pode ser usada para computar valores do conteúdo de um documento XML.

2.4.2 Gekodriver

O selenium necessita de um webdriver para interagir com o browser escolhido. Neste caso decidimos usar o Firefox que necessita então do geckodriver.

2.5 Smtpblib

Para a parte final da execução do nosso projeto, decidimos utilizar esta biblioteca que permite o envio de emails em Python usando o protocolo SMTP que é um dos mais utlizados atualmente. A biblioteca permite a ligação ao servidor SMTP da Google o que permite que enviemos emails através de uma conta Gmail criado para este projeto. A principal desvantagem da biblioteca smtplib é a exposição em *plain text* da password de remetente que é visível a qualquer pessoa com acesso ao *source code*.

3 Descrição do projeto

3.1 Primeiro contacto com o reconhecimento facial

Começamos por criar um *script* em Python que, a partir de fotografias dadas reconhecia as pessoas nas mesmas. Para isto, usamos a biblioteca face_recognition da linguagem escolhida por nós para este projeto, Python. Começamos por criar um diretório com as fotos do Sigarra dos alunos inscritos à cadeira de visão computacional a que demos o nome "known". Seguidamente, criamos outro diretório com fotos nossas para testar se conseguíamos que o programa nos conseguisse identificar ao qual chamamos "unknown".

Usamos então a biblioteca face_recognition para tentar reconhecer caras que estavam no diretório "unknow"a partir das dadas do directório "known". Para cada cara fazemos o *encoding* usando uma função (da biblioteca face_recognition referida anteriormente) em que, dada uma foto retorna um vetor com 128 dimensões em que cada uma representa um componente ortogonal da cara presente na foto.

Figura 1: Funcionamento téorico da função face_encoding

Se usarmos a função face_landmarks() é possivel também identificar os pontos na cara tais como o lábio de cima e o olho da esquerda. Com recurso a essa informação marcamos esses pontos na cara identificada na imagem. Com esta informação é possível verificar, por exemplo, se uma pessoa está a sorrir ou não.

3.2 Uso do openCV

Chegamos a um ponto em que a biblioteca usada anteriormente já não satisfazia completamente as nossas necessidades visto que queríamos desenvolver algo mais complexo e que identificasse mais que imagens estáticas.

Utilizando então a biblioteca OpenCV para Python ,usamos a *webcam* de um computador portátil, de modo a imitar uma câmera na sala de aula e, sem ser necessário dar uma imagem como input, reconhece quem aparece na *webcam* desde que a foto dessa pessoa esteja no diretório "known", ou seja, no âmbito do nosso projeto, se a pessoa estiver inscrita na cadeira.

Quando uma pessoa é identificada na *webcam*, o seu nome é adicionado a uma lista que posteriormente vai ser enviada ao professor da respetiva cadeira por *email*.

3.3 Envio de emails com Python

O uso de emails foi a maneira mais lógica que encontramos para comunicar ao professor respetivo da cadeira as presenças da aula visto que é rápido é fácil de aceder em qualquer lado.

Para tal usamos a biblioteca smtplib, estabelecendo conexão com o servidor SMTP da Google visto que usamos um email da Google criado para este projeto. Para o envio da lista é obtida automaticamente o dia da aula e é usado o email do professor como destinatário do email.

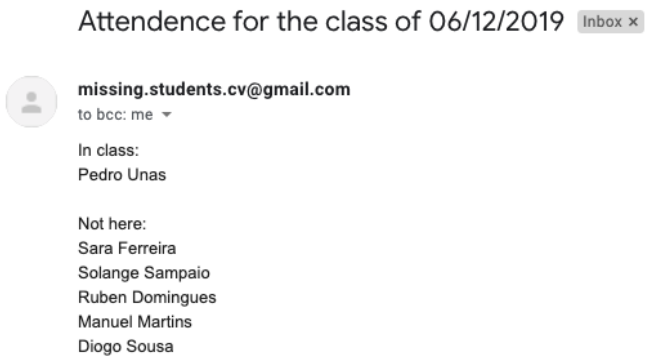

Figura 2: Exemplo de um email com as presenças numa aula

3.4 Scraper para recolha das fotos dos alunos

No Sigarra, para conseguirmos aceder à lista de alunos inscritos numa cadeira é preciso estarmos autenticados. Por isso, para a obtenção da lista de alunos inscritos e das respetivas fotos, desenvolvemos um *scraper* que faz login no Sigarra de forma automática usando as credenciais de um dos membros do grupo.

Após alguma pesquisa constatamos que no endereço da lista de alunos de cada cadeira apenas muda o *ID* da mesma. Por isso, após termos o *ID* da cadeira de Visão Computacional, temos acesso à lista de alunos e, usando o nosso *scraper* guardamos num ficheiro de texto o nome de cada aluno inscrito, mas infelizmente não conseguimos de forma automática guardar também a foto de cada aluno, sendo proibido pelo Sigarra. Por isso, esse processo teve de ser feito à mão.

3.5 Execução do programa final

Ao iniciar o programa principal, começamos por percorrer o diretório com as fotos dos alunos e guardamos os seus nome numa lista previamente obtido usando o *scraper* criado por nós. Seguidamente, fazemos o *encoding* de cada cara usando a biblioteca face_recognition como explicado anteriormente, repetindo o processo para todos as fotos.

Começamos então o *feed* da webcam e usamos o OpenCV para reduzir a dimensão de cada *frame* para um quarto do seu tamanho original para melhor a performance do programa. Quando uma cara é detetada o programa obtém a sua localização e faz *encoding* da mesma usando a função referida acima. Posteriormente itera pela lista que contem os vetores de 128 dimensões que representam a cara dos alunos inscritos na cadeira. Seguidamente, compara ambos os *encoding* (o obtido pela *webcam*, bem como o obtido a partir do diretorio com as cara dos alunos) usando a distância euclidiana e, caso seja melhor que o valor da tolerância dado, neste caso 0.6, retorna *True* sendo esta informação guardada numa lista.

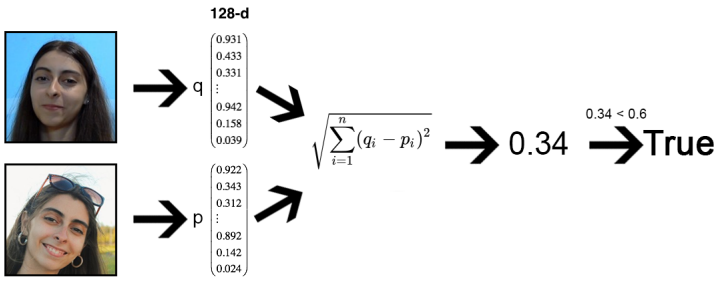

Figura 3: O funcionamento da função que compara o encoding de duas caras

Após isto volta a percorrer a lista de vetores de caras calculando de novo as distâncias euclidianas entre dois *encodings* de modo a obter a cara cuja a distância é menor, ou seja é a mais parecida matematicamente. Verifica depois se a função anterior retornou *True* para esta cara e em caso afirmativa guarda o nome como identificado.

Finalmente, de modo a visualizar que a cara foi identificada é desenhado no *feed* da webcam um retângulo à volta da cara ou das caras em foco com uma etiqueta com o nome identificado ou *"Unknown"* caso a cara não seja conhecida. Caso seja a primeira vez que a cara é identificada é guardada numa lista que será usada na fase final de execução.

Para terminar o *feed* da webcam, o utlizador pressiona a tecla 'q' e automaticamente começa a fase final de execução onde é aberto um ficheiro de texto que será usado como o corpo do email que será enviado para o professor. Depois é percorrida a lista de caras identificadas e são guardadas no ficheiro como alunos presentes, sendo as restantes guardadas no ficheiro como não presentes. Por fim é chamado o programa desenvolvido por nós para o envio do email que faz uso do ficheiro texto criado e envia o seu conteúdo.

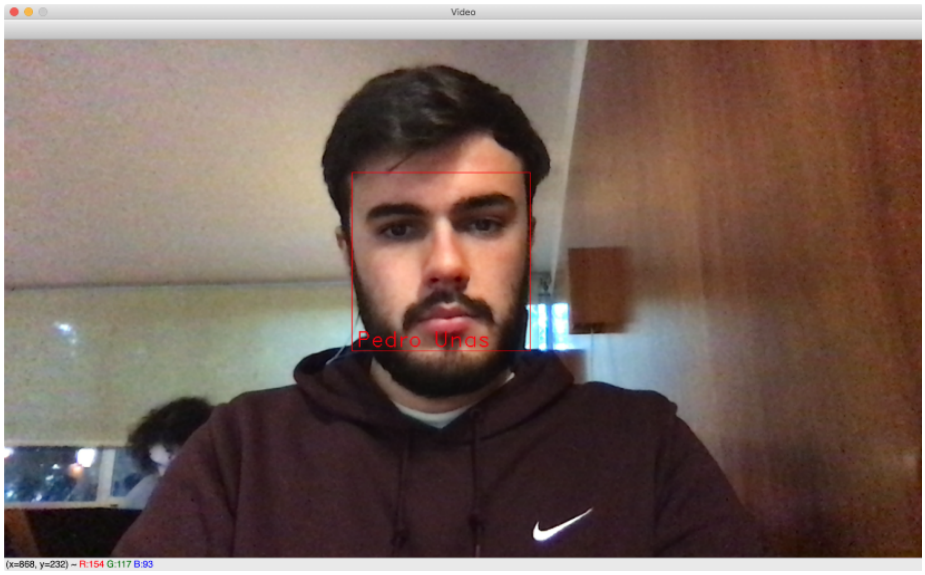

Figura 4: Exemplo da identifação de uma cara no programa principal

4 Conclusão

Com este projeto ficamos a conhecer algumas bibliotecas de python pré-existentes no âmbito do reconhecimento facial e como as usar. O próximo passo no trabalho seria testar os scripts criados com uma câmera numa sala de aula ligada a um Raspberry pi. Outro ponto que poderiamos melhorar era obter automaticamente com recurso ao webdriver, as fotos de todos os alunos inscritos a uma certa cadeira, algo que não estava ao nosso alcance. Para isto, teriamos de ter permissões especiais para o conseguir. De qualquer forma, conseguimos executar a nossa ideia inicial com sucesso e por isso, o nosso objectivo foi concluído.

5 Bibliografia

1- https://github.com/ageitgey/face_recognitions, acedido a 5/12/2019
2- https://en.wikipedia.org/wiki/Euclidean_distance, acedido a 8/12/2019
3- https://docs.opencv.org/master/d0/de3/tutorial_py_intro.html, acedido a 5/12/2019
4- https://pypi.org/project/Pillow/, acedido a 5/12/2019