Partial Least Squares for Face Hashing

Cassio E. dos Santos, Ewa Kijak, William Robson Schwartz, Guillaume Gravier

Partial Least Squares for Face Hashing

Cassio E. dos Santos Jr.[a,*], Ewa Kijak[b], Guillaume Gravier[b], William Robson Schwartz[a]

[a] *Department of Computer Science, Univ. Federal de Minas Gerais, Belo Horizonte, Brazil*
[b] *IRISA & Inria Rennes (CNRS, Univ. Rennes 1), Campus de Beaulieu, Rennes, France*

Abstract

Face identification is an important research topic due to areas such as its application to surveillance, forensics and human-computer interaction. In the past few years, a myriad of methods for face identification has been proposed in the literature, with just a few among them focusing on scalability. In this work, we propose a simple but efficient approach for scalable face identification based on partial least squares (PLS) and random independent hash functions inspired by locality-sensitive hashing (LSH), resulting in the *PLS for hashing* (PLSH) approach. The original PLSH approach is further extended using feature selection to reduce the computational cost to evaluate the PLS-based hash functions, resulting in the state-of-the-art *extended PLSH* approach (ePLSH). The proposed approach is evaluated in the dataset FERET and in the dataset FRGCv1. The results show significant reduction in the number of subjects evaluated in the face identification (reduced to 0.3% of the gallery), providing averaged speedups up to 233 times compared to evaluating all subjects in the face gallery and 58 times compared to previous works in the literature.

Keywords: computer vision, face recognition, image indexing, partial least squares

*Corresponding author.
Email addresses: cass@dcc.ufmg.br (Cassio E. dos Santos Jr.), ewa.kijak@irisa.fr (Ewa Kijak), guillaume.gravier@irisa.fr (Guillaume Gravier), william@dcc.ufmg.br (William Robson Schwartz)

1. Introduction

According to [1], there are three tasks in face recognition depending on which scenario it will be applied: *verification*, *identification* and *watch-list*. In the verification task ($1:1$ matching problem), two face images are provided and the goal is to determine whether these images belong to the same subject. In the identification task ($1:N$ matching problem), the goal is to determine the identity of a face image considering identities of subjects enrolled in a face gallery. The watch-list task ($1:N$ matching problem), which may also be considered as an open-set recognition task [2], consists in determining the identity of a face image, similar to the identification task, but the subject may not be enrolled in the face gallery. In this case, the face recognition method may return an identity in the face gallery or a not-enrolled response for any given test sample.

In this work, we focus on the face identification task. Specifically, the main goal is to provide a face identification approach scalable to galleries consisting of numerous subjects and on which common face identification approaches would probably fail on responding in low computational time. There are several applications for a scalable face identification method: surveillance scenarios, human-computer interaction and social media. The few aforementioned applications show the importance of performing face identification fastly and, in fact, several works in the literature have been developed in the past years motivated by these same types of applications (surveillance, forensics, human-computer interaction, and social media). However, most of the works focus on developing fast methods to evaluate one test face and a single subject enrolled in the gallery. These methods usually develop low computational cost feature descriptors for face images that are discriminative and with low memory footprint enough to process several images per second. Note that these methods still depend on evaluating all subjects in the face gallery. Therefore, if the number of subjects in the gallery increases significantly, these methods will not be able to respond fastly and new methods shall be developed to scale the face identification to this larger gallery.

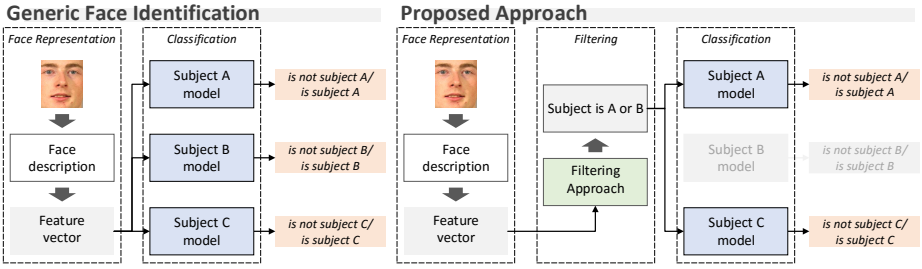


Figure 1: Common face identification pipeline and the proposed pipeline with the filtering approach which is used to reduce the number of evaluations in the classification step with low computational cost. The filtering approach is the main contribution in this work and it is tailored considering recent advances in large-scale image retrieval and face identification based on PLS.

Face identification methods usually consists of a face representation or description in the feature vector where mathematical models can be applied to determine the face identity. In this case, it is used one model to determine each identity in the face gallery, therefore, being necessary a number of models equal to the gallery size. Note that the parameters in each model are learned using samples for each subject in the face gallery and every model must be evaluated to correctly identify a test sample. In this work, we propose a method to reduce the number of models evaluated in the face identification by eliminating identities that are somewhat clearly not the identity in the test sample. Figure 1 illustrates the common face identification pipeline employed in practice and the main component tackled in this work.

There is an extensive literature of works regarding large-scale image retrieval that could be employed in face identification. However, most of these works focus on returning a list containing images from the dataset that are similar to the test image. Although reasonable to recover images in large datasets, such approaches are not suitable to apply directly to the face identification task. The models from subjects in the face gallery should optimally be described regarding the discriminative features related to each subject identity, which might consume

less memory, specially if several samples per subject are available, and less computational time since only discriminative features are evaluated to determine the face identity.

The proposed approach is inspired by the family of methods regarded as *locality-sensitive hashing* (LSH), which are the most popular large-scale image retrieval method in the literature, and the *partial least squares* (PLS), which has been explored intensively in numerous past works regarding face recognition [3, 4, 5, 6]. We call the proposed approach PLS *for hashing*, abbreviated to PLSH and ePLSH in its extension.

The main goal in LSH is to approximate the representation of samples in the high dimensional space using a small binary representation where the search can be implemented efficiently employing a hash structure to approximate near-identical binary representations. The idea in LSH is to generate random hash functions to map the feature descriptor in the high dimensional representation to bits in the binary representation.

In the PLSH approach, the random projection in the aforementioned example is replaced by a PLS regression, which provides discriminability among subjects in the face gallery and allow us to employ a combination of different feature descriptors to generate a robust description of the face image. PLSH is able to provide significant improvement over the brute-force approach (evaluating all subjects in the gallery) and compared to other approaches in the literature. Furthermore, since the evaluation of hash functions in PLSH requires a dot product between the feature and regression vectors, additional speedup can be achieved by employing feature selection methods, resulting on the extended version of PLSH (ePLSH).

The following contributions are presented in this work. (i) A fast approach for face identification that support a combination of several feature descriptors and high dimensional feature vectors. (ii) The proposed approach presents at least comparable performance with other methods in the literature and up to 58 times faster when enough samples per subject are available for train. (iii) Extensive discussion and experimentation regarding alternative implementations

that may guide future development in scalable face identification methods. (iv) The proposed approach is easy to implement and to deploy in practice since only two trade-off parameters need to be estimated. This work is closely related to [7], where we proposed the PLSH approach. The main difference of this work compared to [7] is the additional discussions about the PLSH consistency, relation to Hamming embedding, computational cost, alternative implementations, better feature set and the proposal of the ePLSH approach.

The remaining of this work is organized as follows. In Section 2, we review works related to face identification, fast face identification and large-scale image retrieval. In Section 3, we describe PLS for regression, face identification and face hashing, which are the main components for the proposed face identification pipeline. Experiments and discussions regarding the proposed approach are discussed in Section 4. Finally, we conclude this work with final remarks and author suggestions for future directions in Section 5.

2. Related work

This section reviews works related to face identification (Section 2.1) and large-scale image retrieval (Section 2.2). The reader may find more information regarding face identification in the book titled *Handbook of face recognition* [8]. For large-scale image retrieval, we refer the reader to the work [9] regarding locality-sensitive hashing.

2.1. Face identification

Face identification methods consist generally of two components: classifier and face representation. The classifier is responsible for receiving the face representation and returning an identity in the gallery, more specifically, it evaluates whether a face representation from a test image refers to a subject in the face gallery.

Feature descriptors provide a robust manner to represent face images invariant to misalignment, illumination and pose of the face. Regarding feature

descriptors considered in face identification, the most commons are local binary patterns (LBP) [10, 11], Gabor filters [12, 13] and descriptors based on gradient images [14, 15]. These feature descriptors capture mainly texture and shape of the face image, which are relevant for face identification [4]. There are two manners to represent the face image [16]: *appearance-based* (holistic), where the whole face image is represented in a feature descriptor vector; and *feature-based* (local), where fiducial points of the face image, such as nose tip or corners, eyes and mouth, are represented instead of the whole image.

The advantage of the holistic representation is the rich and easy encoding of the overall appearance of the face image. Since every pixel value contributes somehow to the final feature descriptor, more information is available to distinguish between samples from different subjects. However, preprocessing is usually necessary to correct misalignment, illumination and pose. Feature descriptors commonly employed in holistic methods are the local binary patterns (LBP) [10], Gabor filters [12], combination of both [17], and large feature sets coupled with dimension reduction techniques [4].

The advantage of the local representation is its robustness to differences in pose, partial occlusion and shadowing. If some fiducial points are shadowed or occluded due to pose, for instance, other points may still be used to recognize the face image. However, the resulting feature vector is often ambiguous and imposes difficulties to identify the face image due to the reduced amount of data present in the small patch around the fiducial point. Common feature descriptors employed in local methods include LBP [11] and Gabor filter [13].

Fiducial points can be detected considering salient regions in the face image, which include corners and textured regions in the face. These salient regions, opposed to homogeneous regions such as cheek and forehead, tend to be stable among face images in different poses and lightning conditions. However, a method to match the detected salient regions among face images is necessary to compare feature descriptors. Liu et al. [15] employ the popular SIFT [18] to detect and match salient regions among face images. Another option is to learn common detectors for fiducial points (eye corner, nose tip, among others)

such that the match of fiducial points among face images is no longer necessary since feature descriptors from a common type of fiducial point can be directly compared [19].

In the past few years, a different approach based on sparse representation-based classification (SRC) has been providing high accuracy in face identification datasets [20]. SRC consists in representing a test face image as a linear combination of a dictionary of images, which is learned using samples in the face gallery. Although the original proposal of SRC requires a fair number of controlled samples per subject for training, Deng et al. [21] extended SRC to cope with few uncontrolled samples in the face gallery.

2.1.1. Fast face identification

Fast face identification is not a largely explored research topic and there are few works in the literature about it [22, 23, 24, 25, 4]. In [22], compact descriptors based on local binary patterns are used to compare quickly the candidates in the face gallery. In [23] and [24], a fast optimization algorithm is considered for SRC to reduce the computational cost when calculating the linear combination between the test and the samples in the dictionary. Similar to [26], where least trimmed squares (LTS) is considered to cope with noise in SRC-based face identification, Shen et. al. [27] propose an approximation of the least median squares (LMS), which provides speedup of some order of magnitude in the SRC approach while still dealing with noise in the gallery samples. Although the aforementioned methods provide significant improvement in the test-subject comparison, poor performance is observed when there are numerous subjects in the face gallery since these approaches still present linear asymptotic complexity with the gallery size.

To approach face identification in large galleries, a cascade of classifiers to discard a considerable number of candidates in early initial stages with low computation cost classifiers was proposed by Yuan et al. [25]. To keep high accuracy, the final stages of the cascade consists in more accurate and time-consuming classifiers. In [4], a binary tree structure was used to reduce the

number of subjects tested in the face gallery, resulting in a reduced computational complexity considering the number of subjects in the face gallery when compared to the brute-force approach.

The approach proposed in this work is an extension of [4] and the main difference is the employment of hashing instead of search trees. PLS is also considered with a combination of feature descriptors as in [4], which improves the face identification recognition rate compared to single feature descriptors. In this case, the contribution of the proposed approach lies in the distinct manner in which PLS is employed for hashing and the considerable improvement in speedup compared to the aforementioned scalable face identification approaches.

2.2. Large-scale image retrieval

The goal in the image retrieval task is to return a sorted list of "relevant" images enrolled in the gallery considering their similarity to a test sample. For reference of a few distinguished works, Jegou et. al. [28] employ quantization of feature descriptors considering a random rotation of the PCA transformation to ensure equal variance among projected features. Gong et. al. [29] employ a similar approach but considering the minimal quantization error of zero-mean samples in a zero-centered binary hypercube. In this case, an efficient optimization algorithm can be formulated, referred to as *iterative quantization* (ITQ), which provides better results than the random rotation employed in [28]. Shen et. al. [30] propose a method for embedding the gallery samples on non-parametric manifolds in an iterative manner from an initial subset of the samples, such that the embedding can be applied to large datasets. Shen et. al. [31] employ maximum margin linear classifiers to learn optimal binary codes by relaxing the bit discretization.

In this section, we focus only on locality-sensitive hashing which is the basis of our work. For a complete review of image hashing and large-scale image retrieval methods in the literature, we refer the reader to the work [9].

2.2.1. Locality-sensitive hashing

Locality-sensitive hashing (LSH) refers to a family of embedding approaches that aims at mapping similar feature descriptors to the same hash table bucket with high probability while keeping dissimilar features in different buckets. There are two types of hash functions in LSH [9]: *data independent*, where hash functions are defined regardless of the data; and *data dependent*, where the parameters of the hash functions are selected according to the training data. These two types are different from supervised and unsupervised learning of hash functions, in which the difference lies on whether data label is considered. For instance, data dependent hash functions may not consider the label of the data when learning hash functions. However, all supervised hash functions are intrinsically data dependent, since the family of hash functions $\mathcal{H}$ will be selected to discriminate labels.

Data independent hash functions are employed in the works of Data et al. [32], based on p-stable distributions; Chum et al. [33], based on min-hash; Joly et al. [34] and Poullot et al. [35], both works based on space filling curves. Data independent hash functions are usually employed in heterogeneous data like in the object recognition task. In this case, the overall distribution of the data is not modeled easily using data dependent hash functions. For instance, the distribution of a common object (more samples) may outweigh uncommon objects (few samples). In this case, unsupervised data dependent functions will be biased toward representing the sample distribution of the common object. Other advantages of the data independent hash functions are the fast learning process, which is independent from the gallery size, and the enrollment of new samples, which does not require retraining hash functions.

Data dependent hash functions select a family $\mathcal{H}$ considering aspects of the data, such as discriminability among different labels and dimensions with maximum energy. In this case, hash functions unrelated to the data are discarded, which is not the case in data independent hash functions. Considering the same number of hash functions employed in the data independent approach, the num-

ber of relevant hash functions which raise the gap between higher p_1 and lower p_2 is often higher in data dependent hash functions. Examples of works employing data dependent hash functions include metric learning [36], k-means [32], spectral hashing [37], restricted Boltzmann machine [38], maximum margin [39] and deep learning [40].

There are numerous LSH approaches for different metric spaces. The most common applications include LSH approaches for l_p metric space [32] based on p-stable distributions; random projections [41], which approximate cosine distances; Jaccard coefficient [42]; and Hamming distances [43],

It is important to emphasize that the proposed approach is not included in the LSH family. We do employ hash functions generated independently from each other and the proposed approach considers data labels, but there is no associated distance metric and, therefore, no approximated k-NN solution. We focus on returning correct identities in a shortlist of candidates rather than approximating nearest neighbors in a given metric space.

The proposed approach also behaves similarly to LSH methods, where the increase in the number of hash functions provides improved results, but we cannot prove the approximation limits of the proposed approach in the same way as in LSH. In our experiments, we notice that the results never exceed the recognition rate of the brute-force based on PLS, which might indicate that the proposed method approximates the results from PLS-based approaches.

3. Methodology

This section describes the methods considered in the proposed approach, namely PLS for regression (Section 3.1) and PLS for face identification (Section 3.2). The proposed PLSH is described in Section 3.3 and in Section 3.4, we describe a PLSH extension (ePLSH), which consists in employing PLS-based feature selection to improve the performance of PLSH.

3.1. Partial least squares regression

PLS is a regression method that combines ordinary least squares applied to a latent subspace of the feature vectors. Several works have employed PLS for face identification [4], face verification [3], and open-set face recognition [6]. These works consider PLS mainly due to the robustness to combine several feature descriptors, capability to deal with thousands of dimensions, and robustness to unbalanced classes. In this work, we consider PLS due to the high accuracy presented when used to retrieve candidates in PLSH and the low computational cost to test samples since only a single dot product between the regression coefficients and the feature vector is necessary to estimate the PLS response.

PLS is calculated as follows. The p-dimensional latent subspace is estimated by decomposing the zero mean matrices $X_{n\times d}$, with n feature vectors and d dimensions, and Y_n, with response values, in

$$\begin{aligned} X_{n\times d} &= T_{n\times p}P^T_{d\times p} + E_{n\times d}, \\ Y_{n\times 1} &= U_{n\times p}Q_{p\times 1} + F_{n\times 1}, \end{aligned} \tag{1}$$

where $T_{n\times p}$ and $U_{n\times p}$ denote latent variables from feature vectors and response values, respectively. The matrix $P_{d\times p}$ and the vector Q_p represent loadings and the matrix E and the vector F are residuals from the transformation. PLS algorithms compute P and Q such that the covariance between U and T is maximum [44]. We consider the nonlinear iterative PLS (NIPALS) algorithm [45] which calculates the maximum covariance between the latent variables $T = \{t_1, ..., t_p\}$ and $U = \{u_1, ..., u_p\}$ using the matrix $W_{d\times p} = \{w_1, ..., w_p\}$, such that

$$\arg\max[cov(t_i, u_i)]^2 = \arg\max_{|w_i|=1}[cov(Xw_i, Y)]^2.$$

The regression vector β between T and U is calculated using matrix W according to

$$\beta = W(P^TW)^{-1}(T^TT)^{-1}T^TY. \tag{2}$$

The PLS regression response $\hat{y}$ for a probe feature vector $x_{1\times d}$ is calculated according to $\hat{y} = \bar{y} + \beta^T(x - \bar{x})$, where $\bar{y}$ and $\bar{x}$ denote average values of Y and

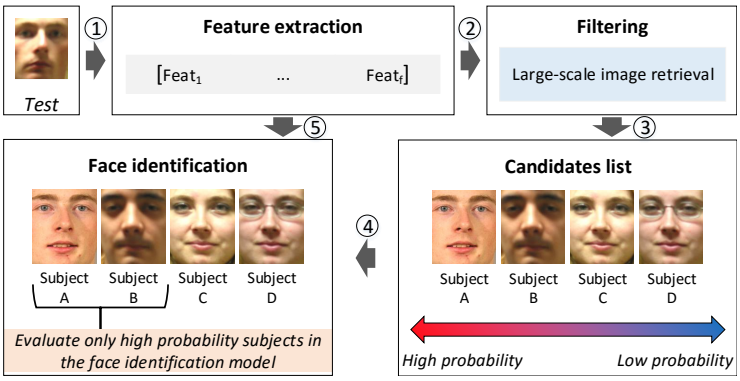


Figure 2: Overview of the filtering and the face identification pipeline. (1) Different feature descriptors are extracted from the test image and concatenated resulting in a large feature vector more robust to image effects than single feature descriptors. (2) The feature vector is presented to the filtering approach, which employs a large-scale image retrieval approach to (3) generate the candidate list sorted by the probability that the candidate is the subject in the test image. (4) A small percentage of the candidate list (high probability candidates) is presented to the face identification which will evaluate only the models relative to these candidates.

elements of X, respectively. The PLS model is defined as the variables necessary to estimate $\hat{y}$, which are β, $\bar{x}$ and $\bar{y}$.

Efficient implementations of the NIPALS algorithm using graphical cards exist in the literature and they can provide speedup of up to 30 times compared to the CPU version [46].

3.2. Face identification based on partial least squares

The proposed approach consists in filtering subjects in the gallery using methods for large-scale image retrieval. For a given face identification approach, the evaluation of all subjects in the gallery (without filtering) is regarded as the *brute-force* approach, which is undesirable since the asymptotic time complexity is linear with the number of subjects enrolled in the gallery. The filtering

approach consists in providing a shortlist to the face identification so that it evaluates only subjects presented in that shortlist.

An overview of the filtering and face identification pipeline is presented in Figure 2, which consists of the following steps. Different feature descriptors are extracted from a probe sample and concatenated in the first step (feature extraction). Then, the combined feature vector is presented to the filtering step, which employs large-scale image retrieval methods to generate a list of candidates sorted in decreasing order of probability that the candidate is the subject in the probe. Then, a small number of high probability candidates in the list is provided to the face identification method, which evaluates subjects following the order in the candidate list until the face identification returns a subject in the face gallery. In this case, speedup is achieved because it is not necessary to evaluate the remaining subjects in the candidate list once a gallery match is found, reducing therefore, the computational cost compared to the brute-force approach.

To evaluate the filtering and face identification pipeline, we consider the face identification method described by Schwartz et al. [4], which consists in employing a large feature set concatenated to generate a high dimensional feature descriptor. Then, a PLS model is learned for each subject in the gallery following a *one-against-all* classification scheme: samples from the subject are learned with response equal to $+1$ and samples from other subjects with response equal to -1. Test samples are presented to each PLS model and associated to the identity related to the model that returns the maximum score. We consider the evaluation of all PLS models as the brute-force approach and, in the proposed pipeline, only PLS models that correspond to subjects in the candidate list are evaluated.

3.3. Partial least squares for face hashing (PLSH)

The PLSH method is based on two principles: (i) data dependent hash functions and (ii) hash functions generated independently among each other. Data dependent hash functions provide better performance in general (see discussion

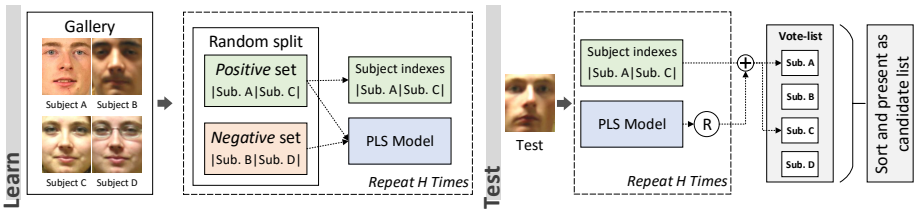


Figure 3: Overview of PLS for face hashing (PLSH) with (left) train and (right) test steps. In the train, a PLS regression model is learned to discriminate between two balanced random subsets of subjects in the face gallery (positive and negative subsets). In the test, the test sample is presented to each PLS model to obtain a regression response r. Then, a vote-list, initially zero, is incremented by r in each position corresponding to subjects in the positive subset.

in Section 2.2.1). Hash functions generated independently are necessary to induce uniform distribution of binary codes among subjects in the gallery [39]. A diagram of the PLSH method is presented in Figure 3.

PLSH consists of the learn and the test steps. In the learn, for each hash model, subjects in the face gallery are randomly divided into two balanced subgroups, *positive* and *negative*. Then, a PLS regression model, regarded as hash function in this work, is learned to discriminate the subjects in the positive subset (response $+1$) from the subjects in the negative subset (response -1). The association of one subject to one of the two subsets consists in sampling from a Bernoulli distribution with parameter p equal to 0.5 and associating that subject to the positive subset in case of "success". Note that, the association to each subset can be viewed as a bit in the Hamming embedding and the Bernoulli distribution with p equal to 0.5 is important to distribute the Hamming strings uniformly among the subjects in the face gallery. A PLSH *hash model* is defined as a PLS model and the subjects in the positive subset necessary to evaluate the test samples.

In the test, the test sample (probe sample) is presented to each PLSH hash

model to obtain a regression value r. We define a vote-list of size equal to the number of subjects in the gallery initially with zeros, then, each position of the vote-list is increased by r according to the indexes of subjects in the positive subset of the same PLSH hash model. Note that this scheme allows us to store half of the subject indexes to increment the vote-list since it will be equivalent to increment subjects in the negative set by $|r|$ when r is negative (the differences among pairs of votes will be the same). Finally, the list of subjects is sorted in decreasing order of values and presented as candidates for the face identification.

In practice, the majority of subjects with low values in the candidate list are discarded because they rarely corresponds to the test sample. The candidate list only serves to indicate the evaluation order for the face identification method. In this case, if an identity is assigned to the probe when evaluating the first candidates in the list, there is no need to evaluate the remaining candidates.

PLSH is similar to the work of Joly et al. [39], in which SVM classifiers are employed to determine each bit in the Hamming embedding. The advantage of employing PLS in this case is the robustness to unbalanced classes and support for high dimensional feature descriptors [6]. We do not provide approximation bounds to PLSH as LSH methods because PLSH is based on regression scores rather than distance metrics, which are not compatible with the LSH framework.

3.3.1. Consistency

The consistency of the PLSH algorithm with the goal to discriminate among the subjects in the face gallery is given as follows. In one hand, if r is approximately equal to +1 in the test, the probe sample is more similar to the subjects in the positive subset and the positions in the vote-list corresponding to the subjects in the positive subset will receive more votes. On the other hand, if r is approximately equal to -1, the votes in the vote-list corresponding to subjects in the positive subset will be decremented. If r is close to zero then the vote-list will not change significantly. Assuming that be equal to +1 whenever the correct subject in the test sample is in the positive subset, even if other subjects in the positive subset receive the same vote, their respective votes in

the vote-list will be decrement whenever they are not in the same subset as the correct subject.

Note that the aforementioned statement holds for a large number of hash functions since the probability of at least two subjects being in the same subsets is negligible. A large number of hash functions also mitigate the problem of a few hash functions not returning r roughly equal to 1 even if the correct subject is in the positive subset and the increase in the number of hash functions is limited only by the computational cost to evaluate them.

3.3.2. Hamming embedding

We do not estimate the Hamming embedding directly since there is no binary string associated to any face sample. However, PLSH is equivalent to estimating the Hamming embedding for a test sample and comparing it with the binary strings generated for each subject in the gallery. In addition, each bit of the test binary string is weighted by the absolute value of the PLS regression response.

To demonstrate the aforementioned claims, consider that PLS responses can be only $+1$ or -1, such that any test sample can be represented by the sequence $X = \{+1, -1\}^H$, where H denotes the number of PLSH hash models. Consider also that each subject s in the face gallery is represented by the binary string $Y_s = \{1, 0\}^H$, where $y_i \in Y_s$ is set to 1 if the subject s was associated to the positive subset of the i-th PLSH hash model in the train step, or 0, otherwise. In this context, the weight w_s given by PLSH to each subject in the gallery is calculated as

$$w_s = \sum_{i=1}^{H} x_i y_i.$$

Note that the maximum w_s is equal to the sum of $+1$ elements in X, which occurs when $y_i = 1$, if $x_i = +1$, and $y_i = 0$, otherwise. Similarly, the minimum weight is equal to the sum of -1 elements in X, which occurs when $y_i = 1$, if $x_i = -1$, and $y_i = 0$, otherwise. If we transform X onto a binary string $\hat{X}$ such that $\hat{x_i} = 1$, if the corresponding x_i is $+1$, and $\hat{x_i} = 0$, otherwise; we can calculate the Hamming distance between $\hat{X}$ and Y_s. In fact, the exactly same

Hamming distance can be calculate using w_s as

$$d(X,Y)_{\mathbb{H}} = w_{\max} - w_s, \tag{3}$$

where $w_{\max}$ denotes maximum possible w_s. The same analogy can be applied to the weighted Hamming distance if we consider x_i assuming any real number. In this case, the weight of each bit α_i is the absolute value of r and the weighted Hamming distance is equivalent to Equation 3.

3.3.3. Computational requirements

The amount of space necessary for the PLS algorithm depends on the number of hash models H, the dimensionality of the data D and the number of subjects in the gallery N. Each hash model holds a PLS regression vector in R^D and the indexes of subjects in the positive subset $(N/2)$, therefore, $H \times D$ real numbers and $(H \times N)/2$ integer indexes of space are necessary. Note that it is not necessary to store the feature vectors used to train the PLS models in the test and they can be safely discarded since the PLS regression vector holds the necessary information to discriminate among the enrolled subjects.

The computational time necessary to evaluate a test sample in the PLSH algorithm depends on the dot product between the PLS regression vectors from all hash models and the feature vector, which is accomplished with $D \times H$ multiplications. Then, there is the computational time to sort the vote-list, which has asymptotical cost $O(Nlog(N))$. It is possible to reduce the computational time to sort the vote-list by eliminating all negative values from the vote-list before sorting it and without any impact on the results [7]. However, since the computational time needed to evaluate all the hash functions is considerably higher than the time spent to sort the vote-list, we do not employ this heuristics in our experiments.

3.3.4. Alternative implementations

In principle, some aspects of the PLSH algorithm can be changed such that PLSH can provide potential performance improvement. For instance, the parameter p of the Bernoulli distribution used to determine the subsets of subjects

may be changed given that PLS hardly finds common discriminative features among subjects in a large set [6]. However, changing p from 0.5 to other value results in a nonuniform distribution of subjects among subsets (raise hash table collisions), therefore, reducing the accuracy. As demonstrated in our previous work [7], maintaining a balanced subset of subjects to learn each hash model ($p = 0.5$) provide the best results.

Another possible implementations of PLSH that does not modify much the results is the product of votes instead of the sum, which is akin to the intersection of subsets among all hash functions. It is also possible to employ multiple partitions instead of only two by using a categorical rather than Bernoulli distribution. However, multiple partitions present no significant difference in the results and they require twice the space requirement since the indexes of subjects that were learned with +1 target response in the PLS model need to be stored to allow them to receive the votes in the test.

The computational cost to evaluate the hash functions can be reduced by calculating the PLS regression value using the few discriminative dimensions in the feature vector. As will be presented in the experiments, the feature selection include a new parameter in the PLSH algorithm, the number of features selected, which can be estimated jointly with the number of hash functions to provide much better results than in PLSH without feature selection.

3.4. Feature selection for face hashing (ePLSH)

The algorithms for PLSH described in Section 3.3 require a dot product between the PLS regression vector and the feature descriptor to calculate each hash function. This section describes methods to reduce the computational cost to evaluate hash functions. To discriminate PLSH with the feature selection version and to maintain consistence with the nomenclature given in our publications, PLSH with feature selection is called *extended PLSH* (ePLSH) in the rest of this work.

In practice, ePLSH is equivalent to PLSH when all features are considered to evaluate hash functions. The main advantage of ePLSH is the possibility of

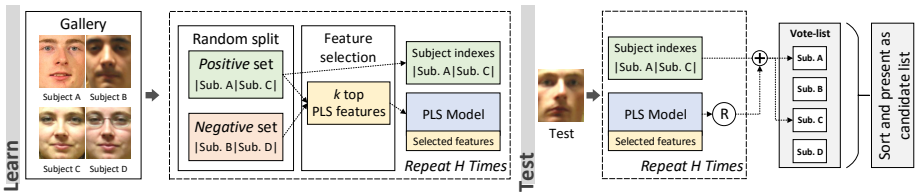


Figure 4: Overview of PLS for face hashing and feature selection (ePLSH) with (left) train and (right) test steps. The train consists of the same procedures employed in PLSH with the difference of the feature selection method based on the top discriminative features between the positive and negative subsets. The indexes of the selected features are stored along with the PLS model and used in the test to calculate an approximate PLS regression score.

employing thousands of additional hash functions, resulting in considerable increase of the recognition rate while keeping low computational cost to calculate the hash functions. The common feature setup considered in the PLSH and in the ePLSH approaches consists in combining four feature descriptors, which leads to a feature vector with 120,059 dimensions. However, we show in our experiments that, for the feature set considered in this work, about 500 dimensions with an increased number of hash functions provides better candidate lists than PLSH with about the same computational cost. A summary of ePLSH is presented in Figure 4.

The ePLSH consists of two steps: *train* and *test*. In the train, it calculates the β regression vector following the same procedure of PLSH. Then, the indexes of the k more discriminative features are stored. Considering that the range of values in the feature vector is known (zero mean and unit variance in our experiments), it is possible to calculate an approximated score using only the more discriminative features. However, if only such features are used to calculate the regression value without rebuilding the PLS model, the result would not be accurate because of the large number of remaining features, even though they present a very low contribution individually. To tackle this issue, we learn a new PLS model to replace the full feature version in PLSH, which is performed

by eliminating the dimensions from the matrix X that do not correspond to the k select features and recalculate β using Equation 2.

We define the ePLSH *hash model* as the PLS model, the subjects in the positive subset and the k selected features. Finally, the test step is carried in the same manner as in PLSH, but with the difference that only features selected in the ePLSH hash model are considered to calculate the regression score.

There are numerous works about PLS-based feature selection in the literature and they are divided in three categories [47]: *filter*, *wrapper* and *embedded*. Filter methods are the simplest of the three and work in two steps. First, the PLS regression model is learned and, then, a relevance measure based on the learned PLS parameters is employed to select the most relevant features. Wrapper methods consist in an iterative filter approach coupled with a supervised feature selection method. Finally, embedded methods consist in nesting feature selection approaches in each iteration of the PLS algorithm. We suggest the work presented by Mehmood et al. [47] for a comprehensive list and description of PLS feature selection methods.

In this work, we focus on PLS filter methods for feature selection for simplicity reasons. However, ePLSH is defined without lost of generality such that other PLS feature selection methods could be considered if necessary. Mehmood et al. [47] describe three filter methods called loading weights (W), variable importance on projection (VIP) and regression coefficients (β). These methods are described in Sections 3.4.1, 3.4.2 and 3.4.3, respectively.

3.4.1. Loading weights

The idea in the loading weight approach is, for each PLS component, to select the features associated with higher absolute w_i value (alternately features above a threshold [47]). Recall W being the output of NIPALS algorithm[1] used to calculate latent variables. In this way, the absolute coefficient $w_{f,i} \in W$, for the f-th PLS component and the i-th feature, is directly associated to the f-th

[1] see Section 3.1 for the PLS description

latent variable. Note that one feature may be relevant to one PLS component and irrelevant for another, specially because the latent variable basis represented by W is orthonormal. Therefore, the goal is to find the set of features that are relevant to calculate at least one PLS latent variable. In this context, the loading weight method consists in selecting features $i \in [1, N]$ with highest relevance measure defined as $\max_{f=1:p}(w_{p,i})$.

3.4.2. Variable importance on projection

Variable importance on projection (VIP) consists in calculating a measure that summarize the loading weights (W) of all factors for each dimension in the feature vector. VIP measure is calculated as[2]

$$v_i = \sqrt{n \sum_{f=1}^{p} \left(b_f^2 w_{f,i}^2 \right) / \sum_{f=1}^{p} b_f^2}. \tag{4}$$

In our experiments, the product by n can be ignored since n is constant for all features. In this case, the VIP measure will not be normalized and the common VIP threshold described in [47], which determines that relevant features present VIP higher than 0.8, cannot be employed directly. Recall b_i as proportional to the covariance between projected features and target values. The sum in the numerator of Equation 4 represents the squared sum of the loading coefficients for a specific feature weighted by the predictive capacity of each coefficient. In this way, the main difference between the loading weights and the VIP approaches is the employment of b_i in the latter.

3.4.3. Regression coefficients

Regression coefficients for feature selection is the simplest of the three filter methods and consists in using the regression vector directly to select the most relevant features. Recall from Section 3.1 the regression vector as $\beta = W(P^T W)^{-1}(T^T T)^{-1} T^T Y$, where P and T are loading matrices from features and target values, respectively. Similar to the loading weights approaches,

[2] see Section 3.1 for variable definitions.

regression coefficients are also related to predictive capacity of the latent variables to estimate target values, however, in a more transparent manner since they are directly employed to estimate the regression values. The main difference from the regression coefficient and the aforementioned filter approaches is the correlation of the latent variables with target values embedded in the PLS regression vector, which provides a small improvement over the loading weights and VIP results.

3.5. Early-stop search heuristic

To stop the search for the correct subject in the candidate list, we employ the heuristic described by Schwartz et al. [4]. For a short number of initial samples (15), all subjects in the candidate list are evaluated and the median value of the scores is taken as threshold for the remaining test samples. Then, subjects in the candidate list are evaluated until a score equal or higher than the threshold is obtained or the end of the list is reached.

Note that, in practice, the candidate list size is a percentage of the subjects enrolled in the gallery and most of the candidates with low weights can be discarded because they rarely corresponds to the probe sample. In this case, the worst case scenario consists in evaluating all subjects in the candidate list for every probe sample. However, the early-stop search heuristic alone is shown to reduce the number of tests in the face identification up to 63% without degrading the recognition rate so the speedup achieved is usually higher than the ratio of the gallery size divided by the number of subjects in the candidate list.

4. Experimental results

In this section, we evaluate PLSH and ePLSH in two standard face identification datasets (FERET and FRGCv1). Section 4.1 contains the common experimental setup, including datasets, number of dimensions in PLS models for the face identification, PLSH and ePLSH, evaluation metric, description of the computer used in the experiments, and feature descriptors. The PLSH parameter validation is presented in Section 4.2. The parameter validation for

ePLSH is discussed in Section 4.3. Evaluation on the datasets and comparisons with other methods in the literature are presented in Section 4.4 (FERET) and in Section 4.5 (FRGCv1).

4.1. Experimental setup

All experiments regarding parameter validation in Sections 4.2 and 4.3 were performed on the FERET dataset, since it is the dataset with the largest number of subjects (1,196 in total). FERET consists of four test sets and we use *dup2* in Sections 4.2 and 4.3, which is considered the hardest of the dataset. The only exception is the experiment regarding the number of hash models and the gallery size in Section 4.3.3, where *fb* test set was employed since it provides more test samples (1,195) than the others (194, 722 and 234 in *fc*, *dup1* and *dup2*, respectively).

The experiments were conducted using an Intel Xeon X5670 CPU with 2.93 GHz and 72 GB of RAM running Ubuntu 12.04 operating system. All tests were performed using a single CPU and no more than 8 GB of RAM were necessary.

4.1.1. FERET dataset

The facial recognition technology (FERET) dataset [48] consists of 1,196 images, one per subject for training, and four test sets designed to evaluate the effects of lightning conditions, facial expression and aging on face identification methods. The test sets are: *fb*, consisting of 1,195 images taken with different facial expressions; *fc*, consisting of 194 images taken in different lightning conditions; *dup1*, consisting of 722 images taken between 1 minute and 1,031 days after the gallery image; *dup2*, is a subset of *dup1* and consists of 234 images taken 18 months after the gallery image. In our experiments, all images were cropped in the face region using annotated coordinates of the face, scaled to 128×128 pixels and normalized using the self-quotient image (SQI) method to remove lightning effects [49].

4.1.2. FRGC dataset

The face recognition grand challenge dataset (FRGC) [50] consists of 275 subjects and samples that include 3D models of the face and 2D images taken with different illumination conditions and facial expressions. We follow the same protocol described by Yuan et al. [25], which considers only 2D images and consists in randomly selecting different percentages of samples from each subject to compose the face gallery and using the remaining samples to test. The process is repeated five times and the mean and standard deviation of the rank-1 recognition rate and speedup (considering the brute-force approach) are reported. The samples were cropped in the facial region, resulting in size 138×160 pixels, and scaled to 128×128 pixels.

4.1.3. Evaluation metric (MARR)

According to the face identification pipeline presented in Section 3.2, the candidate list calculated in the filter approach (PLSH and ePLSH) is employed to reduce the number of PLS models evaluated in the face identification. In this context, the error rate of the pipeline results from errors induced by the filter approach (fail to return identity of test sample in the candidate list) and by the face identification approach (fail to identify correctly the subject in the candidate list). Therefore, to assess the performance of the filter approach alone, we provide results considering the *maximum achievable recognition rate* (MARR), which is calculated considering that a perfect face identification method is employed for different percentages of candidates visited in the list.

Note that the MARR value is the upper bound for the recognition rate achieved by the filter and face identification pipeline. Figure 5 illustrates the MARR evaluation metric where better results present MARR close to one and low percentage of candidates visited (curves close to the upper left corner of the plots).

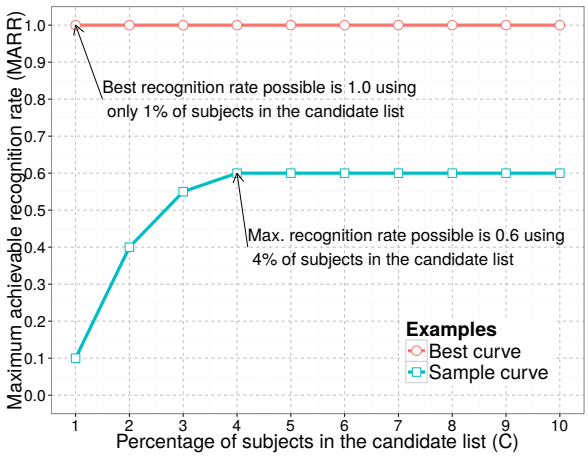


Figure 5: An example of the plots regarding the MARR evaluation metric for two sample curves. The MARR metric (vertical axis) considers that the candidate list is presented to an ideal face identification method, therefore, providing the upper bound of the recognition rate achievable when considering a given filtering method such as PLSH and ePLSH. The horizontal axis presents different percentages of the candidate list that are presented to the face identification approach. The best curve presents MARR equal to one for any percentage of subjects in the candidate list.

4.1.4. Number of dimensions in the PLS models

PLS-based face identification requires only one parameter, the number of dimensions in the PLS latent space (p). Schwartz et al. [4] evaluated p by varying it from 13 to 21 without noticing large variation in the results. Therefore, we set p to 20 for the face identification method in our experiments. We conducted experiments in PLSH by varying p between 4 and 19, in steps of 3, and we did not noticed large difference in the results for p between 7 and 19. Therefore, for PLSH and ePLSH, we set p to 10.

4.1.5. Feature descriptors

We consider four feature descriptors in this work, CLBP [51], Gabor filters [52], HOG [53] and SIFT [18], which mainly captures information about

texture and shape of the face image. This set of features was chosen because they present slightly better results in the face identification and indexing compared to the previous works [4, 7].

On the CLBP feature descriptor, we set the radius parameter to 5, which is the common parameter employed in face recognition tasks. CLBP histograms are calculated in a sliding window approach with size equal to 16 pixels and stride equal to 8 pixels. We also consider accumulating all *normal codes* (codes with more than 2 transitions between bits) in the same histogram bin to reduce the dimensionality. The final descriptor is the concatenation of all histograms in the face image, resulting in $9,971$ dimensions and taking 118 milliseconds, on average, to calculate.

To compute Gabor filters, we convolve the face image with filters of size 16×16 pixels, 8 scales, equally distributed between $[0, \frac{\pi}{2}]$, and 5 orientations, equally distributed between $[0, \pi]$, which results in 40 convolved images. The images were downscaled by a factor of 4 and concatenated to assemble the final feature descriptor, resulting in $40,960$ dimensions and taking $1,475$ milliseconds to calculate per face image, on average.

Two feature setups are considered for HOG. The first setup consists in block size equal to 16×16 pixels, stride equal to 4 pixels and cell size equal to 4×4 pixels. The second setup consists in block size equal to 32×32 pixels, stride equal to 8 pixels and cell size equal to 8×8 pixels. The feature descriptor consists in concatenating the HOG descriptors from the two setups, resulting in $36,360$ dimensions and taking 81 milliseconds to calculate per face image, on average.

We consider SIFT descriptors calculated in 256 keypoints evenly spaced in the face image. We employed the default parameters employed by Lowe [18], which are 4×4 histogram cells, each with 8 bins, contrast threshold 0.04, Gaussian smoothness 1.6 and edge threshold 10. The final feature descriptor is the concatenation of all SIFT descriptors in the face image and has $32,768$ dimensions with average time to calculate equal to 30 milliseconds.

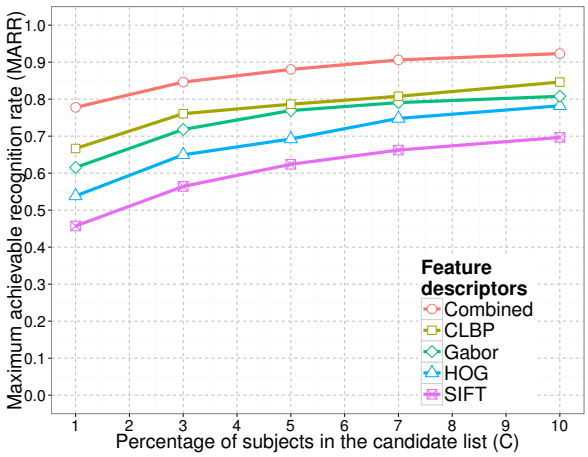


Figure 6: Evaluation of the CLBP, Gabor, HOG and SIFT feature descriptors and their combination.

4.2. PLSH parameters validation

Herein we evaluate the aspects regarding PLSH model and parameter selection. In Section 4.2.1, we evaluate each single feature descriptor with their combination. In Section 4.2.2, we evaluate different numbers of hash models. In Section 4.2.3, we discuss stability regarding PLSH results.

4.2.1. Combination of feature descriptors

Figure 6 presents the MARR curves for each of the four feature descriptors considered in this work and their combination. The number of hash models in this experiment was empirically set to 150. According to Figure 6, the combination of CLBP, Gabor, HOG and SIFT is responsible for an increase of about 10 percentage points (p.p.) in MARR compared to the best individual feature descriptor (CLBP). Therefore, we employ the combination of these feature descriptors in the remaining experiments. The combined feature descriptor has $120,059$ dimensions with averaged time to calculate equal to 1.7 seconds. It is important to point out that the time spent to calculate the feature descriptors for a probe sample is constant (it does not depend on the number of subjects

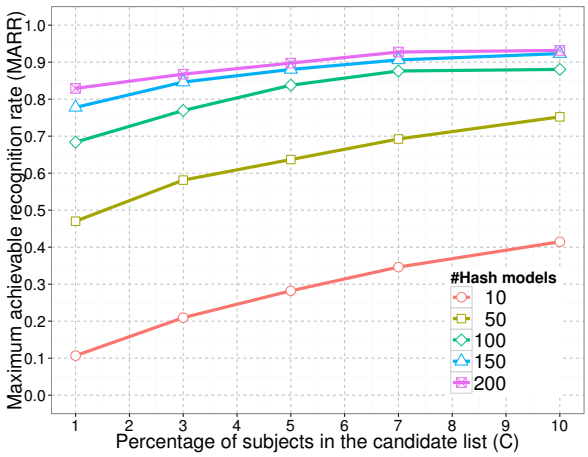


Figure 7: Number of hash models as a function of the MARR for different percentages of subjects in the candidate list.

enrolled in the face gallery). In fact, the computational time to extract the feature descriptors can be adapted in exchange for reduced MARR. For instance, Gabor filters could be discarded to reduce the computational time to extract the features since they take 1.4 seconds per face image to calculate, on average.

4.2.2. Number of hash models

Figure 7 presents MARR for a number of hash models equal to $10, 50, 100, 150$ and 200. According to the results, a large improvement in MARR (for any number of subjects in the candidate list) takes place when the number of hash models increases from 10 to 150 can be seen in Figure 7. However, the increase in MARR is negligible when the number of hash models is raised from 150 to 200. Since the face identification and the PLSH approaches depend on a single dot product between the feature and the PLS regression vectors, the computational cost to evaluate each hash function in PLSH is about the same as the cost to evaluate each subject in the gallery. Therefore, to obtain a low computational cost for testing samples, we consider 150 hash functions in the remaining PLSH experiments. As a reference, the average time to evaluate each hash function in this experiment was 426 microseconds.

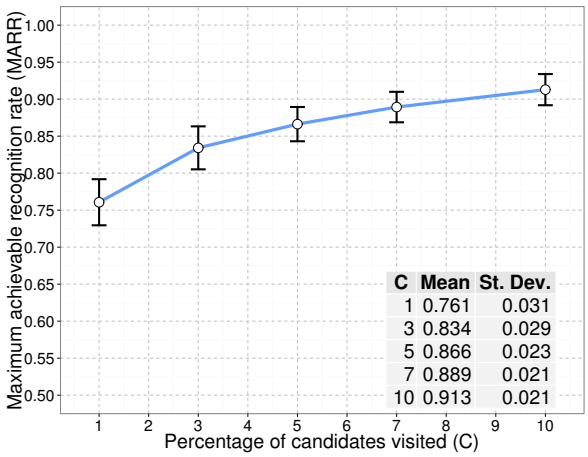


Figure 8: Average MARR and standard deviation for 10 PLSH runs considering 1% of subjects in the candidate list.

4.2.3. Stability of the results

Figure 8 presents the mean MARR and standard deviation when running PLSH 10 times. Although PLSH is a nondeterministic method, it still provide fair stability, assessing that all experiments performed in this sections are easily reproducible. For instance, the best individual feature descriptor in Section 4.2.1, Gabor filter, provides MARR (at 1% of subjects in the candidate list) equal to 0.67, which is considerable lower than the averaged 0.76 MARR presented in Figure 8. The conclusion is that even with the variation in the results from the feature combination, PLSH rarely presents MARR equals to 0.67, assessing that the combination of features is better than individual features.

4.3. ePLSH parameters validation

In this section, we conduct experiments regarding stability and scalability of ePLSH in Sections 4.3.4 and 4.3.3, respectively. The feature selection methods described in Section 3.4 are evaluated in Section 4.3.2. A discussion regarding the number of features selected is presented in Section 4.3.1.

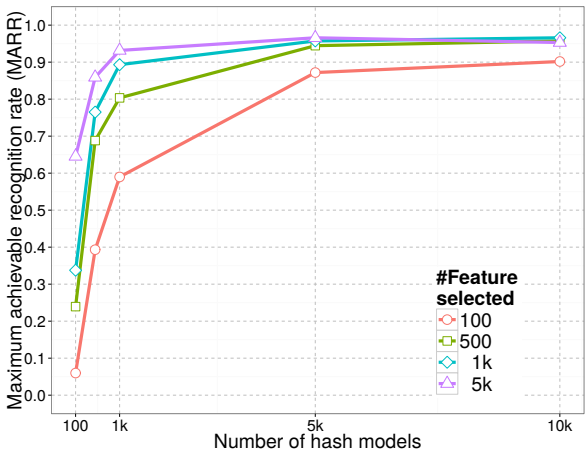


Figure 9: MARR with different numbers of hash models and the feature selected in the ePLSH.

4.3.1. Number of hash models and features selected

Figure 9 presents MARR for 1% of subjects in the candidate list for different numbers of hash models (m) and selected features (d). The ePLSH aims at reducing the computational cost to evaluate PLSH hash functions, which can be roughly approximated to a number of multiplication operations equal to $m \times d$. It is important to point out that d equal to 500 provides nearly the same MARR for a sufficient large enough m. Therefore, we fix d to 500 and vary m for different datasets and number of subjects in the face gallery.

We achieve minimum computational cost with almost maximum MARR using $5,000$ hash models and with 2.5 million multiplications. Note that this number of multiplications refers only to the ePLSH approach such that the total computational cost of the pipeline also includes the number of multiplications in the face identification. As a comparison, the number of multiplications necessary in the brute-force approach for the $1,196$ subjects in the gallery is $1,196 \times 120,059 = 143.5$ millions, which is about 57 times more than the number of multiplications necessary to calculate all of the $5,000$ ePLSH hash functions.

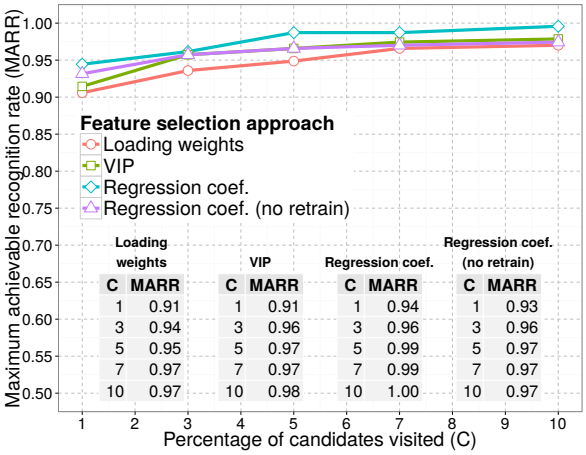
The time spent to calculate each ePLSH hash function is considerable lower

than PLSH hash functions. Since both approaches consists in a dot product between the feature vector and the regression vector, the number of multiplications needed to compute each hash function is equal to the dimensionality of the feature vector in PLSH ($120,059$ multiplications) and equal to the number of selected features in ePLSH (500 multiplications). In this way, ePLSH hash functions should be theoretically 240 times faster than PLSH hash functions. However, the nonlinear access to the feature vector in ePLSH hash functions may induce an additional overhead due to the weak locality of reference (accessing positions in the memory that are far from each other).

The average time to calculate each PLSH hash function is 446 microseconds compared to 12 microseconds for each ePLSH hash function. However, since a considerable number of hash functions is employed in ePLSH compared to PLSH, the time to train ePLSH is significant higher than PLSH. The time spent to train all the $5,000$ hash functions in ePLSH is 14 hours compared to 22 minutes for the 150 hash functions in PLSH, which may not impose an issue because the train is performed offline and only once for a fixed face gallery. The train can also be accelerated considering other PLS algorithms such as SIMPLS [54] rather than NIPALS.

4.3.2. Feature selection

In this section, we compare the feature selection approaches described in Section 3.4. We also compare whether we should retrain the PLS model in the regression coefficients approach to redistribute weights from the discarded feature among the selected features. According to the results presented in Figure 10, there is no significant difference in the feature selection approaches evaluated. Considering the experiments in Section 4.3.4, the MARR at 1% of subjects in the candidate list for the regression coefficients approach vary roughly between 0.92 and 0.96. In this case, it can be concluded that the regression coefficients approach is better than the loading weights and VIP. Furthermore, there is no significant difference between retraining or not the hash model regression coefficients after the feature selection step.



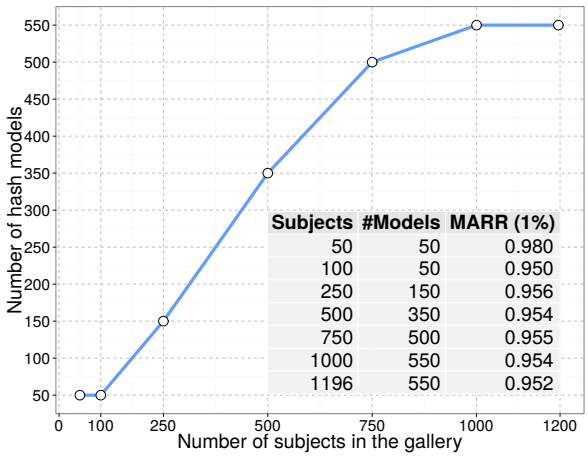
Loading weights		VIP		Regression coef.		Regression coef. (no retrain)	
C	MARR	C	MARR	C	MARR	C	MARR
1	0.91	1	0.91	1	0.94	1	0.93
3	0.94	3	0.96	3	0.96	3	0.96
5	0.95	5	0.97	5	0.99	5	0.97
7	0.97	7	0.97	7	0.99	7	0.97
10	0.97	10	0.98	10	1.00	10	0.97

Figure 10: Evaluation of loading weights, variable importance on projection (VIP) and regression coefficients for feature selection.

4.3.3. Number of hash models and gallery size

In all experiments presented so far, we considered a fixed number of subjects in the gallery, which are in total $1,196$ for the FERET dataset. We still need to assess the ePLSH performance with an increasing number of subjects in the face gallery, which, theoretically, should require a logarithmic number of hash models to index the subjects in the face gallery [7]. For the experiment in this section, we randomly select $50, 100, 250, 500, 750, 1000$ subjects in the FERET dataset to be enrolled onto the face gallery. We consider the *fb* test set in FERET because it has more test samples ($1,195$ in total) and ePLSH because it provides more stable and better results than PLSH. We also consider only test samples of subjects enrolled in the face gallery because we are evaluating the closed set recognition. We raise the number of hash models from 50 to 550, in steps of 50, until we reach at least 0.95 MARR for 1% of subjects in the candidate list.

The results in Figure 11 demonstrate that at least the number of hash models necessary to maintain accuracy is logarithmic with the size of the face gallery. However, the number of subjects in the candidate list still depend on 1% of the



Subjects	#Models	MARR (1%)
50	50	0.980
100	50	0.950
250	150	0.956
500	350	0.954
750	500	0.955
1000	550	0.954
1196	550	0.952

Figure 11: Number of hash models necessary to provide at least 0.95 MARR with different gallery sizes and 1% of subjects in the candidate list.

gallery size. The problem in this case is that the face identification still needs to evaluate 1% of subjects in the face gallery, considering the worst case of the early-stop heuristic. We tried varying the percentage -or fixing to a small value- the number of subjects in the candidate list, but for both cases, the number of hash models did not stabilize for the number of enrolled subjects evaluated. We believe this happens because the number of subjects in our evaluation is not large enough to demonstrate convergence of the number of hash models. Nonetheless, Figure 11 indicates that we can reduce at least in two orders of magnitude the number of subjects evaluated in the face identification, which is so far, the best known result in the literature as will be presented in the next sections.

4.3.4. Stability of the results

The same experiment regarding stability of the PLSH results is performed for ePLSH in this section. The averaged MARR and standard deviation for 10 ePLSH runs are presented in Figure 12. We are considering regression coefficients for feature selection in this experiment and we retrain the PLS model after the feature selection step as discussed in Section 3.4. In this case, the

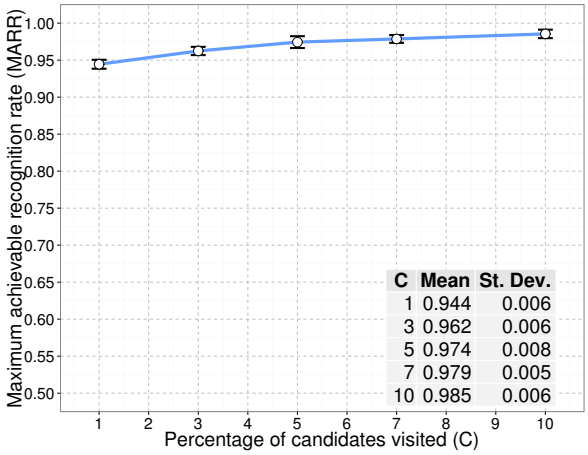


Figure 12: Average MARR and standard deviation for 10 ePLSH runs considering 1% of subjects in the candidate list.

ePLSH presents considerable more stable results than PLSH, with standard deviation around 0.006 compared to 0.03 in PLSH. We believe that the increase in stability is a consequence of the augmented number of hash models, which reduces the variance of the sum of scores in the vote-list, resulting in a more stable distribution.

4.4. Results on the FERET dataset

Results regarding MARR and rank-1 recognition rate for PLSH in all test sets from the FERET dataset are presented in Figures 13a and 13b. For the test sets *fb* and *fc*, about 1% of subjects in the candidates list is enough to achieve more than 95% of the rank-1 recognition rate of the brute-force approach (presented in the legend of Figure 13b for each test set). However, for the test sets *dup1* and *dup2*, about 5% of subjects in the candidate list ensured at least 95% of the brute-force rank-1 recognition rate. The theoretical speedup in the worst case can be calculated considering the 150 PLSH hash function evaluations and the 5% of the gallery size, which consists of 60 PLS projections. In this case, if the early-stop search heuristic is not considered, i.e., all subjects in the candidate list are evaluated for each test sample, the number of PLS projections

would be 210 compared to the $1,196$ projections necessary in the brute-force approach, which would still results in a 5.6 times speedup.

Results from ePLSH are presented in Figures 13c and 13d. Using only 1% of subjects in the candidate list, it is possible to recover all subjects in the rank-1 recognition rate from brute-force approach for all four test sets. In this case, the rank-1 recognition rate from the ePLSH pipeline is the same as the brute-force approach, but with reduction to 1% of the subjects evaluated in the identification. Considering that the cost to evaluate all hash models in ePLSH is about the same as in PLSH, the theoretical speedup is 7.38 times compared to the brute-force approach in the worst case.

4.5. Results on the FRGC dataset

Results from the FRGC dataset for PLSH and ePLSH are presented in Table 1 along with results from three other methods as presented in the literature. The three methods are the cascade of rejection classifiers (CRC) from [25], the PLS-based search tree [4], and our previous published work [7], which consists of PLSH with the combination of HOG, Gabor filter and LBP feature descriptors. For PLSH and ePLSH, we vary the number of hash models and the maximum percentage of subjects visited in the candidate list and we present the results with rank-1 recognition rate close to 0.95 and higher speedups. In this way, it is possible to compare directly the maximum speedup achievable when using PLSH and ePLSH compared to the other approaches, which also provide rank-1 recognition rate close to 0.95.

Results for a fixed setup that provide at least 0.95 rank-1 recognition rate are also provided, consisting of 50 hash models with 25% of subjects in the candidate list for PLSH and 200 hash models with 10% of subjects in the candidate list for ePLSH. The experiments were conducted with the following percentages of subjects in the candidate list (rounding up): $0.1, 0.5, 1, 3, 5, 7, 10, 13, 15, 20, 25, 30$. The number of hash models evaluated are: $10, 15, 20, 25, 30, 35, 40, 45, 50$; for PLSH, and $25, 50, 75, 100, 125, 150, 175, 200$, for ePLSH.

According to Table 1, it is possible to conclude that the rank-1 recognition rate is reasonably stable, with variance in the first decimal place, which is similar to the results regarding stability presented for PLSH and ePLSH. The speedup for PLSH and ePLSH decreases considerable as the number of samples per subject available for train reduce. The reason for that is the increase in the number of hash models and the maximum number of subjects visited in the candidate list to guarantee at least 0.95 rank-1 recognition rate. Even with reduced speedups considering 35% of samples available for train, ePLSH provides significant improvement over the speedup achieved by the tree-based approach (3.6 times faster), while PLSH provides competitive speedup.

The speedup provided by PLSH and ePLSH compared to the tree-based approach is noticed with 90% of the samples available for train, where PLSH is about 5 times faster than the tree-based approach while ePLSH is about 13 times faster than PLSH. Finally, in the worse case, ePLSH provides at least 14 times speedup considering the brute-force approach in the setup with 200 hash models and 10% of subjects in the candidate list.

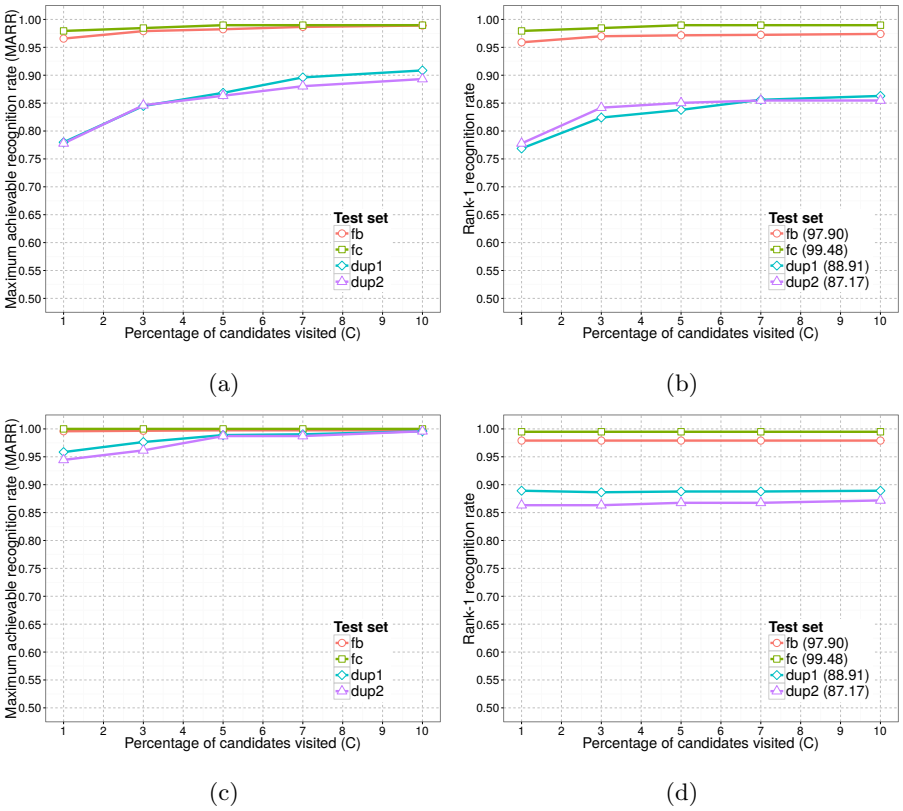


Figure 13: Results on the FERET dataset. (a) PLSH MARR curves, (b) PLSH rank-1 recognition rate, (c) ePLSH MARR curves and (d) ePLSH rank-1 recognition rate. Number in parenthesis indicate rank-1 recognition rate for the brute force approach.

	% of samples for train	90%	79%	68%	57%	35%
CRC [25]	Speedup	1.58×	1.58×	1.60×	2.38×	3.35×
	Rank-1 rec. rate	80.5%	77.7%	75.7%	71.3%	58.0%
Tree-based [4]	Speedup	3.68×	3.64×	3.73×	3.72×	3.80×
	Rank-1 rec. rate	94.3%	94.9%	94.3%	94.46%	94.46%
PLSH [7] HOG,Gabor filter LBP	Speedup	$(16.84 \pm 1.56)\times$	$(7.30 \pm 1.40)\times$	$(5.66 \pm 0.41)\times$	$(3.42 \pm 0.34)\times$	$(2.79 \pm 0.11)\times$
	Rank-1 rec. rate	$(96.5 \pm 0.7)\%$	$(96.7 \pm 1.6)\%$	$(93.4 \pm 1.3)\%$	$(93.6 \pm 0.5)\%$	$(93.3 \pm 0.7)\%$
	Hash models	10	20	25	35	35
	Max. candidates	3%	10%	13%	20%	30%
PLSH	Speedup	$(18.24 \pm 1.28)\times$	$(8.61 \pm 0.30)\times$	$(6.95 \pm 0.31)\times$	$(3.96 \pm 0.05)\times$	$(3.49 \pm 0.17)\times$
	Rank-1 rec. rate	$(95.31 \pm 0.62)\%$	$(95.31 \pm 0.70)\%$	$(93.60 \pm 1.15)\%$	$(94.67 \pm 0.34)\%$	$(94.60 \pm 0.16)\%$
	Hash models	10	20	30	50	50
	Max. candidates	3%	13%	13%	15%	25%
PLSH fixed params.	Speedup	$(2.95 \pm 0.03)\times$	$(4.00 \pm 0.16)\times$	$(4.13 \pm 0.30)\times$	$(3.16 \pm 0.03)\times$	$(3.49 \pm 0.17)\times$
	Rank-1 rec. rate	$(99.69 \pm 0.12)\%$	$(98.26 \pm 0.06)\%$	$(97.74 \pm 0.42)\%$	$(96.19 \pm 0.15)\%$	$(94.60 \pm 0.16)\%$
ePLSH	Speedup	$(\mathbf{233.61 \pm 37.05})\times$	$(\mathbf{98.93 \pm 8.39})\times$	$(\mathbf{45.42 \pm 3.84})\times$	$(\mathbf{22.29 \pm 1.03})\times$	$(\mathbf{14.21 \pm 1.74})\times$
	Rank-1 rec. rate	$(96.03 \pm 0.70)\%$	$(95.02 \pm 0.45)\%$	$(95.98 \pm 0.31)\%$	$(94.67 \pm 0.49)\%$	$(94.44 \pm 0.40)\%$
	Hash models	50	100	150	150	200
	Max. candidates	0.1%	0.5%	3%	5%	10%
ePLSH fixed params	Speedup	$(19.74 \pm 1.35)\times$	$(16.30 \pm 1.01)\times$	$(19.12 \pm 1.89)\times$	$(12.28 \pm 0.57)\times$	$(14.21 \pm 1.74)\times$
	Rank-1 rec. rate	$(99.79 \pm 0.22)\%$	$(98.30 \pm 0.11)\%$	$(97.63 \pm 0.04)\%$	$(96.71 \pm 0.36)\%$	$(94.44 \pm 0.40)\%$

Table 1: Comparison between the proposed approach and other approaches in the literature. The highest speedups are shown in bold. The fixed parameter setup is the same employed when evaluating 35% of samples for train (50 hash models with 25% of subjects in the candidate list for PLSH and 200 hash models with 10% of subjects in the candidate list for ePLSH).

5. Conclusions and future works

In this work, we proposed and evaluated PLSH and its extension ePLSH for face indexing. PLSH is inspired by the well-known locality-sensitive hashing for large-scale image retrieval and PLS for face identification, which provides fast and robust results for face indexing. Additional gain in speedup was achieved with the ePLSH, a method that employs PLS-based feature selection to reduce the computational cost to evaluate hash functions, enabling a large amount of additional hash functions to be employed and raising the indexing precision. We evaluated several parameters and alternative implementations of PLSH in the hope that they will be useful for future face indexing development. The experiments were conducted on two face identification standard datasets, FERET and FRGCv1, with $1,196$ and 275 subjects, respectively. Although these datasets do not provide enough number of subjects for a proper evaluation regarding scalability to large galleries, PLSH and ePLSH still provide significant improvement in speedup compared to other scalable face identification approaches in the literature.

The conclusions and considerations regarding PLSH and ePLSH are the following: (i) they support for high dimensional feature vectors, allowing different complementary feature descriptors to be employed to increase the robustness of the face indexing; (ii) they are easy to implement and deploy in practice since the only parameters needed to be set are the number of hash models and subjects in the candidate list. (iii) they do not provide good performances when the number of samples per subject is reduced and (iv) incremental enrollment of subjects in the framework requires re-training of the hash models, which may be prohibitive to perform in practice, specially for ePLSH which demands considerable more hash models.

In future works, we may consider the incremental learning algorithm for PLS rather than NIPALS [55], which might solve the issue regarding the incremental enrollment of subjects. We also may consider learning PLSH hash models for different subsets of subjects in the gallery, which have already been extensively

studied to make PLS face identification scalable to incremental enrollment of subjects in the gallery [5]. In this way, it is possible, for instance, to distribute the processing among numerous nodes in a computer cluster, which should be necessary to scale the approach for millions of subjects. The performance drop of PLSH and ePLSH when there are few samples per subject in the face gallery might be alleviated by generating synthetic samples using face morphing methods, which has already been considered for PLS face identification to leverage the recognition rates [4].

Acknowledgments

The authors would like to thank the Brazilian National Research Council – CNPq (Grants #487529/2013-8 and #477457/2013-4), CAPES (Grant STIC-AMSUD 001/2013), and the Minas Gerais Research Foundation - FAPEMIG (Grants APQ-00567-14 and CEX – APQ-03195-13). This work was partially supported by the STIC-AmSud program.

References

[1] R. Chellappa, P. Sinha, P. J. Phillips, Face recognition by computers and humans, Computer 43 (2) (2010) 46–55.

[2] H. Wechsler, Reliable face recognition methods: system design, implementation and evaluation, Vol. 7, Springer Science & Business Media, 2009.

[3] H. Guo, W. R. Schwartz, L. S. Davis, Face verification using large feature sets and one shot similarity, in: Biometrics (IJCB), 2011 International Joint Conference on, IEEE, 2011, pp. 1–8.

[4] W. R. Schwartz, H. Guo, J. Choi, L. S. Davis, Face identification using large feature sets, Image Processing, IEEE Transactions on 21 (4) (2012) 2245–2255.

[5] G. C. Paulo, H. Pedrini, W. R. Schwartz, Classification schemes based on partial least squares for face identification, Journal of Visual Communication and Image Representation.

[6] C. E. Santos Jr, W. R. Schwartz, Extending face identification to open-set face recognition, in: Graphics, Patterns and Images (SIBGRAPI), 27th IEEE Conference on, 2014, pp. 188–195.

[7] C. E. Santos Jr, E. Kijak, G. Gravier, W. R. Schwartz, Learning to hash faces using large feature vectors, in: Content-Based Multimedia Indexing (CBMI), 13th IEEE International Workshop on, 2015, pp. 1–6.

[8] S. Z. Li, A. K. Jain, Handbook of Face Recognition, 2nd Edition, Springer Publishing Company, Incorporated, 2011.

[9] J. Wang, H. T. Shen, J. Song, J. Ji, Hashing for similarity search: A survey, arXiv preprint arXiv:1408.2927.

[10] Z. Xie, G. Liu, Z. Fang, Face recognition based on combination of human perception and local binary pattern, in: Intelligent Science and Intelligent Data Engineering, Springer, 2012, pp. 365–373.

[11] B. F. Klare, A. K. Jain, Heterogeneous face recognition using kernel prototype similarities, Pattern Analysis and Machine 35 (6) (2013) 1410–1422.

[12] W. Gu, C. Xiang, Y. Venkatesh, D. Huang, H. Lin, Facial expression recognition using radial encoding of local gabor features and classifier synthesis, Pattern Recognition 45 (1) (2012) 80–91.

[13] J. Oh, S.-I. Choi, C. Kim, J. Cho, C.-H. Choi, Selective generation of gabor features for fast face recognition on mobile devices, Pattern Recognition Letters 34 (13) (2013) 1540–1547.

[14] N.-S. Vu, H. M. Dee, A. Caplier, Face recognition using the POEM descriptor, Pattern Recognition. Elsevier Transactions on 45 (7) (2012) 2478–2488.

[15] S. Liao, A. K. Jain, S. Z. Li, Partial face recognition: Alignment-free approach, Pattern Analysis and Machine Intelligence 35 (5) (2013) 1193–1205.

[16] J.-K. Kämäräinen, A. Hadid, M. Pietikäinen, Local representation of facial features, in: Handbook of Face Recognition, Springer, 2011, pp. 79–108.

[17] S. H. Salah, H. Du, N. Al-Jawad, Fusing local binary patterns with wavelet features for ethnicity identification, Signal Image Process 21 (5) (2013) 416–422.

[18] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International journal of computer vision 60 (2) (2004) 91–110.

[19] M. F. Valstar, M. Pantic, Fully automatic recognition of the temporal phases of facial actions, Systems, Man, and Cybernetics. IEEE Transactions on 42 (1) (2012) 28–43.

[20] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, Y. Ma, Robust face recognition via sparse representation, Pattern Analysis and Machine Intelligence, IEEE Transactions on 31 (2) (2009) 210–227.

[21] W. Deng, J. Hu, J. Guo, In defense of sparsity based face recognition, in: Computer Vision and Pattern Recognition (CVPR). IEEE Conference on, 2013, pp. 399–406.

[22] O. Barkan, J. Weill, L. Wolf, H. Aronowitz, Fast high dimensional vector multiplication face recognition, in: Computer Vision (ICCV). IEEE International Conference on, 2013, pp. 1960–1967.

[23] W. Deng, J. Hu, J. Guo, Extended src: Undersampled face recognition via intraclass variant dictionary, Pattern Analysis and Machine Intelligence. IEEE Transactions on 34 (9) (2012) 1864–1870.

[24] B. He, D. Xu, R. Nian, M. van Heeswijk, Q. Yu, Y. Miche, A. Lendasse, Fast face recognition via sparse coding and extreme learning machine, Cognitive Computation 6 (2) (2014) 264–277.

[25] Q. Yuan, A. Thangali, S. Sclaroff, Face identification by a cascade of rejection classifiers, in: Computer Vision and Pattern Recognition (CVPR) - Workshops. IEEE Conference on, 2005, pp. 152–152.

[26] F. Shen, C. Shen, A. van den Hengel, Z. Tang, Approximate least trimmed sum of squares fitting and applications in image analysis, Image Processing, IEEE Transactions on 22 (5) (2013) 1836–1847.

[27] F. Shen, C. Shen, R. Hill, A. van den Hengel, Z. Tang, Fast approximate $l\infty$ minimization: Speeding up robust regression, Computational Statistics & Data Analysis 77 (2014) 25–37.

[28] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, C. Schmid, Aggregating local image descriptors into compact codes, Pattern Analysis and Machine Intelligence, IEEE Transactions on 34 (9) (2012) 1704–1716.

[29] Y. Gong, S. Lazebnik, A. Gordo, F. Perronnin, Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval, Pattern Analysis and Machine Intelligence, IEEE Transactions on 35 (12) (2013) 2916–2929.

[30] F. Shen, C. Shen, Q. Shi, A. van den Hengel, Z. Tang, H. T. Shen, Hashing on nonlinear manifolds, Image Processing, IEEE Transactions on 24 (6) (2015) 1839–1851.

[31] F. Shen, C. Shen, W. Liu, H. Tao Shen, Supervised discrete hashing, in: Computer vision and pattern recognition (CVPR). IEEE Conference on, 2015, pp. 37–45.

[32] M. Datar, N. Immorlica, P. Indyk, V. S. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, in: Computational Geometry. ACM 20th Symposium on, 2004, pp. 253–262.

[33] O. Chum, J. Philbin, A. Zisserman, et al., Near duplicate image detection: min-hash and tf-idf weighting., in: British Machine Conference (BMVC). Proceedings of, Vol. 810, 2008, pp. 812–815.

[34] A. Joly, C. Frélicot, O. Buisson, Feature statistical retrieval applied to content based copy identification, in: Image Processing (ICIP). IEEE International Conference on, Vol. 1, 2004, pp. 681–684.

[35] S. Poullot, O. Buisson, M. Crucianu, Z-grid-based probabilistic retrieval for scaling up content-based copy detection, in: Image and Video Retrieval. 6th ACM International Conference on, 2007, pp. 348–355.

[36] B. Kulis, P. Jain, K. Grauman, Fast similarity search for learned metrics, Pattern Analysis and Machine Intelligence, IEEE Transactions on 31 (12) (2009) 2143–2157.

[37] Y. Weiss, A. Torralba, R. Fergus, Spectral hashing, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems 21, Curran Associates, Inc., 2009, pp. 1753–1760.

[38] R. Salakhutdinov, A. Mnih, G. Hinton, Restricted Boltzmann machines for collaborative filtering, in: Machine learning. 24th International Conference on, ACM, 2007, pp. 791–798.

[39] A. Joly, O. Buisson, Random maximum margin hashing, in: Computer Vision and Pattern Recognition (CVPR). IEEE Conference on, 2011, pp. 873–880.

[40] A. Torralba, R. Fergus, Y. Weiss, Small codes and large image databases for recognition, in: Computer Vision and Pattern Recognition (CVPR). IEEE Conference on, 2008, pp. 1–8.

[41] A. Andoni, P. Indyk, Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, in: Foundations of Computer Science. 47th IEEE Symposium on, 2006, pp. 459–468.

[42] A. Z. Broder, On the resemblance and containment of documents, in: Compression and Complexity of Sequences. IEEE Proceedings on, 1997, pp. 21–29.

[43] P. Indyk, R. Motwani, Approximate nearest neighbors: towards removing the curse of dimensionality, in: Proceedings of the thirtieth annual ACM symposium on Theory of computing, 1998, pp. 604–613.

[44] R. Rosipal, N. Krämer, Overview and recent advances in partial least squares, in: Subspace, latent structure and feature selection, Springer, 2006, pp. 34–51.

[45] H. Wold, Partial least squares, in: S. Kotz, N. Johnson (Eds.), Encyclopedia of Statistical Science, New York: Wiley, 1985, pp. 581–591.

[46] B. V. Srinivasan, W. R. Schwartz, R. Duraiswami, L. Davis, Partial least squares on graphical processor for efficient pattern recognition, Tech. rep., University of Maryland. Computer Science Department. CS-TR-4968 (2010).

[47] T. Mehmood, K. H. Liland, L. Snipen, S. Sæbø, A review of variable selection methods in partial least squares regression, Chemometrics and Intelligent Laboratory Systems 118 (2012) 62–69.

[48] P. J. Phillips, H. Moon, S. A. Rizvi, P. J. Rauss, The FERET evaluation methodology for face-recognition algorithms, Pattern Analysis and Machine Intelligence 22 (10) (2000) 1090–1104.

[49] H. Wang, S. Z. Li, Y. Wang, Face recognition under varying lighting conditions using self quotient image, in: Automatic Face and Gesture Recognition, 2004, pp. 819–824.

[50] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min, W. Worek, Overview of the face recognition grand challenge, in: Computer vision and pattern recognition (CVPR). IEEE Conference on, Vol. 1, 2005, pp. 947–954.

[51] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary patterns: Application to face recognition, Pattern Analysis and Machine Intelligence, IEEE Transactions on 28 (12) (2006) 2037–2041.

[52] T. Randen, J. H. Husoy, Filtering for texture classification: A comparative study, Pattern Analysis and Machine Intelligence 21 (4) (1999) 291–310.

[53] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society Conference on, Vol. 1, 2005, pp. 886–893.

[54] S. De Jong, Simpls: an alternative approach to partial least squares regression, Chemometrics and intelligent laboratory systems 18 (3) (1993) 251–263.

[55] X.-Q. Zeng, G.-Z. Li, Incremental partial least squares analysis of big streaming data, Pattern Recognition 47 (11) (2014) 3726–3735.