# COMPONENT HASHING OF VARIABLE-LENGTH BINARY AGGREGATED DESCRIPTORS FOR FAST IMAGE SEARCH

*Zhe Wang*[‡] *Ling-Yu Duan*[‡] *Jie Lin*[‡] *Tiejun Huang*[‡] *Wen Gao*[‡] *Miroslaw Bober*[†]

[‡]The Institute of Digital Media,School of EE&CS, Peking University, Beijing, 100871, China
[†]Centre for Vision, Speech, and Signal Processing, University of Surrey, Guildford, GU2 7XH, UK
{zwang,lingyu,jielin,tjhuang,wgao}@pku.edu.cn m.bober@surrey.ac.uk

## ABSTRACT

Compact locally aggregated binary features have shown great advantages in image search. As the exhaustive linear search in Hamming space still entails too much computational complexity for large datasets, recent works proposed to directly use binary codes as hash indices, yielding a dramatic increase in speedup. However, these methods cannot be directly applied to variable-length binary features. In this paper, we propose a Component Hashing (CoHash) algorithm to handle the variable-length binary aggregated descriptors indexing for fast image search. The main idea is to decompose the distance measure between variable-length descriptors into aligned component-to-component matching problems independently, and build multiple hash tables for the visual word components. Given a query, its candidate neighbors are found by using the query binary sub-vectors as indices into their corresponding hash tables. In particular, a bit selection based on conditional mutual information maximization is proposed to reduce the dimensionality of visual word components, which provides a light storage of indices and balances the retrieval accuracy and search cost. Extensive experiments on benchmark datasets show that our approach is 20∼25 times faster than linear search, without any noticeable retrieval performance loss.

***Index Terms***— Image Search, Aggregated Descriptors, Variable-length Binary Codes, Component Hashing

## 1. INTRODUCTION

The problem of large-scale image retrieval concerns the search of similar images containing a rigid object in a large set of database images, given a query image of that object. The key challenge is how to jointly optimize the retrieval accuracy, the search efficiency and the compactness of visual descriptors. To this end, patch-level binary features, such as LDAHash [5], BRIEF [6], ORB [3] and BRISK [4], provide an attractive alternative to the widely used SIFT [11] and SURF [12] as they support fast Hamming distance matching as well as light transmission and storage. Recent works [8][9] proposed to compress high-dimensional descriptors aggregated from local invariant features (e.g., SIFT [11]) into image-level binary signature, bringing significant speedup without noticeable loss of retrieval performance. For instance, the Compressed Fisher vector (CFV) [8] and Residual Enhanced Visual Vector (REVV) [9] proposed to assign local features to nearest visual words in a visual vocabulary and aggregate the statistics of local features (e.g., visual word residuals) into a fixed-length vector representation, followed by element-wise sign binarization to produce low bitrate binary signature.

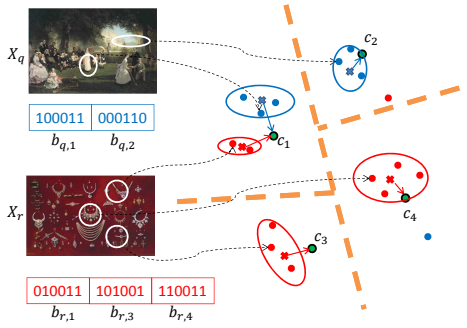



**Fig. 1**: Illustration of variable-length binary aggregated descriptors based on the statistics of local features. $C_i$: the i-th visual word. $[b_{q,1},b_{q,2}]$: the variable-length binary aggregated descriptor of image $X_q$. $[b_{r,1},b_{r,3},b_{r,4}]$: the variable-length binary aggregated descriptor of image $X_r$.

Apart from compactness, descriptors should allow length adaptation to balance the required performance level and database size descriptor scalability [1]. Moreover, variable-length compact descriptors can accommodate the bandwidth variation in wireless network for low latency query delivery in mobile visual search scenarios [7]. In particular, this topic relates to an ongoing MPEG standardization, namely, Compact Descriptors for Visual Search (CDVS) [1]. Our prior work [13][20] introduces a Rate-adaptive Compact Fisher Codes (RCFC) to produce a variable-length binary aggregated descriptors by progressively encoding informative visual words, till the bit budget has been fully occupied. Another variable-length aggregated descriptor is the Robust Visual Descriptor (RVD) proposed in [19]. In this paper, we study the problem of fast image search with binary descriptors, especially the variable-length binary aggregated descriptors (see Fig.1).

Even though the Hamming distance can be computed very quickly, the exhaustive linear scan between the query and each database image is computational expensive. The efficiency problem becomes more serious for very large datasets. To alleviate the problem, hashing techniques such as semantic hashing [17] and spectral hashing [16] use binary codes as indices of hash table directly. Nearest neighbors can be found by exploring the hash buckets within a search radius $r$ around the query. As the number of buckets to be examined grows near exponentially with the radius, this is only applicable to short binary codes (less than 32 bits). Recently, Norouzi

Ling-Yu Duan is the corresponding author of this paper. This work was supported by National Natural Science Foundation of China under grant 61271311, 61121002, 61390515, and 61210005.

et al. [15] presented the Multi-Index Hashing (MIH) to perform fast image search with longer codes. To ensure optimal search speed, MIH contiguously partitions the binary vector with $l$ bits into $m$ disjoint sub-vectors, and indexes them into $m$ different hash tables respectively. Each sub-vector has $\log_2 N$ bits and $m = l/\log_2 N$, where N denotes the database size. For visual search, candidate neighbors are found by using the query sub-vectors as indices into their corresponding hash tables, followed by a descriptor comparison using the entire binary codes to rerank the candidates and return the exact $r$-neighbors. Compared to previous hashing techniques, search cost is dramatically reduced as the search radius for each hash table is reduced to $\lfloor r/m \rfloor$, resulting in a small number of buckets to be checked. However, a common issue with these methods is that they rely on fixed-length binary vectors and consequently fail to handle variable-length presentations, making it unsuitable for the emerging rate-adaptive descriptors in mobile visual search applications.

In this paper, we propose a Component Hashing (CoHash) of variable-length binary aggregated descriptors for fast image search. The main idea is to decompose the distance measure between variable-length binary aggregated descriptors into aligned component-to-component independent matching problems, and build multiple hash tables for these components. In the context of image search, we define the component as visual word obtained by k-means or Gaussian Mixture Model (GMM) clustering. Given a query, the online search follows a similar pipeline as MIH on each component independently. More importantly, if the dimensionality of components is large, it requires huge amount of memory to store the hash table as well as incurs more search cost to find the candidates. A bit selection based on conditional mutual information maximization is proposed to reduce the dimensionality of components. Our results have verified that it provides a light storage of indices and balances the retrieval accuracy and the search speed. To the best of our knowledge, this is the first work to study the hashing algorithm for variable-length vectors in Hamming space.

This paper is organized as follows. We give an introduction to the variable-length binary aggregated descriptors in Section 2. Section 3 presents the proposed CoHash algorithm. Experimental results are presented in Section 4. Finally, we conclude this paper in Section 5.

## 2. VARIABLE-LENGTH BINARY AGGREGATED DESCRIPTORS

State-of-the-art binary aggregated descriptors typically consist of three steps: feature coding, feature aggregation and feature compression. Let $X_n = \{\mathbf{x}_t\}_{t=1}^{T}$ denote a collection of $d$-dimensional local features $\mathbf{x}_t$ from query image $X_n$, the goal of feature coding is to embed local features $\mathbf{x}_t$ in a visual vocabulary space based on an encoder $q(\mathbf{x}_t)$: $\mathbf{x}_t \in \mathbf{R}^d \rightarrow q(\mathbf{x}_t) \in \mathbf{Q}$, where $\mathbf{Q} = \{\mathbf{c}_1, ..., \mathbf{c}_K\}$ is a visual vocabulary comprising $K$ visual words. The encoder $q(\mathbf{x}_t)$ is to assign each local feature $\mathbf{x}_t$ to its nearest visual words $NN(\mathbf{x}_t) \in \mathbf{Q}$ in Euclidean space. After that, feature aggregation converts the statistics of local features into a fixed-length image-level vector representation. Specifically, for each visual word $\mathbf{c}_k$, we derive its statistics $\mathbf{g}_{n,k}$ by accumulating the residual vector $u(\mathbf{x}_t)$ between $\mathbf{c}_k$ and the local features $\mathbf{x}_t$ assigned to it: $\mathbf{g}_{n,k} = \sum_{\mathbf{x}_t : NN(\mathbf{x}_t)=\mathbf{c}_k} u(\mathbf{x}_t)$. The aggregated descriptors $\mathbf{g}_n$ are formed by concatenating the sub-vectors $[\mathbf{g}_{n,1}, ..., \mathbf{g}_{n,K}]$ of all visual words. Feature compression aims to compress high dimensional aggregated descriptors $\mathbf{g}_n$ into binary codes. For instance, the CFV, RCFC, RVD and REVV proposed to quantize each dimension of $\mathbf{g}_n$ into a single bit based on a sign function, which converts any positive value to 1 and any non-positive value to 0. The size of the binarized aggregated descriptors denoted as $\mathbf{b}_n = [\mathbf{b}_{n,1}, ..., \mathbf{b}_{n,K}]$ is therefore $Kd$ bits.

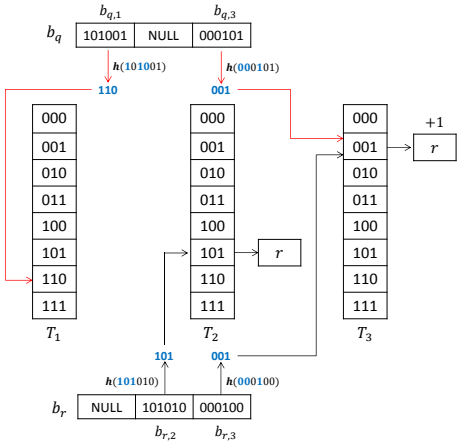



**Fig. 2**: Component Hashing of variable-length binary aggregated descriptors. $b_q$: variable-length binary aggregated descriptor of query image $q$. $b_r$: variable-length binary aggregated descriptor of database image $r$. $h(.)$: function of bit selection, the blue bits are selected. $T_i$: the i-th hash table.

Since commonly there are many outlier local features detected in images, it is beneficial to reject noisy visual words where we expect low reliability. A visual word reliability can be estimated by the distance distribution from local features to this visual word. For example, the RCFC [20] computes the reliability score $s_k$ as the maximum probability of local features $\mathbf{x}_t$ being generated by Gaussian $k$, while the RVD [19] calculates it as the number of local features associated with the $k^{th}$ visual word weighted at each rank of neighborhood. In an extreme case, if no local feature is assigned to visual word $\mathbf{c}_k$, the corresponding reliability score $s_k = 0$ and all the elements of the corresponding sub-vector $\mathbf{g}_{n,k}$ are zero. It means that the visual word is less informative for describing visual content of the image. Finally, the variable-length binary aggregated descriptors $\mathbf{b}_n^{'}$ are formed by a selected subset of visual words ($K^{'} < K$), subject to a bitrate constraint $K^{'}d$. Accordingly, the resulting bitstream has an overhead of $K$ bits to indicate which visual word representations are used [19][20].

**Problem definition**. With ultra-fast Hamming distance computation (XOR operation and bit count), the distance measure between two variable-length binary aggregated descriptors(representing query image $X_q$ and database image $X_r$) can be simply computed as the accumulation of aligned component-to-component distances:

$$d(\mathbf{b}_q^{'}, \mathbf{b}_r^{'}) = \sum_{k=1}^{K} b_{q,k} b_{r,k} h(\mathbf{b}_{q,k}, \mathbf{b}_{r,k}), \quad (1)$$

where $h(\cdot,\cdot)$ is the Hamming distance between binarized sub-vectors. $b_{q,k} = 1$ if the sub-vector $\mathbf{b}_{q,k}$ of the $k^{th}$ component (i.e., visual word) is selected; otherwise, $b_{q,k} = 0$.

On the one hand, the computation cost of this exhaustive search increases linearly with the dimensionality $d$, the number of visual words $K$ and the database image size. This still entails too much computational complexity, hashing techniques are required to index-

ing the feature vectors for fast search. On the other hand, existing hashing algorithms like Locality Sensitive Hashing (LSH) [14] and MIH assume that the lengths of feature vectors are fixed, which cannot directly apply to the variable-length binary vectors.

## 3. COMPONENT HASHING

In this section, we firstly present the proposed CoHash algorithm showing how to offline construct the hash tables for variable-length binary aggregated descriptors, and then introduce the online searching strategy (see Fig. 2). Finally, we analyze the complexity of CoHash.

**Hash table construction**. Given $N$ database images $\{X_n\}_{n=1}^{N}$, their variable-length binary aggregated descriptors $\mathbf{b}_n^{'}, n = 1, ..., N$ are traversed to construct multiple index tables $T$ based on the components. We treat each $d$-dimensional visual word as a component and set up a hash table $T_k (1 \leq k \leq K)$ for each component. Each hash table, following an inverted table structure, is built up as follows: for the variable-length binary aggregated descriptors $\mathbf{b}_n^{'}$ of the $n^{th}$ database image, $\mathbf{b}_{n,k}$ denotes the binary codes of its $k^{th}$ component, we add its descriptor-ID $n$ into the corresponding bucket of $T_k(\mathbf{b}_{n,k})$. Note that $\mathbf{b}_{n,k}$ is ignored (i.e., $\mathbf{b}_{n,k} = NULL$) if the $k^{th}$ visual word is not selected in the $n^{th}$ database image.

There exist $2^d$ buckets in each hash table, hence the number of buckets increases exponentially with the dimension $d$, resulting in heavy storage required for indices. Meanwhile, the search radius increases as well and leads to a huge number of buckets to be checked, even slower than linear scan. There is a need to reduce the dimensionality of components when $d$ is large. In this work, we employ a bit selection approach based on conditional mutual information maximization [18]. For each component, the goal is to select $d^{'}$ ($d^{'} < d$) elements that carry as much information as possible, which provides best separability between hamming distances for matching and non-matching image pairs of binary aggregated descriptors. We denote the selected bits from binary sub-vector $\mathbf{b}_{n,k}$ as $\mathbf{b}_{n,k}^{'}$.

**Online searching**. Given a query image $X_q$, the online search obejective is to generate a shortlist of nearest neighbor images. First, when scanning component hash tables $T$, we adopt a voting scheme to vote the database images based on the conflict with the components of $\mathbf{b}_q^{'}$, yielding a subset of candidates $\{\mathbf{b}_n^{'}\}_{n=1}^{N^{'}}, N^{'} << N$. An exhaustive search based on Hamming distance is subsequently performed within $\{\mathbf{b}_n^{'}\}_{n=1}^{N^{'}}$ to rerank the candidate images, yielding the shortlist.

The voting scheme aims to recall most of the reference images from the database based on conflict. For the $k^{th}$ query sub-vector $\mathbf{b}_{q,k}^{'} \neq NULL$, we enumerate all the binary component-wise vectors having less than $v$-bit differences ($v < d^{'}$) with $\mathbf{b}_{q,k}^{'}$, say $\{h_v\}$, and count the number of conflicts between $\mathbf{b}_{q,k}^{'}$ and each database image through the buckets $T_k(h_v)$ of the corresponding hash table $T_k$. The voting score $s(q, n)$ for the $n^{th}$ database image is formulated as follows:

$$s(q,n) = \sum_{i=0}^{v} \#_{n,i} \tag{2}$$

where $\#_{n,i} \in [0, K]$ denotes the number of components having $i$-bit differences between query and the $n^{th}$ database image. If $s(q, n) > \tau$, we add $\mathbf{b}_n^{'}$ into the candidate set, where $\tau$ is a threshold to control the size of candidate set.

**Complexity analysis**. We analyze the time and space complexity for both the offline component-based hash tables construction and the online searching, as listed in Table 1. The offline hash table construction does not affect the retrieval efficiency. Its space complexity is moderate. For a database with 1 million images($N = 256$, $d^{'} = 16$), the memory cost of hash tables is about $1.5G$ bytes. The time complexity for conflict image voting, $O(K * \sum_{i=0}^{v} C_{d^{'}}^{i} * \frac{N}{2^{d^{'}}})$, can be ignored when $v$ is small, while the search time mainly depends on the cost of exhaustive search in the candidate subset $\{\mathbf{b}_n^{'}\}_{n=1}^{N^{'}}$. Section 4 will discuss the search time in detail.

**Table 1**: Time and space complexity analysis. $C_i^j$ denotes the combination number. $K, N, d^{'}, v$ denote the number of visual word components, the number of database images, the reduced dimensionality of component and the number of bit differences, respectively.

| | Time complexity | space complexity |
|---|---|---|
| Hash table construction | $O(KN)$ | $O(K * 2^{d^{'}} + KN)$ |
| Online searching | $O(K * \sum_{i=0}^{v} C_{d^{'}}^{i} * \frac{N}{2^{d^{'}}}) + O(N^{'})$ | 0 |

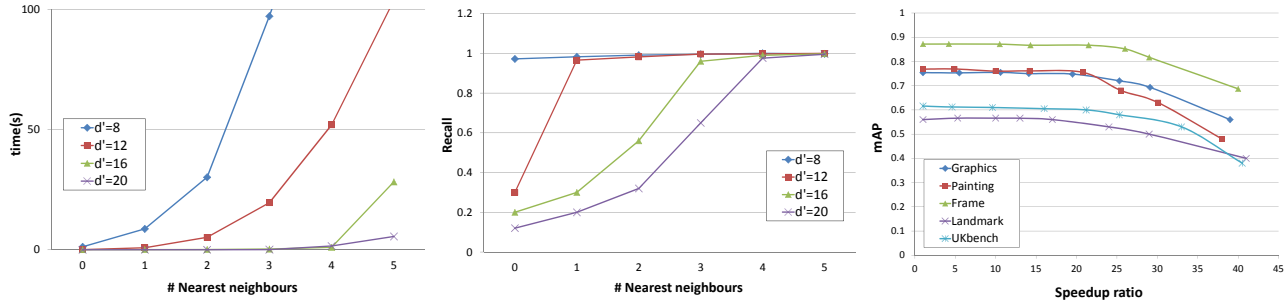
## 4. EXPERIMENTS

**Datasets and evaluation protocols**. We evaluate the retrieval performance of CoHash over the MPEG CDVS benchmark datasets [2]: (1) *Graphics* dataset depicts CD/DVD/book cover, text document and business card. There are 1,500 queries and 1,000 reference images; (2) *Painting* dataset contains 400 queries and 100 reference images of paintings. (3) *Frame* dataset contains 400 queries and 100 reference images of video frames. (4) *Landmark* dataset contains 3,499 queries and 9,599 reference images from building benchmarks. (5) *UKbench* dataset contains 2,550 objects, each containing 4 images taken from different viewpoints. A FLICKR1M dataset containing 1 million images is use as distracters, merging with the reference datasets to evaluate the performance over a large-scale database.

The retrieval performance is measured by mean Average Precision (mAP) and Recall@$N^{'}$, where $N^{'}$ denotes the number of database images in the candidate set.

**Implementation details**. In this work, SIFT descriptor is adopted as the local feature. The dimensionality of raw SIFT is reduced from 128 to $d = 32$ using Principal Component Analysis (PCA), like the state-of-the-arts works [8][10]. We employ the RCFC [20][13] as the variable-length binary aggregated descriptors with the number of Gaussians $K = 256$. For query image, the size $L$ of RCFC is varied with respect to the rate constraint, e.g., ranging from 292 bytes to 632 bytes. However, we fix the length of RCFC for database images at $L = 632$ bytes.

**Impact of parameters**. We first study the impact of varying number of selected bits $d^{'}$ and $v$-neighbors on the average search time (s) and retrieval accuracy in terms of Recall@$N^{'}$ over all queries with $L = 292$ bytes, as shown in Fig. 3 (a) and (b) respectively. With $v$ fixed, we found that the search time and Recall@$N^{'}$ increase when $d^{'}$ reduces from 20 to 8, the reason is that the number of buckets of each hash table is dramatically reduced, leading to a larger set of candidate images. A similar trend is observed as $v$ increases when $d^{'}$ is fixed. It has been empirically shown that the optimal $v$-neighbors depends on the length of hash codes $d^{'}$. In the next experiments, we use $v = 3, d^{'} = 16$ to obtain a tradeoff




(a) (b) (c)

**Fig. 3**: Impact of parameters. (a) Search time vs. different selected bits $d^{'}$ and $v$-neighbors (# Nearest neighbors). (b) Recall@$N^{'}$ vs. different selected bits $d^{'}$ and $v$-neighbors. (c) mAP vs. speedup ratio with different thresholds $\tau$.

**Table 2**: Comparison of CoHash and the-state-of-the-arts in terms of mAP and search time (s) over the various types of datasets with varying size $L$ (bytes) of RCFC.

| Dataset | Method | mAP | | | | Search Time (s) | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | $L=292$ | $L=352$ | $L=372$ | $L=632$ | $L=292$ | $L=352$ | $L=372$ | $L=632$ |
| Graphics | CoHash | 0.748 | 0.786 | 0.792 | 0.865 | 0.057 | 0.067 | 0.079 | 0.140 |
| | MIH | 0.742 | 0.788 | 0.832 | 0.877 | 0.295 | 0.412 | 0.495 | 0.911 |
| | LSH | 0.751 | 0.782 | 0.787 | 0.875 | 0.329 | 0.452 | 0.461 | 0.875 |
| | Linear Search | 0.754 | 0.795 | 0.800 | 0.883 | 1.120 | 1.341 | 1.435 | 2.874 |
| Painting | CoHash | 0.755 | 0.778 | 0.786 | 0.862 | 0.064 | 0.069 | 0.075 | 0.144 |
| | MIH | 0.758 | 0.787 | 0.796 | 0.865 | 0.432 | 0.488 | 0.469 | 0.855 |
| | LSH | 0.762 | 0.791 | 0.797 | 0.862 | 0.451 | 0.443 | 0.511 | 0.885 |
| | Linear Search | 0.769 | 0.791 | 0.802 | 0.874 | 1.345 | 1.440 | 1.467 | 2.762 |
| Frame | CoHash | 0.867 | 0.883 | 0.890 | 0.921 | 0.060 | 0.065 | 0.072 | 0.142 |
| | MIH | 0.872 | 0.891 | 0.891 | 0.925 | 0.321 | 0.512 | 0.567 | 0.928 |
| | LSH | 0.863 | 0.885 | 0.897 | 0.923 | 0.331 | 0.472 | 0.519 | 0.938 |
| | Linear Search | 0.872 | 0.894 | 0.894 | 0.931 | 1.294 | 1.327 | 1.411 | 2.810 |
| Landmark | CoHash | 0.560 | 0.582 | 0.584 | 0.624 | 0.074 | 0.081 | 0.086 | 0.183 |
| | MIH | 0.565 | 0.587 | 0.591 | 0.635 | 0.412 | 0.451 | 0.497 | 1.211 |
| | LSH | 0.561 | 0.592 | 0.593 | 0.631 | 0.386 | 0.442 | 0.483 | 0.961 |
| | Linear Search | 0.560 | 0.593 | 0.594 | 0.645 | 1.185 | 1.292 | 1.430 | 2.901 |
| UKbench | CoHash | 0.600 | 0.642 | 0.656 | 0.714 | 0.059 | 0.061 | 0.065 | 0.152 |
| | MIH | 0.611 | 0.653 | 0.655 | 0.717 | 0.341 | 0.356 | 0.517 | 0.986 |
| | LSH | 0.615 | 0.655 | 0.662 | 0.721 | 0.325 | 0.348 | 0.498 | 0.991 |
| | Linear Search | 0.616 | 0.660 | 0.665 | 0.721 | 1.255 | 1.346 | 1.461 | 2.843 |

between the retrieval accuracy and search time.

Fig. 3 (c) presents the impact of varying threshold $\tau \in \{0, 1, 3, 7, 15, 31, 63, 127\}$ on the search performance in terms of mAP and speedup ratio over various datasets. The number of candidate images reduces as $\tau$ increases, yielding remarkable search speedup. The results show that the search is $\sim$25 ($\tau = 15$) times faster than linear scan with comparable retrieval accuracy. However, when the speedup ratio exceeds 30, the mAP starts to drop significantly for all datasets.

**Variable-length binary descriptors**. Table 2 shows the search accuracy (mAP) and search time (s) of CoHash and the-state-of-arts algorithms with varying size of the binary aggregated descriptors RCFC. Under the same search accuracy, the proposed CoHash algorithm is significantly faster than others over all datasets at all query code sizes. Compared with the linear search, CoHash is 20$\sim$25 times faster while without any noticeable retrieval performance loss. MIH[15] and LSH[21] achieved about 5 times speed up while were still slower than Cohash. Note that for MIH and LSH, since they can't directly handle the varaible-length binary code, all the elements of sub-vector corresponding to the rejected visual words are assigned to 0.

## 5. CONCLUSION

In this paper, we proposed a novel component hashing algorithm to address the problem of fast image search with variable-length binary aggregated descriptors. Our approach significantly outperforms other the-state-of-the-arts methods and achieves about 20$\sim$25 times speedup than linear search. The results have demonstrated the efficiency and effectiveness of CoHash.

## 6. REFERENCES


[1] ISO/IEC JTC1/SC29/WG11/N12201. CFP for Compact Descriptors for Visual Search. 2011.

[2] ISO/IEC JTC1/SC29/WG11/N12202. Evaluation Framework for Compact Descriptors for Visual Search. 2011.

[3] E. Rublee, V. Rabaud and *et. al*. ORB: an efficient alternative to SIFT or SURF. *ICCV*, 2011.

[4] S. Leutenegger, C. Margarita, and *et. al*. BRISK: Binary robust invariant scalable keypoints. *ICCV*, 2011.

[5] C. Strecha, A. Bronstein, M. Bronstein, P. Fua. LDAHash: Improved matching with smaller descriptors. *PAMI*, 2012.

[6] M. Calonder, V. Lepetit, M. Ozuysal and *et al.*. BRIEF: Computing a Local Binary Descriptor Very Fast. *PAMI*, 2012.

[7] B. Girod, V. Chandrasekhar, D. Chen and *et al.*. Mobile Visual Search. *IEEE Signal Processing Magazine*, 2011.

[8] F. Perronnin, Y. Liu, J. Sanchez and *et al.*. Large-Scale Image Retrieval with Compressed Fisher Vectors. *CVPR*, 2010.

[9] D. Chen, S. Tsai and *et al.* Residual enhanced visual vector as a compact signature for mobile visual search. *Signal Processing*, 2012.

[10] H. Jegou, F. Perronnin, M. Douze and *et. al*. Aggregating local images descriptors into compact codes. *PAMI*, 2012.

[11] D. G. Lowe. Distinctive image features from scale invariant keypoints. *IJCV*, 2004.

[12] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. *ECCV*. 2006.

[13] ISO/IEC JTC1/SC29/WG11/M26726. Peking Univ. Response to Core Experiments 1: A Scalable Low-Memory Global Descriptor. 2012.

[14] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Comm.ACM*, 2008.

[15] M. Norouzi, A. Punjani, and D. Fleet. Fast search in hamming space with multi-index hashing. *CVPR*, 2012.

[16] Y. Weiss, A. Torralba, R. Fergus. Spectral hashing. *NIPS*, 2008.

[17] R. Salakhutdinov, G. Hinton. Semantic hashing. *IJAR*, 2009.

[18] F. Fleuret. Fast Binary Feature Selection with Conditional Mutual Information. *J. Mach. Learn. Res.*, 2004.

[19] ISO/IEC JTC1/SC29/WG11/M30311. Improvements to TM6 with a Robust Visual Descriptor: Proposal from University of Surrey and Visual Atoms. 2013.

[20] J. Lin, L.-Y. Duan, Y. Huang and *et al.*. Rate-adaptive Compact Fisher Codes for Mobile Visual Search. *IEEE Signal Processing Letters*, 2014.

[21] Gionis, Aristides and Indyk, Piotr and Motwani, Rajeev and others. Similarity search in high dimensions via hashing. *VLDB*, 1999.

[22] Trzcinski, Tomasz and Lepetit, Vincent and Fua, Pascal. Thick boundaries in binary space and their influence on nearest-neighbor search. *Pattern Recognition Letters*, 2013.