Rapid Face Recognition Using Hashing

Qinfeng Shi
Australian National University, and NICTA
Canberra, Australia

Hanxi Li
Australian National University, and NICTA
Canberra, Australia

Chunhua Shen
NICTA, and Australian National University
Canberra, Australia

Abstract

We propose a face recognition approach based on hashing. The approach yields comparable recognition rates with the random ℓ_1 approach [18], which is considered the state-of-the-art. But our method is much faster: it is up to 150 *times faster than [18] on the YaleB dataset. We show that with hashing, the sparse representation can be recovered with a high probability because hashing preserves the restrictive isometry property. Moreover, we present a theoretical analysis on the recognition rate of the proposed hashing approach. Experiments show a very competitive recognition rate and significant speedup compared with the state-of-the-art.*

1. Introduction

Face recognition often suffers from high dimensionality of the images as well as the large number of training data. Typically, face images/features are mapped to a much lower dimension space (*e.g.*, via down-sample, or linear projection), in which the important information is hopefully preserved. Classification models are then trained on those low-dimensional features. Recently, Wright *et al.* [18] propose a random ℓ_1 minimization approach on sparse representations, which exploits the fact that the sparse representation of the training image indices space helps classification and is robust to noises and occlusions. However, the ℓ_1 minimization in [18] has a computational complexity $O(d^2 n^{3/2})$, where d is the number of measurements and n is the size of the training image set. This makes computation expensive for large-scale datasets. Moreover, a large dense random matrix with size of d by n has to be generated beforehand and stored during the entire processing period. We propose hashing to facilitate face recognition, which has a complexity only of $O(dn)$. Evaluated on the YaleB dataset, the proposed method is up to 150 times faster than the method in [18]. We further show an efficient way to compute hashing matrix implicitly, so that the procedure is potentially applicable to online computing, parallel computing and embedded hardware.

In summary, our main contributions include:

- We discover the connection between hashing kernels and compressed sensing. Existing works on hash kernels [13, 14, 16] use hashing to perform feature reduction with theoretical guarantees that learning in the reduced features space gains much computational power without any noticeable loss of accuracy. The deviation bound and Rademacher margin bound are independent to the line of compressed sensing. Whereas we show the other side of the coin—hashing can actually be viewed as a measurement matrix in compressed sensing, which explains asymptotically no information loss. Also we provide both a theoretical guarantee and empirical evidence that recovering the original signal is possible.

- We apply hashing in the context of compressed sensing to rapid face recognition due to sparse signal recovery. Our experiments show that the proposed method achieves competitive accuracies compared with (if not better than) state-of-the-art in [18, 19]. Yet the proposed hashing with orthogonal matching pursuit is much faster (up to 150 times) than [18, 19].

- We further present bounds on hashing signal recovery rates and face recognition rates for the proposed algorithms.

We briefly review the related work in Section 2, and then introduce two variants of hashing methods for face recognition in Section 3. The theoretical analysis in Section 4 gives justification to our methods, and experimental results in Section 5 demonstrate the excellence of the proposed methods in practice.

2. Related work

Given the abundant literature on face recognition, we only review the work closest to ours.

2.1. Facial features

Inspired by the seminal work of Eigenface using principal component analysis (PCA), learning a meaningful distance metric has been extensively studied for face recognition. These methods try to answer the question that which features of faces are the most informative or discriminative for identifying a face from another. Eigenface using PCA, Fisherface using linear discriminant analysis (LDA), Laplacianface using locality preserving projection (LPP) [9] and nonnegative matrix factorization all belong to this category. These methods project the high-dimensional image data into a low-dimensional feature space. The main justification is that typically the face space has a much lower dimension than the image space (represented by the number of pixels in an image). The task of recognizing faces can be performed in the lower-dimensional face space. These methods are equivalent to learn a Mahalanobis distance as discussed in [17]. Therefore algorithms such as large-margin nearest neighbor (LMNN) [17] can also be applied. Kernelized subspace methods such as kernel PCA and kernel LDA have also been applied for better performances.

2.2. Compressed sensing

Compressive sensing (CS) [6, 4] addresses that if a signal can be compressible in the sense that it has a sparse representation in some basis, then the signal can be reconstructed from a limited number of measurements. Several reconstruction approaches have been successfully presented. The typical algorithm in [4] is to use the so-called ℓ_1 minimization for an approximation to the ideal non-convex ℓ_0 minimization. Yang *et al.* [19, 18] apply CS to face recognition, that is, randomly mapping the down-sampled training face images to a low dimensional space and then using ℓ_1 minimization to reconstruct the sparse representation. The person identity can then be predicted via the minimal residual among all candidates. Unfortunately, ℓ_1 minimization for large matrices is expensive, which restricts the size of the dataset and the dimensionality of the features.

2.3. Hash kernels

Ganchev and Dredze [7] provide empirical evidence that using hashing can eliminate alphabet storage and reduce the number of parameters without severely deteriorate the performance. In addition, Langford *et al.* [10] release the Vowpal Wabbit fast online learning software which uses a hash representation similar to the one discussed here. Shi *et al.* [13] propose a hash kernel to deal with the issue of computational efficiency by a very simple algorithm: high-dimensional vectors are compressed by adding up all coordinates which have the same hash value—one only needs to perform as many calculations as there are nonzero terms in the vector. The hash kernel can jointly hash both label and features, thus the memory footprint is essentially independent of the number of classes used. Shi *et al.* [14] further extend to structured data. Weinberger *et al.* [16] propose a unbiased hash kernel which is applied to a large scale application of mass personalized spam filtering.

2.4. Connection between hash kernels and compressed sensing

Previous works on hash kernels use hashing to perform feature reduction with a theoretical guarantee that learning in the reduced features space gains much computational power without any noticeable loss of accuracy. The deviation bound and Rademacher bound show that hash kernels have no information loss asymptotically due to the internal feature redundancy.

Alternatively, we can view hashing as a measurement matrix (see Section 4.2) in compressed sensing. We provide both theoretical guarantees in Section 4 and empirical results in Section 5 to show that recovering the original signal is possible. Thus hash kernels compress the original signal/feature in a recoverable way. This explains why it works well asymptotically in the context of [13, 14, 16].

3. Hashing for face recognition

We show in this section that hashing can be applied to face recognition.

3.1. Algorithms

Consider face recognition with n frontal training face images collected from $K \in \mathbb{N}$ subjects. Let n_k denote the number of training images $(\mathbf{x}_i, c_i)$ with $c_i = k$, thus $n = \sum_{k=1}^{K} n_k$. Without loss of generality, we assume that all the data have been sorted according to their labels and then we collect all the vectors in a single matrix $\mathbf{A}$ with m rows and n columns, given by

$$\mathbf{A} = [\mathbf{x}_1, ..., \mathbf{x}_{n_1}, ..., \mathbf{x}_n] \in \mathbb{R}^{m,n}. \tag{1}$$

As in [19, 18], we assume that any test image lies in the subspace spanned by the training images belonging to the same person. That is for any test image $\mathbf{x}$, without knowing its label information, we assume that there exists $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_n)$ such that

$$\mathbf{x} = \mathbf{A}\boldsymbol{\alpha}. \tag{2}$$

It is easy to see that if each subject has the number of images in the dataset, then the α for each subject has maximally $1/K$ portion of nonzero entries. In practice, α is

Algorithm 1 Hashing with ℓ_1

```
Input: a image matrix $\mathbf{A}$ for $K$ subjects, a test image $\mathbf{x} \in \mathbb{R}^m$ and an error tolerance $\epsilon$.
Compute $\tilde{\mathbf{x}}$ and $\mathbf{\Phi}$.
Solve the convex optimization problem
    $\min \|\boldsymbol{\alpha}\|_{\ell_1} \quad \text{subject to} \quad \|\tilde{\mathbf{x}} - \mathbf{\Phi}\boldsymbol{\alpha}\|_{\ell_2} \leq \epsilon.$    (6)
Compute the residuals $r_k(\mathbf{x}) = ||\tilde{\mathbf{x}} - \mathbf{\Phi}\alpha^k(\mathbf{x})||_{\ell_2}$ for $k = 1, \ldots, K$, where $\alpha^k$ is the subvector consisting of the components of $\alpha$ corresponding to the basis of class $k$.
Output: identity $c^* = \operatorname{argmin}_k r_k(\mathbf{x})$.
```

more sparse since often only a small subset of images from the same subjects have nonzero coefficients.

Yang *et al.* [19] and Wright *et al.* [18] use a random matrix $\mathbf{R} \in \mathbb{R}^{d,m}$ to map $\mathbf{A}\boldsymbol{\alpha}$, where $d \ll m$, and seek for $\boldsymbol{\alpha}$ by following ℓ_1 minimization:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \|\widetilde{\mathbf{x}} - \widetilde{\mathbf{A}}\boldsymbol{\alpha}\|_{\ell_2}^2 + \lambda\|\boldsymbol{\alpha}\|_{\ell_1}, \tag{3}$$

where $\widetilde{\mathbf{A}} := \mathbf{RA}$, $\widetilde{\mathbf{x}} := \mathbf{Rx}$ and λ is the regularizer controlling the sparsity of $\boldsymbol{\alpha}$. However, they did not provide a theoretical result on the reconstruction rate and the face recognition rate. We show both of our algorithm in Section 4.

3.2. Hashing with ℓ_1

Computing $\mathbf{R}$ directly can be inefficient, therefore we propose hashing to facilitate face recognition. Denote by $h_s(j, d)$ a hash function $h_s : \mathbb{N} \to \{1, \ldots, d\}$ uniformly, where $s \in \{1, \ldots, S\}$ is the seed. Different seeds give different hash functions.

Given $h_s(j, d)$, the hash matrix $\mathbf{H} = (H_{ij})$ is defined as

$$H_{ij} := \begin{cases} 2h_s(j,2) - 3, & h_s(j,d) = i, \forall s \in \{1, \ldots, S\} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

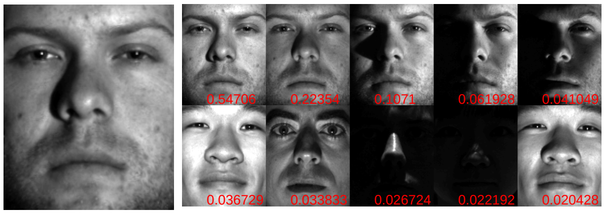
Apparently, $H_{ij} \in \{0, \pm 1\}$. Equally likely ± 1 result in an unbiased estimator (see [16]). Let $\mathbf{\Phi} := \mathbf{H}\,\mathbf{A} = (\Phi_{ij}) \in \mathbb{R}^{d,n}$. We look for $\boldsymbol{\alpha}$ by

$$\min \|\boldsymbol{\alpha}\|_{\ell_1} \quad \text{subject to} \quad \|\tilde{\mathbf{x}} - \mathbf{\Phi}\boldsymbol{\alpha}\|_{\ell_2} \leq \epsilon, \tag{5}$$

where $\tilde{\mathbf{x}} = \mathbf{H}\,\mathbf{x}$. The hashing with ℓ_1 is illustrated in Algorithm 1. It is known that the ℓ_1 has complexity $O(d^2 n^{3/2})$.

3.3. Hashing with orthogonal matching pursuit

Tropp and Gilbert [15] propose Orthogonal Matching Pursuit (OMP) which is faster than ℓ_1 minimization and but requires more measurements than ℓ_1 does for achieving the same precision. Equipped with hashing, the hashing OMP (see Algorithm 2) is much faster than random ℓ_1, random OMP, and hashing ℓ_1 without significantly loss of accuracy. It is known that OMP has complexity $O(dn)$. Hashing OMP is faster than random OMP due to sparsity of hash matrix $\mathbf{H}$ (see a sparse $\mathbf{H}$ in Figure 2).



(a) (b)

Figure 1. Demonstration of the recognition procedure of Hashface+ℓ_1. (a) is the test face; (b) is the training faces corresponding to the 10 largest weighted entries in α, the absolute value of their weights are shown on the images in red.

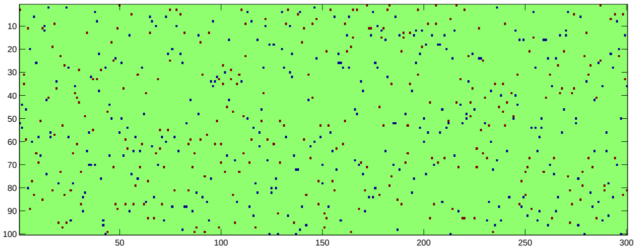


Figure 2. Demonstration of a hash matrix. The area with green color means the entry's value is 0, brown indicates value -1 while blue indicates 1.

Algorithm 2 Hashing with OMP

```
Input: a image matrix $\mathbf{A}$ for $K$ subjects, a test image $\mathbf{x} \in \mathbb{R}^m$.
Compute $\tilde{\mathbf{x}}$ and $\mathbf{\Phi}$.
Get $\boldsymbol{\alpha}$ via OMP procedure
    $\boldsymbol{\alpha} = \text{OMP}(\tilde{\mathbf{x}}, \mathbf{\Phi})$    (7)
Compute the residuals $r_k(\mathbf{x}) = ||\tilde{\mathbf{x}} - \mathbf{\Phi}\alpha^k(\mathbf{x})||_{\ell_2}$ for $k = 1, \ldots, K$, where $\alpha^k$ is the subvector consisting of the components of $\alpha$ corresponding to the basis of class $k$.
Output: identity $c^* = \operatorname{argmin}_k r_k(\mathbf{x})$.
```

3.4. Efficiency on Computation and Memory Usage

For random ℓ_1, the random matrix $\mathbf{R}$ needs to be computed beforehand and stored throughout the entire routine. When training set is large or the feature dimension is high, computing and storing $\mathbf{R}$ are expensive especially for a dense $\mathbf{R}$. We will show now with hashing, $\mathbf{H}$ is no longer needed to be computed beforehand explicitly. For example $\mathbf{\Phi}$ and $\tilde{\mathbf{x}}$ can be directly computed as follows without computing $\mathbf{H}$.

$$\begin{aligned} &\forall i = 1, \ldots, d, j = 1, \ldots, n \\ &\Phi_{ij} = \sum_{1 \le s \le S} \Big(\sum_{1 \le t \le m; h_s(t,d)=i} A_{jt} \xi_{st} \Big), \end{aligned} \tag{8}$$

where

$$\xi_{st} = \begin{cases} 1, & h_s(t,2) = 2 \\ -1, & \text{otherwise.} \end{cases}$$

$$\forall i = 1, \ldots, d \quad \tilde{x_i} = \sum_{1 \le s \le S} \Big(\sum_{1 \le j \le m; h_s(j,d)=i} y_j \xi_{sj} \Big). \tag{9}$$

It means for even very large image set, hashing with OMP can still be implemented on hardware with very limited memory.

4. Analysis

In this section, we show that hashing can be used for signal recovery, which is principle behind the application to face recognition. We further give a lower bound on its face recognition rate under some mild assumptions.

4.1. Restricted isometry property and signal recovery

A n-dimensional real valued signal is called η-sparse if it has at most η many nonzero components. The following Restricted Isometry Property (RIP) [5, 3] provides a guarantee for embedding high dimensional signal to a lower dimensional space without suffering a great distortion.

Definition 1 (Restricted Isometry Property) *Let Φ be an $m \times n$ matrix and let $\eta < n$ be an integer. Suppose that there exists a constant β such that, for every $m \times \eta$ submatrix Φ_η of Φ and for every vector x,*

$$(1 - \epsilon)\|x\|_{\ell_2}^2 \le \|\Phi_\eta x\|_{\ell_2}^2 \le (1 + \epsilon)\|x\|_{\ell_2}^2. \tag{10}$$

Then, the matrix Φ is said to satisfy the η-restricted isometry property with restricted isometry constant ϵ.

Baraniuk *et al.* [2] proves the RIP holds with high probability for some random matrices by the well-known Johnson-Lindenstrauss Lemma. With RIP, it is possible to reconstruct the original sparse signal by randomly combining the entries as the following theorem.

Theorem 4.1 (Recovery via Random Map [15, 5, 12]) *For any η-sparse signal $\boldsymbol{\alpha} \in \mathbb{R}^n$ and two constants $z_1, z_2 > 0$, let $m \ge z_1 \eta \log(n/\eta)$, and draw m row vectors $\mathbf{r}_1, \ldots, \mathbf{r}_m$ independently from the standard Gaussian distribution on $\mathbb{R}^n$. Denote the stacked vectors $\{\mathbf{r}_i\}_{i=1}^m$ as the matrix $\mathbf{R} \in \mathbb{R}^{m,n}$ and take m measurements $x_i = \langle \mathbf{r}_i, \boldsymbol{\alpha} \rangle, i = 1, \ldots, m$,* i.e.*, $\mathbf{x} = \mathbf{R}\boldsymbol{\alpha}$. Then with probability at least $1 - e^{-z_2 m}$, the signal α can be recovered via*

$$\alpha^* = \underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{R}\boldsymbol{\alpha}\|_{\ell_2}^2 + \lambda \|\boldsymbol{\alpha}\|_{\ell_1}. \tag{11}$$

The condition on m in the theorem above comes from RIP condition. This immediately leads to following corollary when recovery is on a specific basis $\mathbf{A}$.

Corollary 4.2 (Recovery on a Specific Basis) *For any η-sparse signal $\boldsymbol{\alpha} \in \mathbb{R}^n$ and two constants $z_1, z_2 > 0$, let $d \ge z_1 \eta \log(n/\eta)$, and draw d row vectors $\mathbf{r}_1, \ldots, \mathbf{r}_d$ independently from the standard Gaussian distribution on $\mathbb{R}^m$. Denote the stacked vectors $\{\mathbf{r}_i\}_{i=1}^d$ as the matrix $\mathbf{R} \in \mathbb{R}^{d,m}$. For any matrix $\mathbf{A} \in \mathbb{R}^{m,n}$ with unit length columns, with probability at least $1 - e^{-z_2 d}$, the signal α can be recovered via*

$$\alpha^* = \underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\operatorname{argmin}} \|\mathbf{R}\,\mathbf{x} - (\mathbf{R}\,\mathbf{A})\boldsymbol{\alpha}\|_{\ell_2}^2 + \lambda \|\boldsymbol{\alpha}\|_{\ell_1}. \tag{12}$$

Proof Let $\mathbf{A}_j, j = 1, \ldots, n$ denote the j-th column vector of the matrix $\mathbf{A}$ and let $\widetilde{\mathbf{A}} := \mathbf{R}\,\mathbf{A}$, *i.e.*, the row vectors $\widetilde{\mathbf{A}}_i = \big(\langle \mathbf{r}_i, \mathbf{A}_1 \rangle, \ldots, \langle \mathbf{r}_i, \mathbf{A}_n \rangle \big)$ for $(i = 1, \ldots, d)$. Note that the inner product $\langle \mathbf{r}_i, \mathbf{A}_j \rangle = \sum_{k=1}^m \mathbf{r}_{i,k}\, \mathbf{A}_{k,j}$ is still a random variable drawn from Gaussian distribution $N(0, \sum_{k=1}^m \mathbf{A}_{k,j}^2)$. Hence $\{\widetilde{\mathbf{A}}_i\}_{i=1}^d$ are random vectors independently drawn from the Gaussian distribution in $\mathbb{R}^m$. Corollary 4.2 follows Theorem 4.1.

4.2. Recovery with hashing

Can one reconstruct the signal via hashing rather than Gaussian random mapping? The answer is affirmative. Achlioptas [1] constructs an embedding with the property that all elements of the projection matrix $\mathbf{U}$ belong in $\{\pm 1, 0\}$ and shows such embedding has a Johnson-Lindenstrauss Lemma type of distance preservation property. Due to uniformity, a hashing matrix $\mathbf{H}$ with $S = d$ is such a projection matrix $\mathbf{U}$ ignoring the scaling. Since distance preservation property implies RIP [2], signal recovery still holds by replacing gaussian matrix with $\mathbf{U}$, and it leads to the corollary below.

Corollary 4.3 (Hashing ℓ_1 Recovery) *For any η-sparse signal $\boldsymbol{\alpha} \in \mathbb{R}^n$ and two constants $z_1, z_2 > 0$ depending on ϵ, given hash matrix $\mathbf{H}$, let $d \ge z_1 \eta \log(n/\eta)$, for any matrix $\mathbf{A} \in \mathbb{R}^{m,n}$, with probability at least $1 - e^{O(-z_2 d)}$, the signal α can be recovered via*

$$\alpha^* = \underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\operatorname{argmin}} \|\mathbf{H}\,\mathbf{x} - (\mathbf{H}\,\mathbf{A})\boldsymbol{\alpha}\|_{\ell_2}^2 + \lambda \|\boldsymbol{\alpha}\|_{\ell_1}. \tag{13}$$

Here the big O notation is to take into account of the scaling.

Tropp and Gilbert [15] show that the OMP recovery theorem holds for all admissible measurement matrices such as Gaussian random matrix and Bernoulli random matrix. Applying OMP to hashing matrix $\mathbf{H}$, we get following theorem:

Theorem 4.4 (Hashing OMP Recovery) *For any η-sparse signal $\boldsymbol{\alpha} \in \mathbb{R}^n$ and confidence $\delta > 0$, given hash matrix $\mathbf{H}$, let $d \geq 16\eta^2 \log(n/\delta)$, for any matrix $\mathbf{A} \in \mathbb{R}^{m,n}$, take the measurements such that $\mathbf{H}\,\mathbf{x} = (\mathbf{H}\,\mathbf{A})\boldsymbol{\alpha}$. Then with probability at least $1 - \delta$, the signal α can be recovered via Algorithm 2.*

Proof Admissibility mainly relies on the coherence statistic $\mu := \max_{j<k} |\langle \mathbf{R}_j, \mathbf{R}_k\rangle|$. In a hash matrix $\mathbf{H}$, $\{-1,+1\}$ are equally likely to appear so $\mathbb{E}[\langle \mathbf{H}_j, \mathbf{H}_k\rangle] = 0$. By hoeffding inequality, $P(|\langle \mathbf{H}_j, \mathbf{H}_k\rangle > \epsilon|) \leq 2e^{-\epsilon N/2}$. Union bound argument further gives bound on $P(\mu) < d$ of $\mathbf{H}$ as of Bernoulli random matrix. This then leads to the same bound on the smallest singular value. Also we know the columns of $\mathbf{H}$ with multiple hash functions are independent and normalization only changes the scale, hence $\mathbf{H}$ is admissible. Admissibility implies reconstruction, so the theorem holds.

4.3. Recognition rates

A commonly used assumption is that any test face image can be represented as a weighted sum of face images belonging to the same person, which has been used in [18, 19]. Ideally, once we achieve the exact weights, the classification should be perfect. However, because the similarity of human face appearance and noise, it is no longer true. So we propose a weakened assumption below.

Assumption 2 *There exists a high dimension representation in the training face images indices space, in which the classification can be conducted with recognition rate at least q.*

The following theorem provides bounds on recognition rate for any test image via hashing.

Theorem 4.5 (Recognition Rate via Hashing) *The recognition rates via Algorithm 1 and 2 are, at least $(1 - e^{O(-z_2 d)})q$, and $(1-\delta)q$, respectively, under Assumption 2.*

Proof We know that with probability at least $1 - e^{O(-z_2 d)}$, the signal can be recovered via Corollary 4.3. With Assumption 2, we know that even the $e^{O(-z_2 d)}$ portion of not-perfectly-recovered signals are all misclassified, the classification accuracy is still greater or equal to $(1 - e^{O(-z_2 d)})q$. Similarly for Algorithm 2.

Note that the bound in above theorem is possible be further tighten by salvaging the portion of not-perfectly-recovered signals for classification. Indeed, predictions on those signals are usually not complete wrong.

5. Experiments

To compare the proposed hashing approaches with random ℓ_1 [19, 18], we use the same databases, namely, the Extended YaleB and AR as Wright *et al.* used in [18]. The Extended YaleB database [8] contains $2,414$ frontal-face images from 38 individuals. The cropped and normalized 192×168 face images were captured under various laboratory-controlled lighting conditions. Each subject has 62 to 64 images. Thus we randomly select 32, 15, 15 of them (no repetition) as the training, validation and testing sets. The AR database consists of over $4,000$ front images for 126 individuals. Each individual has 26 images. The pictures of each individual were taken in two different days [11]. Unlike Extended YaleB, the faces in AR contain more variations such as illumination change, expressions and facial disguises. 100 subjects (50 male and 50 female) are selected randomly. And for each individual, 13, 7 and 6 images (since 26 images in total for each individual) are chosen as training, validation and testing set respectively.

5.1. Comparisons on accuracy and efficiency

We run the experiment 10 times on each method and report the average accuracy with the standard deviations as well as the running time. In each round we run the experiment, the databases are split according to above scheme and different algorithms are performed on the same training, validation and test data set. The number of hash function L is tuned via model selection assessed on the validation set. Given a feature dimension dim in the reduced feature space, L is the rounded up integer of $u\times$ Dim. For hashing ℓ_1 $u \in \{0.02, 0.04, 0.06, ...0.38, 0.40\}$ and for hashing OMP ℓ_1 $u \in \{0.05, 0.10, 0.15, ...0.95, 1.00\}$. The error tolerance ε for random ℓ_1 is fixed to 0.05 which is identical to the value adopted in [19].

We evaluate the our methods and state-of-arts on Yale B and AR database shown in Tab 1. The best accuracies are highlighted in bold. As we can see, when $\mathrm{Dim} = 300$, hashing ℓ_1 gets the best accuracies on both datasets. An example is given in Fig 3. Fig 3 (d) (e) show that the hashing ℓ_1 weight vector is more sparse than random ℓ_1. We conjecture that the sparsity is a distinct pattern for classification, which may help to improve the performance as observed in [14]. Overall, hashing has competitive accuracy with random ℓ_1.

Hashing OMP is significantly faster than random ℓ_1 (from 30 to 150 times shown in Tab 2). This is further verified in Fig 4, which shows that as the feature dimensionality increases, running time of the hashing OMP is almost con-

		Dim-25	Dim-50	Dim-100	Dim-200	Dim-300
AR	Hash+OMP	0.572 ± 0.074	0.658 ± 0.063	0.778 ± 0.066	0.937 ± 0.032	0.969 ± 0.019
	Random+OMP	0.563 ± 0.070	0.689 ± 0.077	0.784 ± 0.060	0.835 ± 0.036	0.908 ± 0.034
	Eigen+OMP	0.435 ± 0.132	0.449 ± 0.131	0.449 ± 0.112	0.606 ± 0.068	0.671 ± 0.040
	Hash+ℓ_1	0.660 ± 0.051	0.727 ± 0.064	0.915 ± 0.037	0.961 ± 0.029	$\mathbf{0.985 \pm 0.013}$
	Random+ℓ_1	0.653 ± 0.068	0.855 ± 0.047	0.915 ± 0.042	0.929 ± 0.028	0.958 ± 0.016
	Eigen+ℓ_1	0.627 ± 0.137	0.705 ± 0.094	0.751 ± 0.061	0.758 ± 0.035	0.806 ± 0.050
	Eigen+KNN	0.452 ± 0.102	0.500 ± 0.102	0.537 ± 0.101	0.555 ± 0.097	0.558 ± 0.096
	Fisher+KNN	0.575 ± 0.060	0.740 ± 0.045	0.920 ± 0.026	0.977 ± 0.011	0.981 ± 0.011
	Eigen+SVM	0.758 ± 0.063	$\mathbf{0.903 \pm 0.048}$	$\mathbf{0.959 \pm 0.021}$	0.976 ± 0.017	0.979 ± 0.011
	Fisher+SVM	$\mathbf{0.760 \pm 0.054}$	0.896 ± 0.043	0.953 ± 0.020	$\mathbf{0.979 \pm 0.013}$	0.980 ± 0.012
YaleB	Hash+OMP	0.722 ± 0.056	0.806 ± 0.057	0.856 ± 0.050	0.939 ± 0.022	0.964 ± 0.016
	Random+OMP	0.704 ± 0.065	0.821 ± 0.059	0.908 ± 0.039	0.945 ± 0.033	0.944 ± 0.029
	Eigen+OMP	0.094 ± 0.033	0.289 ± 0.075	0.669 ± 0.078	0.882 ± 0.053	0.911 ± 0.048
	Hash+ℓ_1	$\mathbf{0.853 \pm 0.053}$	0.899 ± 0.030	0.951 ± 0.021	0.977 ± 0.017	$\mathbf{0.982 \pm 0.013}$
	Random+ℓ_1	0.844 ± 0.058	$\mathbf{0.928 \pm 0.036}$	$\mathbf{0.966 \pm 0.018}$	$\mathbf{0.980 \pm 0.017}$	0.979 ± 0.016
	Eigen+ℓ_1	0.648 ± 0.102	0.822 ± 0.072	0.911 ± 0.049	0.936 ± 0.037	0.945 ± 0.036
	Eigen+KNN	0.459 ± 0.080	0.589 ± 0.101	0.662 ± 0.109	0.702 ± 0.100	0.714 ± 0.096
	Fisher+KNN	0.759 ± 0.079	0.891 ± 0.050	0.920 ± 0.038	0.948 ± 0.029	0.954 ± 0.030
	Eigen+SVM	0.793 ± 0.081	0.890 ± 0.063	0.919 ± 0.041	0.940 ± 0.036	0.953 ± 0.029
	Fisher+SVM	0.790 ± 0.064	0.880 ± 0.068	0.913 ± 0.040	0.939 ± 0.035	0.948 ± 0.031

Table 1. Comparison on accuracy for Hashface+OMP, Randomface+ℓ_1 and Eigenface+ℓ_1. On both datasets, Hash+ℓ_1 achieves the best classification accuracy. When the dimensionality is low, sparse representation based algorithms do not perform as well as SVM.

		Dim-25	Dim-50	Dim-100	Dim-200	Dim-300
AR	Hash+OMP	4.45 ± 0.08	$\mathbf{11.55 \pm 0.22}$	$\mathbf{24.8 \pm 0.17}$	$\mathbf{78.25 \pm 0.41}$	$\mathbf{101.15 \pm 1.34}$
	Random+OMP	$\mathbf{3.3 \pm 0.07}$	12.05 ± 0.23	80.25 ± 0.93	812.55 ± 0.74	1323.45 ± 2.00
	Eigen+OMP	3.55 ± 0.09	12.45 ± 0.24	77.25 ± 0.32	299.55 ± 1.54	422.1 ± 2.03
	Hash+ℓ_1	359.1 ± 1.39	714.55 ± 2.96	1740.5 ± 12.69	6125.85 ± 99.22	15718.9 ± 290.25
	Random+ℓ_1	367.25 ± 0.96	814.35 ± 5.44	2276.95 ± 10.28	11266 ± 73.18	31731 ± 292.63
	Eigen+ℓ_1	334.85 ± 1.78	751.95 ± 7.10	2637.9 ± 37.68	8758.3 ± 132.26	19632.9 ± 477.55
YaleB	Hash+OMP	3.65 ± 0.05	$\mathbf{10.05 \pm 0.02}$	$\mathbf{67.4 \pm 0.80}$	$\mathbf{61.45 \pm 0.34}$	$\mathbf{138.05 \pm 0.24}$
	Random+OMP	$\mathbf{3.4 \pm 0.05}$	10.75 ± 0.18	74.3 ± 0.11	944.25 ± 0.53	2944.45 ± 2.90
	Eigen+OMP	3.65 ± 0.05	10.8 ± 0.19	75 ± 0.30	190.65 ± 0.49	291.35 ± 0.78
	Hash+ℓ_1	335.5 ± 1.83	724.45 ± 2.53	1713.3 ± 14.69	5191.9 ± 120.27	9536.8 ± 311.48
	Random+ℓ_1	329.4 ± 2.48	823.25 ± 5.63	2401 ± 19.56	8655.6 ± 71.23	21887.8 ± 164.97
	Eigen+ℓ_1	330.8 ± 2.36	742.55 ± 5.42	2006.6 ± 38.53	4621.65 ± 143.30	8444.65 ± 273.76

Table 2. Comparison on running time(ms) for Hashface+OMP, Randomface+ℓ_1 and Eigenface+ℓ_1. Hash+OMP is much faster than other methods.

stant whereas that of random ℓ_1 increases dramatically. In real world application, the speed of algorithms is a big issue. Hence we further compare hashing OMP with random ℓ_1 by restricting their running time to the same level. This way, hashing OMP gets much better accuracies than random ℓ_1 shown in Tab 3. In fact, one may further improve the hashing OMP accuracy by increasing the feature dimensionality, for Fig 4 suggests that the running time curse for hashing OMP is almost flat.

5.2. Predicting via residuals or α directly?

Algorithm 1 uses the residuals to predict the label. Alternatively we can learn a classifier on the sparse α directly. To investigate it, we estimated α via Algorithm 1 (*i.e.*, ℓ_1 minimization) on the test set and the validation set of AR dataset. Then we split the union of the two sets into 10 folds. We ran 10 folds cross-validation (8 for training, 1 for testing, and 1 for validation) with SVM. We used both the original α and the normalized one denoted as $\alpha_{[0,1]}$, which is normalized to $[0, 1]$. Because α has both positive and negative entries, the normalization step introduces many nonzero entries to

Running time(ms)	Hash+OMP	10.05 ± 0.020	46.65 ± 2.394	85.4 ± 3.891	340.95 ± 4.080
	Random+ℓ_1	**NA**	58.35 ± 1.152	97.15 ± 7.926	329.4 ± 2.480
Accuracy	Hash+OMP	$\mathbf{0.658 \pm 0.063}$	$\mathbf{0.687 \pm 0.060}$	$\mathbf{0.835 \pm 0.037}$	$\mathbf{0.998 \pm 0.034}$
	Random+ℓ_1	**NA**	0.0571 ± 0.010	0.2 ± 0.047	0.653 ± 0.068
Dimension	Hash+OMP	50	85	180	1000
	Random+ℓ_1	**NA**	5	10	25

Table 3. Comparison on accuracies given running time constraint for Hashface+OMP and Randomface+ℓ_1 on AR. "Dimension" shows the dimensions under which the two manners could achieve similar running speed. "Running time" shows the real running time that should be similar to each other for a certain running speed. NA means impossible to achieve that speed.

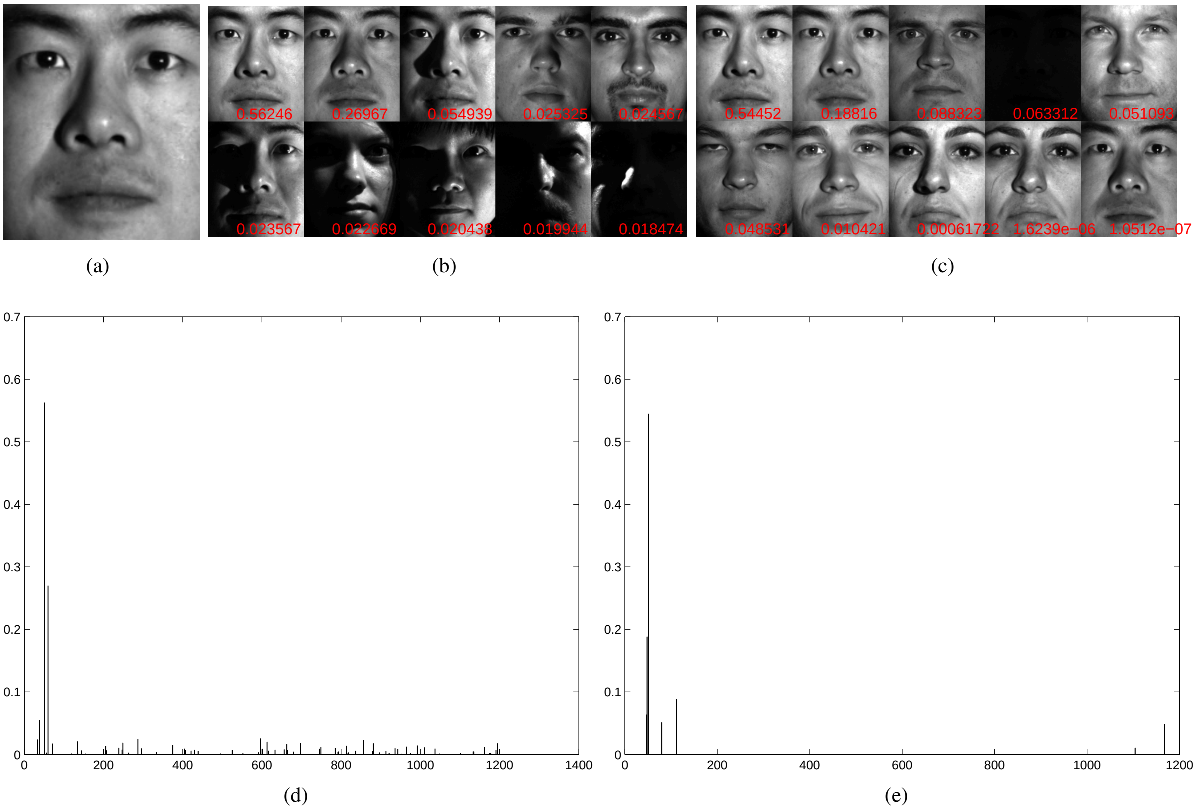


Figure 3. The comparison of the recognition procedure of Hashing + ℓ_1 and Random +ℓ_1 on YaleB. (a) is the test face; (b)&(c) are the top 10 weighted training faces for random ℓ_1 and hashing ℓ_1 respectively. The absolute value of the weights are shown in red (view in color); (d)&(e) are the bar charts corresponding to the absolute value of top 100 largest weighted entries in the weight α for random ℓ_1 and hashing ℓ_1 respectively.

$\alpha_{[0,1]}$. As we can see in Table 4, when Dim $= 50$, SVM gets better result than hashface OMP and ℓ_1. When Dim ≥ 100 hashface OMP and ℓ_1 beat SVM. The experiment suggests that, when the feature dimensionality is low (e.g. ≤ 50), predicting via α is a good idea; when the feature dimensionality is high, predicting via residuals is better.

6. Conclusion

We have proposed a new face recognition methodology with hashing, which speeds up the state-of-the-art in [18] by up to 150 times, with comparable recognition rates. Both theoretical analysis and experiments justify the excellence of the proposed method.

Acknowledgments NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Center of Excellence program.

This work is partially supported by the IST Program of the European Community, under the Pascal 2 Network of Excellence.

	Dim 50	Dim 100	Dim 200	Dim 300
Accuracy on α	0.865 ± 0.006	0.876 ±0.010	0.875 ± 0.007	0.835 ± 0.009
Accuracy on $\alpha_{[0,1]}$	0.853 ± 0.006	0.877 ± 0,011	0.878± 0.007	0.849 ± 0.010

Table 4. Test accuracy via predicting on α on AR dataset with 10 fold cross-validation.

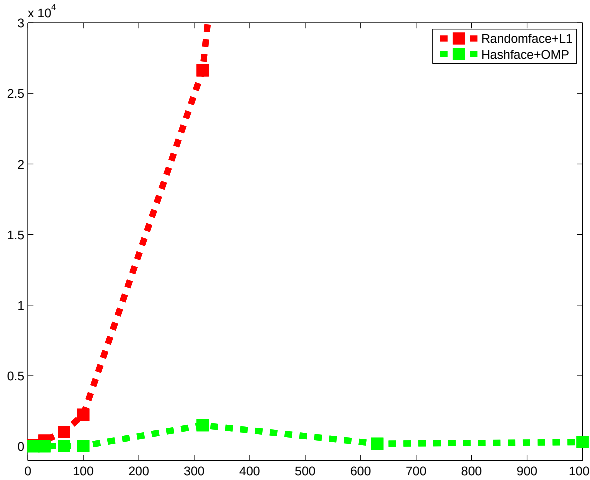


Figure 4. The running time curves of Hashing+OMP and Random ℓ_1 on AR. The horizontal axis represents the dimensionality and the vertical axis is the running time in ms.

We thank Junbin Gao, Tiberio Caetano, Mark Reid and Oliver Nagy for discussions about compressed sensing. We also would like to thank Anton van den Hengel and David Suter whose useful comments improve the presentation of this work.

References

[1] D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci*, 66(4):671–687, 2003.

[2] R.G. Baraniuk, M. Davenport, R. DeVore, and M.B. Wakin. A simple proof of the restricted isometry principle for random matrices. *Constructive Approximation*, 2007.

[3] E. Candés. The restricted isometry property and its implications for compressed sensing. *C. R. Acad. Sci. Paris, Ser. I*, 346:589–592, 2008.

[4] E. Candés, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Information Theory*, 52(2):489–509, 2006.

[5] E. Candés and T. Tao. Decoding by linear programming. *IEEE Trans. Information Theory*, 51(12):4203–4215, 2005.

[6] D. L. Donoho. Compressed sensing. *IEEE Trans. Information Theory*, 52(4):1289–1306, 2006.

[7] K. Ganchev and M. Dredze. Small statistical models by random feature mixing. In *Proc. 9th SIGdial Workshop Discourse & Dialogue*, 2008.

[8] A. Georghiades, P. Belhumeur, , and D. Kriegman. From few to many: Illumniation cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelli.*, 23(6):643–660, 2001.

[9] X. He, S. Yan, Y. Hu, and P. Niyogi. Face recognition using Laplacianfaces. *IEEE Trans. Pattern Anal. Mach. Intelli.*, 27(3):328–340, 2005.

[10] J. Langford, L. Li, and A. Strehl. Vowpal wabbit online learning project, 2007. `http://hunch.net/?p=309`.

[11] A. Martinez and R. Benavente. The ar face database. Technical Report 24, CVC Tech. Report, 1998.

[12] M. Rudelson and R. Veshynin. Geometric approach to error correcting codes and reconstruction of signals. *Int. Math. Res. Notices*, 64:4019–4041, 2005.

[13] Q. Shi, J. Petterson, G. Dror, J. Langford, A. Smola, A. Strehl, and S. V. N. Vishwanathan. Hash kernels. In *Proc. Int. Workshop Artificial Intell. & Statistics*, 2009.

[14] Q. Shi, J. Petterson, G. Dror, J. Langford, A. J. Smola, and S.V.N. Vishwanathan. Hash kernels for structured data. *J. Mach. Learn. Res.*, 10:2615–2637, 2009.

[15] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Information Theory*, 53(12):4655–4666, 2007.

[16] K. Weinberger, A. Dasgupta, J. Attenberg, J. Langford, and A.J. Smola. Feature hashing for large scale multitask learning. In L. Bottou and M. Littman, editors, *Proc. Int. Conf. Mach. Learn.*, 2009.

[17] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, 2009.

[18] J. Wright, A. Y. Yang, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intelli.*, 2008.

[19] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry. Feature selection in face recognition: A sparse representation perspective. *Tech. Report*, 2007.