Volume 21 Issue 8 September 2006 ISSN 0923-5965
ELSEVIER
SIGNAL PROCESSING
IMAGE
COMMUNICATION
Theory, Techniques & Applications
A publication of the European Association to Signal Processing (EURASIP)
Available online at ScienceDirect www.sciencedirect.com
EURASIP

Available online at www.sciencedirect.com
ScienceDirect
Signal Processing: *Image Communication* 21 (2006) 688–703
SIGNAL PROCESSING: IMAGE COMMUNICATION
www.elsevier.com/locate/image

# Adaptive image replica detection based on support vector classifiers

Yannick Maret*, Frédéric Dufaux, Touradj Ebrahimi

*Ecole Polytechnique Fédérale de Lausanne, EPFL-STI-ITS-LTS1, Yannick MARET, ELD241, Station 11, Institut de Traitement des Signaux, CH-1015 Lausanne, Switzerland*

Received 8 November 2005; received in revised form 27 June 2006; accepted 12 July 2006

**Abstract**

This paper presents a system for image replica detection. The idea behind the proposed approach is to adapt a system for detecting the replica of a specific reference image. The system is then able to classify test images as replicas of the reference image or as unrelated images. More precisely, the test procedure is as follows. A set of features is extracted from a test image, representing texture, colour and grey-level characteristics. These features are then feed into a preprocessing step, which is fine-tuned to the reference image. Finally, the resulting features are entered to a support vector classifier that determines if the test image is a replica of the reference image. Experimental results show the effectiveness of the proposed system. Target applications include search for copyright infringement (e.g. variations of copyrighted images) and known illicit content (e.g. paedophile images known to the police).
© 2006 Elsevier B.V. All rights reserved.

*Keywords:* Image replica detection; Features extraction; Support vector machine; Dimensionality reduction; Copyright infringement detection

## 1. Introduction

In this paper, we propose a system to detect image replicas. By replica, we refer not only to a bit exact copy of a given reference image, but also to modified versions of the image after minor manipulations, malicious or not, as long as these manipulations do not change the perceptual meaning of the image content. In particular, replicas include all variants of the reference image obtained after common image processing manipulations such as compression, filtering, and adjustments of contrast, saturation or colours.

The proposed image replica detection system can be applied to *detect copyright infringement* by identifying variations of a given copyrighted image. Another application is to *discover known illicit content* such as child pornography images known to the police.

The problem of image replica detection is a particular subset of the more general problem of content-based search and retrieval of multimedia content. In recent years, multimedia search and retrieval have been the subject of extensive research

*Corresponding author. Tel.: +41 21 693 4621; fax: +41 21 693 7600.
*E-mail addresses:* yannick.maret@epfl.ch (Y. Maret), frederic.dufaux@epfl.ch (F. Dufaux), touradj.ebrahimi@epfl.ch (T. Ebrahimi).

0923-5965/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.image.2006.07.001

works and standardisation activities (MPEG-7 [16,29] and more recently JPSearch [17]). However, the specific problem of image replica detection has so far been the focus of fewer research efforts.

Two approaches to detect image replicas are *watermarking* [12] and *robust fingerprinting* [21,33,36]. Watermarking techniques [12] consist in embedding a signature in the reference image before dissemination. Replicas of the reference image can subsequently be detected by verifying the presence of the watermark. This class of techniques typically achieves high efficiency for the correct classification of replicas and non-replicas. However, it requires to modify the reference image, namely to embed a signature, prior to its distribution. Unfortunately, this is not always possible. For instance, the method is not applicable to already disseminated copyrighted content or in the case of illicit content. Robust fingerprinting techniques [21,33,36] analyse the reference image in order to extract a signature associated with the image content. Replicas are then identified whenever their signatures are close to that of the reference. This class of techniques is often based on a single feature, for example characteristic points of the Radon transform [21], log-mapping of the Radon transform [33], or intra-scale variances of wavelet coefficients [36]. While it is usually robust, computationally efficient, and suitable for fast database indexing and retrieval, it often performs poorly for the accurate classification of replicas and non-replicas.

More recently, techniques for image replica detection have been described in [18,30]. Ke et al. [18] propose a method based on the extraction of features, referred to as key points (KPs), which are stable in a scale-space representation. An image is typically represented by thousands of KPs. Test images are then classified as replicas or non-replicas using local sensitive hashing to match their KPs to those of the reference image. More specifically, no distance is directly computed, but it is rather the number of matching KPs which quantifies the similarity between two images. While this approach achieves very good performance for replica detection, it requires a computationally complex features extraction step. Qamra et al. [30] propose a different method based on the computation of a perceptual distance function (DPF). More precisely, a DPF is generated for each pair of reference and unknown image, to measure the similarity between the two. The main idea of the approach is to activate different features for different image pairs. Hence, only the most similar features are taken into account to compute the distance. While this method achieves good performance, it is inferior to [18].

In this paper, we introduce a new approach for image replica detection based on our earlier works [25–27]. The idea behind our approach is to adapt a system for detecting the replica of a specific reference image. The system is then able to classify test images as replicas of the reference image or as unrelated images. More precisely, the test procedure is as follows. A set of 162 features is extracted from a test image, representing texture, colour and grey-level characteristics. These features are then feed into a preprocessing step, which is fine-tuned to the reference image. First, the extracted features are weighted by comparing the proportion of pixels contributing to each feature in the test image to the corresponding one in the reference image. Second, the dimensionality of the features space is reduced to keep only a subset of features relevant to replica detection of the specific reference image. In the final step, the resulting features are entered to a support vector classifier that determines if the test image is a replica of the reference image.

Simulation results show the effectiveness of the proposed system. For instance, for an average false negative rate of 8%, one achieves a fixed false positive rate of $1 \times 10^{-4}$. Indeed, the proposed technique significantly outperforms DPF [30], at low false positive rates, even if we use fewer features. Although the achieved performance is not as good when compared to KPs [18], a speed up in terms of computational complexity, in the range of one to two orders of magnitudes, is achieved.

The paper is structured as follows. We present an overview of the proposed replica detection system in Section 2, and a thorough description of the various algorithmic steps in Section 3. In order to evaluate the performance of the proposed system, an evaluation methodology is defined in Section 4, and experimental results are reported in Section 5. Section 6 discusses applications of the proposed algorithm. Finally, conclusions are drawn in Section 7.

## 2. Overview and preliminary remarks

The notations used throughout this paper are first detailed. Subscripts in greek letters index vector elements. Subscripts in roman letters index vectors (or scalars). Training patterns (or examples) are denoted as $\mathbf{x}_i$, with $i = 1, \ldots, m$ where $m$ is the total number of training patterns. During the training phase, a label $y_i$ is assigned to each pattern $\mathbf{x}_i$. A pattern corresponding to a replica is simply called a *replica* and labelled $y_i = +1$. Otherwise it is called a *non-replica* and labelled $y_i = -1$. We denote the $\alpha$th feature of an image as $f_\alpha$. All features of an image can be held in a column vector denoted as $\mathbf{f}$.

We now present an overview of the proposed replica detection system. The system consists of six steps as shown in Fig. 1a. An outline of each step is provided in Section 2.1. The method can be decomposed into two distinct parts. The first one, consisting of the steps shown in the upper part of Fig. 1a, is independent from the reference image. Conversely the second one, comprising the steps shown in the lower part of Fig. 1a, depends on the reference image. Therefore, a training is needed for the latter steps. To achieve this, training examples are needed for both replicas and non-replicas, as detailed in Section 2.2. The training phase is outlined in Fig. 1b. The training performance is assessed using the F-score metric described in Section 2.3.

## 2.1. Method overview

*Image preprocessing*: In the first step, the test image is preprocessed. More specifically, the image is resized, and represented in a modified HSI colour space. This adds some degree of invariance against common image processing operations, such as resizing and illumination changes.

*Feature extraction*: Feature extraction maps images into a common space, where comparison is easier. For this purpose global statistics, such as colour channels and textures, are extracted from the test image.

*Weighted inter-image differences*: In the third step, the test image features are subtracted from those of the reference image, and 'incommensurable' features are penalised. For example, statistics about yellow pixels are considered as incommensurable when the test and reference images contain very different proportions of yellow pixels.

*Statistical normalisation*: In the fourth step, the inter-image differences are statistically normalised. In other words, the same importance is given to each feature, independently of their value range.

*Dimensionality reduction*: In the fifth step, the feature dimensionality is reduced. Less training examples are needed to train the decision function, and only feature mixtures relevant to the replica detection task are kept.

*Decision function*: Finally, in the last step, a decision function is used to determine if the test image is a replica of the reference image.

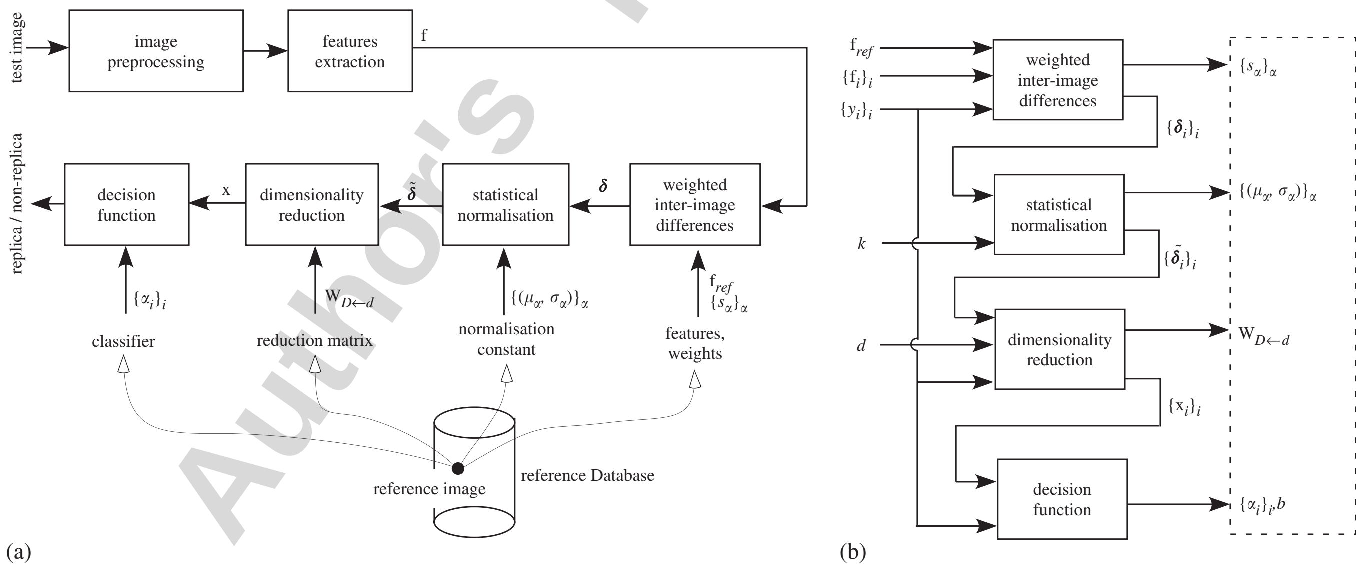

Fig. 1. System block diagram. The system is composed of two phases, namely training and testing. (a) presents the *block diagram for the testing phase*. A test image is given to the system that determines if it is a replica of a given reference image contained in the database. The method can be decomposed into two distinct parts: the upper part of (a) shows steps that are independent from the reference image while the lower part of (a) depicts steps that depend on the reference image. (b) depicts the *block diagram for the training phase of a single reference image*. The features $\mathbf{f}_{\text{ref}}$ of the reference image, the training examples $\{\mathbf{f}_i\}_i$, and the corresponding labels $\{y_i\}_i$ are fed to the training algorithm that produces the parameters of the steps depending on the reference image.

### 2.2. Training examples

Examples of *replica images* can be generated artificially. Indeed, the reference image can be modified using different operations, resulting in several replicas. In this work, the replicas are generated by the operations listed in Table 1. Furthermore, it is possible to have a richer set of training examples by nesting two or more operations to form a new operator known as a *composition*. However, we assume that an operation cannot be nested more than once in the same composition. For example, a JPEG compression cannot be followed by another JPEG compression with the same or a different quality factor. In this way, 419 replicas of the reference image are synthesised by using up to two nesting levels of compositions. Note that the number of generated examples becomes quickly unmanageable when the number of levels increases. For example, 14,293 replica examples are already generated by using only up to three nesting levels of composition.

Examples of *non-replica* images can be obtained by using a set of images that are known to be different from the reference image. This set can also be enriched by applying operations on its elements. In this study, we only consider the grey-level conversion. It permits to enrich the training set with grey-level images in order to avoid relying too heavily on the colour features.

### 2.3. Training metric

The F-score metric $F(\cdot)$ is used to assess the detection performance during the training phase. The F-score is defined as follows [9]:

$$F(TP, FP, P) = \frac{TP}{P} \times \frac{TP}{TP + FP}, \quad (1)$$

where $P$ is the total number of positive instances, $TP$ is the number of positive instances correctly classified, and $FP$ is the number of negative instances wrongly classified. The first term in the right-hand side of (1) corresponds to the *recall*. Conversely, the second term represents the *precision*. F-score balances these two conflicting properties: precision increases as the number of false positives decreases and recall decreases as the number of false negatives diminish (usually meaning that the number of false positives increases). Eq. (1) can be rewritten as

$$F_\rho(r_{\text{fp}}, r_{\text{fn}}) = (1 - r_{\text{fn}}) \times \frac{(1 - r_{\text{fn}})}{1 + \rho \cdot r_{\text{fp}} - r_{\text{fn}}}, \quad (2)$$

where $r_{\text{fp}} = FP/N$ and $r_{\text{fn}} = FN/P$ are the false positive and false negative rates. $\rho = N/P$ gives the ratio between the number of negative and positive instances. As for (1), the first term in the right-hand side of (2) corresponds to the recall, and the second one to the precision. In the rest of the document, we use the formulation given by (2). One 'drawback' of this metric lies in the ratio $\rho$ between the number of negative and positive instances; it has to be known beforehand.

Table 1
Training replicas generation

| Operations | Parameters |
|---|---|
| JPEG compression | $Q = 10, 50$ |
| Gaussian noise addition | $\sigma = 20/255, 60/255$ |
| Resizing | $s = 0.8, 1.2$ |
| Averaging filter | order= 2, 4 |
| Gamma correction | $\gamma = 0.8, 1.2$ |
| Horizontal flipping | NA |
| Grey-level conversion | NA |
| Rotation | $\theta = 90°, 180°, 270°$ |
| Cropping | keep 50% and 80% |
| V channel change | −10% and +10% |
| S channel change | −10% and +10% |

Image operations and their parameters.

## 3. Replica detection system

We now thoroughly describe the proposed replica detection system. In particular, each step presented in Fig. 1 is detailed along with the training procedures whenever required.

### 3.1. Image preprocessing

Before extracting features, an image is first cropped such that only 70% of its centre region is kept. This introduces a kind of weak robustness to operations such as framing. Then, it is resized such that it contains approximately $2^{16}$ pixels (corresponding to a square image of $256 \times 256$ pixels), while keeping its original aspect ratio. A weak form of scale invariance is introduced since the size of the preprocessed image is mostly constant regardless of the test image size. Moreover, this permits to speed up feature extraction by reducing the number of pixels to process.

The cropped and scaled image is then represented in a modified Hue saturation intensity (HSI) space: the logarithmic Hue, saturation, and equalised-intensity space. More specifically, the logarithmic Hue, $H_{\log}$, is defined as follows [10]:

$$H_{\log} = \frac{\log R - \log G}{\log R + \log G - 2\log B}, \tag{3}$$

where $R$, $G$ and $B$ are the red, green and blue values of a pixel. The logarithmic Hue has the advantage to be invariant to gamma and brightness changes [10]. The saturation, $S$, is the same as for classical HSI [11]:

$$S = 1 - \frac{3\min(R,G,B)}{R+G+B}. \tag{4}$$

By construction, the saturation is quite invariant to changes in illumination. Finally, the equalised illumination, $I_{\text{equ}}$, is given by

$$I_{\text{equ}} = \mathrm{T}\left(\frac{R+G+B}{3}\right), \tag{5}$$

where $\mathrm{T}(\cdot)$ is the global histogram equalisation operator [11]. The equalisation permits to make the intensity mostly invariant to changes of gamma and brightness as shown in the Appendix.

### 3.2. Features choice and extraction

In order to compare the similarity between two images, visual features are extracted. The goal of feature extraction is twofold. First, it maps images onto a common space where they can be more easily compared. Second, it reduces the space dimensionality by keeping only the relevant information.

Several visual features can be envisioned: colour, texture, shape, etc. For an extensive survey on general features extraction, refer to [31]. The choice of features depends on the image type. In the case of the image replica detection problem, it also depends on the type of replicas that are to be detected. For instance, if rotated images are considered, it would make sense to choose at least one feature that is rotation invariant.

The features used in this work are of three types: texture, colour and grey-level statistics. They are similar to those used in [30], in which they are found to give good results in image retrieval applications. The main differences are the added 24 grey-level features and the absence of 'local' statistics. The added grey-level features capture characteristic missed by the colour features (namely the distribution of the illumination level) and bring increased performance as demonstrated in Section 5.3. As shown in Table 2, we extract 162 features which are detailed in the following subsections.

#### 3.2.1. Texture features

The texture features are composed of the first and second order statistics of each subband of the Gabor transform. The latter is performed as in [24] on the equalised illumination. More precisely, the used parameters are 0.75 for the upper centre frequency, 0.05 for the lower centre frequency, five scales and six orientations. For more details about these parameters, refer to [24]. Mean and standard deviation estimates of the squared coefficients are computed for each of the 30 subbands. This results in a total of 30 mean and 30 variance estimates.

#### 3.2.2. Colour features

The colour features are based on the modified HSI colour space presented in Section 3.1. Each pixel in the image is classified into one of 10 *colour classes* depending on its position in this space. The classes are the achromatic colours ($S = 0$) *black*, *grey* and *white*, and the chromatic colours ($S > 0$) *red*, *orange*, *yellow*, *green*, *cyan*, *blue* and *purple*. The equalised illumination is used to classify a pixel into one of the three achromatic classes. The logarithmic Hue is used to classify a pixel in one of the seven chromatic classes.

This is similar to the 'culture' colour approach proposed in [30]. In this study, pink and brown are also considered, whereas in our case they are classified as red or orange. Brown and pink have similar values for the Hue channel as red or orange, but differ in the intensity and/or saturation

Table 2
Features overview

| Name | ♯ features |
|---|---|
| Gabor, squared coeff. mean | 30 |
| Gabor, squared coeff. std. dev. | 30 |
| Colour, histogram | 10 |
| Colour, channel mean | 24 |
| Colour, channel std. dev. | 24 |
| Colour, spatial distribution | 20 |
| Grey level, histogram | 8 |
| Grey level, spatial distribution | 16 |
| Total | 162 |

List the types of used features and the number of extracted statistics.

channels. Operations such as saturation or intensity changes are common in image processing; they modify the intensity and the saturation channels but not the Hue channel. If brown and pink are considered, red or orange pixels could be transformed into brown or pink pixels, or vice versa. For this reason, we have decided to include brown and pink within the red and orange classes.

*Colour classes histogram*: A histogram is computed, giving the proportion of each colour class in the image. It is normalised such that it sums to one, and comprises 10 values.

*Channel statistics*: Mean and variance estimates of the equalised intensity channel are computed for each colour class. Mean and variance estimates of saturation and logarithmic Hue channels are also computed for each chromatic colour class. This results in a total of 24 mean and 24 variance estimates.

*Spatial distribution shape*: The shape of the spatial distribution of each colour class is computed. This is achieved by computing two shape characteristics for each colour class: spreadness and elongation [13,22]. The first characteristic measures the compactness of the spatial distribution of a colour class. The second one gives the ratio between the shape length and width. Note that even if pixels assigned to a colour form totally disconnected components, this feature still captures useful information (namely the spatial distribution of these components). This results in 10 spreadness and 10 elongation measures.

### 3.2.3. Grey-level features

The grey-level features are based on the equalised intensity channel of the HSI model. The dynamic range of the image is linearly partitioned into eight bins corresponding to as many classes. Each pixel of the image falls into one of these bins.

The use of grey-level feature is important because the colour features can be unsuited in some cases. For instance, it can happen when the reference or the test images are grey level, or when conversion to grey level is one of the considered operations in the replica detection system.

*Grey-level classes histogram*: A grey-level classes histogram is computed, giving the proportion of eight intensity ranges in the image. It is normalised such that it sums to one, and comprises eight values.

*Spatial distribution shape*: Similarly to colour, the shape of the spatial distribution of each grey-level class is computed. This results in eight spreadness and eight elongation measures.

## 3.3. Weighted inter-image differences

Inter-image differences are computed during this step. They are basically the difference between the statistics of the test images and those of the reference image. Additionally, a weight is used to penalise certain statistics.

The channels statistics and spatial distribution shape for colour classes, and the spatial distribution shape for grey level correspond to non-overlapping regions of the images. These regions have not necessarily the same size for different images. Therefore, a legitimate question is whether image statistics are comparable when the regions from which they are estimated differ significantly in size.

In the following, a method is proposed that generates small inter-image features when inter-image class proportions are similar and large ones when they are dissimilar. More precisely, inter-image statistics corresponding to significantly different region size are penalised.

A 'weight' $w_\alpha$ is associated to each feature $f_\alpha$. The weight gives the proportion of pixels taken into account to compute $f_\alpha$. For the colour features, the weights are the entries of the colour histogram. Likewise, the weights corresponding to grey-level features are those of the grey-level histogram. For the remaining features, no weight is used.

We define the weighted inter-image difference $\delta_\alpha$ as

$$\delta_\alpha = f_\alpha - f_\alpha^{\mathrm{ref}} + \hat{s}_\alpha \cdot \mathrm{sgn}(f_\alpha - f_\alpha^{\mathrm{ref}}) \cdot (w_\alpha - w_\alpha^{\mathrm{ref}})^2, \tag{6}$$

where $f_\alpha^{\mathrm{ref}}$ and $w_\alpha^{\mathrm{ref}}$ are the $\alpha$th feature, respectively weight, of the reference image, and $\hat{s}_\alpha$ is a non-negative parameter that gives more or less importance to the discrepancy between the weights.

The idea behind (6) is as follows. On the one hand, a replica feature is assumed to be close to that of the reference image. Consequently, $f_\alpha - f_\alpha^{\mathrm{ref}}$ and $w_\alpha - w_\alpha^{\mathrm{ref}}$ are small, resulting in small $\delta_\alpha$. On the other hand, a non-replica feature has no relation to that of the reference image. Therefore, it is less likely that both $f_\alpha$ and $w_\alpha$ are simultaneously close than only $f_\alpha$ or $w_\alpha$. That is, (6) ensures that $\delta_\alpha$ is, with higher probability, smaller for a replica than for a non-replica.

The parameters $s_\alpha$ are found through a training procedure. To achieve this, simple classifiers are considered. There are as many simple classifiers as there are features, and the decision functions are given by

$$\mathrm{sgn}(T_\alpha - |\delta_\alpha(s_\alpha)|), \tag{7}$$

where $T_\alpha$ are thresholds chosen among the $|\delta_\alpha(s_\alpha)|$ corresponding to the replicas. The $\hat{s}_\alpha$ maximising the F-score of these classifiers are then chosen. Namely, the $\hat{s}_\alpha$ are computed as follows:

$$\hat{s}_\alpha = \arg_{s_\alpha} \max_{s_\alpha, T_\alpha} F_\rho(\hat{r}_{\mathrm{fp}}, \hat{r}_{\mathrm{fn}}), \tag{8}$$

where $\hat{r}_{\mathrm{fp}}$ and $\hat{r}_{\mathrm{fn}}$ give estimates of the false positive and false negative rates on the simple classifiers using thresholds $T_\alpha$ and parameters $s_\alpha$. To reduce computational complexity, the $s_\alpha$ are chosen among the following candidates $\tilde{\mu}_\alpha \times 10^k$, where the $\tilde{\mu}_\alpha$ are the average value of the $|f_\alpha|$ over all training non-replicas for the $\alpha$th feature, and $k = -\infty, -1, 0, +1, +2$.

### 3.4. Normalisation

The weighted inter-image difference are normalised using a statistical normalisation method [34]. More precisely, let $\mu_\alpha$ and $\sigma_\alpha$ be the mean and standard deviation estimates of the $\alpha$th inter-image difference over a subset of the training set. Training examples for which any feature is an extremum over the training set are ignored. Therefore, outliers are not taken into account. The normalised inter-image difference $\tilde{\delta}_\alpha$ is then given by

$$\tilde{\delta}_\alpha = \frac{\delta_\alpha - \mu_\alpha}{k \cdot \sigma_\alpha}, \tag{9}$$

where $\delta_\alpha$ is the inter-image feature given in (6). By Tchebychev's theorem, at least a fraction $1 - 1/k^2$ of the $\tilde{\delta}_\alpha$ are within the interval $[-1, 1]$. In the following $k$ is set to 10 so that more than 99% of the features are within $[-1, 1]$. The features outside this interval are clipped to $+1$ or $-1$.

The goal of normalisation is to ensure that the feature elements are commensurable. This is especially important for dimensionality reduction.

### 3.5. Dimensionality reduction

The relatively large number of features prevents to directly use many classification techniques on the weighted inter-image differences, as this would require a prohibitively large number of training examples, and the optimisation process would probably yield an overtrained decision function due to the 'curse of dimensionality' [5]. For this reason, the dimensionality $D$ of the vector $\tilde{\boldsymbol{\delta}}$ is reduced to $d$. In our case, the initial dimension $D$ is 162. The dimensionality reduction makes use of the following linear transformation:

$$\mathbf{x} = \mathbf{W}_{D \to d} \cdot \tilde{\boldsymbol{\delta}}, \tag{10}$$

where the $d \times D$ matrix $\mathbf{W}_{D \to d}$ is computed using the method proposed in [20]. This method is based on independent component analysis (ICA) [14]. It adds the class information to the feature vector in order to elect the independent components best suited to the binary classification problem.

As opposed to the other steps, the dimensionality reduction step is not based on the optimisation of the F-score. Future research works include therefore the development of a specific dimensionality reduction step based on the maximisation of the F-score.

### 3.6. Decision function

The decision function needs to determine whether the vector $\mathbf{x}$ corresponds to a replica of the reference image. This is a binary classification problem, where the two classes correspond to replicas and non-replicas, respectively. The goal is to build, using a limited number of training examples, a classifier that generalises well to novel patterns. Many classification algorithms can be used for this purpose. In our previous works [26,27], we showed that support vector machine (SVM) yielded good performance for the replica detection problem.

The basic SVM [32,2] is a binary classifier that separates two classes with a hyperplane. Furthermore, non-linear kernels allow to map patterns into a space where they can be better discriminated by a hyperplane.

#### 3.6.1. Support vector machine

We use the $\nu$-parameterisation [32,4] of the SVM, and a radial basis function as kernel. The dual constrained optimisation problem is given in (11). In the dual form, the Lagrangian is maximised by optimising the Lagrangian multipliers $\alpha_i$

$$\max_{\boldsymbol{\alpha}} -\frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \mathrm{k}(\mathbf{x}_i, \mathbf{x}_j), \tag{11}$$

subject to the constraints $\sum_{i=1}^{m} \alpha_i y_i = 0$, $\sum_{i=1}^{m} \alpha_i \geqslant \nu$, and $0 \leqslant \alpha_i \leqslant 1/m$. In this work, we use a radial basis function (RBF) kernel given by

$$\mathrm{k}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}\right). \tag{12}$$

The particular choice of kernel is motivated by several considerations. Not only is the linear SVM a particular case of the RBF kernel, but also the sigmoid and the RBF kernels behave similarly for certain choices of parameters [19]. Additionally, the RBF kernel presents less numerical difficulties than, for instance, the polynomial kernel. Finally, the RBF kernel is governed by only one parameter instead of two for the polynomial kernel.

The parameters of this classification technique are $\nu \in [0, 1]$ and $\sigma \in \mathbb{R}^+$. The parameter $\nu$ can be shown to be an upper bound on the fraction of training errors, and a lower bound on that of support vectors [32,4]. The kernel parameter $\sigma$ controls the complexity of the decision boundary. The constrained optimisation problem given in (11) can be solved by means of standard quadratic programming techniques.

Finally, the decision function indicates to which class the test pattern $\mathbf{z}$ belongs. It is given by

$$f(\mathbf{z}) = \operatorname{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i \mathrm{k}(\mathbf{z}, \mathbf{x}_i) + b\right), \tag{13}$$

where the constant $b$ is determined by the support vectors. More precisely, $b = y_k - \sum_{i=1}^{m} y_i \alpha_i \mathrm{k}(\mathbf{x}_i, \mathbf{x}_k)$, for all $\mathbf{x}_k$ such that $0 < \alpha_k < 1/m$. The name support vectors stems from the fact that many of the optimised $\alpha_i$ are equal to 0. Hence, only a relatively small fraction of the training patterns defines the decision function.

### 3.6.2. *Determination of the classification parameters*

In the $\nu$-SVM, the kernel parameter $\sigma$ and the parameter $\nu$ are to be determined in order to minimise the *generalisation error*. The latter is the error obtained when testing novel pattern with a trained decision function.

More precisely, we want to minimise the F-score $F(r_{\mathrm{fp}}, r_{\mathrm{fn}}, \rho)$ where $r_{\mathrm{fp}}$ is the generalisation error for false positive (novel non-replicas classified as replicas), $r_{\mathrm{fn}}$ is the generalisation error for false negative (novel replicas classified as non-replicas), and $\rho$ is the ratio between the number of novel non-replicas and replicas. In the considered application, there are usually many more non-replicas than replicas so that $\rho \gg 1$. Nevertheless, $\rho$ remains a priori unknown. Moreover, $r_{\mathrm{fp}}$ and $r_{\mathrm{fn}}$ are also unknown and need to be estimated.

Cross-validation is a popular technique for estimating generalisation error. In $k$-fold cross-validation, the training patterns are randomly split into $k$ mutually exclusive subsets (the folds) of approximately equal size. The SVM decision function is obtained by training on $k - 1$ of the subsets, and is then tested on the remaining subset. This procedure is repeated $k$ times, with each subset used for testing once. Averaging the test error over the $k$ trials gives an estimate of the expected generalisation error. This method has been shown to yield a good estimation of the generalisation error [6].

In the following, we use a normalised version of the radial basis function kernel where $\sigma$ in (12) is replaced by $\kappa \cdot \sigma$. The normalisation constant $\kappa$ is set to the second decile of the distribution of the intra-replica distances within the training set. It ensures that the optimal value of $\sigma$ is larger than one with high probability.

While $\nu$ has an intuitive signification, it is not clear what should be its optimal value [35,4]. It was shown that twice $\bar{R}$, a close upper bound on the expected optimal Bayes risk, is an asymptotically good estimate [35]. While no such bound can be easily determined a priori, this theorem induces an algorithm to find a good $\nu$ by starting with the classification error of a well-trained classifier as an approximation of the optimal Bayes risk [35].

In this work, a good a priori approximation of the optimal Bayes risk is not available. Consequently, good parameters for $\sigma$ and $\nu$ are estimated through a full grid search [35]. The procedure is divided into two steps: *coarse* and *fine* grid searches. In each step, a tenfold cross-validation is carried out for each feasible pairs $(\nu, \sigma)$. The pair for which the estimated F-score is the highest is then chosen. The tried pairs are the following:

- Coarse search: $(\sigma, \nu)$ for $\nu = 0.05 \cdot 2^k, k = -4, \ldots, 4$ and $\sigma = k, k = 1, \ldots, 10$.
- Fine search: $(\sigma, \nu)$ for $\nu = \nu_1 \cdot (1 + k/6), k = -2, \ldots, +2$ and $\sigma = \sigma_1 \cdot (1 + k), k = -2, \ldots, +2$. Here, $\nu_1$ and $\sigma_1$ denote the value determined in the first step.

## 4. Evaluation methodology

### 4.1. *Test images*

To simulate the performance of the proposed approach, we used the same image collection as in [18]. It contains 18,785 photographs including (but not limited to) landscapes, animals, constructions, and people. The image sizes and aspect ratios are variables, for example $900 \times 600$, $678 \times 435$, or

640 × 480. They are mostly colour images, except for about 1000 images that are grey levels.

For training, we randomly selected 700 images in the collection. Among the selected pictures, 200 are randomly chosen to be the reference images, and the remaining are used as non-replica examples during the training phase. For each reference image, a replica detector is built as described in Section 3.

The replica detectors are then tested on the remaining images in the collection. This permits to estimate the false positive rate for each reference image. The false negative rate is estimated by testing the replica detectors on test replica examples. They are generated by the transforms listed below. These operations are the same than those used in [30,18]. They are implemented using the free software suite ImageMagick [15]. There are 12 categories, as shown thereafter. An example for each of them is depicted in Fig. 2.

- *Colourising*: Tint the red, green, or blue channel by 10%;
- *Changing contrast*: Increase or decrease the contrast using ImageMagick's default parameter;
- *Cropping*: Crop by 5%, 10%, 20%, or 30%;
- *Despeckling*: Apply ImageMagick's despeckling operation;
- *Downsampling*: Downsample by 10%, 20%, 30%, 40%, 50%, 70%, or 90% (without antialiasing filtering);
- *Flipping*: Flip along the horizontal axis.
- *Colour depth*: Reduce the colour palette to 256 colours;
- *Framing*: Add an outer frame of 10% the image size. Four images are produced with different frame colour.
- *Rotating*: Rotate by 90°, 180° or 270°.
- *Scaling*: Scale up by 2, 4, 8 times, and down by 2, 4, 8 times (use antialiasing filter).
- *Changing saturation*: Change the values of the saturation channel by 70%, 80%, 90%, 110%, or 120%.
- *Changing intensity*: Change the intensity channel by 80%, 90%, 110%, or 120%.

### 4.2. Evaluation metrics

In order to evaluate the performance of the proposed system, we measure the trade-off between the *false positive* and *false negative* rates. The receiver operating characteristic (ROC) curve [9] is often used to represent the trade-off between error types; in this representation the true positive rate (one minus false negative rate) is plotted as a function of the false positive rate. In this study, we use a variant of the ROC curve called detection error trade-off (*DET*) curve [28]. Contrary to ROC curves, the *DET* curves represent the false negative rate as a function of the false positive rate. Since both axes correspond to error measurements, they can both make use of a logarithmic scale. The interpretation of *DET* curves is analogous to that of ROC curves: a classifier X is more accurate than a classifier Y when its *DET* curve is below that of Y.

The system's trade-off between false positive and false negative rates is assessed by making use of a single *DET* curve constructed as follows. For each reference image detector, a test set is produced (using the test images proposed in Section 4.1) by gathering the values under the sign function in the right-hand side of (13). For each test, a *DET* curve is then computed using Algorithm 2 in [9] (the false

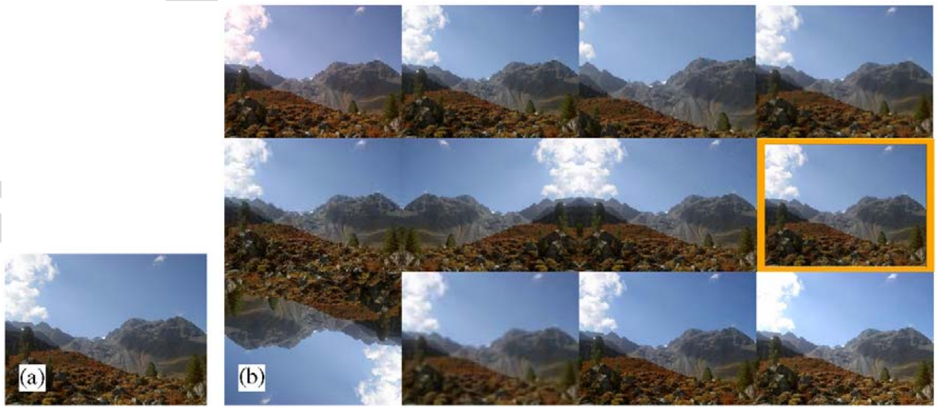

Fig. 2. Examples of test replicas. (a) is the original while (b) represents examples of test replicas. There is one replica example per category, the order used is the same than in the text (left–right, top–down).

negative rates are given by one minus the true positive rates); this algorithm works by using the different values in the test set as the thresholds. Finally, all curves are synthesised in a single *DET* curve, denoted $\overline{DET}$, by using vertical averaging [9, Algorithm 5]. This implies that a working point on the average *DET* curve corresponds to thresholds that are (possibly) different for each reference image detector. In practice, a lookup table can be used to determine the correct threshold values in function of the chosen working point (which depends on the application as shown in Section 6). Additionally, the precision of the *DET* curve is limited since the number of test examples is relatively low. Accordingly, the maximal absolute precision that can be achieved on the vertically averaged *DET* curve is reported in Table 3. It takes into account the number of test replicas ($P = 40$), the number of test non-replicas ($N = 18,085$), and the number of individual *DET* curves ($n = 200$).

Table 3
$\overline{DET}$ curve precision

| Error type | Maximal absolute precision |
|---|---|
| $r_{\text{fp}}$ | $\pm\frac{1}{18,085} = \pm 5.5 \times 10^{-5}$ |
| $r_{\text{fn}}$ | $\pm\frac{1}{40 \cdot 200} = \pm 1.3 \times 10^{-4}$ |

The precision is obtained by taking into account the number of test examples (18,085 for the non-replicas, and 40 for the replicas), and the number of individual *DET* curves used for averaging (200).

## 5. Results

In this section, we present experimental results in order to evaluate the performance of the proposed replica detection system. In the following, the used parameters are $d = 50$ for dimensionality reduction, and $\rho = 10^4$ for the F-score parameterisation, unless stated otherwise. Additionally, whenever appropriate, the equal error rate (*EER*) working points of the $\overline{DET}$ curve is also shown by depicting the *EER* curve.

### 5.1. Influence of the F-score metric parameterisation

In a first experiment, we explore the effect of possible parameterisation of the F-score metric. The value $\rho$ gives the ratio between the number of expected non-replica instances and that of expected replica instances. In the considered applications, these numbers can hardly be determined a priori. However, we can safely assume that $\rho$ is much larger than one because there are many more non-replicas than replicas.

Fig. 3a shows $\overline{DET}$ curve for $\rho = 1$. Additionally, Fig. 3b depicts the vertical differences between $\overline{DET}$ curves for $\rho = \{10^2, 10^4, 10^6\}$ and that for $\rho = 1$. Globally, different values of $\rho$ influence only slightly on the results, namely the difference is less than 1%. However, high $\rho$ values favour classifiers

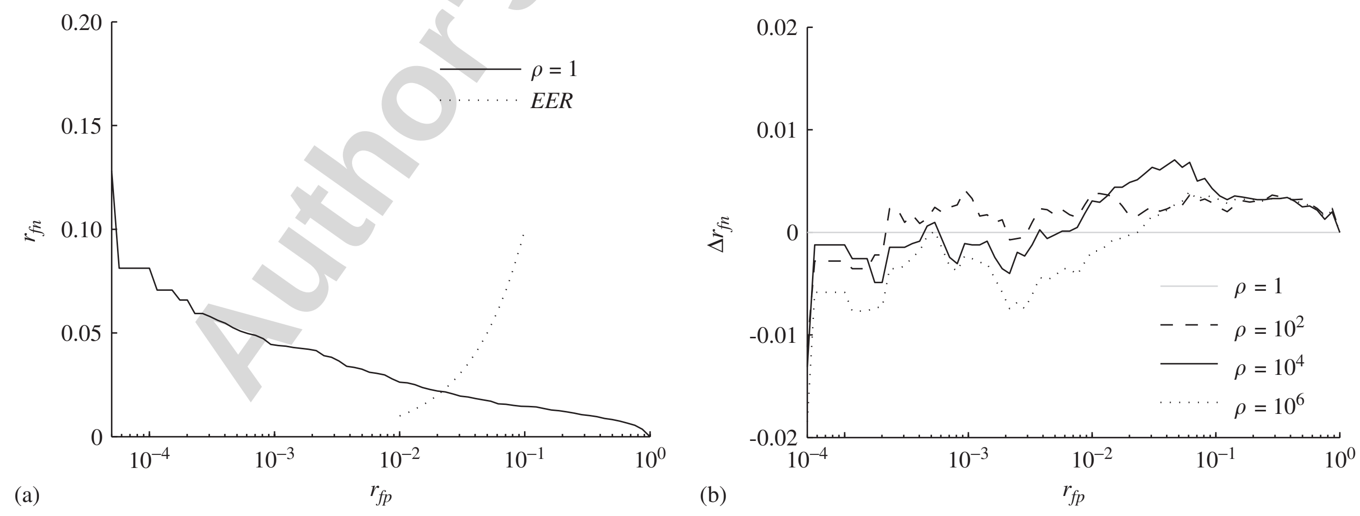

Fig. 3. Influence of $\rho$. All curves used a fixed number of dimensions $d = 50$, only the value of $\rho$ is changed. (a) shows the $\overline{DET}$ curve for $\rho = 1$, as well as the equal error rate (*EER*) curve. (b) depicts the vertical differences between the $\overline{DET}$ curves corresponding to $\rho = \{10^2, 10^4, 10^6\}$ and that corresponding to $\rho = 1$.

with very low false positive rates while keeping reasonable false negative rates.

### 5.2. DET curves distribution

We now analyse the distribution of the *DET* curves before vertical averaging. Fig. 4 shows the false negative rates histograms for a fixed false positive rate $r_{\mathrm{fp}} = 5 \times 10^{-5}$ (leftmost value of the curves in Fig. 3) and two different values of $\rho$, namely 1 and $10^4$. On the one hand, the two histograms show that over half of the individual classifiers have false negative rates below 0.05. Actually, about one-fourth of the classifiers have no false negative at all. On the other hand, the two histograms show that a small number of classifiers presents false negative rates above 0.5. The corresponding reference images possess few colours, or are grey-level images. It shows that colour related features are very powerful discriminating features, and that the lack of colour variety complicates the replica detection task. Moreover, these 'bad' classifiers participate heavily in increasing the average false negative rates. When their proportion diminishes, so does the average false negative error. This explain the results obtained in Fig. 3 since the number of 'bad' classifiers decreases from 15 to 9 when $\rho$ augments from 1 to $10^4$.

### 5.3. Grey-level features

In this trial, the detection performance obtained by using grey-level features is compared to that when not using them. Fig. 5 depicts the performance improvement brought by adding grey-level features. The performance gap augments as the false positive rate decreases. Note that grey-level images are present in both the reference images and the test images. Directly making use of grey-level features greatly improves the performance on these images. Moreover, it also increases the performance for colour images. Indeed, grey-level features capture information that is missed by the colour features, namely the global intensity shape.

A possible improvement is to use a replica detection system to test colour images, and another one which does not make use of colour features to test grey-level images. The drawback of such an approach is that it requires storing two descriptions for each reference image.

### 5.4. Weighted inter-image differences

In this experiment, we analyse the advantage of using weighted inter-image differences. Fig. 6 illustrates the performance increase due to the weighted inter-image differences. The *DET* curve with $s_\alpha = 0$ corresponds to the situation where the differences are not weighted. In this case, (6) becomes $\delta_\alpha = f_\alpha - f_\alpha^{\mathrm{ref}}$. The other *DET* curve corresponds to the case where the $s_\alpha$ have been optimised as described in Section 3.3. The performance gap increases as the false positive rate diminishes. For low false positive rates, the false replicas are mainly images that are similar to the reference image. For example, if the reference is a

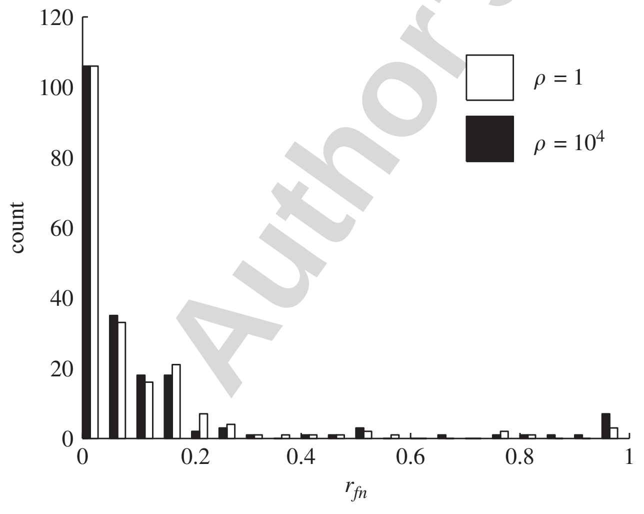

Fig. 4. False negative histograms. For a fixed false positive rate $r_{\mathrm{fp}} = 5 \times 10^{-5}$ (leftmost value of the curves in Fig. 3), number of dimensions $d = 50$, and two different values of $\rho$.

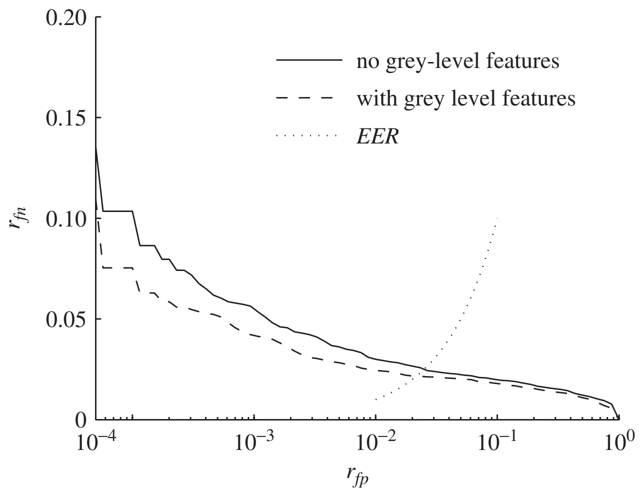

Fig. 5. Influence of grey-level features. $\overline{DET}$ curves with, and without, using grey-level features. The equal error rate (*EER*) curve is also shown.

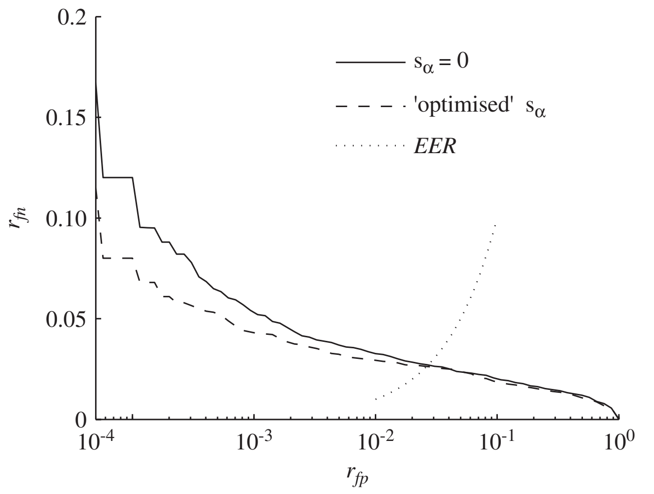

Fig. 6. Influence of the weighted inter-image differences. $\overline{DET}$ curves for $s_\alpha = 0$ and optimised $s_\alpha$. The equal error rate (*EER*) curve is also shown.

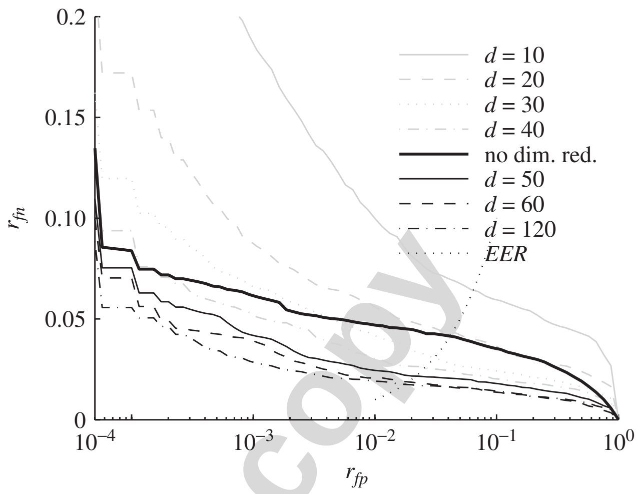

Fig. 7. Influence of the dimensionality reduction. $\overline{DET}$ curves for different values of *d*. The equal error rate (*EER*) curve is also shown.

picture of a city, most false replicas contain buildings or straight structures. That is, many of the features of the false replicas are close to those of the reference image. In this situation, the use of weighted inter-image differences helps in taking the correct decision.

### 5.5. Dimensionality reduction performance

We now explore the efficiency of the dimensionality reduction step. Fig. 7 illustrates the performance obtained with different dimensions *d*. In general, the larger the number of dimensions, the better the performance. However, the gain is not very important when going beyond $d = 50$. The thick line represents the performance achieved without dimensionality reduction. In this case, the performance is mostly equivalent to that of $d = 40$ for low false positive rates ($r_{\mathrm{fp}} < 10^{-4}$) and significantly lower for higher false positive rates. This can be easily explained by the limited number of training examples. Indeed, the number of needed training examples grows as the number of dimensions increases [7,5].

Another factor explaining the results obtained in Fig. 7 is related to metric consideration. When no dimensionality reduction is applied, the distance used for the kernel computation is proportional to a weighted Euclidian distance where the weights are given by the inverse of the standard deviation, as shown by (9) and (12). This gives a priori equal weights to each feature. However, there is no reason that each feature carries the same amount of discriminative information. Moreover, they depend on each other. When dimensionality reduction is applied, the reduced patterns $\mathbf{x}_i$ are of unit variance, and theoretically *independent* [14]. In that case, giving equal weights to each feature makes more sense and gives better results. In fact, recent works have shown that using metrics that are learned from side-information can improve classification results [37,1]. In the case of the replica detection problem, two side-information are available: the class (replica or non-replica), and the relative distance to the reference for replicas (for example, a JPEG compressed image with a quality factor of 10 is farther to the reference than one compressed with a quality factor of 90). Future research will study the improvements brought by using learned metric in the replica detection problem.

### 5.6. Efficiency

The replica detection method efficiency is now analysed in terms of storage requirement and computational effort.

A number of parameters are needed to compare a test image to a given reference. Namely, they are the reference image features and weights, the normalisation constants, the dimensionality reduction matrix, and the support vectors of the decision functions. In the following, we refer to the aforementioned elements as the *description* of the reference image. The storage requirement are listed in Table 4. On average, about 25 kbytes are needed to store each description. In other words, one megabyte can held, on average, up to 40 descriptions. This is a negligible amount of memory for today's computer.

Another important aspect is that of computational complexity of the method. The proposed method requires a training for each reference image. The training is computationally complex and takes up to 15 min per reference image on a PC with a 2.8 GHz processor and 1 Go of RAM. Feature extraction from the replica examples is the most complex part of the training, and takes up to 75% of the running time. Since training can be done off-line, its computational complexity is less critical.

The computational complexity of testing is estimated in Table 5. Note that except for the SVM part, the method is implemented in Matlab without any optimisation. This incurs longer running time. For instance, the feature extraction could be reduced to, at least, 0.1 s [30]. In the discussion that follows, we assume an optimised feature extraction step. The preprocessing and feature extraction steps are reference image independent, and take about 0.3 s. The remaining steps are reference image dependent, and take about $0.2 \times 10^{-3}$ s. When the reference image database contains less than 1500 images, most of the testing time is spent on the test image preprocessing and the feature extraction. In that case, up to four test images can be processed per second. For larger reference databases, most of the testing time is spent on the reference image dependent steps. The number of test images that can be processed each second decreases linearly as the number of reference images grows. Future research will concentrate on pruning the reference images, in order to avoid testing them all. That is, only the reference images for which the test image can be potentially a replica are selected. Such methods can reduce the testing time, and have been applied with success in [18,30].

Table 4
Storage requirements estimation

| Name | Size (bytes) |
|---|---|
| Image features | $162 \times 2 = 324$ |
| Image weights | $162 \times 0 = 0$ |
| Norm. const. | $2 \times 162 \times 2 = 648$ |
| Dim. reduct. matrix | $50 \times 162 \times 2 = 16,200$ |
| Support vectors | $50 \times 85 \times 2 = 8,500$ |
| Total | 25,672 = 25 kb |

Real number are coded on 16 bits (two bytes). The number of dimensions after dimensionality reduction is 50. In our experiment (using $d = 50$ and $\rho = 10^4$), the average number of support vectors is found to be about 85.

Table 5
Average running time for testing

| | Name | Time [s] |
|---|---|---|
| Reference image independent | Preprocessing | 0.2 |
| | Feature extraction | 1.8[a] |
| Reference image dependent | Weighted features | $48 \times 10^{-6}$ |
| | Normalisation | $90 \times 10^{-6}$ |
| | Dim. reduction | $17 \times 10^{-6}$ |
| | Decision function | $50 \times 10^{-6}$ |

The experiments were carried out on a PC with a 2.8 GHz processor and 1 Go of RAM.
[a]0.1 s when optimised as in [30].

## 5.7. *Comparison with existing replica detection methods*

Fig. 8 compares the performance of the proposed replica detection system with state of the arts techniques in [18,30]. The continuous line corresponds to the vertically averaged $\overline{DET}$ curve obtained with our system, using $d = 50$ and $\rho = 10^4$. The dashed line represents the performance of a replica detection method based on DPF [30]. The circle point indicates the performance of a replica detection system based on KPs [18]. Both methods are set in the image retrieval framework and, therefore, give the result in terms of precision–recall measurements. It is, however, possible to translate a precision–recall curve into a $\overline{DET}$ curve. Indeed, the first term in the right-hand side of (2) is equal to the recall, and permits to trivially compute the false negative rate. Similarly, since the second term in the right-hand side of (2) is equal to the precision, it is also straightforward to determine the false positive rate given the ratio $\rho$ and the previously computed false negative rate. Accordingly, the $\overline{DET}$ curve for the DPF method is obtained by inspecting the precision–recall curve reported in Fig. 5 of [30] and $\rho = \frac{20,000}{40}$. The point for KPs method is computed using the information reported in Tables 1 and 2 of [18] and $\rho = \frac{18,722}{40}$.

It can be seen that the proposed method achieves good performance. For instance, an average false negative rate of 8% corresponds to a fixed false positive rate of $1 \times 10^{-4}$. On the one hand, the proposed method outperforms that of DPF for false positive rates below $10^{-2}$. Moreover, it should be

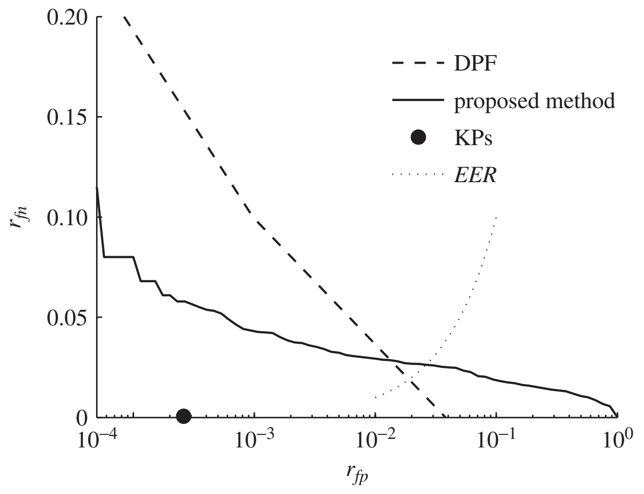

Fig. 8. Comparison with other methods. The proposed method is compared against other methods: KPs [18] and DPF [30]. The equal error rate (*EER*) curve is also shown.

noted that the features used in the current work are mainly a subset of those used in DPF: we use 162 features against 298 in the latter study. On the other hand, the proposed method is outperformed by KPs. In our method, most of the wrongly classified replicas (false negative errors) correspond to replicas for which the illumination or the intensity have been changed to a great extent. The KPs method uses features (salient points, or key points [23]) invariant to this change but computationally more complex to extract. Indeed, the feature extraction time of KPs is between 1 and 10 s per image [18,30]. This is between 10 to 100 times slower than that for the proposed method (when optimised as in [30]).

## 6. Applications and scenarios

The proposed image replica detection system is suitable to detect copyright infringement or to identify illicit content known to the police. In this section, we discuss in more details the scenarios to use the proposed technique in such a task.

In the design of our system, we assume a given database of reference images, and we test input test images towards this database in order to identify replicas. Furthermore, we assume that the set of test images can be extremely large, but the set of reference images is moderate in size. To guarantee a fast testing procedure, we consider a limited number of low-level features, as previously discussed. Note that in order to handle large set of reference images, a subset of all the features can be used in a first step in order to quickly drop test images which are classified as non-replicas with high confidence.

In the target applications, the database of reference images can for instance be a collection of copyrighted images or illicit child pornography images. To perform the task of copyright infringement or illicit content detection, several configurations are possible. In one scenario, we consider a proxy server performing network sniffing at nodes of the Internet. The proxy server contains the database of reference images, or more specifically the extracted features, normalisation constants, reduction matrix and classifier for each reference. Subsequently, the proxy server processes each incoming image, applying preprocessing and features extraction, compares it pairwise with each reference image, and finally decides whether it is a replica of one image in the database. In another scenario, we consider a web crawler to search for replicas on the Internet. Similarly to the previous case, the crawler looks for test images and compares them to a database of reference images in order to identify replicas.

Other applications and scenarios are also possible, although the proposed system may be less suited for them. For instance, the technique can also be used in an image web search engine in order to prune the results of a query by eliminating replicas. Alternatively, it is also possible to build an index of web images, and to check whether a given reference image has replicas in this index.

As common in a classification technique, a trade-off exists between the false positive and false negative rates. On the one hand, a low false positive rate is desired whenever a user does not want to be overwhelmed by false positive. On the other hand, a low false negative rate is preferable for a user who wishes to detect all possible replicas. The optimal trade-off is therefore application dependent.

## 7. Conclusion

In this paper, we have described a technique to decide whether a test image is a replica of a given reference image. We performed experiments on a large collection containing 18,785 photographs representing a wide range of content. We were able to detect, on average, 92% of the replicas while achieving a fixed false positive rate of only $1 \times 10^{-4}$. Moreover, we showed that using a replica detector that is fine-tuned to each reference image can greatly improve the performance.

Future works include the addition of a pruning step in order to decrease the number of reference image to be tested (presently all reference images are to be tested). It can be accomplished by means of tree-based indexing techniques. Another direction consists in using additional side information to improve the fine-tuning of the replica detector. Finally, the performance of the system could be improved by taking into account a larger set of invariant features, which is also among the future research directions under consideration.

## Acknowledgements

The first author is partly supported by the Swiss National Science Foundation—"Multimedia Security", Grant number 200021-1018411. This research was also partially supported by the European Commission under contract FP6-027026-K-SPACE [8]; this work presents the view of the authors but not necessarily that of the community. Additionally, the authors would like to thank the creators of the SVM library libsvm [3], and those of the image processing software suite ImageMagick [15]. Finally, we are grateful to David Marimón Sanjuán for fruitful discussions and insights.

## Appendix A. Invariance of equalised illumination to reversible transformation

In this appendix, we give the proof of the invariance of equalised illumination to gamma and illumination changes. Intensity and gamma changes are modelled as

$$\mathrm{g}(r) = \alpha r^{\gamma}, \tag{A.1}$$

where $r$ is the intensity of the input image (in the range $[0, 1]$), and $\mathrm{g}(r)$ that of the output image.

Let $\mathrm{p}_i(w)$ be the probability density of pixels of the input image. It follows that the probability density of the output image is given by

$$\mathrm{p}_o(w) = \mathrm{h}'(w)\mathrm{p}_i(\mathrm{h}(w)), \tag{A.2}$$

where $\mathrm{h}(w) = \mathrm{g}^{-1}(w) = (w/\alpha)^{1/\gamma}$ and $\mathrm{h}'(w)$ is its first derivative.

The global histogram equalisation maps the image with a given probability density $p(w)$ to an image with an uniform probability density. The mapping is given by [11]

$$s = \mathrm{T}(r) = \int_0^r p(w)\,\mathrm{d}w, \tag{A.3}$$

where $r$ is the intensity before equalisation, and $s$ the one after.

Let $s_0 = \mathrm{T}(r_0)$ be the equalised intensity after changes of illumination and gamma on $r_i$. More precisely,

$$\begin{aligned}
s_0 = \mathrm{T}(r_0) = \mathrm{T}(\mathrm{g}(r_i)) &= \int_0^{\mathrm{g}(r_i)} \mathrm{p}_o(w)\,\mathrm{d}w &\text{(A.4)}\\
&= \int_0^{\mathrm{g}(r_i)} \mathrm{h}'(w)\mathrm{p}_i(\mathrm{h}(w))\,\mathrm{d}w &\text{(A.5)}\\
&= \int_0^{\mathrm{h}(\mathrm{g}(r_i))=r_i} \mathrm{h}'(\mathrm{g}(v))\mathrm{p}_i(\mathrm{h}(\mathrm{g}(v)))\mathrm{g}'(v)\,\mathrm{d}v &\text{(A.6)}\\
&= \int_0^{r_i} \mathrm{h}'(\mathrm{g}(v))\mathrm{p}_i(v)\mathrm{g}'(v)\,\mathrm{d}v &\text{(A.7)}\\
&= \int_0^{r_i} \frac{\mathrm{d}}{\mathrm{d}v}\left[\underbrace{\mathrm{h}(\mathrm{g}(v))}_{=v}\right]\mathrm{p}_i(v)\,\mathrm{d}v &\text{(A.8)}\\
&= \int_0^{r_i} \mathrm{p}_i(v)\,\mathrm{d}v = \mathrm{T}(r_i) = s_i. &\text{(A.9)}
\end{aligned}$$

An image and its versions processed by (A.1) are mapped to the same equalised image by (A.3). The only fact used above is that an inverse exists for the function $\mathrm{g}(\cdot)$. Therefore, the above results can be generalised to any reversible transformation of the image.

Note that it cannot be proved in general that the discrete version of the histogram equalisation produces the discrete equivalent of a uniform probability function [11]. However, in practice, discrete histogram equalisation yields images that are mostly invariant to the gamma and illumination changes.

## References

[1] A. Bar-Hillel, T. Hertz, N. Shental, D. Weinshall, Learning distance functions using equivalence relations, in: Proceedings of International Conference on Machine Learning, Washington, DC, USA, 2003, pp. 11–18.
[2] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowledge Discovery 2 (2) (1998) 121–167.
[3] C. Chang, C. Lin, LIBSVM: a library for support vector machines, 2001, URL ⟨http://www.csie.ntu.edu.tw/~cjlin/libsvm⟩.
[4] P.-H. Chen, C.-J. Lin, B. Schölkopf, A tutorial on $\nu$-support vector machines, Appl. Stochastic Models Bus. Ind. 21 (2005) 111–136.
[5] D.L. Donoho, High-dimensional data analysis: the curses and blessings of dimensionality, August 2000, URL ⟨http://www-stat.stanford.edu/~donoho/⟩.

[6] K. Duan, S.S. Keerthi, A.N. Poo, Evaluation of simple performance measures for tuning SVM hyperparameters, Neurocomputing 51 (2003) 41–59.

[7] R. Duda, P. Hart, D. Stork, Pattern Classification, Wiley-Interscience, New York, 2000.

[8] A European Network of Excellence on ''Knowledge Space of Semantic inference for automatic annotation and retrieval of multimedia content'', URL ⟨http://www.kspace-noe.net/⟩.

[9] T. Fawcett, ROC Graphs: notes and practical considerations for data mining researchers, Technical Report HPL-2003-4, 2003.

[10] G. Finlayson, G. Schaefer, Hue that is invariant to brightness and gamma, in: Proceedings of British Machine Vision Conference, Manchester, England, pp. 303–312.

[11] R. Gonzalez, R. Woods, Digital Image Processing, 2/E, Prentice-Hall, Englewood Cliffs, NJ, 2002.

[12] F. Hartung, M. Kutter, Multimedia watermarking techniques, Proc. IEEE 87 (7) (1999) 1079–1107.

[13] M.-K. Hu, Visual pattern recognition by moment invariants, IEEE Trans. Inf. Theory (1962) 179–187.

[14] A. Hyvarinen, E. Oja, Independent component analysis: algorithms and applications, Neural Networks 13 (4–5) (2000) 411–430.

[15] ImageMagick: a free software suite to create, edit, and compose bitmap images, URL ⟨http://www.imagemagick.org⟩.

[16] ISO/IEC JTC1/SC29/WG11 WG11N4358, Final Draft International Standard 15938-3 Information Technology—Multimedia Content Description Interface—Part 3 Visual, July 2001.

[17] ISO/IEC JTC1/SC29/WG1 WG1N3684, JPSearch Part 1 TR—System Framework and Components, July 2005.

[18] Y. Ke, R. Sukthankar, L. Huston, An efficient parts-based near-duplicate and sub-image retrieval system, in: ACM International Conference on Multimedia, New York, USA, 2004, pp. 869–876.

[19] S.S. Keerthi, C.-J. Lin, Asymptotic behaviors of support vector machines with Gaussian kernel, J. Neural Comp. 15 (7) (2003) 1667–1689.

[20] N. Kwak, C.-H. Choi, Feature extraction based on ICA for binary classification problems, IEEE Trans. Knowledge Data Eng. 15 (6) (2003) 1374–1388.

[21] F. Lefèbvre, B. Macq, J.-D. Legat, Rash: radon soft hash algorithm, in: EURASIP European Signal Processing Conference, Toulouse, France, 2002.

[22] J.-L. Leu, Computing a shape's moments from its boundary, Pattern Recognition 24 (10) (1991) 949–957.

[23] D. Lowe, Distinctive image features from scale-invariant keypoints, Internat. J. Comput. Vision 60 (2) (2004) 91–110.

[24] B.S. Manjunath, W.Y. Ma, Texture features for browsing and retrieval of image data, IEEE Trans. Pattern Anal. Machine Intell. 18 (8) (1996) 837–842.

[25] Y. Maret, G. Molina, T. Ebrahimi, Images identification based on equivalence classes, in: Proceedings of Workshop on Image Analysis for Multimedia Interactive Services, Montreux, Switzerland, 2005.

[26] Y. Maret, G. Molina, T. Ebrahimi, Identification of image variations based on equivalence classes, in: Proceedings of SPIE Visual Communications and Image Processing, SPIE, Beijing, China, 2005.

[27] Y. Maret, F. Dufaux, T. Ebrahimi, Image replica detection based on support vector classifier, in: Proceedings of SPIE Applications of Digital Image Processing XXVIII, Santa Barbara, USA, 2005.

[28] A. Martin, G. Doggintgton, T. Kamm, M. Ordowski, M. Przybocki, The *DET* curve in assessment of detection task performance, in: Proceedings of Eurospeech, Rhodes, Greece, 1997, pp. 1895–1898.

[29] J.M. Martínez, R. Koenen, F. Pereira, MPEG-7: the generic multimedia content description standard, IEEE Multimedia 9 (2) (2002) 78–87.

[30] A. Qamra, Y. Meng, E.Y. Chang, Enhanced perceptual distance functions and indexing for image replica recognition, IEEE Trans. on Pattern Anal. Machine Intell. 27 (3) (2005) 379–391.

[31] Y. Rui, T. Huang, S. Chang, Image retrieval: current techniques, promising directions and open issues, J. Visual Commun. Image Representation 10 (4) (1999) 39–62.

[32] B. Schölkopf, A. Smola, R. Williamson, P.L. Bartlett, New support vector algorithms, Neural Networks 22 (2000) 1083–1121.

[33] J. Seo, J. Haitsma, T. Kalker, C. Yoo, Affine transform resilient image fingerprinting, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, Hong Kong, China, 2003, pp. III61–III64.

[34] J. Smith, A. Natsev, Spatial and feature normalization for content based retrieval, in: IEEE International Conference Multimedia and Expositions, vol. 1, Ischia, Italy, 2002, pp. 193–196.

[35] I. Steinwart, On the optimal parameter choice for $\nu$-support vector machines, IEEE Trans. Pattern Anal. Machine Intell. 25 (10) (2003) 1274–1284.

[36] R. Venkatesan, S.-M. Koon, M.-H. Jakubowski, P. Moulin, Robust image hashing, in: IEEE International Conference on Image Processing, Vancouver, Canada, 2000, pp. 664–666.

[37] E. Xing, A. Ng, M. Jordan, S. Russell, Distance metric learning with application to clustering with side-information, in: Proceedings of Advances in Neural Information Processing Systems, Vancouver, Canada, 2003, pp. 505–512.