Image Replica Detection based on Binary Support Vector Classifier

Yannick Maret, Frédéric Dufaux, and Touradj Ebrahimi

***Abstract*— In this paper, we present a system for image replica detection. More specifically, the technique is based on the extraction of 162 features corresponding to texture, color and gray-level characteristics. These features are then weighted and statistically normalized. To improve training and performances, the features space dimensionality is reduced. Lastly, a decision function is generated to classify the test image as replica or non-replica of a given 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 illicit content (e.g. pedophile images).**

***Index Terms*— image replica detection, features extraction, support vector machine, dimensionality reduction, image search, copyright infringement**

I. 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 colors.

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 illicit content*, such as child pornography images.

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 has been the subject of extensive research works and standardization activities such as MPEG-7 [1], [2] and the more recent JPSearch [3]. However, the problem of image replica detection has so far been the focus of fewer research efforts.

Two approaches to detect image replicas are *watermarking* [4] and *robust fingerprinting* [5]–[7]. Watermarking techniques [4] 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 [5]–[7] analyze the reference image in order to extract a signature associated with the image content. Replicas are then identified whenever their signatures are close to the one of the reference. This class of techniques is often based on a single feature, for example characteristic points of the Radon transform [5], log-mapping of the Radon transform [6], or intra-scale variances of wavelet coefficients [7]. While it is usually robust, computationally efficient, and suitable for fast database indexing and retrieval, it however performs poorly for the accurate classification of replicas and non-replicas.

More recently, techniques for image replica detection have been described in [8], [9]. Ke *et al* [8] propose a method based on the extraction of features, referred to as Key Points (KPs), which are stable in the 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* [9] 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 test images, to measure the similarity between the two. The main idea of the approach is to activate different features for different images pairs. More specifically, only the most similar features are taken into account to compute the distance. While this method achieves good performance, it is inferior to [8].

In this paper, we introduce a new approach for image replica detection based on our earlier work in [10]–[12]. More precisely, we extract 162 features from each image, representing texture, color and gray-level characteristics. The resulting features are then weighted by the proportion of pixels contributing to each feature, and statistically normalized to ensure that the features are commensurable. In the next step, the dimensionality of the features space is reduced. In this way, less training examples are needed and only features relevant to the given images are kept. Finally, a decision function is built to determine if the test image is a replica of the given reference image. Note that the considered approach consider a replica detector that is specifically fine-tuned to *each* reference image.

Simulation results show the effectiveness of the proposed system. For instance, for an average false negative rate of 8%, we achieve a fixed false positive rate of $1 \cdot 10^{-4}$. Indeed, our technique significantly outperforms DPF [9] even though we use fewer features. While our performance is not as good as KPs [8], we obtain a speed up in terms of computational complexity in the range of 2 to 3 orders of magnitudes.

This paper is structured as follow. We present an overview of the proposed replica detection system in Sec. II, and a more thorough description of the various algorithmic steps in Sec. III. In order to evaluate the performance of the proposed system, the evaluation methodology is defined in Sec. IV, and experimental results are reported in Sec. V. In Sec. VI, we discuss applications of the system. Finally, we draw conclusions in Sec. VII.

II. Overview and Preliminary Remarks

We first 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 Sec. II-A. 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 the latter steps need to be trained. To achieve this, training examples are needed for both replicas and non-replicas, as detailed in Sec. II-B. The training phase is outlined in Fig. 1b. The training performance is assessed using the F-score metric described in Sec. II-C. Notations used throughout this paper are detailed in Sec. II-D.

A. Method Overview

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

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

c) Weighted Inter-image Differences: In the third step, the test image features are subtracted from those of the reference image, and 'incommensurable' features are penalized. For example, statistics about yellow pixels are incommensurable when the test and reference images contain very different proportions of yellow pixels.

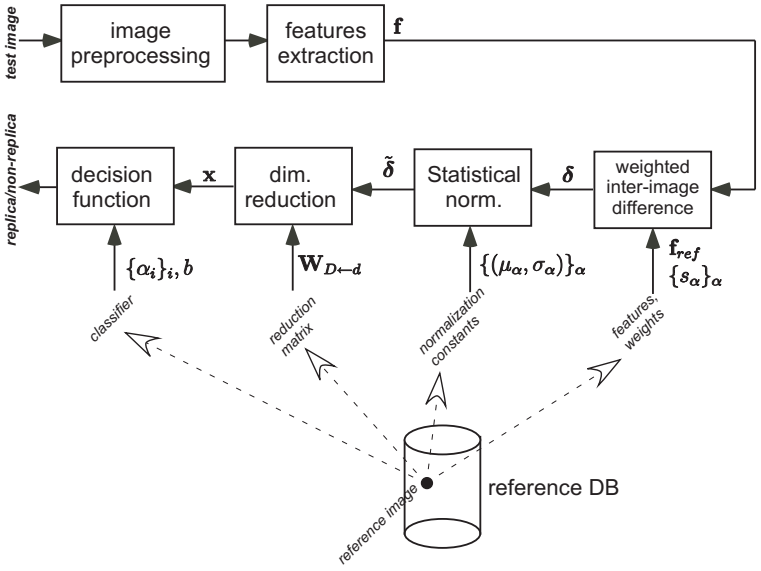
d) Statistical Normalization: In the fourth step the inter-image differences are statistically normalized. In other words, the same importance is given to each feature, independently of their value range.

e) Dimensionality Reduction: In the fifth step the feature dimensionality is reduced. Less training examples are needed, and only feature mixtures relevant to the replica detection task are kept.

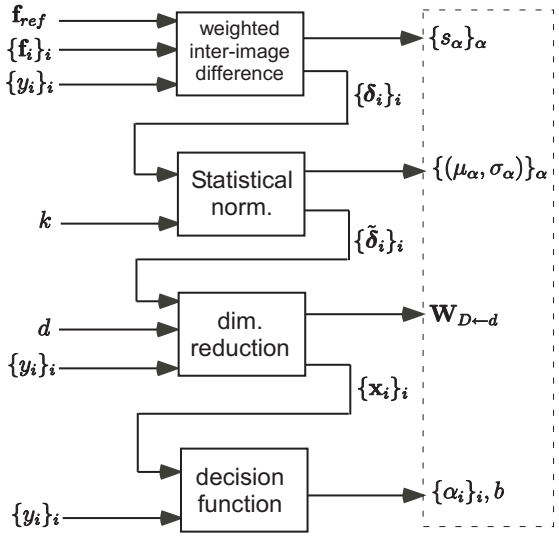
f) Decision Function: Finally, in the last step a decision function is used to determine if the test image is a replica of the reference image.

B. Training Examples

Examples of *replica images* can be generated artificially. Indeed, the reference image can be mod-



(a) *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: steps that are independent from the reference image (upper part of the figure), and steps that depend on the reference image (lower part of the figure).



(b) *Block Diagram for the Training Phase*. The features $\mathbf{f}_{ref}$ 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 reference image dependent steps.

Fig. 1: *Block diagram of the replica detection system*. The system is composed of two phases, namely training and testing.

ified using different operations, resulting in several replicas. In this works, the replicas are generated by the operations listed in Table I. 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, 420 replicas of the reference image are synthesized by using up to two nesting levels of compositions.

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

C. Training Metric

The F-score metric $F(\cdot)$ is used to assess the detection performance during the training phase.

TABLE I: *Training replicas generation.* Image operations and their parameters.

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
gray 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%

The F-score is defined as follows [13]:

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

where P is the 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).

Equation (1) can be rewritten as:

$$F_\rho\left(r_{fp}, r_{fn}\right) = \frac{(1 - r_{fp}) \cdot (1 - r_{fn})}{1 + \rho \cdot r_{fp} - r_{fn}}, \qquad (2)$$

where r_{fp} and r_{fn} are the false positive and false negative rates. $\rho = N/P$ gives the ratio between the number of negative and positive instances. In the rest of the document, we use the formulation given by (2). One drawback of this metric lies in the ratio ρ between the number of negative and positive instances. It has to be known beforehand.

D. Notations

Subscript 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 labeled $y_i = +1$. Otherwise it is called a *non-replica* and labeled $y_i = -1$.

We denote the α-th feature of an image as f_α. All features of an image can be held in a column vector denoted as $\mathbf{f}$.

III. Replica Detection System

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

A. Image Preprocessing

Before extracting features, an image is first cropped such that only 70% of its center region is kept. It introduces a 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 256×256 pixels), while keeping its original aspect ratio. It introduces a weak form of scale invariance and 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 equalized-Intensity space. More specifically, the logarithmic Hue, $H_{\log}$, is defined as follows [14]:

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

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 [14]. The Saturation, S, is the same as for classical HSI [15]:

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

By construction, the Saturation is invariant to changes in illumination. Finally, the equalized Illumination, I_{equ}, is given by

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

where $\mathrm{T}(\cdot)$ is the global histogram equalization operator [15]. The equalization permits to make the Intensity mostly invariant to changes of gamma and brightness as shown in Appendix .

B. 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: color, texture, shape, etc. For an extensive survey on general features extraction, refer to [16]. 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, color and gray-level statistics. They are similar to those used in [9]. The main differences are the added 24 gray-level features, and the absence of 'local' statistics. These features are found to give good results in image retrieval applications [9]. In total, we extract 162 features, as shown in Table II. They are detailed in the following subsections.

TABLE II: *Features overview.* List the types of used features and the number of extracted statistics.

name	♯ **features**
Gabor, squared coeff. mean	30
Gabor, squared coeff. std dev.	30
Color, histogram	10
Color, channel mean	24
Color, channel std dev.	24
Color, spatial distribution	20
Gray-level, histogram	8
Gray-level, spatial distribution	16
total	**162**

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 [17] on the equalized Illumination. More precisely, the used parameters are 0.75 for the upper center frequency, 0.05 for the lower center frequency, five scales and six orientations. Mean and standard deviation estimates of the squared coefficients are computed for each of the 30 subbands. It results in a total of 30 mean and 30 variance estimates.

2) Color Features: The color features are based on the modified HSI color space presented in Sec. III-A. Each pixel in the image is classified into one of ten *color classes* depending on its position in this space. The classes are the achromatic colors ($S = 0$) *black*, *gray* and *white*, and the chromatic colors ($S > 0$) *red*, *orange*, *yellow*, *green*, *cyan*, *blue* and *purple*. The equalized 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' color approach proposed in [9]. In this study, pink and brown are also considered, whereas in our case these two colors are classified as red or orange. Brown and pink have the same Hue as red or orange, but differ in the Intensity and/or Saturation channels. Operations such as saturation or intensity changes are common in image processing, and 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.

a) Color Classes Histogram: A histogram is computed, giving the proportion of each color class in the image. It is normalized such that it sums to one. It comprises 10 values.

b) Channel Statistics: Mean and variance estimates of the equalized Intensity channel are computed for each color class. Mean and variance estimates of Saturation and logarithmic Hue channels are also computed for each chromatic color class. It results in a total of 24 mean and 24 variance estimates.

c) Spatial Distribution Shape: The shape of the spatial distribution of each color class is computed. This is achieved by computing two shapes characteristics for each color class: spreadness and elongation [18], [19]. The first characteristic measures the compactness of the spatial distribution of a color class. The second one reflects how much the spatial distribution has more length than width. It results in 10 spreadness and 10 elongation measures.

3) Gray-Level Features: The gray-level features are based on the equalized 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 gray-level feature is important because the color features can be unsuited in some cases. For instance, it can happen when the reference or the test images are gray-level, or when conversion to gray-level is one of the considered operations in the replica detection system.

a) Gray-level Classes Histogram: A gray-level classes histogram is computed, giving the proportion of the eight intensity range in the image. It is normalized such that it sums to one. It comprises 8 values.

b) Spatial Distribution Shape: As for color, the shape of the spatial distribution of each gray-level class is computed. It results in 8 spreadness and 8 elongation measures.

C. Weighted Inter-Image Differences

Inter-image differences are computed in this steps. They are basically the difference between the statistics of the test image and those of the reference image.

The channels statistics and spatial distribution shape for color classes, and the spatial distribution shape for gray 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 penalized.

A 'weight' w_α is associated to each feature f_α. The weight gives the proportion of pixels taken into account to compute f_α. For the color features, the weights are the entries of the color histogram. Likewise, the weights corresponding to gray-level features are those of the gray-level histogram. For the remaining features, no weight is used.

We define the weighted inter-image difference δ_α as:

$$\delta_\alpha = f_\alpha - \bar{f}_\alpha + \hat{s}_\alpha \cdot \mathrm{sgn}\left(f_\alpha - \bar{f}_\alpha\right) \cdot \left(w_\alpha - \bar{w}_\alpha\right)^2, \quad (6)$$

where $\bar{f}_\alpha$ and $\bar{w}_\alpha$ are the α-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 sign function is such that $\mathrm{sgn}(0) = +1$.

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, $f_\alpha - \bar{f}_\alpha$ and $w_\alpha - \bar{w}_\alpha$ are small, resulting in small δ_α. 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_α and w_α are simultaneously close than only f_α or w_α. That is, (6) ensures that δ_α is, with higher probability, smaller for a replica than for a non-replica.

The parameters s_α are found through a training procedure. To achieve this, simple classifiers are considered. There are as many simple classifier than features α, and the decision functions are given by:

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

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

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

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

D. Normalization

The weighted inter-image difference are normalized using a statistical normalization method [20]. More precisely, let μ_α and σ_α be the mean and standard deviation estimates of the α-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 examples are not taken into account. The normalized inter-image difference $\tilde{\delta}_\alpha$ is then given by:

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

where δ_α 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 normalization is to ensure that the feature elements are commensurable. This is especially important for dimensionality reduction.

E. Dimensionality Reduction

The relatively large number of features prevents to directly use many classification techniques on the weighted inter-image differences. It would requires a prohibitively large number of training examples, and the optimization process would probably yield an overtrained decision function due to the 'curse of dimensionality' [21]. 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:

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

where the $d \times D$ matrix $\mathbf{W}_{D \to d}$ is computed using a slightly modified version of the method proposed in [22]. The method is based on independent component analysis (ICA) [23]. It adds the class information to the feature vector in order to elect

Require: $\{\mathbf{x}_i\}_i$, $\{y_i\}_i$, d
Ensure: $\mathbf{W}_{D\to d}$
  for each example i, set $\tilde{\mathbf{x}}_i = [\mathbf{x}_i^T \;\; y_i]^T$
  apply ICA to $\{\tilde{\mathbf{x}}_i\}_i \rightarrow$ separating matrix $\mathbf{W}^a$
  for each row α of $\mathbf{W}$, set $a_\alpha = \frac{1}{D+1}\sum_{\beta=1}^{D+1} |w_{\alpha\beta}|$
  set $\gamma = 0.9$ and $\Delta_\gamma = 0.001$
  repeat
    set $\mathbf{R} = \mathbf{W}$
    for each element $\alpha\beta$ of $\mathbf{R}$, set $r_{\alpha\beta} = 0$ if $|r_{\alpha\beta}| < \gamma \cdot a_\alpha$
    delete rows α of $\mathbf{R}$ for which $r_{\alpha(D+1)}\sum_{\beta=1}^{D} |r_{\alpha\beta}| = 0$
    set $\tilde{d}$ = "number of rows in $\mathbf{R}$"
    if $\tilde{d} = d$ **then**
      delete the $(D+1)$-th column of $\mathbf{R}$
      set $\mathbf{W}_{D\to d} = \mathbf{R}$
    else
      set $\gamma = \gamma + \mathrm{sgn}(\tilde{d} - d) \cdot \Delta_\gamma$
    end if
    if no convergence to d **then**
      set $\Delta_\gamma = \Delta_\gamma / 2$
    end if
  until $\tilde{d} = d$

  [a] $\mathbf{W}$ size is $(D+1) \times (D+1)$

Fig. 2: *Dimensionality Reduction Algorithm.* It is a modified version of [22] that enables to set *a priori* the number of dimension d.

the independent components best suited to the binary classification problem. The algorithm [22] is modified so that the number of dimensions can be set *a priori*. The method is briefly described in the Fig. 2.

F. Decision Function

The decision function needs to determine whether the vector $\mathbf{x}$ corresponds to a replica of the reference image. It 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 generalizes well to novel patterns. Many classification algorithms can be used for this purpose. In our previous works [11], [12], we showed that Support Vector Machine (SVM) yielded good performances for the replica detection problem.

The basic SVM [24], [25] is a binary classifier that separates two classes with an hyperplane. Furthermore, non-linear kernels allow to map patterns into a space where they can be better discriminated by a hyperplane.

1) Support Vector Machine: We use the ν-parametrization [24], [26] of the SVM, and a radial basis function as kernel. The dual constrained optimization problem is given in (11). In the dual form, the Lagrangian is maximized by optimizing the Lagrangian multipliers α_i.

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

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

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

The parameters of this classification technique are $\nu \in [0, 1]$ and $\sigma \in \mathbb{R}^+$. The parameter ν can be shown to be an upper bound on the fraction of training errors, and a lower bound on that of support vectors [24], [26]. The kernel parameter σ controls the complexity of the decision boundary. The constrained optimization problem given in (11) can be solved by means of standard quadratic programming techniques.

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

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

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 optimized α_i are equal to 0. Hence, only a relatively small fraction of the training patterns defines the decision function.

2) Determination of the Classification Parameters: In the ν-SVM, the kernel parameter σ and the parameter ν are to be determined. More precisely, they need to be set in order to minimize the *generalization error* is minimized, which is the error obtained when testing novel pattern with a trained decision function.

More precisely, we want to minimize the F-score $F(r_{fp}, r_{fn}, \rho)$ where r_{fp} is the generalization error for false positive (novel non-replicas classified as replicas), r_{fp} is the generalization error for false negative (novel replicas classified as non-replicas), and ρ 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, ρ remains *a priori*

unknown. Moreover, r_{fp} and r_{fn} are also unknown and need to be estimated.

Cross-validation is a popular technique for estimating generalization 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 generalization error. This method was shown to yield a good estimation of the generalization error [27].

In the following, we use a normalized version of the radial basis function kernel where σ in (12) is replaced by $\kappa \cdot \sigma$. The normalization constant κ 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 σ is larger than one with high probability.

While ν has an intuitive signification, it is not clear what should be its optimal value [26], [28]. It was shown that twice $\bar{R}$, a close upper bound on the expected optimal Bayes risk, is an asymptotically good estimate [28]. However, no such bound can be easily determined *a priori*.

In this work, good parameters for σ and ν are estimated in two steps: *coarse* and *fine* grid searches. In each step, a tenfold cross-validation is carried out for each feasible pairs (ν, σ). The pair for which the estimated F-score is the highest is then chosen. The tried pairs are the following:

- Coarse search: (σ, ν) for $\nu = 0.05 \cdot 2^k, k = -4, ..., 4$ and $\sigma = k, k = 1, \ldots, 10$.
- Fine search: (σ, ν) for $\nu = \nu_1 \cdot (1 + k/6), k = -2, \ldots, +2$ and $\sigma = \sigma_1 \cdot (1 + k), k = -2, \ldots, +2$. Here, ν_1 and σ_1 denote the value determined in the first step.

IV. Evaluation Methodology

A. Test Images

To simulate the performance of the proposed approach, we used the same image database as in [8][1]. 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 900×600, 678×435, or 640×480. They are mostly color images, except for about one thousand images that are gray-levels.

For training, we randomly selected 700 images in the database. 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 Sec. III.

The replica detectors are then tested on the remaining images in the database. This permits to estimate the false positive rate for each reference images. 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 the ones used in [8], [9]. They are implemented using the free software suite ImageMagick[2]. There are twelve categories, as shown thereafter. An example for each of them is depicted in Fig. 3.

- *Colorizing*. 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.
- *Color Depth*. Reduce the color palette to 256 colors;
- *Framing*. Add an outer frame of 10% the image size. Four images are produced with different frame color.
- *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%.

[1] http://www-2.cs.cmu.edu/~yke/retrieval/

[2] http://www.imagemagick.org

(a) Original

(b) Replicas

Fig. 3: *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).

B. Evaluation Metrics

In order to evaluate the performance of the proposed system, we measure the tradeoff between the *false positive* and *false negative* rates.

The Receiver Operating Characteristic (ROC) curve [13] is often used to represent the tradeoff between error types. In this study, we use a variant of the ROC curve called Detection Error Tradeoff (DET) curve [29]. In DET curve error rates are plotted on both axis. It means that both axis can 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 DET curve is computed using the values under the sign function in the right-hand side of (13). Every reference image detector produces a DET curve, which are synthesized in a single DET curve by using vertical averaging [13]. The maximal absolute precision that can be achieved on the vertically averaged DET curve is reported in Table III. 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 III: *Vertically averaged DET curve precision.* 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).

error type	**maximal absolute precision**
r_{fp}	$\pm\frac{1}{18,085} = \pm 5.5 \cdot 10^{-5}$
r_{fn}	$\pm\frac{1}{40 \cdot 200} = \pm 1.3 \cdot 10^{-4}$

V. 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 parameterization, unless stated otherwise.

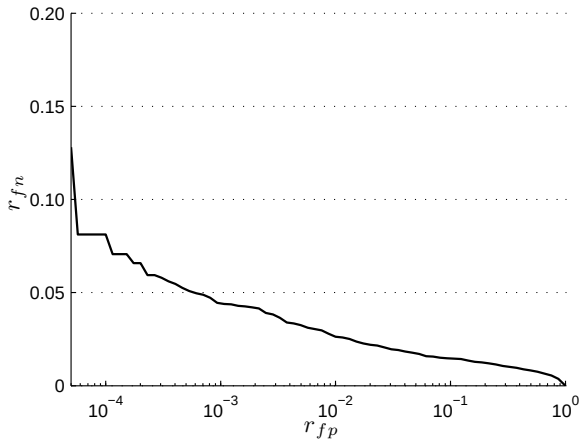
A. Influence of the F-Score Metric Parameterization

In this experiment, we explore the effect of possible parameterizations of the F-score metric. The value ρ 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 ρ is much larger than one because there are many more non-replicas than replicas.

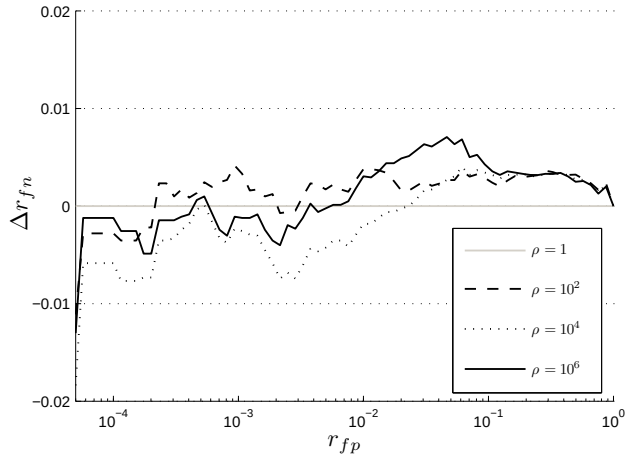
Figure 4a shows the vertically averaged DET curve for $\rho = 1$. Then, Fig. 4b depicts the vertical differences between the averaged DET curves for $\rho = \{10^2, 10^4, 10^6\}$ and that for $\rho = 1$. Globally, different values of ρ influence only slightly on the results, namely the difference is less than 1%. However, high ρ values favor classifiers with very low false positive rates while keeping reasonable false negative rates.

B. DET Curves Distribution

We now analyse the distribution of DET curves before vertical averaging. Figure 5 shows the false



(a) Average DET curve for $\rho = 1$



(b) Vertical differences to curve 4a

Fig. 4: *Influence of* ρ. Vertically averaged DET curves for a fixed number of dimensions $d = 50$, and different values of ρ.

negative rates histograms for a fixed false positive rate $r_{fp} = 5 \cdot 10^{-5}$ (leftmost value of the curves in Fig. 4) and two different values of ρ, namely 1 and 10^4. On the one hand, the two histograms show that about 50% of the individual classifiers have false positive rates below 5%. Actually, about 25% of the classifiers have no false negative at all. On the other hand, the two histograms show that a small number of classifiers (respectively 15 and 9) presents false positive rates above 50%. Moreover, this proportion can be seen to decrease when ρ increases. The corresponding reference images possess few colors, or are gray-level images. It shows that color related features are very powerful discriminating features, and that the lack of color variety complicates the replica detection task. The bad classifiers partici-

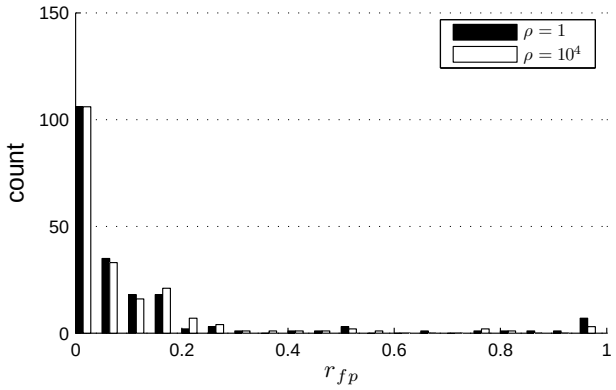


Fig. 5: *False negative histograms*. For a fixed false positive rate $r_{fp} = 5 \cdot 10^{-5}$ (leftmost value of the curves in Fig. 4), number of dimensions $d = 50$, and two different values of ρ.

pate heavily in increasing the average false positive rates. When their proportion diminishes, so does the average false positive error, explaining the results obtained in Fig. 4.

C. Gray Level Features

In this trial, the detection performance obtained by using gray-level features are compared to that when not using them. Figure 6 depicts the performance improvement brought by adding gray-level features. The performance gap augments as the false positive rate decreases. Note that gray-level images are present in both the reference images and the test images. Straightforwardly making use of gray-level features greatly improves the performance on these images. Moreover, it also increases the performance for color images. Indeed, gray-level features capture information that is missed by the color features, namely the global intensity shape.

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

D. Weighted Inter-Image Differences

In this experiment, we analyse the advantage of using weighted inter-image differences. Figure 7 illustrates the performance increase due to the weighted inter-image differences. The DET curve

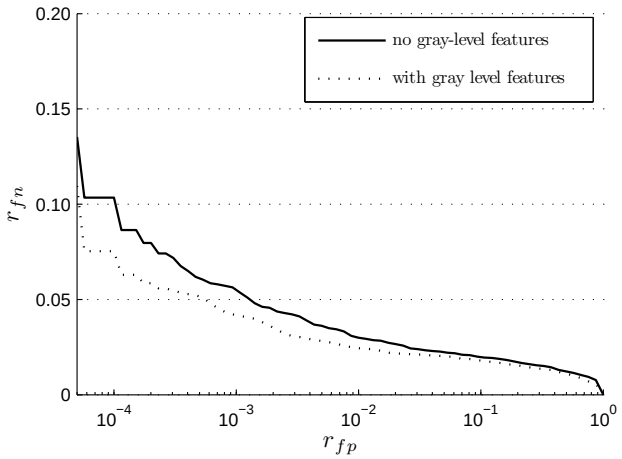


Fig. 6: *Influence of gray-level features*. Vertically averaged DET curves with, and without, using gray-level features.

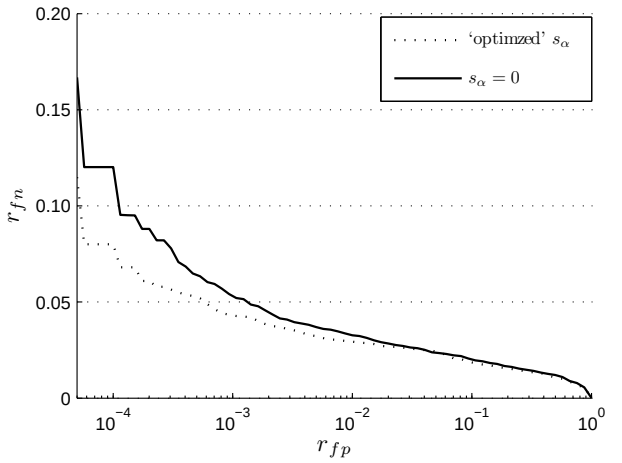


Fig. 7: *Influence of the weighted inter-image differences*. Vertically averaged DET curves for $s_\alpha = 0$ and optimized s_α.

with $s_\alpha = 0$ corresponds to the situation where the differences are not weighted. In this case, (6) becomes $\delta_\alpha = f_\alpha - \bar{f}_\alpha$. The other DET curve corresponds to the case where the s_α have been optimized as described in Sec. III-C. 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 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.

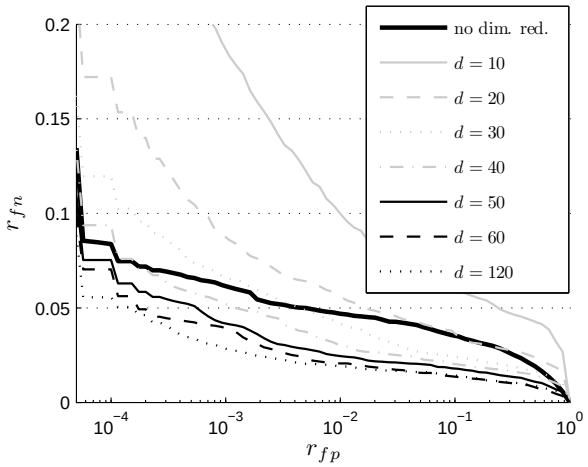


Fig. 8: *Influence of the dimensionality reduction*. Vertically averaged DET curves for different values of d.

E. Dimensionality Reduction Performance

We now explore the efficiency of the dimensionality reduction step. Figure 8 illustrates the performance obtained with different dimensions d: in general, the larger the number of dimensions, the better the performances. However, the gain is not very important when going over $d = 50$. The thick line represents the performances achieved without dimensionality reduction. In this case, the performance is mostly equivalent to that of $d = 40$ for low false positive rates ($r_{fp} < 10^{-4}$) 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 dimension increases [21], [30].

Another factor explaining the results obtained in Fig. 8 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 every features carry 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* [23]. In that case, giving equal weights to every features makes more sense and gives better results. In fact, recent works

TABLE IV: *Storage requirements estimation.* 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.

name	**size in bytes**
image features	$162 \cdot 2 = 324$
image weights	$162 \cdot 0 = 0$
norm. const.	$2 \cdot 162 \cdot 2 = 648$
dim. reduct. matrix	$50 \cdot 162 \cdot 2 = 16,200$
support vectors	$50 \cdot 85 \cdot 2 = 8,500$
total	25,672 = 25 kilobytes

TABLE V: *Average running time for testing.* The experiments were carried out on a PC with a 2.8 GHz processor and 1 Go of RAM.

	name	**time, s**
reference image indep.	preprocessing	0.2
	feature extraction	1.8^a
reference image dep.	weighted features	$48 \cdot 10^{-6}$
	normalization	$90 \cdot 10^{-6}$
	dim. reduction	$17 \cdot 10^{-6}$
	decision function	$50 \cdot 10^{-6}$

[a] 0.1 s when optimized as in [9].

have shown that using metrics that are learned from side-information can improve classification results [31], [32]. In the case of the replica detection problem, two side-informations are available: the class (replica or non-replica), and the relative distance to the reference for replicas (for example, a JPEG compressed image with $Q = 10$ is farther to the reference than one with $Q = 90$). Future research will study the improvements brought by using learned metric in the replica detection problem.

F. Efficiency

The replica detection method efficiency is now analysed in term 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 normalization 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 IV. On average, about 25 kilobytes are needed to store each description. In other words, one megabyte can held, on average, up to forty descriptions. This is a negligible amount of memory for today computers.

Another important aspect is that of computational complexity of the method. The proposed method require a training for each reference image. The training is computationally complex and take up to fifteen minutes 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 offline, its computational complexity is less critical.

The computational complexity of testing is estimated in Table V. Note that except for the SVM part, the method is implemented in Matlab without any optimization. This incurs longer running time. For instance, the feature extraction could be reduced to, at least, 0.1 seconds [9]. In the discussion that follows, we assume an optimized feature extraction step. The preprocessing and feature extraction steps are reference image independent, and take up about 0.3 seconds. The remaining steps are reference image dependent, and take up about $0.2 \cdot 10^{-3}$ seconds. When the reference image database contains less than $1,500$ 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 [8], [9].

G. Comparison with Existing Replica Detection Methods

Figure 9 compares the performance of the proposed replica detection system with state of the arts techniques in [8], [9]. The continuous line corresponds to the vertically averaged DET curve obtained with our system, using $d = 50$ and $\rho =$

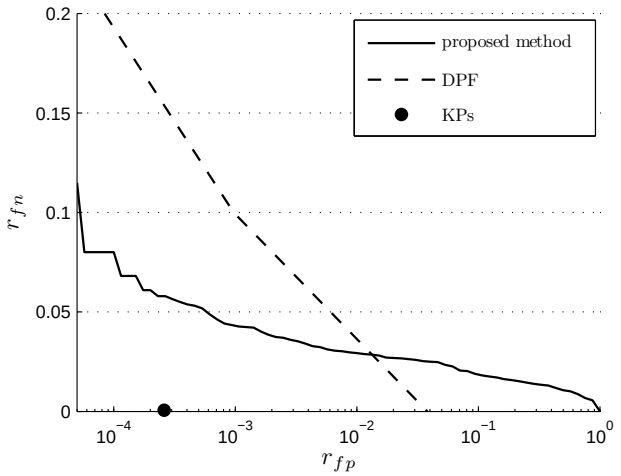


Fig. 9: *Comparison with other methods*. The proposed method is compared against other methods: KPs [8], and DPF [9].

10^4. The dashed line represents the performances of a replica detection method based on Perceptual Distance Function (DPF) [9]. The circle point indicates the performance of a replica detection system based on key points (KPs) [8]. For DPF the curve is obtained by inspecting the precision-recall curve reported in Fig. 5 of [9], and using $N = 20,000$ and $P = 40$. For KPs, the point is computed using the information reported in Table 1 and 2 of [8], and using $N = 18,722$ and $P = 40$.

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 $1 \cdot 10^{-4}$. On the one hand, our method outperforms that of DPF for false positive rates below 10^{-2}. Moreover, it should be 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 extend. The method KPs uses features (salient points, or key points [33]) invariant to this change but computationally more complex to extract. Indeed, the feature extraction time of KPs is between 1 and 10 seconds per image [8], [9]. This is between 10 to 100 times slower than that for the proposed method (when optimized as in [9]).

VI. Applications and Scenario

The proposed image replica detection system is suitable to detect copyright infringement or to identify illicit content. 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, normalization 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 wished to detect all possible replicas. The optimal trade-off is therefore application dependent.

VII. Conclusion

In this paper, we have described a technique to classify whether a test image is a replica of a given reference image. We performed experiments on a large database 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 $1 \cdot 10^{-4}$. 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 work consists in using additional side information to improve the fine-tuning of the replica detector.

Appendix
Invariance of Equalized Illumination to Gamma and Illumination Changes

Intensity and gamma changes are modeled as:

$$\mathrm{g}(r) = \alpha r^{\gamma}, \tag{14}$$

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:

$$\mathrm{p}_o(w) = \mathrm{h}'(w)\,\mathrm{p}_i(\mathrm{h}(w)), \tag{15}$$

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

The global histogram equalization maps the image with a given probability density $p(w)$ to an image with an uniform probability density. The mapping is given by [15]:

$$s = \mathrm{T}(r) = \int_0^r p(w)\,dw \tag{16}$$

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

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

$$s_o = \mathrm{T}(r_o) = \mathrm{T}(\mathrm{g}(r_i)) = \int_0^{\mathrm{g}(r_i)} \mathrm{p}_o(w)\,dw \tag{17}$$

$$= \int_0^{\mathrm{g}(r_i)} \mathrm{h}'(w)\,\mathrm{p}_i(\mathrm{h}(w))\,dw \tag{18}$$

$$= \int_0^{\mathrm{h}(\mathrm{g}(r_i))=r_i} \mathrm{h}'\left(\mathrm{g}(v)\right)\mathrm{p}_i\left(\mathrm{h}(\mathrm{g}(v))\right)\mathrm{g}'(v)\,dv \tag{19}$$

$$= \int_0^{r_i} \mathrm{h}'\left(\mathrm{g}(v)\right)\mathrm{p}_i(v)\,\mathrm{g}'(v)\,dv \tag{20}$$

$$= \int_0^{r_i} \frac{d}{dv}\left[\underbrace{\mathrm{h}\left(\mathrm{g}(v)\right)}_{=v}\right]\mathrm{p}_i(v)\,dv \tag{21}$$

$$= \int_0^{r_i} \mathrm{p}_i(v)\,dv = \mathrm{T}(r_i) = s_i. \tag{22}$$

An image and its versions processed by (14) are mapped to the same equalized image by (16). The only fact used above is that an inverse exists for the function $\mathrm{g}(\cdot)$. Therefore the above results can be generalized to any reversible transformation of the image.

Note that it cannot be proved in general that the discrete version of the histogram equalization produces the discrete equivalent of a uniform probability function [15]. However, in practice, discrete histogram equalization yields images that are mostly invariant to the gamma and illumination changes.

Acknowledgements

The first author is partly supported by the Swiss National Science Foundation — "Multimedia Security", grant number 200021-1018411. The work was partially developed within VISNET[3], a European Network of Excellence, funded under the European Commission IST FP6 programme. Also, the authors would like to thanks the creators of the SVM library libsvm [34], and those of the image processing software suite ImageMagick.

References

[1] ISO/IEC JTC1/SC29/WG11 WG11N4358, "Final Draft Int'l Standard 15938-3 Information Technology - Multimedia Content Description Interface - Part 3 Visual," July 2001.

[3] http://www.visnet-noe.org

[2] Martínez, J.M. and Koenen, R. and Pereira, F., "MPEG-7: The Generic Multimedia Content Description Standard," *IEEE Multimedia*, vol. 9, no. 2, pp. 78–87, April 2002.

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

[4] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proc. of the IEEE*, vol. 87, no. 7, pp. 1079 – 1107, July 1999.

[5] F. Lefèbvre, B. Macq, and J.-D. Legat, "Rash: Radon soft hash algorithm," in *EURASIP European Signal Processing Conf.*, France, September 2002.

[6] J. Seo, J. Haitsma, T. Kalker, and C. Yoo, "Affine transform resilient image fingerprinting," in *IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing*, Hong Kong, April 2003.

[7] R. Venkatesan, S.-M. Koon, M.-H. Jakubowski, and P. Moulin, "Robust image hashing," in *IEEE Int'l Conf. on Image Processing*, Vancouver, September 2000.

[8] Y. Ke, R. Sukthankar, and L. Huston, "An Efficient Parts-Based Near-Duplicate and Sub-Image Retrieval System," in *ACM Int'l Conf. on Multimedia*, 2004, pp. 869–876.

[9] A. Qamra, Y. Meng, and E. Y. Chang, "Enhanced Perceptual Distance Functions and Indexing for Image Replica Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 379–391, March 2005.

[10] Y. Maret, G. Molina, and T. Ebrahimi, "Images Identification Based on Equivalence Classes," in *Proc. Workshop on Image Analysis for Multimedia Interactive Services*, April 2005.

[11] ——, "Identification of Image Variations based on Equivalence Classes," in *Proc. SPIE Visual Communications and Image Processing*. SPIE, July 2005.

[12] Y. Maret, F. Dufaux, and T. Ebrahimi, "Image Replica Detection based on Support Vector Classifier," in *Proc. SPIE Applications of Digital Image Processing XXVIII*, August 2005.

[13] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Data Mining Researchers," 2003, technical Report HPL-2003-4.

[14] G. Finlayson and G. Schaefer, "Hue that is Invariant to Brightness and Gamma," in *Proc. British Machine Vision Conf.*, 2001.

[15] R. Gonzalez and R. Woods, *Digital Image Processing, 2/E*. Prentice-Hall, 2002.

[16] Y. Rui, T. Huang, and S. Chang, "Image retrieval: current techniques, promising directions and open issues," *J. of Visual Communication and Image Representation*, vol. 10, no. 4, pp. 39–62, April 1999.

[17] B. S. Manjunath and W. Y. Ma, "Texture Features for Browsing and Retrieval of Image Data," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, August 1996.

[18] M.-K. Hu, "Visual Pattern Recognition by Moment Invariants," *IEEE Trans. on Information Theory*, pp. 179–187, 1962.

[19] J.-L. Leu, "Computing a Shape's Moments from its Boundary," *Pattern Recognition*, vol. 24, no. 10, pp. 949–957, 1991.

[20] J. Smith and A. Natsev, "Spatial and Feature Normalization for Content Based Retrieval," in *IEEE Int'l Conf. Multimedia and Expositions*, vol. 1, 2002, pp. 193–196.

[21] D. L. Donoho, "High-dimensional data analysis: The curses and blessings of dimensionality," August 1998. [Online]. Available: http://www-stat.stanford.edu/~donoho/

[22] N. Kwak, C.-H. Choi, and C.-Y. Choi, "Feature extraction using ica," in *Proc. Int'l Conf. on Artificial Neural Networks*, August 2001, pp. 568–576.

[23] A. Hyvarinen and E. Oja, "Independent component analysis: Algorithms and applications," *Neural Networks*, vol. 13, no. 4-5, pp. 411–430, 2000.

[24] B. Schölkopf, A. Smola, R. Williamson, and P. L. Bartlett, "New support vector algorithms," *Neural Networks*, vol. 22, pp. 1083–1121, 2000.

[25] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[26] P.-H. Chen, C.-J. Lin, and B. Schölkopf, "A tutorial on nu-support vector machines," *Applied Stochastic Models in Business and Industry*, vol. 21, pp. 111–136, 2005.

[27] K. Duan, S. S. Keerthi, and A. N. Poo, "Evaluation of simple performance measures for tuning SVM hyperparameters," *Neurocomputing*, vol. 51, pp. 41–59, 2003.

[28] I. Steinwart, "On the Optimal Parameter Choice for ν-Support Vector Machines," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1274–1284, October 2003.

[29] A. Martin, G. Doggintgton, T. Kamm, M. Ordowski, and M. Przybocki, "The DET Curve in Assessment of Detection Task Performance," in *Proc. Eurospeech*, Greece, 1997, pp. 1895—1898.

[30] R. Duda, P. Hart, and D. Stork, *Pattern Classification*. Wiley-Interscience, 2000.

[31] E. Xing, A. Ng, M. Jordan, and S. Russell, "Distance Metric Learning with Application to Clustering with Side-Information," in *Proc. Advances in Neural Information Processing Systems*, 2003, pp. 505–512.

[32] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall, "Learning Distance Functions using Equivalence Relations," in *Proc. Int'l Conf. on Machine Learning*, Washington DC, August 2003.

[33] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int'l J. of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[34] C. Chang and C. Lin, *LIBSVM: a library for support vector machines*, 2001. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm