Fast and robust duplicate image detection on the web

E. Gadeski, Hervé Le Borgne, A. Popescu

Fast and robust duplicate image detection on the web

Etienne Gadeski[1] · Hervé Le Borgne[1] · Adrian Popescu[1]

Abstract Social media intelligence is interested in detecting the massive propagation of similar visual content. It can be seen, under certain conditions, as a problem of detecting near duplicate images in a stream of web data. However, in the context considered, it requires not only an efficient indexing and searching algorithm but also to be fast to compute the image description, since the total time of description and searching must be short enough to satisfy the constraint induced by the web stream flow rate. While most of methods of the state of the art focus on the efficiency at searching time, we propose a new descriptor satisfying the aforementioned requirements. We evaluate our method on two different datasets with the use of different sets of distractor images, leading to large-scale image collections (up to 100 million images). We compare our method to the state of the art and show it exhibits among the best detection performances but is much faster (one to two orders of magnitude).

Keywords Social media intelligence · Near duplicate detection · Copy detection · Visual web data · Image retrieval

1 Introduction

Social media intelligence consists in collecting public media content in order to understand trends or drive communications and business strategy. It requires a massive processing of data, that must often be fast as well, in order to be able to react quickly to a particular event. It exists for a while some tools that process the textual content, but the advent of social web determined a rise of visual content (image/video) propagation on websites or on users' online social network (OSN) profiles, either verbatim or with minor modifications. From a

✉ Hervé Le Borgne
herve.le-borgne@cea.fr

[1] CEA, LIST, Vision and Content Engineering Laboratory, Gif-Sur-Yvette, France

media intelligence perspective, it is very interesting to detect massive content propagation, even slightly tampered, since it usually relates to the same "social event". This last relates here to an event in the real life: when a celebrity does something for instance, the (almost) same picture will be published many times. Such a phenomenon has also been observed in the early years of the Web with the diffusion of visual memes [37].

Near-duplicate image detection is of high interest to several multimedia applications and was thoroughly studied for checking unauthorized use of protected visual content. Our focus here is nevertheless on another application, namely visual media intelligence, which consists in detecting similar visual content published on the (social) web in a relative short period of time. We argue that under certain conditions, near-duplicate image detection is a good model to respond to the problem of detecting a massive content propagation of visual content that could feed a platform of social media intelligence.

Technically, regardless of its use, near-duplicate copy detection can be cast as a content based retrieval task. An important characteristic to be taken into account in media intelligence is that processed content comes as a *stream* of visual data, that must be processed continuously. To do this, the domain of copy detection usually focus on the speed of the search process and on the robustness to different image transformations. Consequently, most of the state of the art approaches rely on visual signatures that are built from local features, later aggregated into a small vector in order to speed up the search process. However, local features computation and aggregation has a non-negligible cost and indexing is usually performed offline. In the media intelligence scenario, the computation time of the visual signatures has to keep the pace with the reception of new input data. More precisely, the "indexing+search" operation must be executed at a higher rate than that of new data collection. For instance, if a system ingests half-million visual multimedia items per day, their comparison with recent content (assumed to include 10 to 100 million documents, already indexed during the previous months from daily inputs and reduced to significant and interesting ones only) must be performed in less than $\frac{24\times3600}{500000} = 172.8$ milliseconds, that is to say around 6 images per second. Such a processing rate requirement makes the use of signatures based on transforms and compression of local features difficult to use if computational resources are constrained. In practice, since the computational capacities directly impact the cost of the media intelligence platform, it is interesting to minimize their need for this particular processing.

Hence, to support such a processing rate, one must wonder the real need of near-duplicate detection in the context of visual media intelligence. The authors of [34] performed a user study which shows that the most frequent image transformations are usually quite simple. Frequent transformations include: format conversion, rescaling, changing the image aspect ratio, small crops or rotation, image or text embedding, (JPEG) compression, color transforms ("filters"), and possibly right-left flip. From a practical usage perspective, copy detection methods that aims at detecting re-post of a similar content should focus on these usual transformations in order to maximize their effectiveness.

The objective of this paper is thus to address the problem of fast indexing *and* searching of visual content in a media intelligence scenario. To achieve this objective, we propose a new signature which has the following properties: (i) fast to compute ($\leq$ 5 ms on a single core Intel Xeon processor @ 2.10 GHz) (ii) very compact ($<$ 100 bytes), thus enabling fast search of a large database (10^7 to 10^8 images), even exhaustively (iii) suitable for an inverse index representation to speed up the search (iv) robust to frequent Web image transformations. Our approach consists in extracting a compact hash signature, based on a particular pattern that make it quasi invariant to resizing and left-right flips. We refine its design in

order to make it robust to other usual image transformations on the Web, such as blur, rotation, crop, image compression as well as text and image incrustation.

This paper is built upon our prior conference publication [10] and include the following additions: 1) we propose a new scheme to enhance the signature that does not change its memory print (thus it preserves its efficiency at searching time) but improve its robustness to 'crop' transformations. 2) we compare the proposed pattern to alternative ones, and thus exhibits its most important aspects. 3) we conducted complementary experiments, including a run on a corpus containing more than 100 millions images.

We present recent and older works relative to our problem in Section 2. Then the proposed method is presented in Section 3, including the design of our "core" hash signature, its enhancement to make it more robust and the associated method for efficient search. In Section 4 we report a set of experiments that demonstrates the efficiency of the proposed approach, in comparison to alternative methods and former state-of-the-art works. The Section 5 is dedicated to highlights the main points of this paper and draw some perspectives to this work.

2 Related work

Watermarking-based approaches rely on a "mark" inserted into a document. The digital watermark must be perceptually invisible to a human and robust to the transforms the document can be subjected to. In addition to robustness, watermarking requires to perform the insertion from the beginning, a constraint which is unrealistic if the publisher does not control the initial publication of the content. A more flexible solution to copy detection is to find of near-duplicate documents directly from content. This process is an instantiation of the content-based image retrieval (CBIR), which first extracts a vectorial representation of visual content and then exploits it to find similar images in a test database. The choice of the image representation is usually made to ensure a good compromise between result accuracy and speed, two characteristics which are often contradictory.

From the first systems dedicated to copy detection [4] up to mid 2000's, many works were based on the use of global descriptors [21], in particular to detect repeated "clips" into a video stream [6, 23]. Although fast to compute, these methods usually suffered from a poor robustness to severe image transforms. They were thus progressively replaced by local feature based approaches [3, 8, 19]. These consist in detecting a region around points of interest then describing them with robust descriptors such as SIFT [22], SURF [2] or robust local binary patterns [13]. Local features are quantized according to a pre-defined codebook and then indexed into an inverse-file structure, which corresponds to a forward index described by bag-of-features (BoF) with hard coding [31], and can thus be retrieved efficiently. Further improvements have been proposed to compress or enhance this representation, making the retrieval process faster and more precise. In image classification, it has been shown that finer coding strategies can lead to better BoF representation [15]. However, for retrieval, few of these strategies have been tested. The improvement proposed in [16] better takes into account the local features quantization and add an efficient way to consider the spatial arrangement of interest points in each image. For very large scale databases, local descriptors are usually aggregated in a unique vector that describes the deviation of a given image with respect to an average representation of the visual world. This (VLAD or Fisher Kernel) vector is then compressed using PQ-codes [17] in order to be able to efficiently search a large database while maintaining good precision [18].

This compression of large signatures is in the vein of works aiming at representing images with compact signatures, that is to say a *hash*, that reflect the semantic of the image. Hashing based image representations were also used in the domain of image authentication and forensics, named robust image hashing [32]. Regarding content-based image retrieval, the semantic hashing [27] consists in learning a compact binary signature that preserve the class-based information. The original work [26, 28] used an autoencoder to learn such codes. Later, Weiss et al. proposed to compact a GIST descriptor [24] using spectral hashing [36]. A kind of combination of these last works was proposed simultaneously [35], consisting in learning a binary code from a GIST descriptor using several approaches, including Restricted Boltzman Machine [14]. In [17], Jegou et al. proposed to quantize several parts of a large signature such that it is compressed into a small number of integers. They also defined a distance to perform an exhaustive search in this space, showing that it is equivalent to an approximate search in the original space.

Hence, local feature based methods exhibit good performances and efficient indexing schemes have been proposed to exploit them for fast image search. However, these works usually report only searching time and the proposed methods are still too slow to compute in a "online streaming data" scenario for which the feature extraction time must be considered.

3 Proposed approach

We propose a descriptor and a search process that allow to detect near duplicate images under the media intelligence constraints related to processing time, memory size and copy detection accuracy of actual Web content. The core of the method is to extract a hash from a resized version of the image. Its construction is mostly based on pixel values comparisons and is therefore cheap to compute. Moreover, such a binary signature allows us to use the Hamming distance during the search, that is faster to compute than for example the Euclidean distance. The full descriptor is an enhancement of this hash signature, composed of additional information. A particular version of these enhancement consists to compute the same hash on the image in polar coordinates, to be more robust to rotation and scale transformations. In the following paragraphs, we describe the descriptor extraction process, common to the original image and its polar coordinates counterpart. The process of our method is illustrated on Fig. 1.

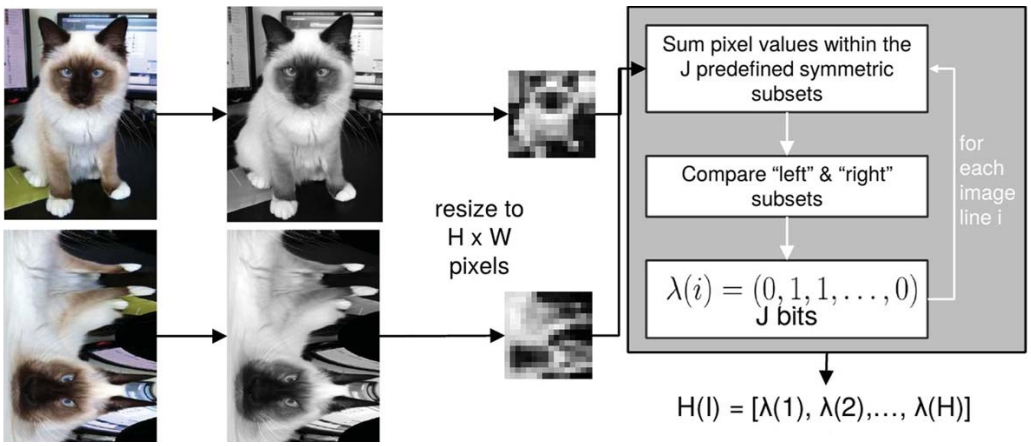

Fig. 1 Pipeline of the process to compute the proposed signature. It is applied to an image (top) and its transform into polar coordinates (bottom). See Section 3.2 for details

3.1 Rescaling

The first step of our method consists to convert the image to grayscale and reduce its size to $H \times W$ pixels. The image can be rescaled with any interpolation technique but taking the average of neighbouring pixels seems sufficient. This step insures a quasi invariance with respect to resampling transformations (with or without keeping the original aspect ratio). It also gives a good robustness with respect to small details of the image such as small watermarks or text incrustation. The conversion to greyscale combined to the extreme resizing leads in practice to a good robustness to color transformations. To make the descriptor invariant to flip, the descriptor should be built upon symmetric comparisons, therefore W must be an even number: $W = 2w$, $w \in \mathbb{N}^*$. One can also resize the image with an odd W and ignore the central column in the following.

3.2 Hash extraction

Let consider the rescaled image I_r of size $H \times W$. Each of its line (i) is a set of spatially ordered pixels $\mathcal{P} = (p_1, p_2, \ldots, p_{2w})$ with p_n being on the left of p_m when $n < m$. Let consider two subsets $(\mathcal{P}_x, \mathcal{P}_y)$ such that $\mathcal{P}_x \cap \mathcal{P}_y = \emptyset$ and:

$$p_j \in \mathcal{P}_x \Rightarrow p_{2w+1-j} \in \mathcal{P}_y \tag{1}$$

There are 2^w such couples of subsets, corresponding to $\#\mathcal{P}_x$ (and $\#\mathcal{P}_y$). We select J of these couples and define the following aggregating values on each of them:

$$x_i^j = \sum_{p_k \in \mathcal{P}_x^j} I_r(p_k) \text{ and } y_i^j = \sum_{p_k \in \mathcal{P}_y^j} I_r(p_k). \tag{2}$$

Considering the comparison function s defined by:

$$s(x, y) = 1 - \mathcal{H}(y - x) = \begin{cases} 1 & \text{if } x > y \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where $\mathcal{H}(.)$ is the Heaviside step function. We compute the hash value for line i by:

$$\lambda(i) = \left(s(x_i^1, y_i^1), \ldots, s(x_i^J, y_i^J) \right). \tag{4}$$

Finally, the resulting hash for the full image I is

$$h(I) = [\lambda(1), \lambda(2), \ldots, \lambda(H)]. \tag{5}$$

The hash of image I, noted $h(I)$, has thus a size of $J \times H$ bits. When J is a multiple of 8, each value of $h(.)$ can be efficiently encoded as an integer on J bits.

The definition of the aggregating functions (Equation (2)) insures a quasi invariance to flip transforms. However, it exists $H.2^{W/2}$ possibilities to compute $h(.)$ since there are 2^w possible couple $(\mathcal{P}_x, \mathcal{P}_y)$ and H lines. The question of the choice of the subsets is investigated in Section 4. In particular, we show that good results are obtained with $H = W = J = 16$ and the choices given in Table 3 for the subsets. It is compared to random choices of subset, showing nevertheless still good results as long as it contains the simplest subsets with one pixel.

3.3 Descriptor enhancement

In addition to the hash, we add information by keeping track of the mean intensity of the image $\mathrm{m}(I)$ and of $\mathrm{eq}(I) = \sum_{i=1}^{H} \sum_{j=1}^{J} \delta_{x_i^j, y_i^j}$, where $\delta_{a,b}$ is the Kronecker delta, corresponding to the number of comparisons which lead to an equality. These values are used to assert if one image is a mirrored duplicate of another. We also have noted in our experiments that using them globally improves the performance. The values $\mathrm{m}(I)$ and $\mathrm{eq}(I)$ are coded on a single byte (8 bits) each.

Another enhancement consists to compute the previously described components, both on the original image and its transform in polar coordinates. Such a process improves the robustness to rotation and crops since these transforms induce a translation of the pixel values. The induced change in the reduced image (of size $H \times W$) can be easily reduced by an appropriate choice of the subsets $(\mathcal{P}_x, \mathcal{P}_y)$ in the hash design, for instance by including several consecutive pixels in the subsets. However, this is only true of the center used to compute polar coordinates is approximately the same as that of the transform. In practice, using the spatial center of the image is often a good choice. An alternative is to determine it automatically according to the actual content of the image, for instance by the method given in [20] to obtain an invariant centroid, consisting in choosing the expectation of the luminance levels on each coordinate. It nevertheless did not improve our experimental results in practice, thus we only use the spatial center of the image to compute the polar coordinates in the following.

Let I be the original image and $\theta(I)$ the image in polar coordinate. The final descriptor is a concatenation of all the values computed and encoded:

$$\phi(I) = \left[h(I), \mathrm{m}(I), \mathrm{eq}(I), h\left(\theta(I)\right), \mathrm{m}\left(\theta(I)\right), \mathrm{eq}\left(\theta(I)\right)\right] \tag{6}$$

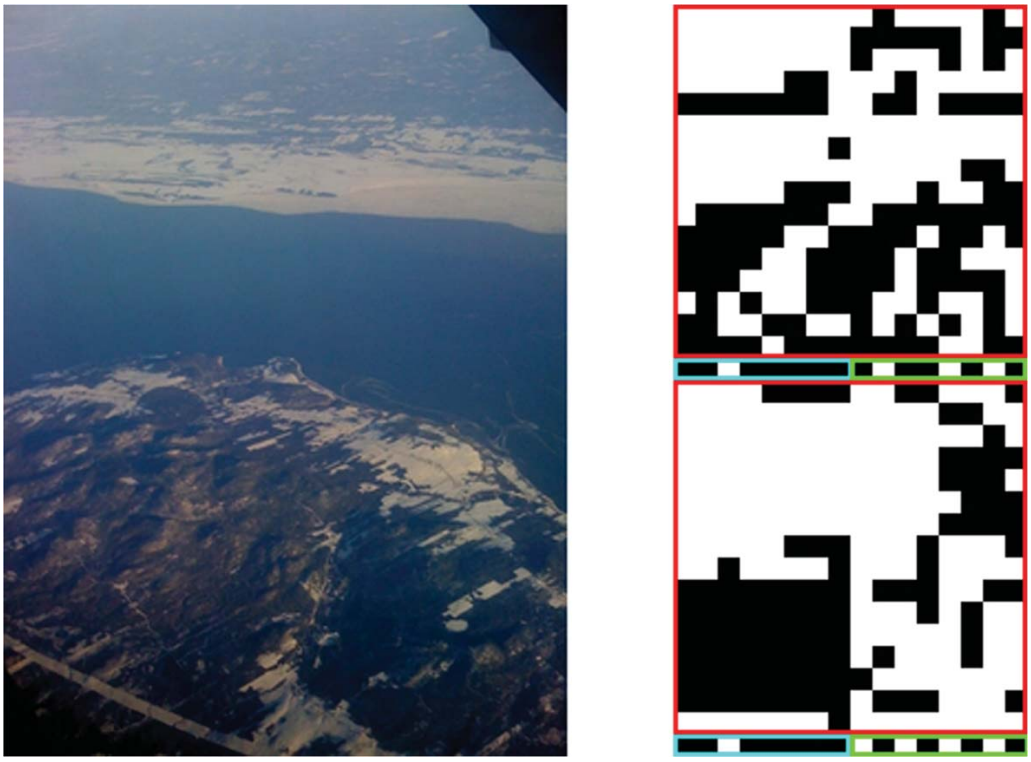

Fig. 2 Right part represents the binary description $\phi(I)$ of the image displayed on the left using the 68 bytes descriptor. There are 2 distinct parts corresponding to the original image and the image in polar coordinates. For each representation, 3 categories are identified with different colors: the red one represents $h(.)$, the blue one m(.) and the green one eq(.). Each black and white square within these categories represent a bit

It has a size of $2 \times (J \times H + 2 \times 8)$ bits. Figure 2 shows a binary representation of the description for one image using a setting which gives a 68 bytes signature.

An alternative is to use $\theta(I)^T$ instead of $\theta(I)$, leading to:

$$\phi_T(I) = \left[h(I), \mathrm{m}(I), \mathrm{eq}(I), h\left(\theta(I)^T\right), \mathrm{m}\left(\theta(I)^T\right), \mathrm{eq}\left(\theta(I)^T\right)\right]. \tag{7}$$

$\phi_T(.)$ has the same size as $\phi(.)$. Since the hash function acts on the lines of the reduced image of size $H \times W$, the "polar coordinates" part of $\phi(.)$ improves the robustness to the rotation while $\phi_T(.)$ improves that on the zoom (crops).

We tried to design a signature able to take both effects at the same time by considering consecutive lines (in addition to symmetric columns as specified by Equation (1)) but the experimental results were not improved significantly.

3.4 Searching process

Since our descriptor is composed of hashes (*i.e.* it is a binary signature), we use the Hamming distance d_H to compute the distance between two descriptors. d_H is defined as follows:

$$\forall a, b \in \{0,1\}^n, \mathrm{d}_H(a,b) = \#\mathcal{D}, \tag{8}$$

where $\mathcal{D} = \{i : a_i \neq b_i\}$. The Hamming distance thus counts the number of bits which differ between two values. However, in our framework, we do not compute Hamming distance on m(.) and eq(.): computing absolute difference on these values, allows us to determine if one image is a flipped (in x-axis, y-axis and both ways combined) duplicate of the other. The distance $d\left(\phi(I_1), \phi(I_2)\right)$ between two images I_1 and I_2 is

$$d\left(\phi(I_1), \phi(I_2)\right) = \mathrm{d}_H\left(h(I_1), h(I_2)\right) + \mathrm{d}_H\left(h\left(\theta(I_1)\right), h\left(\theta(I_2)\right)\right) + r, \tag{9}$$

where $r = \frac{|\mathrm{eq}(I_1) - \mathrm{eq}(I_2)| + |\mathrm{m}(I_1) - \mathrm{m}(I_2)|}{2}$. In our implementation we use the `POPCNT` instruction, which is available for recent processors, to compute the Hamming distance. The distance computation is done with one processor instruction whereas naive and Look-Up Tables based implementations use several instructions. Consequently, `POPCNT` is faster than other implementations we tested.

The searching process can also be dramatically speeded-up with the use of an inverse-file structure. It consists in quantizing the descriptor according to a discrete codebook that allows efficient indexing and fast search. Such a codebook is usually learned with k-means, from a large representative set of vectors in the description space (thus vectors $\phi(.)$ resulting from our process). Then, at indexing time, each vector $\phi(I_i)$ is assigned to an element c_j of the codebook, named a codeword. Then, at searching time, if a testing image I_t is represented by the vector $\phi(I_t)$ and that vector is closer to c_j than to any other codeword, then the searching process defined by Equation (9) will be applied to the vectors assigned to c_j only, not to all the vectors of the reference database. Hence, if the vectors are equally divided among the codeword (and K-means clustering should lead approximately to such a result) the searching time is divided by the size of the codebook.

However, preliminary experiments showed that the usual process based on k-means [7] led to a dramatic drop of the performance of our method. We thus propose to learn the codebook with a k-medians instead of k-means and show experimentally in Section 4 that we obtain the intended effect with an $80\times$ speed-up at the price of a very moderate performance drop.

Fig. 3 The first image starting from the left is the original image and the following 13 are detailed in Table 1

4 Experiments

As we mentioned, we focus on streamed visual data in the context of a social media intelligence scenario, in which all new images ingested by the system need to be compared with a dataset made of recent images. Experiments are carried out with two different benchmarks. The *Image Copy Detection dataset* is a benchmark that has been released recently and allows us to compare our method to the state of the art. We also propose our own benchmark (*WebTransforms*) that includes other transformations we consider important such as small rotation, flip and partial blur (e.g. on a human face) and that has been extended to a larger scale (14 to 100 million distractors). All experiments have been conducted on the same computer equipped with an Intel Xeon processor @ 2.10 GHz, 32 GB of RAM and running Linux.

4.1 Datasets and evaluation protocol

WebTransforms We propose a dataset in which thirteen transformations (see Table 1) have been applied to the 5,011 training image of the PascalVOC'07 test dataset [9], resulting into 70,154 images. We also merged our dataset into 2,000,000, 14,158,566, 85,308,719 and 100,158,012 distractor images extracted from Flickr to see how it behaves with relatively large-scale databases. A visual representation of the transformations is available in Fig. 3. We evaluate the search performance on this dataset with the recall@14, *i.e.* the fraction of relevant images retrieved at the top 14 positions, averaged for all the queries. By looking at rank 14, this measure is equal to precision@14 as well. We considerate this measure is

Table 1 Transformations used in the WebTransforms dataset

#	Name	Short description
1	identity	-
2	blur	moderate image blurring
3	partial blur	severe image blurring on a small portion of the image
4	rotation	centered, angle -10°
5	flip	left-right
6	rcrop	80 % area, random center
7	crop1	44 % area, centered
8	crop2	25 % area, centered
9	image incrustation	lena is positioned in the center and takes 25 % of the image
10	text incrustation	big colored text superposed
11	sepia	sepia filtering
12	compress	10 % JPEG compression ratio
13	resize1	scale factors: $x = 0.6$, $y = 1$
14	resize2	scale factors: $x = 1.2$, $y = 0.8$

more relevant than classical mean average precision when one consider a large number of distractors, since we are mainly interested in the relevance of the first retrieved document. In other words, the recall@14 criterion reflects the performance of a nearest neighbor (NN) classifier that would take its decision based on the first 14 retrieved documents. Note that the center of transforms 4, 7 and 8 are the same as that used to compute polar coordinates in Equations (6) and (7), that is to say the spatial center of the image. However, the center of transform 6 (random crop 80 %) is usually different from that used to compute the polar coordinates.

Image Copy Detection dataset The authors of [34] introduced a dataset to evaluate image detection methods for searching duplicate images on the web. This dataset is composed of 6,000 query pictures which are taken at various locations in the world. Sixty transformations (Table 2) have been applied to these images leading to a total of 360,000 images. Transformations were chosen after a survey which involved 45 persons familiar with image processing who were asked to report the most common transformations they encountered when looking at images with their favorite search engine. Following the evaluation protocol from [34], we merged near-duplicate images with 2,000,000 Flickr images collection. We also compared our method with GIST descriptor [24], TOP-SURF [33] and Convolutional Neural Network (CNN) intermediate features (layer name: `fc7`) extracted from a 16-layer network [30] trained on ImageNet [25]. The search quality is measured by the mean average precision (mAP) computed over the 6,000 queries for each transform.

4.2 Results

WebTransforms The evaluation is conducted with five settings of our descriptor: (I) $H = W = J = 8$ (20 bytes signature), (II) $H = W = J = 16$ (68 bytes signature), (III) $H = W = J = 16$ with J random couples of subsets of u to v pixels per subset, (IV) $H = W = J = 16$ where we pick the 8 couples of subsets containing one pixel and the others are random couples with 2 pixels and (V) $H = W = J = 16$ where we use $\phi_T(.)$ to extract the descriptor. The J subsets used in (II) mode are enumerated in Table 3. The first eight couples are used for setting (I). In practice, for settings (III) and (IV) we sample the random couples of subsets using a simple procedure. First, for each unique couple $(\mathcal{P}_x, \mathcal{P}_y)$ we randomly set, in the interval $[u; v]$, the number n of pixel positions to choose. In particular, setting (IV) has $n = 2$. For each pixel position, we pick a random one, p_j, to construct $\mathcal{P}_x$ and put its symmetric counterpart, p_{2w+1-j}, in $\mathcal{P}_y$. The J subsets used in (III) mode with $u = 1$ and $v = 2$ are enumerated in Table 4.

We compare our method to three descriptors: TOP-SURF [33], GIST [24], a global descriptor popular for web-scale image indexing [7] that is one of the best methods for content-based duplicate image detection [34] and deep CNN features which have demonstrated astounding results in different computer vision tasks [11, 29]. In the case of CNN features, due to their high dimensionality (4096 dimensions), we reduce them to 512 dimensions via PCA. Experiments show that even with fewer dimensions, this kind of features is still performing almost as good as uncompressed [1, 5]. We additionally binarize CNN features with iterative quantization (ITQ) [12], an unsupervised method which finds a rotation matrix that minimizes the quantization error of mapping the features to a binary representation. PCA and ITQ are computed on an independent dataset of 10,000 images. We chose to use binary codes in order to have a better comparison with our method in terms of number of bits per features, as our signature uses 544 bits or less. The results are reported in Table 5. The 68 bytes descriptor [line (b)] outperforms the 20 bytes one [line (a)] by +5.3 %. It is

Table 2 Transforms used in the copy detection dataset

Image compression	10 transformations	compress from 95 % to 50 % in steps of 5 %
Image scaling	13 transformations	scale factor goes from 20 % to 200 % in steps of 20 %, squash 5 % and 10 % along width or height
Image framing	12 transformations	crop 5 % or 10 % along width and/or height, add black border 5 % or 10 % along width and/or height
Image recoloring	16 transformations	convert to grayscale, reduce to 256 colors, brightness +10 % to +50 % and -10 % to -50 % in steps of 10 %, contrast +10 % and +20 %, saturate +50 % and +100 %
Image overlaying	9 transformations	add small or large copyright logo and/or text, add decorative lines, add simple or elaborate menu overlay

Table 3 Chosen configuration of subsets $(\mathcal{P}_x, \mathcal{P}_y)$ with $H = W = J = 16$ to conduct the experiments

$\mathcal{P}_x$	$\mathcal{P}_y$
p_1	p_{16}
p_2	p_{15}
p_3	p_{14}
p_4	p_{13}
p_5	p_{12}
p_6	p_{11}
p_7	p_{10}
p_8	p_9
p_1, p_2	p_{16}, p_{15}
p_3, p_4	p_{14}, p_{13}
p_5, p_6	p_{12}, p_{11}
p_7, p_8	p_{10}, p_9
p_1, p_2, p_3, p_4	$p_{16}, p_{15}, p_{14}, p_{13}$
p_5, p_6, p_7, p_8	$p_{12}, p_{11}, p_{10}, p_9$
$\{p_i\} : i \in [1, 8]$	$\{p_i\} : i \in [9, 16]$
$\{p_{2i}\} : i \in [1, 8]$	$\{p_{2i-1}\} : i \in [1, 8]$

also slightly better than GIST [line (k)] by +0.8 %, worse than TOP-SURF [line (l)] by -0.9 % and than CNN features [line (m)] by -0.6 %. Most of the performance loss are due to severe crops (transf. 7 and 8).

Our descriptor reacts quite well when merged to a large-scale image collection [lines (n), (r), (s) and (t)] since the performance drops only by 4 points with 2M distractors, giving a similar performance to GIST [line (o)]. Once again, this is mainly due to severe crops,

Table 4 Configuration of subsets $(\mathcal{P}_x, \mathcal{P}_y)$ of setting (III) with $u = 1$ and $v = 2$

$\mathcal{P}_x$	$\mathcal{P}_y$
p_1	p_{16}
p_2	p_{15}
p_3	p_{14}
p_4	p_{13}
p_7	p_{10}
p_8	p_9
p_3, p_7	p_{14}, p_{10}
p_1, p_6	p_{16}, p_{11}
p_4, p_5	p_{13}, p_{12}
p_4, p_7	p_{13}, p_{10}
p_3, p_4	p_{14}, p_{13}
p_5, p_7	p_{12}, p_{10}
p_4, p_8	p_{13}, p_9
p_3, p_5	p_{14}, p_{12}
p_6, p_8	p_{11}, p_9
p_3, p_6	p_{14}, p_{11}

Table 5 Results on the WebTransform dataset

Setting	1	2	3	4	5	6	7	8	9	10	11	12	13	14	recall@14
(a) (I)	100	100	100	64.28	99.98	78.31	6.35	0.44	53.02	99.16	92.64	**99.90**	100	100	78.1
(b) (II)	100	100	100	79.33	100	89.66	9.10	0.92	90.16	**99.82**	99.26	99.74	100	100	83.4
(c) (III), $u = 1, v = 8$	100	99.98	99.98	59.29	99.89	78.65	8.88	1.10	96.20	99.70	51.81	99.76	100	100	78.2
(d) (III), $u = 2, v = 8$	100	100	99.96	34.46	99.86	62.44	5.53	0.62	93.71	99.48	27.74	99.76	100	100	73.1
(e) (III), $u = 1, v = 2$	100	100	99.94	70.88	99.96	86.89	7.52	0.64	80.86	99.80	97.01	99.58	100	100	81.6
(f) (III), $u = 1, v = 5$	100	100	99.96	54.58	99.94	78.31	8.42	0.72	90.30	99.52	51.27	99.80	100	100	77.4
(g) (III), $u = 2, v = 3$	100	100	99.94	60.87	99.96	77.35	8.04	0.90	80.30	99.52	74.88	99.76	100	100	78.6
(h) (III), $u = 6, v = 8$	100	99.96	99.94	4.77	99.76	20.06	0.72	0.20	94.27	99.56	10.44	99.48	100	100	66.4
(i) (IV)	100	100	99.98	77.99	100	87.61	9.76	1.10	82.76	99.76	97.84	99.74	100	100	82.6
(j) (V)	100	100	100	78.33	100	90.08	14.65	1.32	**98.54**	99.66	98.62	99.68	100	100	**84.4**
(k) GIST	100	100	100	**96.95**	25.66	99.54	59.37	4.33	81.38	91.30	98.82	99.50	100	100	82.6
(l) TOP-SURF	99.96	99.36	99.66	82.22	22.41	96.79	66.49	19.42	96.24	98.92	**99.42**	99.70	99.94	99.94	84.3
(m) CNN	100	96.85	96.83	95.41	100	**99.98**	**89.32**	**48.83**	13.81	61.94	79.49	93.29	100	100	84
(n) (II) with 2M distr.	100	99.96	99.96	51.65	100	73.32	1.84	0.11	80.10	99.58	97.78	99.56	100	100	78.9
(o) GIST with 2M distr.	100	100	100	94.21	15.15	99.26	36.40	0.94	74.34	89.22	98.12	99.28	100	100	79.1
(p) TOP-SURF with 2M distr.	99.96	98.52	99.54	43.96	5.77	85.59	24.79	2.26	91.80	98.02	98.94	99.44	99.90	99.88	74.9
(q) CNN with 2M distr.	100	92.66	93.97	90.76	100	99.88	77.77	27.46	6.61	47.89	67.09	88.07	100	100	78
(r) (II) with 14M distr.	100	99.92	99.86	39.47	100	65.95	1.18	0.10	75.53	99.48	96.95	99.48	100	99.96	77
(s) (II) with 85M distr.	100	99.90	99.86	26.24	100	56.08	0.50	0.02	67.81	99.30	95.53	99.34	100	99.96	74.6
(t) (II) with 100M distr.	100	99.72	99.68	24.47	100	55.86	0.32	0.02	66.97	99.16	95.09	99.16	99.90	99.88	74.3

Transform numbers are those of Table 1. The score per column represents the fraction of queries for which the transform has been returned in the top 14 positions (recall@14)

although one can observe a significant drop for rotation as well. However according to [34] rotations are not often encountered on the web. With 14 million, 85 million and 100 million distractors, the performance of our descriptor drops by respectively 2, 4.5 and 4.8 more points only. We can also observe that TOP-SURF does not react well when merged with distractors [line (p)], the performance drops by 9 points which is similar to our descriptor with 85 million distractors. CNN features react quite well with distractors but the performance gap (-6 %) is still higher than with our descriptor. We can also note that CNN features are quite robust to crops, even severe ones, and invariant to flip and resize. This can be explained by the training procedure: multiple crops are extracted per training image at multiple scales, and subsequently randomly flipped horizontally. However, CNN features lack robustness to image incrustation and text incrustation: this might be caused by the objective function used to train the network. In fact, the network is trained to recognize 1,000 classes of objects. Incrusting images, text, watermark in an image may change the final decision of the network and lead to intermediate representations that are far apart from the unmodified counterpart.

However, the main advantage of our approach is that it is much faster, as shown in Table 6. For instance, GIST and TOP-SURF require respectively 316 ms and 120 ms per query (average on 5,011 queries) to search into the 70,154 images database, while our

Table 6 Results on the WebTransform dataset with computation time

	Computation time (sec. / query)		
Setting	Description	Matching	recall@14
(a) (I)	0.005	0.002	78.1
(b) (II)	0.005	0.004	83.4
(c) (III), $u = 1, v = 8$	0.005	0.004	78.2
(d) (III), $u = 2, v = 8$	0.005	0.004	73.1
(e) (III), $u = 1, v = 2$	0.005	0.004	81.6
(f) (III), $u = 1, v = 5$	0.005	0.004	77.4
(g) (III), $u = 2, v = 3$	0.005	0.004	78.6
(h) (III), $u = 6, v = 8$	0.005	0.004	66.4
(i) (IV)	0.005	0.004	82.6
(j) (V)	0.005	0.004	84.4
(k) GIST	0.050	0.316	82.6
(l) TOP-SURF	0.340	0.120	84.3
(m) CNN	1.07	0.003	84.0
(n) (II) with 2M distr.	0.005	0.106	78.9
(o) GIST with 2M distr.	0.050	8	79.1
(p) TOP-SURF with 2M distr.	0.340	1.93	74.9
(q) CNN with 2M distr.	1.07	0.102	78.0
(r) (II) with 14M distr.	0.005	0.845	77.0
		0.053†	
(s) (II) with 85M distr.	0.005	6.1	74.6
(t) (II) with 100M distr.	0.005	0.496†	74.3

Transform numbers are those of Table 1. Time are usually reported for a single core. Time with † are reported with 16 cores

approach needs 4 ms. On the 200 times larger database (14M images) our method requires 845 ms with a single core and scales almost linearly, requiring 53 ms (496 ms for 100M images) with 16 cores. The binarized CNN features require approximately the same amount of time (3 ms) to search into the database but the computation time associated to the description is very high on a single CPU core ($>$ 1 second).

For our method, the extraction is quasi-independent on the actual image. Indeed, the only "image-dependent" step is the conversion to grayscale and resizing. Once the input reduced to a fixed $H \times W$ grayscale image and the J subsets fixed, the extraction time is independent on the input. n the same vein, the search process is an exhaustive search on the full signature and thus only depends on its size and that of the database. Hence, once the reference database fixed and the signature design chosen, the search time is the same for any query.

We conducted several supplementary experiments to find a better choice of subsets than the particular arbitrary split of setting (II). The results are reported in lines (c) to (h) in Table 5. We can clearly see that the more dense is our description *i.e.* the number of pixels per subset is small, the better are the performances. For instance, with random subsets comprised of 6 up to 8 pixels [line (h)] the recall@14 is 15.2 % behind a setting where random subsets have 1 up to 2 pixels [line (e)]. We can also note that the setting (IV) [line (i)] is slightly better (+1 %) in comparison with (e). Then, we believe that building our descriptor with all unitary subsets is the most important aspect of our method to achieve the strongest robustness possible. We could not find any combination of random couples able to outperform our initial intuitive choice.

The results of setting (III) on line (c) to (h) are lower than other settings for transformation 4 (rotation) and 11 (sepia). It is particularly the case when $u > 1$ that is to say when the setting do not use unitary subsets. For instance, when these unitary subsets are removed, the settings drop from 51.8 % [line (c)] or the sepia transformation to 27.7 % [line (d)], while it is almost the same at 51.2 % [line (f)] when the largest subset are removed. on the contrary, when the smallest subsets are kept only [line (e)] the results are very good (97 %). This is probably due to largest change in Equation (2) (average luminance) when the pixel subset is larger. A similar rationale can be conducted on the polar-coordinates part of the signature in the case of the rotation transform. For these two transformations in particular, the importance of the unitary subsets is even more highlighted.

We also include the evaluation of setting (V) [line (j)] which gives better results than other methods. The improvement over setting (II) (+1 %) is mostly caused by a better robustness to moderate crops (transform 7) and image incrustation (transform 9). Robustness to transform 7 is aligned with expectation since (V) was specifically designed to be

Table 7 Results on the Image Copy Detection dataset, compared to methods at the state-of-the art

Method	Computation time (sec.) Description	Matching	mAP
TOP-SURF [33]	0.340	2.2	93.7 %
GIST [24]	0.05	9	93.2 %
CNN [30]	1.07	0.116	97.3 %
(II) (full)	0.005	0.120	99.1 %
(II) (quantized)	0.005	0.0015	96.7 %
(V) (full)	0.005	0.120	99.3 %

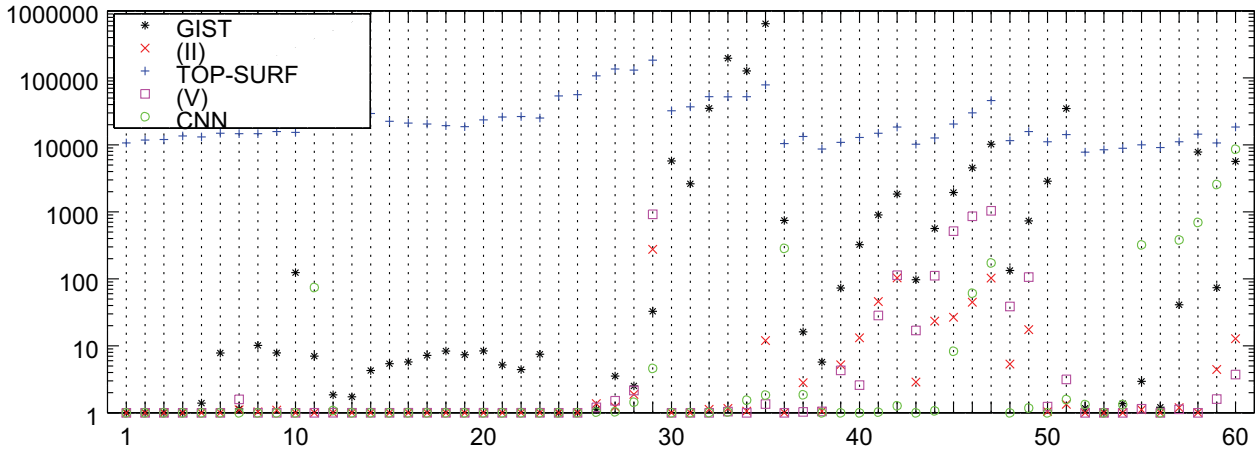

Fig. 4 Average rank per duplicate (the lower, the better) for each of the transformations of the Image Copy Detection dataset. Note the logarithmic scale for y-axis. The x-axis refers to the transformation number of the dataset

more robust to crops. Robustness to transform 9 can be interpreted in the same vein. Indeed, the incrustation of Lena hides 25 % of the center of the image and the similarity with copies holds on the periphery. Hence, since $\phi_T(.)$ catches the information at a given distance with respect to the image center, it make this setting particularly robust to this transform.

Image Copy Detection dataset For this experiment we compare two settings, (II) and (V), to the methods of the state of the art. We include both speed and accuracy measures in

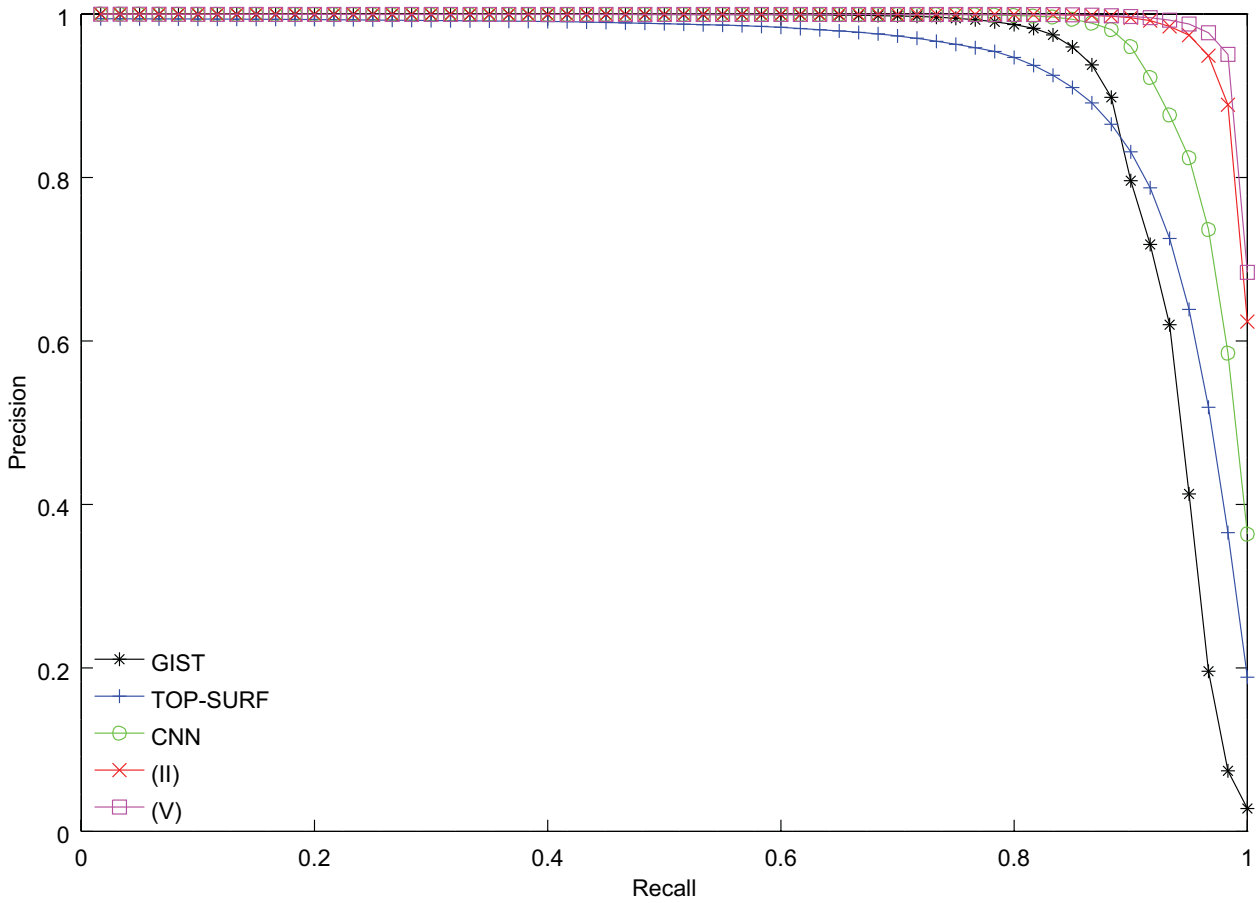

Fig. 5 Precision-recall curve of all methods for all transformations of the Image Copy Detection dataset combined

the evaluation. Therefore we measure, for each method, the description and matching times, the precision and the average rank per duplicate for each transformation. Description and matching time and mean average precision for each method are reported in Table 7. Our descriptor is at least one order of magnitude faster than other methods.

Our method with setting (II) obtains a mAP of 99.1 % and setting (V) reaches a mAP of 99.3 % over all the 60 transformations, whereas GIST, TOP-SURF and CNN features are behind with a mAP of 93.2 %, 93.7 % and 97.3 % respectively. To give further insight into the obtained results, we present average rank per duplicate in Fig. 4 and the precision-recall curve in Fig. 5. With our method, the duplicate image is, on average, ranked at the first position for a wide majority of transformations. Setting (V) works slightly better than

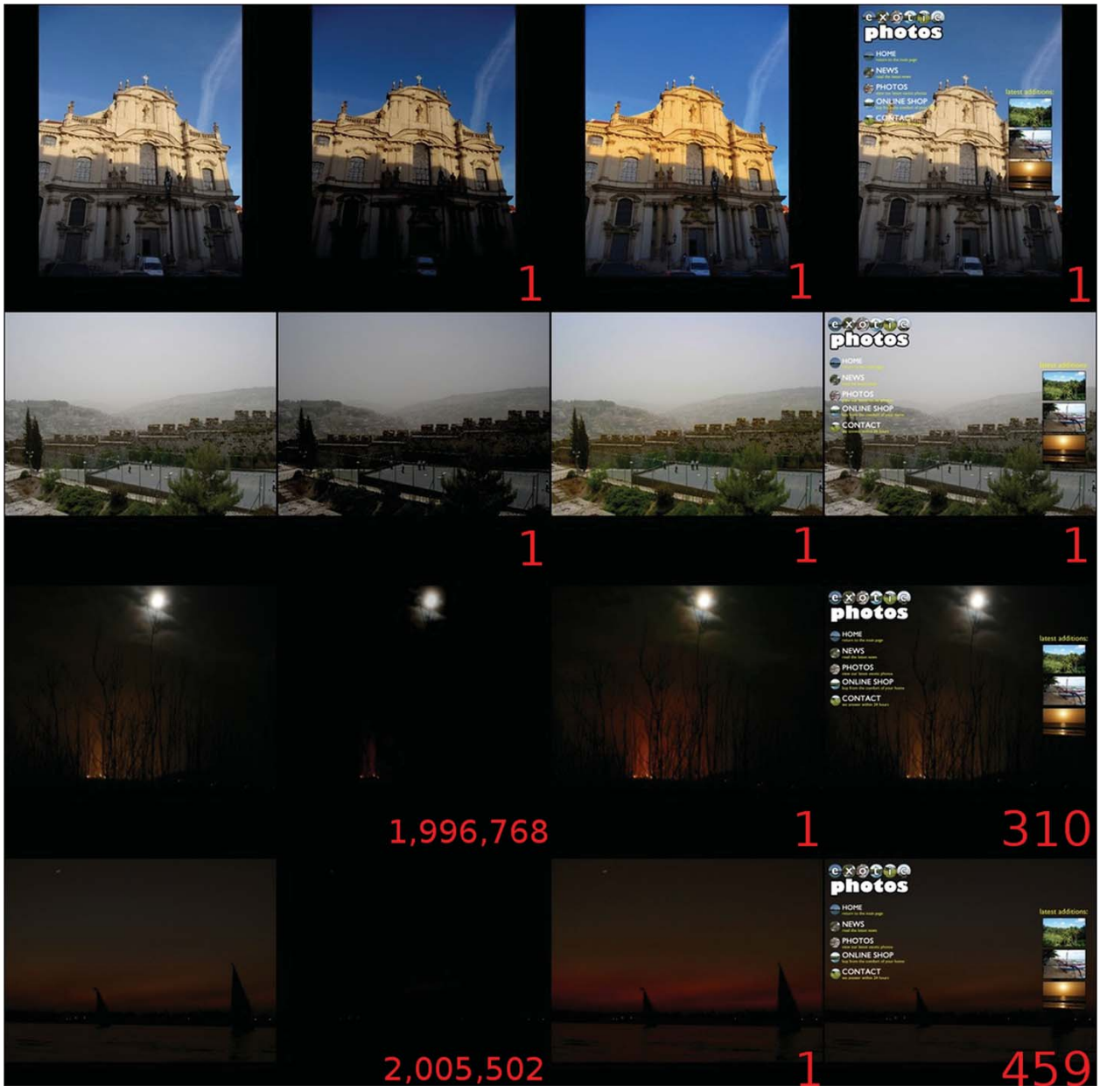

Fig. 6 Query image is displayed on the left, transformed images (-50 % brightness, +100 % saturate, big menu incrustation) are displayed on the right for each image. The ranks of the duplicate images returned by our descriptor are displayed on the bottom right of each image. The last two queries are dark, when -50 % brightness is applied the transformed images are almost completely black, therefore our method fails to capture their structure

setting (II): for several transformations the duplicate image is ranked closer with setting (V). However, for some of the transformations the mean rank is high but this is mostly due to few images ranked very far. Figure 6 shows several examples of ranked images when -50 % brightness, +100 % saturate and big menu incrustation transforms are applied. Overall, the obtained results show that our descriptor outperforms TOP-SURF, GIST and CNN features for this task both in terms of accuracy and scalability.

We also implemented an inverse indexing structure using a vocabulary of 20,000 codewords learned with k-medians as explained in Section 3.4, in order to achieve significant search time speedup while maintaining a good accuracy. In this scheme, a query x is associated to its m nearest codewords $(q_1(x), q_2(x), \ldots, q_m(x))$. For each codeword $q(x)$ we compute the distance between the query vector x and each base vector y_i within the list. We found that $m = 200$ was a satisfying trade-off between search speed and performance in our experimental results. This approach allows to compare on average each image to only 1.5 % of the database and thus to reduce the search time from 120 ms to 1.5 ms on the same hardware while keeping a mAP of 96.7 %. The $\frac{120}{1.5} = 80$ speedup is consistant with the theory that should be $\frac{20000}{200} = 100$. The difference $(1.5 - 1.2 = 0.3$ ms) is due to the search of the m closest centroids to the query.

5 Conclusions

We introduce a fast and robust image description which is well suited for indexing and searching through image data streams. Its compact size ($<$ 100 bytes) and the use of a efficient Hamming distance computation allows us to extract a descriptor for one image and exhaustively search a large image collection ($\approx$ 100M images) for duplicates in half a second on a 16-core processor. We also showed that our method can support an efficient inverse index structure, leading to a huge speed-up ($80\times$) while keeping a better accuracy than recent state of the art.

We conducted several experiments to show the efficiency of our method that demonstrate that it has better performance than the state-of-the art ones, both in precision but above all regarding the efficiency. Giving a base of reference of 2,360,000 images our method supports a rate of 150 independent queries per second on a single core (with the inverse indexing structure speedup). With such performances, one can henceforth consider to detect duplicate images in a real system processing a stream of web images.

We also proposed several alternative designs of our descriptor and studied their consequences. Among them, we proposed a "best mode" which gives the best overall robustness without altering the size nor the extraction process. However, despite its good performance, the particular choice we propose as "best mode" may not be the best. Finding such an optimal pattern to design the core of our signature will be the subject of future works.

References

1. Babenko A, Slesarev A, Chigorin A, Lempitsky VS (2014) Neural codes for image retrieval. In: Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, 2014, Proceedings, Part I, pp 584–599
2. Bay H, Ess A, Tuytelaars T, Van Gool L (2008) Speeded-up robust features (surf). Comput Vis Image Underst 110(3):346–359

3. Berrani S-A, Amsaleg L, Gros P (2003) Robust content-based image searches for copyright protection. In: Proceedings of the 1st ACM International Workshop on Multimedia Databases, MMDB '03. ACM, NY, USA, pp 70–77
4. Chang EY, Wang JZ, Li C, Wiederhold G (1998) Rime: a replicated image detector for the world wide web. In: Proceedings of SPIE, vol 3527, pp 58–67
5. Chatfield K, Simonyan K, Vedaldi A, Zisserman A (2014) Return of the devil in the details: Delving deep into convolutional nets. In: British Machine Vision Conference
6. Döhring I, Lienhart R (2009) Mining tv broadcasts for recurring video sequences. In: CIVR. ACM, NY, USA, pp 28:1–28:8
7. Douze M, Jégou H, Sandhawalia H, Amsaleg L, Schmid C (2009) Evaluation of gist descriptors for web-scale image search. In: International Conference on Image and Video Retrieval. ACM, NY, USA, pp 19:1–19:8
8. Douze M, Jegou H, Schmid C (2010) An image-based approach to video copy detection with spatio-temporal post-filtering. IEEE Transactions on Multimedia:257–266
9. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results
10. Gadeski E, Le Borgne H, Popescu A (2015) Duplicate image detection in a stream of web visual data. In: 13th International Workshop on Content-Based Multimedia Indexing (CBMI), pp 1–6
11. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
12. Gong Y, Lazebnik S (2011) Iterative quantization: A procrustean approach to learning binary codes. In: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR'11. IEEE Computer Society, DC, USA, pp 817–824
13. Heikkilä M, Pietikäinen M, Schmid C (2009) Description of interest regions with local binary patterns. Pattern Recogn 42(3):425–436
14. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
15. Huang Y, Wu Z, Wang L, Tan T (2014) Feature coding in image classification A comprehensive study. IEEE Trans Pattern Anal Mach Intell 36(3):493–506
16. Jegou H, Douze M, Schmid C (2008) Hamming embedding and weak geometric consistency for large scale image search. In: ECCV. Berlin, Heidelberg, pp 304–317
17. Jégou H, Douze M, Schmid C, Pérez P (2010) Aggregating local descriptors into a compact image representation. In: IEEE Conference on Computer Vision & Pattern Recognition, pp 3304–3311
18. Jegou H, Perronnin F, Douze M, Sanchez J, Perez P, Schmid C (2012) Aggregating local image descriptors into compact codes. IEEE T Pattern Anal Mach Intell 34(9):1704–1716
19. Joly A, Buisson O, Frelicot C (2007) Content-based copy retrieval using distortion-based probabilistic similarity search. IEEE T Multimed 9(2):293–306
20. Kim B-S, Choi J-G, Park K-H (2003) Digital Watermarking: First International Workshop, IWDW 2002 Seoul, Korea, 2002 Revised Papers, chapter Image Normalization Using Invariant Centroid for RST Invariant Digital Image Watermarking. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 202–211
21. Law-to J, Buisson O, Chen L, Gouet-brunet V, Joly A, Boujemaa N, Laptev I, Stentiford F (2007) Video copy detection: a comparative study. In: CIVR, pp 371–378
22. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
23. Naturel X, Gros P (2005) A fast shot matching strategy for detecting duplicate sequences in a television stream. In: Proceedings of the 2Nd International Workshop on Computer Vision Meets Databases, CVDB '05. ACM, NY, USA, pp 21–27
24. Oliva A, Torralba A (2001) Modeling the shape of the scene A holistic representation of the spatial envelope. Int J Comput Vis 42(3):145–175
25. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L (2015) Imagenet large scale visual recognition challenge. Int J Comput Vis (IJCV) 115(3):211–252

26. Salakhutdinov R, Hinton G (2007) Semantic hashing. In: SIGIR workshop on Information Retrieval and applications of Graphical Models
27. Salakhutdinov R, Hinton G (2009) Semantic hashing. Int J Approx Reason 50(7):969–978
28. Salakhutdinov R, Hinton GE (2007) Learning a nonlinear embedding by preserving class neighbourhood structure. In: Meila M., Shen X. (eds) Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07), vol 2. Journal of Machine Learning Research - Proceedings Track, pp 412–419
29. Sharif Razavian A, Azizpour H, Sullivan J, Carlsson S (2014) Cnn features off-the-shelf: An astounding baseline for recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops
30. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409:1556
31. Sivic J, Zisserman A (2003) Video google: A Video a text retrieval approach to object matching in videos. In: ICCV, vol 2, pp 1470–1477
32. Swaminathan A, Mao Y, Mu W (2006) Robust and Secure image hashing. IEEE Trans Inf Forensics Secur 1(2):215–230
33. Thomee B, Bakker EM, Lew MS (2010) Top-surf: a visual words toolkit. In: ACM Multimedia. ACM, pp 1473–1476
34. Thomee B, Huiskes M, Bakker E, Lew M (2013) An evaluation of content-based duplicate image detection methods for web search, vol 2013
35. Torralba A, Fergus R, Weiss Y (2008) Small codes and large image databases for recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008
36. Weiss Y, Torralba A, Fergus R (2009) Spectral hashing. In: Koller D., Schuurmans D., Bengio Y., Bottou L. (eds) Advances in Neural Information Processing Systems 21. Curran Associates, Inc., pp 1753–1760
37. Xie L, Natsev A, Kender JR, Hill M, Smith JR (2011) Visual memes in social media: Tracking real-world news in youtube videos. In: Proceedings of the 19th ACM International Conference on Multimedia, MM'11. ACM, NY, USA, pp 53–62