How adding L1 and L2 Regularization terms effects the performance of L2 Orthonormal Face Recognition ?

Amit H Patel
The University of Adelaide
a1697700@student.adelaide.edu.au

Abstract

In this research, regularization is focused to manipulate the face recognition method in a way, we can avoid over-fitting and under-fitting the data.From research[1], the orthonormal l2 face recognition method is used to add the hyper-parameterλ with l1 and l2 norm. Before, understanding effect of lambda value, concept of l1 norm, l2 norm, linear regression, regularization and general effects of hyper-parameter on alpha is briefed out. After that the results from different lambda are compared with the accuracy of L1 and L2 norm regularization with l2 orthonormal face recognition[1] method. For L2 regularized algorithm, it is solved with closed form solution while, for L1 regularized algorithm optimization solution is used. The effect of different lambda values are examined and noticed that increasing lambda results in declining the accuracy result but in different pace for l1 and l2 norm regularized algorithm.

1. Introduction

In computer vision, face recognition is most concerned topic. To achieve the goal of assignment, there is YaleB dataset is used to examine effects of lambda over l1 and l2 regularized l2 orthonormal algorithm[1]. Before, showing analytic main methods, ideas are briefed out as follow:

2. Briefing on L1 norm and L2 norm

L1 norm: L1 distance(Manhattan) which takes the sum of the absolute values between the pixels. It forms the square shape around the origin. it depends on the choice of coordinate system. Rotating the frame will change the distance L1-norm has the property of producing many coefficients with zero values or very small values with few large coefficients

$$||\alpha||_1 = \sum_i |\alpha_i| \tag{1}$$

L2 Norm: L2 Norm is where we take the square root of the sum of the squares. Rotating the coordinate frame in l2 distance does not change the distance.L2 is more used if the the features are more generic vector then l2 is more useful.

$$||\alpha||_2 = \sqrt{\sum_i \alpha_i^2} \tag{2}$$

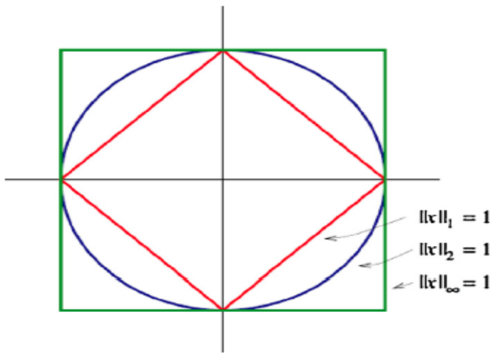


Fig. 1. Illustrations of the 1-, 2- and ∞-norms.

As above image illustrates, l1 norm forms square shape in the space. while, l2 norm forms circle on the space. specifying simple different distance metrics for data(like in l1 and l2 norm), helps to generalize the weights(λ) to run on any data.

In L1 the likelihood to converge/hit the corners are greater than the likelihood in the L2, so L1 penalizes the coefficients more than L2 which leads to sparsity.l1 encourage the alpha to be very sparse, most of them are 0 just.

3. The Orthonormal L2 norm Face Recognition Method

The method includes total K different subjects, where n_k is a number of training images for Kth subject.Thus, We have $n = \sum_k^K = 1 n_k$. Here, all raw images of human faces expressed as columns of A are clustered with very small variance (Courtesy of John Wright [2]). $A = [x_1, x_2 ... x_n] \in |R^m * n$. Here, n is the nth number of face image in single column vector and m is number of pixels in any image. Assuming, testing image x is vector with linear combination of matrix A(train image matrix). In following formula α is a linear combination.

$$x = A\alpha \tag{3}$$

3.1. Understanding of Linear Regression

It is a statistical measure that determine the strength of relation in between one dependent variable with another changing variable. For image data-set, running linear regress we can see A metrics in it which flatten the image: transfer image to vectors and all bunch of images putting together call A=[x1,x2,...xn]n can be presented with weighted sum of all faces. If everyone faces are totally different that one person's face is weighted with the sum of his/her face, can not defined by others average. As in sparse coding, we can formulate the following equation:

$$\alpha = |x|_{l1} s.t. x = A * \alpha \tag{4}$$

Testing process for two images: Taking images in the form of metrics. we can substract on image metrix from another and the result gives the difference between those two images. This process can be understood as a basics of equation(5), where minimising the difference by changing α = training image(in metrics) - weighted face image(in metrics test image with alpha as a weight) try to minimize this difference and that way to choose the alpha, which can be learnt through the training set and use that alpha for testing image.

3.2. L2 Orthonormal Algorithm

By minimizing α in following euation, the orthonormal L2 norm algorithm tries to minimize error function as describe in equation(2).

$$\alpha_x = argmin\alpha ||x - A.\alpha||_{l2} \tag{5}$$

the problem can be solved with closed form solution and we can get optimal solution for α as,

$$\alpha_x = (A^T A)^{-1} A^T x \tag{6}$$

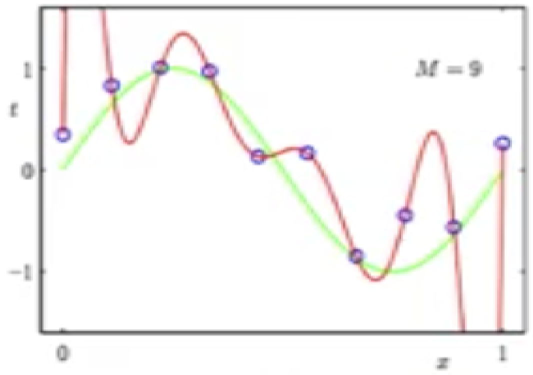
By calculating the α as above and using that in equation(2) and then calculate the sum of squares of differences help to recognize the which face is the similar(in other words, has less errors)

4. Why Regularization

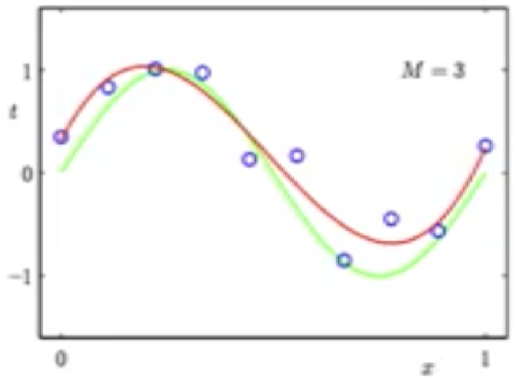
Running face recognition L1 and L2 orthonormal algorithm wihout regularization from [1] on YaleB gives us following accuracy. l2 orthonormal algorithm give 98.91+-1.38% [1] Using l1 norm algorithm give 96.63+-3.03% [1]

over-fitting is a problem you see when your machine learning model is too large (has too many parameters) comparing to your available training data. In this case, the model tends to remember all training cases including noisy to achieve better training score. To avoid this, regularization is applied to the model to (essentially) reduce its size. One way of regularization is making sure the trained model is sparse so that the majority of it's components are zeros. Those zeros are essentially useless, and your model size is in fact reduced.

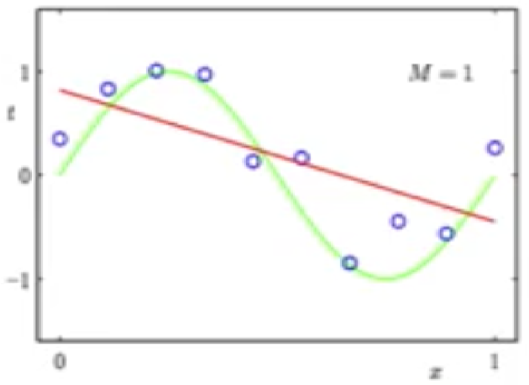
We use the training data to find some classifiers. Suppose we have some data sets of blue points and we are going to fit some curve to training data(Red points) then only thing we have told our classifier to do is to train and fit training data that might have very wiggly curve to perfectly classify all the training data points.



However, it is not useful as this performance of training data set is not cared but we care about the testing data set which can be again different points between the space. Thus, that wiggly Red line is going to be totally wrong. Thus, smooth line almost parallel to green line can be used as a optimal solution.



In fact, predicting the straight Red line would be more preferred to classify the test data instead of complexity wiggly line what fits all data.Although, it might be underfit the data.

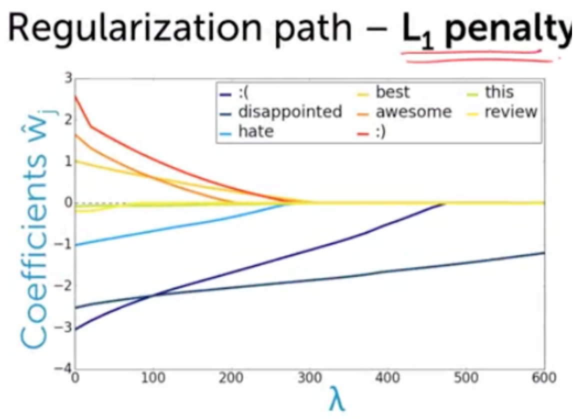


The model should be simple. so, it works on test data.[occam's razor Among competing hypothesis "The simplest is the better"]. This means generalising the future observation.This can be achieved through adding a regularisation function. Here,regularization hyper-parameter lambda trades of between the data loss.It is more important while training the models in practise.

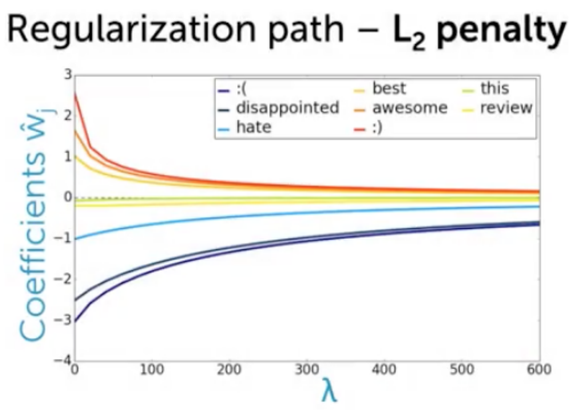
4.1. Why L2 norm gives a good result

L1-norm does not have an analytical solution, but L2-norm does. This allows the L2-norm solutions to be calculated computationally efficiently. However, L1-norm solutions does have the sparsity properties which allows it to be used along with sparse algorithms, which makes the calculation more computationally efficient.

4.2. Effect on alphaα(weights) with changing lambdaλ(Hyper-parameter) for L1 and L2 regularization



From graph chart it is seen that increasing the hyper-parameter with l1 norm regularization makes most features(α) to become 0.



From above graph it is seen that increasing the hyper-parameter with l2 norm regularization makes most features(α) value to be very small but not 0.

5. L1 norm regularization with L2 Orthonormal

To recognise faces with more accuracy on L2 Orthonormal algorithm, We need to get sparse vector from non-sparse data. The dataset YaleB have the label on the faces, representing 1,2nd,3rd person and so on as labels, equation(5) tries to learn the faces according to their label to compare with unlabeled faces and gives good alpha for choosing weight.

After calculating edges from a image, we can see sparse, many pixels will be 0 and others different. there are some sparse elements to identify an image instead of using whole image metrics, those sparse pixels are weighted and can run l1 norm regression on it. learning process includes two steps, training the α and using on testing image with following formula,

$$argmin_{\alpha}||x - A\alpha||_2^2 + \lambda||\alpha||_1 \qquad (7)$$

where, x is each image,A_i are bases(all patches), alpha different for each patches. and λ is the Lagrangian Multiplier. lambda with l1 sparsity term of $||\alpha||_1$, for each patch we want to use k weights and with k weights, we try to minimize the alpha with the sum of absolute value, it is hyper-parameter(helps to control strength of sparsity),

Here, lambda = 0 means No regularization(standard solution) lambda = infinity all weight on regularization = all α=0 lambda in between some α = not 0 and some are o

Image l1 norm penalty : disappointed is the not the high value then other coefficients to stay longer but it the longest value that stays negative respect to increase in lambda For very small lambda we have over-fitted boundary, while with larger lambda we have nice and smoother boundary.

Each of the features in image are been weighted and becomes y so,a images with weighted features are collected in y(y=x1alpha1, x2alpha2...n) Suppose, we do not want all of the features from the given image. So, we can weight the features such a way that we can ignore the some useless features by putting the alpha as 0

Also, equation(16) can be written as a following form as it is sum of absolute values of the given vector:

$$argmin\alpha||x - A\alpha||_2^2 + \lambda \sum_{l=1}^{n} |\alpha_i| \tag{8}$$

5.1. Solution:

Solving the above optimisation problem as a quadratic programming problem by gradient projection sparse representation(GPSR)[3] First step is to seperate the positive and negative coefficients α_+ and α_-

$$min\alpha Q(\alpha) = ||x - [A, -A][\alpha_+; \alpha_-]||_2^2 + \lambda 1^T(\alpha_+ + \alpha_-) \tag{9}$$

Subj to. $\alpha_- \geq 0, \alpha_+ \geq 0$

In standard QP form

$$argminQ(z) = C^T z + z^T Bz \tag{10}$$

s.t. z ≥ 0
Where,

$$z = [\alpha_+; \alpha_-]$$
$$C = \lambda 1^T(\alpha_+; \alpha_-)$$

and

$$\text{B} = \begin{bmatrix} A^T A & -A^T A \\ -A^T A & A^T A \end{bmatrix}$$

gradient of Q(z) is defined as,

$$\nabla_z Q(z) = C + Bz \tag{11}$$

Searching each iterate $z^{(k)}$ along the negative gradient $-\nabla_z Q(z) = C + Bz$ using steepest descent algorithm.

$$z^{(k+1)} = z^{(k)} - \phi^{(k)} \nabla Q(z^{(k)}) \tag{12}$$

$\phi^{(k)}$ is the step size.

Using a standard line-process[28], we can define direction vector $g^{(k)}$ as,

$$g_i^{(k+1)} = \begin{cases} (\nabla(z^{(k)}))_i, & \text{if } z_i^{(k)} > 0 \text{ or} \\ ((z^{(k)})) & \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

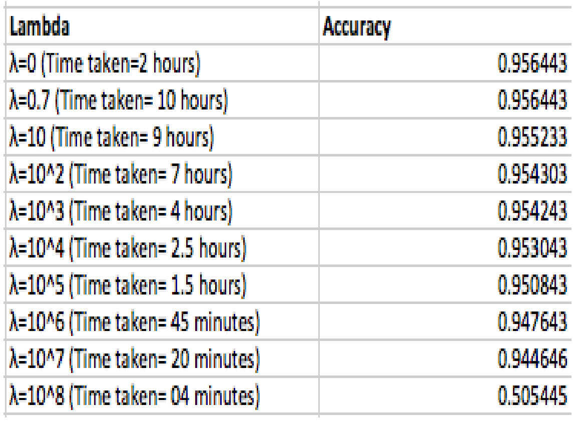
Step size for update can be choosen as

$$\phi^{(k)} = argmin\phi Q(z^{(k)} - \phi g^{(k)}) \tag{14}$$

For above problem, we have following closed form solution,

$$\phi^{(k)} = \frac{(g^{(k)T} g^{(k)})}{(g^{(k)T} B g^{(k)})} \tag{15}$$

5.2. Accuracy on Changing hyper-perameter λ



Lambda	Accuracy
λ=0 (Time taken=2 hours)	0.956443
λ=0.7 (Time taken= 10 hours)	0.956443
λ=10 (Time taken= 9 hours)	0.955233
λ=10^2 (Time taken= 7 hours)	0.954303
λ=10^3 (Time taken= 4 hours)	0.954243
λ=10^4 (Time taken= 2.5 hours)	0.953043
λ=10^5 (Time taken= 1.5 hours)	0.950843
λ=10^6 (Time taken= 45 minutes)	0.947643
λ=10^7 (Time taken= 20 minutes)	0.944646
λ=10^8 (Time taken= 04 minutes)	0.505445

6. L2 norm regularization with L2 Orthonormal

$$||x - A\alpha||_{l2}^2 + \lambda||\alpha||_{l2}^2 \tag{16}$$

where, x is each image,A_i are bases(all patches), alpha different for each patches.
and λ is the Lagrangian Multiplier. lambda with l1 sparsity term of $||\alpha||_1$, for each patch we want to use k weights and with k weights, we try to minimize the alpha with the squar root of sum of the given value, it is hyper-parameter,

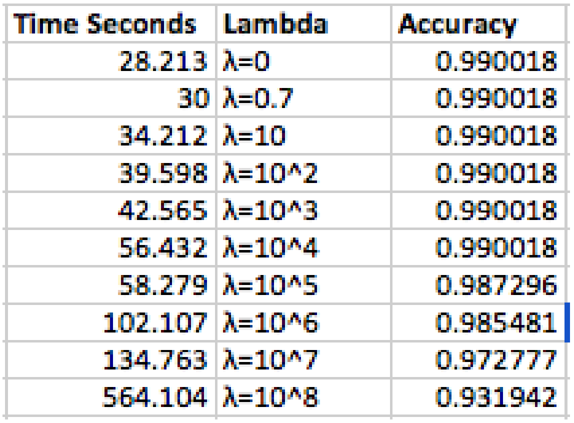
6.1. The closed form Solution:

$$\nabla_\alpha(||x - A\alpha||_{l2}^2 + \lambda||\alpha||_{l2}^2) = 0 \tag{17}$$
$$\nabla_\alpha((x - A\alpha)^T(x - A\alpha) + \lambda\alpha^T\alpha) = 0 \tag{18}$$
$$-A^T x + A^T A\alpha + \nabla_\alpha(\lambda\alpha^T\alpha) = 0 \tag{19}$$
$$-A^T x + A^T A\alpha + \lambda\alpha = 0 \tag{20}$$
$$-A^T x + A^T A\alpha + \lambda I\alpha = 0 \tag{21}$$
$$-A^T x + (A^T A + \lambda I)\alpha = 0 \tag{22}$$
$$(A^T A + \lambda I)\alpha = -A^T x \tag{23}$$
$$\alpha = (A^T A + \lambda I)^{-1} A^T x \tag{24}$$

Used code in Matlab

```
Input: atraining image matrix A for K subjects, a test image matrix X. Compute OR=A.Lam is a value of lambda. for x $\in$ X do $[Aq, Q] = qr(X, 0);$
$T = transpose(Q) * Q;$
$IL = eye(size(T));$
$mainin = inv(T + (Lam * IL));$
$Q = mainin * transpose(Q);$
$C = Q * Aq';$
find the identity of image x via (4.3)
end for
Output: identity for all test images
```

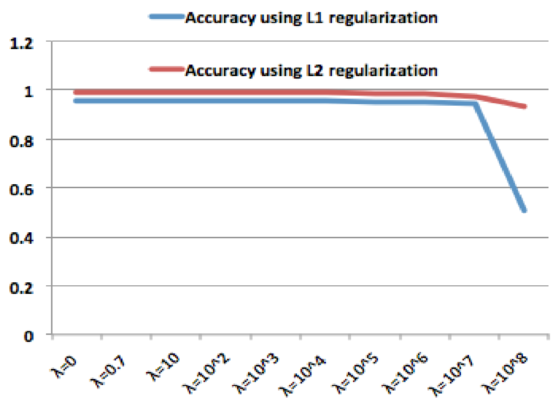
6.2. Accuracy on Changing hyper-perameter λ



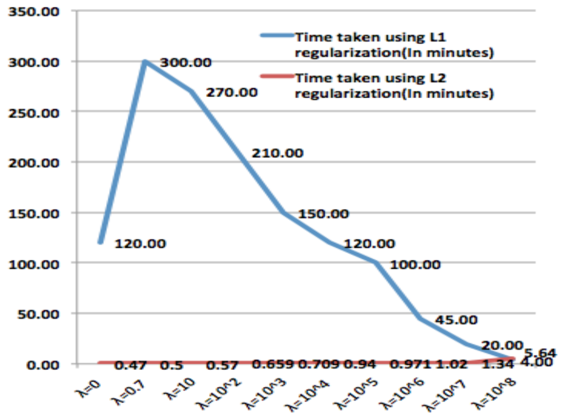
Time Seconds	Lambda	Accuracy
28.213	λ=0	0.990018
30	λ=0.7	0.990018
34.212	λ=10	0.990018
39.598	λ=10^2	0.990018
42.565	λ=10^3	0.990018
56.432	λ=10^4	0.990018
58.279	λ=10^5	0.987296
102.107	λ=10^6	0.985481
134.763	λ=10^7	0.972777
564.104	λ=10^8	0.931942

7. Comparison of changing Lambdaλbetween L1 and L2 regularization

Executing the l2 orthonormal algorithm with l1 regularization and l2 regularization, we able to get the accuracy. While, changing hyper-parameter(λ) in that resulted different accuracy which are shown in above sections(5.2 and 6.2) Here, we can see the overview of both the values and understand the effects on accuracy as well as on time:



The shown graph illustrates that till lambla = 10000000, the accuracy was slowly decreasing and after that it steeped down a lot. This results that till some errors, the accuracy was better.



The above graph explains that L1 regularization takes huge time to compute accuracy while, L1 take very less time using fast-track method, instead of examining all the images for training. Although, it provides a better result than the L1 method, where all images are trained and test is taken.

8. Face detection with Regularized L2 Orthonormal Algorithm

8.1. Face Detection Technique

The most important step in face recognition is the detection of the face. Face detection system is analyzing same faces and non-faces to train the identifier that can identify face.So, we train some faces and non-faces. In the end of process, identification of faces from any image become possible. It analysis the whole face with their alignments, modeling, authentication of face and its expressions.There are many factors those effects the face detection such as pose, noise in image, focus and other image processing factors. Moreover, face detection depends on shape, motion or color as well. Generally, frontal facial images are considered and then compared to database. For object class detection, locations and sizes of objects in image belongs to a given class. There are many algorithms to be used for face detection according to different approaches. The most well-known and used method is ”Viola-Jones” that uses Ada-boost and other components.

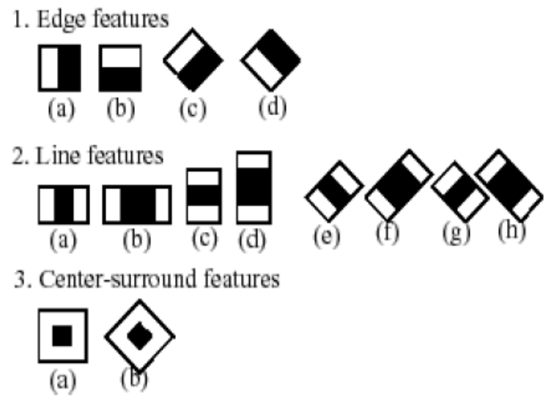
8.2. Viola- Jones Method

The Viola-Jones object detection was introduced by Paul Viola and Michael Jones in 2001.It was the first algorithm to detect the objects using the four steps described below: Before explaining the process of Viola-Jones algorithm, concept of edge detection is briefed out. Edge detection is converting a image with high intensity at pixels of that image.

which is generated by convolution kernel from an input image.

8.2.1 Haar Feature Selection

Haar features are similar to convolution kernels, which are used to identify the features in the given image. Haar features have some characteristics of faces. For instance: Nose sides will be darker with long bright path in between. A Haar feature considers adjacent rectangular regions at a specific location in detection window. This results in a single value that is calculated by subtracting the sum of pixels of white rectangle from the sum of pixels of white rectangle from the sum of pixels under black rectangle.



Suppose using a feature from above features and apply those on whole image then it will identify the same pattern on that image, which means wherever the portion of image resembles the feature, it will able to extract by putting high values. Thus, applying Haar feature gives the high value at the region only where feature matches the pattern of the image. Viola Jones uses 24*24 sub window from an image and calculates these features all over the image.

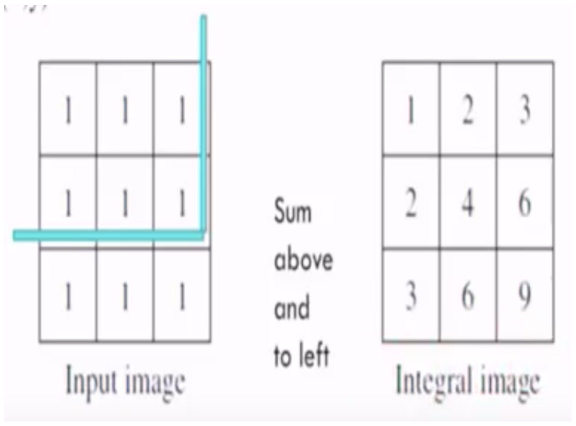
8.2.2 Integral image creation

Due to applying 24 * 24 window as a base window size to evaluate Haar features in any given image. Considering all possible parameters of the Haar features like: position, scale and type, more than 160000 features has to be calculated in this window. Therefore, to calculate the rectangular features in fixed time, an integral image can be computed. This can be done by making each pixels equal to the entire summation of all pixels above and to the left of the concerned pixel. The following formula is used to convert an image in integral image.

$$I(x, y) = \sum_{x' \leq x, y' \leq y} (x', y') \tag{25}$$

Where, O is original image. Integral image provides extremely high efficiency to complete the summation of any rectangular area. The summation of any rectangle area can be calculated as below:

$$I(x, y) = \sum_{(x,y) \in z} O(A4) + O(A1) + O(A2) + O(A3) \tag{26}$$



As we can see from above equation z is all corner values of rectangle. Integral image allows calculating the sum of all pixels inside any given rectangle by using four values srom the corners of the rectangle.

8.2.3 Adaboost Training

After having all the size, posion and other values by extracting features through applying the Haar feature on integral image. As it is mentioned, these can be approximately 160000+ feature values within a window at 24 * 24 base resolution. It is questionable that is all features are useful for an face?. Indeed, we do not need all these features. Adaboost is an algorithm helps to eliminate not useful features from the 160000+ feature. After these features are found a weighted combination of all these features is used deciding any given window has a face or not. If any feature perform better than rendom guessing, then that feature will be considered to use. These features are called weak classifiers Adaboost constructs a strong classifier as a linear combination of these weak classifiers. It is shown in following formila where, F(x) is a strong classifier and $\alpha f(x)$ is weak classifier with its weight.

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + ... + \alpha_n, f_n(x) \tag{27}$$

8.2.4 Cascading Amplifiers

Cascade of more complex classifiers achieves even better detection rates. The concept is to scan the detecting window frequently with a new size on the same image. Even if an image should contain one or more faces, it is obvious that an excessive large amount of the evaluated sub-windows would still be non-faces. Hence, a single strong classifier formed out of linear combination of all the best feature is not a good to evaluate on each window. Cascade classifier is a composed of stages each containing a strong classifier. So, all the feature groups are taken as into several stages. Each stage is used to identify that sub-window is definitely not a face or may be a face. This way stages with non-faces are discarded.

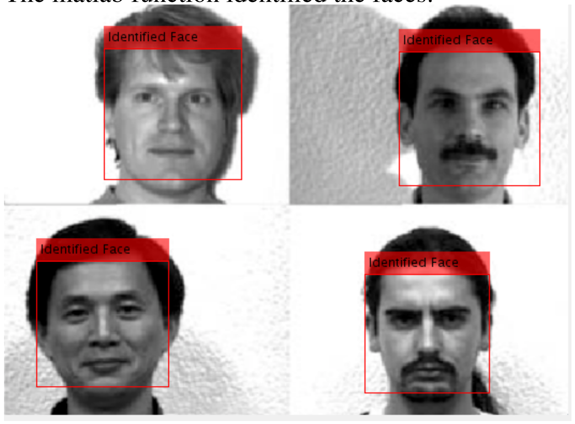
8.3. Implementing Viola-Jones in Matlab

Viola-Jones algorithm in Matlab

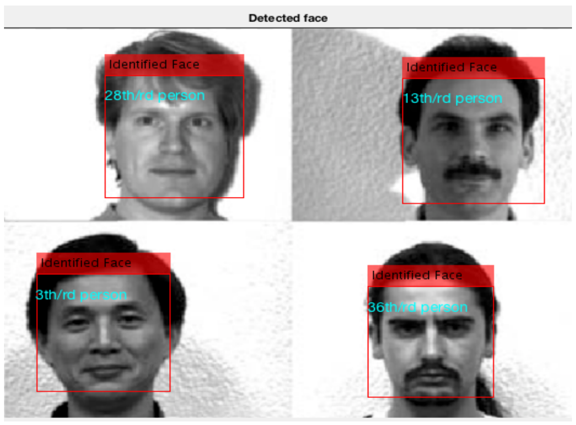
$img = input_Image$
$Vision.CascadeObjectDetector$
$Main_box = step(faceDetector, img)$
$Ibox = insertObjectAnnotation(img, Rectangle$
$, Main_box,' detectedfaces',' color',' yellow')$
$J = rgb2gray(img)$

The following image was give as an imput:

The matlab function identified the faces:



Running L2 norm regularization with face recognition:

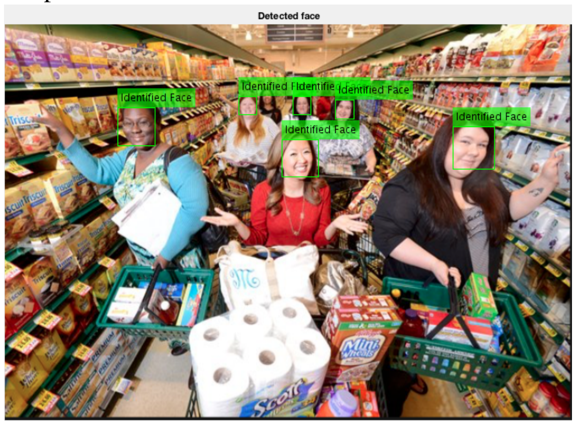


9. Face Detection in Supermarket Image

Running the same Viola-jones algorithm of the following input, it able to give most of the faces from image of Supermarket.

Input:

Output:



10. Conclusion

To conclude, face recognition is one of the most researched area in computer vision. The system starts with the face detection and fetch the faces from given images. After that, running the face recognition algorithm helps to identify the faces are same or not. While speaking about face recognition, generally the image patches and features are weighted(using alpha) to compare another image.The face recognition algorithm trains the weights through training faces and then we can use that weight to identify similar faces. many time training an algorithm over fits the data and in result, it does not able to perform well on the testing data. Thus, introducing the regularization to the face recognition methods prevents data to be over-fitted or under-fitted. The above research shows how lambda effected the accuracy of the face recognition. When, lambda is 0, data was over-fitted and increasing lambda, putting manual errors in training data reduces the accuracy over 10000 difference of lambda. Eventually, using hyper-parameter in any optimization or machine-learning algorithm, we can able to prevent over-fitting and under-fitting of data during training algorithm.

11. references

[1] Q. Shi, A. Eriksson, A. van den Hengel, and C. Shen. Is face recognition really a compressive sensing problem? In Proc. IEEE Conf. Computer Vision and Pattern Recognition, Colorado Springs, USA, 2011.

[2] J. Wright and Y. Ma, Dense error correction via ‘1-minimization, IEEE Transactions on Information Theory, 56 (2010), pp. 3540–3560.

[3] M. Figueiredo, R. Nowak, and S. Wright, Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems, IEEE Journal of Selected Topics in Signal Processing, 1 (2007), pp. 586–597.

[4] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. A method for largescale l1-regularized least squares. IEEE Journal on Selected Topics in Signal Processing, 1(4):606– 617, 2007.

[5] Jensen, Ole Helvig. Implementing the Viola-Jones face detection algorithm. MS thesis. Technical University of Denmark, DTU, DK-2800 Kgs. Lyngby, Denmark,