Face Recognition via Sparse Representation

Wabartha - Durand - Darmet - Vayer

March 25, 2017

Overview

Plan

Article's origins

- University of Illinois
- PHD work of John Wright. Work on Face and object recognition. Student Member of IEEE. 16405 citations since 2012. Now work for Columbia University
- Published in : IEEE transactions on pattern analysis and machine intelligence (TPMAI) in 2009 (a reference specialised in computer vision and patter analysis/recognition) -
- Article cited by 6386 since 2009 -
- IEEE stand for Institute of Electrical and Electronics Engineers : almost a monopoly in the area. They organise enowned conferences. -
- J.Wright worked under the supervision of notably Yi Ma (a reference in Computer Vision)

Contributions

The main contribution of the article is concerning the sparse approach for image processing which leads to :

- features extraction (eigenfaces,FisherFaces,...) is not crucial anymore
- occlusion, corruption and noises robust
- fast (light computation)
- Sparsity Concentration Index (efficently detect invalid images)

Plan

Face recognition

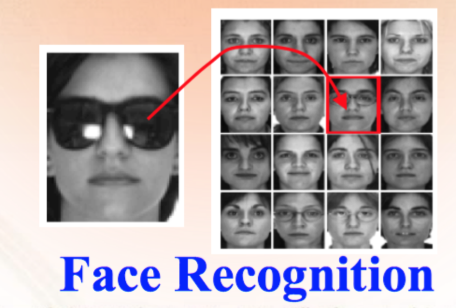

Figure: Face recognition

Symbols

- Let's consider k distinct object classes in the training data (k faces)
- An image of size $w \times h$ is indentified as a vector $v \in \mathbb{R}^m$ where $m = wh$ given by stacking it columns
- The n_i given training samples, taken from the i-th class (face) are arranged as columns of a matrix $A_i = [v_{i,1}, v_{i,2}, ..., v_{i,n_i}] \in \mathbb{R}^{m \times n_i}$
- So the columns of A_i are the training face images of the i-th subject

Sparse Linear Combination

Given some training samples of the i-th object class, $A_i = [v_{i,1}, v_{i,2}, ..., v_{i,n_i}] \in \mathbb{R}^{m \times n_i}$, any new test sample $y \in \mathbb{R}^m$ **from the same class** will be described as as linear combination of the training samples associated with object i :

$$y = \alpha_{i,1} v_{i,1} + \alpha_{i,2} v_{i,2} + ... + \alpha_{i,n_i} v_{i,n_i}$$

for some scalars $\alpha_{i,j} \in \mathbb{R}$

Sparse Linear Combination

Since the membership i of a test sample is unknown we define a new matrix A as the concatenation of n training samples of all k object classes :

$$A = [A_1, A_2, ..., A_k]$$

Such that the linear representation of y can be written :

$$y = Ax_0 \in \mathbb{R}^m$$

where

$$x_0 = [0, ..., 0, \alpha_{i,1}, \alpha_{i,2}, ..., \alpha_{i,n_i}, 0, ..., 0] \in \mathbb{R}^m$$

Sparse Linear Combination

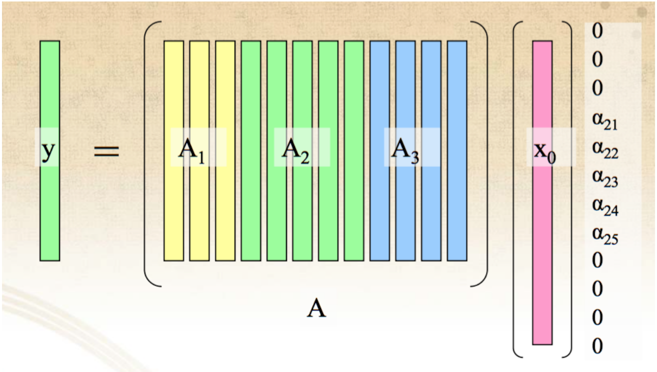

Figure: Sparse Linear Representation

To solve $y = Ax$ via L^0

- The idea is **to seek the sparsest solution** to $y = AX$
- Method :
  - We can do it by solving :

$$\hat{x}_0 = \text{argmin} \, \|x\|_0 \text{ subject to } Ax = y$$

  - Where $\|.\|_0$is the L^0 norm which is the number of nonzero entries in a vector
- Problem : this problem is NP-hard, and difficult even to approximate (combinatorial optimization)

Sparse solution via L^1 minimization

- Some results in the theory of sparse representation show that if the solution x_0 **is sparse enough** the solution to the problem above is equal to the solution of the L^1 minimization problem :

$$\hat{x}_1 = \text{argmin} \, ||x||_1 \text{ subject to } Ax = y$$

- This problem can be solved in **polynomial time** by standard linear programming method

The main result

Sparse solution for L^0 minimization via L^1

As long as the number of nonzero entries of x_0 is a small fraction of the dimension m, L^1 minimization will recover x_0

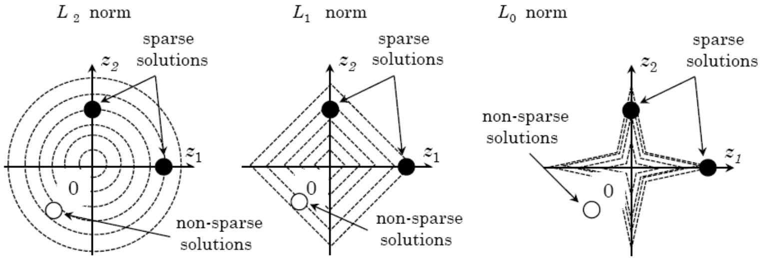

Figure: Sparse solution

Plan

Methods

- For each class i, we denote $\delta_i : \mathbb{R}^n \to \mathbb{R}^n$ the characteristic function which selects the coefficients associated with the i-th class
- Using only the coefficients associated with the i-th class, we can approximate the test sample y as $\hat{y}_i = A\delta_i(\hat{x}_1)$
- We classify y based on the approximation that minimizes the residual $\|y - A\delta_i(\hat{x}_1)\|_2$

Algorithm 1: Sparse representation based Classification (SRC)

SRC

1. Input : a matrix of training samples $A = [A_1, A_2, ..., A_k] \in \mathbb{R}^{m \times n}$ for k classes and a test sample $y \in \mathbb{R}^m$
2. Normalize the columns of A to have unit L^2 norm
3. Solve the L^1 minimization problem :

$$\hat{x}_1 = \text{argmin } \|x\|_1 \text{ subject to } Ax = y$$

4. Compute the residuals
   for all $i = 1, ...k$, $r_i(y) = \|y - A\delta_i(\hat{x}_1)\|_2$
5. $\mathcal{C}_y = \text{argmin}_i \, r_i(y)$

Algorithm 2: SRC with noise (relaxed problem)

- Since the real data are noisy we can rewrite the model :

$$y = Ax_0 + z$$

  where $z \in \mathbb{R}^m$ is a noise term with bounded energy, i.e, $\|z\|_2 < \epsilon$
- The sparse solution x_0 can still be approximately recovered by solving the stable L^1 minimization problem (eq to LASSO) :

$$\hat{x}_1 = \text{argmin} \|x\|_1 \text{ subject to } \|Ax - y\|_2 \leq \epsilon$$

Algorithm 3: SRC with feature extraction

- The feature extraction is crucial because it reduce data dimension and computational cost
- Most feature transformation involve only linear operations into a **feature space** represented by $R \in \mathbb{R}^{d \times m}$ with $d << m$
- The first problem becomes

$$\tilde{y} = Ry = RAx_0 \in \mathbb{R}^d$$

- The main results is if x_0 is **sparse enough** then with overwhelming probability it can be recovered via L^1 minimization
- Random features can be used ! (extremely efficient to generate)

Algorithm 4: SRC with occlusion

- Some corrupted pixels (in a small portion of the image)
- The problem becomes $y = y_o + e_0$ where $e_0 \in \mathbb{R}^m$ is a vector of errors (a fraction ρ of its entries are nonzero where pixels are corrupted).
- So we can rewrite the roblem as :

$$y = [A, I] \begin{bmatrix} x_0 \\ e_0 \end{bmatrix} = Bw_0$$

- If the occlusion e covers less than 50 % of the image the sparsest solution to $y = Bw$ is the true generator $w_0 = [x_0, e_0]$

Plan