Learning Sparse Face Features : Application to Face Verification

Pierre Buyssens, Marinette Revenu
GREYC Laboratory – CNRS UMR 6072
ENSICAEN, University of Caen, Caen, France

Abstract

We present a low resolution face recognition technique based on a Convolutional Neural Network approach. The network is trained to reconstruct a reference per subject image. In classical feature–based approaches, a first stage of features extraction is followed by a classification to perform the recognition. In classical Convolutional Neural Network approaches, features extraction stages are stacked (interlaced with pooling layers) with classical neural layers on top to form the complete architecture of the network. This paper addresses two questions : 1. Does a pretraining of the filters in an unsupervised manner improve the recognition rate compared to the one with filters learned in a purely supervised scheme ? 2. Is there an advantage of pretraining more than one feature extraction stage ? We show particularly that a refinement of the filters during the supervised training improves the results.

1. Introduction

Face recognition is a topic which has been of increasing interest during the last two decades due to a vast number of possible applications: biometrics, video–surveillance, advanced HMI or image/video indexation. The main challenges in face recognition are illumination changes, head poses, artefacts (like glasses) or facial expressions.

1.1 Classical approaches of the task

Several approaches have been proposed [3], they can mainly be divided into two parts :

- the local approaches, which extract features and combine them into a global model to do a classification.
- the global approaches which realize often a kind of linear projection of the high–dimensional space (i.e. the face images) onto a low–dimensional space.

The local approaches first extract some features (like eyes, nose and mouth) by the use of special feature extractors. The recognition task is then performed using some measures (like the distance between the eyes) on these features.

The most popular local technique is the *Elastic Graph Matching* (EGM) where a set of interest points is extracted from the face, and then a graph is created. It has been widely used in the literature [2], [17], [20]. Most of the time, some Gabor filters are applied on the neighborhood of these points to take into account appearance information of the face.

The main drawback of the local approaches is that the extractors have to be chosen by hand and can be sub–optimal. Moreover, it is difficult to deal with different scales and poses.

The global approaches perform a statistical projection of the images onto a face space. The most popular technique called *Eigenfaces* is based on a Principal Components Analysis. First introduced by Turk and Pentland [19], it has been intensely studied in the face recognition community [11], [6], [18]. Another popular technique is the *FisherFaces* method [12], [9], [7] based on a Linear Discriminant Analysis (LDA), which divides the faces into classes according to the Fisher criterion.

The main drawback of the global approaches is their sensitivity to the illumination changes. When the illumination of a face changes, the appearance of it undergoes a non–linear transformation, and due to the linear projection of these global approaches, the classification can fail.

Extensions of these linear approches have been proposed like kernel–PCA [15], or kernel–LDA [8] for face recognition. The drawback of these extensions is that there is no invariance unless this one is built into the kernel, and once again by hand. This is also the drawback of other machine learning technics like Support Vector Machine.

1051-4651/10 $26.00 © 2010 IEEE
DOI 10.1109/ICPR.2010.169

2. Our Method

Our method relies on a special type of Convolutional Neural Network. Acting like a *diabolo* network [16] where the output vector is the same vector as the input with a less dimensional intermediate layer, our *Face–Reconstruction Network* takes in input some faces for each subject and tries to reconstruct a *reference* face that has been chosen beforehand. Here, the inputs of the network are images of size 56×46 which is a quite low resolution compared to most of other face recognition algorithms. Inspired by the work of Duffner and Garcia [5], the network learns automatically the feature extractors, and combines them to produce a more global model. The classification is then made by some classical feed–forward layers.

The principle problem of such an architecture is that the whole network is trained traditionnally with gradient descent, including the first layers. The saliency of the filters learned with backpropagation then depends essentially on the connections between layers. Moreover, the number of sufficient filters per layer is difficult to evaluate a priori.

One way to tackle these problems is to pretrain the first layers in an unsupervised manner. In order to have salient face features, we use a sparse coding algorithm which learns an overcomplete basis of filters. These filters are then used to initialize the first layers of our network.

The paper is organized as follow : the filter banks learning is described in section 3. The architecure is detailled in section 4. The training procedure and the results are presented in sections 5 and 6. Finally, we present our conclusions and further work in section 7.

3. Learning the sparse filters banks

The aim of the unsupervised sparse coding algorithm is to find a representation $Z \in \Re^m$ for a given signal $Y \in \Re^n$ by linear combination of an overcomplete set of basis vectors, which are the columns of a matrix $B \in \Re^{n \times m}$ with $m > n$ [13]. In optimal sparse coding, the problem is formulated as :

$$\min \|Z\|_0 \; s.t. \; Y = BZ$$

where the 'l^0–norm' is defined as the number of non–zero elements in a given vector. We can fortunately make a convex relaxation by turning the l^0–norm into an l^1–norm [4]. The problem can then be written as :

$$L(Y, Z; B) = \frac{1}{2}\|Y - BZ\|_2^2 + \lambda\|Z\|_1$$

In this work, we used the PSD algorithm (for *Predictive Sparse Decomposition* proposed in [14]) slightly modified to fit our data. It is composed of an encoder which produces a sparse code Z from an input Y, and a decoder which reconstructs the input from the sparse code. The global loss to minimize is defined as :

$$\begin{aligned} L(Y; G, W, D) \quad = \quad & \|Y - BZ\|_2^2 \; + \; \lambda\|Z\|_1 \\ & + \; \alpha\|Z - F(Y, P_f)\|_2^2 \end{aligned}$$

where the first term represents the error reconstruction, the third term denotes the error of the encoder prediction, and the l^1–norm ensures the sparsity of the code Z. $F(Y; P_f)$ defines the output of the encoder, where P_f denotes the parameters that are learned by the encoder, and specially the filters matrix $W \in \Re^{m \times n}$.

Learning proceeds in an iterative way, alternating the two steps : (1) minimize L with respect to Z keeping P_f and B constant, and (2) update P_f and B by stochastic gradient descent, using the coefficients of Z.

To proceed the learning of the sparse filters bank, we extracted 20000 patches of size 7×7 with sufficient standard deviation (to avoid too uniform patch) from the ORL face database [1]. The number m of filters (rows of W) has been set to 100, which is more than twice the dimension of the patches, so as to ensure the sparsity of the produced codes.

After training, filters are localized oriented edge detectors, see Fig.1.

A second sparse feature extraction stage is then processed, where the training patches are the outputs of the first feature extraction stage. 150 filters of size 6×6 have been learned from a training set composed of 40000 patches.

4. Architecture

The Face–Reconstruction Network (see Fig.2) takes as input an image of size 56×46 (i.e. the size of the retina of the network) and passes it through a succession of convolution C_i, subsampling S_i and fully connected F_i layers. The output of the Network is an image, with the same size than the input, which is reconstructed by the last layer F_7. Each pixel of the output is one neuron, so there are $56 \times 46 = 2576$ neurons on the last layer.

Due to our aim to pretrain some layers with a sparse coding technic, our network is considerably larger than classical ones (like the LeNet Network[10]):

- C_1. Feature maps : 100; Kernel size: 7×7; (Maps) Size: 50×40. Fully connected to the input.
- S_2. Feature maps: 100; Kernel size: 2×2; Size: 25×20.

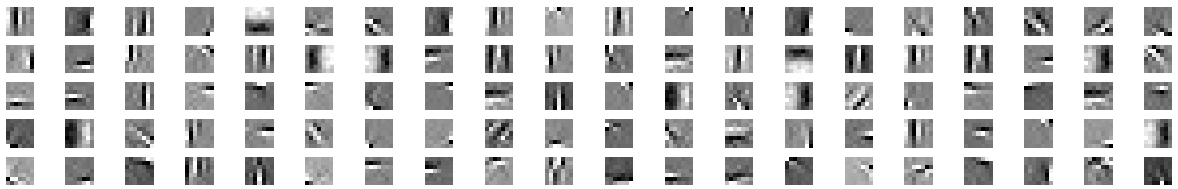

Figure 1. The 100 learned filters of size 7×7.

- C_3. Feature maps: 150; Kernel size: 6×6; Size: 20×15. Connection rate = 0.5.
- S_4. Feature maps: 150; Kernel size: 4×3; Size: 5×5.
- C_5. Feature maps: 200; Kernel size: 5×5; Size: 1×1. Fully connected to S_4.
- F_6. Neurons: 100; Fully connected to C_5.
- F_7. Neurons: 2576; Fully connected to F_6.

All the neurons use a sigmoid activation function. Note that when testing the network, this is not the state of the last layer which is taken into account, but the compact code represented by the state of the penultimate layer (that is to say 100 values).

5. Training the Network

Training the network consists on finding the optimal parameters of the network for a loss function. Here, we want to minimize the classical regression cost function $E = \frac{1}{2}\|\boldsymbol{o_p} - \boldsymbol{t_p}\|^2$ where $\boldsymbol{o_p}$ and $\boldsymbol{t_p}$ are the output values and the target values respectively for the pattern p.

The ORL database [1] is used to train the network, it contains 10 images for each of 40 subjects, with variations in lighting, facial expressions, accessories and head positions. The images have been resized to 56×46, and their pixel values normalized between -1 and 1.

The database is divided into two disjoint parts, the first one for the training, and the second one for the tests. For each subject in the train set, a *reference* image is chosen to be the target of the subject that the network has to reconstruct. In the 9 per subject remaining images, one image is chosen randomly to form the validation set. During the training phase, a cross–validation is performed with this last set to avoid overfitting the data.

6. Results

In all the experiments, the database is divided randomly into two disjoint parts, the first one composed of 35 subjects (350 images) is used to train the network, and the second one composed of 5 subjects (50 images) to test. The test protocol consists in 4 steps.(1) For a test image I_s of a subject s, the projection P_{I_s} (the vector of dimension 100 extracted from layer F_6) is computed. (2) For each subject in the database, a model is created. It is defined as the mean vector of the projections of all the subject images (except the image I_s). (3) Distances of P_{I_s} to all the models are then processed and (4) the rank of the recognition for I_s is computed. In this study, we choose the l^2 distance for all the experiments.

All the trainings and tests are processed 20 times with different (random) separations of the database, and the mean result is computed.

We conducted 6 different experiments to test the usefullness of the sparse pretraining, differing in the number of pretrained layers. We denote afterwards RR the experiment where the two feature extraction layers are classicaly initialized randomly. U denotes the experiment where the parameters of the first convolutional layer are initialized with the sparse filters that have been learned previously in an unsupervised manner. UU denotes the experiment where the first two convolutional layers are initialized with the sparse filters that have been learned during the first and second stages respectively. Each of these experiments is processed twice, depending on if we allow or not a refinement of the filters by gradient descent during final supervised training. The results of these experiments are presented in the table 1.

Table 1. Results. R: randomly initialized layer, U: layer pretrained in an unsupervised manner.

Training \ Experiment	RR	U	UU
Not refined	84%	88.7%	82%
Stand. dev.	4.67	3.10	3.52
Refined	87.5%	90.2%	88.2%
Stand. dev.	3.90	2.54	2.91

From the results in the table 1, we can draw three conclusions : 1. The most surprising result is the good recognition rate with the two feature extraction layers

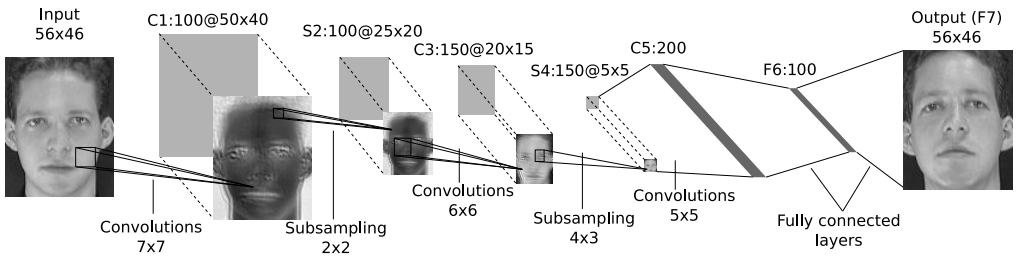

Figure 2. Architecture of the Network.

initialized at random and *kept fixed* during the training phase. We think this is due to the high number of filters, so we can capture sufficient variability of the input. Nevertheless, the standard deviation of this experiment is of course higher than the others. 2. The refinement of the filters gives always better results than keeping them fixed. 3. Using a second stage of pretrained filters gives in our case worst recognition rates than only one.

7. Conclusion and Future Work

We presented a low resolution face recognition method based on a Convolutional Neural Network. This network takes a face image in input and projects it onto a low–dimensional space where the recognition is performed. This paper adresses more particularly the problem of pretraining the features extraction layers in an unsupervised sparse manner. We successively show that, in our case, using a second stage of unsupervised pretrained filters does not give better results than only one, using only random filters gives decent recognition rates, and refining the filters during the final supervised training gives always better recognition rates. We are currently conducting experiments to better understand the surprising good results with the pure random filters, and eventually apply the best method to infrared face images.

References

[1] www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[2] R. Brunelli and T. Poggio. Face recognition: Features versus templates. *IEEE PAMI*, 15(10):1042–1052, 1993.

[3] D. P. Delacretaz, G. Chollet, and B. Dorizzi. *Guide to Biometric Reference Systems and Performance Evaluation*. Springer, 2009.

[4] D. L. Donoho. Sparse components of images and optimal atomic decomposition. Technical report, Department of Statistics, Stanford University, Dec. 1998.

[5] S. Duffner and C. Garcia. Face recognition using non-linear image reconstruction. In *i-LIDS: Bag and vehicle detection challenge*, pages 459–464, 2007.

[6] C. Gomez and B. P. Popescu. A simple and efficient eigenfaces method. In *ACIVS*, pages 364–372, 2007.

[7] D. D. Gomez, J. Fagertun, B. Ersboll, F. M. Sukno, and A. F. Frangi. Similarity-based fisherfaces. *Pattern Recognition Letters*, 30(12):1110–1116,, Sept. 2009.

[8] J. Huang, P. C. Yuen, W. Chen, and J.-H. Lai. Choosing parameters of kernel subspace LDA for recognition of face images under pose and illumination variations. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 37(4):847–862, 2007.

[9] D. J. Kriegman, J. P. Hespanha, and P. N. Belhumeur. Eigenfaces vs. fisherfaces: Recognition using class-specific linear projection. In *ECCV*, pages I:43–58, 1996.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Intelligent Signal Processing*, pages 306–351. IEEE Press, 2001.

[11] A. Lemieux and M. Parizeau. Experiments on eigenfaces robustness. In *ICPR*, pages I: 421–424, 2002.

[12] W. Li and C. Zhou. A novel combined fisherfaces framework. In *ICIP*, pages 2011–2014, 2004.

[13] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed in V1. *Vision Research*, 37:3311–3325, 1997.

[14] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*. IEEE Press, 2007.

[15] H. Sahbi. Kernel PCA for similarity invariant shape recognition. *Neurocomputing*, 70(16-18):3034–3045, 2007.

[16] H. Schwenk. The diabolo classifier. *Neural Computation*, 10(8):2175–2200, 1998.

[17] A. Stergiou. Face recognition by elastic bunch graph matching. Master's thesis, 2004.

[18] F. Tsalakanidou, D. Tzovaras, and M. G. Strintzis. Use of depth and colour eigenfaces for face recognition. *Pattern Recognition Letters*, 24(9-10):1427–1435, June 2003.

[19] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Proceedings IEEE CVPR*, pages 586–590, Hawai, June 1992.

[20] L. Wiskott, J. M. Fellous, N. Kruger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE PAMI*, 19(7):775–779, July 1997.