COMP4121 Project

Facial Recognition Algorithm

Zhengye MA
z5158505

Content:

1 Introduction

Face recognition system can be defined as a biometric technology based on human facial features. A person's face is as unique as a fingerprint. Building on the fact that no two faces are alike, face recognition systems can determine the identity of a face inside a picture or video. This technology began in the 1960s and was called "man-machine" at that time because the algorithm was at an initial stage of development. Back then, photo feature detail and location need to be set by the user before computer recognition. This technology has advanced by leaps and bounds since the 1990s. Nowadays, the algorithm has become highly accurate and ingenious. It is widely used in daily life such as at boarding gates, face ID, security access control, attendance machine, electronic passports and more. This essay will explore some basic and important algorithms applied during the face recognition process.

2 Workflow

Face recognition system comprises two major steps. The first step is face detection. As the name implies, this step works by locating all the faces inside a picture or video. The second step is recognition. This contains several layers. First layer is the alignment of crooked or inclined faces detected in an image. After that, the most important part is to extract features from the face. This step significantly influences the recognition result. Finally, the face will be compared to a set face for verification or a database of faces for identification purposes.

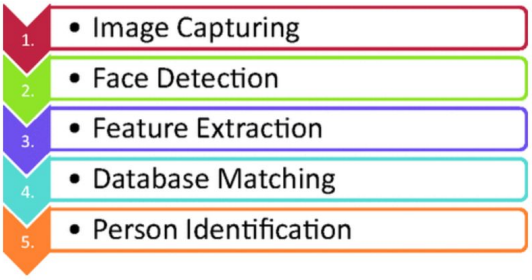
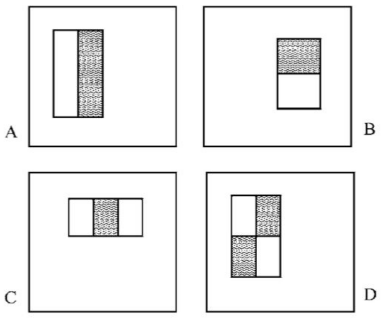


Figure 1. Workflow of a basic face-recognition system.

3 Face detection

The initial step of face recognition is detection, it can be implemented based ter than other type detectors. This section will describe the Viola-Jones detector which is an algorithm based on facial appearance. It is one of the most impactful face recognition algorithms invented in the 2000s. It contains three major components: Haar feature value, Adaboost and the Attentional cascade. Below is a detailed explanation.

Haar feature value can be defined as the sum of the pixels difference between different rectangles. Pixel is the smallest component of an image. The Haar feature has four basic types: edge, line, centre and diagonal. Viola and Jones use three types of the Haar feature but four shapes (A, B, C, D). Haar feature value for C shape is total outside white rectangles minus black rectangle. Haar feature value for D shape is diagonal white rectangle pairs minus diagonal black rectangle pairs.



applied to picture

When a Haar feature is applied to a picture it can make outstanding

reflection of local characteristics such as nose, mouth, eyebrow etc. This is because Haar feature value clearly shows the colour difference, for example the ala nose colour will be brighter whereas the nose is relatively dark. By changing the position, shape and size of the Haar feature, this will result in a large amount of Haar feature value calculation. Therefore, if the resolution of the picture is 24X24, it will be a disaster. Hence, Viola and Jones came up with Integral Image.

Integral Image can be referred to as a convenient algorithm for computing the value of the rectangle in a grid. Viola and Jones state that:

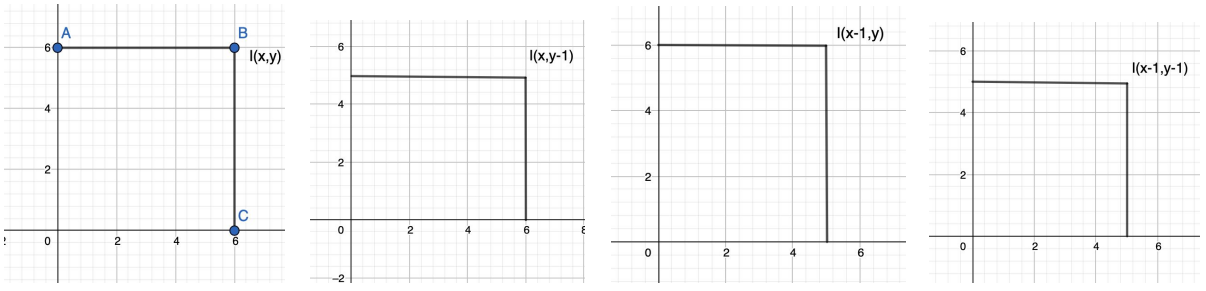
$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y'),$$

It shows that the pixel of the point (x, y) is the sum of all point, using the following pair of recurrences:

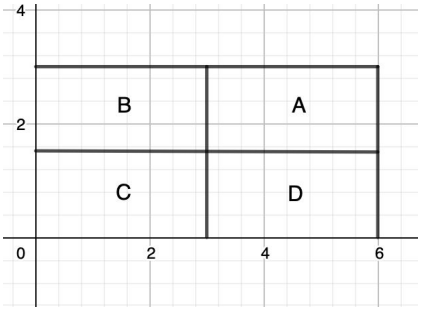
$$s(x, y) = s(x, y - 1) + i(x, y) \qquad (1)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \qquad (2)$$

A better way to understand it is:



the I node in each picture represents the corresponding point pixel value. Based on these images, the integral image value of the grid I (x, y) can be written as ii (x, y) = ii (x-1, y) + ii (x, y-1) + I (x, y) - ii (x-1, y-1). Then to calculate sum of the following image only need four lookups:



ii (ABCD) = ii(A) + ii(C) - (ii(B) + ii(D)) and it can generalise to all Haar feature rectangle value, moreover, it is not hard to find that it is also the solution of shape D Haar feature value.

In order to form an effective classifier which identifies feature based on whether it is useful or not, Viola and Jones indicate that the AdaBoost learning algorithm is used to boost the classification performance for classifier algorithm (base algorithm):

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$
  2. Select the best weak classifier with respect to the weighted error
     $$\epsilon_t = \min_{f,p,\theta} \sum_i w_i \, |h(x_i, f, p, \theta) - y_i| \, .$$
     See Section 3.1 for a discussion of an efficient implementation.
  3. Define $h_t(x) = h(x, f_t, p_t, \theta_t)$ where f_t, p_t, and θ_t are the minimizers of ϵ_t.
  4. Update the weights:
     $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
     where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:

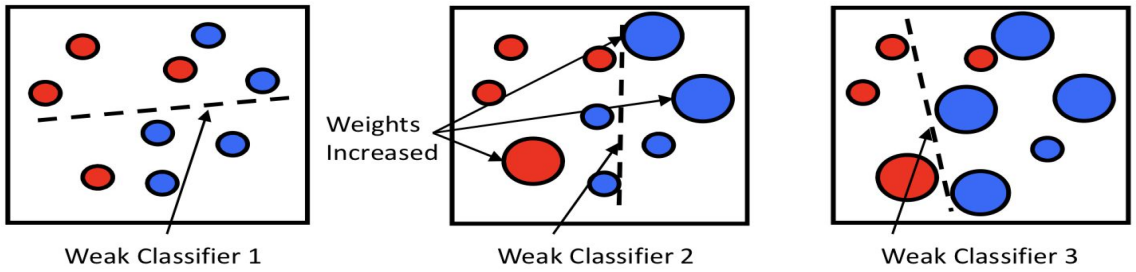
$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

The pseudo code by Viola and Jones

The overview of AdaBoost algorithm is that repeat training the base algorithm which produces weak classifiers to become an efficient algorithm which produces a better classifier until obtaining a strong classifier. Before the boosting starts, every individual sample has the

same weight, then if a weak classifier fails to distinguish some of samples, these samples weight will correspondingly increase. It influences the next weak classifier to focus on solving these failed sample as these samples have higher weight:



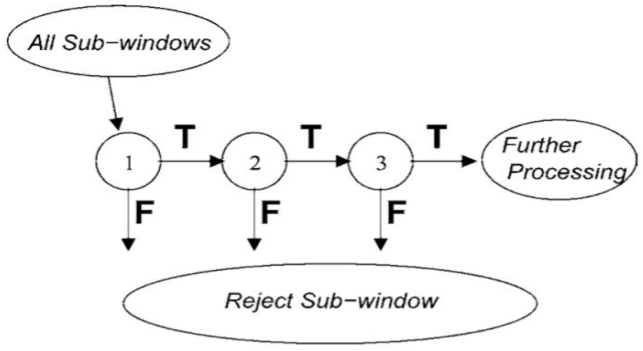
From the picture, it is not hard to see that the performance of the classifier is getting better.
The weak classifier is set as:

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases}$$

The weak feature classifier

Θ is the threshold such as minimum number of examples are misclassified, f(x) is a feature, p is polarity.

For the sake of increasing classifier performance and reducing computation time, a cascade classifier is raised by Viola and Jones:



The cascade classifier is made up of several strong classifiers that are arranged from relatively weak to strong. If a classifier has 99% detect rate and 30% false rate, cascade ten classifiers will get 99%^10=90% detect rate and very minor false rate. During the face detection, if the sub window fails to pass through the classifier, it will never

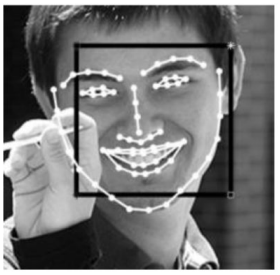
process to the next classifier, in other words, it is discarded immediately. If an image passes all the classifiers then it seems like a face object. The cascade classifier needs vast amounts of training because testing facial features is a rare task, to build a highly efficient face detector, for training purpose Viola and Jones set a threshold at every classifier. Each classifier will scan the sample to find more unrejected non-face which will be stored for the training of the next classifier.

4 Face recognition

The purpose of face recognition is to further confirm the face frame that has been detected. It contains several steps as mentioned earlier, strictly speaking there is a step called face alignment before face feature extraction, then based on different goals choose face verification or face identification. This section will introduce various algorithms for these corresponding steps. Supervised Descent Method (SDM) for face alignment, Principal Component Analysis (PCA) for face feature extraction and Joint Bayesian/Cosine similarity applied to face verification/identification.

4.1 Face alignment

Before exploring the face alignment method, it is necessary to introduce a basic tool for achieving face alignment called face landmark. Face landmarks can be defined as facial feature points, it is located around characteristic position on the face such as eyes, mouth, nose etc:



Detected face and 68 facial landmarks

The process of locating landmarks can be referred to as face alignment. It often requires a large amount of training for the same face. The Supervised Descent Method (SDM) is used to help landmarks that locate to the relative precise position of feature contour. Its essence is to solve least square problem (curve fitting). In professional terms, this method learns the direction of gradient descent from training data and establishes a corresponding regression model. Then, this model is used to estimate the direction of the gradient.

Newton's method is a common method used to solve least square problem but under high dimensions, it needs to solve the Hessian matrix which is hard. SDM intelligently avoids it:
Initial landmark's coordinate is x_0, face alignment can be framed as minimizing the following function over Δx:

$$f(\mathbf{x}_0 + \Delta\mathbf{x}) = \|h(\mathbf{x}_0 + \Delta\mathbf{x}, I) - h(\mathbf{x}*, I)\|_2^2$$

h (x, l) is the feature values around the landmark location x of image I, x* is the ground-truth value of landmarks, apply Newton's law and with second-order Taylor expansion above equation transformed to:

$$f(\mathbf{x}_0 + \Delta\mathbf{x}) \approx f(\mathbf{x}_0) + \mathbf{J}_f(\mathbf{x}_0)^T \Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T \mathbf{H}_f(\mathbf{x}_0)\Delta\mathbf{x}$$

Then differentiating this equation by Δx:

$$\begin{aligned}\Delta\mathbf{x} &= -\mathbf{H}_f(\mathbf{x}_0)^{-1}\mathbf{J}_f(\mathbf{x}_0) \\ &= -2\mathbf{H}_f(\mathbf{x}_0)^{-1}\mathbf{J}_h^{\mathrm{T}}(\mathbf{x}_0)\left(h(\mathbf{x}_0, I) - h(\mathbf{x}*, I)\right) \\ &= -2\mathbf{H}_f(\mathbf{x}_0)^{-1}\mathbf{J}_h^{\mathrm{T}}(\mathbf{x}_0)\, h(\mathbf{x}_0, I) + 2\mathbf{H}_f(\mathbf{x}_0)^{-1}\mathbf{J}_h^{\mathrm{T}}(\mathbf{x}_0)\, h(\mathbf{x}*, I)\end{aligned}$$

From above Jacobian and Hessian could be calculated approximately.

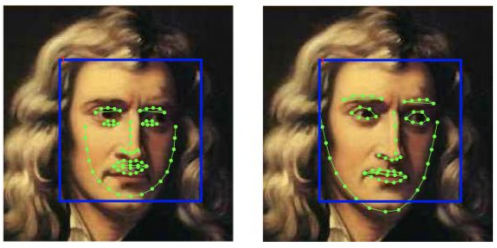
But it is still difficult to compute, SDM uses an identical pair R and b to represent above formula: $-2\mathbf{H}_f^{-1}\mathbf{J}\mathbf{T}h$ and $2\mathbf{H}_f^{-1}\mathbf{J}\mathbf{T}hh(\mathbf{x}_*, I)$, the name of the R and b is descended direction it can be learned to minimizing:

$$\sum_{i=1}^{N} \left\| \Delta \mathbf{x}_*^i - \mathbf{R}h\left(x_0^i, I_i\right) - \mathbf{b} \right\|_2^2$$

Which is a well-known linear least square problem. N is the number of training images and $\Delta x_*^i = x_*^i - x_0^i$, for every single training, the descend direction will be stored in {R_k} and {b_k}. Then for a new image, the shape can be update by:

$$\Delta \mathbf{x}_k = \mathbf{R}_k h\left(\mathbf{x}_{k-1}, I\right) + \mathbf{b}_k$$

Below is the final output example:



4.2 Face feature extraction

Principle Component Analysis method (PCA) is a common method used for face feature extraction, the core idea of PCA is to reduce the dimensions of the data and remove redundant information. Every facial image can be represented as the weighted sum of eigenface. Eigenface is formed by eigenvectors of the covariance matrix of the training images. Below is the detail process for how to build up eigenface by PCA method.

Take M vectors of size N = rows of image x columns of image, represent a set of images, p_ j represent the pixel value. Image vector can be represented as:

$$x_{\mathrm{i}} = \left[p_1, \cdots, p_{\mathrm{N}}\right]^{\mathrm{T}}, i = 1, \cdots, \mathbf{M}$$

Let m be the mean image:

$$m = \frac{1}{\mathbf{M}} \sum_{i=1}^{M} x_1$$

Then image vector needs to be decentralised means subtract mean value:

$$w_i = x_i - m$$

To find the group of e_i which have the biggest potential projection onto w_i, it requires M orthonormal vectors e_i for which the quantity is:

$$\lambda_i \frac{1}{\mathbf{M}} \sum_{n=1}^{M} (e_i^T w_n)^2 \qquad e_i^T e_k = \delta_{lk}$$

note: δ is unit matrix

After that, e_i and λ_i corresponding to eigenvectors and eigenvalues can be calculated from covariance matrix:

$$C = \mathbf{W}\mathbf{W}^T$$

And W matrix has following relationship:

$$\mathbf{W}^T \mathbf{W} d_i = \mu_i d_i$$

more explanation: Cd = ud calculate the eigenvalues of the covariance matrix C and the corresponding eigenvectors.

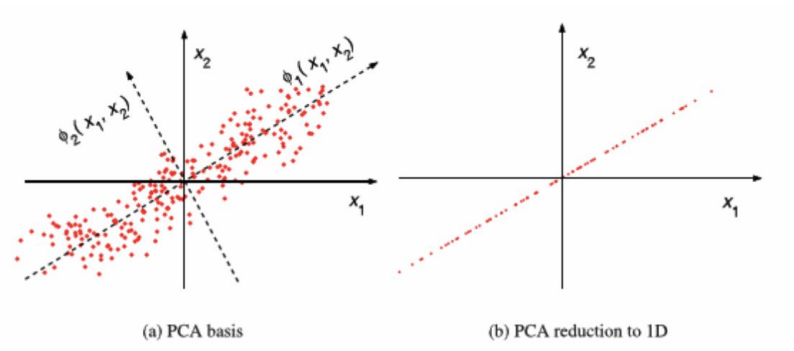
Finally, the facial image can be represented as:

$$\Omega = [v_1, v_2, v_3, \cdots, v_m]^T$$

Here are some important tips:

The reason for using covariance instead of variance is to avoid the biggest overlap projection in high dimension. The multiplication of two matrices is the transformation of the right-hand side column vector into the space represented by the left-hand side matrix. These mentioned points help PCA to reduce the dimension without losing important data information.

This is a two dimension reduction to one dimension example of PCA, the dots are selected at regular space, dash lines are the new basis which is orthogonality under normal circumstances.



(a) PCA basis (b) PCA reduction to 1D

4.3 Face verification

Verification is a one compare by one process. After feature extraction, a measurement method should be used to calculate the distance of different features. The commonly used method is Joint-Bayesian. It contains two parts naive formulation and joint formulation, the Joint-Bayesian method is based on Bayesian face formulation which is also used for face verification. Joint Bayesian is an improvement for Bayesian. Bayesian face formulation can be represented as following:
Suppose there are two faces x1 and x2, H_I represent the same face, H_E represent different faces. Then the verification problem transformed to classify the difference of Δx = x1-x2, apply Bayesian formula:

$$P(H_I|\Delta) = \frac{P(\Delta|H_I) * P(H_I)}{P(\Delta)}$$

$$P(H_D|\Delta) = \frac{P(\Delta|H_D) * P(H_D)}{P(\Delta)}$$

After that, based on Maximum a Posterior rule, it can transform to compare Posterior value which involves likelihood ratio r (x1, x2):

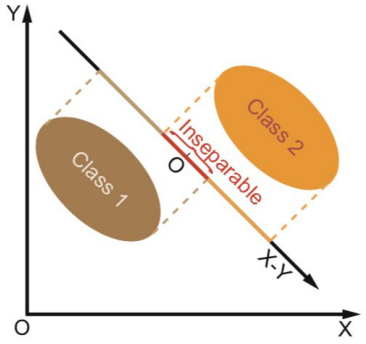
$$\frac{P(H_I|\Delta)}{P(H_D|\Delta)} = \frac{P(\Delta|H_I)}{P(\Delta|H_D)}$$

divide above two equation

$$r(x_1, x_2) = \log \frac{P(\Delta|H_I)}{P(\Delta|H_D)}$$

In the end, if r > 0, then P (H_I| Δ) > P (H_E | Δ), it can seem like the same face.

But there is an issue that Bayesian face method will decrease the separability of discriminative information, consider if the value of P (Δ | H_I) and P (Δ | H_E) are very similar:



The projection of class 1 and class 2 are hard to discriminate. Joint Bayesian directly joins the two faces {x1, x2} and based on this build up model. Then the problem similarly transforms to above process, the naive function is as:

$$r(x_1, x_2) = \log \frac{P(x_1, x_2|H_I)}{P(x_1, x_2|H_D)}$$

Set P(x1,x2|H_I) and P(x1,x2|H_D) as a Gaussian:

$$P(x_1, x_2|H_I) = N(0, \Sigma_I)$$

$$P(x_1, x_2|H_E) = N(0, \Sigma_E)$$

Covariance matrix Σ_I and Σ_E can be estimated from data training. If the covariance matrix is high dimension it will be difficult to estimate since the training will be less sufficient. To solve this problem, Joint function is applied:

$$x = \mu + \epsilon$$

Face can be written as x and u is the identity, ε is the variable for lighting, pose etc. u and ε are Gaussian then:

$$\mu = N(0, S_\mu)$$

$$\epsilon = N(0, S_\epsilon)$$

Covariance of two faces is:

$$\mathbf{cov}(x_i, x_j) = \mathbf{cov}(\mu_i, \mu_j) + \mathbf{cov}(\varepsilon_i, \varepsilon_j), \quad i, j \in \{1, 2\}.$$

Under H_I hypothesis u1 and u2 identity is same hence the P(x1,x2|H_I) can be derived as:

$$\Sigma_I = \begin{bmatrix} S_\mu + S_\varepsilon & S_\mu \\ S_\mu & S_\mu + S_\varepsilon \end{bmatrix}$$

Under H_E hypothesis u and ε are different hence the P(x1,x2|H_E) can be derived as:

$$\Sigma_E = \begin{bmatrix} S_\mu + S_\varepsilon & 0 \\ 0 & S_\mu + S_\varepsilon \end{bmatrix}$$

The final likelihood ratio is:

$$r(x_1, x_2) = \log \frac{P(x_1, x_2|H_I)}{P(x_1, x_2|H_E)} = x_1^T A x_1 + x_2^T A x_2 - 2 x_1^T G x_2$$

Note that:

$$A = (S_\mu + S_\varepsilon)^{-1} - (F + G),$$

$$\begin{pmatrix} F+G & G \\ G & F+G \end{pmatrix} = \begin{pmatrix} S_\mu + S_\varepsilon & S_\mu \\ S_\mu & S_\mu + S_\varepsilon \end{pmatrix}^{-1}$$

Then if S_u and S_ε can be calculated, the likelihood ratio can be known. This process involves an EM step (Model Learning).

E-step:

Assume one person have m images, u will not change, but ε will change, so latent variables h $[\mu; \epsilon_1; \ldots; \epsilon_m]$ and x $[x_1; x_2; \ldots; x_m]$ relationship will be:

$$\mathbf{x} = \mathbf{P}\mathbf{h}, \quad \text{where } \mathbf{P} = \begin{bmatrix} \mathbf{I} & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I} \end{bmatrix}$$

Given the observation x, the expectation value of h will be:

$$E(\mathbf{h}|\mathbf{x}) = \Sigma_h \mathbf{P}^T \Sigma_x^{-1} \mathbf{x}.$$

Combination above formulas, u and ε can be derived as:

$$\mu = \sum_{i=1}^{m} S_\mu (F + mG) x_i$$

$$\epsilon_j = x_j + \sum_{i=1}^{m} S_\epsilon G x_j$$

M-step:

This step can simply estimate out S_u and S_ ε value, the reason is that one person has a certain u value and many ε values, if there are many persons, the amount of u and ε will be sufficient.

$$S_\mu = \mathbf{cov}(\mu)$$
$$S_\varepsilon = \mathbf{cov}(\varepsilon)$$

Finally, the value of likelihood ratio can be evaluated. Based on the ratio, one can make out the conclusion whether it is the same face.

4.4 Face identification

Identification is 1 compared to the N process. Cosine similarity is a method used to measure the cosine value difference between two vectors which is called similarity value. The cosine similarity between x and y vectors can be written as:

$$CS(x, y) = \frac{x^T y}{\|x\| \, \|y\|}$$

The similarity value is checked by a threshold to determine whether two faces are the same or not. Once there is a similarity value that passes the threshold, the system identifies these two specific faces are the same.

5 Conclusion

This essay has described components of face recognition system with corresponding algorithms.

Viola and Jones face detector is efficient for feature selection and computation, in addition, it can also be used to detect other objects such as vehicles, buildings. SDM overcomes the computation difficulty of Jacobians and Hessians. In addition, it supervises the descent directions and based on this for prediction learning.

PCA attempts to retain as much of the main information of the original data as possible after reducing dimension, at the same time it abandons redundant information.

Joint-Bayesian is a development of Bayesian, this algorithm provides better separability between different face classes by modeling the two different classes together. Cosine similarity is different from Euclidean, Euclidean reflects the absolute distance, while Cosine similarity is more focused on the direction difference which is a suitable method to distinguish with many features.

In recent years, thanks to algorithm innovation, Convolutional Neural Network (CNN) is used for face recognition, it is a deep learning method that is more powerful.

6 Reference

Zhang, C., & Zhang, Zhengyou. (2010). Boosting-based face detection and adaptation. San Rafael, Calif.: Morgan & Claypool.

Li, Stan Z, & Jain, Anil K. (2005). Handbook of Face Recognition. In Handbook of Face Recognition. New York, NY: Springer New York.

Schapire, R. E., & Freund, Y. (2012). Boosting : foundations and algorithms. Cambridge, Mass.: MIT Press.

Viola, Paul, & Jones, Michael J. (2004). Robust Real-Time Face Detection. International Journal of Computer Vision, 57(2), 137–154. https://doi.org/10.1023/b:visi.0000013087.49260.fb

Kaur, Paramjit, Krishan, Kewal, Sharma, Suresh K, & Kanchan, Tanuj. (2020). Facial-recognition algorithms: A literature review. Medicine, Science and the Law, 60(2), 131–139. https://doi.org/10.1177/0025802419893168

Y Vijaya Lata, Chandra Kiran Bharadwaj Tungathurthi, H Ram Mohan Rao, A Govardhan, & L P Reddy. (2009). Facial Recognition using Eigenfaces by PCA. International Journal of Recent Trends in Engineering, 1(1), 587.

Turk, M.A, & Pentland, A.P. (1991). Face recognition using eigenfaces. Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 586–591. IEEE Comput. Soc. Press. https://doi.org/10.1109/CVPR.1991.139758

Sharma, R, & Patterh, M. S. (2015). A broad review about face recognition - feature extraction and recognition techniques. The Imaging Science Journal, 63(7), 361–377. https://doi.org/10.1179/1743131X14Y.0000000071

Wang, C. (2019). The Development and Challenges of Face Alignment Algorithms. Journal of Physics. Conference Series, 1335, 12009. https://doi.org/10.1088/1742-6596/1335/1/012009

Lou, Jianwen, Cai, Xiaoxu, Wang, Yiming, Yu, Hui, & Canavan, Shaun. (2019). Multi-subspace supervised descent method for robust face alignment.

Multimedia Tools and Applications, 78(24), 35455–35469. https://doi.org/10.1007/s11042-019-08129-4

Xuehan Xiong, & De la Torre, Fernando. (2013). Supervised Descent Method and Its Applications to Face Alignment. 2013 IEEE Conference on Computer Vision and Pattern Recognition, 532–539. IEEE. https://doi.org/10.1109/CVPR.2013.75

Fitzgibbon, Andrew, Lazebnik, Svetlana, Perona, Pietro, Sato, Yoichi, & Schmid, Cordelia. (2012). Computer Vision - ECCV 2012. In Computer Vision - ECCV 2012 Berlin, Heidelberg: Springer Berlin / Heidelberg.

Kimmel, Ron, Klette, Reinhard, & Sugimoto, Akihiro. (2011). Computer Vision - ACCV 2010. In Computer Vision - ACCV 2010 Berlin, Heidelberg: Springer Berlin / Heidelberg.