REDUCED COMPLEXITY REPLICA DETECTION SYSTEMS USING BINARY CLASSIFIERS AND R-TREES

Y. Maret[†], *S. Nikolopoulos*[‡], *F. Dufaux*[†], *C. Cotsaces*[‡], *T. Ebrahimi*[†], *N. Nikolaidis*[‡]

[†]Ecole Polytechnique Fédérale de Lausanne(EPFL)
Institut de Traitement des Signaux
CH-1015 Lausanne, Switzerland

[‡]Department of Informatics
Aristotle University of Thessaloniki
Box 451, Thessaloniki 54124, Greece

ABSTRACT

Replica detection is an important prerequisite for the discovery of copyright infringement and detection of illicit content. For this purpose, content-based image protection can be an efficient alternative to watermarking. Rather than imperceptibly embedding a signal, content-based systems rely on image similarity. Certain content-based systems use binary classifiers to detect replicas, each classifier being fine-tuned to a particular original. However, since a suspect image has to be tested against every original, such a comparison becomes computationally prohibitive as the number of original images grows. In this paper, we propose an indexing method to efficiently prune the number of comparisons with the originals in the database. For this purpose, a multidimensional indexing structure, namely R-trees, is incorporated to rapidly select the most likely originals. Experimental results showed that up to 97% of the database can be discarded before applying the binary classifiers.

1. INTRODUCTION

The recent progresses in multimedia technologies and the advent of the World Wide Web (Web) have permitted to copy and distribute digital content at negligible costs. Unfortunately, many valuable digital images are now illegally redistributed. Consequently many foundations, companies, or educational centres are reluctant to publish their contents on the Web. In this context it becomes not only important to protect the content but also to detect copyright infringements.

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, adjustments of contrast, saturation or colors, and geometric manipulations.

Numerous systems address the replica detection problem, most of them using watermarking techniques. Watermarking is the technique of imperceptibly embedding information within the content of the original image. The hidden mark is invisible to human eye and can only be detected by making use of a private key (known only by the content owner). An overview of watermarking techniques can be found in [1].

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 (`http://www.visnet-noe.org`), a European Network of Excellence, funded under the European Commission IST FP6 programme.

Recently, the scientific community started to investigate replica detection from a content-based perspective [2,3]. Indeed, the problem can be reformulated in terms of assessing the similarity between a suspect image and an original image, or more generally in terms of querying a database of original images with a suspect image. Consequently, there is no necessity of embedding marks in images. Nevertheless, this new approach also raises new challenges. For instance, images should be represented by features that are unique for each image and robust to image manipulations.

In the proposed technique, features are first extracted from the images. The feature vector dimensionality is then reduced through a linear transformation. Subsequently, the reduced features are used to index the database of originals by means of an R-tree. When testing a suspect image, the R-tree returns a list of images that are candidates for being the original of the suspect images. For each candidate, a binary classifier, defined in [4], is used to determine the probability that the suspect image is one of its replica. Finally, the candidate with the highest probability is indeed the original if the probability is above a certain threshold. In this preliminary work, we test separately the two main components (R-tree and binary classifiers) of the proposed system.

2. PROBLEM DEFINITION

The image replica detection problem can be defined in, at least, two ways. In the first case a query of the type "*Is this suspect image a replica of any of the elements present in a database of originals?*" is issued. Conversely, in the second case a query of the type "*Are there any replicas of this original image in a database containing suspect images?*" is issued.

We argue that the first approach is better suited to the detection of replicas on the Web. For the moment, let us assume that a brute force approach, i.e., a sequential search, is carried out. In this case, none of the definitions are preferable. Of course this approach is hardly feasible for real world applications. Therefore an indexing mechanism has to be used in order to avoid a sequential search. In such case, indexing the database of originals is easier. Indeed, the second definition of the problem would basically require indexing every image contained in the Web. Apart from the tremendous number of images to index in the first place, the database need to be kept up to date. On the other hand, copyrighted originals represent only a small fraction of all images. Therefore, maintaining a database containing originals is much easier than indexing the

Web. Moreover, suspect images can be simply found by crawling the Web and maintaining a list containing the recently visited sites.

Using the first definition, the replica detection problem can be reduced to a query $\mathrm{Q}_{\mathcal{O}}(\cdot)$ to a database $\mathcal{O}$ of originals:

$$\mathcal{R} = \mathrm{Q}_{\mathcal{O}}(\mathbf{I}_s), \tag{1}$$

where $\mathbf{I}_s$ denotes a suspect image, and $\mathcal{R}$ is a binary answer (yes/no) to the question "*Is $\mathbf{I}_s$ a replica of any orignal?*", which in the case of a positive answer is accompanied by the corresponding original. Lets assume that no two originals are the same, and furthermore that a replica cannot be derived from more than a single original. Then, $\mathcal{R}$ is either an empty set or a singleton corresponding to the original of $\mathbf{I}_s$. The empty set corresponds to the answer "*$\mathbf{I}_s$ is not a replica of any orignal*".

Our definition of replica detection bears similarity with that of image retrieval systems. However, there are important differences. Firstly, image retrieval systems return a group of images that are similar to the image given as query, whereas replica detection systems shall return at most a single image. Secondly, image retrieval systems use a notion of similarity that differs from that of replica detection systems. Indeed, similarity in image retrieval systems is often used at a semantic level, for example two sunset images are similar for such a system. On the other hand, two images are similar for replica detection if and only if one of them derives from the other through a series of manipulations that do not alter the content of the image.

3. REPLICA DETECTION SYSTEM

The main idea behind the proposed replica detection system is to use a classifier to determine whether the suspect image is a replica of an original contained in a database. Although the number of originals is quite small compared to that of all images on the Web, it can still be fairly large depending on the application (e.g. in the thousands or even millions). When using a set of binary classifiers, as in [4], the entire database has to be sequentially scanned, which is cumbersome as the number of originals grows. Therefore, we propose to use a preprocessing step based on an indexing structure where, given a suspect image, the most likely original images are efficiently selected. More precisely, a query $\mathrm{Q}_{\mathcal{I}(\mathcal{O})}(\cdot)$ is issued to the structure $\mathcal{I}(\mathcal{O})$ that indexes the database $\mathcal{O}$ of originals:

$$\mathcal{R}' = \mathrm{Q}_{\mathcal{I}(\mathcal{O})}(\mathbf{I}_s), \tag{2}$$

where $\mathbf{I}_s$ denotes a suspect image, and the answer $\mathcal{R}'$ contains the images that are candidates for being the original that corresponds to $\mathbf{I}_s$. Ideally, $\mathcal{R}'$ contains few candidates and includes the answer $\mathcal{R}$ given in (1).

In order to select a single image from the list of candidates provided by (2), binary classifiers are used. That is, a classifier is specifically trained for a certain original contained in the database. Each classifier outputs the probability that the suspect image is a replica of the corresponding original image. Subsequently, the answer $\mathcal{R}$ is given by

$$\mathcal{R} = \begin{cases} \mathbf{I}_{o_M} & \text{if } p_M > T \\ \varnothing & \text{otherwise} \end{cases}, \tag{3}$$

where p_M denotes the largest probability returned by the classifiers corresponding to the originals in $\mathcal{R}'$, $\mathbf{I}_{o_M}$ is the corresponding original, and T is a threshold that provides with the option to decide that none of the images given in $\mathcal{R}'$ correspond to $\mathbf{I}_s$.

Table 1: Features overview.

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**

In the following subsections, some steps of the method are explained in more details. First, we list the features used to represent each image. Then, the dimensionality of the feature vector is reduced for the purpose of efficient indexing. Third, the construction of the index structure is described. Finally, we give a short overview of the binary classifiers defined in [4].

3.1. Chosen Features

In order to compare the similarity between two images, representative 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. We use the same features as in [4]. Note that the chosen features, when combined together, exhibit a certain robustness against image manipulations. More precisely, we extract the 162 global features summarized in Table 1. For each image, the extracted features are placed in a 162-dimensional vector $\mathbf{F}$.

3.2. Dimensionality Reduction for Indexing

Many features are needed in order to have enough information to discriminate between replicas and non-replicas. Nonetheless, 162 features are too many for building an efficient indexing structure. For this reason, the dimensionality of the feature vector is reduced to d by using ICA-FX [5]. This method is based on independent component analysis and is well suited to binary classification problems. It makes use of examples from both classes. In our case, the replicas examples are generated by modifying the originals using a set of manipulations, and the non-replica examples are images chosen at random in a database (for more details refer to Sec. 4). More precisely, the features extracted from the examples and the originals are used to compute the following set of differences:

$$\left\{\mathbf{F}_{(r,o)} - \mathbf{F}_o\right\}_{(r,o)} \bigcup \left\{\mathbf{F}_f - \mathbf{F}_o\right\}_{(f,o)} , \tag{4}$$

where $\mathbf{F}_{(r,o)}$, $\mathbf{F}_f$ and $\mathbf{F}_o$ are features corresponding to replicas, non-replicas and to originals, respectively. The samples on the right hand side of the union are labeled $+1$, while those on the left hand side are labeled -1. Then, the reduction matrix $\mathbf{W}_d$ is computed by applying ICA-FX on these samples and the corresponding labels. Finally, the features $\mathbf{f}_o$ used for indexing are given by $\mathbf{f}_o = \mathbf{W}_d \cdot \mathbf{F}_o$. In the experiments, several values for the dimension d of the feature set are tested.

3.3. R-Tree based Indexing

We implemented an indexing structure based on R-trees [6]. An R-tree is a dynamic structure for efficiently indexing high-dimensional spaces. An R-tree is a height-balanced tree with index records in its leaf nodes (containing pointers to data objects). Originally, R-trees were created to index spatial objects using their bounding boxes (BBs). Therefore, the R-tree structure is constructed so as to efficiently answer the point-based query "*Return all records with BBs including the point* $\mathbf{p}$", and the radius-based query "*Return all records with BBs intersecting the sphere centered in* $\mathbf{p}$ *and having radius* r".

The fact that the extracted features present some robustness against image manipulations means that the features of a replica should be localized around those of the corresponding original image. Therefore, an R-tree, optimized for replica detection, can be constructed by computing a "bounding box" for each original image in the database $\mathcal{O}$. The choice of these bounding boxes is critical for the performance of the R-tree. Indeed, if the BBs are too large many of them overlap, resulting in a large number of images in the answer $\mathcal{R}'$. On the other hand, if the BBs are too small a replica can fall outside the BB corresponding to its original, and the original is not included in the answer $\mathcal{R}'$.

In order to construct the bounding box associated with an original image, we generate replica examples by making use of a set of image manipulations. More precisely, the center of the bounding box is given by the feature vector $\mathbf{f}_o$ of the original, and the side length s_α for the dimension α is given by:

$$s_\alpha = 2 \cdot \max_r(\left|f_{(r,\alpha)} - f_{(o,\alpha)}\right|), \qquad (5)$$

where $f_{r,\alpha}$ corresponds to the α-th feature of the r-th replica example, and $f_{o,\alpha}$ corresponds to the α-th feature of the original.

The feature vector of a replica obtained by a manipulation less severe than those used to build the R-tree is contained in the bounding box corresponding to its original. Conversely, the feature vector of a replica generated by a more severe manipulation falls outside the bounding box corresponding to its original. Nonetheless, it can still be retrieved by making use of a radius-based query.

3.4. Binary Classifiers

We use the binary classifier described in [4]. In that work, each classifier is specifically trained for a particular original contained in the database. Each classifier outputs the probability that the suspect image is a replica of the corresponding original image. The main idea behind using binary classifiers is to fine-tune each classifier to the corresponding original.

We now briefly describe the different steps composing a binary classifier. For more details, refer to [4]. A binary classifier consists of the four steps outlined thereafter. In the *Weighted Inter-image Differences* step, the features of the suspect image are subtracted from those of the original image, and 'incommensurable' features are penalized. For example, statistics about yellow pixels are incommensurable when the suspect and original images contain very different proportions of yellow pixels. In the *Statistical Normalization* step the inter-image differences are statistically normalized. In other words, the same importance is given to each feature, independently of their value range. In the *Dimensionality Reduction* step the feature dimensionality is reduced. Less training examples are needed, and only feature mixtures relevant to the replica detection task are kept. Finally, in the *Decision Function* step, a decision function is used to determine if the test image is a replica of the reference image. This decision function is based on Support Vector Machine.

Table 2: Replicas used for testing and training.

Categories	**♯ replicas**
Colorizing	4
Contrast changes	2
Cropping	4
Despeckling	1
Downsampling[a]	6
Flipping	1
Color depth red.	1
Outer frame	4
Rotation	3
Scaling	6
Saturation changes	4
Intensity changes	4
total	**40**

[a]without antialiasing filter

(a) Set Q [2, 3].

Categories	**♯ replicas**
Median filtering	3
Gaussian filtering	1
JPEG comp.	12
Shearing	6
Cropping	9
Flipping	1
Scaling	6
Line/row removal	5
Random bending	1
Aspect ratio	8
Rotation	16
Rotation/scaling	16
Linear transform	3
FMLR	3
total	**88**

(b) Set S, StirMark [7].

4. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed approach, we used the same image database as in [2]. It contains $18,785$ photographs including (but not limited to) landscapes, animals, constructions, and people. The image sizes and aspect ratios are diverse, for example 900×600, 678×435, or 640×480. They are mostly color images, except for about one thousand images that are gray-levels. 200 images are randomly chosen to be the original images. The remaining pictures are either used for training or for testing.

The dimensionality reduction matrix $\mathbf{W}_d$ is built by using 250 randomly picked images. 50 of them are used as originals, and the remaining as non-replicas examples. The replicas are generated using the manipulations proposed in [3] and outlined in Table 2(a). The originals are different than the ones picked for testing the R-tree and the binary classifiers so as to ascertain that the dimensionality reduction step generalizes well to novel patterns. The R-tree is constructed for the 200 originals. The bounding boxes are computed using (5) and the replicas generated by the manipulations listed in Table 2(a). We found that this training set gives a good compromise for the size of the bounding boxes. Moreover, a binary replica detector is trained for each original as in [4] (500 additional images are randomly chosen from the database to serve as non-replicas examples).

Two set of replicas are used to assess the performance of the R-tree, as well that of the binary classifiers. One set, denoted Q, is generated by the transforms listed in Table 2(a). The second set, denoted S, is generated with the well known watermarking benchmark tool StirMark 3.1 [7]. The corresponding manipulations are listed in Table 2(b).

4.1. R-tree Performance

The R-tree performance is estimated by using replicas as queries. For the test set Q point-based queries were used. The results are shown in Table 3 for different number of dimensions. As expected, the correct original is always a member of the candidates. Moreover, the number of candidates is quite low. For example, when 20 dimensions are used for the features indexing the database, only 3% of the database is returned. Since the image manipulations of StirMark are quite severe, we used radius-based queries for testing

Table 3: R-tree miss-rates for the test set Q.

number of dimension d	10	15	20	25
miss rate of the R-tree	0	0	0	0
fraction of the database returned	0.08	0.04	0.03	0.02

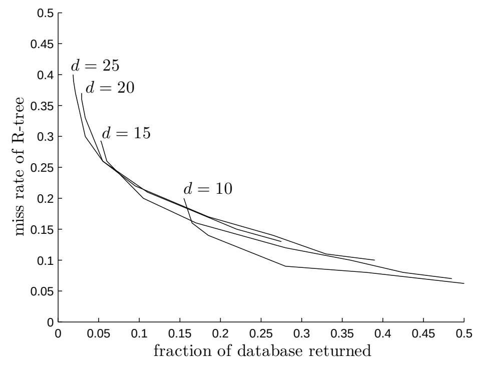

Fig. 1: R-tree miss-rates for the test set S.

the replicas generated by S. The corresponding results are shown in Fig. 1. The results show that for a miss-rate of 10% only 25% of the database is returned. Moreover, using more dimensions for indexing permits more pruning but at the price of an increasing miss-rate.

4.2. Binary Classifiers Performance

The tradeoff between false negative and false positive rates of the binary classifiers is now explored. A false positive occurs when an image that is not a replica of the original corresponding to the binary classifier is classified as such. Conversely, a false negative happens when a replica of the original is classified as a non-replica. In this study, we use a variant of the ROC curve called Detection Error Tradeoff [8] (DET) curve in order to asses the tradeoff between false positive and negative rates. Each binary classifier produces a DET curve. All curves are subsequently synthesized in a single DET curve by using vertical averaging [9]. The negative examples are provided by the remaining $17,835$ images.

The DET curves for both the test sets Q and S are given in Fig. 2. The proposed system performs better with the test set Q than for StirMark. For example, for a fixed false positive rate of 10^{-3} the average false negative rate is 4% for the test set Q and 13% for StirMark. Moreover, in the case of StirMark the miss-rate of R-tree has also to be taken into account. For example, for a miss rate of 10% and for a fixed false positive rate of 10^{-3}, the complete system false negative rate increases up to $1 - 0.9 \cdot 0.87 = 0.21$. This is a worst case result because a replica missed by the binary classifier might already have been omitted by the R-tree.

5. CONCLUSION

In this work, we give two different definitions of the replica detection problem. In the first definition a database of original images is queried with suspect images. Conversely, another definition is to query a database of suspect images with original images. The method introduced in this paper fits the first definition. In this

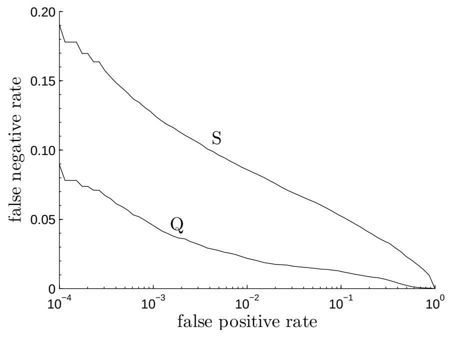

Fig. 2: Performance for the test sets Q and S.

case, the suspect image has to be tested against every orignal contained in the database, which can be cumbersome as the number of originals grows. In this paper we showed that this drawback can be overcame by making use of an indexing structure such as R-tree. The R-tree is constructed by taking into account the dispersion of the replica set associated to each original. The experiments showed that, in some cases, up to 97% of the database images can be pruned at the expense of an increased miss-rate. Future work includes the full integration and testing of the binary classifiers with the R-tree structure.

6. REFERENCES

[1] Cox, I. and Miller, M. and Bloom, J., *Digital Watermarking: Principles & Practice*, Morgan Kaufmann, 2001.

[2] 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.

[3] 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*, 2005.

[4] Y. Maret, F. Dufaux, and T. Ebrahimi, "Image Replica Detection based on Binary Support Vector Classifier," Tech. Rep. 2005-27, EPFL-ITS, 2005.

[5] N. Kwak and C.-H. Choi, "Feature extraction based on ica for binary classification problems," *IEEE Trans. on Knowledge and Data Engineering*, 2003.

[6] A. Guttman, "R-trees: a dynamic index structure for spatial searching," in *Proc. ACM Int'l Conf. on Management of Data*, 1984.

[7] F. A. P. Petitcolas and M. Kutter, "Fair Evaluation Methods for Image Watermarking Systems," *J. of Electronic Imaging*, 2001.

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

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