Face recognition through RGB-D images and matrices of SVMs

Based on Hayat et al., 2016

Federico Simonetta, matricola 1129912
Alberto Cenzato, matricola 1134707

January 22, 2018

DEPARTMENT OF
INFORMATION
ENGINEERING
UNIVERSITY OF PADOVA

Overview

- Hayat et al., 2016: *An RGB-D based image set classification for robust face recognition from Kinect data*
  - Preprocessing
  - Image sets representation
  - SVMs training and prediction
- Datasets
- Our implementation
  - Preprocessing tweaks
  - Training strategies
  - Prediction
- Technical issues
  - Computational issues
  - Implementation issues
- Evaluation
  - Description of the experiments
  - Discussion and results
- Future Development

Hayat et al., 2016: An RGB-D based image set classification for robust face recognition from Kinect data

Purpose of the algorithm

- Purpose of the proposed algorithm is to recognize persons given an image set of that person in different poses
- This is useful especially for face recognition in video recording
- The algorithm take advantage of depth info captured through Kinect cameras

Preprocessing

- An important step is the image preprocessing
- First of all, depth images let a simple *kmeans* algorithm being enough to background segmentation
- Next, a precise cropping can be made, using precise pose informations computed on depth images
- Pose info are used to adjust cropping window sizes

Image sets representation

- Authors use faces poses to discover 3 different clusters for each original image set
- They represent each of these clusters with a 16×16 covariance matrix computed through LBP features vectors
  - Each image is subdivided in 16 blocks of equal size through a 4×4 grid
  - Over each block is computed the LBP feature vector
  - Each entry of the covariance is computed on all the LBP of the same block of the person in the same pose:

$$C_{p,q} = \frac{1}{n} \sum_{i=1}^{n} y(p,i) y(q,i)$$

where $y(k,j)$ is the difference of each k-th LBP vector from its own mean

Training

- Authors are not very clear in the training description
- They used Stein function as kernel for a vector of SVMs
- Each SVM is charged of recognizing one person in one of the three poses
- For each person in each pose, two SVMs exist, one for RGB images and one for depth images
- Stein kernel is defined by this equation:

$$k(X, Y) = e^{\sigma S(X,Y)}$$

where

$$S(X, Y) = \log\det\left(\frac{X+Y}{2}\right) - \frac{1}{2}\log\det(X \times Y)$$

- They also tested *k-fold* testing showing that results decrease increasing k

Prediction

- A query is composed by a set of images that are preprocessed and clusterized in three sets based on poses
- On each subset the covariance matrix is computed and is given in input to each SVM model
- Each of the three covariance matrices of the query will receive $6 \times n$ votes, where n is the umber of SVMs training sets
- The one that will receive the maximum number of votes will be predicted as the recognized face
- If no SVM will recognize it, it will be labeled as 'unknown'

Datasets

Datasets

- The project assignment was to test the proposed algorithm on a new small dataset of 338 images coming from 26 people, with different lightings and background scenes
- This was unsuccessful, hence, after many attempts, we changed the dataset and we tested the algorithm on a subset of the dataset used by the authors
- This new dataset contains more than 15.000 RGB-D images with 24 different images sets of 20 different peoples

Our implementation

Preprocessing tweaks (1)

The very poor results on the first small dataset forwarded us to implement some changes to the preprocessing algorithm

- *Face-detection-first*: we used Haar cascade classifiers made available by OpenCV to detect a face, crop it and then segment background and refine cropping; if no face was detected, we use a fixed threshold segmentation
- *Outlier removal*: we first compute the maximum are connected component and crop it with its smallest rectangle approximation; this is useful to a better clustering
- *Histograms segmentation*: we tried to segment the histogram region containing the highest peak and if face detection or pose estimation failed we retried with the region containing the second higher peak. We abandoned it because too much slow (but nice precision)

Preprocessing tweaks (2)

- *Adjusted face cropping*:
  - G. Fanelli et al. (2011) algorithm to compute face detection and pose estimation
  - We changed parameters for face cropping
  - Corrected ROI position along the X axis
- *Removed rotation matrix*: pose clustering directly on Euler angles: rotation matrix computing enlarged eventual pose detection errors

Training strategies

The proposed paper did not specify any particular training strategy. Because of this, we tried four different training algorithms:

1. All covariance matrices but one had negative labels
2. All SVMs of the same correct identity had positive label (very bad results)
3. Covariance matrices of the correct identity but of different poses were removed from the training set of each SVM (only on the first dataset)
4. *Leave-one-out*: without success; authors say that *k-fold* cross validation gave low results (only on the first dataset)

Prediction

- We noted that some identity received too many votes
- We introduced a new prediction rule: if an identity has more than t candidates with the maximum number of votes, it is forced to 'unknown'
- At first, $t = k \times c$, where:
  - $k = 1$ if only RGB or only Depth is used
  - $k = 2$ if they are used together
  - c is the number of pose subsets

  In fact, $k \times c$ is the maximum number of votes that an identity should receive.
- From experiments we found that using RGB images only, *FP-unknown* was always 0. We changed the rule so that it took in account only RGB images; consequently, $t = c$

Technical issues

Computational issues

- Preprocessing is really slow, especially the covariance computation
- We implemented multithread version but not GPU support
- Using Intel i7-4510HQ CPU, (4 cores, 2 threads per core, 2.5 GHz frequency, boosted to 3.5 GHz), preprocessing and covariance computing of all the dataset takes less than 10 minutes
- There are not particular issue on RAM usage, it is needed about 1GB during preprocessing. This is due to the covariance computing which need all images of an identity in RAM: fewer images per identity would need less RAM

Implementation issues

- Code for pose estimation needed depth images represented in OpenCV *Mat* objects
- We had to switch completely to OpenCV
- Other issues were found in the use of machine learning module of OpenCV, namely
  - in kmeans algorithm
  - in loading and saving from file of custom kernel SVMs

Evaluation

Description of the experiments

We setted many experiments, with different combination of 'known'-'unknown' ids. Every one was setted up in this way:

- 40 images per person were randomly removed and used as testing set. Each query was made by a set of 40 images
- 5 peoples chosen randomly were completely removed from the training set to simulate the 'unknown' behaviour
- People who had two recordings was never removed
- 1/3 of each sequence of the training set was used as validation set to evaluate SVMs during parameters optimization

Evaluation measures

We computed results in terms of:

- *rank-1*: ratio between correct recognitions and the total number of sets in the query, where 'unknown' identities are considered correct if detected as 'unknown'
- *FP-unknown*: ratio between incorrect recognition of 'unknown' persons and the total number of 'unknown' persons

Discussion

- Our results are comparable to the ones of the authors, taking in account the minor size of our dataset
- Despite the smaller sizes, the second dataset that we used had a very larger mean number of images per persons (about 700 images) with respect to the dataset used by the authors (about 300 images per identity)
- This let us argue that the major contribute to good performances is not given by the total number of images, but it is given by the number of identities
- Our results with only RGB or D images are much lower than those claimed by authors. We believe that in small datasets depth info is more representative than in bigger dataset

Results (1)

Table 1: Results with first version of the prediction rule

Removed ids	Using RGB-D		Using RGB only		Using D only	
	Rank-1	**FP-unk**	**Rank-1**	**FP-unk**	**Rank-1**	**FP-unk**
04, 06, 10, 11, 19	0.83	0.4	0.29	0.0	0.33	0.4
06, 10, 19, 20, 24	0.87	0.0	0.33	0.0	0.50	0.0
08, 09, 14, 17, 24	0.79	0.4	0.375	0.0	0.33	0.2
01, 16, 17, 19, 24	0.92	0.2	0.375	0.0	0.42	0.2
01, 09, 12, 16, 19	0.83	0.6	0.25	0.0	0.375	0.4
09, 10, 16, 19, 24	0.67	0.8	0.29	0.0	0.125	0.8
04, 09, 10, 11, 16	0.92	0.2	0.375	0.0	0.33	0.2
01, 08, 09, 10, 19	0.50	0.8	0.375	0.0	0.125	0.6
04, 10, 11, 13, 24	0.54	1.0	0.375	0.0	0.20	0.0
08, 14, 17, 23, 24	0.75	0.6	0.375	0.0	0.29	0.6
mean	**0.76**	**0.5**	**0.3785**	**0.0**	**0.33**	**0.34**

Results (2)

Table 2: Results with second version of the prediction rule

Removed ids	Using RGB-D		Using RGB only		Using D only	
	Rank-1	**FP-unk**	**Rank-1**	**FP-unk**	**Rank-1**	**FP-unk**
06, 09, 13, 17, 24	1.0	0.0	0.33	0.0	0.375	0.0
06, 10, 11, 17, 20	0.92	0.0	0.42	0.0	0.54	0.0
01, 09, 13, 14, 19	0.875	0.0	0.42	0.0	0.33	0.2
04, 06, 11, 16, 17	0.79	0.0	0.42	0.0	0.42	0.2
04, 11, 12, 17, 20	0.79	0.0	0.29	0.0	0.125	0.8
04, 11, 12, 16, 23	0.71	0.0	0.33	0.0	0.08	1.0
01, 06, 10, 17, 20	0.875	0.0	0.375	0.0	0.21	1.0
10, 14, 16, 20, 24	0.92	0.0	0.25	0.0	0.375	0.0
04, 06, 08, 14, 23	0.67	0.0	0.33	0.0	0.08	1.0
04, 08, 11, 16, 20	0.92	0.0	0.375	0.0	0.08	1.0
mean	**0.85**	**0.0**	**0.354**	**0.0**	**0.2615**	**0.52**

Future Development

Future Development

- Computational optimization
  - training and prediction not in multithreading
  - GPU (CUDA?)
- SVMs saving and loading to file
- Studying in deep the gap in results relative to training and prediction by using just RGB or just Depth
- Further exploring how results change when changing dataset:
  - how much is useful the prediction rule when using original author's dataset?
  - do results are still high by using few images per identity but many different identities (enterprise context)?
- Checking true positives of "unknown" predictions

Thank you for your kind attention