Int J Multimed Info Retr (2012) 1:59–70
DOI 10.1007/s13735-012-0003-7

INVITED PAPER

Multimedia semantics-aware query-adaptive hashing with bits reconfigurability

Yadong Mu · Xiangyu Chen · Xianglong Liu · Tat-Seng Chua · Shuicheng Yan

Received: 20 December 2011 / Accepted: 8 January 2012 / Published online: 9 March 2012

Abstract In the past decade, *locality-sensitive hashing* (LSH) has gained a large amount of attention from both the multimedia and computer vision communities owing to its empirical success and theoretic guarantee in large-scale multimedia indexing and retrieval. Original LSH algorithms are designated for generic metrics such as Cosine similarity, ℓ_2-norm and Jaccard index, which are later extended to support those metrics learned from user-supplied supervision information. One of the common drawbacks of existing algorithms lies in their incapability to be flexibly adapted to the metric changes, along with the inefficacy when handling diverse semantics (e.g., the large number of semantic object categories in the *ImageNet* database), which motivates our proposed framework toward reconfigurable hashing. The basic idea of the proposed indexing framework is to maintain a large pool of over-complete hashing functions, which are randomly generated and shared when indexing diverse multimedia semantics. For specific semantic category, the algorithm adaptively selects the most relevant hashing bits by maximizing the consistency between semantic distance and hashing-based Hamming distance, thereby achieving reusability of the pre-computed hashing bits. Such a scheme especially benefits the indexing and retrieval of large-scale databases, since it facilitates one-off indexing rather than continuous computation-intensive maintenance toward metric adaptation. In practice, we propose a sequential bit-selection algorithm based on local consistency and global regularization. Extensive studies are conducted on large-scale image benchmarks to comparatively investigate the performance of different strategies for reconfigurable hashing. Despite the vast literature on hashing, to our best knowledge rare endeavors have been spent toward the reusability of hashing structures in large-scale data sets.

Keywords Locality-sensitive hashing · Query-adaptive hashing · Bits reconfigurability · Shannon entropy

This research is supported by the CSIDM Project No. CSIDM-200803 partially funded by a grant from the National Research Foundation (NRF), which is administered by the Media Development Authority (MDA) of Singapore, and also supported by National Major Project of China "Advanced Unstructured Data Repository" (No. 2010ZX01042-002-001-00).

Y. Mu (✉)
Electrical Engineering Department, Columbia University, New York, NY 10027, USA
e-mail: muyadong@gmail.com; ym2372@columbia.edu

X. Chen · T.-S. Chua
School of Computing, National University of Singapore, Singapore 117576, Singapore
e-mail: chenxiangyu@nus.edu.sg

T.-S. Chua
e-mail: chuats@nus.edu.sg

X. Liu
State Key Laboratory of Software Development Environment, Beihang University, 100191 Beijing, People's Republic of China
e-mail: xlliu@nlsde.buaa.edu.cn

S. Yan
Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117576, Singapore
e-mail: eleyans@nus.edu.sg

1 Introduction

With the explosive accumulation of multimedia data in these domains such as shared photos or video clips on the Web, various multimedia applications suffer from large data scales and feature dimensions. Usually such databases are represented by uniform-length high-dimensional feature vectors. Defined on the video features, a simple yet essential

operation is to efficiently find a set of nearest neighbors for an arbitrary query by comparing pairwise feature proximity. A naive linear-scan implementation involves pairwise computations between the query and all items in the database, which has linear complexity with respect to the data set scale and is time-consuming for large-scale data and high dimensionality. Fortunately, in most applications, there is no need to identify the exact nearest neighbors. Instead, *approximate nearest neighbors* (ANN) [3,2] achieve comparable performance in many scenarios, while greatly decreasing the computational cost. It motivates the research on efficient indexing for large-scale image and video data sets.

Recent progress has witnessed the popularity of *locality-sensitive hashing* (LSH) [2] as an invaluable tool for retrieving approximate nearest neighbors in the aforementioned setting. The basic idea of LSH is to randomly generate a number of "buckets" according to specific hashing scheme and map data into the hashing buckets. Unlike other kinds of hashing algorithms, LSH is characterized by the so-called "locality-sensitive" property. Namely, denote collision probability to be the probability that two data points are mapped into the same bucket. A valid LSH algorithm will guarantee higher collision probability for similar data. The line of work has gained considerable empirical success in a variety of tasks such as image search, near-duplicate image detection [13], human pose estimation [26], etc.

The key factor for an LSH algorithm is the underlying metric to measure data similarity. Original LSH algorithms are devised for uniform-length feature vectors equipped with "standard" metrics, including Jaccard Index [4], Hamming distance [11], ℓ_2-norm [1], Cosine similarity [5] or general ℓ_p-norm ($p \in (0, 2]$) [6]. Although strict collision-bound analysis is presented, unfortunately it is seldom the case that in real-world multimedia applications the pairwise similarity between visual identities (e.g., images, three-dimensional shapes, video clips) are gauged using aforementioned metrics. It is essentially caused by the well-known semantic gap between low-level features and high-level multimedia semantics. Instead, the so-called Mercer kernels [25] provide more flexibility by implicitly embedding original features into high-dimensional Hilbert spaces. Representative Mercer kernels widely used by multimedia practitioners include the Radial Basis Function (RBF) kernel [25] and Pyramid Match Kernel (PMK) [8]. Previous study [14,18,9] shows that the extension of LSH algorithms to the kernelized case is feasible.

Note that all of the aforementioned metrics (including those induced from Mercer kernels) are explicitly predefined. More complications stem from the ambiguous metrics implicitly defined by a bunch of pairwise similarity (or dissimilarity) constraints, which frequently occur in the research field of metric learning [32]. Hashing with this kind of partial supervision is challenging. Previous efforts address this task toward two directions: (1) hashing with learned metric [14], which transfigures the original metrics (typically via the modulation of Mahalonobius matrix) and then applies standard hashing techniques, and (2) data-dependent hashing with weak supervision [28,18], which seeks most consistent hashing hyperplanes by constrained optimization. The methods from the first category are computationally efficient, since it decouples the overall complicated problem into two sub-problems, each of which is relatively easier. However, when the similarity (or dissimilarity) constraints are given in a very sparse manner, the input will be insufficient to learn a high-quality metric; therefore, they are probably not applicable. The methods from the second category are more tightly related to the final performance, since they simultaneously optimize the hashing hyperplanes and discriminative functions. Their advantages lie in the high complexity in non-convex optimization [18] or eigen-decomposition [28]. However, despite their success, existing techniques fail to handle the diverse semantics in real-world multimedia applications. The cruxes of the dilemma originate from two factors:

- The ambiguity and inconstancy of the multimedia semantics. An example is the visual semantics induced from pairwise affinity relationship, which is either constructed from manual specification or community-contributed noisy tags. Unfortunately, both information sources are usually subject to frequent update, which potentially causes semantic drifting. Since both the hashing scheme and the resultant indexing structure are seriously hinged on the underlying semantics or metric, one-off data indexing is unfeasible under such circumstance of unstable semantics, which triggers unnecessary labors spent on indexing structure maintenance.
- The diversity of the semantics [30]. Most of previous studies assume that data are associated with a small number of distinct semantics, which is usually not the true case in real-world benchmarks. For example, the hand-labeled ImageNet data set[1] contains more than ten million images that depict 10,000+ object categories. Facing such input, one possible solution is to simultaneously pursue the optimal hashing functions for all categories. However, it is unwise considering the unknown and complex intrinsic data structures. Another possible solution is to separately conduct hashing for each unique category and concatenate all to form the final indexing structure, which unfortunately is uneconomic in terms of storage (actually the overlapped semantic subspace between two categories implies that several hashing bits can be shared to save the storage) and vulnerable to semantic changes and new emerging categories due to the expensive re-indexing effort for the large-scale data set.

[1] http://www.image-net.org/challenges/LSVRC/2010.

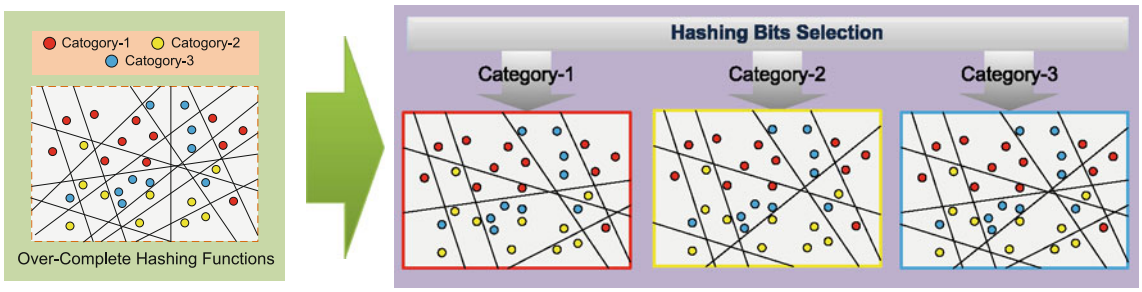
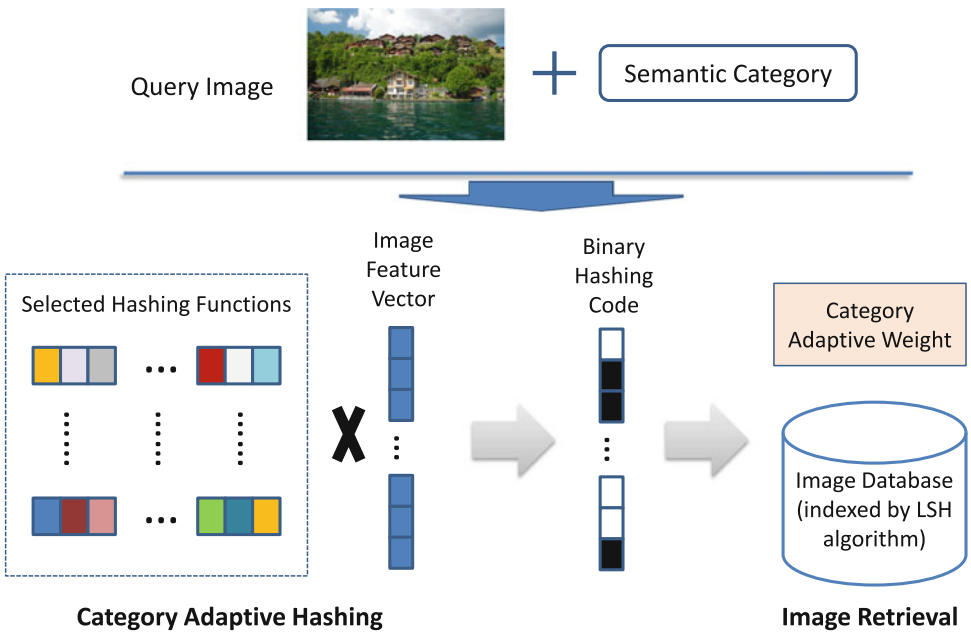


Fig. 1 Illustration of the proposed hashing scheme with bit reconfigurability. The *left subfigure* shows a toy data set and the pre-computed redundant hashing functions, while the contents on the *right panel sketch* the idea of optimal hashing bit selection toward specific semantic category

Fig. 2 Illustration of the proposed image retrieval system



The above-mentioned drawbacks of existing methods motivate "reconfigurable hashing" proposed in this paper. Figure 1 illustrates the basic idea of reconfigurable hashing, whose basic operation is to generate a set of over-complete hash functions and perform one-off data indexing. When the semantic annotations or constraints are available, the algorithm optimally chooses a small portion of relevant hashing bits from the pool and re-weight them to best fit the target semantic metrics. Obviously, the so-called reconfigurable hashing is in hash bit level.

Figure 2 presents the processing pipeline of the image retrieval system. The images in the database are indexed according to a large number of hashing functions. In the retrieval stage, a new image is introduced as the query. We assume that the semantic category associated with the query image is also known. Based on the semantic category, the algorithms discussed in this paper are capable of selecting category-adaptive hashing functions, which is a small subset of the overall hashing pool. Low-level features are extracted from the query image and indexed to obtain the binary hashing code, which is afterward compared with those stored in the image database to find the nearest images.

In this paper, the goal is to develop a novel indexing framework that supports an unlimited number of diverse semantics based on one-off indexing, and also admits the adaptation to the metric changes at very low computational cost and zero re-indexing effort. In detail, our contributions in this paper can be summarized as follows:

- A novel hashing algorithm named *random-anchor-random-projection* (RARP), which is equivalent to redundant random partition of the ambient feature space and proves superior to other candidate LSH algorithms. Strict collision analysis for RARP is supplied.
- We discuss different strategies for optimal hash function selection and further propose a sequential algorithm based on local consistence and global regularization.
- The idea of reconfigurable hashing is content agnostic and consequently domain independent, but the performances of different selection strategies vary. Compar-

ative investigation of the proposed and other candidate strategies is provided on four popular multiple-semantics image benchmarks, which validates the effectiveness of reconfigurable hashing and its scalability to large-scale data set.

The rest of the paper is organized as follows. Section 2 provides a brief survey of relevant literature. Section 3 defines the notations used in this paper and formally states the problem to be solved. Sections 4 and 5 elaborate on the proposed formulation and also other alternative strategies. More details of the hashing collision analysis are found in Sect. 6. Extensive experiments are conducted on four real-world benchmarks in Sect. 7 and in Sect. 8 we give the concluding remarks and point out several directions for future work.

2 Related work

In this section, we provide a brief review of various *locality-sensitive hashing* (LSH) [2,6,11,17] methods that are
recently proposed to tackle the large-scale retrieval problem.

Let $\mathcal{H}$ be a family of hashing functions mapping $\mathbb{R}^d$ to some universe U. The family $\mathcal{H}$ is called *locality sensitive* if it satisfies the following conditions:

Definition 1 (*Locality-sensitive hashing* [2]) A hashing family $\mathcal{H}$ is called $(1, c, p_1, p_2)$-sensitive if the following properties hold for any two samples $x, y \in \mathbb{R}^d$, i.e.,

$(K1)$ If $\|x - y\| \leq 1$ then $\Pr_{\mathcal{H}}(h(x) = h(y)) \geq p_1$.
$(K1)$ If $\|x - y\| \geq c$ then $\Pr_{\mathcal{H}}(h(x) = h(y)) \leq p_2$.

To guarantee that the hashing functions from $\mathcal{H}$ are meaningful, typically we have $c > 1$ and $p_1 > p_2$. Other alternative definition exists, such as $\forall h \in \mathcal{H}, \Pr[h(x) = h(y)] = \kappa(x, y)$, where $\kappa(\cdot, \cdot)$ denotes the similarity measure between samples x and y. In other words, x and y's collision probability (i.e., being mapped to the same hash bucket) monotonically increases with respect to their similarity value, which is known as the *locality-sensitive* property.

Existing LSH algorithms can be roughly cast into the following categories:

– *Element sampling or permutation* Well-known examples include the hashing algorithms developed for the Hamming distance [11] and Jaccard Index [4]. For example, in the Hamming case, feature vectors are all binary valued. The work in [11] presents a hashing scheme $h(x) = x_i$, where i is randomly sampled from the dimension index set $\{1, \ldots, d\}$ and x_i is the binary value of the i-th dimension. The guarantee of the locality-sensitive property is also given in [11].

– *Project-shift-segment* The idea is to map a feature point in $\mathbb{R}^d$ onto $\mathbb{R}^1$ along a random projection direction in $\mathbb{R}^d$, and then randomly shift the projection values. Finally, the range of projection values is partitioned into several intervals of length l_w (l_w is a data-dependent parameter and needs fine tuning). In the extreme case, there are only two partitions and the output is binary bit. Examples include the algorithm for ℓ_1 norm [1], for Cosine similarity [5,7], for ℓ_p norm [6] and for kernel-based metrics or semi-metrics [14,20].

Here are two representative examples:

(1) Arccos distance: for real-valued feature vectors lying on hypersphere $S^{d-1} = \{x \in \mathbb{R}^d \mid \|x\|_2 = 1\}$, an angle-oriented distance can be defined as $\Theta(x, y) = \arccos\left(\frac{\langle x, y\rangle}{\|x\|\|y\|}\right) = \arccos(\langle x, y\rangle)$. Charikar et al. [5] propose the following LSH family:

$$h(x) = \begin{cases} 0, & \text{if } \omega^\top x < 0 \\ 1, & \text{if } \omega^\top x \geq 0 \end{cases} \tag{1}$$

where the hashing vector ω is uniformly sampled from the unit hypersphere S^{d-1}. The collision probability is $\Pr[h(x) = h(y)] = 1 - \Theta(x, y)/\pi$.

(2) ℓ_p distance with $p \in (0, 2]$: for linear vector spaces equipped with the ℓ_p metric, i.e., $D_{\ell_p}(x, y) = (\sum_{i=1}^{d} |x_i - y_i|^p)^{\frac{1}{p}}$, Datar et al. [6] propose a hashing algorithm based on linear projections onto a one-dimensional line and chopping the line into equal-length segments, as below:

$$h(x) = \left\lfloor \frac{\omega^\top x + b}{W} \right\rfloor, \tag{2}$$

where the hashing vector $\omega \in \mathbb{R}^d$ is randomly sampled from the *p-stable distribution* and $\lfloor \cdot \rfloor$ is the flooring function for rounding. W is the data-dependent window size and b is sampled from the uniform distribution $U[0, W)$.

– *Prototype-based methods* Another LSH family uses predefined prototypes, such as polytopes on 24-D Leech lattice in ℓ_2 space [1] (i.e., E2LSH) or 8-D lattice [23].
– *Learning-based methods* Assisted with semantic annotations or labels, LSH can be adapted via various learning methods like the classic SpectralHash [31] and SemanticHash [24]. Recent progress has also been made on hashing with weak supervision [18,28] and sequential optimization [29].

From the brief survey in this section, it is observed that prior research has mainly focused on designing LSH algorithms for specific metrics, while the task of our work aims to provide a meta-hashing method applicable to the existence of scalable diverse semantics and adaptive metrics. To our best knowledge, few related work can be found. Study on this

topic still lacks in-depth exploration and remains an open problem.

3 Notations and problem setting

Before continuing, let us formally establish the notations and the problem setting. Denote $\mathcal{X} = \{x_1, \ldots, x_n\}$ as the set of feature vectors in $\mathbb{R}^d$. Let $h_i : \mathbb{R}^d \mapsto \{0, 1\}, i = 1 \ldots m$ be m independently generated hashing functions, where m is large enough to form an over-complete hashing pool. All samples in $\mathcal{X}$ are hashed to obtain binary bits according to the collection of hashing functions $\{h_i\}$. The hashing operation is performed only once and not required to be redone any more. The aim of reconfigurable hashing is to select compact hashing bit configuration from the pool to approximate any unknown metrics in terms of Hamming distance.

It is reasonable to assume that the maximum number of active hashing functions for each semantic category is budgeted. Denote it as l and assume $l \ll m$. To explicitly define the target semantics (or equivalently, metrics), assume that a fraction of data in $\mathcal{X}$ are associated with side information. Specifically, we focus on the widely used pairwise relationship [18,28] throughout this paper, which reveals the proximal extent of the two samples.

Define two sets $\mathcal{M}$ and $\mathcal{C}$. For any sample pair $(x_i, x_j) \in \mathcal{M}$, it reflects the acknowledgement from the annotators that x_i, x_j semantically form a neighbor pair in the context of target category. Similarly, $(x_i, x_j) \in \mathcal{C}$ implies that they are far away in the unknown metric space or have different class labels. Note that the manual annotation is typically labor intensive; therefore, normally we assume that the labeled samples only cover a small portion of the whole data set. Also for large-scale data set associated with diverse semantics, the annotation is heavily unbalanced. In other words, the cardinality of $\mathcal{M}$ is far less than that of $\mathcal{C}$, which mainly follows from the fact that $\mathcal{C}$ is the amalgamation of all other non-target categories. A qualified algorithm on reconfigurable hashing is expected to survive in such settings.

Generally, we can regard the hashing function h_i as a black box and only visit the binary hashing bits during the optimization. Different hashing schemes notably affect the retrieval quality given budgeted hashing bits. Ideally, most hashing functions are expected to be relevant to a target semantic category and complementary to each other. In this paper, we target the data lying in the ℓ_p-normed spaces $(0 < p \leq 2)$ since it covers most of the feature representations used in multimedia applications. Most of the traditional hashing approaches [(e.g., the one presented in Eq. (1)] often ignore the data distribution, which potentially results in lower efficiency for unevenly distributed data. For example, the well-known SIFT feature [15] resides only within one of the quadrants. When applying the hashing algorithm in (1), more empty hashing buckets will be found. To attack this issue, we propose a hashing scheme named *random-anchor-random-projection* (called RARP hereafter), which belongs to the random projection-based hash family, yet differentiates itself from others by taking data distribution into account.

In the proposed method, to generate a hashing function, a sample x^o is randomly sampled from the data set to serve as the so-called "anchor point". Also, a random vector ω is sampled uniformly from the p-stable distribution [6,12]. The projection value can be evaluated as $\langle \omega, x - x^o \rangle = \langle \omega, x \rangle - b_{\omega,x^o}$, where $b_{\omega,x^o} = \langle \omega, x^o \rangle$ is used as the hashing threshold, i.e.,

$$h(x) = \begin{cases} 0, & \text{if } \langle \omega, x \rangle < b_{\omega,x^o} \\ 1, & \text{if } \langle \omega, x \rangle \geq b_{\omega,x^o} \end{cases} \tag{3}$$

where $\langle \omega, x \rangle$ denotes the inner product between ω and x. The collision analysis for RARP is discussed in Sect. 6.

In the hashing literature, it is common to utilize Hamming distance as a proxy of the distance or similarity in the original feature space, which is defined as:

$$H(x, x') = \sum_{b=1}^{B} (h_b(x) \oplus h_b(x')), \tag{4}$$

where $\oplus$ denotes the logical XOR operation (the output of XOR will be one if two input binary bits are different, and otherwise zero). Recall that the range of each hashing function is {0, 1}. Equation (4) can be expressed in a more tractable form:

$$\|h(x) - h(x')\| = (h(x) - h(x'))^{\mathrm{T}} (h(x) - h(x')). \tag{5}$$

Here, we adopt a generalized Hamming distance to ease numerical optimization. Specifically, we introduce the parametric Mahalonoisbu matrix M for modulating purpose. To ensure the positiveness of the resulting measure, M is required to reside in the positive semi-definite (p.s.d.) cone, or mathematically $M \succeq 0$. The distance under specific M can be written as follows:

$$\|h(x) - h(x')\|_M = \left(h(x) - h(x')\right)^{\mathrm{T}} \cdot M \cdot \left(h(x) - h(x')\right). \tag{6}$$

4 The proposed algorithm

As a meta-hashing framework, the ultimate goal of reconfigurable hashing is the selection of hashing bits from a pre-built large pool. In this section, we first present a novel algorithm based on the idea of *averaged margin* and *global regularization*. Moreover, we also describe the other algorithm that simultaneously optimizes the hashing functions and bit weights.

Later, we also present four more baseline algorithms for the same task, based on random selection, maximum variance, maximum local margin and Shannon information entropy, respectively. The empirical evaluation of the above methods is postponed to the experimental section.

4.1 Formulation

As stated above, we rely on sets $\mathcal{M}$ and $\mathcal{C}$ to determine the underlying semantics. However, the construction of pairwise relationship has quadratic complexity of the sample number. To mitigate the annotation burden, a practical solution is instead to build two sets $\mathcal{L}_+$ and $\mathcal{L}_-$. The former set consists of the samples assigned to the target semantic label, and $\mathcal{L}_-$ collects the rest samples. We further generate *random homogeneous pair* and *random heterogeneous pair* to enrich $\mathcal{M}$ and $\mathcal{C}$, respectively. For each sample $x_i \in \mathcal{L}_+$, we randomly select $x_j \in \mathcal{L}_+$ with the guarantee $i \neq j$. The pair (x_i, x_j) is called *random homogeneous pair*. Likewise, given $x_k \in \mathcal{L}_-$, (x_i, x_k) constitutes a *random heterogeneous pair*. Therefore, the construction of $\mathcal{M}$ and $\mathcal{C}$ is efficient.

Matrix M in Eq. (6) can be eigen-decomposed to obtain $M = \sum_{k=1}^{K} \sigma_k w_k w_k^{\mathrm{T}}$. To simplify numerical optimization, we impose $\sigma_k = 1$ such that $M = WW^{\mathrm{T}}$ where $W = [w_1, \ldots, w_K]$. Denote the index set $\mathcal{I}$ to be the collection of selected hashing bits at the current iteration. Let $h_{\mathcal{I}}(x_i)$ be the vectorized hashing bits for x_i. Two margin-oriented data matrices can be calculated by traversing $\mathcal{M}, \mathcal{C}$, respectively, and piling the difference column vectors, i.e.,

$$X_m = \{h_{\mathcal{I}}(x_i) - h_{\mathcal{I}}(x_j)\}_{(x_i,x_j)\in\mathcal{M}}$$
$$X_c = \{h_{\mathcal{I}}(x_i) - h_{\mathcal{I}}(x_j)\}_{(x_i,x_j)\in\mathcal{C}}$$

We adopt the *averaged local margin* [27] based criterion to measure the empirical gain of $\mathcal{I}$, which is defined as:

$$J(W) = \frac{1}{n_c}\mathrm{tr}\{W^{\mathrm{T}} X_c X_c^{\mathrm{T}} W\} - \frac{1}{n_m}\mathrm{tr}\{W^{\mathrm{T}} X_m X_m^{\mathrm{T}} W\}, \tag{7}$$

where n_c and n_m are cardinalities of $\mathcal{C}$, $\mathcal{M}$, respectively. Intuitively, $J(W)$ maximizes the difference between *random heterogeneous pair* and *random homogeneous pair* in terms of averaged Hamming distances, analogous to the concept of margin in kernel-based learning [25].

Moreover, prior work such as the well-known *spectral hashing* [31] observes an interesting phenomena, i.e., hashing functions with balanced bit distribution tend to bring superior performance. In other words, the entire data set is split into two equal-size partitions. Intuitively, balanced hashing function separates more nearest neighbor pairs. Coupling the independence condition of different bits, such a scheme results in more buckets. Consequently, the collisions of heterogeneous pairs are reduced with high probability. Motivated by this observation, we introduce a global regularizer regarding bit distribution, i.e.,

$$\mathcal{R}(W) = \mathbb{E}(\|W^{\mathrm{T}}(h_{\mathcal{I}}(x_i) - \mu)\|_2^2), \tag{8}$$

where μ represents the statistical mean of all hashing-bit vectors. In practice, a small subset $\mathcal{X}_s$ with cardinality n_s is sampled and serves as a statistical surrogate. Equation (8) can be rewritten as:

$$\mathcal{R}(W) = \frac{1}{n_s}\mathrm{tr}(W^{\mathrm{T}} X_s X_s^{\mathrm{T}} W) - \mathrm{tr}(W^{\mathrm{T}}\mu\mu^{\mathrm{T}} W). \tag{9}$$

For brevity, denote $L_J = X_c X_c^{\mathrm{T}}/n_c - X_m X_m^{\mathrm{T}}/n_m$ and $L_R = X_s X_s^{\mathrm{T}}/n_s - \mu\mu^{\mathrm{T}}$. Considering all together, finally we get the regularized objective function:

$$F(W) = \mathrm{tr}(W^{\mathrm{T}} L_J W) + \eta \cdot \mathrm{tr}(W^{\mathrm{T}} L_R W), \tag{10}$$

where $\eta > 0$ is a free parameter to control the regularizing strength. It is easily verified that

$$\max F(W) = \sum_{k=1}^{K} \lambda_k, \tag{11}$$

where $\{\lambda_k\}$ comprise the non-negative eigenvalues of matrix $L_J + \eta \cdot L_R$ (the negative eigenvalues stem from the indefinite property of L_J) and the value of K is thereby automatically determined.

Due to the large number of the hashing pool, global optimization is computationally forbidden. Here, we employ a greedy strategy for sequential bit selection. In the t-th iteration, each unselected hashing function h_p is individually added into current index set $\mathcal{I}^{(t)}$ and the optimum of $F(W)$ under $\mathcal{I}^{(t)} \cup \{p\}$ is computed. The hashing function that maximizes the gain will be eventually added into $\mathcal{I}^{(t)}$. The procedure iterates until the hashing bit budget is reached.

Unfortunately, one potential selection bias is rooted in the term $tr\{W^{\mathrm{T}} X_c X_c^{\mathrm{T}} W^{\mathrm{T}}\}$ in Eq. (7), which can be equivalently expressed as $\sum_{(x_i,x_j)\in\mathcal{C}} W^{\mathrm{T}} h_{ij} h_{ij}^{\mathrm{T}} W^{\mathrm{T}}$ with $h_{ij} = h_{\mathcal{I}}(x_i) - h_{\mathcal{I}}(x_j)$. Owing to the summation operation over the constraint set $\mathcal{C}$, the estimation is smooth and robust. However, recall that $\mathcal{C}$ is randomly rendered. In some extreme case, the selected optimal hashing functions may be trapped in the regions where the density of (x_i, x_j) is relatively high, resulting the zero-norm values of some difference vectors (i.e., $\|h_{ij}\|_0$) are extremely high.

To mitigate this selection bias, we truncate too-high zero-norm to avoid over-penalizing. Given a predefined threshold θ (in implementation we set $\theta = 5$, which is a conservative parameter since hashing buckets with distances larger than five are rarely visited in approximate nearest neighbor retrieval), we re-scale the difference vector via the following formula:

$$h_{ij} = \frac{\min(\|h_{ij}\|_0, \theta)}{\|h_{ij}\|_0} \cdot h_{ij}. \tag{12}$$

4.2 Hashing function refinement

Note that the proposed algorithm in the previous section is intrinsically a meta-hashing algorithm, since it does not take the construction of hashing functions into account. Instead, it directly works with the binary codes produced by the random hashing functions. An interesting problem is that whether it helps or not if we further refine the selected hashing functions. As a tentative attempt, we propose another formulation that makes refinement to the hashing functions.

Suppose we have obtained the hashing vectors $F_0 \in \mathbb{R}^{d\times l}$ for the k-th category. To further refine the hashing vectors, a natural solution is to jointly optimize the bit re-weighting parameter and hashing vectors. For ease of optimization, here we abandon the transform matrix W in the previous section and introduce the vector $\alpha_k \in \mathbb{R}^{l\times 1}$ for bit re-weighting purpose. Denote the hashing vectors after refinement to be F. The key idea is akin to the idea of supervised locality-preserving method based on graph Laplacian [10]. Specifically, the (i, j)-th element $(i \neq j)$ in Laplacian matrix L_k is only non-zero when x_j belongs to the k-NN of x_i and x_i, x_j are from the same semantic category. The overall formulation is as follows:

$$\begin{aligned} \arg\min_{F,\alpha_k} \ & \alpha_k^{\mathrm{T}} F^{\mathrm{T}} X L_k X^{\mathrm{T}} F \alpha_k + \beta \|F - F_0\|_F^2, \\ \text{s.t.} \ & \forall\, i, \|f_i\|_2 \leq 1, \\ & \alpha_k \geq 0, \ \|\alpha_k\|_1 = 1, \end{aligned} \tag{13}$$

where $\|\cdot\|_F$ denotes the matrix Frobenius norm. β is a free parameter to control the proximity of the final solution to the initial value F_0.

The overall formulation is non-convex. However, it becomes convex when fixing one of the variables (either F or α_k). When F is known, it is a quadratic programming with linear constraints. When α_k is fixed, F can be updated by the projected gradient method. This alternating minimization algorithm guarantees a fixed point for (F, α_k).

5 Baseline algorithms

Besides our proposed hashing bit selection strategy, we also explore other alternatives. In detail, we choose the following:
Method-I: random selection (RS). In each iteration, select a hashing bit from the pool by uniform sampling. The procedure terminates when maximum budgeted number of hashing functions is reached.
Method-II: maximum unfolding (MU). As previously mentioned, previous research has revealed the superior performance of balanced (or max-variance) hashing function. In other words, it prefers hashing schemes with maximum unfolding. This strategy selects top-ranked maximum-variance hashing bits from the pool.

Method-III: maximum averaged margin (MAM). Similar to Eq. (7), we can compute the *averaged margin* of each hashing function in the pool according to the formula and keep top-scored hashing bits via greedy selection.

$$\begin{aligned} \text{score}(h_p) = & \ \mathbb{E}_{(x_i,x_j)\in\mathcal{C}}(h_p(x_i) \oplus h_p(x_j)) \\ & - \mathbb{E}_{(x_i,x_j)\in\mathcal{M}}(h_p(x_i) \oplus h_p(x_j)). \end{aligned} \tag{14}$$

Method-IV: weighted Shannon entropy (WSE). For each candidate in the pool, we calculate a score based on the Shannon entropy [16]. For completeness, we give its definition. Assume the index set of data as L, two disjoint subsets L_l and L_r can be created by a Boolean test $\mathcal{T}$ induced by a hashing function $h(\cdot)$. The Shannon entropy is computed as:

$$S_C(L, \mathcal{T}) = \frac{2 \cdot I_{C,\mathcal{T}}(L)}{H_C(L) + H_{\mathcal{T}}(L)}, \tag{15}$$

where H_C denotes the entropy of the category distribution in L. Formally,

$$H_C(L) = -\sum_c \frac{n_c}{n} \log_2 \frac{n_c}{n}, \tag{16}$$

where n is the cardinality of L and n_c is the number of samples in the category with index c. Maximal value is achieved when all n_c are the same. Similarly, the *split entropy* $H_{\mathcal{T}}$ is defined for the test $\mathcal{T}$, which splits the data into two partitions:

$$H_{\mathcal{T}}(L) = -\sum_{p=1}^{2} \frac{n_p}{n} \log_2 \frac{n_p}{n}, \tag{17}$$

where n_p ($p = 1$ or 2) denotes the sample number in L_l or L_r. The maximum of $H_{\mathcal{T}}(L)$ is reached when the two partitions have equal sizes. Based on the entropy of L, the *impurity* of $\mathcal{T}$ can be calculated by the mutual information of the split, i.e.,

$$I_{C,\mathcal{T}}(L) = H_C(L) - \sum_{p=1}^{2} \frac{n_p}{n} H_C(L_p). \tag{18}$$

Intuitively, $S_C(L, \mathcal{T})$ prefers $\mathcal{T}$ that is as balanced as possible and meanwhile separates different categories. As aforementioned, in the setting of reconfigurable hashing, the numbers of labeled samples from target category and non-target categories are heavily unbalanced, therefore we re-scale the sample weights such that the summed weights for the target category and non-target categories are equal. Finally, the hashing functions with the highest scores are kept.

6 Hashing collision probability

Before delving into the experimental results, we would like to highlight the asymptotic property of the proposed *random-anchor-random-projection* (RARP) hashing functions.

For two samples x_1 and x_2, let $c = \|x_1 - x_2\|_p$. In the hashing literature, it is well acknowledged [11] that the computational complexity of a hashing algorithm is dominated by $\mathcal{O}(n^\rho)$, where n is the data set size and $\rho < 1$ is dependent on algorithm choice and c. Suppose ω determines the parametric random hashing hyperplane. It is known that $\langle \omega, x_1 - x_2 \rangle$ is distributed as cX, where X is drawn from the p-stable distribution. Denote the range of projected values as $R = \max_i \langle \omega, x_i \rangle - \min_i \langle \omega, x_i \rangle$ and let $\eta = \frac{\langle \omega, x^o \rangle - \min_i \langle \omega, x_i \rangle}{R}$ (x^o is the random anchor). Without loss of generality, we assume $\eta > 0.5$. Let $g_p(t)$ be the probability density function of the absolute value of the p-stable distribution. The collision probability of RARP can be written as

$$\Pr(h_{\omega,x^o}(x_1) = h_{\omega,x^o}(x_2)) \approx \int_0^{\eta R} \frac{1}{c} g_p\left(\frac{t}{c}\right)\left(\eta - \frac{t}{R}\right) \mathrm{d}t + \int_0^{(1-\eta)R} \frac{1}{c} g_p\left(\frac{t}{c}\right)\left(1 - \eta - \frac{t}{R}\right) \mathrm{d}t \tag{19}$$

The two terms in Eq. (19) reflect the chances that x_1, x_2 collides in the two sides of x^o, respectively. Note that the equality relationship only approximately holds in (19) due to the uneven data distribution (computing the accurate probability involves double integrals along ω), and rigorously holds in case of uniform distribution. Moreover, when R is large enough and the uniformity holds, analytic bound for ρ exists. Analysis in this section follows closely [6], and therefore the detailed proofs are omitted.

Theorem 1 *For any $p \in (0, 2]$ and $c > 1$, there exists hashing family $\mathcal{H}$ for ℓ_p-norm such that for any scalar $\gamma > 0$,*

$$\lim_{R\to\infty} \rho \le (1 + \gamma) \cdot \max\left(\frac{1}{c}, \frac{1}{c^p}\right). \tag{20}$$

7 Experiments

In this section, we justify the effectiveness of the proposed reconfigurable hashing through empirical evaluations on four benchmarks: Caltech-101,[2] MNIST-Digit,[3] CIFAR-10 and CIFAR-100.[4] In the experiments, we compare the proposed hashing bit selecting strategy with other alternatives presented in Sect. 5. To reduce the effect of randomness, all experiments are iterated 30 times to get the statistical average. By default, we set $\eta = 0.5$ and choose both four samples from target category and non-target categories to construct $\mathcal{M}$ and $\mathcal{C}$. The size of the hashing pool is fixed to be 10K in all experiments unless otherwise mentioned. Figure 3 shows selected images in the adopted benchmarks.

Fig. 3 Exemplar images on selected benchmarks

7.1 Caltech-101 and CIFAR-100

Caltech-101 is constructed to test object recognition algorithms for semantic categories of images. The data set contains 101 object categories and one background category, with 40–800 images per category. As preprocessing, the maximum dimension of each image is normalized to be 480 pixels. We extract 5,000 SIFT descriptors from each image whose locations and scales are determined in a random manner (see [21] for more details). For the visual vocabulary construction, we employ the recently proposed *randomized locality-sensitive vocabularies* (RLSV) [19] to build 20 independent bag-of-words feature, each of which consists of roughly 1K visual words. Finally, they are concatenated to form a single feature vector and reduced to be 1000-dimensional by dimensionality reduction.

CIFAR-100 comprises 60,000 images selected from 80M Tiny-Image data set.[5] This data set is just like the CIFAR-10, except that it has 100 classes containing 600 images each. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. Each image comes with a "fine" label (the class

[2] http://www.vision.caltech.edu/Image_Datasets/Caltech101/.

[3] http://yann.lecun.com/exdb/mnist/.

[4] http://www.cs.utoronto.ca/~kriz/cifar.html.

[5] http://people.csail.mit.edu/torralba/tinyimages/.

Table 1 Exemplar labels in the CIFAR-100 data set

Coarse labels	Fine labels
Fish	Aquarium fish, flatfish, ray, shark, trout
Flowers	Orchids, poppies, roses, sunflowers, tulips
Food containers	Bottles, bowls, cans, cups, plates
Trees	Maple, oak, palm, pine, willow
People	Baby, boy, girl, man, woman
Vehicles 1	Bicycle, bus, motorcycle, pickup truck, train
Vehicles 2	Lawn mower, rocket, streetcar, tank, tractor
Reptiles	Crocodile, dinosaur, lizard, snake, turtle
Insects	Bee, beetle, butterfly, caterpillar, cockroach

to which it belongs) and a "coarse" label (the superclass to which it belongs). Table 1 presents some examples of these two-granularity categories. For the 32×32 pixel images, we extract ℓ_2-normalized 384-D GIST features [22].

We randomly generate 15 samples from each category in Caltech-101 and 30 samples for CIFAR-100 respectively, used for hashing bit selection. For each category, a unique hashing scheme is learned either by our proposed method or other methods mentioned in Sect. 5 with hashing bit budget equal to 14 on Caltech-100 or 16 on CIFAR-100, therefore in total 102 different hashing schemes for Caltech-101 and 100 for CIFAR-100. During the learning on specific category, the ensemble of the rest categories serves as the negative class. For each training sample from the target category, four random homogenous pairs and four random heterogenous pairs are generated by uniform sampling, forming the constraint sets $\mathcal{M}$ and $\mathcal{C}$, respectively.

We report the averaged results over 30 runs of our proposed method, along with the results obtained by four baselines, i.e., random selection (RS), maximum unfolding (MU), maximum averaged margin (MAM) and weighted Shannon entropy (WSE). The results of naive linear scan (NLS) are also reported. However, recall that NLS utilizes no side information. There is no guarantee that NLS provides the upper bound of the performance, as illustrated in the cases of Caltech-101 and CIFAR-100. We collect the proportions of "good neighbors" (samples belonging to the same category) in the first hundreds of retrieved samples (300 for Caltech-101, and 1000 for CIFAR-100). The samples within every bucket are randomly shuffled, and multiple candidate buckets with the same Hamming distance are also shuffled, so that the evaluation will not be affected by the order of the first retrieved samples (this operation is usually ignored in the evaluations of previous work). See Table 2 for the detailed experimental results, where the winning counts of each algorithm are also compared. To better illustrate the evolving tendencies of reconfigurable hashing, Figs. 4 and 5 plot the accuracies of selected categories from Caltech-101 and CIFAR-100, respectively. It is observed that the plotted curves on CIFAR-100 have more gentle slopes compared with Caltech-101's, which reveals the different characteristics of underlying data distributions, i.e., the samples from the same category in Caltech-101 gather more closely.

Although reconfigurable hashing is a meta-hashing framework, the ground hashing algorithms seriously affect the final performance. In Fig. 6, we plot the logarithm of the accuracy for each category on Caltech-101, employing either our proposed RARP or conventional LSH as described in Eqs. (3) and (1), respectively. RARP shows superior performance, which indicates that data-dependent hashing algorithms such as RARP are promising for future exploration.

7.2 MNIST-digit and CIFAR-10

The sample number of each category on Caltech-101 and CIFAR-100 is relatively small, ranging from 31 to 800. To complement the study in Sect. 7.1, we also conduct experiments on the benchmarks MNIST-Digit and CIFAR-10, which have larger sample number (6K or 7K) per category.

MNIST-Digit is constructed for handwritten digits recognition. It consists of totally 70,000 digit images, 7,000 images for each digit in $0 \sim 9$. The digits have been size-normalized to be 28×28 pixels. In our study, each digit image is transformed by matrix-to-vector concatenation and normalized to

Table 2 Reconfigurable hashing on multi-category benchmarks Caltech-101 and CIFAR-100

	RS	MU	MAM	WSE	Ours	NLS
Caltech-101	3.99	4.92	10.31	10.15	11.08	10.20
CIFAR-100	1.75	2.01	3.93	3.93	4.26	4.09
Caltech-101	0	0	2	1	99	
CIFAR-100	0	0	4	5	91	

The top table illustrates the averaged accuracies (%) of the first k retrieved samples ($k = 300$ for Caltech-101 and $k = 1,000$ for CIFAR-100). We also count the number of categories on which an algorithm beats all the others (102 or 100 in total on these two benchmarks). The results are reported in the bottom table

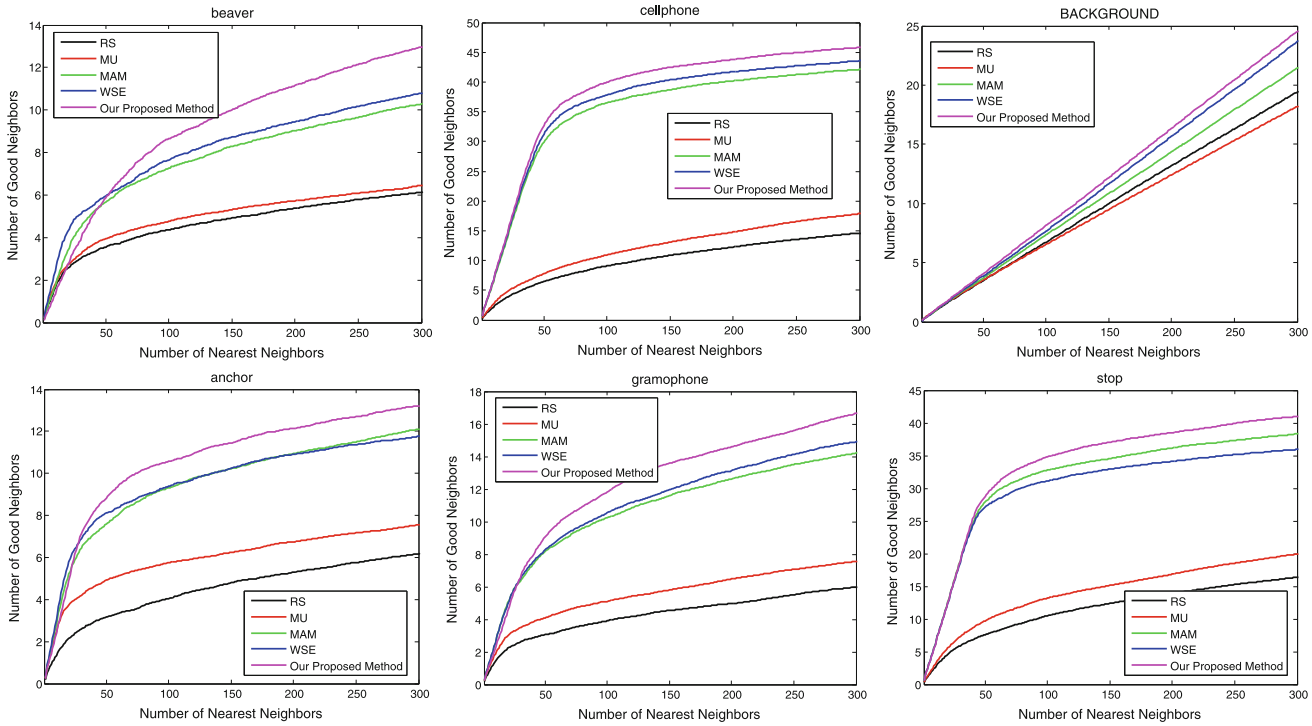


Fig. 4 The evolution of accuracies of the first 300 retrieved samples on randomly selected Caltech-101 categories

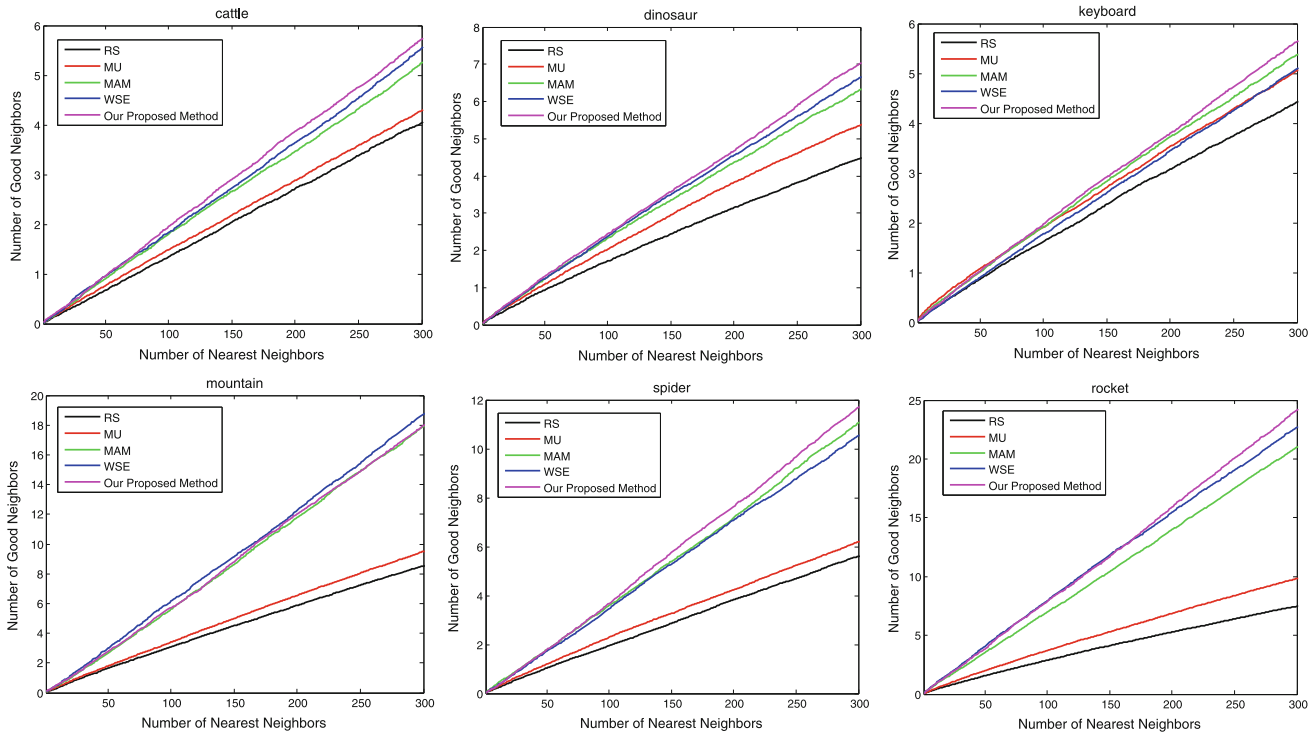


Fig. 5 The evolution of accuracies of the first 1,000 retrieved samples on randomly selected CIFAR-100 categories

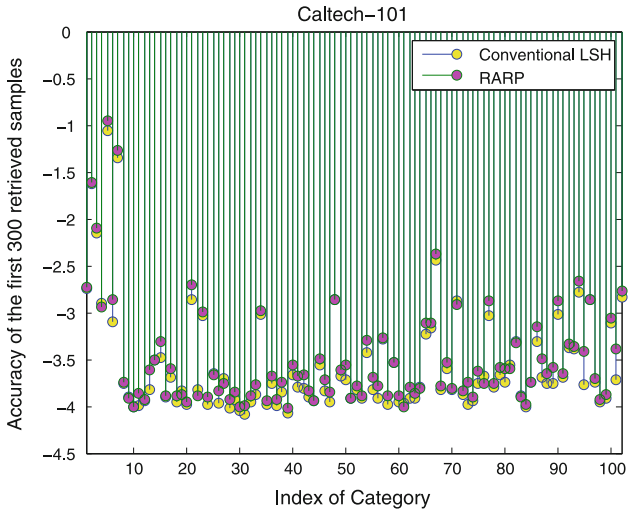


Fig. 6 Comparison of conventional random projection-based LSH (RP) and our proposed RARP on Caltech-101. Both utilize 14 bits in total. The accuracies for RARP and RP is 3.99 versus 3.61% averaged on 102 categories, obtained by 30 independent runs. Note that the accuracies are transformed by logarithm function for better viewing

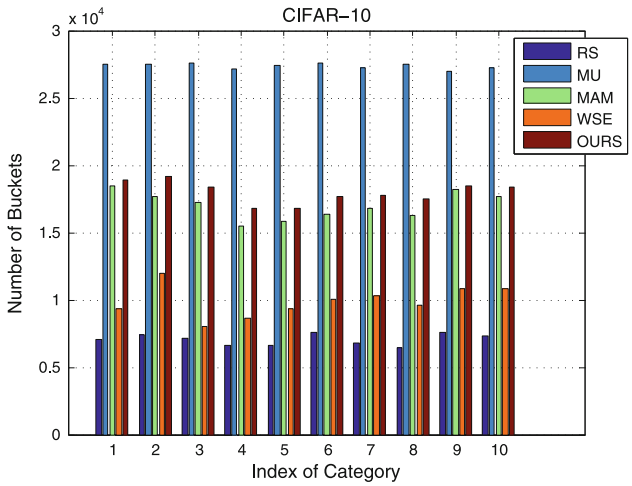


Fig. 7 Comparison of bucket numbers on CIFAR-10

be unit-length feature. These raw grayscale vectors directly serve as the low-level feature for recognition purpose.

Similar to CIFAR-100, CIFAR-10 is also a labeled subset of the 80 million tiny images data set, containing 60K 32×32 color images in ten classes (6K images for each class). The data set is constructed to learn meaningful recognition-related image filters whose responses resemble the behavior of human visual cortex. In the experiment we use the 387-d GIST image feature.

We learn category-dependent hashing schemes with 16 hashing bit budget. The experimental settings are identical to those on CIFAR-100, except that in the testing stage, only a portion of testing samples (300 in our implementation) are chosen for evaluation. Table 3 presents the results in terms of accuracy and winning count.

It is meaningful to investigate the correlation of the bucket number and the final performance. In Fig. 7, we plot the bucket number for each of the ten categories averaged over 30 independent runs. It is observed that MU results in the largest bucket numbers, which is consistent with its design principle. However, the retrieval performance of MU is only slightly better than random selection (RS), which negates the hypothesis that increasing bucket number will promote the performance with high probability. In contrast, WSE has the fewest buckets compared with the other three non-random algorithms, yet the performance is amazingly excellent (see Table 3). Intuitively, the Shannon entropy adopted in WSE favors hashing hyperplanes that cross the boundary between target category and its complemental categories. Such a strategy tends to keep the samples from target category stay closely in terms of Hamming distance and reduces unnecessary bucket creation. The high contrast between the small bucket number and high effectiveness suggests that the intelligent category-aware bucket creation is crucial for reconfigurable hashing. On the other hand, although both MAM and our proposed strategy utilize the idea of averaged margin, the latter brings slightly larger bucket number, which is supposed to stem from the regularization term $\mathcal{R}(W)$ defined in Eq. (8). It is observed that the combination of averaged margin and maximum unfolding improves the hashing quality.

8 Conclusions

In this paper, we investigate the possibility of effective hashing in the existence of diverse semantics and metric adap-

Table 3 Reconfigurable hashing on two 10-category image benchmarks MNIST-Digit and CIFAR-10

	RS	MU	MAM	WSE	Ours	NLS
CIFAR-10	14.23	15.58	21.06	20.51	21.92	25.14
MNIST-Digit	28.24	34.96	60.97	60.00	63.60	74.74
CIFAR-10	0	0	0	0	10	
MNIST-Digit	0	0	0	0	10	

The implications of the top and bottom tables are the same as in Table 2. Note that our proposed strategy wins on all the categories

tation. We propose a novel meta-hashing framework based on the idea of reconfigurable hashing. Unlike directly optimizing the parameters of hashing functions in conventional methods, reconfigurable hashing constructs a large hash pool by one-off data indexing and then selects the most effective hashing-bit combination at runtime. The contributions in this paper include a novel RARP-based hashing algorithm for ℓ_p norm, a novel bit-selection algorithm based on averaged margin and global unfolding-based regularization, and a comparative study of various bit-selection strategies. For the future research direction, we are working toward two directions:

- How to identify the correlation of different hashing bits and then mitigate its adverse effect is still an open problem in reconfigurable hashing. The current techniques are far from satisfactory. We believe that some tools developed in the information theory community are helpful.
- The effectiveness of a hashing algorithm heavily hinges on the characteristics of underlying data distributions. Developing a taxonomy about data distribution in the hashing context is especially useful.

References

1. Andoni A, Indyk P (2006) Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In: FOCS
2. Andoni A, Indyk P (2008) Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Commun ACM 51(1):117–122
3. Bentley J (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509–517
4. Broder AZ, Charikar M, Frieze AM, Mitzenmacher M (1998) Min-wise independent permutations. In: Proceedings of the thirtieth annual ACM symposium on theory of computing (STOC)
5. Charikar M (2002) Similarity estimation techniques from rounding algorithms. In: STOC
6. Datar M, Immorlica N, Indyk P, Mirrokni V (2004) Locality-sensitive hashing scheme based on p-stable distributions. In: SCG
7. Dong W, Wang Z, Charikar M, Li K (2008) Efficiently matching sets of features with random histograms. In: ACM Multimedia
8. Grauman K, Darrell T (2005) The pyramid match kernel: discriminative classification with sets of image features. In: ICCV
9. He J, Liu W, Chang SF (2010) Scalable similarity search with optimized kernel hashing. In: SIGKDD
10. He X, Niyogi P (2003) Locality preserving projections. In: NIPS
11. Indyk P, Motwani R (1998) Approximate nearest neighbors: towards removing the curse of dimensionality. In: STOC
12. Indyk P (2006) Stable distributions, pseudorandom generators, embeddings, and data stream computation. J ACM 53(3):307–323
13. Ke Y, Sukthankar R, Huston L (2004) An efficient parts-based near-duplicate and sub-image retrieval system. In: ACM Multimedia
14. Kulis B, Grauman K (2009) Kernelized locality-sensitive hashing for scalable image search. In: ICCV
15. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
16. Moosmann F, Nowak E, Jurie F (2008) Randomized clustering forests for image classification. IEEE Trans PAMI 30(9):1632–1646
17. Motwani R, Naor A, Panigrahi R (2006) Lower bounds on locality sensitive hashing. In: SCG
18. Mu Y, Shen J, Yan S (2010) Weakly-supervised hashing in kernel space. In: CVPR
19. Mu Y, Sun J, Han TX, Cheong LF, Yan S (2010) Randomized locality sensitive vocabularies for bag-of-features model. In: ECCV
20. Mu Y, Yan S (2010) Non-metric locality-sensitive hashing. In: AAAI
21. Nowak E, Jurie F, Triggs B (2006) Sampling strategies for bag-of-features image classification. In: ECCV
22. 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
23. Paulevé L, Jégou H, Amsaleg L (2010) Locality sensitive hashing: a comparison of hash function types and querying mechanisms. Pattern Recognit Lett 31(11):1348–1358
24. Salakhutdinov R, Hinton G (2009) Semantic hashing. Int J Approx Reason 50(7):969–978
25. Scholkopf B, Smola AJ (2001) Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge
26. Shakhnarovich G, Viola PA, Darrell T (2003) Fast pose estimation with parameter-sensitive hashing. In: ICCV
27. Wang F, Zhang C (2007) Feature extraction by maximizing the average neighborhood margin. In: CVPR
28. Wang J, Kumar S, Chang SF (2010) Semi-supervised hashing for scalable image retrieval. In: CVPR
29. Wang J, Kumar S, Chang SF (2010) Sequential projection learning for hashing with compact codes. In: ICML
30. Wang M, Song Y, Hua XS (2009) Concept representation based video indexing. In: SIGIR
31. Weiss Y, Torralba A, Fergus R (2008) Spectral hashing. In: NIPS
32. Xing EP, Ng AY, Jordan MI, Russell SJ (2002) Distance metric learning with application to clustering with side-information. In: NIPS