



HAL
open science

Partial Least Squares for Face Hashing

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

► **To cite this version:**

Cassio E. dos Santos, Ewa Kijak, William Robson Schwartz, Guillaume Gravier. Partial Least Squares for Face Hashing. *Neurocomputing*, 2016, 213, pp.34-47. 10.1016/j.neucom.2016.02.083 . hal-01399660

HAL Id: hal-01399660

<https://hal.science/hal-01399660v1>

Submitted on 20 Nov 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

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.

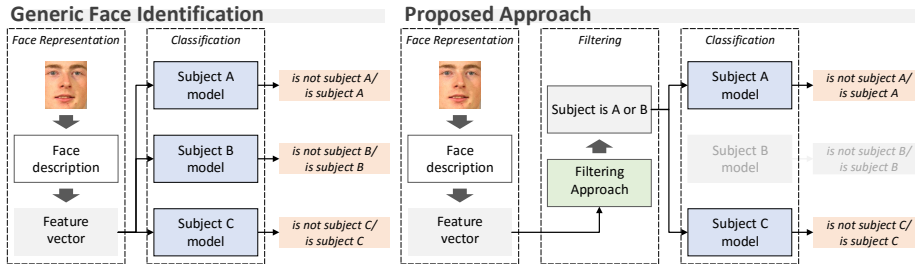


Figure 1: Common face identification pipeline and the proposed pipeline with the filtering approach which is used to reduce the number of evaluations in the classification step with low computational cost. The filtering approach is the main contribution in this work and it is tailored considering recent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

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

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

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

52 The proposed approach is inspired by the family of methods regarded as
53 *locality-sensitive hashing* (LSH), which are the most popular large-scale image
54 retrieval method in the literature, and the *partial least squares* (PLS), which has
55 been explored intensively in numerous past works regarding face recognition [3,
56 4, 5, 6]. We call the proposed approach PLS *for hashing*, abbreviated to PLSH
57 and ePLSH in its extension.

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

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

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

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

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

94 2. Related work

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

100 2.1. Face identification

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

106 Feature descriptors provide a robust manner to represent face images in-
107 variant to misalignment, illumination and pose of the face. Regarding feature

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

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

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

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

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

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

149 2.1.1. Fast face identification

150 Fast face identification is not a largely explored research topic and there
151 are few works in the literature about it [22, 23, 24, 25, 4]. In [22], compact
152 descriptors based on local binary patterns are used to compare quickly the
153 candidates in the face gallery. In [23] and [24], a fast optimization algorithm is
154 considered for SRC to reduce the computational cost when calculating the linear
155 combination between the test and the samples in the dictionary. Similar to [26],
156 where least trimmed squares (LTS) is considered to cope with noise in SRC-
157 based face identification, Shen et. al. [27] propose an approximation of the least
158 median squares (LMS), which provides speedup of some order of magnitude
159 in the SRC approach while still dealing with noise in the gallery samples.

160 Although the aforementioned methods provide significant improvement in the
161 test-subject comparison, poor performance is observed when there are numerous
162 subjects in the face gallery since these approaches still present linear asymptotic
163 complexity with the gallery size.

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

169 number of subjects tested in the face gallery, resulting in a reduced computa-
170 tional complexity considering the number of subjects in the face gallery when
171 compared to the brute-force approach.

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

179 2.2. Large-scale image retrieval

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

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

197 *2.2.1. Locality-sensitive hashing*

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

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

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

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

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

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

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

248 3. Methodology

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

254 *3.1. Partial least squares regression*

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

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

$$\begin{aligned} X_{n \times d} &= T_{n \times p} P_{d \times p}^T + E_{n \times d}, \\ Y_{n \times 1} &= U_{n \times p} Q_{p \times 1} + F_{n \times 1}, \end{aligned} \quad (1)$$

267 where $T_{n \times p}$ and $U_{n \times p}$ denote latent variables from feature vectors and response
 268 values, respectively. The matrix $P_{d \times p}$ and the vector Q_p represent loadings
 269 and the matrix E and the vector F are residuals from the transformation.
 270 PLS algorithms compute P and Q such that the covariance between U and
 271 T is maximum [44]. We consider the nonlinear iterative PLS (NIPALS) algo-
 272 rithm [45] which calculates the maximum covariance between the latent variables
 273 $T = \{t_1, \dots, t_p\}$ and $U = \{u_1, \dots, u_p\}$ using the matrix $W_{d \times p} = \{w_1, \dots, w_p\}$, such
 274 that

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

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

$$\beta = W(P^T W)^{-1}(T^T T)^{-1} T^T Y. \quad (2)$$

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

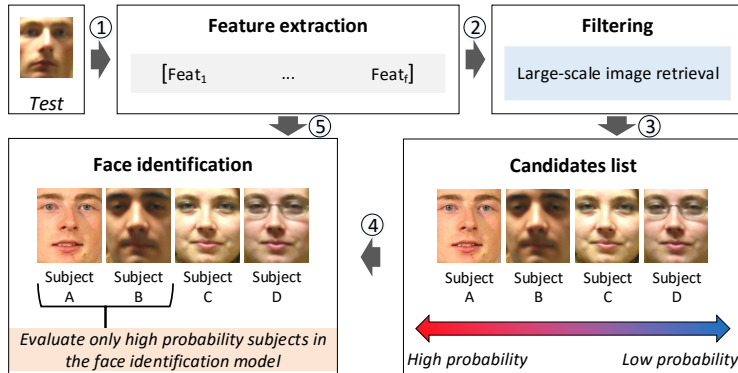


Figure 2: Overview of the filtering and the face identification pipeline. (1) Different feature descriptors are extracted from the test image and concatenated resulting in a large feature vector more robust to image effects than single feature descriptors. (2) The feature vector is presented to the filtering 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.

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

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

284 3.2. Face identification based on partial least squares

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

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

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

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

316 3.3. Partial least squares for face hashing (PLSH)

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

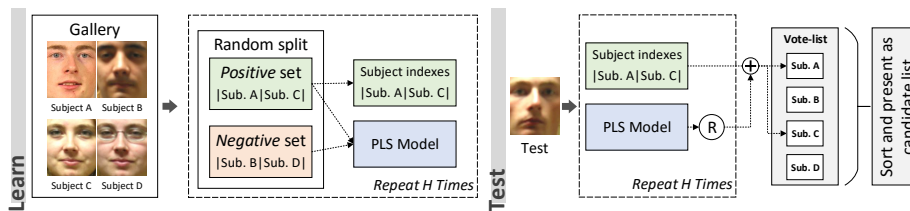


Figure 3: Overview of PLS for face hashing (PLSH) with (left) train and (right) test steps. In the train, a PLS regression model is learned to discriminate between two balanced random subsets of subjects in the face gallery (positive and negative subsets). In the test, the test sample is presented to 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.

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

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

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

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

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

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

356 3.3.1. Consistency

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

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

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

375 3.3.2. Hamming embedding

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

381 To demonstrate the aforementioned claims, consider that PLS responses can
382 be only $+1$ or -1 , such that any test sample can be represented by the sequence
383 $X = \{+1, -1\}^H$, where H denotes the number of PLSH hash models. Consider
384 also that each subject s in the face gallery is represented by the binary string
385 $Y_s = \{1, 0\}^H$, where $y_i \in Y_s$ is set to 1 if the subject s was associated to the
386 positive subset of the i -th PLSH hash model in the train step, or 0, otherwise.
387 In this context, the weight w_s given by PLSH to each subject in the gallery is
388 calculated as

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

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

395 Hamming distance can be calculate using w_s as

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

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

400 3.3.3. Computational requirements

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

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

419 3.3.4. Alternative implementations

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

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

429 Another possible implementations of PLSH that does not modify much the
430 results is the product of votes instead of the sum, which is akin to the intersec-
431 tion of subsets among all hash functions. It is also possible to employ multiple
432 partitions instead of only two by using a categorical rather than Bernoulli dis-
433 tribution. However, multiple partitions present no significant difference in the
434 results and they require twice the space requirement since the indexes of sub-
435 jects that were learned with +1 target response in the PLS model need to be
436 stored to allow them to receive the votes in the test.

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

443 3.4. Feature selection for face hashing (*ePLSH*)

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

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

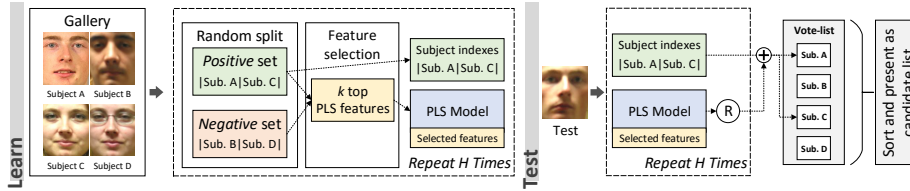


Figure 4: Overview of PLS for face hashing and feature selection (ePLSH) with (left) train and (right) test steps. The train consists of the same procedures employed in PLSH with the difference of the feature selection method based on the top discriminative features between the positive and negative 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.

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

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

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

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

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

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

494 3.4.1. Loading weights

495 The idea in the loading weight approach is, for each PLS component, to
496 select the features associated with higher absolute w_i value (alternately features
497 above a threshold [47]). Recall W being the output of NIPALS algorithm¹ used
498 to calculate latent variables. In this way, the absolute coefficient $w_{f,i} \in W$, for
499 the f -th PLS component and the i -th feature, is directly associated to the f -th

¹see Section 3.1 for the PLS description

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

506 3.4.2. Variable importance on projection

507 Variable importance on projection (VIP) consists in calculating a measure
 508 that summarize the loading weights (W) of all factors for each dimension in the
 509 feature vector. VIP measure is calculated as²

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

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

519 3.4.3. Regression coefficients

520 Regression coefficients for feature selection is the simplest of the three
 521 filter methods and consists in using the regression vector directly to select
 522 the most relevant features. Recall from Section 3.1 the regression vector as
 523 $\beta = W(P^T W)^{-1}(T^T T)^{-1}T^T Y$, where P and T are loading matrices from fea-
 524 tures and target values, respectively. Similar to the loading weights approaches,

²see Section 3.1 for variable definitions.

525 regression coefficients are also related to predictive capacity of the latent vari-
526 ables to estimate target values, however, in a more transparent manner since
527 they are directly employed to estimate the regression values. The main differ-
528 ence from the regression coefficient and the aforementioned filter approaches is
529 the correlation of the latent variables with target values embedded in the PLS
530 regression vector, which provides a small improvement over the loading weights
531 and VIP results.

532 3.5. *Early-stop search heuristic*

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

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

547 4. **Experimental results**

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

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

557 4.1. Experimental setup

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

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

569 4.1.1. FERET dataset

570 The facial recognition technology (FERET) dataset [48] consists of 1,196
571 images, one per subject for training, and four test sets designed to evaluate the
572 effects of lightning conditions, facial expression and aging on face identification
573 methods. The test sets are: *fb*, consisting of 1,195 images taken with different
574 facial expressions; *fc*, consisting of 194 images taken in different lightning con-
575 ditions; *dup1*, consisting of 722 images taken between 1 minute and 1,031 days
576 after the gallery image; *dup2*, is a subset of *dup1* and consists of 234 images
577 taken 18 months after the gallery image. In our experiments, all images were
578 cropped in the face region using annotated coordinates of the face, scaled to
579 128×128 pixels and normalized using the self-quotient image (SQI) method to
580 remove lightning effects [49].

581 *4.1.2. FRGC dataset*

582 The face recognition grand challenge dataset (FRGC) [50] consists of 275
583 subjects and samples that include 3D models of the face and 2D images taken
584 with different illumination conditions and facial expressions. We follow the
585 same protocol described by Yuan et al. [25], which considers only 2D images
586 and consists in randomly selecting different percentages of samples from each
587 subject to compose the face gallery and using the remaining samples to test.
588 The process is repeated five times and the mean and standard deviation of
589 the rank-1 recognition rate and speedup (considering the brute-force approach)
590 are reported. The samples were cropped in the facial region, resulting in size
591 138×160 pixels, and scaled to 128×128 pixels.

592 *4.1.3. Evaluation metric (MARR)*

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

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

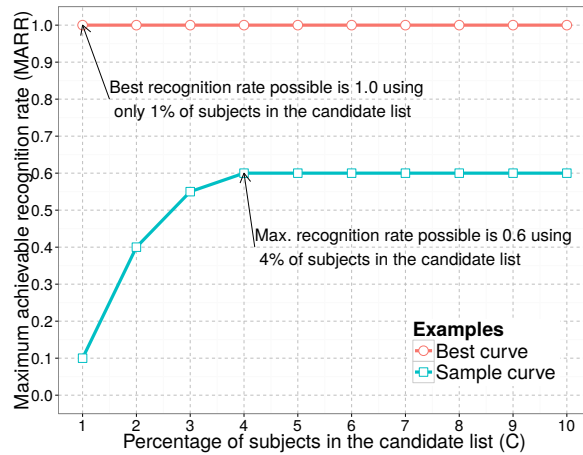


Figure 5: An example of the plots regarding the MARR evaluation metric for two sample curves. The MARR metric (vertical axis) considers that the candidate list is presented to an ideal face identification method, therefore, providing the upper bound of the recognition rate achievable when considering 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.

608 *4.1.4. Number of dimensions in the PLS models*

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

616 *4.1.5. Feature descriptors*

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

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

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

630 To compute Gabor filters, we convolve the face image with filters of size
631 16×16 pixels, 8 scales, equally distributed between $[0, \frac{\pi}{2}]$, and 5 orientations,
632 equally distributed between $[0, \pi]$, which results in 40 convolved images. The
633 images were downsampled by a factor of 4 and concatenated to assemble the final
634 feature descriptor, resulting in 40,960 dimensions and taking 1,475 milliseconds
635 to calculate per face image, on average.

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

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

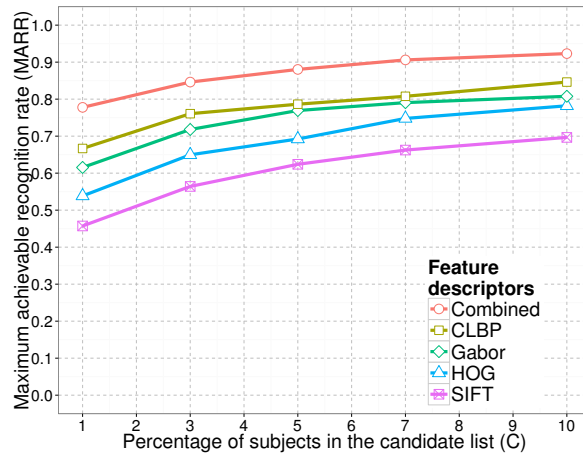


Figure 6: Evaluation of the CLBP, Gabor, HOG and SIFT feature descriptors and their combination.

649 *4.2. PLSH parameters validation*

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

654 *4.2.1. Combination of feature descriptors*

655 Figure 6 presents the MARR curves for each of the four feature descriptors
 656 considered in this work and their combination. The number of hash models in
 657 this experiment was empirically set to 150. According to Figure 6, the combi-
 658 nation of CLBP, Gabor, HOG and SIFT is responsible for an increase of about
 659 10 percentage points (p.p.) in MARR compared to the best individual feature
 660 descriptor (CLBP). Therefore, we employ the combination of these feature de-
 661 scriptors in the remaining experiments. The combined feature descriptor has
 662 120,059 dimensions with averaged time to calculate equal to 1.7 seconds. It is
 663 important to point out that the time spent to calculate the feature descriptors
 664 for a probe sample is constant (it does not depend on the number of subjects

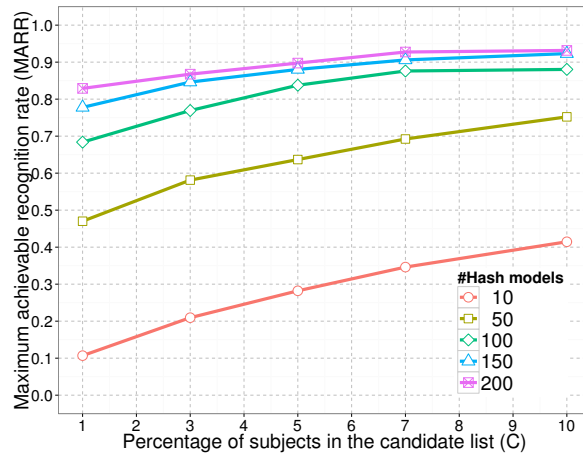


Figure 7: Number of hash models as a function of the MARR for different percentages of subjects in the candidate list.

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

669 4.2.2. Number of hash models

670 Figure 7 presents MARR for a number of hash models equal to
 671 10, 50, 100, 150 and 200. According to the results, a large improvement in MARR
 672 (for any number of subjects in the candidate list) takes place when the number
 673 of hash models increases from 10 to 150 can be seen in Figure 7. However, the
 674 increase in MARR is negligible when the number of hash models is raised from
 675 150 to 200. Since the face identification and the PLSH approaches depend on
 676 a single dot product between the feature and the PLS regression vectors, the
 677 computational cost to evaluate each hash function in PLSH is about the same
 678 as the cost to evaluate each subject in the gallery. Therefore, to obtain a low
 679 computational cost for testing samples, we consider 150 hash functions in the
 680 remaining PLSH experiments. As a reference, the average time to evaluate each
 681 hash function in this experiment was 426 microseconds.

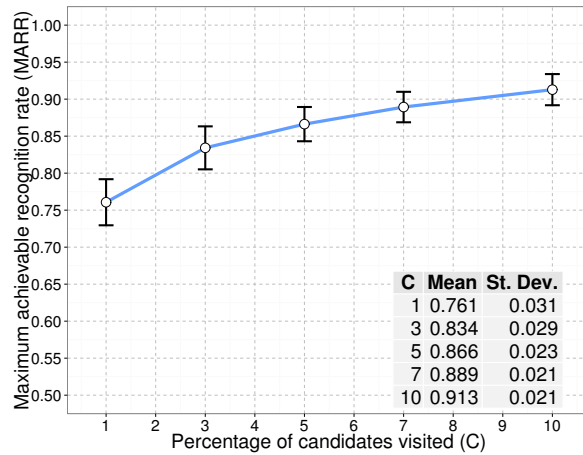


Figure 8: Average MARR and standard deviation for 10 PLSH runs considering 1% of subjects in the candidate list.

682 *4.2.3. Stability of the results*

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

692 *4.3. ePLSH parameters validation*

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

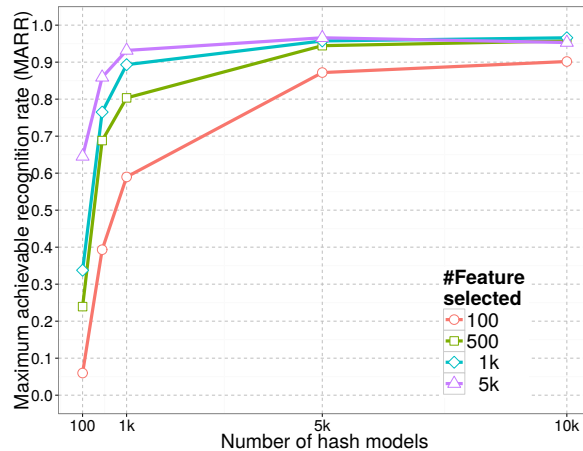


Figure 9: MARR with different numbers of hash models and the feature selected in the ePLSH.

697 *4.3.1. Number of hash models and features selected*

698 Figure 9 presents MARR for 1% of subjects in the candidate list for different
 699 numbers of hash models (m) and selected features (d). The ePLSH aims at
 700 reducing the computational cost to evaluate PLSH hash functions, which can
 701 be roughly approximated to a number of multiplication operations equal to
 702 $m \times d$. It is important to point out that d equal to 500 provides nearly the same
 703 MARR for a sufficient large enough m . Therefore, we fix d to 500 and vary m
 704 for different datasets and number of subjects in the face gallery.

705 We achieve minimum computational cost with almost maximum MARR us-
 706 ing 5,000 hash models and with 2.5 million multiplications. Note that this
 707 number of multiplications refers only to the ePLSH approach such that the
 708 total computational cost of the pipeline also includes the number of multiplica-
 709 tions in the face identification. As a comparison, the number of multiplications
 710 necessary in the brute-force approach for the 1,196 subjects in the gallery is
 711 $1,196 \times 120,059 = 143.5$ millions, which is about 57 times more than the number
 712 of multiplications necessary to calculate all of the 5,000 ePLSH hash functions.

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

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

723 The average time to calculate each PLSH hash function is 446 microseconds
724 compared to 12 microseconds for each ePLSH hash function. However, since
725 a considerable number of hash functions is employed in ePLSH compared to
726 PLSH, the time to train ePLSH is significant higher than PLSH. The time
727 spent to train all the 5,000 hash functions in ePLSH is 14 hours compared to
728 22 minutes for the 150 hash functions in PLSH, which may not impose an issue
729 because the train is performed offline and only once for a fixed face gallery.
730 The train can also be accelerated considering other PLS algorithms such as
731 SIMPLS [54] rather than NIPALS.

732 4.3.2. Feature selection

733 In this section, we compare the feature selection approaches described in Sec-
734 tion 3.4. We also compare whether we should retrain the PLS model in the re-
735 gression coefficients approach to redistribute weights from the discarded feature
736 among the selected features. According to the results presented in Figure 10,
737 there is no significant difference in the feature selection approaches evaluated.
738 Considering the experiments in Section 4.3.4, the MARR at 1% of subjects in
739 the candidate list for the regression coefficients approach vary roughly between
740 0.92 and 0.96. In this case, it can be concluded that the regression coefficients
741 approach is better than the loading weights and VIP. Furthermore, there is
742 no significant difference between retraining or not the hash model regression
743 coefficients after the feature selection step.

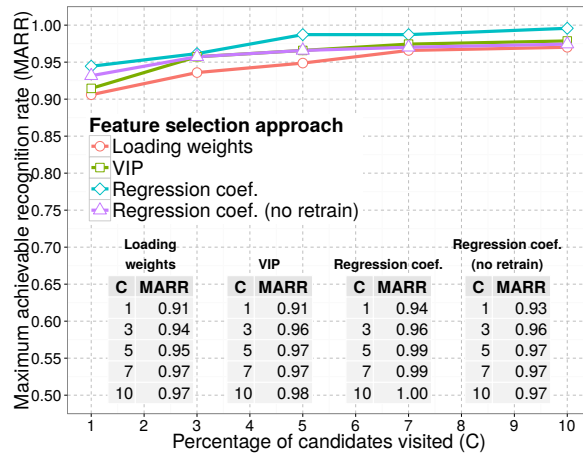


Figure 10: Evaluation of loading weights, variable importance on projection (VIP) and regression coefficients for feature selection.

744 4.3.3. Number of hash models and gallery size

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

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

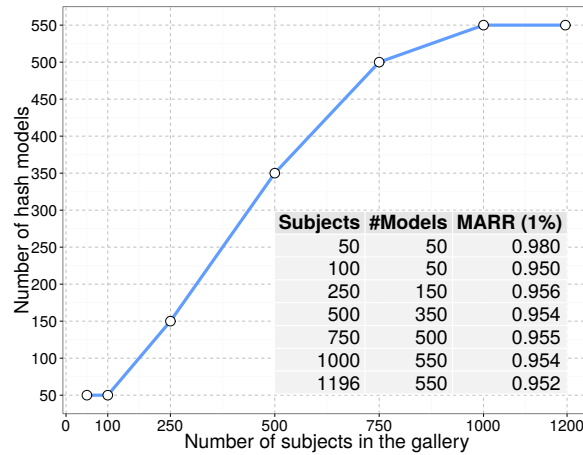


Figure 11: Number of hash models necessary to provide at least 0.95 MARR with different gallery sizes and 1% of subjects in the candidate list.

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

772 4.3.4. Stability of the results

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

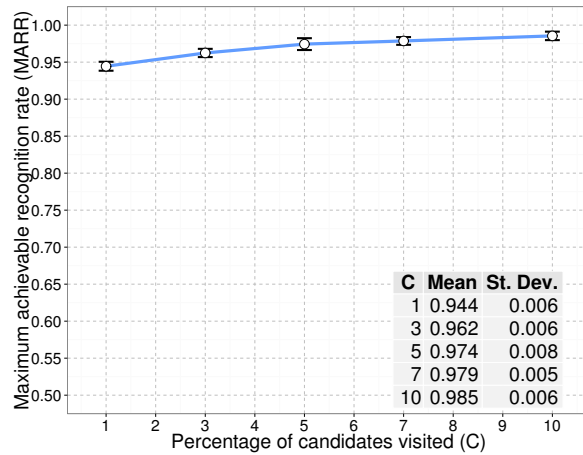


Figure 12: Average MARR and standard deviation for 10 ePLSH runs considering 1% of subjects in the candidate list.

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

783 *4.4. Results on the FERET dataset*

784 Results regarding MARR and rank-1 recognition rate for PLSH in all test
 785 sets from the FERET dataset are presented in Figures 13a and 13b. For the
 786 test sets *fb* and *fc*, about 1% of subjects in the candidates list is enough to
 787 achieve more than 95% of the rank-1 recognition rate of the brute-force approach
 788 (presented in the legend of Figure 13b for each test set). However, for the test
 789 sets *dup1* and *dup2*, about 5% of subjects in the candidate list ensured at least
 790 95% of the brute-force rank-1 recognition rate. The theoretical speedup in the
 791 worst case can be calculated considering the 150 PLSH hash function evaluations
 792 and the 5% of the gallery size, which consists of 60 PLS projections. In this
 793 case, if the early-stop search heuristic is not considered, i.e., all subjects in the
 794 candidate list are evaluated for each test sample, the number of PLS projections

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

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

805 4.5. Results on the FRGC dataset

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

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

824 According to Table 1, it is possible to conclude that the rank-1 recogni-
825 tion rate is reasonably stable, with variance in the first decimal place, which is
826 similar to the results regarding stability presented for PLSH and ePLSH. The
827 speedup for PLSH and ePLSH decreases considerable as the number of samples
828 per subject available for train reduce. The reason for that is the increase in
829 the number of hash models and the maximum number of subjects visited in the
830 candidate list to guarantee at least 0.95 rank-1 recognition rate. Even with re-
831 duced speedups considering 35% of samples available for train, ePLSH provides
832 significant improvement over the speedup achieved by the tree-based approach
833 (3.6 times faster), while PLSH provides competitive speedup.

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

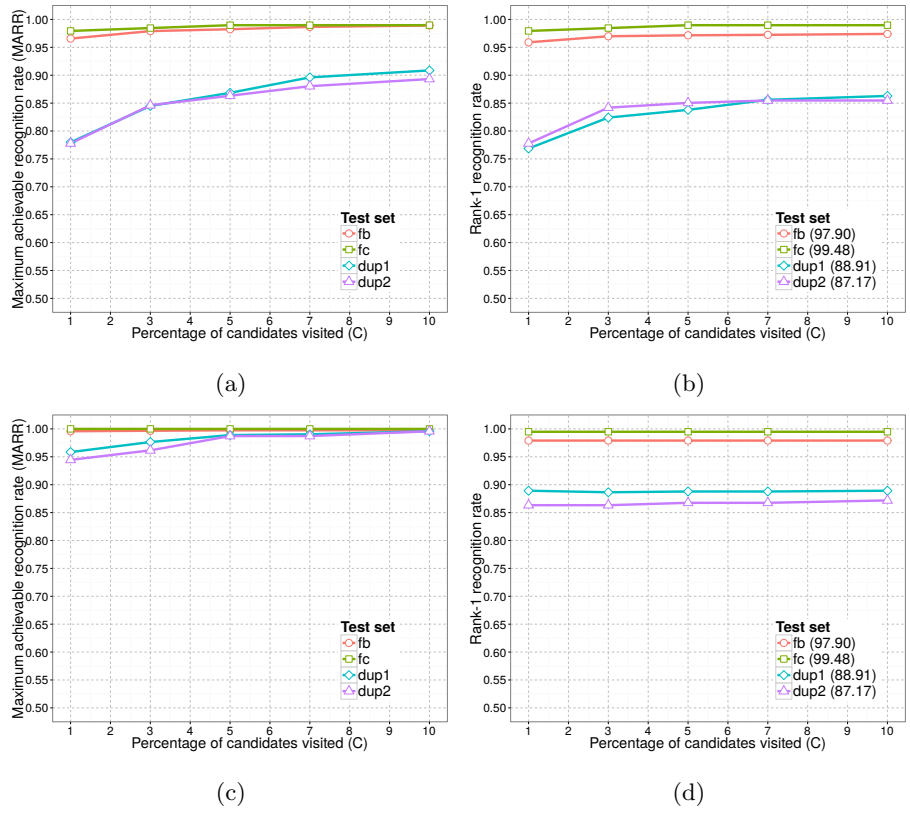


Figure 13: Results on the FERET dataset. (a) PLSH MARR curves, (b) PLSH rank-1 recognition rate, (c) ePLSH MARR curves and (d) ePLSH rank-1 recognition rate. Number in parenthesis indicate rank-1 recognition rate for the brute force approach.

		% 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	$(233.61 \pm 37.05) \times$	$(98.93 \pm 8.39) \times$	$(45.42 \pm 3.84) \times$	$(22.29 \pm 1.03) \times$	$(14.21 \pm 1.74) \times$
	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).

840 5. Conclusions and future works

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

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

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

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

878 **Acknowledgments**

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

884 **References**

- 885 [1] R. Chellappa, P. Sinha, P. J. Phillips, Face recognition by computers and
886 humans, *Computer* 43 (2) (2010) 46–55.
- 887 [2] H. Wechsler, *Reliable face recognition methods: system design, implemen-
888 tation and evaluation*, Vol. 7, Springer Science & Business Media, 2009.
- 889 [3] H. Guo, W. R. Schwartz, L. S. Davis, Face verification using large feature
890 sets and one shot similarity, in: *Biometrics (IJCB)*, 2011 International Joint
891 Conference on, IEEE, 2011, pp. 1–8.
- 892 [4] W. R. Schwartz, H. Guo, J. Choi, L. S. Davis, Face identification using
893 large feature sets, *Image Processing, IEEE Transactions on* 21 (4) (2012)
894 2245–2255.

- 895 [5] G. C. Paulo, H. Pedrini, W. R. Schwartz, Classification schemes based on
896 partial least squares for face identification, *Journal of Visual Communica-*
897 *tion and Image Representation*.
- 898 [6] C. E. Santos Jr, W. R. Schwartz, Extending face identification to open-
899 set face recognition, in: *Graphics, Patterns and Images (SIBGRAPI)*, 27th
900 *IEEE Conference on*, 2014, pp. 188–195.
- 901 [7] C. E. Santos Jr, E. Kijak, G. Gravier, W. R. Schwartz, Learning to hash
902 faces using large feature vectors, in: *Content-Based Multimedia Indexing*
903 *(CBMI)*, 13th *IEEE International Workshop on*, 2015, pp. 1–6.
- 904 [8] S. Z. Li, A. K. Jain, *Handbook of Face Recognition*, 2nd Edition, Springer
905 *Publishing Company, Incorporated*, 2011.
- 906 [9] J. Wang, H. T. Shen, J. Song, J. Ji, Hashing for similarity search: A survey,
907 *arXiv preprint arXiv:1408.2927*.
- 908 [10] Z. Xie, G. Liu, Z. Fang, Face recognition based on combination of human
909 perception and local binary pattern, in: *Intelligent Science and Intelligent*
910 *Data Engineering*, Springer, 2012, pp. 365–373.
- 911 [11] B. F. Klare, A. K. Jain, Heterogeneous face recognition using kernel proto-
912 type similarities, *Pattern Analysis and Machine* 35 (6) (2013) 1410–1422.
- 913 [12] W. Gu, C. Xiang, Y. Venkatesh, D. Huang, H. Lin, Facial expression recog-
914 nition using radial encoding of local gabor features and classifier synthesis,
915 *Pattern Recognition* 45 (1) (2012) 80–91.
- 916 [13] J. Oh, S.-I. Choi, C. Kim, J. Cho, C.-H. Choi, Selective generation of gabor
917 features for fast face recognition on mobile devices, *Pattern Recognition*
918 *Letters* 34 (13) (2013) 1540–1547.
- 919 [14] N.-S. Vu, H. M. Dee, A. Caplier, Face recognition using the POEM descrip-
920 tor, *Pattern Recognition. Elsevier Transactions on* 45 (7) (2012) 2478–2488.

- 921 [15] S. Liao, A. K. Jain, S. Z. Li, Partial face recognition: Alignment-free ap-
922 proach, *Pattern Analysis and Machine Intelligence* 35 (5) (2013) 1193–1205.
- 923 [16] J.-K. Kämäräinen, A. Hadid, M. Pietikäinen, Local representation of facial
924 features, in: *Handbook of Face Recognition*, Springer, 2011, pp. 79–108.
- 925 [17] S. H. Salah, H. Du, N. Al-Jawad, Fusing local binary patterns with wavelet
926 features for ethnicity identification, *Signal Image Process* 21 (5) (2013)
927 416–422.
- 928 [18] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *Inter-
929 national journal of computer vision* 60 (2) (2004) 91–110.
- 930 [19] M. F. Valstar, M. Pantic, Fully automatic recognition of the temporal
931 phases of facial actions, *Systems, Man, and Cybernetics. IEEE Transac-
932 tions on* 42 (1) (2012) 28–43.
- 933 [20] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, Y. Ma, Robust face recogni-
934 tion via sparse representation, *Pattern Analysis and Machine Intelligence,
935 IEEE Transactions on* 31 (2) (2009) 210–227.
- 936 [21] W. Deng, J. Hu, J. Guo, In defense of sparsity based face recognition, in:
937 *Computer Vision and Pattern Recognition (CVPR). IEEE Conference on,*
938 2013, pp. 399–406.
- 939 [22] O. Barkan, J. Weill, L. Wolf, H. Aronowitz, Fast high dimensional vector
940 multiplication face recognition, in: *Computer Vision (ICCV). IEEE Inter-
941 national Conference on,* 2013, pp. 1960–1967.
- 942 [23] W. Deng, J. Hu, J. Guo, Extended src: Undersampled face recognition via
943 intraclass variant dictionary, *Pattern Analysis and Machine Intelligence.
944 IEEE Transactions on* 34 (9) (2012) 1864–1870.
- 945 [24] B. He, D. Xu, R. Nian, M. van Heeswijk, Q. Yu, Y. Miche, A. Lendasse, Fast
946 face recognition via sparse coding and extreme learning machine, *Cognitive
947 Computation* 6 (2) (2014) 264–277.

- 948 [25] Q. Yuan, A. Thangali, S. Sclaroff, Face identification by a cascade of re-
949 jection classifiers, in: Computer Vision and Pattern Recognition (CVPR)
950 - Workshops. IEEE Conference on, 2005, pp. 152–152.
- 951 [26] F. Shen, C. Shen, A. van den Hengel, Z. Tang, Approximate least trimmed
952 sum of squares fitting and applications in image analysis, Image Processing,
953 IEEE Transactions on 22 (5) (2013) 1836–1847.
- 954 [27] F. Shen, C. Shen, R. Hill, A. van den Hengel, Z. Tang, Fast approximate
955 l_∞ minimization: Speeding up robust regression, Computational Statistics
956 & Data Analysis 77 (2014) 25–37.
- 957 [28] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, C. Schmid, Aggre-
958 gating local image descriptors into compact codes, Pattern Analysis and
959 Machine Intelligence, IEEE Transactions on 34 (9) (2012) 1704–1716.
- 960 [29] Y. Gong, S. Lazebnik, A. Gordo, F. Perronnin, Iterative quantization: A
961 procrustean approach to learning binary codes for large-scale image re-
962 trieval, Pattern Analysis and Machine Intelligence, IEEE Transactions on
963 35 (12) (2013) 2916–2929.
- 964 [30] F. Shen, C. Shen, Q. Shi, A. van den Hengel, Z. Tang, H. T. Shen, Hashing
965 on nonlinear manifolds, Image Processing, IEEE Transactions on 24 (6)
966 (2015) 1839–1851.
- 967 [31] F. Shen, C. Shen, W. Liu, H. Tao Shen, Supervised discrete hashing, in:
968 Computer vision and pattern recognition (CVPR). IEEE Conference on,
969 2015, pp. 37–45.
- 970 [32] M. Datar, N. Immorlica, P. Indyk, V. S. Mirrokni, Locality-sensitive hash-
971 ing scheme based on p-stable distributions, in: Computational Geometry.
972 ACM 20th Symposium on, 2004, pp. 253–262.
- 973 [33] O. Chum, J. Philbin, A. Zisserman, et al., Near duplicate image detection:
974 min-hash and tf-idf weighting., in: British Machine Conference (BMVC).
975 Proceedings of, Vol. 810, 2008, pp. 812–815.

- 976 [34] A. Joly, C. Frélicot, O. Buisson, Feature statistical retrieval applied to
977 content based copy identification, in: Image Processing (ICIP). IEEE In-
978 ternational Conference on, Vol. 1, 2004, pp. 681–684.
- 979 [35] S. Poullot, O. Buisson, M. Crucianu, Z-grid-based probabilistic retrieval for
980 scaling up content-based copy detection, in: Image and Video Retrieval. 6th
981 ACM International Conference on, 2007, pp. 348–355.
- 982 [36] B. Kulis, P. Jain, K. Grauman, Fast similarity search for learned metrics,
983 Pattern Analysis and Machine Intelligence, IEEE Transactions on 31 (12)
984 (2009) 2143–2157.
- 985 [37] Y. Weiss, A. Torralba, R. Fergus, Spectral hashing, in: D. Koller, D. Schu-
986 urmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Pro-
987 cessing Systems 21, Curran Associates, Inc., 2009, pp. 1753–1760.
- 988 [38] R. Salakhutdinov, A. Mnih, G. Hinton, Restricted Boltzmann machines for
989 collaborative filtering, in: Machine learning. 24th International Conference
990 on, ACM, 2007, pp. 791–798.
- 991 [39] A. Joly, O. Buisson, Random maximum margin hashing, in: Computer
992 Vision and Pattern Recognition (CVPR). IEEE Conference on, 2011, pp.
993 873–880.
- 994 [40] A. Torralba, R. Fergus, Y. Weiss, Small codes and large image databases
995 for recognition, in: Computer Vision and Pattern Recognition (CVPR).
996 IEEE Conference on, 2008, pp. 1–8.
- 997 [41] A. Andoni, P. Indyk, Near-optimal hashing algorithms for approximate
998 nearest neighbor in high dimensions, in: Foundations of Computer Science.
999 47th IEEE Symposium on, 2006, pp. 459–468.
- 1000 [42] A. Z. Broder, On the resemblance and containment of documents, in: Com-
1001 pression and Complexity of Sequences. IEEE Proceedings on, 1997, pp.
1002 21–29.

- 1003 [43] P. Indyk, R. Motwani, Approximate nearest neighbors: towards removing
1004 the curse of dimensionality, in: Proceedings of the thirtieth annual ACM
1005 symposium on Theory of computing, 1998, pp. 604–613.
- 1006 [44] R. Rosipal, N. Krämer, Overview and recent advances in partial least
1007 squares, in: Subspace, latent structure and feature selection, Springer,
1008 2006, pp. 34–51.
- 1009 [45] H. Wold, Partial least squares, in: S. Kotz, N. Johnson (Eds.), Encyclope-
1010 dia of Statistical Science, New York: Wiley, 1985, pp. 581–591.
- 1011 [46] B. V. Srinivasan, W. R. Schwartz, R. Duraiswami, L. Davis, Partial
1012 least squares on graphical processor for efficient pattern recognition, Tech.
1013 rep., University of Maryland. Computer Science Department. CS-TR-4968
1014 (2010).
- 1015 [47] T. Mehmood, K. H. Liland, L. Snipen, S. Sæbø, A review of variable se-
1016 lection methods in partial least squares regression, *Chemometrics and In-*
1017 *telligent Laboratory Systems* 118 (2012) 62–69.
- 1018 [48] P. J. Phillips, H. Moon, S. A. Rizvi, P. J. Rauss, The FERET evaluation
1019 methodology for face-recognition algorithms, *Pattern Analysis and Machine*
1020 *Intelligence* 22 (10) (2000) 1090–1104.
- 1021 [49] H. Wang, S. Z. Li, Y. Wang, Face recognition under varying lighting con-
1022 ditions using self quotient image, in: *Automatic Face and Gesture Recog-*
1023 *nition*, 2004, pp. 819–824.
- 1024 [50] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoffman,
1025 J. Marques, J. Min, W. Worek, Overview of the face recognition grand
1026 challenge, in: *Computer vision and pattern recognition (CVPR)*. IEEE
1027 Conference on, Vol. 1, 2005, pp. 947–954.
- 1028 [51] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary
1029 patterns: Application to face recognition, *Pattern Analysis and Machine*
1030 *Intelligence*, *IEEE Transactions on* 28 (12) (2006) 2037–2041.

- 1031 [52] T. Randen, J. H. Husoy, Filtering for texture classification: A comparative
1032 study, *Pattern Analysis and Machine Intelligence* 21 (4) (1999) 291–310.
- 1033 [53] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection,
1034 in: *Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer
1035 Society Conference on, Vol. 1, 2005, pp. 886–893.
- 1036 [54] S. De Jong, Simpls: an alternative approach to partial least squares re-
1037 gression, *Chemometrics and intelligent laboratory systems* 18 (3) (1993)
1038 251–263.
- 1039 [55] X.-Q. Zeng, G.-Z. Li, Incremental partial least squares analysis of big
1040 streaming data, *Pattern Recognition* 47 (11) (2014) 3726–3735.