Fast and efficient face recognition system based on kernel sparse representation and fast CoSaMP

Zied Bannour Lahaw (✉ ziedd.bannour@gmail.com)

Universite de Tunis Ecole Nationale Superieure d'Ingenieurs de Tunis <https://orcid.org/0000-0002-8302-6139>

Hassene Seddik

Universite de Tunis Ecole Nationale Superieure d'Ingenieurs de Tunis

Research

Keywords: face recognition, Kernel sparse representation based classification, Compressive sensing, Fast compressive sampling matching pursuit.

Posted Date: April 13th, 2020

DOI: <https://doi.org/10.21203/rs.3.rs-15509/v2>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Fast and efficient face recognition system based on kernel sparse representation and fast CoSaMP

Zied BANNOUR LAHAW ✉
Products Research Centre (CEREP)
ENSIT, 5 Av. Taha hussein, 1008,
Tunis, Tunisia
ziedd.bannour@gmail.com

Hassene SEDDIK
Products Research Centre (CEREP)
ENSIT, 5 Av. Taha Hussein, 1008,
Tunis, Tunisia
Seddikhassne@gmail.com

Abstract

Kernel sparse representation based classification (KSRC) in compressive sensing (CS) represents one of the most interesting research areas in pattern recognition, image processing and especially facer recognition and identification. First, it applies dimensionality reduction method to reduce data dimensionality in kernel space and then employs the L_1 -norm minimization to reconstructing sparse signal. Nevertheless, these classifiers suffer from some shortcomings. KSRC is greedy in time to achieve an approximate solution of sparse representation based on L_1 -norm minimization. In this paper, a new method is mathematically developed and applied for face recognition based on Gabor-wavelets for feature extraction and KSRC for precise classification. The aim of the proposed algorithm is to improve the computational efficiency of KSRC by applying a supervised kernel locality preserving projections (SKLPP) for dimension reduction. In fact, the L_1 -norm minimization is performed by the use of the fast compressive sampling matching pursuit method (FCoSaMP). Experimental results prove that the proposed classification method is efficient, fast, and robust against variations of illumination, expression, and pose. Indeed, its computation time is highly reduced compared to the baseline performances.

Keywords: face recognition, Kernel sparse representation based classification, Compressive sensing, Fast compressive sampling matching pursuit.

1. Introduction and state of the art overview

Pattern recognition in images or videos, represent an important process in objects identification. In the last decades, face recognition has become an interesting research field,

naming a few; security system and access control. Its principal goal is to identify person's faces from a database. The features extraction and the classification of facial images represent the most important and decisive tasks in recognition process.

In order to find the most appropriate representation and discriminative features of the face image data, several attributes extraction methods are proposed in the literature. The technique of local appearance-based features represent an important approach in this way. Its role is to extract local descriptors by using appropriate image filters. It ensures the extraction of the local structures of face images even if the facial appearances are changed. The Gabor wavelet-based features represent the typical local-feature descriptor. The recent research proved its important representative capacity [1, 2].

On the other hand, classification is the second important step in pattern recognition. Consequently, classifier design and type is vital for efficient recognition tasks. In this order, several approaches are widely proposed. Based on compressed sensing (CS) [3], Wright et al. proposed a sparse representation based-classification (SRC) [4] to classify face image data with corruption and occlusion. SRC implements sparse representation of data by using the methods for sparse signal reconstruction and classifies data in terms of reconstruction errors.

A typical SRC method is performed in the original input space. Nevertheless, the standard SRC cannot capture the non-linear information within the data, mainly for high-dimensional data such as face image databases [5]. In order to solve this problem, Zhang et al. proposed the kernel sparse representation-based classifier (KSRC) to transform the input data into a high-dimensional feature space. The KSRC is based on some non-linear mapping associated with the reproducing kernel Hilbert space (RKHS). Hence, data in feature space are often of high-dimensionality and L_1 -norm minimization in this space is impractical. Before optimizing the corresponding cost function, adding a dimensionality reduction stage in feature space could be considered one of the ideal solutions. The dimensionality of the feature space can be reduced by exploiting non-linear subspace learning and manifold learning methods. The classical kernel principle component analysis (KPCA) [6] and kernel fisher discriminant analysis (KFDA) [7] methods consider only the global structure of training data. The manifold learning methods, such as Isomap [8], locally linear embedding (LLE) [9] and supervised kernel locally preserving projection (SKLPP) [10], effectively utilize the manifold structure or local neighbor structure of training data.

The optimal solution for reconstructing sparse signal in KSRC is achieved through L_1 -norm minimization. Two interesting approaches are implemented in this order: The first one

is convex optimization methods includes the least absolute shrinkage and selection operator (LASSO) [11], homotopy method [12] and gradient projection for sparse representation method (GPSR) [13]. The second one is based on greedy recovery methods which consist in reducing the computational complexity of the optimum L_1 -norm minimization, while maintaining good reconstruction accuracy. This approach includes compressive sampling matching pursuit (CoSaMP) method [14], orthogonal matching pursuit (OMP) method [15], iterative hard thresholding method (IHT) [16] and subspace pursuit (SP) method [17].

The majority of the existing greedy recovery methods perform non-regulated selection, leading to the selection of either too few or too many elements, generating a larger reconstruction time and error. Moreover, the sparse signal is estimated through least square minimization and calculating a pseudo-inverse of a matrix at each iteration. Such matrix inversion becomes more intensive as the size of the data increases, which brings high computation cost with each iteration and makes such approaches not suitable for real-time applications.

In order to develop an effective method for face recognition, we propose in this work Gabor wavelet to extract the features from face database followed by the proposed modified KSRC for face classification. Gabor feature extraction method improves the recognition performance and ensures the robustness of the new proposed face recognition method against illumination variation, expression, and geometric asynchronous variation. For dimensionality reduction, we used the supervised kernel locality preserving projections (SKLPP) in the KSRC method. SKLPP takes into consideration the nonlinear within-class structure in high-dimensional feature space. Also, in order to solve sparse recovery problem, we propose a new technique that we called fast compressive sampling matching pursuit (FCoSaMP). A fast and accurate method for greedy recovery methods. Instead of choosing either too few or too many elements, FCoSaMP selects a sufficient number of elements. Moreover, FCoSaMP performs least square minimization iteratively through Sherman-Morrison-Woodbury formula, to avoid large matrix inversion which results in a significant speedup. Despite its complexity, the inverse in each iteration is computed from the given data at the previous iteration in two steps: First, the inverse is updated by adding data corresponding to the added elements. Secondly, after the estimated signal is pruned, the inverse is updated by removing data corresponding to the pruned elements.

The remainder of the paper is organized as follows: The next section represents the proposed feature extraction method. In section 3, we present the KSRC method for

classification and the used SKLPP method for dimensionality reduction. Section 4 introduces the relatively new method to find sparse signal and the new variation based on the CoSaMP known as FCoSaMP. The experimental results and the effectiveness of the proposed method are given in section 5.

2. Gabor Feature extraction

The variations in illumination, facial expressions, and poses represents the major challenges in the face recognition problem. In order to overcome such challenges, it was found that the local features in face images are more robust against such variations [18]. So, a spatial-frequency analysis was found to extract such features. In fact, the wavelet analysis provides a good space-frequency localization. However, Gabor functions provide the optimized resolution in both the spatial and frequency domains [20, 21]. Therefore, the Gabor-wavelets are proposed to be the optimal basis for extracting local features for pattern recognition [18]. The 2D Gabor filter in the spatial domain is defined by the following expression [1, 22, 23]:

$$\Psi_{u,v}(z) = \frac{\|k_{u,v}\|^2}{\sigma^2} e^{-\frac{\|k_{u,v}\|^2 \|z\|^2}{2\sigma^2}} \left[e^{(i k_{u,v} z)} - e^{(-\frac{\sigma^2}{2})} \right] \quad (1)$$

where $z = (x, y)$, u and v is the orientation and scale of the Gabor kernels, and $k_{u,v} = k_v e^{i\phi_u}$ is the wave vector with $k_v = \frac{f_{\max}}{f^v}$ and $\phi_u = \frac{u\pi}{8}$. $f_{\max}$ is the maximum frequency and f is the spacing factor between wavelets in the frequency domain[1].

For feature extraction, a 40 Gabor filter, consisting of eight different orientations $u \in \{0, 1, \dots, 7\}$ and five different scales $v \in \{0, 1, \dots, 4\}$, is usually used to extract multi-orientation and multi-scale features from the given grey-scale face image.

3. Kernel Sparse Representation Classifier

In this section, we introduce the compressed sensing (CS) method, then we briefly review the kernel sparse representation-based classifier (KSRC). Finally, we apply supervised kernel locality preserving projections (SKLPP) for dimensionality reduction.

3.1. Compressed sensing

Let there be a sparse signal $x \in R^N$, with s -sparse, and a measurement system that acquires M linear measurements presented by:

$$y = Ax \quad (2)$$

Where $y \in R^M$ is the measured vector and $A \in R^{M \times N}$ represent the measurement or sensing matrix, with $s < M < N$. A signal x is called s -sparse if x has s nonzero entries. For reconstructing the sparse signal, Donoho originally suggested using L_1 -norm minimization as follows [3]:

$$\hat{x} = \arg \min_x \|x\|_1 \quad s.t. \quad y = Ax \quad (3)$$

L_1 -norm minimization is based on linear programming techniques and requires the condition of Restricted Isometry Property (RIP) on the sensing matrix A to obtain a unique sparse solution [24]. The sensing matrix A satisfies the RIP, if a Restricted Isometry Constant (RIC) δ_s ($0 < \delta_s < 1$) exists, such that

$$(1 - \delta_s) \|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_s) \|x\|_2^2 \quad (4)$$

holds for all s -sparse signals x [25].

3.2. Kernel sparse representation classifier

Assume a set of training data of C -class where each class C_i containing N_c face images with M dimension. Let $A_i \in R^{M \times N_c}$, called dictionary, the data of the i th class, can be represented by:

$$A_i = [a_{i,1}, a_{i,2}, \dots, a_{i,j}, \dots, a_{i,N_c}] \in R^{M \times N_c} \quad (5)$$

where $a_{i,j} \in R^M$, called atom, denotes the j th training data in the i th class. The new dictionary A containing N data from all C classes can be formulated as:

$$A = [A_1, A_2, \dots, A_C] \in R^{M \times N} \quad (6)$$

In KSRC, to make the training data separable, the data are mapped from the input feature space R^M into a high-dimensional kernel feature space H by a nonlinear mapping function ϕ . The dictionary A composed of all N data after the nonlinear mapping ϕ can be formulated as:

$$\phi(A) = [\phi(a_1), \phi(a_2), \dots, \phi(a_N)]^T \in R^{D \times N} \quad (7)$$

Where $\phi(a) \in R^D$ is the transformation of the data of $a_{i,j}$ into a high-dimensional space H , and $D \gg M$ is the dimension of the feature space H .

In KSRC, any new data of test $y \in R^M$, after transformation to kernel feature space H can be expressed with a linear combination of all N training data as follows:

$$\phi(y) = \phi(A)x \quad (8)$$

where $\phi(y) \in R^D$ is the test data in the high-dimensional kernel feature space $\mathbf{H}$. It corresponds to $\mathbf{y}$ in the input feature space. $\mathbf{x} = [x_1, x_2, \dots, x_N] \in R^N$ is a sparse vector where $\mathbf{x}_j$ are the coefficients corresponding to the j th training data $\phi(a_j)$. If the data of test $\mathbf{y}$ represents a part to the i th class, hence the entries of $\mathbf{x}$ are expected to be zero excluding those associated with this class.

$$\mathbf{x} = [0, \dots, 0, x_{i,1}, \dots, x_{i,N_c}, 0, \dots, 0]^T \in R^N \quad (9)$$

In order to find the identity of $\mathbf{y}$, KSRC used the sparse solution $\hat{\mathbf{x}}$ by solving the following stable L_1 -norm minimization problem [26]:

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad s.t. \quad \phi(\mathbf{y}) = \phi(\mathbf{A})\mathbf{x} \quad (10)$$

where $\|\mathbf{x}\|_1$ is the L_1 -norm minimization of $\mathbf{x}$. However, Eq. (10) is even harder to solve. So it is necessary to reduce the dimensionality in feature space $\mathbf{H}$ into low-dimensional subspace. In fact, the dimensionality of the feature space $\mathbf{H}$ is far greater than the original feature space R^M and requires a huge computation. Hence, the system may slow down terribly or run out of memory. Moreover, it has been observed that a large number of features may actually degrade the performance of classifiers. Indeed, the number of training data is small in relation to the number of features [26]. That is why, in order to reduce the dimensionality in kernel feature space $\mathbf{H}$ into low-dimensional subspace we modify the Eq. (8) as follows:

$$\mathbf{P}^T \phi(\mathbf{y}) = \mathbf{P}^T \phi(\mathbf{A})\mathbf{x} \quad (11)$$

$\mathbf{P} \in R^{D \times d}$ represents the transformation matrix, $d \ll D$ is the dimension of the low-dimensional subspace. The given matrix can be expressed in the kernel-based dimensionality reduction as:

$$\mathbf{P} = \phi(\mathbf{A})\mathbf{B} \quad (12)$$

where each column $\mathbf{P}_j$ of $\mathbf{P} = [\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_d]$ is a linear combination of all the training data in space $\mathbf{H}$. $\mathbf{B} = [\beta_1, \beta_2, \dots, \beta_d]$ is the pseudo-transformation matrix and $\beta_j = [\beta_{j,1}, \beta_{j,2}, \dots, \beta_{j,N}]^T$ is the pseudo-transformation vector corresponding to j th transformation vector $\mathbf{P}_j$. Substituting Eq. (11) into Eq. (12), we get

$$(\phi(\mathbf{A})\mathbf{B})^T \phi(\mathbf{y}) = (\phi(\mathbf{A})\mathbf{B})^T \phi(\mathbf{A})\mathbf{x} \quad (13)$$

The inner product of data in the high-dimensional kernel feature space can be computed by Mercer's kernel function. The kernel function must satisfy Mercer's conditions which are continuity, symmetry and being positive definite. Namely, for any data $\mathbf{a}$ and $\mathbf{a}'$, we have

$\phi(a)^T \phi(a') = k(a, a')$, where $k(a, a')$ is a kernel function. Some popular kernels that meet Mercer's condition are linear kernel, polynomial kernel, sigmoid kernel and Gaussian radial basis function (RBF). The RBF function transforms the data to a specific high-dimensional space which is computed by the form of the corresponding kernel function. The RBF kernel is given by the following equation:

$$k(a, a') = e^{\left(-\gamma \|a - a'\|_2^2\right)} \quad (14)$$

where γ is a positive constant called RBF kernel parameter. Consequently, Eq. (13) is transformed to the following expression:

$$B^T k(A, y) = B^T K x \quad (15)$$

where $k(A, y) = \phi(A^T) \phi(y) = [k(A_1, y), k(A_2, y), \dots, k(A_N, y)]^T \in R^N$ and $K = \phi(A^T) \phi(A) \in R^{N \times N}$ is the kernel Gram matrix which is symmetric and positive definite. After zero centralization, the centering kernel gram matrix K is defined by the following relationship:

$$K = \Pi K \Pi \quad (16)$$

The term $\Pi = I - 1_N$ where $I \in R^{N \times N}$ is the identity matrix and $1_N \in R^{N \times N}$ is a matrix with all entries $\frac{1}{N}$. Following [26], the problem of Eq. (10) can be rewritten as:

$$\hat{x} = \arg \min_x \|x\|_1 \quad s.t. \quad B^T k(A, y) = B^T K x \quad (17)$$

Consider the case of noisy data, the following optimization problem of Eq. (17) is modified by the following expression:

$$\hat{x} = \arg \min_x \|x\|_1 \quad s.t. \quad \|B^T k(A, y) - B^T K x\|_2^2 \leq \varepsilon \quad (18)$$

where ε is small positive error. To compute the identity of a test data y , KSRC uses the coding coefficients $\hat{x}$. After solving Eq. (17) or Eq. (18), the representation residual of each class is computed by the following expression:

$$r_i(y) = \arg \min_i \|B^T k(A, y) - B^T K \delta_i(\hat{x})\|_2^2 \quad (19)$$

where δ_i is the characteristic function which is used to select the associated coefficients to the i th class. The characteristic function is given by the following relationship:

$$\delta_i(\hat{x}_j) = \begin{cases} \hat{x}_j & \text{if } y_j = i \\ 0 & \text{otherwise} \end{cases} \quad (20)$$

Thus, based on the representation residual, the class label of the test data is computed by the following expression:

$$label(y) = \arg \min_i r_i(y) \quad (21)$$

In the above equation, if a class achieves the minimal representation residual, the test data is classified into this class. The complete classification procedure of KSRC is summarized in the algorithm 1. This algorithm introduces subspace projection method SKLPP to perform dimensionality reduction and FCoSaMP to solve the L_1 -norm minimization problem.

Algorithm 1: kernel sparse representation-based classifier (KSRC)

Input: A matrix of training data A and a test data y

Step 1: Kernelize of A and y .

Step 2: Get the corresponding pseudo-transformation matrix B using SKLPP.

Step 3: Center the kernel gram matrix K .

Step 4: Normalize the columns of $B^T K x$ and $B^T k(A, y)$ to have unit L_2 -norm.

Step 5: Solve the L_1 -norm minimization problem to get the coefficient vector x using FCoSaMP (Algorithm 3).

Step 6: Compute the residuals $r_i(y) = \arg \min_i \|B^T k(A, y) - B^T K \delta_i(\hat{x})\|_2^2, i = 1, \dots, C$

Output: the class label of y as $label(y) = \arg \min_i r_i(y)$

3.3. Supervised Kernel locality-preserving projections

In KSRC, we mapping the training data from the input feature space R^M into a high-dimensional kernel feature space H by a nonlinear mapping function ϕ . The objective function of KLPP in the kernel feature space is given by the following equation:

$$\min \sum_{i,j=1}^N \|z_i - z_j\|^2 W(i, j) \quad (22)$$

where $z_i = P_\phi^T \phi(a_i)$ is the low-dimensional representation of $\phi(a_i)$ with a projection vector P_ϕ^T and the weight $W(i, j)$ represent the similarity of a_i and a_j . the obtained results after minimizing Eq. (22) is given by the following expression [10]:

$$\sum_{i,j=1}^N \|z_i - z_j\|^2 W(i, j) = 2\beta^T K(D - W)K\beta \quad (23)$$

where $D_{ii} = \sum_j W(i, j)$ is a diagonal matrix, β is a coefficient vector and $K = k(i, j) = \phi(a_i)\phi(a_j)$ is defined as the kernel Gram matrix which is symmetric and positive definite. Thus, the transformation matrix B can be obtained by the following expression:

$$K(D - W)K\beta = \lambda KDK\beta \quad (24)$$

The column vectors $\beta_1, \beta_2, \dots, \beta_d$ be the solutions of Eq. (24). We take d eigenvectors corresponding to the first d largest eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d$. Then, the pseudo-transformation matrix can be represented by $B = [\beta_1, \beta_2, \dots, \beta_d] \in R^{N \times d}$.

For unsupervised KLPP, the weight matrix W is computed if two data are connected in the n nearest neighbours. For supervised KLPP (SKLPP), W is computed if the two data are connected and they belong to the same class. Therefore, the weight matrix $W(i, j)$ in SKLPP is computed by the following expression [10]:

$$W(i, j) = \begin{cases} \phi(a_i)\phi(a_j) & \text{if } a_i \text{ and } a_j \text{ belong to the same class,} \\ 0 & \text{otherwise} \end{cases} \quad (25)$$

4. Fast Compressive Sampling Matching Pursuit

Motivated by the need to reach computationally inexpensive solutions to solve L_1 -norm minimization, we propose in this section a fast compressive sampling matching pursuit (FCoSaMP) for reconstructing sparse signal. First, we review compressive sampling matching pursuit (CoSaMP), then FCoSaMP. Finally, Sherman-Morrison-Woodbury formula to update the inverse of matrix. In this section, the sensing matrix Φ will be taken as the kernel Gram matrix $K = \phi(A^T)\phi(A) \in R^{N \times N}$.

4.1. Compressive Sampling Matching Pursuit (CoSaMP)

Compressive Sampling Matching Pursuit (CoSaMP) is one of the Greedy recovery methods proposed by Needell and Tropp [14] for reconstructing the sparse signal. In each iteration, the CoSaMP selects $2s$ elements and the sparse signal is estimated based on the identified support set Ω through least square minimization. The signal is pruned maintaining only its s largest values and the residual is calculated based on the pruned signal. The previous steps are repeated until a stopping condition is met. The CoSaMP method is summarized in Algorithm 2 and its steps are explained as follows.

- 1) **Identification:** The residual vector r is correlated with the columns of the sensing matrix Φ , forming a signal proxy g .
- 2) **Support Merger:** the support set Ω is obtained by selects $2s$ columns of sensing matrix Φ corresponding to the $2s$ largest absolute values of signal proxy g , where s defines the number of nonzero elements. The indices of the selected elements are united with the identified support set T .
- 3) **Estimation:** Based on the identified support set T , the signal estimate x is formed by using least square minimization. Is typically performed by multiplication the

measurement vector $\mathbf{y}$ by the pseudo-inverse of the nonsingular matrix Φ_T , where $\Phi_T \in R^{d \times k}$ is a sub-matrix containing the selected columns of the sensing matrix Φ at indices from the support set T .

- 4) **Pruning:** After solving the least squares, a $2s$ -sparse estimate is obtained. Then, the $2s$ -sparse vector is pruned to obtain the s largest magnitude elements in the signal estimate and set to zero the rest.
- 5) **Sample Update:** The new residual is calculated based on the pruned signal.

Algorithm 2: Compressive Sampling Matching Pursuit (CoSaMP)

Input: the sensing matrix Φ , Measurement vector $\mathbf{y}$, sparsity level s

Initialize: $\hat{x}^0 \leftarrow 0, r^0 \leftarrow y, i \leftarrow 0$

Repeat

$i \leftarrow i + 1$

$g \leftarrow \Phi * r^{i-1}$ { Form signal proxy }

$\Omega \leftarrow \text{supp}(g_{2s})$ { Identify large components }

$T \leftarrow \Omega \cup \text{supp}(x^{i-1})$ { Merge supports }

$b|_T \leftarrow \Phi_T^\dagger y$ { Signal estimation by least-squares }

$b|_{T^c} \leftarrow 0$

$\hat{x}^i \leftarrow b_s$ { Prune to obtain next approximation }

$r^i \leftarrow y - \Phi \hat{x}^i$ { Update current data }

until halting criterion true

Output: An s -sparse approximation $\hat{x}$ of the target signal

4.2. Proposed Fast Compressive Sampling Matching Pursuit (FCoSaMP):

In each iteration, the FCoSaMP method selects a sufficient number of elements using a double thresholding technique. The number of selected elements is not constant for all iterations. The new indices of the selected elements are merged with the previous ones. Based on the merged support set T , FCoSaMP estimate the sparse signal using the least square minimization. The least square minimization is typically performed through the multiplication of the measurement vector $\mathbf{y}$ by the pseudo-inverse of the nonsingular matrix Φ_T , $\Phi_T^\dagger = (\Phi_T^T \Phi_T)^{-1} \Phi_T^T$, where $\Phi_T \in R^{d \times k}$ is a sub-matrix containing the k selected columns of the

sensing matrix Φ from the support set T . In general, the inverse of the nonsingular matrix $(\Phi_T^T \Phi_T)$ requires $O(k^3)$ operations, which might be heavy to implement in case there is a large number of columns. Take this into consideration, it is possible, using the matrix inversion lemma, to decrease the number of computations from $O(k^3)$ to $O(k^2)$ [27]. Instead of directly forming the pseudo-inverse necessary for least square minimization in each iteration with matrix inversion, FCoSaMP updates the inverse matrix from data in the previous iteration by adding a new columns corresponding to the added elements. FCoSaMP then prunes the signal estimate to keep only its s largest values and update the inverse matrix by removing the columns corresponding to the pruned elements for the next iteration. Finally, a residual is calculated and the previous steps are repeated until a stopping condition is met. In the following discussion, we will denote Φ_t in the i th iteration, with Φ_i . The FCoSaMP method is summarized in Algorithm 3 and its steps are explained in what follows.

1) Identification:

The residual vector r is correlated with the columns of the sensing matrix Φ , forming a signal proxy g .

2) Support Merger:

Selection sufficient number of elements is based on a double thresholding technique and the number of selected elements is not constant for all iterations. First, the support set Ω is obtained by selects $2s$ columns of sensing matrix Φ corresponding to the $2s$ largest absolute values of signal proxy g , where s defines the number of nonzero elements. Then, the elements which absolute values are larger than the threshold $\frac{\|g_\Omega - \bar{g}_\Omega\|}{\sqrt{s}}$ are selected from the support set Ω , and their newly indices of the selected elements are added to the merged support set T . The g_Ω containing the columns of the signal proxy g from the support set Ω and $\bar{g}_\Omega$ is the average of g_Ω . It is clear that the shaped threshold $\frac{\|g_\Omega - \bar{g}_\Omega\|}{\sqrt{s}}$, take into consideration the size change and it's mean in the same time. This dynamic smart threshold auto-adapts itself to generate optimal value in each time.

3) Estimation:

Based on the identified support set T , the signal estimate x is formed by using least square minimization. In the i th iteration, let us denote $\Phi_i \in R^{d \times k}$ a nonsingular matrix has full

column rank, $\text{rank}(\Phi_i) = k$, and $U \in R^{d \times u}$ a matrix containing u newly columns. In FCoSaMP, least square minimization is performed iteratively evading large matrix inversion, which results in significant complexity reduction. Then we are interested in updating the matrix $B_i = (\Phi_i^T \Phi_i)^{-1}$ from the previous iteration $B_{i-1} = (\Phi_{i-1}^T \Phi_{i-1})^{-1}$ by adding the newly selected columns U to Φ_{i-1} , without performing the matrix inversion in each iteration. The matrix Φ_i can be expressed as $\Phi_i = [\Phi_{i-1} \ U]$.

4) Pruning:

After solving the least squares and Φ_i is updated with the new columns, the support set T is updated by removing the columns to obtain the s largest magnitude elements in the signal estimate and set the rest to zero. In the i th iteration, let us denote Φ_p the nonsingular matrix after removing the pruned columns, and P a matrix containing p columns, which will be removed from Φ_i to obtain Φ_{i+1} . This is necessary for the iterative structure of the method, since new columns will be added in the next iteration. The matrix Φ_i can be expressed as $\Phi_i = [\Phi_p \ P]$. Then, for the next iteration, we are interested in updating the matrix $B_p = (\Phi_p^T \Phi_p)^{-1}$ from the matrix $B_i = (\Phi_i^T \Phi_i)^{-1}$ by removing the columns corresponding to the pruned elements P from Φ_i .

5) Sample Update:

The new residual is calculated based on the pruned estimated signal and the measurement vector y .

$$r = y - \Phi \hat{x} \quad (26)$$

Algorithm 3 summarizes the FCoSaMP method. The function $\text{supp}(\cdot)$ returns the index set of the s largest absolute values of the elements of its argument vector. The function $Ls(\cdot)$ retains only the s elements with the largest absolute values and sets the rest to zero. The stopping condition of those algorithms occurs if the norm of the residual is less than ϵ_1 , if the difference between the norms of the residuals in two successive iterations is less than ϵ_2 , or a maximum of i th iterations is performed.

4.3. Update inverse matrix

Let us denote $\Phi \in R^{d \times k}$ is symmetric positive definite matrix and $k < d$. In this section, we are interested in updating the matrix inverse B by adding a symmetric matrix U to Φ or removing a symmetric matrix P from Φ . This matrix inverse has been calculated in various

Algorithm 3: Fast Compressive Sampling Matching Pursuit (FCoSaMP)

Input: the sensing matrix Φ , Measurement vector y , sparsity level s

Initialize: $\hat{x} \leftarrow 0, r \leftarrow y, T^0 \leftarrow [], i \leftarrow 0$

Repeat

$i \leftarrow i + 1$

$g \leftarrow \Phi * r$

{ Form signal proxy }

$\Omega \leftarrow \text{supp}(g_{2s})$

{ Identify large components }

$\Delta \leftarrow \{g_k \mid \frac{\|g_\Omega - \bar{g}_\Omega\|}{\sqrt{s}} \geq \frac{\|g_k - \bar{g}_\Omega\|}{\sqrt{s}}, k \in \Omega\}$

{ Indices of elements in Ω larger than or equal

to $\frac{\|g_\Omega - \bar{g}_\Omega\|}{\sqrt{s}}$ }

$u \leftarrow \Delta \setminus T^{i-1}$

{ Indices of newly added column }

$T^i \leftarrow [T^{i-1} \ u]$

{ Merge supports }

if $i == 1$:

$B \leftarrow (\Phi_u^T \Phi_u)^{-1}$

$\Phi^\dagger \leftarrow B \Phi_u^T$

else:

$U \leftarrow \Phi_u$

$B \leftarrow \text{addcol}(\Phi_{T^{i-1}}, B, U, T^i)$

{ Update B by adding new columns using algorithm 4 }

$\Phi^\dagger \leftarrow B \Phi_{T^i}^T$

end if

$b|_T \leftarrow \Phi^\dagger y$

{ Signal estimation by least-squares }

$b|_{T^c} \leftarrow 0$

$\hat{x} \leftarrow Ls(b)$

{ Prune to obtain next approximation }

$p \leftarrow T^i \setminus \text{supp}(b_s)$

{ Pruned indices }

if $|p| > 0$:

$P \leftarrow \Phi_p$

{ Pruned columns }

$B \leftarrow \text{remocol}(\Phi_{T^i}, B, P, T^i)$

{ Update B by removing pruned columns using algorithm 5 }

$T^i \leftarrow T^i \setminus p$

{ Update support set by removing pruned indices }

end if

$r \leftarrow y - \Phi \hat{x}$

{ Update current data }

until halting criterion true

Output: An s -sparse approximation $\hat{x}$ of the target signal

disciplines with different applications, derived from the Sherman-Morrison formula [28]. The Sherman-Morrison-Woodbury formula relate the inverse of a matrix after a small rank

perturbation to the inverse of the original matrix [29]. The Sherman-Morrison-Woodbury formula is the generalization to a rank q modification to B and can be expressed by the following expression:

$$(B + VCV^T)^{-1} = B^{-1} - B^{-1}V(C^{-1} + V^TB^{-1}V)^{-1}V^TB^{-1} \quad (27)$$

Here $B = (\Phi^T\Phi) \in R^{k \times k}$ is square nonsingular matrix, $V \in R^{k \times q}$, $C \in R^{q \times q}$ and $V^T \in R^{q \times k}$ with $q \ll k$. Eq.(27) is useful in situations where q is much smaller than k and the structure of B is “nice” so that the effort involved in evaluating the correction $B^{-1}V(C^{-1} + V^TB^{-1}V)^{-1}V^TB^{-1}$ relative to the effort involved in inverting a general $k \times k$ matrix is small [29]. The Sherman-Morrison-Woodbury formula can be formulated using block-partitioned matrix inversion [29]. Let the symmetric matrix R be partitioned as:

$$R = \begin{bmatrix} B & V \\ V^T & C \end{bmatrix} \quad (28)$$

In order to find the inverse of a partitioned matrix R , we use the general result provided by the inverse of a partitioned matrix [29], which is written by the following expression [30, Appendix B]:

$$R^{-1} = \begin{bmatrix} F_{11}^{-1} & -F_{11}^{-1}VC^{-1} \\ -C^{-1}V^TF_{11}^{-1} & F_{22}^{-1} \end{bmatrix} \quad (29)$$

where

$$F_{11} = B - VC^{-1}V^T \quad (30)$$

$$F_{22} = C - V^TB^{-1}V \quad (31)$$

To find the inverse of a matrix F_{11} and F_{22} , we use the Sherman-Morrison-Woodbury formula:

$$F_{11}^{-1} = B^{-1} + B^{-1}VF_{22}^{-1}V^TB^{-1} \quad (32)$$

$$F_{22}^{-1} = C^{-1} + C^{-1}V^TB^{-1}VC^{-1} \quad (33)$$

4.3.1. Updating inverse matrix by adding columns

Assuming that we receive new data in i th iteration, let U be a matrix containing the newly added columns. The new matrix Φ_i is positive definite and corresponds to the newly added selected columns U to the previous matrix Φ_{i-1} , $\Phi_i = [\Phi_{i-1} \ U]$. The matrix $B_i = (\Phi_i^T\Phi_i)$ corresponding to the new matrix Φ_i can be expressed by the following expression:

$$B_i = B_{i-1} + UU^T = \Phi_{i-1}^T \Phi_{i-1} + UU^T = \begin{bmatrix} \Phi_{i-1}^T \\ U^T \end{bmatrix} \begin{bmatrix} \Phi_{i-1} & U \end{bmatrix} \quad (34)$$

$$B_i = \begin{bmatrix} \Phi_{i-1}^T \Phi_{i-1} & \Phi_{i-1}^T U \\ U^T \Phi_{i-1} & U^T U \end{bmatrix} \quad (35)$$

To find the inverse of a partitioned matrix B_i , we use the result provided in Eq. (29). The partitioned matrix B_i^{-1} can be expressed by the following expression:

$$B_i^{-1} = \begin{bmatrix} F_{11}^{-1} & -B_{i-1}^{-1} \Phi_{i-1}^T U D^{-1} \\ -D^{-1} U^T \Phi_{i-1} B_{i-1}^{-1} & D^{-1} \end{bmatrix} \quad (36)$$

where

$$F_{11}^{-1} = B_{i-1}^{-1} + B_{i-1}^{-1} \Phi_{i-1}^T U D^{-1} U^T \Phi_{i-1} (B_{i-1}^{-1})^T \quad (37)$$

$$D = U^T U - U^T \Phi_{i-1} B_{i-1}^{-1} \Phi_{i-1}^T U \quad (38)$$

The method of updating the inverse of a partitioned matrix by adding columns is summarized in Algorithm 4.

Algorithm 4: Updating inverse matrix by adding columns

Input: the sensing matrix $\Phi_{T^{i-1}}$, the inverse matrix B , the matrix containing newly columns U , the support set T^i

$$f_1 \leftarrow \Phi_{T^{i-1}}^T U$$

$$f_2 \leftarrow B f_1$$

$$D \leftarrow (U^T U - f_1^T f_2)^{-1}$$

$$f_3 \leftarrow f_2 D$$

$$F_{11} \leftarrow B + f_2 D f_2^T$$

$$B \leftarrow \begin{bmatrix} F_{11} & -f_3 \\ -f_3^T & D \end{bmatrix}$$

change each column $u \in U$ and the corresponding row of B to correct position with respect to T^i

Output: B

4.3.2. Updating inverse matrix by removing columns

Let $\mathbf{P}$ is a matrix containing p columns, we consider the case when we need to remove a p columns from matrix Φ_i to obtain Φ_p . The matrix Φ_i can be expressed as $\Phi_i = [\Phi_p \ P]$. In i th iteration, we have the inverse of a matrix $B_i = (\Phi_i^T \Phi_i)$ and we want to obtain the inverse of a matrix $B_{i+1} = B_p = (\Phi_p^T \Phi_p)$ for the next iteration. In initial permute the j th column and the corresponding j th row to last column and last row of Φ_i . The matrix B_i can be expressed by the following expression:

$$B_i = \begin{bmatrix} \Phi_p^T \Phi_p & \Phi_p^T P \\ P^T \Phi_p & P^T P \end{bmatrix} \quad (39)$$

And the inverse of a partitioned matrix B_i can be expressed by the following expression:

$$B_i^{-1} = \begin{bmatrix} F_{11}^{-1} & -B_p^{-1} \Phi_p^T P D^{-1} \\ -D^{-1} P^T \Phi_p B_p^{-1} & D^{-1} \end{bmatrix} \quad (40)$$

where

$$F_{11}^{-1} = B_p^{-1} + B_p^{-1} \Phi_p^T P D^{-1} P^T \Phi_p (B_p^{-1})^T \quad (41)$$

Let $Z = B_p^{-1} \Phi_p^T P$. Eq. (41) is modified by the following expression:

$$F_{11}^{-1} = B_p^{-1} + Z D^{-1} Z^T \quad (42)$$

and we can obtain the matrix B_p^{-1} for the next iteration by the following expression:

$$B_p^{-1} = F_{11}^{-1} - Z D^{-1} Z^T \quad (43)$$

The method of updating the inverse of a partitioned matrix by removing columns is summarized in Algorithm 5.

5. Results and Discussion

In this section, the performance of our proposed method is evaluated by a series of experiments on four face databases: ORL [31], Yale [32], Extended Yale B [33] and the GT [34] face databases.

The ORL face database: The ORL face database contains 400 face images of 40 persons with 10 grayscale images per person under varying pose, illumination and facial expression. Fig.1. represents some images of one person in this dataset. Each image is resized to a resolution of 64×64 pixels. For each person, we randomly select half of the images for training, the other half is used for testing, and we will repeat the same experiments ten times to take the average accuracy and computation time.

Algorithm 5: Updating inverse matrix by removing columns

Input: the sensing matrix Φ_{T^i} , the inverse matrix B , the matrix containing pruned columns P , the support set T^i

permute each column $p \in P$ and the corresponding row of Φ_{T^i} to last column and last row.

$t \leftarrow \text{cardinality of set } T^i$

$p \leftarrow \text{cardinality of set } P$

$F_{11} \leftarrow B[0:t-p, 0:t-p]$

$D \leftarrow B[t-p:t, t-p:t]$

$f_1 \leftarrow -B[0:t-p, t-p:t]$

$f_2 \leftarrow f_1 D^{-1}$

$f_3 \leftarrow f_2 D$

$B_p \leftarrow F_{11} - f_3 f_2^T$

Output: B_p

Fig.1. Some images of one person in ORL face database.

The Yale face database: The Yale database contains 165 face images of 15 persons with 11 grayscale images per person under varying illumination and facial expression. Fig. 2 represents some images of one person in this dataset. Each image is first cropped and resized to a resolution of 64×64 pixels. For each person, we randomly select six of the images for training and the rest are used for testing, and we will repeat the same experiments ten times to take the average accuracy and computation time.

Fig.2. Some images of one person in Yale face database.

The Extended Yale face database B: The Extended Yale face database B contains 2432 face images of 38 persons, with 64 grayscale images per person under varying pose and illumination. Fig.3 represents some images of one person in this dataset. Each image is first cropped and resized to a resolution of 64×64 pixels. For each person, we randomly select the half of the images for training, the other half is used for testing, and we will repeat the same experiments ten times to take the average accuracy and computation time.

Fig.3. Some images of one person in Extended Yale face database B with luminosity variation.

The Georgia Tech face database: The Georgia Tech (GT) contains 750 face images of 50 persons, with 15 color images per person under varying pose, expression and illumination. Fig.4 represents some images of one person in this dataset. Each image is first changed to grayscale, and then cropped and resized to a resolution of 64×64 pixels. For each person, we randomly select eight of the images for training and the rest are used for testing, and we will repeat the same experiments ten times to take the average accuracy and computation time.

Fig.4. Some images of one person in GT face database.

For all face databases, we randomly choose the half of the images for training and the other half is used for testing. Each 10 times we will repeat the same experiments to take the average accuracy and computation time. This technique has the merit that randomly choosing the training set ensures that the classification results will be unbiased [4]. For each face databases, the same training and test data selections are used by all the methods to guarantee unbiased comparisons.

The proposed method has two main components: In the first component, the Gabor-wavelets method is used to extract face features. In our implementation the parameters of Gabor filters are set as $u \in \{0, 1, \dots, 7\}$, $v \in \{0, 1, \dots, 4\}$, $\sigma = \pi$, $f_{\max} = \frac{\pi}{2}$, $f = \sqrt{2}$ and the features are down-sampled by a factor $\rho = 4$. In the second component, KSRC method based

KSLPP for dimensionality reduction and FCoSaMP for solving the L_1 -norm minimization problem. In our implementation KSRC, the dimensionality of kernel space was less than the number of training data and the n nearest neighborhood in KSLPP is chosen as $Nc - 1$, where Nc represents the training data number per class. The kernel function adopted is the RBF kernel. The kernel parameter γ is set by the median value of $\frac{1}{(\|x_i - \bar{x}\|_2^2)}$, where $i = 1 \dots N$ represents the training data number and $\bar{x}$ is the average of all training data.

In this experiment, we analyze and compare the performance of our method against other state-of-the-art methods. First, we test the dimensionality reduction using KSLPP against other related methods, KPCA, KFDA and Isomap. Then, we test the L_1 -norm minimization problem using FCoSaMP against other related methods CoSaMP, OMP and LASSO. All of the experimental results were simulated using Python by a PC with Intel(R) Core(TM) i5 CPU 2.7GHz and 4 Go of memory.

5.1. Comparison of different dimensions reduction methods

In order to further investigate the recognition performance of all methods, we compare the performance of different dimensionality reduction methods. KSRC technique is used for classification, applying FCoSaMP. Fig. 5-8 respectively depicts the recognition rate and time using different dimensionality reduction methods applied to Gabor features on the ORL, Yale, Extended Yale and GT face databases. In each figure, Dimension indicates the number of features of each face reduced by KSLPP, KPCA, KFDA and Isomap. It is noted that the dimension in KFDA is necessarily strictly less than the number of classes.

Table 1 shows the best recognition rates of four face databases and the corresponding dimensions and time. From Table 1, it can be noted that compared to other methods except for the Isomap technique on the Extended Yale face database B, KSLPP achieve better recognition rates using smaller number of dimension and less time. Nevertheless, Isomap is not stable since it fails to achieve best recognition rates using smaller number of dimension and less time on other face databases. This fact demonstrates that the SKLPP used in our method can indeed successfully improve the recognition effectiveness because SKLPP takes into consideration the nonlinear within-class structure in kernel feature space. On the ORL database, SKLPP and PCA achieve better recognition rates using smaller number of dimension. We can observe that SKLPP need about 1.99 seconds to achieve a recognition rate 99.15 % at 50 features, KPCA need about 2.91 seconds to achieve a recognition rate of

99.15% at 50 features. SKLPP needs only about 0.01 seconds to represent and recognize one test data on this database. On the other hand KPCA needs only about 0.015 seconds to represent and recognize one test data on this database.

Fig.5. Recognition rate and time using different dimensionality reduction techniques on the ORL database: (a) Recognition rate, (b) time

Fig.6. Recognition rate and time using different dimensionality reduction techniques on the Yale database: (a) Recognition rate, (b) time

Fig.7. Recognition rate and time using different dimensionality reduction techniques on the Extended Yale database B: (a) Recognition rate, (b) time

Fig.8. Recognition rate and time using different dimensionality reduction techniques on the GT database: (a) Recognition rate, (b) time

Table 1: Average of the best recognition rates and time of dimensions reduction techniques

Database	Algorithms	Features	Recognition %	Time (s)
ORL	KPCA	50	99.15	2.91
	KFDA	35	99.1	2.04
	Isomap	100	99.1	3.43
	SKLPP	50	99.15	1.99
Yale	KPCA	60	97.35	0.9
	KFDA	12	96.39	1.07
	Isomap	45	97.22	1.42
	SKLPP	55	97.35	0.92
Extended Yale B	KPCA	700	87.99	108.26
	KFDA	37	83.5	80.37
	Isomap	400	88.32	93.35
	SKLPP	450	88.29	57.36
GT	KPCA	270	93.07	5.94
	KFDA	50	93.04	6.19
	Isomap	240	93.15	7.05
	SKLPP	150	93.15	4.71

5.2. Comparison of CS recovery methods

Next, we compare each component of our method separately against corresponding components in other methods. We demonstrate the benefit of using KSRC to classification based SKLPP for feature reduction and FCoSaMP for solving L_1 -norm minimization. Fig. 9-12 respectively depicts the recognition rate and time using different CS reconstruction methods FCoSaMP, CoSaMP, OMP and Lasso on the ORL, Yale, Extended Yale and GT face databases. The proposed FCoSaMP has obtained better average recognition rate than the other methods, and the average time of FCoSaMP has also been much less than those of the other methods, as shown in Table 2.

However, using FCoSaMP takes the least reconstruction time of about 57.36 seconds at 450 features on the Extended Yale face database B, compared to 124.4 seconds for CoSaMP, and about 77.84 seconds for OMP. FCoSaMP needs only about 0.05 seconds to represent and recognize one test data on this database. On the other hand OMP needs only about 0.06 seconds and CoSaMP needs only about 0.1 seconds to represent and recognize one test data on this database. This shows that FCoSaMP is capable of accurate reconstruction, considerably faster than other methods.

Fig.9. Recognition rate and time using different CS recovery algorithms on the ORL database: (a) Recognition rate, (b) time

Fig10. Recognition rate and time using different CS recovery algorithms on the Yale database: (a) Recognition rate, (b) time

(a) (b)
Fig.11. Recognition rate and time using different CS recovery algorithms on the Extended Yale database B: (a) Recognition rate, (b) time

(a) (b)
Fig.12. Recognition rate and time using different CS recovery algorithms on the GT database: (a) Recognition rate, (b) time

Table 2: Average of the best recognition rates and time of CS recovery techniques

Database	Algorithms	Features	Recognition %	Time (s)
ORL	CoSaMP	80	99.15	3.67
	OMP	120	99.1	2.46
	Lasso	60	99.15	2.8
	FCoSaMP	50	99.15	1.99
Yale	CoSaMP	70	97.23	0.98
	OMP	45	97.1	0.97
	Lasso	35	96.95	1.16
	FCoSaMP	55	97.34	0.92
Extended Yale B	CoSaMP	500	88.13	124.4
	OMP	500	87.18	77.84
	Lasso	250	87.18	57.92
	FCoSaMP	450	88.29	57.36
GT	CoSaMP	180	93.13	7.02
	OMP	210	92.89	6.71
	Lasso	150	92.57	4.67
	FCoSaMP	150	93.15	4.71

6. Conclusion

This paper proposes an efficient and fast algorithm for face recognition robust against position, luminosity and expression variation. This algorithm is splitted into two phase, feature extraction based on Gabor-wavelets method and face classification based on KSRC technique. The Gabor-wavelets method are used to improve the recognition performance and

ensures the robustness of the proposed algorithm. Indeed, to reduce the dimensionality in the KSRC method, a SKLPP method is used which attempts to preserve local geometric relations of the within-class samples in nonlinear kernel feature space. In other hand, a FCoSaMP algorithm based on the CoSaMP technique is newly proposed to solve the L_1 -norm minimization. The proposed technique estimates the sparse signal using least square minimization by updating the inverse of matrix through Sherman-Morrison-Woodbury formula which makes the FCoSaMP has a faster computation time compared to other CS recovery algorithms. Indeed the selection of a sufficient number of elements by using a double thresholding technique increase the recognition rate.

Many experiments and simulations are performed to compare the proposed algorithm to other recent related works in the literature actually in use. Experimental results on four face databases show that our proposed algorithm achieves a good performance for both face recognition and time computation. In fact the FCoSaMP boosts the entire recognition process and achieves a significant speedup and better recognition rate compared to other CS recovery algorithms, including OMP, Lasso and CoSaMP methods. The FCoSaMP runs 1.8 times faster than CoSaMP in ORL face databases and more than 2 times faster than those of CoSaMP in Extended Yale B databases.

ABBREVIATIONS

KSRC	Kernel Sparse Representation based-Classification
SRC	Sparse Representation based-Classification
CS	Compressed Sensing
SKLPP	Supervised Kernel Locality Preserving Projections
KPCA	Kernel Principle Component Analysis
KFDA	Kernel Fisher Discriminant Analysis
LLE	Locally Linear Embedding
FCoSaMP	Fast Compressive Sampling Matching Pursuit
CoSaMP	Compressive Sampling Matching Pursuit
OMP	Orthogonal Matching Pursuit
IHT	Iterative Hard Thresholding
SP	Subspace Pursuit
Lasso	Least Absolute shrinkage and selection operator
GPDR	Gradient Projection for Sparse Representation

RKHS	Reproducing Kernel Hilbert Space
------	----------------------------------

References

1. M. Lades, J.C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R.P. Wurtz and W. Konen, "Distortion invariant object recognition in the dynamic link architecture," *IEEE Trans. Comput.* vol. 42, no. 3, pp. 300-311, (1993).
2. L. Wiskott, J.M. Fellous, N. Kruger and C. von der Malsburg, "Face recognition by elastic bunch graph matching," *IEEE Trans. Pattern Analysis Machine Intelligent*, vol. 19, no. 7, pp. 775-779, (1997).
3. D.L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289-1306, (2006).
4. J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Analysis Machine Intelligent*, vol. 31, no. 2, pp. 210-227, (2009).
5. Z. Fan, D. Zhang, X. Wang, Q. Zhu and Y. Wang, "Virtual dictionary based kernel sparse representation for face recognition," *Pattern Recognition*, vol. 76, pp. 1-13, (2018).
6. B. Schölkopf, A.J. Smola, and K.R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299-1319, (1998).
7. S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.R. Müller, "Fisher discriminant analysis with kernels", *Proc. IEEE Signal Process. Soc. Workshop Neural Netw. Signal Process.*, pp. 41-48, (1999).
8. J. Tenenbaum, V. de Silva, and J.C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319-2323, (2000).
9. S.T. Roweis and L.K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323-2326, (2000).
10. J. Cheng, Q. Liu, H. Lu and Y. Chen, "Supervised kernel locality preserving projections for face recognition," *Neurocomputing*, vol. 67, pp. 443-449, (2005).
11. R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B*, vol. 58, no. 1, pp. 267-288, (1996).

12. D.L. Donoho and Y. Tsaig, "Fast solution of ℓ_1 -norm minimization problems when the solution may be sparse," *IEEE Trans. Inf. Theory*, vol. 54, no.11, pp. 4789-4812, (2008).
13. T. Blumensath and M.E. Davies, "Gradient pursuits," *IEEE Trans. Sig. Process.*, vol. 56, no. 6, pp. 2370-2382, (2008).
14. D. Needell and J.A. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301-321, (2008).
15. J.A. Tropp and A.C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol.53, no.12, pp. 4655-4666, (2007).
16. T. Blumensath and M.E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265-274, (2008).
17. W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2230-2249, (2009).
18. L. Shen, L. Bai, "A review on Gabor wavelets for face recognition," *Pattern Anal. Appl.*, vol. 9, pp. 273-292, (2006).
19. Z. B. Lahaw, D. Essaidani, H. Seddik, "Robust Face Recognition Approaches Using PCA ICA LDA Based on DWT and SVM Algorithms", *International Conference on Telecommunications and Signal Processing*, pp. 1-5, (2018).
20. D. Gabor, "Theory of communication. Part 1: the analysis of information," *J. Inst. Electr. Eng. Part III: Radio Commun. Eng.*, vol. 93, no. 26, pp. 429-441, (1946).
21. J. G. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," *J. Opti. Socie. Amer. A*, vol. 2, no. 7, pp. 1160-1169, (1985).
22. J. G. Daugman, "Two-dimensional spectral analysis of cortical receptive field profiles," *Vision Research*, vol. 20, no. 10, pp. 847-856, (1980).
23. C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition," *IEEE Trans. Image Process.*, vol. 11, no. 4, pp. 467-476, (2002).
24. E.J. Candes and M.B. Wakin, "An introduction to compressive sampling," *IEEE Sig. Process. Magazine*, vol. 25, no. 2, pp. 21-30, (2008).

25. E.J. Candes, J.K. Romberg and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements", *Commun. Pure Appl. Math.*, vol. 59, no. 8, pp. 1207-1223, (2006).
26. Y. Xu, Z. Fan and Q. Zhu, "Feature space-based human face image representation and recognition," *Optical Engineering*, vol. 51, no. 1, (2012).
27. F. Rosário, F.A. Monteiro and A. Rodrigues, "Fast Matrix Inversion Updates for Massive MIMO Detection and Precoding," *IEEE Signal Process. Letters*, vol. 23, no. 1, pp. 75-79, (2016).
28. M.A. Woodbury, "Inverting modified matrices," *Memorandum Report 42, Statistical Research Group, Princeton*, (1950).
29. W.W. Hager, "Updating the Inverse of a Matrix," *SIAM Review*, vol. 31, no. 2, pp. 221-239, (1989).
30. M.J. Beal, "Variational algorithms for approximate Bayesian inference," *PhD thesis, University of Cambridge*, (2003).
31. F.S. Samiria, A.C. Harter, "Parameterisation of a stochastic model for human face identification," *Proceedings of the Second IEEE workshop Appl. Comput. Vis.*, pp. 138–142, (1994).
32. P.N. Bellhumer, J. Hespanha, D.J. Kriegman, "Eigen faces vs fisher faces: recognition using class specific linear projection," *IEEE Trans. Pattern Analysis Machine Intelligent*, vol. 17, no. 7, pp. 711–720, (1997).
33. A.S. Georgiades, P.N. Belhumeur, D.J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Analysis Machine Intelligent*, vol. 23, no. 6, pp. 643-660, (2001).
34. Y. Xu, Z. Zhang, G. Lu, J. Yang, "Approximately symmetrical face images for image preprocessing in face recognition and sparse representation based classification," *Pattern Recognition*, vol. 54, pp. 68–82, (2016).

1. FIGURE LEGEND/CAPTIONS

Figure 1. Some images of one person in ORL face database.

Figure 2. Some images of one person in Yale face database.

Figure 3. Some images of one person in Extended Yale face database B with luminosity variation.

Figure 4. Some images of one person in GT face database.

Figure 5. Recognition rate and time using different dimensionality reduction techniques on the ORL database: **(a)** Recognition rate, **(b)** time

Figure 6. Recognition rate and time using different dimensionality reduction techniques on the Yale database: **(a)** Recognition rate, **(b)** time

Figure 7. Recognition rate and time using different dimensionality reduction techniques on the Extended Yale database B: **(a)** Recognition rate, **(b)** time

Figure 8. Recognition rate and time using different dimensionality reduction techniques on the GT database: **(a)** Recognition rate, **(b)** time

Figure 9. Recognition rate and time using different CS recovery algorithms on the ORL database: **(a)** Recognition rate, **(b)** time

Figure 10. Recognition rate and time using different CS recovery algorithms on the Yale database: **(a)** Recognition rate, **(b)** time

Figure 11. Recognition rate and time using different CS recovery algorithms on the Extended Yale database B: **(a)** Recognition rate, **(b)** time

Figure 12. Recognition rate and time using different CS recovery algorithms on the GT database: **(a)** Recognition rate, **(b)** time

2. DECLARATION

- CONSENT FOR PUBLICATION

Not applicable

- AVAILABILITY OF DATA AND MATERIAL

Please contact author for data requests.

- COMPETING INTERESTS

The authors declare that they have no competing interests.

- FUNDING

Not applicable

- AUTHORS' CONTRIBUTIONS

Zied Bannour Lahaw: coding and writing

Pr. Hassene Seddick: supervision and correction

- ACKNOWLEDGEMENTS

The author is grateful to the anonymous reviewers and the associate editor for their valuable comments, which have greatly helped to improve this work.

Figures

Figure 1

Some images of one person in ORL face database.

Figure 2

Some images of one person in Yale face database.

Figure 3

Some images of one person in Extended Yale face database B with luminosity variation.

Figure 4

Some images of one person in GT face database.

Figure 5

Recognition rate and time using different dimensionality reduction techniques on the ORL database: (a) Recognition rate, (b) time

Figure 6

Recognition rate and time using different dimensionality reduction techniques on the Yale database: (a) Recognition rate, (b) time

Figure 7

Recognition rate and time using different dimensionality reduction techniques on the Extended Yale database B: (a) Recognition rate, (b) time

Figure 8

Recognition rate and time using different dimensionality reduction techniques on the GT database: (a) Recognition rate, (b) time

Figure 9

Recognition rate and time using different CS recovery algorithms on the ORL database: (a) Recognition rate, (b) time

Figure 10

Recognition rate and time using different CS recovery algorithms on the Yale database: (a) Recognition rate, (b) time

Figure 11

Recognition rate and time using different CS recovery algorithms on the Extended Yale database B: (a) Recognition rate, (b) time

Figure 12

Recognition rate and time using different CS recovery algorithms on the GT database: (a) Recognition rate, (b) time