Eigenfaces para Reconhecimento Facial

Fidel Luis Como
Anna Clara Damasio

1 Introdução

Eigenfaces é um método útil para reconhecimento facial que foi desenvolvido com a finalidade de computadores reconhecerem faces. Esse modelo pode ser usado para solucionar uma grande variedade de problemas como: identificação criminal, sistemas de segurança, processamento de imagens e vídeos e interação homem-máquina.

De forma resumida os eigenfaces, extraem as informações faciais relevantes, que podem ser considerados um conjunto de características global entre os rostos humanos, como olhos, nariz e lábios. Um jeito de fazer isso é pegar a variação estatística entre os rostos. Logo Depois, cada imagem é aproximada usando um conjunto de eigenfaces, que será aquelas com maior autovalor.

O objetivo deste trabalho é explicar o método eigenfaces e mostrar a Álgebra Linear por trás dele, construir o algoritmo em Python e aplicar em testes que possa ser discutido sua relevância.

2 Eigenfaces

Antes de começar a explicar as etapas do Eigenface é necessário entender a Análise de Componentes Principais ou PCA. É uma técnica de redução de dimensionalidade, ou seja, ela ajuda a definir os componentes que melhor descrevem um conjunto de dados reduzindo a dimensionalidade desses dados. Algumas informações serão perdidas neste processo, mas perdemos pouca precisão em troca de um conjunto de dados muito mais fácil de trabalhar. A ideia por trás dele é projetar linearmente os dados originais em um subespaço dimensional menor oferecendo aos componentes principais variação máxima dos dados projetados ou erro mínimo de distorção da projeção. O subespaço gerado é chamado de subespaço principal.

Em 1991 os professores Matthew Turk e Alex Pentland chegaram à conclusão que, diferentemente do que foi proposto anteriormente, as características que são relevantes para o reconhecimento facial não devem ser, necessariamente, as mesmas que têm importância para o ser humano. Eigenfaces busca capturar a variação de várias imagens de rostos e pega essas informações para codifica e comparar as imagens das faces individuais, com uma abordagem baseada em partes.

O Eigenface estabelece um rosto como um vetor, esse método armazena mais informações em tamanhos menores, usando a Álgebra Linear. Sendo assim, a face é decomposta em vários componentes principais, podendo então formar a rosto humano através de formas fantasmas. São autovetores da matriz de covariância, definidos por um conjunto de outras faces de referência. Cada face pode ser representada como a combinação linear de diversas Eigenfaces.

3 Calculando as Eigenfaces

3.1 Base de Dados

Primeiramente é necessário um dataset composto por fotos de rostos que tenham aproximadamente a mesma centralização e o mesmo tamanho. Utilizamos o dataset, onde temos imagens de 36 pessoas (Figura 1) e 64 fotos em diferentes condições de luminosidade para cada pessoa (Figura 2) para um total de 2304 imagens.

Figura 1: Caras das pessoas presentes na base de dados.

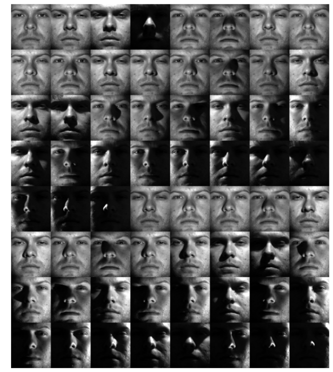


Figura 2: Diferentes condições de luminosidade.

Posteriormente iremos a pegar cada imagem de 168×192 pixeis de dimensão e juntar elas dentro de uma única matriz F em forma de vetores coluna $\mathbf{f}_i$ de $168 \cdot 192 = 32256$ dimensões.

$$F = \begin{bmatrix} | & & | \\ \mathbf{f}_1 & \dots & \mathbf{f}_{2304} \\ | & & | \end{bmatrix}_{32256 \times 2304}$$

3.2 Face media e decomposição SVD

Somando todas as colunas de F e dividindo pelo numero de colunas obtemos a face media (Figura 3):

$$\mathbf{f}_{media} = \frac{1}{2304} \sum_{i=1}^{2304} \mathbf{f}_i$$

Figura 3: A face media obtida juntando todas as caras da base de dados.

O seguente passo será subtrair a face média de cada coluna da matriz para depois aplicar a decomposição SVD.

$$\overline{F} = \begin{bmatrix} | & & | \\ \mathbf{f}_1 - \mathbf{f}_{media} & \dots & \mathbf{f}_{2304} - \mathbf{f}_{media} \\ | & & | \end{bmatrix}_{32256 \times 2304}$$

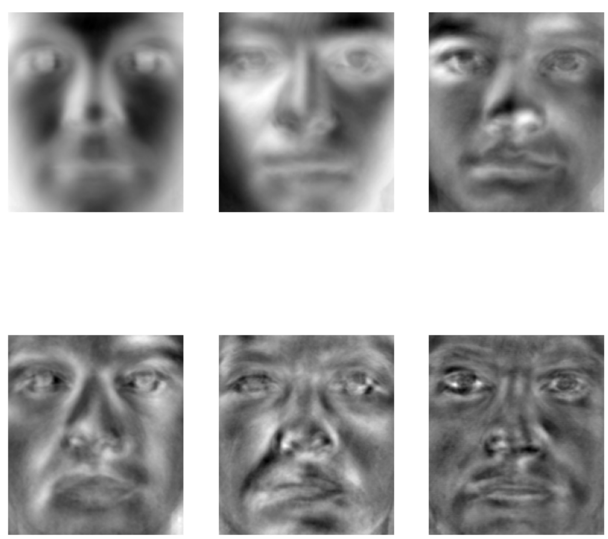
$$\overline{F} = U \Sigma V^T$$

3.3 Matriz U e as Eigenfaces

As colunas da matriz U representa as bases ortonormais do espaço que contém as faces da matriz $\overline{F}$, que podem ser redimensionadas na forma de uma imagem e que serão as próprias eigenfaces (Figura 4).

$$\overline{F} = \underbrace{\begin{bmatrix} | & & | \\ \mathbf{u}_1 & \dots & \mathbf{u}_{2304} \\ | & & | \end{bmatrix}}_{eigenfaces} \Sigma V^T$$

Figura 4: Algumas das colunas de U redimensionadas em forma de eigenfaces.

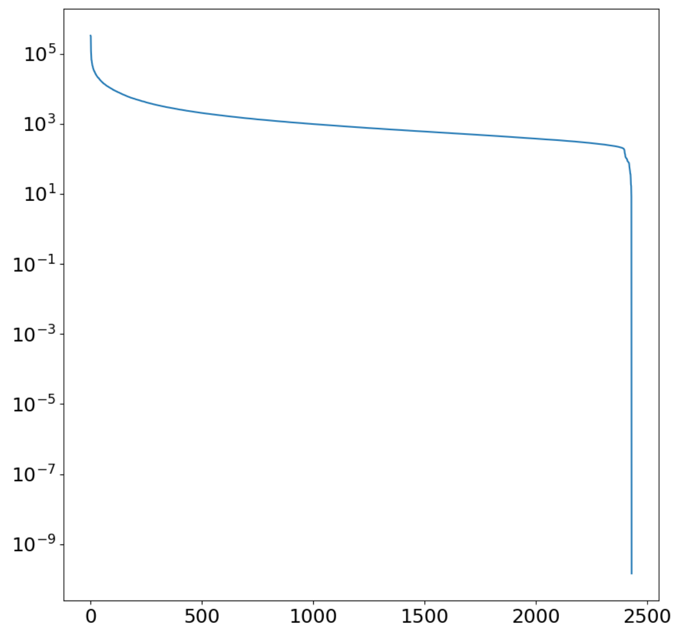


É interessante observar como as eigenfaces correspondentes as primeiras colunas da matriz U são muito parecidas com a face média, no sentido que elas capturam características comuns a todas as caras, e avançando dentro das colunas da matriz começamos a pegar sempre mais detalhes específicos.

3.4 Matriz Σ e PCA

A matriz sigma é uma matriz diagonal com os valores singulares na diagonal em ordem decrescente (Figura 5), o que nos fala que as primeiras colunas da matriz U são aquelas que carregam maior parte das informações.

Figura 5: Valores da diagonal de sigma em escala logarítmica.



Podemos observar como esses valores decrescem muito rápido e se estabilizam. Isso é o que nos permite aplicar o processo de PCA, ou seja aproximar o espaço coluna da matriz original usando só as primeiras r colunas da matriz U, e essa descida repentina no último valor de sigma, significa que só quando todas as colunas de U foram incluídas conseguiremos uma aproximação perfeita desse espaço.

Para achar uma aproximação de uma imagem x dentro do espaço gerado por r colunas de U, iremos a projetar a imagem dentro desse espaço, multiplicando a transposta dessas colunas pelo vetor da imagem, o que nos dá o vetor α de r dimensões, que de certa forma é a digital da imagem projetada, pois ele vai me dizer a combinação linear das r colunas escolhidas para obter a minha aproximação.

$$\alpha = \underbrace{\begin{bmatrix} — & \mathbf{u}_1^T & — \\ & \vdots & \\ — & \mathbf{u}_r^T & — \end{bmatrix}}_{U_r^T} \cdot x$$

Podemos agora utilizar esse vetor α para reconstruir uma aproximação da imagem original, multiplicando-o de volta pelas mesmas colunas de U (Figura 6).

$$x_{aprox} = U_r \cdot \alpha$$

4 O Processo de Reconhecimento

Agora iremos a plotar as projeções das faces de dois indivíduos da base de dados usando as colunas 5 e 6 da matriz U (Figura 7).

É possível perceber como os dois conjuntos de dados ficam bem separados, portanto se tivéssemos uma imagem a mais de um dos indivíduos conseguiríamos distinguir a qual dos dois pertencem essa nova imagem apenas observando para a posição da sua projeção no mesmo espaço.

Figura 6: Aproximações de uma face que não estava presente na base de dados original usando as primeiras r colunas de U.

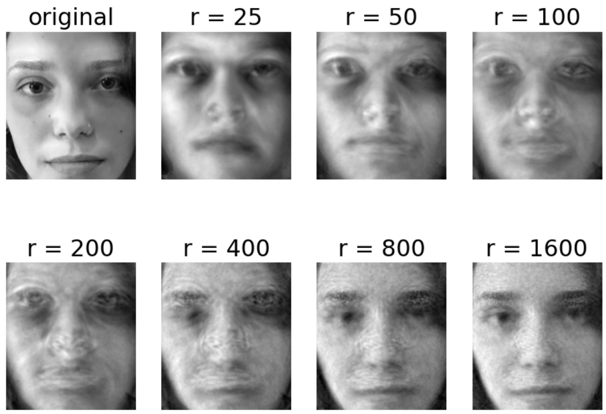
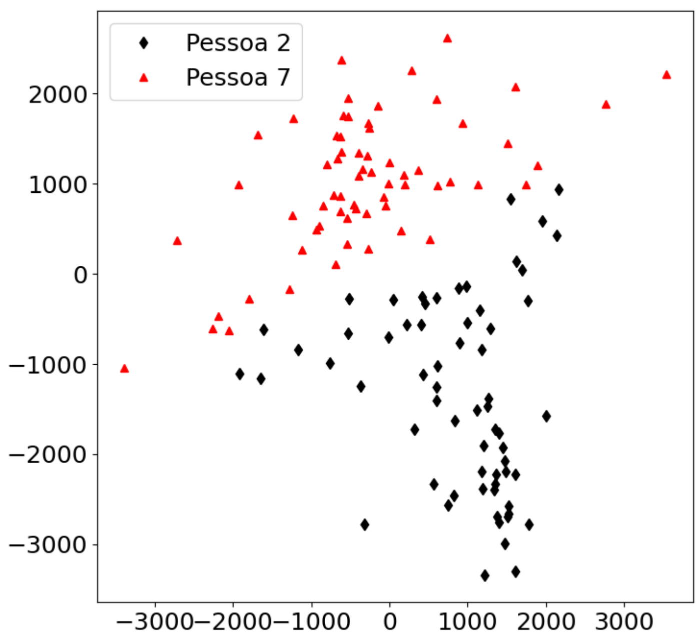
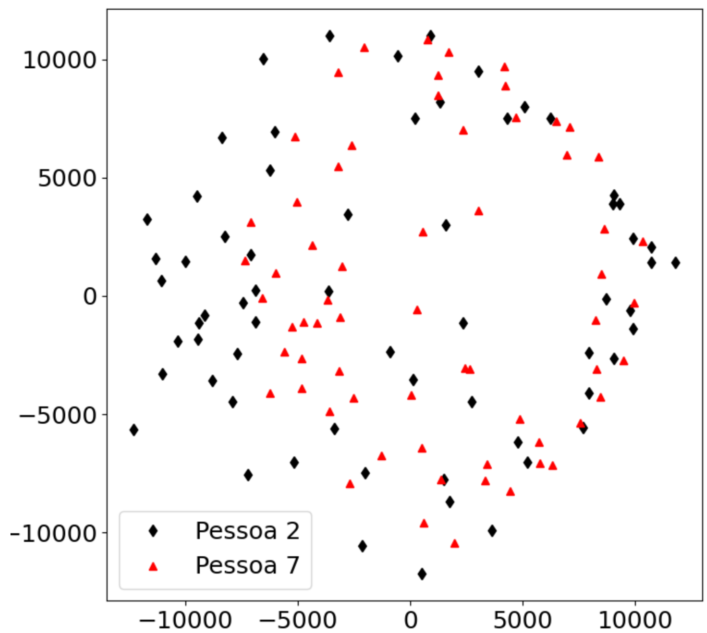


Figura 7: Projeções dos indivíduos 2 e 7 dentro do espaço gerado pelas colunas 5 e 6.



Agora uma observação é que, se tivéssemos escolhido uma das primeiras colunas de U, por exemplo a primeira e a segunda, para projetar essas imagens, não teríamos conseguido a mesma separação entre os dados, devido precisamente ao fato que as primeiras colunas capturam as características comuns a todas as faces (Figura 8).

Figura 8: Projeções dos indivíduos 2 e 7 dentro do espaço gerado pelas colunas 1 e 2.



5 Demonstração

Vamos pegar imagens de duas pessoas que não estejam na base de dados original e vamos tentar distinguir entre as duas por meio de uma algoritmo que encontre a menor distancia entre os α das imagens de referencia e as imagens a ser reconhecidas.

Vamos por exemplo a escolher imagens das celebridades Ben Affleck (Figura 9) e Elton John (Figura 10):

Figura 9: Imagens de referencia para Ben Affleck.

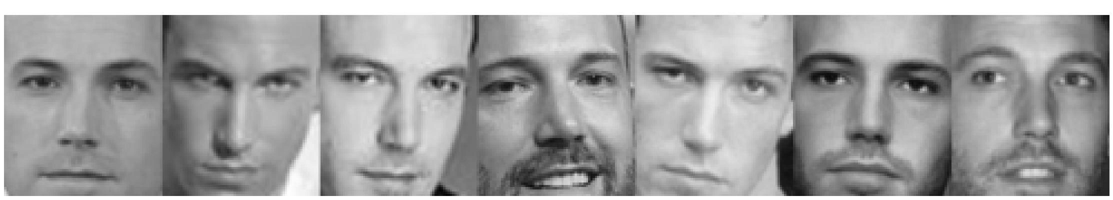
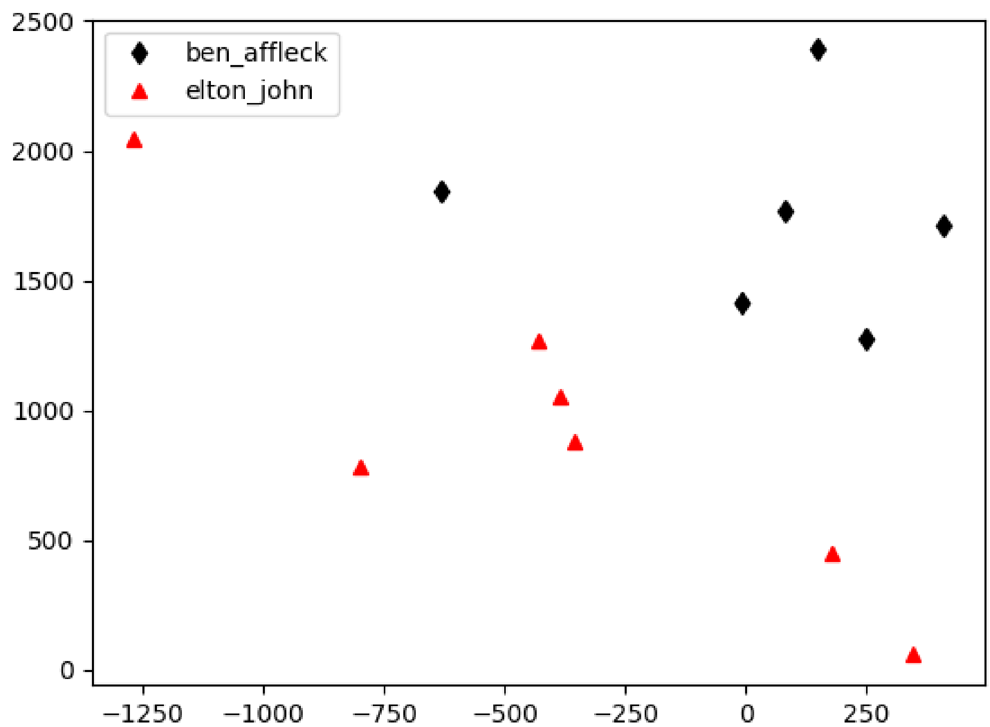


Figura 10: Imagens de referencia para Elton John.

Plotando os vetores α de cada imagem dentro do espaço gerados pelas colunas 6 e 7, além de estar usando poucas imagens, já da para ver essa separação entre os dois conjuntos (Figura 11).

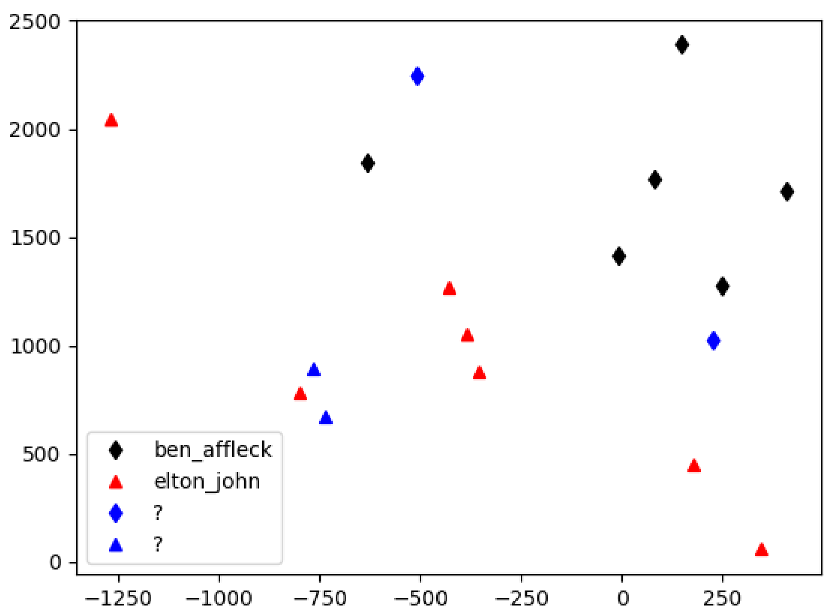
Figura 11: Projeções das imagens usando as colunas 6 e 7.



Agora, vamos a pegar outras imagens das mesmas pessoas e ver se o algoritmo consegue distinguir entre elas (Figura 12, 13).

Figura 12: Resultados da aplicação do algoritmo.

Figura 13: Plotagem das imagens a ser reconhecidas.



Podemos ver que as imagens foram individuadas com sucesso.

6 Conclusão

Neste trabalho foi possível concluir a relevância da Álgebra Linear para o algoritmo de eigenface. O qual se mostrou bastante eficaz na reconstrução de faces, porém para outras imagens sem ser de rostos ele não tem eficácia. Também obtemos bons resultados para o reconhecimento de faces quando comparamos a face de duas pessoas, o algoritmo consegue separar bem os dados de cada indivíduo.

Portanto o algoritmo é eficaz para solução de vários problemas, mas é notório que não é o método mais automatizado, devido a que ele termina sendo bastante superficial. Coisas como tonalidade da pele ou cor do cabelo acabam sendo priorizadas pelo simples fato delas ocupar maior parte dos pixeis na imagem e portanto carregando a maior parte da informação, e è por isso que não é o sistema usado no mercado atual.