Computer Vision Laboratory - Tsukuba University

Monthly report - NOV. 2015

Tony Duong

December 7, 2015

Contents

Introduction

In this monthly report, I will show the work done during the month of November 2015.

After learning some basic concepts about computer vision such as Principal Component Analysis and the two fundamental subspace methods (Subspace Method and Mutual Subspace Method), I broadened my knowledge in this field by learning more advanced subspace methods, the Constrained Subspace Method and the Orthogonal Subspace Method. I also studied about kernel PCA as, I think, it is the first step to learn the kernel subspace methods. These kernel subspace methods will be studied the next month, in December. Until now, I was only focused on face recognition but face detection is also a part of computer vision and in order to develop a face recognition system, that part cannot be skipped. Nowadays, the Viola-Jones algorithm is said to be one of the most efficient algorithm to detect faces in images.

As a result, I developed during this past month an application on Matlab that can detect face on a random image and classify this face. The learning phase is also included in the application.

Part 1 : Preparation

Introduction

After learning the basics of computer vision in October, learning new concepts and algorithms as well as reading research papers became easier. I will in this section explain the different concepts studied : Constrained Mutual Subspace Method, Orthogonal Mutual Subspace Method, Kernel PCA, and Viola-Jones for face detection.

Constrained Mutual Subspace Method

The Constrained Mutual Subspace Method or CMSM is a evolved method of Mutual Subspace that increase the efficiency of the latter.

To improve the performance of MSM, we project both the input subspace and the references subspaces onto a constraint subspace. The constraint subspace is defined as the subspace which results by removing the principal component subspace (which is the subspace that holds the best common features of the images). So the constraint subspace is spanned by the N eigenvectors associated with the N smallest eigenvalue. N has to be chosen. This constraint satisfies the following condition : "it includes only the essential component for recognition between class subspaces". Indeed, as the eigenvectors associated with the highest eigenvalues "holds" the best common features, those associated with the smallest eigenvalues "holds" the unique features of each face.

The generation of the constraint subspace enhances the performance because it represents the difference among the class subspaces. Therefore, projecting the class subspaces onto it would maximize the distance between all the subspaces and that would allow a higher classification accuracy.

Orthogonal Mutual Subspace Method

This method consists in orthogonalizing the class subspaces in advance. That way, the classification ability will be higher.

OMSM is another enhancement of the MSM method, like the CMSM. Their performance are similar even though the technique used is different. In this method, we want to orthogonalize each subspace in order to "increase" their separability. For that, we have to calculate the whitening matrix which will be

used to orthogonalize each subspace. The details can be found in the following paper, [1]. After orthogonalizing all the subspaces using the whitening matrix, the only step left is to calculate the similarity between all the reference subspaces and the input subspace (which also has to be orthogonalized with the whitening matrix).

Kernel PCA

Kernel PCA is an extension of PCA using techniques of kernel methods. The goal of this method is to be able to separate class that are cannot be separated linearly using a nonlinear technique. For example,

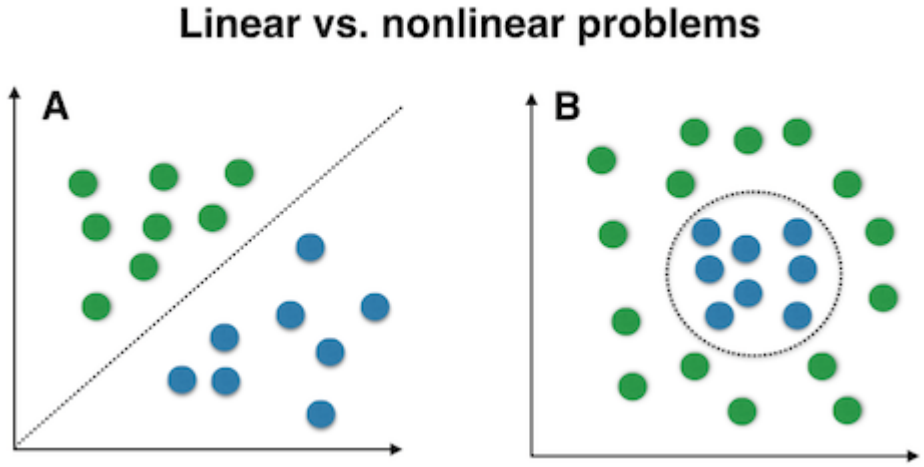

Figure 1 - Linear vs nonlinear problems

The idea is to project the data onto a higher dimensional space where it becomes linearly separable. Following are some examples from Wikipedia.

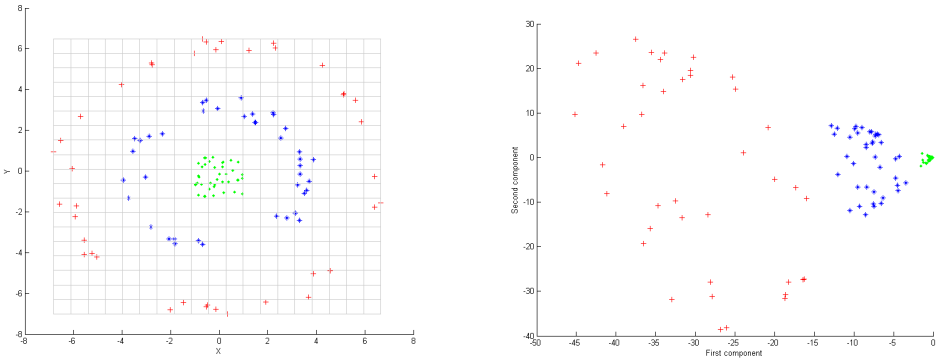

Figure 2 - Original data and data projected after application of kPCA

For now, I can only understand the main idea of this method. I will, during this month continue my learning on kernel methods and will explain more in details the several method using Kernel in the next report.

Viola-Jones

Viola-Jones differs from the previous concepts as it is a face detection algorithm. It is considered as one of the fastest and most reliable algorithm for face detection.

Viola-Jones algorithm has four steps :

1. Haar features
2. Integral images
3. Adaboost Training
4. Cascading Classifiers ...

All human faces share some similar properties. These regularities may be matched using Haar Features. For example, the eye region is darker than the upper-cheeks or the nose bridge region is brighter than the eyes. To determine the best features that can represent a human face, we use an image representation called integral image that evaluates the rectangular Haar features very quickly.

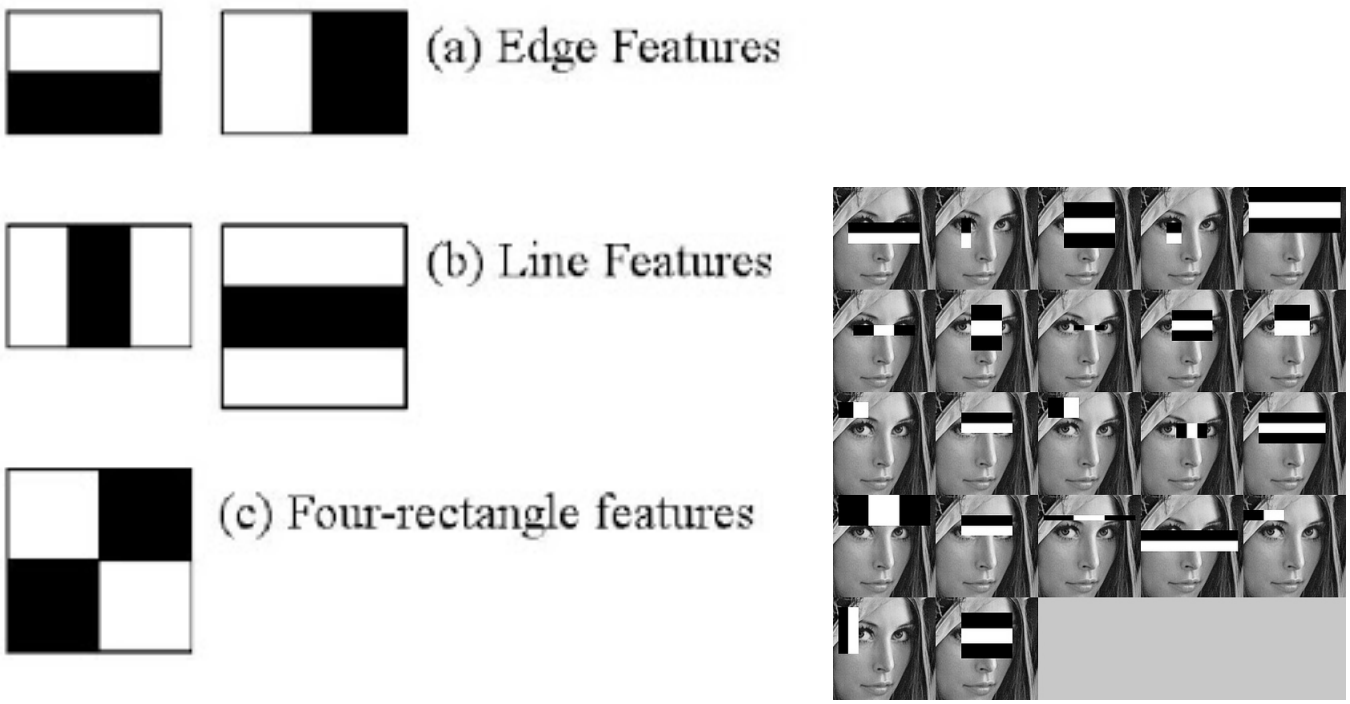

Figure 3 - Haar features and some examples of its application on face image

Each feature results in a single value which is calculated by subtracting the sum of pixels under white rectangle from the sum of pixels under black rectangle.

A problem about Haar features is that approximately 160000 features need to be calculated within a detector at 24*24 base resolution because we consider all the possible parameters of the Haar features

such as position, scale and types. But of course, only a few of them will be useful and we will use an algorithm called Adaboost which will help in finding only the best features.

To calculate the sum of pixels in a rectangle, the best way is to use integral images. Following is an image that illustrate the integral image. To calculate the sum in a rectangle, we only need four values as you can see in the following image. That enables very fast processing.

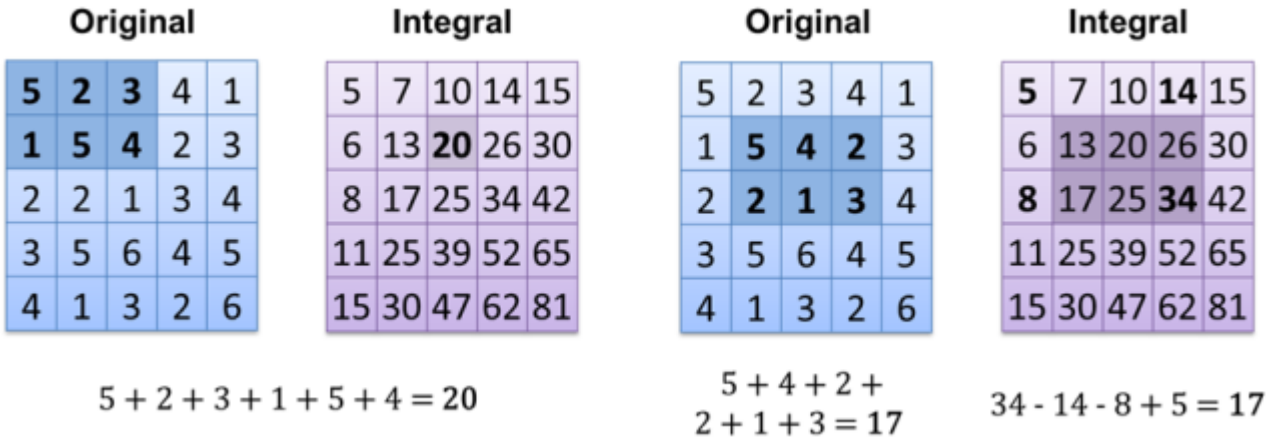

Figure 4 - Examples of integral images

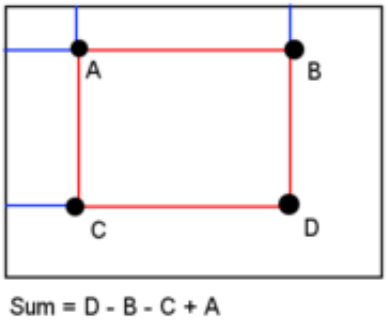

Figure 5 - Method for calculating the pixels' sum in an random rectangle in an image using Integral image

In the image above, calculating the sum of pixels in the red rectangle requires only the four values A, B, C and D of the integral image.

After finding the best features, we are going to use a cascade classifier. This classifier is composed of stages each containing a strong classifier. So, all the features are grouped into several stages. The job of each stage is to determine if a given sub window may be face or not. At each stage, if it fails, the sub window is immediately discarded. A sub window is considered a face if it passed all the stages.

A sub window, mentioned above, is only a portion of the image (in Viola-Jones it is a square). We try to cover and apply the classifier on the entire image and then check if that sub-window may be a face. To do that, the sub-windows are of different sizes and different positions on the original image.

Following are some examples of the application of Viola-Jones.

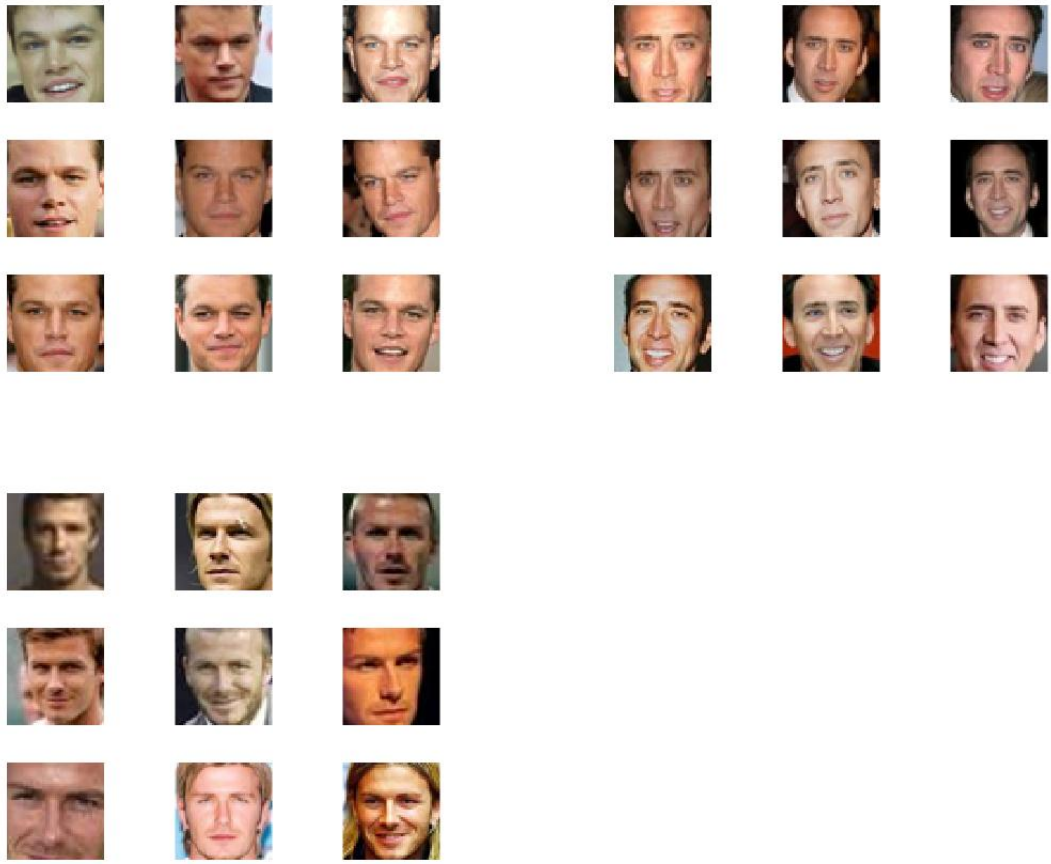

Figure - Cropped images

When using this algorithm, as a result, many sub-windows can be detected as being a face and overlapping is frequent. But we need only one window per face. That is why I developed a simple algorithm that can merge many sub-windows detected for one face into one "average" window.

Figure 6 - Windows found

Figure 7 - Result of the windows merging algorithm

In the face recognition application that I developed, after detecting the face, I cropped the original image and just keep the face part and then use it as a training image or an input image for recognition.

Part 2 : Development

Introduction

Now that I understand the fundamental about face detection, I can develop a more advanced application than the last one developed in October. These are the features of the application.

1. Training using data set of N classes, only the images with face detected will be retained in the training. N subspaces will be generated (I used Constrained Subspace Method).
2. Training using a m. file. When using 1., a file called data.m will be generated in the folder. It can be used again as the training takes time (because face detection is also used).
3. Recognition using a single picture. Face detection is applied first, and then with the cropped image we apply CSM to classify it.

Training phase

As I said, there are two ways to "train" the system. First is to select a dataset of images. These images can be of any kind and any size. The system will look at each image and then try to find a single face inside the picture with the Viola-Jones algorithm. After that, if a face is found, then the image is cropped and the face part will be used for training. Otherwise, the system skips the image and goes to process the next image. So, it is not important if the images have different backgrounds or are different because the system will only keep the "good" ones anyway. At the end of the training, a file named "data.m" will be generated in the current folder and can be used for the next time. This will allow to avoid training everytime we open the application (a training of 30 images lasts around 1 minute).

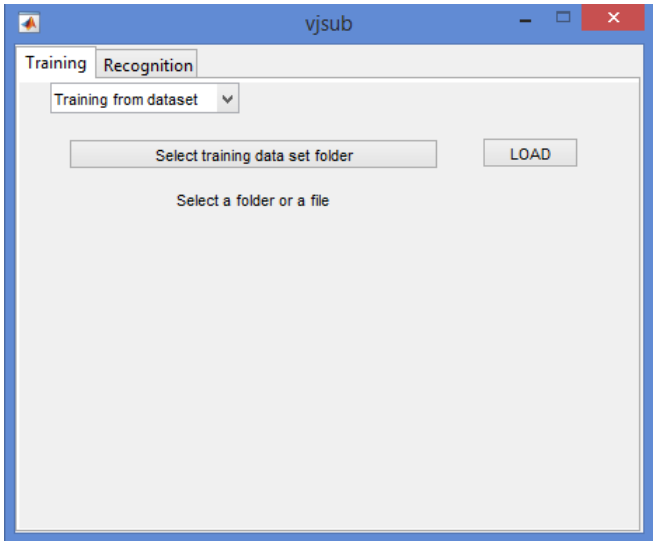

Figure 8 - Training dataset mode

The other way is to load the generated m file "data.m" which contains all the necessary variables for recognition (the subspaces generated for each class, the matrix composed of vector images).

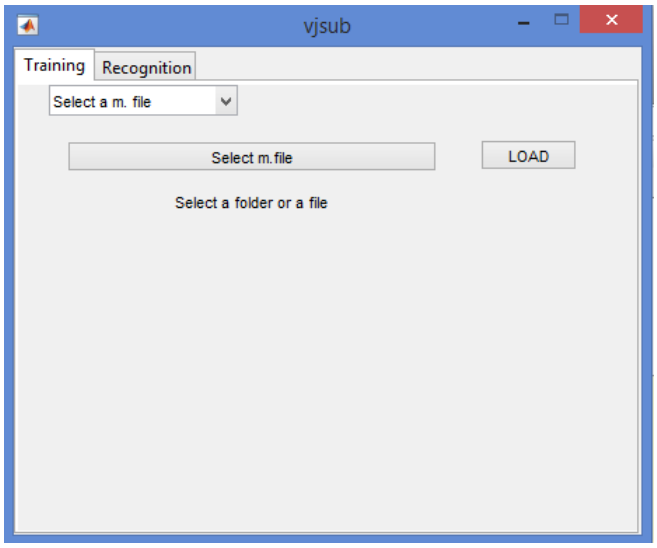

Figure 9 - m file mode

Recognition phase

After loading the reference subspaces, we can now proceed to recognition. The recognition process is rather simple. First, you have to choose an image with the face to be recognized. Then, the system will, like in the training phase, use the Viola-Jones algorithm to find a face in the image. After resizing the image, it uses the Constrained Mutual Subspace Method (CMSM) for classifying the face. The result will then be displayed.

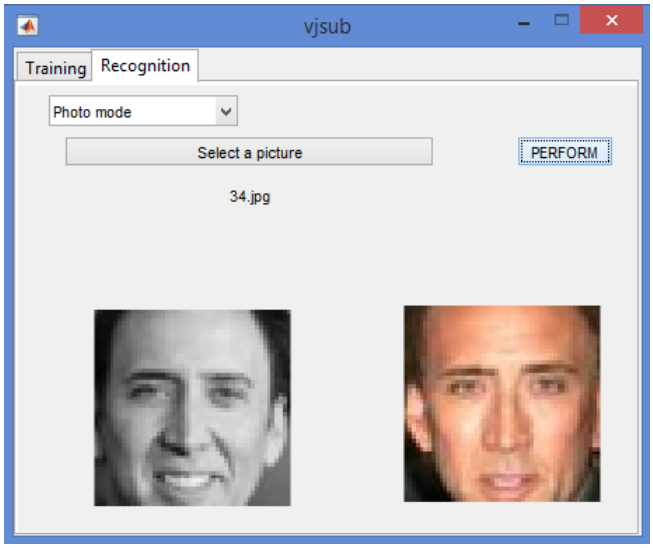

Figure 10 - Recognition mode

In this example, the two images were the following images.

The left image is the input image and the right one is one random image from the reference class that has been found.

Part 3 : Difficulties

This month has been better than the last one. Reading research papers and understanding the main idea of them were easier as I learned many things related to computer vision.

First, I faced difficulties with the Viola-Jones algorithm. When applying it to an image, it tries to find as many faces as possible in the image. The problem is that one face can be detected many times and then, many windows have been found. The problem was "How to find the best window among them ?" or "How could I combine all these windows to create the best one for representing the face?". I opted for calculating the average of all these windows and the result was satisfactory even though some irregular images still come out.

Second, OMSM was also difficult for me to understand. I know that it has a better ability than MSM and that the main point is that it orthogonalizes all the subspaces. But "How does this process make it better than the MSM ?" or "Why orthogonalizing the subspaces would increase the classification ability ?" were my questions. I couldn't find clear answers in the research papers, but after asking Lincon about that problem, I could understand it better.

Third, one of my problem is that the classification ability of my system is not satisfactory. When putting an image from the dataset as an input image, sometimes the system produces a wrong result. According to Chendra, I should put more references image in the training process (as of now, only 10 images per person) or try to crop to images differently because my cropping sometimes takes the background too which can be confusing for the system. Other suggestions were to change the recognition algorithm from CMSM to Kernel methods or to choose the reference image more appropriately because some of my images of the same person are very different (hair length, facial hair, position of the face).

Part 4 : Conclusion

During this month, I learned more advanced methods of the subspace methods, CMSM and OMSM. Viola-Jones was also very important as face detection will be indispensable for my face recognition system. For the next month, I plan to start studying the kernel methods which are more efficient and have more abilities in separating non-linear data set. I will also try to add the camera function where my application would be able to capture faces in real-time from the camera (Kinect, webcam...).

References

[1] FUKUI Kazuhiro, YAMAGUCHI Osamu, Comparison between Contrained Mutual Subspace Method and Orthogonal Mutual Subspace Method (2006), http://www.cs.tsukuba.ac.jp/internal/techreport/data/CS-TR-06-7.pdf

[2] VIOLA Paul, JONES Michael, Rapid Object Detection using a Boosted Cascade of Simple Features (2001), https://www.cs.cmu.edu/ efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

[3] MAKOVETSKY Ruslana, PETROSYAN Gayane, Face Detection and Tracking with Web Camera (2012), http://cs.mcgill.ca/ gpetro6/links/report.pdf

[4] FUKUI Kazuhiro, YAMAGUCHI Osamu, Face Recognition Using Multiviewpoint Patterns for Robot Vision (2003), http://www.cvlab.cs.tsukuba.ac.jp/ kfukui/english/epapers/isrrModifiedwithHeaders.pdf

[5] NISHIYAMA Masashi, YAMAGUCHI Osamu, FUKUI Kazuhiro, Face Recognition with the Multiple Constrained Mutual Subspace Method (2005), http://www.cvlab.cs.tsukuba.ac.jp/ kfukui/english/epapers/AVBPA05.pdf