This paper is to be published in the proceedings of the IEEE International Conference on Image Processing 2006

A NOVEL REPLICA DETECTION SYSTEM USING BINARY CLASSIFIERS, R-TREES, AND PCA

Y. Maret[†], *S. Nikolopoulos*[‡], *F. Dufaux*[†], *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 a prerequisite for the discovery of copyright infringement and detection of illicit content. For this purpose, content-based systems 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 adaptive classifiers to detect replicas. In such systems, a suspect image is tested against every original, which can become computationally prohibitive as the number of original images grows. In this paper, we propose using R-tree indexing to decrease the necessary number of comparisons and rapidly select the most likely originals. Experimental results show that the proposed system performs very satisfactorily and that up to 99.3% of the originals can be discarded before applying the binary classifiers.

Index Terms— Copyright protection, Image databases, Image analysis, Image classification, Indexes

1. INTRODUCTION

The recent progress in multimedia technologies and the advent of the World Wide Web (Web) have permitted to process and distribute digital content at negligible costs. Unfortunately, many valuable digital images are now illegally redistributed. In this context, both content protection and detection of copyright infringements becomes important. In this paper, we propose a system to detect image replicas. By the term replica, we refer not only to a bit exact copy of a given original image, but also to modified versions of the image after certain 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 original image obtained after common image processing manipulations such as compression, filtering, adjustments of contrast, or geometric manipulations.

Numerous systems address the replica detection problem. Most of them use watermarking techniques to imperceptibly embed ownership information within the content of the original image. Recently, the scientific community started to investigate replica detection from a content-based perspective [1, 7]. Indeed, the problem can be reformulated in terms of assessing the similarity between a suspect image and an original image. Consequently, there is no necessity to embed any marks in the original 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.

The technique proposed in this paper is similarity-based and composed of the following steps. Features are first extracted from the images. The feature vector dimensionality is then reduced through Principal Component Analysis (PCA). Subsequently, the reduced features are used to index the database of originals by means of an R-tree [2]. When testing if a suspect image is among those stored in the database, the R-tree returns a list of images that are candidates for being the original of the suspect image. For each candidate, a binary classifier, proposed in [3], is used to determine the probability that the suspect image is a replica of this image. Finally, the candidate with the highest probability is deemed as the original if the probability is above a certain threshold.

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. If we assume that an indexing mechanism is used, it is much easier to index the database of originals as the second definition would essentially require indexing every image contained in the Web. Apart from the tremendous number of images to index in this case, the database need to be kept up to date. On the other hand, copyrighted originals represent only a relatively small fraction of all images, and thus maintaining a database of originals is much easier.

Our definition of replica detection bears certain similarities with that of content-based 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. Indeed, the answer to the query associated with the first definition of the replica detection system presented above can be either an empty set, corresponding to the answer "*the suspect image is not a replica of any original*", or a singleton corresponding to the original of the suspect 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 understood at a semantic level, for example two sunset images are similar for such a system. On the other hand, two images are similar for a replica detection system 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.

The first author is partly supported by the Swiss National Science Foundation, under grant number 200021-1018411. S. Nikolopoulos and N. Nikolaidis have been partially supported by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT.

3. REPLICA DETECTION SYSTEM

The main idea behind the proposed replica detection system is to use a binary classifier to determine whether the suspect image is a replica of an image contained in a database of originals. 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 (for example in the thousands or even millions). When using a set of binary classifiers, each being able to detect whether a suspect image is a replica of a specific image in the database, the entire database has to be sequentially scanned, which becomes quickly 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. We denote the set of likely originals, or candidates, $\mathcal{C}$. Ideally, $\mathcal{C}$ contains few elements and, includes the correct original if the suspect image is indeed a replica of one of the images in the database.

In order to select a single image from the set of candidates $\mathcal{C}$, binary classifiers are used as in [3]. 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 $\mathcal{R}$ of the system is given by:

$$\mathcal{R} = \begin{cases} \mathbf{I}_{o_M} & \text{if } p_M > T \\ \varnothing & \text{otherwise} \end{cases}, \qquad (1)$$

where p_M denotes the largest probability returned by the classifiers corresponding to the originals in $\mathcal{C}$, $\mathbf{I}_{o_M}$ is the corresponding original, and T is a threshold that provides to the system the option to decide that none of the images given in $\mathcal{C}$ correspond to the suspect image $\mathbf{I}_s$.

In the following subsections, certain steps of the method are explained in more detail. First, we list the features used to represent each image. Then, the method used for the dimensionality reduction of the feature vector for the purpose of efficient indexing is explained. Third, the construction of the indexing structure is described. Finally, we give a short overview of the binary classifiers defined in [3].

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. In this work we use the same features as in [3], which, when combined together, exhibit a certain degree of robustness against image manipulations. More precisely, the features used in this work are of three types: texture, colour and grey-level statistics. In total, 162 global features, summarised in Table 1, are extracted and then placed in a vector $\mathbf{f}$. More precisely, the texture features are composed of the first and second order statistics of each subband of the Gabor transform of the image. The colour features are based on a partition of the HSI colour space. Each pixel in the image is classified into one of ten *colour classes*[1] depending on its position in this space, and statistics are computed for each class. Finally, the grey-level features are based on the Intensity channel of the HSI model. The dynamic range of the image is linearly partitioned into eight bins corresponding to as many classes, and as for colour, statistics are computed for each class.

[1] black, grey, white, red, orange, yellow, green, cyan, blue, and purple.

Table 1: Features overview.

name	**♯ features**
Gabor, squared coefficient average	30
Gabor, squared coefficient standard deviation	30
Colour, histogram	10
Colour, channel average	24
Colour, channel standard deviation	24
Colour, spatial distribution	20
Grey-level, histogram	8
Grey-level, spatial distribution	16
total	**162**

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 making use of PCA. The PCA algorithm is applied to a training set containing the features of replicas of original images (see Sec. 4), and results in a dimensionality reduction matrix $\mathbf{W}_d$. Then the reduced features are given by $\mathbf{f}_i = \mathbf{W}_d \cdot \mathbf{f}$.

We found out that PCA gives better results than ICA-FX[2] used in [5] for this purpose. Indeed, if all remaining parameters are kept the same, an R-tree built on features given by PCA returns, on average, two to ten times less candidates than one constructed using features derived by ICA-FX. A possible reasoning for this is the following. A 'good' projection must separate sufficiently well the clusters of feature vectors representing the replicas of each original in the database. With ICA-FX this separation works well for the originals used for training, but no guarantee is provided for other images, since the algorithm seeks to maximise the separability of those classes used for training. On the other hand, PCA reduces the dimensionality of the feature space by finding the directions along which the scatter of the cloud of points is maximised [6]. These directions are therefore not linked to a particular classification problem, thus leading to a 'good' representation of the data.

3.3. R-Tree based Indexing

The chosen indexing structure is based on R-trees [2], which are dynamic structures 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 box-based query "*Return all records with BBs intersecting the box* $\mathbf{b}$".

The fact that the extracted features exhibit a certain degree of robustness against image manipulations implies that the features of a replica are localised around those of the corresponding original image in the feature space. Therefore, an R-tree, optimised for replica detection, can be constructed by associating a "bounding box" with each original image in the database $\mathcal{O}$. In fact, since we are dealing with a d-dimensional space, the bounding boxes are d-dimensional orthotopes (generalised rectangular parallelepiped). 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

[2] ICA-FX [4] is a linear dimensionality reduction technique adapted to (binary) classification problem.

a large number of images in the answer $\mathcal{C}$. On the other hand, if the BBs are too small a replica can fall outside the BB corresponding to its original, and the original will not be included in the answer $\mathcal{C}$.

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 bounding box is defined by two vectors $\mathbf{c}^-$ and $\mathbf{c}^+$, which control its extent in each dimension:

$$c_\alpha^- = \min_r f_{r,\alpha}, \tag{2}$$

$$c_\alpha^+ = \max_r f_{r,\alpha}, \tag{3}$$

where the $f_{r,\alpha}$ correspond to the α-th feature of the r-th replica example, and the c_α to the α-th element of the vectors $\mathbf{c}$. The examples used to compute the bounding boxes are detailed in Sec. 4.

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

3.4. Binary Classifiers

The binary classifier described in [3] is used in the proposed system. 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 adapt each classifier to the corresponding original.

A trained binary classifier consists of the four steps outlined thereafter (for more details, refer to [3]). 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 penalised. For example, statistics about yellow pixels are incommensurable when the suspect and original images contain very different proportions of yellow pixels. In the *Statistical Normalisation* step the inter-image differences are statistically normalised. 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, keeping only feature mixtures relevant to the replica detection task. Finally, in the *Decision Function* step, a decision function is used to determine if the test image is a replica of the original image. This decision function is based on Support Vector Classifiers with a Gaussian kernel.

4. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed approach, we used the same image collection as in [7]. This collection contains $18,785$ photographs including 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 colour images, except for about one thousand grey-levels images.

The image collection is used as follows. First, 200 images are randomly chosen to be the original images. Secondly, 50 randomly picked images, and their replicas, are used to build the dimensionality reduction matrix $\mathbf{W}_d$. Then, the 200 originals and their replicas are used to compute the bounding boxes given by (2) and (3). In both cases, the replicas are generated using the manipulations proposed in [1] and outlined in Table 2(a). Finally, a binary replica detector is trained for each original as in [3]. 500 randomly picked images serve as non-replicas examples during the training phase. The $18,035$ images remaining in the collection are used as non-replica test images.

Table 2: Replicas used for testing and training.

Categories	♯ replicas
Colourising	4
Contrast changes	2
Cropping	4
Despeckling	1
Downsampling[a]	6
Flipping	1
Colour 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 [1, 7].

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 [8].

Two sets of test images, denoted Q and S, are used to assess the performance of the R-tree and of the whole system. Each set contains the $18,035$ non-replica test images, as well as different replicas of the original images. The test set Q contains the replicas generated by the manipulations listed in Table 2(a). On the other hand, the test set S includes the replicas generated by the manipulations of the well known watermarking benchmark tool StirMark 3.1 [8] as listed in Table 2(b).

4.1. R-tree Performance

The R-tree performance is assessed by measuring the miss-rate (i.e. the average probability that the R-tree does not return among its results the corresponding original when the test image is a replica) and the average number of returned candidates. For this purpose, the subsets Q^* and S^* (of Q and S respectively), that do not include the non-replica images, are used. Note that in general, the average number of returned candidates for non-replica images is one less than for replica images.

For the test set Q^*, point-based queries are used, and the results are shown in Table 3 for different dimensions of the feature vector. As expected (since the replicas used to build the bounding boxes are the same than those contained in Q), the correct original is always a member of the candidates. Moreover, the number of candidates is quite low. For example, when the feature vectors indexing the database are 25-dimensional, 1.3 originals are returned on average, or in other words about 99.3% of the 200 originals are discarded.

For the test set S^*, box-based queries are used (because the image manipulations of StirMark are quite severe) and the resulting curves, obtained by varying the size of the query bounding box, are shown in Fig. 1. In this case, a good performance (for example an average miss-rate of 0.01) leads to an almost useless R-tree as almost all originals are returned. This shows that the choice of the bounding boxes used to build the R-tree is quite important, as discussed in Sec. 3.3. However, the results obtained using the test set S^* should be mitigated because the replicas in this set correspond to more severe manipulations than those used to build the R-tree.

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

number of dimension d	15	25
miss rate of the R-tree	0	0
average size of $\mathcal{C}$	1.5	1.3

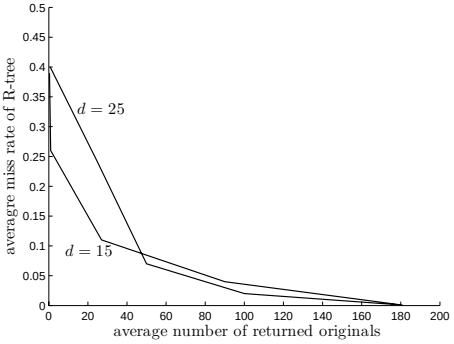

Fig. 1: R-tree miss-rates for test set S^*.

4.2. System Performance

The tradeoff between false negative and false positive rates for the entire system is now explored. A false positive occurs when an image that is not a replica of any original is classified as such, or when a wrong original is given for an image that is indeed a replica of an original. Conversely, a false negative happens when a replica of the original is classified by the system as a non-replica, or as a replica of another original. The false positive versus false negative curves are obtained by varying the threshold T in (1). Note that point-based queries are used for the test set Q, while box-based queries are issued for the test set S. In the latter case, the average miss-rate of the R-tree is set to 0.1. The resulting curves (for sets Q and S, respectively) are given in Fig. 2. The proposed system performs better for the test set Q than for S. For example, for a fixed false positive rate of 10^{-2} the average false negative rate is 0.06 for the test set Q and 0.2 for S. This was expected since S includes replicas generated by severe image manipulations and because the system is optimised for detecting replicas of the same type as the ones in the test set Q.

Finally, it can be observed that the false positives are mainly dominated by a few 'bad' originals. Indeed, when the experiments are rerun by leaving out the three originals responsible for the largest number of false positives the performance improves dramatically, as shown by curves Q_{-3} and S_{-3}, respectively. These originals are images containing few colours and/or having low contrast. Not only are these images not well described by the chosen features, but the bounding boxes used to index them have also large extents. Consequently, they will be frequently contained in $\mathcal{C}$. Additionally, the binary classifier associated with such an image gives often a high score to images that are not among its replicas.

5. CONCLUSION

In this work, a replica detection system capable of retrieving from a database of originals the one that corresponds to a given suspect image was presented. Since binary classifiers are used by the system, the suspect image has to be tested against every original contained in the database, which can be cumbersome as the number of originals grows. We showed that this drawback can be overcome by making use of an indexing structure such as an 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 99.3% of the database images can be pruned by the R-tree at the expense of an increased miss-rate. Nevertheless, the performance of the proposed system in terms of false positive and false negative rates is satisfactory.

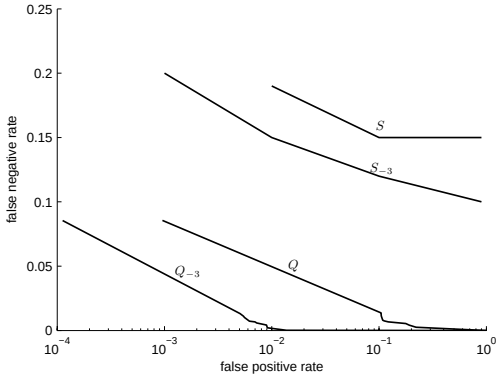

Fig. 2: Performance for the test sets Q and S with $d = 25$.

Future work will focus on improving performance of the system by taking into account a larger set of invariant features. Another research direction consists in describing the replicas of an image by more than a single bounding box. More thorough tests with a larger number of original images will also be conducted.

6. REFERENCES

[1] A. Qamra, Y. Meng, and E. Y. Chang, "Enhanced Perceptual Distance Functions and Indexing for Image Replica Recognition," *IEEE PAMI*, 2005.

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

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

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

[5] Y. Maret, S. Nikolopoulos, F. Dufaux, C. Cotsaces, T. Ebrahimi, and N. Nikolaidis, "Reduced Complexity Replica Detection System Using Binary Classifiers and R-Tree," in *Workshop On Immersive Communication And Broadcast Systems*, 2005.

[6] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*, Wiley-Interscience, 2000.

[7] 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.

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