Binary SIFT: Towards Efficient Feature Matching Verification for Image Search

Wengang Zhou[1], Houqiang Li[2], Meng Wang[3], Yijuan Lu[4], Qi Tian[1]
Dept. of Computer Science, University of Texas at San Antonio[1], Texas, TX 78249
Dept. of EEIS, University of Science and Technology of China[2], Hefei, P.R. China, 230027
School of Computer and Information, Hefei University of Technology[3], P.R. China, 230009
Dept. of Computer Science, Texas State University[4], Texas, TX 78666
zhwgeeis@gmail.com[1], lihq@ustc.edu.cn[2], eric.mengwang@gmail.com[3], yl12@txstate.edu[4], qitian@cs.utsa.edu[1]

ABSTRACT
Recently, great advance has been made in large-scale content-based image search. Most state-of-the-art approaches are based on the Bag-of-Visual-Words model with local features, such as SIFT. Visual matching between images is obtained by vector quantization of local features. Two feature vectors from different images are considered as a match, if they are quantized to the same visual word, even though the L_2-distance between them is large. Thus, it may introduce many false positive matches. To address this problem, in this paper, we propose to generate binary SIFT from the original SIFT descriptor. The L_2-distance between original SIFT descriptors is demonstrated to be well kept with the metric of Hamming distance between the corresponding binary SIFT. Two feature vectors quantized to the same visual word are considered as a valid match only when the Hamming distance between their binary SIFT vectors is below a threshold. With our binary SIFT, most false positive matches can be effectively and efficiently identified and removed, which greatly improves the accuracy of large-scale image search.

We evaluate the proposed approach by conducting partial-duplicate image search on a one-million image database. The experimental results demonstrate the effectiveness and efficiency of our scheme.

Categories and Subject Descriptors
I.2.10 [**Vision and Scene Understanding**]: VISION

General Terms
Algorithms, Experimentation, Verification.

Keywords
Keywords are your own designated keywords.

1. INTRODUCTION
In recent years, great advance has been made in large-scale content-based image retrieval [1, 2, 3, 4, 6, 7, 8, 24]. Two kinds of work make major contribution to it. The first one is the introduction of local invariant feature, involving interest point detector and local patch descriptor. Popular interesting point detectors include difference of Gaussian (DoG) [5], MSER [14], and Hessian affine [15], *etc*. Local patch descriptors make a representation of the local appearance around interest points. Well-acknowledged descriptors include SIFT [5], SURF [16], *etc*. The second work is the Bag-of-Visual-Words (BoW) model [1] leveraged from information retrieval [11]. With BoW model, an image is compactly represented with a "bag" of visual words, and can be efficiently indexed with an inverted file structure for on-line query. Usually, local features in images are quantized to visual words by vector quantization.

In essence, image search has to address the problem of visual matching between images. When images are represented with local SIFT features, image matching is realized by visual matching of local features. In large-scale image search, two features from different images are considered as a match, if they are quantized to the same visual word. Since the dimension of SIFT feature space is as high as 128, the sub-spaces corresponding to visual words are still likely to be large even with millions of visual words, *i.e.*, the SIFT feature space is divided into millions of sub-spaces. As a result, we frequently observe that, SIFT features quantized to the same visual word may have large Euclidean distance between each other, which causes many false positive feature matches and consequently degrades the accuracy of image search. Generally, in visual matching with SIFT descriptors [5], it is the vector distance between two SIFT descriptors that should be used to determine whether they are likely to be a true match. Therefore, it is very necessary to further check the distance between SIFT features after vector quantization. However, it is infeasible to store SIFT descriptors in an inverted index file as it will cast too high cost in memory. Besides, it is also not efficient to compute vector distance with the original SIFT descriptors. Therefore, some compact representations of SIFT descriptor are desired.

In literature, there are some works, such Hamming Embedding [8, 12], converting SIFT descriptor to binary signature to remove false SIFT matches. However, to our best knowledge, none of them explicitly demonstrate the Hamming distance from binary signature is consistent to the L_2-distance of SIFT descriptor. As a result, the improvement from current works [7, 8, 12] is limited.

In this paper, we propose a new scheme to transform a SIFT descriptor to a binary bit stream, called binary SIFT. Extensive

ICIMCS'12, Sep. 9–11, 2012, Wuhan, Hubei, China.

study with large-scale (trillion) samples reveal that the generated binary SIFT effectively keeps the distance metric of the original SIFT descriptor. We apply the binary SIFT to large-scale image search. To adapt to the classic BoW model for large-scale image search, the binary SIFT is stored in the inverted file list. During online retrieval, for each query feature quantized to a visual word, we further compare its binary SIFT with those in the inverted file list following the visual word. Only those features with sufficiently small enough Hamming distance from the query feature are regarded as true matches. Since the main computation in Hamming distance is logic operation, the computational cost is low. Experiments on image search in million-scale dataset demonstrate the effectiveness of the proposed scheme.

The rest of the paper is organized as follows. Section 2 reviews related work in literature. Section 3 discusses our approach in details. Section 4 shows the experimental results. Finally, conclusion is given in Section 5.

2. RELATED WORK

In literature, there are lots of works on large-scale content-based image retrieval. Most of them are based on BoW model and utilize local invariant features, such as SIFT [5], for image representation. Since local features are of high-dimensional and an image may contain hundreds or thousands of local features, vector quantization is popularly applied to quantize a local feature to a visual word. Consequently, an image is compactly represented by a "bag" of visual word ID, which effectively adapts to the classic inverted index structure for real-time retrieval. To date, many algorithms have been proposed to improve different stages of the classic image retrieval framework. In the following, we make a review of related work on feature quantization, hashing, and post-processing.

In feature quantization, *k*-means is widely used to cluster feature samples to generate visual words for feature quantization [1]. In [3], a hierarchical visual vocabulary tree structure is adopted to greatly increase the quantization efficiency. In [6], instead of quantizing one feature to one visual word, each SIFT is mapped to and represented by multiple nearest visual words. It effectively alleviates the quantization loss but with high computational cost. In [8], to reduce the quantization error, an interesting binary signature is used to verify features quantized to the same visual word. The binary signature is generated with a thresholding vector computed with large training samples for each visual word, respectively. In [12], an asymmetric version of Hamming Embedding is developed by exploiting the precise query location instead of the binarized query vector. In [7], the high dimensional SIFT descriptor space is partitioned into regular lattices. In [13], a novel scheme is proposed to aggregate all local features in an image with a very small codebook, and complex quantization techniques are exploited for indexing. In [10], a novel quantization method based on randomized trees is introduced to build visual vocabulary. The conjunction of randomized *k*-d trees creates an overlapping partition of the SIFT feature space and helps to mitigate quantization error. In [24], visual (code) words are generated with the first 32 bits from scalar quantization.

Some algorithms design better hashing techniques for visual word vectors. In [17, [20], an interesting min-hash scheme is proposed to independently select a set of visual words from an image as global descriptors and define image similarity as the set overlap. It is based on the philosophy that the more common features in two images, the higher the probability of having the same min-Hash. Such scheme is very effective and efficient in detection of near identical images and video shots. In [18], geometric min-hash exploits local spatial context to construct repeatable hash keys and increase the discriminability of the description.

Some other schemes improve the image search performance in the post-processing stage. In [1], local spatial consistency is imposed to filter visual-word matches with low support. In [8], weak geometric consistency of SIFT orientation and scale is used to remove potential false matches. In [10], global spatial verification is performed to estimate an affine model [19] to filter local matches. In [2], geometric context is represented with coding maps. It recursively removes geometrically inconsistent matches by analyzing those coding maps. In [21], geometric coding improves [2] by generating coding maps with full use of SIFT orientation, scale and key point location. The obtained coding maps are invariant to translation, rotation and scale changes. Besides the above spatial verification techniques, query expansion is another important post-processing scheme. It reissues the initial highly-ranked results to generate new queries so as to improve recall performance. General techniques, such as average query expansion, transitive closure expansion, resolution expansion, are discussed in [4]. In [9], two novel expansion strategies, *i.e.*, intra-expansion and inter-expansion, are proposed. Intra-expansion expands more target feature points similar to those in the query, while inter-expansion explores those feature points co-occurring with the search targets but not present in the query.

In this paper, our focus is the feature quantization stage. We propose a binary SIFT for efficient and effective feature matching verification for large-scale image search. Unlike the binary signature in [8], our binary SIFT is independent of image collection and is demonstrated to keep the vector distance of SIFT descriptor. Our approach can also be easily integrated with those hashing algorithms and post-processing approaches discussed above to achieve better retrieval performance. On the other hand, it should be noted that the binary SIFT matching scheme is not limited to image search. It can also be applied in video tracking, image registration, *etc*.

3. OUR APPROACH

In Section 3.1, we first discuss how to generate binary SIFT and demonstrate that the vector distance of SIFT descriptor is kept with Hamming distance of binary SIFT. In Section 3.2, we will introduce how to embed the binary SIFT into the image search framework. Some details in index and retrieval are also discussed.

3.1 Binary SIFT Generation

The standard SIFT descriptor is extracted from a local image patch by concatenating 8-D orientation histograms of all 16 (4 by 4) sub-patches. We observe that the coefficients in most bins of SIFT descriptor vector are very stable even under various changes in rotation and scaling or noises addition. Such property accounts for its discriminative power in visual identification. In other words, the differences between bins and a predefined threshold are stable for most bins. Based on such observation, we propose a simple scheme to convert a standard SIFT descriptor to a binary bit stream, *i.e.*, binary SIFT (BSIFT).

Given a SIFT descriptor vector $f = (f_1, f_2, \cdots, f_d)^T \in R^d$, where $f_i \in R, (i = 1,2,\cdots,d)$, $d = 128$, we transform f to a bit vector (binary SIFT, or BSIFT) $b = (b_1, b_2, \cdots, b_d)^T$, as follows:

$$b_i = \begin{cases} 1 & \text{if } f_i > \hat{f} \\ 0 & \text{if } f_i \le \hat{f} \end{cases} \qquad (i = 1,2,\cdots,d) \qquad (1)$$

where $\hat{f}$ is the median value of $(f_1, f_2, \cdots, f_d)^T$.

It should be noted that, our binary SIFT generation is significantly different from the binary signature defined in Hamming Embedding [8]. Hamming Embedding has to train a thresholding vector with large training samples for each visual word, respectively. In contrast, our binary SIFT doesn't involve any training step. It is independent of image collection. And it is simple and computationally efficient.

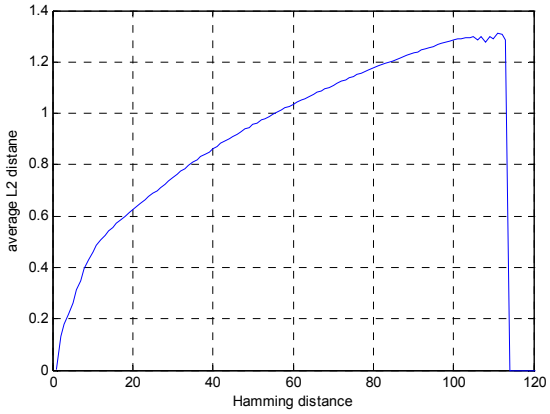
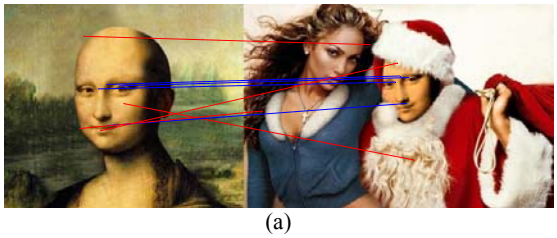


Figure 1. The average L_2-distance vs. Hamming distance. The statistics are obtained based on 400 billion pairs of SIFT descriptors. The L_2-distance is computed with SIFT descriptors unit-normalized.



(a)

(b)

Figure 2. Matching results verification by binary SIFT (t = 18). The initial matches are obtained with vector quantization. The red line segments denote those false matches identified by binary SIFT verification, while blue ones denote true matches passing matching verification. Both images in (a) contain the duplicate patch of Mona Lisa face. (Best viewed in color PDF)

To demonstrate that the discriminative power of SIFT descriptors is well kept in the transformed binary SIFT, we have made a statistical study on over 400 billion SIFT descriptor pairs, which take every SIFT pair extracted from image pairs randomly sampled from a large image dataset. For each descriptor pair, its L_2 distance on the standard SIFT and Hamming distance on the binary SIFT are calculated. From Fig. 1, we can observe that the Hamming distance between binary SIFT is consistent with the average L_2-distance. (The drop of L_2 distance after the Hamming distance grows over 114 is due to the fact that there is no binary SIFT features with that large Hamming distance.) In other words, the Hamming distance between binary SIFT can be used to approximate the Euclidean distance between the corresponding SIFT descriptors. Therefore, we can use the binary SIFT instead of the original SIFT descriptor to check the distance between two candidate features. Another advantage of binary SIFT is that, the memory cost of binary SIFT is low, making it feasible to store the whole binary SIFT in the index list.

3.2 Image Search with BSIFT Verification

Our image search method is based on the Bag-of-Visual-Words model. We construct a quantizer by hierarchically clustering large-scale SIFT descriptor samples. The clustering leaf nodes are considered as visual words. The obtained visual vocabulary tree is used to quantize a SIFT feature to a visual word.

In the traditional approach [1, 2, 3], two SIFT features from two images are considered as a match, if they are quantized to the same visual word. In our implementation, a further binary SIFT verification is performed. That is, the hamming distance between two binary SIFT features is no greater than a threshold t. The impact of threshold t will be studied in Section 4.1.

Fig. 2 shows two examples of feature matching based on binary SIFT. It can be observed that false local matches exist on both relevant and irrelevant image pairs after feature quantization. With further verification on binary SIFT, most false matches can be identified and removed. More matching sample results are shown in Fig. 6.

We formulate the image search as a match voting scheme. For each SIFT feature in the query image, each verified feature match will cast a vote to the corresponding image of the database. The similarity between images is defined by the cardinality of matched feature set. Consequently, the database images are ranked by their similarity scores and returned as the image retrieval results.

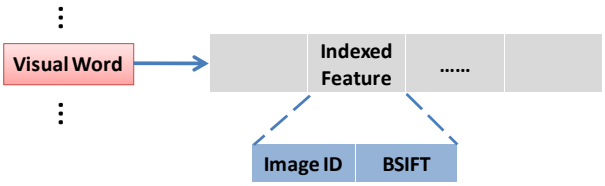


Figure 3. Inverted file structure.

In our image search scheme, the binary SIFT is indexed with an inverted file structure for large-scale image database, as illustrated in Fig. 3. Each visual word is followed by a list of entries and each entry contains the ID of images in which the visual word appears. Besides, for each indexed feature, we also store its binary SIFT (BSIFT). With the inverted index structure, we only need

check those images sharing common visual words with the query image and therefore achieve real-time response.

4. EXPERIMENTS

We build our basic dataset with one million images crawled from the Web. Images in the basic dataset are used as distracters. We take the partial-duplicate image dataset in [17] as the ground-truth dataset, which contains 1104 images from 33 groups, including "Mona Lisa", "American Gothic Painting", "Seven-eleven logo", *etc*. From the ground truth dataset, 108 representative query images are randomly selected for evaluation comparison. Mean average precision (mAP) is selected to measure the accuracy performance of all methods.

We select the standard SIFT feature [1] implemented with an open-source library [22] for image representation. Key points are detected with the Difference-of-Gaussian (DoG) detector. A 128-D orientation histogram (SIFT descriptor) is extracted to describe the visual appearance of local patch around each key point. Before extracting SIFT features, large images are scaled to have a maximum axis size of 400.

4.1 Parameter Impact

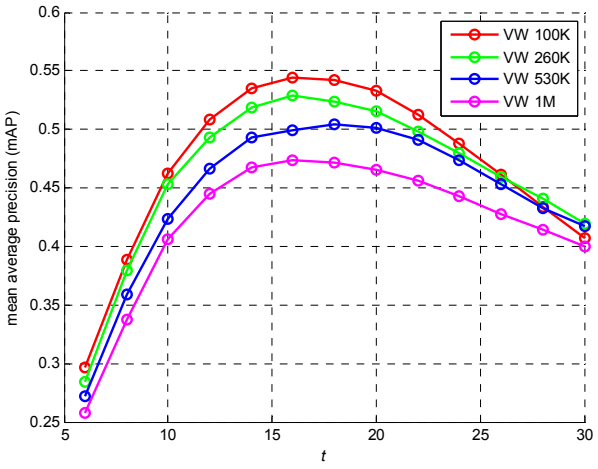
There are two parameters in our approach: visual codebook size and Hamming distance threshold *t*. To evaluate their impact on search accuracy and efficiency, we test different parameter settings with all query images on the one-million image database. Four visual codebooks with different sizes, *i.e.*, VW 100K, VW 260K, VW 530K and VW 1M, are involved for evaluation. VW 100K represents a visual codebook with 100 thousand visual words. The results are shown in Fig. 4.

From Fig. 4(a), it is observed that, when the visual codebook size decreases, the search accuracy increases. This is because, smaller visual codebook will keep more true matches, and the false matches can be effectively removed with our binary SIFT verification. On the other hand, when the Hamming threshold *t* increases, the accuracy first increases to a peak, and then drops gradually. This is because smaller *t* introduces more false negatives while larger *t* incurs more false positives. From Fig. 4(b), it is shown that the time cost increases sharply when the visual codebook size decreases. This is due to that more indexed features have to be checked with smaller visual codebook.

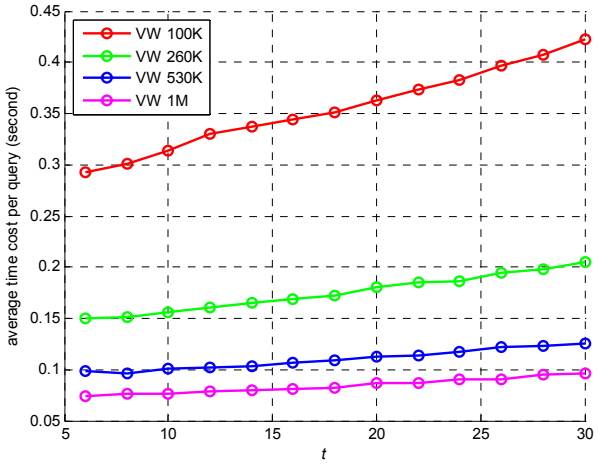
To make a tradeoff between accuracy and efficiency, in our implementation, we select the visual codebook with 530K visual words and choose the Hamming threshold *t* as 18.

4.2 Evaluation

Comparison Algorithms: We compare our approach with three state-of-the-art feature quantization algorithms in large-scale image search. The BoW approach with classic visual vocabulary tree [3] is selected as the "baseline" method. We test various sizes of visual vocabulary for the baseline, and find the one with 1-million visual words gives the best overall performance. To enhance the baseline, two other algorithms, *i.e.*, soft assignment [13] and Hamming Embedding (HE) [6], are also included for comparison. We implement these three comparison algorithms based on the original papers. Since our focus in this paper is feature quantization, the weak geometric consistency verification in [6] is not included in the implementation of HE. In the implementation of soft assignment approach [13], the approximate nearest neighbor search is developed with the ANN library [23].



(a)



(b)

Figure 4. Parameter impact on different sizes of visual codebooks. (a) Mean average precision; (b) Average time cost per query. (Best viewed in color PDF)

Accuracy: From Table 1, it can be observed that our approach outperforms all the other three methods on large image databases. On the 1-million dataset, the mAP of the baseline is 0.376. Our approach hits 0.505, a realtively 25.5% improvement. Since Hamming codes can effectively filter false features, the Hamming Embdding approach achieves a mAP of 0.429, but still 7.6% lower than our approach. The mAP improvement of soft assignment approach is higher than HE. It reaches a mAP of 0.482. Compared with soft assignment, our approach still enjoys a relatively 4.8% improvement.

Efficiency The experiments are performed on a server with 3.4 GHz CPU and 16 GB memory. Fig. 5 shows the average online search time cost per query of all four approaches. The time cost on SIFT feature extraction is not included. It takes the baseline 0.12 second in average to perform one query. Although HE is the most time-efficient one and costs only 0.05 second to finish one online query in average, it suffers more expensive off-line training process since it has to additionally train thresholding vectors for each visual word. Soft assignment is the most time-consuming approach, consuming 0.52 second in average per

query. The efficiency of our approach is comparable to the baseline, with 0.11 second on average per query. It is much faster than the soft assignment approach and saves offline training cost compared with HE.

Table 1. Performance (mAP) comparison of different methods on 1-million image database.

	Baseline	HE	Soft assignment	Our approach
mAP	0.376	0.429	0.482	**0.505**

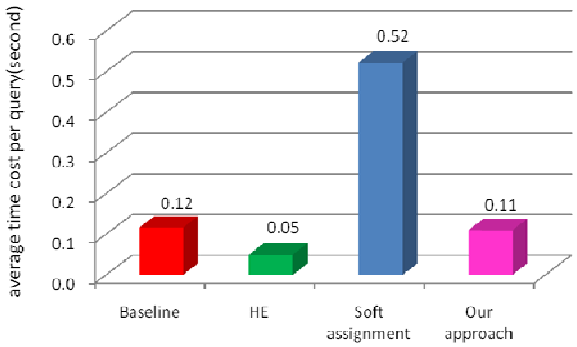


Figure 5. Comparison of average query time cost of different methods on the 1M database. (Not including the time cost for SIFT feature extraction)

Table 2. Memory cost per indexed feature for four approaches.

	Baseline	HE	Soft assignment	Our approach
Memory cost per feature (byte)	8	12	24	20

Memory Cost: The memory cost is linear to the number of features to be indexed. Therefore, we compare memory cost per feature on all four approaches, as shown in Table 2. For each feature, the baseline approach needs 4 bytes to store image ID and another 4 bytes to store the *tf-idf* weight. The soft assignment has to store each indexed feature in three visual word lists. Therefore it costs 24 bytes, three times the memory cost of the baseline approach. In Hamming Embedding approach, for each feature it allocates 4 bytes on image ID and 8 bytes on the 64-bit binary signature. In our approach, besides the 4 bytes for image ID, another 16 bytes are needed to store the 128-bit binary SIFT.

4.3 Sample Results

We show more sample matching results in Fig. 6. We can see that even irrelevant images can share many local feature matches from vector quantization. With verification on binary SIFT, most false positive matches can be removed. In Fig. 7, we show some retrieval results of four query images with our scheme. It can be observed that our approach can effectively retrieve those target images with large editing in scale, rotation, illumination, and clutter background.

Figure 6. Matching results verification by binary SIFT (t = 18). The initial matches are obtained by vector quantization with VW 500K. Red line: false matches identified by binary SIFT verification; blue lines: true matches passing matching verification. (left) irrelevant image pairs; (right) relevant image pairs. (Best viewed in color PDF)

5. CONCLUSION

In this paper, we present a binary representation of SIFT feature for SIFT matching verification. Study with large-scale samples demonstrates that binary SIFT preserves the quality of vector comparison on the original SIFT descriptor. Inverted file structure is used for large-scale indexing and the binary SIFT is stored in inverted file list for feature matching verification. Experiments on image search with large-scale database reveal the effectiveness of the proposed approach.

Our binary SIFT generation is independent of image collection. Therefore, it does not involve any training and can be efficiently generated. Besides, comparison between binary SIFT can be achieved with efficient exclusive OR operation.

In our next work, we will study the gap between vector quantization and visual matching in large-scale image search. We will also investigate better strategies to transform SIFT to binary version preserving the quality of vector comparison.

6. ACKNOWLEDGMENTS

This work was supported in part to Dr. Li by NSFC general project "Intelligent Video Processing and Coding Based on Cloud Computing", in part to Dr. Lu by Research Enhancement Program (REP), start-up funding from the Texas State University and DoD HBCU/MI grant W911NF-12-1-0057, and in part to Dr. Tian by ARO grant W911NF-12-1-0057, NSF IIS 1052851, Faculty Research Awards by Google, NEC Laboratories of America and FXPAL, UTSA START-R award, respectively.

7. REFERENCES

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

[2] W. Zhou, Y. Lu, H. Li, Y. Song, and Q. Tian. Spatial coding for large scale partial-duplicate Web image search. In *Proc. ACM Multimedia*, 2010.

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

[4] 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.

[5] D. Lowe. Distinctive image features form scale-invariant keypoints. *IJCV*, 20(2): 91-110, 2004.

[6] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. CVPR*, 2008.

[7] T. Tuytelaars, C. Schmid. Vector quantizing feature space with a regular lattice. In *Proc. ICCV*, 2010.

[8] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proc. ECCV*, 2008.

[9] Y. Kuo, K. Chen, C. Chiang, and W. H. Hsu. Query expansion for hash-based image object retrieval. In *Proc. ACM Multimedia*, 2009.

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

[11] R. Baeza-Yates and B. Ribeiro-Neto. Modern information retrieval. *ACM Press*, ISBN: 020139829, 1999.

[12] M. Jain, H. Jegou and P. Gros. Asymmetric Hamming embedding: taking the best of our bits for large scale image search. In *Proc. ACM Multimedia*, 2011.

[13] H. Jegou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proc*. CVPR 2010.

[14] J. Matas, O. Chum, U. Martin, T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proc. BMVC*, 2002.

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

[16] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded up robust features. In *Proc. ECCV*, 2006.

[17] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-Hash and tf-idf weighting. In *Proc. BMVC*, 2008.

[18] O. Chum, M. Perdoch, and J. Matas. Geometric min-Hashing: finding a (thick) needle in a haystack. In *Proc. CVPR,* 2009.

[19] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24: 381–395, 1981.

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

[21] W. Zhou, H. Li, Y. Lu, and Q. Tian. Large scale image search with geometric coding. In *Proc. ACM Multimedia*, 2011.

[22] R. Hess. An open-source SIFT library. In *Proc. ACM Multimedia*, 2010.

[23] S. Arya and D. Mount. Ann: Library for approximate nearest neighbor searching. *Available at* http://www.cs.umd.edu/~mount/ANN/.

[24] W. Zhou, Y. Lu, H. Li, and Q. Tian. Scalar quantization for large scale image search. In *Proc. ACM Multimedia*, 2012.

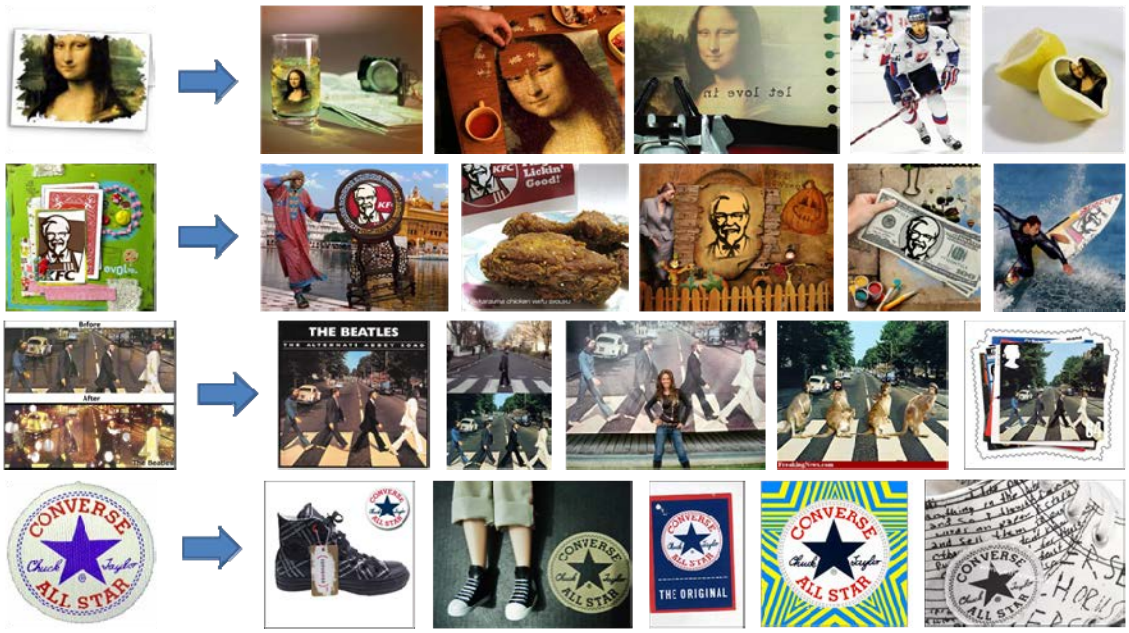


Figure 7. Example results from retrieval. In each row, queries are shown on left of the arrow, while highly-ranked images (selected from those before the first false positive result) from the query results are shown on the right of the arrow.