Near Duplicate Image Detection: min-Hash and tf-idf Weighting

Ondřej Chum[1] James Philbin[2] Andrew Zisserman[2]
[1]CMP, Czech Technical University in Prague [2]VGG, University of Oxford

Abstract

This paper proposes two novel image similarity measures for fast indexing via locality sensitive hashing. The similarity measures are applied and evaluated in the context of near duplicate image detection. The proposed method uses a visual vocabulary of vector quantized local feature descriptors (SIFT) and for retrieval exploits enhanced min-Hash techniques. Standard min-Hash uses an approximate set intersection between document descriptors was used as a similarity measure. We propose an efficient way of exploiting more sophisticated similarity measures that have proven to be essential in image / particular object retrieval. The proposed similarity measures do not require extra computational effort compared to the original measure.

We focus primarily on scalability to very large image and video databases, where fast query processing is necessary. The method requires only a small amount of data need be stored for each image. We demonstrate our method on the TrecVid 2006 data set which contains approximately 146K key frames, and also on challenging the University of Kentucky image retrieval database.

1 Introduction

The definition of a near duplicate image varies depending on what photometric and geometric variations are deemed acceptable. The application ranges from exact duplicate detection where no changes are allowed to a more general definition that requires the images to be of the same scene, but with possibly different viewpoints and illumination. In this paper, we build on a min-Hash method [2, 4] that addresses (through a similarity threshold parameter) a whole range of near duplicate images: from images that appear, to a human observer, to be identical or very similar to images of the same scene or object.

Detection of near duplicate images in *large databases* imposes two challenging constraints on the methods used. Firstly, for each image only a small amount of data (a fingerprint) can be stored; secondly, queries must be very cheap to evaluate. Ideally, enumerating all the duplicates of an image should have complexity close to linear in the number of duplicates returned.

1.1 Efficient image representations and search

The choice of an image representation and a distance measure affects both the amount of data stored per image and the time complexity of the database search. The amount of stored data ranges from a constant (small) amount of data per image to storing large sets of image features, whose size often far exceeds the size of the images themselves. When searching the database for relevant images, algorithms of different time complexity are

used, the most naive approach being computing the similarity between every image pair in the database.

Recently, a *bag* of visual words with *tf-idf* (term frequency – inverse document frequency) weighting [19] has proven to be a very successful approach for image and particular object retrieval, even on large corpora [17, 18]. The *tf* part of the weighting scheme captures the number of features described by a given visual word. The frequency of visual word in the image provides useful information about repeated structures and textures. The *idf* part captures the informativeness of visual words – visual words that appear in many different images are less informative than those that appear rarely.

In this paper, we are interested in near duplicate image detection (NDID), specifically in the min-Hash algorithm. The algorithm originates from text retrieval [2] and was previously used for NDID in [4]. The min-Hash method stores only a small constant amount of data per image, and a complexity for duplicate enumeration that is close to linear in the number of duplicates returned. The method represents the image by a sparse *set* of visual words. Similarity is measured by the set overlap (the ratio of sizes between the intersection and the union). The advantage of such a choice of image representation and the similarity measure is that it enables very efficient retrieval. The drawback is that some relevant information is not preserved in the set of visual words (binary) representation.

In this paper, we propose two more complex similarity measures that are inspired by image retrieval systems. We show that the proposed similarity measures can be efficiently computed using a modified min-Hash procedure with no extra computational cost. The first extension represents an image by a set of weighted visual words. In this approach, each visual word is assigned a weight (e.g. *idf*). The similarity function is a set overlap that is weighted by the word weights, i.e. words with low weight (common to most of the documents) contribute to the similarity less than rare, discriminative words. As a second step towards *tf-idf* weighting, we propose extending the similarity measure to compute a weighted histogram intersection score, which is able to take the term frequency into account.

1.2 Related work

The closest work to ours is of Chum et al. [4] on NDID based on min-Hash. We extend the method and directly compare the results. Ke et al. [12] demonstrate near-duplicate detection and sub-image retrieval by using sparse features, taken from each image, coupled with a disk-based Locality Sensitive Hashing (LSH) for fast approximate search on the individual feature descriptors. They demonstrate the efficacy of their method on a synthetic database of "corrupted" images but show the system only scaling to handle 18K images with query times many times slower than the min-Hash method. Zhang & Chang [22] use a parts-based representation of each scene by building Attributed Relational Graphs (ARG) between interest points. They then compare the similarity of two images by using Stochastic Attributed Relational Graph Matching, to give impressive matching results. Unfortunately, they only demonstrate their method on a few hundred images and don't discuss any way to scale their system to larger data sets of images.

Relevant work has been published on near duplicate shot detection (NDSD). These methods typically use strong temporal constraints than are not available in NDID. For example, [1, 6, 23] use an edit distance. The method of Joly *et al.* [10, 11] represents each keyframe by a set of 20D spatio-temporal descriptors computed about Harris interest points (requiring to store a large amount of data per keyframe – possibly hundreds of

Harris points and their descriptors).

Recently, attention has been drawn to hashing based image retrieval. In [20] Torralba et al. proposed to learn short descriptors to retrieve similar images from a huge database. The method is based on a dense 128D global image descriptor, which limits the approach to no geometric / viewpoint invariance. Jain et al. [8] introduced a method for efficient extension of Locally Sensitive Hashing scheme [7] for Mahalanobis distance. Both aforementioned approaches use bit strings as a fingerprint of the image. In such a representation, direct collision of similar images in a single bin of the hashing table is unlikely and a search over multiple bins has to be performed. This is feasible (or even advantageous) for approximate nearest neighbour or range search when the query example is given. However, for clustering tasks (such as finding all groups of near duplicated images in the database) the bit string representation is less suitable.

2 Image Representation and Similarity Measures

Recently, most of the successful image indexing approaches are based on the bag-of-visual-words representation [5, 9, 17, 18, 19]. In this framework, for each image in the data set affine invariant interest regions are detected. Popular choices are MSER [15], DoG (difference of Gaussians) [14] or multi-scale Hessian interest points [16]. Each detected feature determines an affine covariant measurement region, typically an ellipse defined by the second moment matrix of the region. An affine invariant descriptor is then extracted from the measurement regions. Often a 128-dimensional SIFT [14] descriptor is used.

A 'visual vocabulary' [19] is then constructed by vector quantization of feature descriptors. Often, k-means or some variant is used to build the vocabulary [17, 18]. The image database or a random subset can be used as the training data for clustering. The k-means cluster centers define visual words and the SIFT features in every image are then assigned to the nearest cluster center to give a visual word representation.

Assume a vocabulary $\mathcal{V}$ of size $|\mathcal{V}|$ where each visual word is encoded with unique identifier from $\{1,\ldots,|\mathcal{V}|\}$. A bag-of-visual-words approach represents an image by a vector of length $|\mathcal{V}|$, where each element denotes the number of features in the image that are represented by given visual word. A *set* $\mathcal{A}_i$ of words $\mathcal{A}_i \subset \mathcal{V}$ is a weaker representation that does not store the number of features but only whether they are present or not.

We will discuss three different image similarity measures. Two measures use a set of visual words image representation; the last one uses a bag of visual words representation. All of them can be efficiently approximated by randomized algorithms. Note that the proposed similarity measures shares some properties of the *tf-idf* scheme which is known to perform well in image retrieval.

Set similarity. The distance measure between two images is computed as the similarity of sets $\mathcal{A}_1$ and $\mathcal{A}_2$, which is defined as the ratio of the number of elements in the intersection over the union:

$$\mathrm{sim}_s(\mathcal{A}_1,\mathcal{A}_2) = \frac{|\mathcal{A}_1 \cap \mathcal{A}_2|}{|\mathcal{A}_1 \cup \mathcal{A}_2|}. \tag{1}$$

This similarity measure is used by text search engines [2] to detect near-duplicate text documents. In NDID, the method was used in [4]. The efficient algorithm for retrieving near duplicate documents, called min-Hash, is reviewed in section 3.

Weighted set similarity. The set similarity measure assumes that all words are equally important. Here we extend the definition of similarity to sets of words with differing importance. Let $d_w \geq 0$ be an importance of a visual word X_w. The similarity of two sets $\mathscr{A}_1$ and $\mathscr{A}_2$ is

$$\mathrm{sim}_w(\mathscr{A}_1, \mathscr{A}_2) = \frac{\sum_{X_w \in \mathscr{A}_1 \cap \mathscr{A}_2} d_w}{\sum_{X_w \in \mathscr{A}_1 \cup \mathscr{A}_2} d_w}. \tag{2}$$

The previous definition of similarity (1) is a special case of the new definition (2) for $d_w = 1$. Efficient algorithm for retrieval using sim_w similarity measure is derived in section 4.1.

Histogram intersection. Let t_i be a vector of size $|\mathscr{V}|$ where each coordinate t_i^w is the number of visual words X_w present in the i-th document. The histogram intersection measure is defined as

$$\mathrm{sim}_{h_0}(\mathscr{A}_1, \mathscr{A}_2) = \frac{\sum_w \min(t_1^w, t_2^w)}{\sum_w \max(t_1^w, t_2^w)}. \tag{3}$$

This measure can be also extended using word weightings to give:

$$\mathrm{sim}_h(\mathscr{A}_1, \mathscr{A}_2) = \frac{\sum_w d_w \min(t_1^w, t_2^w)}{\sum_w d_w \max(t_1^w, t_2^w)}. \tag{4}$$

This similarity measure (4) is closer to the *tf-idf* weighting scheme, while preserving the advantages of very fast retrieval of near identical documents using the min-Hash algorithm – see section 4.2 for details.

3 Min Hash Background

In this section, we describe how a method originally developed for text near-duplicate detection [2] is adopted to near-duplicate detection of images.

Two documents are near duplicate if the similarity sim_s is higher than a given threshold ρ. The goal is to retrieve all documents in the database that are similar to a query document. This section reviews an efficient randomized hashing based procedure that retrieves near duplicate documents in time proportional to the number of near duplicate documents. The outline of the algorithm is as follows: First a list of min-Hashes are extracted from each document. A min-Hash is a single number having the property that two sets $\mathscr{A}_1$ and $\mathscr{A}_2$ have the same value of min-Hash with probability equal to their similarity $\mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2)$. For efficient retrieval the min-Hashes are grouped into n-tuples called sketches. Identical sketches are then efficiently found using a hash table. Documents with at least h identical sketches (sketch hits) are considered as possible near duplicate candidates and their similarity is then estimated using all available min-Hashes.

min-Hash algorithm. A number of random hash functions is given $f_j : \mathscr{V} \to R$ assigning a real number to each visual word. Let X_a and X_b be different words from the vocabulary $\mathscr{V}$. The random hash functions have to satisfy two conditions: $f_j(X_a) \neq f_j(X_b)$ and $P(f_j(X_a) < f_j(X_b)) = 0.5$. The functions f_j also have to be independent. For small vocabularies, the hash functions can be implemented as a look up table, where each element of the table is generated by a random sample from $\mathrm{Un}(0,1)$.

Note that each function f_j infers an ordering on the set of visual words $X_a <_j X_b$ iff $f_j(X_a) < f_j(X_b)$. We define a min-Hash as a smallest element of a set $\mathscr{A}_i$ under ordering induced by function f_j

$$m(\mathscr{A}_i, f_j) = \arg\min_{X \in \mathscr{A}_i} f_j(X).$$

For each document $\mathscr{A}_i$ and each hash function f_j the min-Hashes $m(\mathscr{A}_i, f_j)$ are recorded. The method is based on the fact, which we show later on, that the probability of $m(\mathscr{A}_1, f_j) = m(\mathscr{A}_2, f_j)$ is

$$P(m(\mathscr{A}_1, f_j) = m(\mathscr{A}_2, f_j)) = \frac{|\mathscr{A}_1 \cap \mathscr{A}_2|}{|\mathscr{A}_1 \cup \mathscr{A}_2|} = \mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2). \tag{5}$$

To estimate $\mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2)$, N independent hash functions f_j are used. Let l be the number of how many times $m(\mathscr{A}_1, f_j) = m(\mathscr{A}_2, f_j)$. Then, l follows the binomial distribution $Bi(N, \mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2))$.The maximum likelihood estimate of $\mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2)$ is l/N.

How does it work? Let $X = m(\mathscr{A}_1 \cup \mathscr{A}_2, f_j)$. Since f_j is a random hash function, each element of $\mathscr{A}_1 \cup \mathscr{A}_2$ has the same probability of being the least element. Therefore, we can think of X as being drawn at random from $\mathscr{A}_1 \cup \mathscr{A}_2$. If X is an element of both $\mathscr{A}_1$ and $\mathscr{A}_2$, i.e. $X \in \mathscr{A}_1 \cap \mathscr{A}_2$, then $m(\mathscr{A}_1, f_j) = m(\mathscr{A}_2, f_j) = X$. Otherwise either $X \in \mathscr{A}_1 \setminus \mathscr{A}_2$ and $X = m(\mathscr{A}_1, f_j) \neq m(\mathscr{A}_2, f_j)$; or $X \in \mathscr{A}_2 \setminus \mathscr{A}_1$ and $m(\mathscr{A}_1, f_j) \neq m(\mathscr{A}_2, f_j) = X$. The equation (5) states that X is drawn from $|\mathscr{A}_1 \cup \mathscr{A}_2|$ elements at random and the equality of min-Hashes occurs in $|\mathscr{A}_1 \cap \mathscr{A}_2|$ cases.

Sketches. For efficiency of the retrieval, the min-Hashes are grouped into n-tuples. Let F be an n-tuple $(f_1, \ldots, f_n)$ of different independent random hash functions on $\mathscr{V}$. Let $S_F(\mathscr{A}_1)$ be a *sketch* $(m(\mathscr{A}_1, f_1), \ldots, m(\mathscr{A}_1, f_n))$. The probability that two sets $\mathscr{A}_1$ and $\mathscr{A}_2$ have identical sketches $S_F(\mathscr{A}_1) = S_F(\mathscr{A}_2)$ is $\mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2)^n$ since the hash functions in F (and hence the min-Hashes in the sketch) are independent. Grouping min-Hashes significantly reduces the probability of false positive retrieval. The retrieving procedure then estimates $\mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2)$ only for image pairs that have at least h identical sketches out of k sketches $(F_1, \ldots, F_k)$. The probability $P(\mathscr{A}_1 \overset{h}{\sim} \mathscr{A}_2)$ that the sets $\mathscr{A}_1$ and $\mathscr{A}_2$ have at least h identical sketches out of k is given

$$P(\mathscr{A}_1 \overset{h}{\sim} \mathscr{A}_2) = \sum_{i=h}^{k} \binom{k}{i} p^{in} (1 - p^n)^{k-i}, \tag{6}$$

where $p = \mathrm{sim}_s(\mathscr{A}_1, \mathscr{A}_2)$. Note that the parameters k and n have to be fixed at the database design time, whereas h can be specified at query time. In this paper we use $h = 1$.

4 Extensions

In this section, two similarity measures inspired by the *tf-idf* weighting are introduced. The proposed extensions to min-Hash are aimed at image search and would be difficult to apply in the textual domain. In text, the vocabulary of shingles [2] (uses sequences of words rather than single words) is very large and specific. Accurate estimation of the *tf-idf* weights for such a vocabulary is not possible.

4.1 Word weighting

In this section we describe how the min-Hash method can be adopted to estimate (2). First, we make an illustrative (but inefficient) extension to the method that considers positive integer values of d_w[1]. As a second step, we show the equivalence of the illustrative method

[1]Note that assuming d_w to be integral is not a limitation. If the values of d_w were positive rational numbers, all the values can be transformed by a multiplicative constant $d'_w = Cd_w$ so that the weights d'_w are integers. Using d'_w instead of d_w does not change the similarity measure sim_w.

to a more efficient method based on generating random *non-uniform* hash functions.

For now, assume that d_w are positive integers. For each set $\mathscr{A}_i$, we construct a set $\mathscr{A}_i'$ as follows. Each element $X_w \in \mathscr{A}_i$ is represented by d_w elements X_w^k, $k = 1 \dots d_w$, in $\mathscr{A}_i'$. A min-Hash of $\mathscr{A}_i'$ is obtained as described in the previous section: random hash functions $f_j'(X_w^k)$ are used to define min-Hashes on documents $\mathscr{A}_i'$. Each element X_w^k is assigned a different value by the hash function, but all X_w^k represent the same visual word X_w. Let X_w^j be a min-Hash of $\mathscr{A}_i'$, a min-Hash of $\mathscr{A}_i$ is then defined as X_w. Again, the probability that a min-Hash of two sets $\mathscr{A}_1$ and $\mathscr{A}_2$ are identical is given by the ratio

$$\frac{|\mathscr{A}_1' \cap \mathscr{A}_2'|}{|\mathscr{A}_1' \cup \mathscr{A}_2'|} = \frac{\sum_{X_w \in \mathscr{A}_1 \cap \mathscr{A}_2} d_w}{\sum_{X_w \in \mathscr{A}_1 \cup \mathscr{A}_2} d_w}.$$

The same result is obtained when the following hash function is used on the original vocabulary

$$f_j(X_w) = \min_{k=1\dots d_w} f_j'(X_w^k).$$

In the rest of this section we derive how to generate the value of the hash function directly without generating d_w uniformly distributed random numbers for each word.

Let m_w be a random variable $m_w = \min_k r_w^k$, where $k = 1 \dots d_w$ and $r_w^k \sim \mathrm{Un}(0,1)$. The cumulative distribution of m_w is given by

$$P(m_w \leq a) = 1 - (1-a)^{d_w}. \tag{7}$$

It follows that a random uniformly distributed variable $x \sim \mathrm{Un}(1,0)$ can be transformed to a random variable with cumulative distribution function (7) as $m_w = 1 - \sqrt[d_w]{1-x}$. The expression can be further simplified using the fact that $1 - x \sim \mathrm{Un}(1,0)$ to give $m_w = 1 - \sqrt[d_w]{x}$. Note that m_w is also defined for real non-negative values of d_w. Since for the purposes of the min-Hash algorithm, only the ordering of the hashes is important, further simplification can be obtained by applying a monotonic transformations:

$$f_j(X_w) = \frac{-\log x}{d_w}, \text{ where } x \sim \mathrm{Un}(1,0). \tag{8}$$

4.2 Histogram intersection

In this section, the bag-of-words image representation will be used. We show how a new vocabulary can be constructed so that the min-Hash algorithm can be directly applied to approximate histogram intersection. Let t_i be a vector of size $|\mathscr{V}|$ where each coordinate t_i^w is a number of visual words X_w present in i-th document. Let y_w denote the highest number of occurrences of visual word X_w in a document in the database $y_w = \max_i t_i^w$. We can construct a new vocabulary $\mathscr{V}'$ as follows. For each visual word X_w the vocabulary will contain y_w *different* elements $X_w^1, \dots, X_w^{y_w}$. The bag-of-words representation t_i of a document can be equivalently represented a set $\mathscr{A}_1' \subset \mathscr{V}'$, where the set $\mathscr{A}_1'$ contains t_i^w elements representing visual word X_w: $X_w^l \in \mathscr{A}_1'$ iff $t_i^w \geq l$. For example, if an image contains two features represented by visual word X_w, elements X_w^1 and X_w^2 will be present in the set representation of that image[2].

[2]Note the difference between expanding the vocabulary in section 4.1 and here. In sim_w the number of repeated elements representing one visual word was either 0 (if the visual word was not present in the image) or d_w in all images, and the elements were indistinguishable. Here, each document can contain different number of repeated elements, depending on how many instances of the visual word appear in the image. Also, each instance is unique, elements X_w^1 and X_w^2 are different.

The min-Hash algorithm can be applied to the new set representation directly. The size of set intersection $|\mathscr{A}'_1 \cap \mathscr{A}'_2|$ is equal to $\sum_w \min(t_1^w, t_2^w)$ and the size of the set union $|\mathscr{A}'_1 \cup \mathscr{A}'_2| = \sum_w \max(t_1^w, t_2^w)$. Applying these equalities to sim_s eqn (3) we directly obtain eqn (4). The extension to weighted histogram intersection is straightforward.

5 Experimental Results

We demonstrate our method for NDID on two data sets: the TrecVid 2006 data set and the University of Kentucky data set.

It is difficult to evaluate near duplicate image retrieval, especially on large data sets. Labelling of large data sets is difficult in its own right and the subjective definition of near duplicate images complicates things further. There is no ground truth available for the TrecVid data set, hence a precise comparison of accuracy of the methods is not possible for this data. Therefore, in the first experiment we mainly focus on measuring the efficiency of the methods on a large (146k images) TrecVid data set.

To evaluate the quality of the retrieval, we present an extensive comparison of the original min-Hash method and the two proposed methods (word weighting and weighted histogram intersection) on an image retrieval database – University of Kentucky database [17] – where the ground truth is available.

The *idf* weights were used in sim_w and sim_h as word weights in our experiments.

5.1 TrecVid 2006

TrecVid [21] database consists of 146,588 JPEG keyframes automatically pre-selected from 165 hours (17.8M frames, 127 GB) of MPEG-1 news footage, recorded from different TV stations from around the world. Each frame is at a resolution of 352×240 pixels and normally of quite low quality. The frames suffer from compression artefacts, jitter and noise typically found in highly compressed video. In this experiment a vocabulary of 64K visual words, $N = 192$ min-Hashes, sketch size $n = 3$, and $k = 64$ number of sketches were used as in [4].

We measured the number of sketch hits, i.e. how many pairs of documents were considered to be near duplicates. Figure 1 displays the number of sketch hits plotted against the similarity measures of the colliding documents. For document pairs with high value of the similarity measure, the number of hits is roughly equal for sim_s and sim_w and slightly higher for sim_h. This means that about the same number of near duplicate images will be recovered by the first two methods and the histogram intersection detects slightly higher number of near duplicates. The detected near duplicate results appear similar after visual inspection and no significant discrepancy can be observed between the results of the methods.

However, for document pairs with low similarity (pairs that are of no interest) using sim_w and sim_h similarity significantly reduces the number of sketch hits. In the standard version of the algorithm, even uninformative visual words that are common to many images are equally likely to become a min-Hash. When this happens, a large number of images is represented by the same frequent min-Hash. In the proposed approach, common (non-informative) visual words are down-weighted by a low value of *idf*. As a result, a lower number of sketch collisions of documents with low similarity is observed.

The average number of documents examined per query is 8.5, 7.1, and 7.7 for sim_s, sim_w, and sim_h respectively. Compare this to $43,997.3$ of considered documents (images

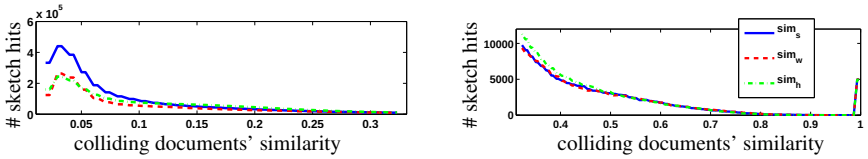

Figure 1: The number of sketch hits as a function of the document similarity for different similarity measures (TrecVid data set). Left: similarities 0 – 0.35, right: high similarities (different scale).

having at least one visual word in common) using *tf-idf* inverted file retrieval using a vocabulary of the same size.

5.2 University of Kentucky database

This database contains $10,200$ images in sets of 4 images of one object / scene. Querying the database with each image should return three more examples. This is used to score the retrieval by the average number of correctly returned images in top four results (the query image is to be retrieved too). We are probing lower values of the similarity measures due to larger variations between images of the same scene in this data set. Therefore more min-Hashes and more sketches have to be recorded.

In the experiments, we varied several parameters of the method: the size of the vocabulary (30k and 100k), the number of independent random hash functions, and the number of hashed sketches. The number of min-Hashes per sketch was set to $n = 2$. The average number of documents considered (the average number of sketch hits)[3] and the average number of correctly retrieved images in the top 4 ranked images were recorded. The results of the experiment are summarized in table 1.

The results consistently show that the number of sketch hits is significantly decreased while the retrieval score is improved when the *idf*-weighting is used. The results are further improved when the histogram intersection is used. Some example queries and results are shown in figure 2. It can be seen on the results, that sim_s often retrieves images based on the object background. The background is repeated on many images and is down-weighted by both sim_w and sim_h *idf* weighting. For comparison, the number of considered documents using standard *tf-idf* retrieval with inverted files would be $10,089.9$ and $9,659.4$ for vocabulary sizes 30k and 100k respectively.

We are not trying to compete with image or specific object retrieval. The method is designed to find images with high similarity by 'trying out' only a few possibilities. This database is too small to highlight the advantages of rapid retrieval and reduced image representation. Despite this, the scores for the histogram intersection similarity measure sim_h exceed the score of 3.16 for flat *tf-idf* scoring in [17].

Acknowledgements. We are grateful for support from EC grant 215078 DIPLECS, and the Czech Government reseach program MSM6840770038.

6 Conclusions

We have proposed two novel similarity measures whose retrieval performance is approaching the well established *tf-idf* weighting scheme for image / particular object re-

[3]The average number of sketch hits per image is computed as 2 · total number of sketch hist / number of documents. The multiplication by 2 is introduced because each hit involves two documents.

		documents considered						top 4 score					
		vocab 30k			vocab 100k			vocab 30k			vocab 100k		
mh	ske	sim_s	sim_w	sim_h	sim_s	sim_w	sim_h	sim_s	sim_w	sim_h	sim_s	sim_w	sim_h
512	256	553.8	362.2	**207.0**	143.8	87.3	**49.3**	2.54	2.54	**2.67**	2.43	2.42	**2.57**
512	512	908.6	664.1	**394.6**	281.3	181.1	**94.4**	2.70	2.72	**2.85**	2.65	2.68	**2.80**
512	1024	1671.9	1200.1	**730.9**	543.0	340.2	**178.7**	2.74	2.79	**2.94**	2.80	2.85	**2.97**
512	1536	2325.4	1626.8	**1041.3**	871.4	469.6	**260.7**	2.75	2.80	**2.96**	2.81	2.90	**3.03**
640	320	657.4	434.3	**255.6**	177.0	107.2	**60.4**	2.65	2.65	**2.77**	2.54	2.53	**2.67**
640	640	1141.9	810.2	**488.3**	340.2	206.5	**117.7**	2.76	2.81	**2.93**	2.73	2.77	**2.89**
640	1280	1924.3	1443.4	**889.4**	642.9	396.5	**225.5**	2.80	2.86	**3.01**	2.84	2.92	**3.04**
640	1920	2691.4	1949.0	**1258.4**	969.7	567.0	**330.7**	2.80	2.87	**3.02**	2.88	2.96	**3.09**
768	384	748.5	520.5	**302.8**	215.4	127.7	**72.0**	2.71	2.73	**2.84**	2.62	2.62	**2.74**
768	768	1362.3	957.0	**578.2**	419.9	244.7	**140.3**	2.83	2.86	**2.99**	2.81	2.85	**2.95**
768	1536	2242.9	1669.1	**1035.7**	761.2	637.8	**264.1**	2.85	2.90	**3.05**	2.90	2.98	**3.08**
768	2304	2978.1	2230.6	**1423.1**	1154.0	816.5	**382.1**	2.85	2.91	**3.06**	2.91	3.01	**3.13**
896	448	979.0	595.2	**352.5**	251.2	145.5	**83.6**	2.77	2.79	**2.90**	2.69	2.68	**2.80**
896	896	1578.5	1082.2	**683.8**	481.6	275.4	**163.0**	2.86	2.90	**3.03**	2.86	2.90	**3.00**
896	1792	2743.1	1878.6	**1371.7**	869.5	515.1	**318.8**	2.88	2.93	**3.08**	2.94	3.02	**3.13**
896	2688	3398.8	2496.4	**1790.8**	1238.7	734.9	**452.8**	2.87	2.93	**3.09**	2.96	3.05	**3.17**

Table 1: University of Kentucky data set. Number of min-Hashes (mh), number of sketches (ske), number of considered documents, and average number of correct images in top 4 are shown for three similarity measures sim_s, sim_w, and sim_h. Better results (lower for documents considered, higher for top 4 score) are highlighted among the methods.

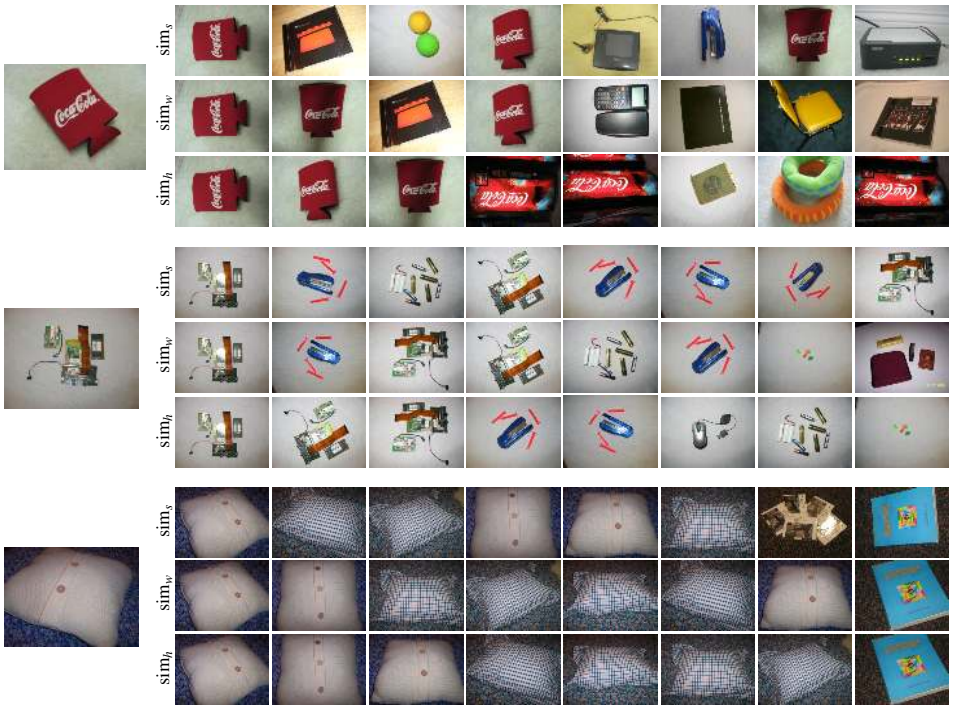

Figure 2: University of Kentucky data set: sample queries (left column), results (three rows each) for sim_s (top row), sim_w (middle row), and sim_h (bottom row).

trieval. We show that pairs of images with high values of similarity can be efficiently (in time proportional to the number of retrieved images) retrieved using the min-Hash algorithm. We have shown experimental evidence that the *idf* word weighting improves both the search efficiency and the quality of the results. The weighted histogram intersection is the best similarity measure (out of the three examined) in both retrieval quality and search efficiency. Promising results on the retrieval database encourage the use of the hashing scheme beyond near duplicate detection, for example in clustering of large database of images [3].

References

[1] M. Bertini, A. D. Bimbo, and W. Nunziati. Video clip matching using mpeg-7 descriptors and edit distance. In *CIVR*, pages 133–142, 2006.

[2] A. Broder. On the resemblance and containment of documents. In *SEQS: Sequences '91*, 1998.

[3] O. Chum and J. Matas. Web scale image clustering. Technical report, CMP, CTU in Prague, May 2008.

[4] O. Chum, J. Philbin, M. Isard, and A. Zisserman. Scalable near identical image and shot detection. In *Proc. CIVR*, 2007.

[5] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proc. ICCV*, 2007.

[6] T. C. Hoad and J. Zobel. Fast video matching with signature alignment. In *MIR*, pages 262–269, 2003.

[7] P. Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. In *IEEE Symposium on Foundations of CS*, 2000.

[8] P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In *Proc. CVPR*, 2008.

[9] H. Jegou, H. Harzallah, and C. Schmid. A contextual dissimilarity measure for accurate and efficient image search. In *Proc. CVPR*, 2007.

[10] A. Joly, O. Buisson, and C. Frélicot. Content-based copy detection using distortion-based probabilistic similarity search. *IEEE Transactions on Multimedia*, to appear, 2007.

[11] A. Joly, C. Frelicot, and O. Buisson. Robust content-based video copy identification in a large reference database. In *Proc. CIVR*, 2003.

[12] Y. Ke, R. Sukthankar, and L. Huston. Efficient near-duplicate detection and sub-image retrieval. In *ACM Multimedia*, 2004.

[13] J. Law-To, A. Joly, L. Joyeux, N. Boujemaa, O. Buisson, and V. Gouet. Video and image copy detection demo. In *Proc. CIVR*, 2007.

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[15] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proc. BMVC.*, pages 384–393, 2002.

[16] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 1(60):63–86, 2004.

[17] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. CVPR*, 2006.

[18] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007.

[19] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003.

[20] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Proc. CVPR*, 2008.

[21] TRECVID. http://trecvid.nist.gov/.

[22] D. Zhang and S. Chang. Detecting image near-duplicate by stochastic attributed relational graph matching with learning. In *ACM Multimedia*, 2004.

[23] J. Zhou and X.-P. Zhang. Automatic identification of digital video based on shot-level sequence matching. In *ACM MM*, pages 515–518, 2005.