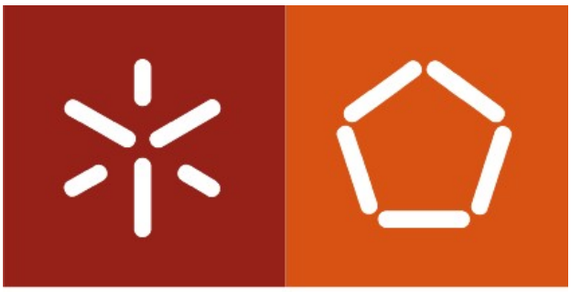
Mestrado Integrado em Engenharia Informática

Métricas em Machine Learning
Grupo B



Universidade do Minho

Relatório Trabalho Prático
EigenFaces: Reconhecimento de Faces

Bruno Nascimento, a67647
Bruno Sousa, a78997
João Palmeira, a73864
Rafael Silva, a74264
Ricardo Pereira, a77045

12 de Janeiro de 2020

Conteúdo

1 Introdução

O primeiro foco de atenção dos seres humanos nas relações sociais são as faces (rostos), ou seja, a maior parte da informação captada por cada pessoa é através da visão. Toda esta informação recolhida através do sentido visual desempenha um papel fundamental na captação de emoções, sentimentos permitindo retirar conclusões de cada indivíduo ou mesmo do seu estado de espírito, reconhecendo sempre a dificuldade de inferir inteligência ou carácter de uma aparência facial, a capacidade do ser humano reconhecer rostos é notável.[**livro˚2**]

O reconhecimento facial é um sistema desenvolvido para identificar uma pessoa por intermédio de uma imagem ou um vídeo, sendo o seu maior entrave a riqueza de expressões faciais humanas. Esta é uma tecnologia que já existe há décadas, mas o seu uso tornou-se mais perceptível e acessível nos últimos anos devido ao desenvolvimento nas áreas de segurança e telecomunicações, uma vez que é utilizada em soluções inovadoras, como aplicações pessoais de fotografia e de autenticação para dispositivos móveis. Neste contexto, o ser humano possuí o melhor sistema reconhecimento facial, capaz de resistir às alterações físicas dos indivíduos tanto devido ao envelhecimento ou pela utilização de adereços, como é o caso dos óculos, brincos, maquilhagem, entre outros.

Neste trabalho, vamos desenvolver um sistema de reconhecimento de faces com base no método *Principal component analysis* (PCA). O PCA foi inventado em 1901 por *Karl Pearson*, como um análogo do principal teorema do eixo em mecânica, mais tarde, em 1991, *Matthew Turk* e *Alex Pentland* utilizaram o PCA para criar um método para o reconhecimento facial através dos vectores próprios associados aos maiores valores próprios da matriz de covariância de um certo *dataset*.[**livro˚2**] Como tal vamos descrever o PCA e o modo de aplicação no nosso trabalho, passaremos pela implementação do sistema em *Python* e os testes com os *datasets* escolhidos e, por último, uma análise dos resultados obtidos.

2 PCA no reconhecimento de faces

A aplicação do PCA (*Principal Component Analysis*) tem por base a redução a dimensão dos dados, constituído por bastantes variáveis correlacionadas entre si, umas mais que outras. Ao mesmo tempo que mantém a variação presente no *dataset*, o mesmo é feito ao transformar as variáveis existentes num novo conjunto de variáveis, que têm o nome de *Principal Components*, são ortogonais e estão ordenados de tal forma que os valores decrescem persistentemente.

Desta forma o primeiro valor da componente retém a máxima variação que estava presente nas componentes originais.

Os principais componentes são os ***eigenvectors* da matriz de covariância**, daí serem ortogonais. É importante referir que o *dataset* onde vai ser aplicado **PCA** deve ser processado á síntese de escala, os resultados são bastante sensíveis a este critério de tratamento de dados.

Intuitivamente, o **PCA** pode retornar um objecto de menores dimensões, quer seja uma imagem, projecção, sombra, etc.[**ref˚url2**]

- ***Dimensão***: Número de variáveis aleatórias num *dataset* ou o número de características (número de colunas presentes no *dataset*);
- ***Correlação***: Demonstra a similaridade entre duas variáveis fortes, o valor varia entre -1 e +1, neste último caso quando o valor de uma variável aumenta a outra também aumenta e vice-versa;
- ***Ortogonal***: Valores não correlacionais;
- ***EigenVectors***: *Eigenvectors* e *EigenValues* são um domínio importante
- ***Matriz Covariância***: Matriz constituída pelos valores de covariância entre pares de variáveis;

2.1 Passos do Algoritmo

Neste sub capitulo serão descritos os principais passos de construção de um **algoritmo de PCA** aplicado a imagens e todos os passos necessários, desde do input de dados á elaboração de componentes principais.

2.1.1 Uniformização dos Dados

Em *datasets* que contem diferentes medidas de escala, deve-se uniformizar os dados para cumprir o requisito de melhorar a performance do algoritmo de *Machine Learning*

2.1.2 Cálculo da Matriz de Covariância

$$MatrizCov = \begin{pmatrix} Var[X_1] & Cov[X_1, X_2] \\ Cov[X_2, X_1] & Var[X_2] \end{pmatrix} \tag{1}$$

A seguinte fórmula representa a covariância entre duas componentes

$$cov_{x,y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N - 1} \tag{2}$$

2.1.3 Cálculo de *EigenValues* e *EigenVectors*

Por intermédio da matriz de covariância, retira-se os *EigenValues* a partir da equação $\det(\lambda_I$ - A) = 0, onde **I** identifica a matriz de mesma dimensão que **A** que é necessária para a subtracção da matriz.

Para cada *EigenValue* obtém-se o correspondente *EigenVector* **v**, ao resolver a seguinte equação: (λI-A)v = 0.

2.1.4 Escolha de Componentes e elaboração do vector Característico

Ordenamos os *EigenValues* do superior para o inferior para que seja possível obter as características e a sua significância.

Agora ao analisar cada *EigenVector*, observamos todos os n *EigenValues*, o que possuir maior valor é o principal componente, para reduzir as dimensões devemos escolher p *EigenValues* que possuem maior valor dos n iniciais. De seguida cria-se um vector de características, em que neste caso é uma matriz de vectores, *EigenVectors*, mas apenas os que queremos prosseguir, e numa dimensão mais reduzida. Esta nova matriz tem o nome de Projecção com p de n *EigenValues* selecionados.

2.1.5 Formação dos Componentes Principais

Depois de todos os cálculos efetuados até este ponto, ao vector característico calcula-se a transposta e multiplica-se esse valor com a transposta da matriz original de escala do *dataset*.

$$NovosDados = VectorCaracteristico^T \times ScaledData^t \tag{3}$$

2.2 EigenFaces

A razão de toda esta explicação do **PCA** consiste no facto de ser frequentemente utilizado como técnica na redução de dimensões em domínios como, por exemplo, reconhecimento facial, visão por computador, compressão de imagem, entre outros. Também é usado para descobrir padrões em grandes quantidades de dados.

Em reconhecimento facial, uma imagem teste possui tamanho **N** $\times$ **N** e pode ser representada por uma matriz também **N** $\times$ **N**.

Ao introduzir uma imagem que não faz parte das previsões, a máquina deteta várias discrepâncias. Com o **PCA** aplicado o processo é eficaz ao comparar os dados da imagem original com a matriz trabalhada do conjunto de testes, além disso o **PCA** permite retirar algumas componentes sem perder demasiada informação, e consequentemente reduz a complexidade do problema.

3 *Datasets*

Neste trabalho prático, de modo a testar o modelo desenvolvido, foram utilizados dois *datasets*: ***Yale Face Database B*** e ***Dataset* do Grupo**.

3.1 *Yale Face Database B*

Este *dataset* foi publicado pela *UCSD Computer Vision*, sendo composto por 165 fotografias de 15 indivíduos diferentes com 11 expressões faciais diferentes e/ou configurações distintas (feliz, normal, triste, sonolento, surpreendido, a piscar um olho, com óculos, sem óculos, luz frontal, luz à esquerda, luz à direita). Estas fotografias estão em formato GIF e com dimensões 320x243.

3.2 *Dataset* do Grupo

Este *dataset* é composto por 62 fotografias do grupo de trabalho com dimensões de 255x255 e em formato GIF, sendo 48 para treino e 14 para teste, onde todas foram tiradas sob um fundo branco e com uma boa iluminação. Tentou-se tirar as fotografias todas do mesmo ângulo para um melhor reconhecimento. À semelhança do *dataset* anterior, este conjunto de imagens contém diferentes expressões faciais, tais como normal, sorrir, piscar o olho, surpreso, sério, com óculos, triste, nervoso e olhos fechados, para que seja um conjunto com alguma variedade.

4 Implementação do modelo

Neste capítulo vamos apresentar o modelo desenvolvido em *Python* e começaremos por enumerar as diferentes funções do modelo:

- ***resizeImageAndConvert***
- ***readImages***
- ***pca***
- ***coefProj***
- ***testar***
- ***euclidian***
- ***mahalanobis***

Começando pela função ***resizeImageAndConvert***, recebe um conjunto de imagens e faz o seu tratamento, isto é, irá fazer o recorte de cada imagem e, posteriormente, irá redimensá-las para dimensões de 255x255. De seguida temos a função ***readImages*** que irá ao *path* definido pelo utilizar recolher as imagens do *dataset* passando para uma matriz as informações da cada imagem convertida para um tom monocromático, retornando a matriz e a quantidade de imagens do *dataset*. Passando para a função ***pca*** que, à semelhança do que foi descrito no capítulo anterior, vai receber a matriz com as informações do *dataset* (proveniente da *readImages*) (no caso do *dataset* do grupo, fornece-se também um número estático de componentes que se pretende usar) e um parâmetro de nível de confiança, calculando assim o número de componentes principais em relação ao espaço original. Esta função começa por calcular a média do *dataset* e utiliza essa média para centrar os dados, posteriormente obtém os valores e vectores próprios através do cálculo do SVD e realiza o armazenamento e ordenação desses valores obtidos. Posto isto, obtém as componentes principais, determina-se a quantidade de vectores a utilizar e, desses vectores, defini-se os que são associados, retornando uma lista ordenada dos k vectores próprios associados aos maiores valores próprios, uma lista ordenada dos k maiores valores próprios, os valores centrados na média, a média e a variância.

Feito o cálculo do PCA, realiza-se o cálculo dos coeficientes de projeção através da função ***coefProj*** que recebe a matriz com os dados centrados, retornada pela função *pca*, e realiza a projeção desses mesmos dados no espaço gerado pelas componentes principais, retornando os coeficientes numa lista. Passando agora para a função ***euclidian***, a sua essência reflete-se na receção de 2 elementos, x e y, e no cálculo da distância entre esses dois elementos através da fórmula da distância Euclidiana. Semelhante à anterior está a função ***mahalanobis*** que recebe dois elementos, x e y, a lista dos valores próprios e o k, de modo a utilizar os k maiores valores próprios para o cálculo da distância de *mahalanobis*.

Por último, a função ***testar*** que irá receber um *input*, isto é, uma imagem para realizar o teste de reconhecimento, e o tipo de distância que pretende utilizar fazer realizar o teste, bem como os outros parâmetros necessários (média, valores e vetores próprios, tamanho e coeficientes de projeção). Inicia-se o processo centrando esta imagem recebida pela média e projecta-se este resultado nas *eigenfaces*. De seguida, testa-se, através da distância

indicada (*mahalanobis* ou euclidiana), se este novo elemento pertence ou não ao conjunto de dados. Em caso afirmativo indica as distâncias máxima e mínima e o número da imagem que é compatível com o *input*.

5 Resultados obtidos

O modelo anteriormente descrito foi testado no *software Jupyter Notebook* e utilizando dois *dataset*: o que nos foi fornecido e o definido pelo grupo. Iniciou-se a fase de testes pelo conjunto de dados do *Yale Faces* e, após uma aperfeiçoamento dos resultados obtidos neste *dataset*, realizou-se os testes no conjunto de dados do grupo.

Decidiu-se por começar com uma avaliação do número de componentes principais (k), para isso foi necessário efetuar o seu cálculo. Para realizar o cálculo do k é necessário fornecer um valor do nível de confiança para a função ***pca***, ou seja, o k será obtido através do quociente entre o somatório dos valores próprios até k sobre o somatório de todos os valores próprios enquanto menor que o nível de confiança. Posto isto, criou-se tabelas para analisar a relação entre diferentes níveis de confiança e diferentes números de componentes principais.

Realizada a avaliação do número de componentes através da confiança, procurou-se obter o k tendo por base do gráfico do método do cotovelo e averiguar qual seria o melhor valor para o k.

5.1 *Yale Face Database B*

Do total do conjunto de dados (165 fotografias), optou-se por retirar **126 imagens para treino**, **30 para teste** e foi excluído o sujeito número 8, utilizando só apenas duas fotos do mesmo para teste.

Para realizar os testes, foram definidos os limites de 14500 para a distância Euclidiana e de 0.09 para a distância de *Mahalanobis*, com um pp(k)=10 a 0.8 de nível de confiança.

5.1.1 Número de componentes principais

Mediante o que foi referido previamente, foi realizado o cálculo das relações entre o k e a confiança, determinou-se os valores, ***True Positive, False Positive, True Negative*** e ***False Negative***, para as duas distâncias utilizadas: *mahalanobis* e euclidiana. Por intermédio destes valores, foi-nos possível obter os obter dados estatísticos como a **Precisão**(qualidade de cada previsão individual), ***Recall*** (proporção de respostas corretas) e **F1** (intervalo entre 0 e 1, quanto mais próximo de 1, melhor a previsão).

Fórmula *Precision*:

$$Precision = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{4}$$

Fórmula *Recall*:

$$Recall = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{5}$$

Fórmula F1:

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

- **Distância de *Mahalanobis***

Confiança	k	TP	FP	TN	FN	Precisão	Recall	F1
1	117	1	27	2	0	4%	100%	0.0779
0.95	39	13	15	2	0	46%	100%	0.6300
0.9	21	21	7	1	1	75%	95.40%	0.8397
0.85	14	21	7	0	2	75%	91.30%	0.8235
0.8	10	21	7	0	2	75%	91.30%	0.8235
0.75	8	19	9	0	2	67.80%	90.40%	0.7761
0.7	6	19	9	0	2	67.80%	90.40%	0.7761

Tabela 1: Resultados de teste com distância de *Mahalanobis* para o 1º *dataset*

Analisando a tabela desenvolvida anteriormente, é de notar que, por exemplo, no caso das confianças 0.75 e 0.85 utiliza-se 8 e 14 vetores próprios, respetivamente. Nestes dois casos não existe grande variação de valores como a precisão ou F1, logo

não foram afetados na mesma proporção que o valor de k, pois esses valores sofreram uma variação de quase o dobro.

- **Distância Euclidiana**

Confiança	k	TP	FP	TN	FN	Precisão	Recall	F1
1	117	22	6	0	2	78.57%	91.66%	0.8461
0.95	39	22	6	0	2	78.57%	91.66%	0.8461
0.9	21	22	8	0	2	73.33%	91.66%	0.8147
0.85	14	20	8	0	2	71.43%	90.90%	0.7999
0.8	10	20	8	0	2	71.43%	90.90%	0.7999
0.75	8	19	9	0	2	67.86%	90.48%	0.7755
0.7	6	19	9	0	2	67.86%	90.48%	0.7755

Tabela 2: Resultados de teste com distância Euclidiana para o 1º *dataset*

O caso da distância Euclidiana é semelhante à distância de *Mahalanobis*, permitindo retirar o mesmo tipo de conclusões.

5.1.2 *Elbow Method*

Outra forma de obter um valor para o k é o método do cotovelo. Assim sendo, através da análise do gráfico gerado, podemos concluir que o k sugerido é igual a 10 e vai de encontro com os melhores valores obtidos nas tabelas anteriores.

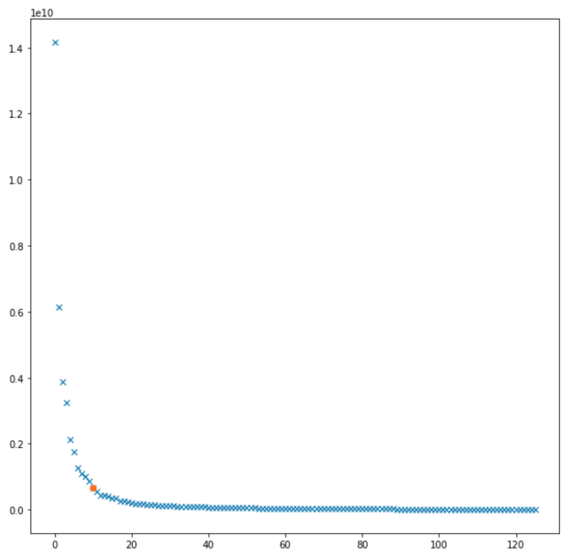


Figura 1: Método do cotovelo

5.2 Discussão de resultados

Analisando os testes realizados, é possível retirar algumas ilações para além do que já foi mencionado. Este modelo tem alguma dificuldade a reconhecer quando existe algum tipo de sombra, o que é um desfecho normal, em contrapartida, se a iluminação for boa, realiza bem o reconhecimento.

5.3 *Dataset* do Grupo

Para o *dataset* do grupo, optou-se por numa fase inicial calcular novos limites e, para tal, foi necessário calcular as distâncias máxima e mínima de cada tipo de distância para todos os casos de teste. Para efetuar estas medições, definiu-se os valores do nível de confiança e de k de **0.75** e **7**, respetivamente, onde se tentou que não fossem exageradamente altos de maneira a obter certa credibilidade nos resultados bem como não restringir em demasia o modelo. A partir destes cálculos foi possível construir a seguinte tabela.

k=5, confidence=0.75	**Euclidiana**		**Mahalanobis**	
Fotografia	Mínimo	Máximo	Mínimo	Máximo
joao_angry	1670.95	14705.62	0.0123	0.652
joao_closed_eyes	2188.95	16091.23	0.0291	0.5806
ricardo_oculos_serious	1120.75	12120.39	0.0055	0.44
ricardo_smile	630.23	15401.07	0.0014	0.5055
rafa_wink	1486.59	15355.22	0.0055	0.5001
rafa_oculos_serious	1995.04	15041.1	0.0145	0.7881
nascimento_oculos_smile	2059.45	14298.32	0.0111	0.7556
nascimento_serious	3024.48	14354.38	0.0387	0.603
bruno_closed_eyes	853.64	13294.13	0.0054	0.4399
bruno_wink	666.51	13442.79	0.0026	0.4794
daniel_wink_oculos	2971.87	13960.67	0.0438	0.6317
diana_closed_eyes_smile	2489.37	14675.88	0.0271	0.495
catarina_smile2	1709.1	11801.59	0.0152	0.4127
elisa_normal	4107.19	17707.72	0.0461	0.6649

Tabela 3: Cálculo das distâncias máximas e mínimas para os casos de teste

Como se pode verificar pela tabela anterior, o nosso limite para a distância Euclidiana vai situar-se nos 3200, uma vez que a fotografia ***nascimento_serious*** tem uma distância mínima de 3024.48, ou seja, o limite para esta distância tem de ser um pouco maior que o valor observado nesta fotografia de modo a cobrir todas as imagens. Já para a distância de *Mahalanobis*, o nosso limite fixou-se no 0.045, visto que a fotografia ***daniel_wink_oculos*** tem um valor de 0.0438 e o limite terá de ser ligeiramente maior. É importante referir que a fotografia ***elisa_normal*** apenas funciona para fixar valores que o limite não pode ultrapassar, pois é um sujeito que apenas dispõe desta fotografia, logo tem de falhar no reconhecimento.

5.3.1 Número de componentes principais

À semelhança do *dataset* anterior foi realizado uma análise de precisão, *recall* e F1 considerando os limites obtidos.

- **Distância de *Mahalanobis***

Confiança	**k**	**TP**	**FP**	**TN**	**FN**	**Precisão**	**Recall**	**F1**
1	47	0	0	1	13	Error	Error	Error
0.95	27	3	0	1	10	100%	23%	0.37
0.9	18	4	0	1	9	100%	30.7%	0.4697
0.85	13	6	0	1	7	100%	46.1%	0.63
0.8	9	10	0	1	3	100%	76.9%	0.869
0.75	7	13	0	1	0	100%	100%	1
0.7	6	13	1	0	0	93%	100%	0.963

Tabela 4: Resultados de teste com distância de *Mahalanobis* para o *dataset* do grupo

Verificamos que os valores determinados, para os limites com o k=7 e com o nível de confiança de 0.75, satisfazem muito positivamente, uma vez que foi possível atingir uma precisão de 100%, um *recall* de 100% e ainda um F1 de 1. É possível ainda dizer que para um nível de confiança de 0.7 e k=6, os valores também foram muito interessantes, tornando uma opção bastante viável.

- **Distância Euclidiana**

Confiança	k	TP	FP	TN	FN	Precisão	Recall	F1
1	47	4	0	1	9	100%	30.7%	0.4697
0.95	27	5	0	1	8	100%	38.5%	0.555
0.9	18	8	0	1	5	100%	61.5%	0.7616
0.85	13	9	0	1	4	100%	69%	0.8165
0.8	9	12	0	1	1	100%	92.3%	0.9599
0.75	7	13	0	1	0	100%	100%	1
0.7	6	13	0	1	0	93%	100%	1

Tabela 5: Resultados de teste com distância Euclidiana para o *dataset* do grupo

Em conformidade com o que aconteceu na distância de *Mahalanobis*, obteve-se valores iguais para este k e nivél de confiança, algo que já estávamos à espera.

5.3.2 *Elbow Method*

Observando o gráfico do *elbow method* constata-se que o valor de k sugerido é de 7.

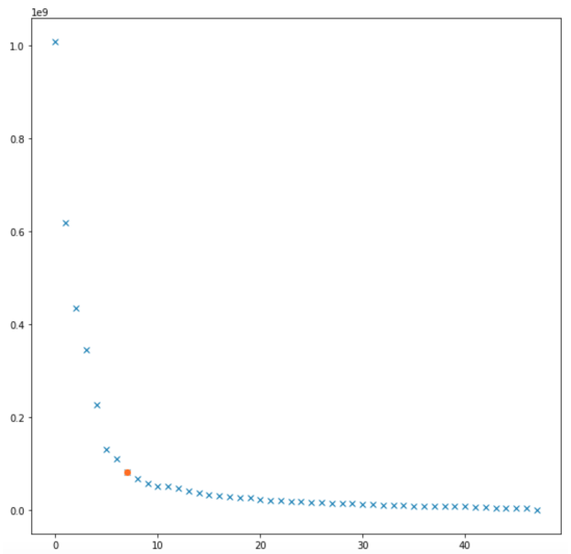


Figura 2: Método do cotovelo

5.4 Discussão de resultados

A partir das duas técnicas para avaliar o número de vetores e o nível de confiança, é plausível proferir que a melhor escolha para o k é 7, uma vez que se verifica nas duas técnicas como melhor opção permitindo fazer um excelente reconhecimento.

6 Conclusão

Tendo em conta os resultados obtidos e as dificuldades aos longo deste projeto, podemos inferir que o *Principal component analysis* (PCA) limita bastante o reconhecimento facial.

Primeiramente foi encontrado dificuldades tanto no *dataset Yale Faces*, quando apareciam fotografias de sujeitos com qualquer tipo de sombra, como no *dataset* do grupo, quando tentamos obter as fotografias, pois estas, para além de não poderem conter qualquer tipo de sombra, as imagens precisam de estar centradas, com um bom recorte, e praticamente na mesma posição. Queremos com isto dizer que, a junção de todas as imagens deve resultar numa imagem o mais homogénea possível, caso contrário os resultados ficarão mais fracos. Outra dificuldade encontrada pelo grupo foi o cálculo dos limites para as distâncias, uma vez que o método utilizado para esse cálculo foi através do cálculo das distâncias mínimas tanto da *Mahalanobis* como da Euclidiana dos casos de teste.

Em suma, o trabalho foi bem conseguido no qual o grupo obteve resultados satisfatórios, também fruto de vários testes antes de definir o modelo final. Quanto ao modelo desenvolvido, efetua um bom reconhecimento para os dados disponíveis, tornando-se um modelo bastante aceitável.

7 Referências

Referências

[1] PCA: Eigenvectors and Eigenvalues , https://towardsdatascience.com/pca-eigenvectors-and-eigenvalues-1f968bc6777a, Last accessed 11 Jan 2020

[2] Principal Component Analysis Tutorial , https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial, Last accessed 11 Jan 2020

[3] Principal Component Analysis in 3 Simple Steps , https://sebastianraschka.com/Articles/2015_pca_in_3_steps.html, Last accessed 11 Jan 2020

[4] 17.7.1 Principal Component Analysis , https://www.originlab.com/doc/Origin-Help/PrincipleComp-Analysis,Last accessed 11 Jan 2020

[5] PCA and SVD explained with numpy , https://towardsdatascience.com/pca-and-svd-explained-with-numpy-5d13b0d2a4d8, Last accessed 11 Jan 2020

[6] I Smith, Lindsay:A tutorial on Principal Components Analysis (2002) https://ourarchive.otago.ac.nz/bitstream/handle/10523/7534/OUCS-2002-12.pdf?sequence=1&isAllowed=y

[7] PCA and SVD explained with numpy , https://towardsdatascience.com/pca-and-svd-explained-with-numpy-5d13b0d2a4d8, Last accessed 11 Jan 2020

[8]] M. Turk, A. Pentland, Eigenfaces for Recognition, Journal of Cognitive Neurosicence, Vol. 3, No. 1, 1991, pp. 71-86,http://www.face-rec.org/algorithms/PCA/jcn.pdf

[9] The Yale Face Database, http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.html, normalizadas em http://vismod.media.mit.edu/vismod/classes/mas622-00/datasets/

[10] Ralph Gross, Face Databases, in S.Li and A.Jain, (ed). Handbook of Face Recognition. Springer-Verlag, 2005, http://ri.cmu.edu/pub_files/pub4/gross_ralph_2005_1/gross_ralph_2005_1.pdf

[11] http://www.face-rec.org/databases/