Accelerated Large Scale Optimization by Concomitant Hashing

Yadong Mu, John Wright, and Shih-Fu Chang

Electrical Engineering Department,
Columbia University, New York, NY 10027.
{muyadong, johnwright, sfchang}@ee.columbia.edu

Abstract. Traditional locality-sensitive hashing (LSH) techniques aim to tackle the curse of explosive data scale by guaranteeing that similar samples are projected onto proximal hash buckets. Despite the success of LSH on numerous vision tasks like image retrieval and object matching, however, its potential in large-scale optimization is only realized recently. In this paper we further advance this nascent area. We first identify two common operations known as the computational bottleneck of numerous optimization algorithms in a large-scale setting, *i.e.*, min/max inner product. We propose a hashing scheme for accelerating min/max inner product, which exploits properties of order statistics of statistically correlated random vectors. Compared with other schemes, our algorithm exhibits improved recall at a lower computational cost. The effectiveness and efficiency of the proposed method are corroborated by theoretic analysis and several important applications. Especially, we use the proposed hashing scheme to perform approximate ℓ_1 regularized least squares with dictionaries with millions of elements, a scale which is beyond the capability of currently known exact solvers. Nonetheless, it is highlighted that the focus of this paper is not on a new hashing scheme for approximate nearest neighbor problem. It exploits a new application for the hashing techniques and proposes a general framework for accelerating a large variety of optimization procedures in computer vision.

1 Introduction

The rapid development of camera hardware and online sharing communities has enabled the proliferation of the gigantic image and video corpora at the scale of millions or even billions. For example, ImageNet[1] consists of over 12 Million Web-crawled images corresponding to around 17,000 WordNet nouns. Another recent notable large-scale benchmark called Multimedia Event Detection (MED)[2] (part of TRECVID 2011), is comprised of over 45,000 videos with an average duration of 2 minutes. To cope with the increased content complexity associated with gigantic data sets, more sophisticated feature representations have been introduced in recent literatures, but unfortunately often lead to very

[1] `http://www.image-net.org`
[2] `http://www.nist.gov/itl/iad/mig/med11.cfm`

high dimensionality. Facing the aforementioned trends, significant attempts have been devoted to overcoming the curses of gigantic corpora and dimensionality, either by speeding up non-linear kernel computation (*e.g.*, accelerated histogram intersection kernel [1]), efficient large-scale visual vocabulary construction [2] or low-rank matrix approximation [3].

Our work in this paper is inspired by the recent impressive advance of data indexing techniques, especially *locality-sensitive hashing* (LSH) [4]. Many computer vision tasks (*e.g.*, image retrieval, object classification) can be treated as k-nearest-neighbor (k-NN) problem under task-specific metrics. The goal of LSH is to provide fast approximation for the exact, albeit expensive, linear scan method for k-NN retrieval. For a database of N samples, it can be proved that LSH-based approximate k-NN can be accomplished with a sub-linear complexity of $\mathcal{O}(N^{\rho})$ where $\rho \in (0, 1)$ is a constant that depends on the desired level of approximation. This greatly mitigates the computational burden. Previous endeavors have also tailored LSH along several directions, including hashing new types of objects (*e.g.*, subspaces [5]) or hashing with non-Euclidean metrics (*e.g.*, kernelized hashing [6], non-metric LSH [7] and anchor graph [8]).

It is only recently realized that LSH can also facilitate efficient large-scale optimization. A very relevant work is the *point-to-hyperplane hashing* method proposed in [9]. It is used to accelerate margin-based active learning SVM, which iterates between informative sample selection and SVM hyperplane update. To maximally reduce the uncertainties of un-labeled data, a theoretically-guaranteed effective strategy is to choose those samples closest to the current SVM hyperplane [10], which can be achieved in reduced complexity via the afore-mentioned point-to-hyperplane hashing technique [9].

Our paper focuses on the two fundamental linear operations (min/max inner product) that are at the core of solving several large optimization problems (like sparse coding, OMP, max-margin clustering and active learning SVM). Assume $\omega, x \in \mathbb{R}^d$, where ω is the "query" vector (*e.g.*, the hyperplane normal in SVM) and x is a specific sample in a database of size N.

- **Min-Product**: the aim is to select a k-cardinality set which encompasses the samples with the smallest absolute responses to ω.

$$\mathcal{X} = \left\{ x \;\; | \;\; |\omega^T x| \leq |\omega^T x'|, \;\; \forall \;\; x' \notin \mathcal{X} \;\; \right\}. \tag{1}$$

- **Max-Product**: the aim is akin to Min-Product, yet pursuing those with largest absolute response magnitudes.

$$\mathcal{X} = \left\{ x \;\; | \;\; |\omega^T x| \geq |\omega^T x'|, \;\; \forall \;\; x' \notin \mathcal{X} \;\; \right\}. \tag{2}$$

Conventional LSH algorithms are not suitable for solving Min-Product or Max-Product. Especially, Min-Product frustrates most of prior LSH schemes since they are not designed to find perpendicular vectors. The only relevant work was done by Jain et al. [9], yet it suffers from the low collision probability as later shown in Fig. 2.

We propose a new hashing scheme based on concomitant order statistics [11, 12]. The proposed *concomitant hashing* scheme is suitable for handling both Min-Product and Max-Product operations with very nice theoretical guarantees. It can serve as a general tool to accelerate a large family of large-scale optimization problems. The reminder of the paper is organized as following: Section 2 illustrates several exemplar applications of the concomitant hashing. Section 3 describes our proposed hashing scheme and theoretic analysis. Section 4 elaborates on experimental study. Finally we conclude the work in Section 5.

2 Exemplar Applications

2.1 Min-Product: Active Learning SVM and Maximum Margin Clustering

The proposed scheme can be applied to scale up a number of tasks fitting the specifications described in Section 1. Two representative applications for the Min-Product case are active learning SVM [10] and maximum margin clustering [13]. A popular active-selection strategy for the former problem is to choose those samples closest to current hyperplane (*i.e.*, those with minimal absolute inner products to the hyperplane normal), based on the theoretic guarantee of optimally reducing the volume of version space [10].

On the other hand, the formulation of maximum margin clustering problem is similar to conventional SVM, *i.e.*,

$$\min_{y\in\{-1,1\}^N,\ b,\ \xi\geq 0} \frac{1}{2}w^T w + C\sum_{i=1}^{N}\xi_i \quad s.t.\ \ y_i\left(w^T x_i + b\right) \geq 1-\xi_i,\ \forall i,$$

where the key difference to conventional SVM is that the data labels y are unknown variables. Zhao et al. [13] tailored the *cutting plane* method for iterative minimization. The set of constraints is initially empty. A new linear constraint is added in each iteration to progressively tighten the lower bound of the objective function. Since the constraint is generated by identifying those samples close to current hyperplane, *i.e.*, $|w^T x_i + b| < 1$, it falls into the Min-Product case.

2.2 Max-Product: Sparse Optimization and Gaussian Process Regression

Max-Product encompasses a variety of optimization problems with maximal absolute correlation greedy selection, including Lasso, orthogonal matching pursuit (OMP), and Gaussian process regression (GPR), whose formulations are:

$$\textbf{(Lasso)}:\ \min_{\alpha}\ \|x - D\alpha\|^2 + \lambda\|\alpha\|_1. \tag{3}$$

$$\textbf{(OMP)}:\ \min_{\alpha}\ \|x - D\alpha\|^2,\ s.t.\ \|\alpha\|_0 \leq k. \tag{4}$$

$$\textbf{(GPR)}:\ \min_{\alpha}\ \frac{1}{2}\alpha^T\left(K + \sigma^2 I\right)\alpha - y^T\alpha. \tag{5}$$

We postpone more details about OMP and GPR to the supplementary material, and focus on the Lasso problem here. The Lasso is widely used for computing sparse representations [14] of signals arising in vision tasks. When the optimal solution $\alpha^\star$ is truly sparse, a natural solution is to pre-detect a subset that covers the true supporting columns of D, which is the idea underlying Grafting [15] and Grafting-Light [16]. Both methods maintain a working set $\mathcal{S}$ initialized as empty and alternate between the following steps until convergence:

Step-1: Perform one-step [16] or multi-step gradient descent [15] over the working set $\mathcal{S}$ for Problem (3).

Step-2: Compute the gradient $D^T(D\alpha_0 - x)$ at current solution α_0 and select top m candidate samples from the inactive set $\{1, \ldots, |D|\} - \mathcal{S}$ with largest gradient magnitudes above λ. If the candidate set is not empty, add them into working set $\mathcal{S}$ and go to Step-1, otherwise exit.

The computational bottleneck of above algorithm lies in efficiently selecting active columns in the scenario of large D (possibly on million scale). It obviously fits Max-Product case and can be largely accelerated via data hashing.

3 Concomitant Hashing

3.1 Basics of Concomitant Order Statistics

David et al. [11] gave a nice introduction to the concomitant rank order statistics (it was termed "induced order statistics" in [17] and [18]) based on small-sample theory. More asymptotic results can be found in [19]. Assume the pair set $\{(x_i, y_i)\}_{i=1}^n$ $(\forall\ i, x_i, y_i \in \mathbb{R})$ are independently sampled from the latent distribution $f(x, y)$. In the literature [11], y_i is called the concomitant of x_i, and vice-verse. Let x_k be the r-th smallest element among $x_1, \ldots, x_n$. We follow the notations in [11], using $\Pi_{r,s}^n$ to represent the probability of its concomitant y_k being the s-th smallest among $y_1, \ldots, y_n$. It can be defined as:

$$\Pi_{r,s}^n = \sum_{k=1}^{n} Pr\left(rank(x_k) = r, rank(y_k) = s\right), \tag{6}$$

where $rank(\cdot)$ returns the ordinal rank in ascending order.

3.2 Our Proposed Concomitant Hashing Method

Recently, Eshghi et al. introduced a new class of hashing functions based on the concomitant rank order statistics [12]. However, it was designed for the conventional nearest neighbor search problem. We extend the work of [12] to expedite solving both Min-Product and Max-Product operations facing large-scale data. Fig. 1 depicts the algorithmic pipeline. For each feature $x \in \mathbb{R}^d$, the hashing scheme consists of three steps: 1) generate a random matrix $R \in \mathbb{R}^{n \times d}$ with elements randomly drawn from normal distribution (typically $n \ll d$) and compute $u = Rx \in \mathbb{R}^n$, 2) sort the elements of $u = (u(1), \ldots, u(n))$ in the

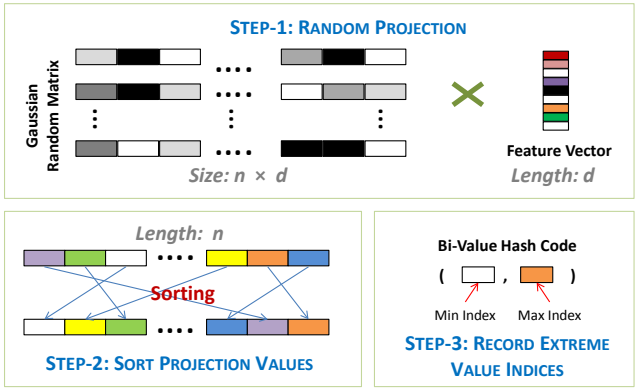

Fig. 1. Illustration of the algorithmic pipeline of the proposed hashing scheme.

ascending order, and finally 3) denote i_{min}, i_{max} to be the indices of the smallest and largest elements of u respectively. Generate the bi-value hash code $h(x) = \{h_1(x), h_2(x)\}$ with $h_1(x) \leftarrow i_{min}$ and $h_2(x) \leftarrow i_{max}$. Two vectors x, y collide if they generate the same set $h(x)$, ignoring the order of h_1 and h_2. The subtle order-removing operation in the third step clearly distinguishes ours from prior work [12], which enables the new scheme to detect min/max correlations.

Given a new query, we use Hamming distance to the query to rank all the data. Extensions based on the hash tables as in conventional LSH algorithms [4] will be introduced in a longer version. Our algorithm first computes the query's hash code, and afterwards calculates the Hamming distances to the hash codes in the database. These distances are then used to filter out most irrelevant samples, obtaining a shortlisted candidate set. Particulary, Min-Product and Max-Product operations significantly diverge in the filtering criterion. Our proposed hashing scheme tends to throw nearly-orthogonal vectors into different hash buckets, and keeps those highly-correlated ones in the same bucket. Therefore, the filter rule for Max-Product is to keep those samples with smallest Hamming distances, as in the conventional random projection-based LSH [20]. For Min-Product operation, the candidate set is comprised of those with largest Hamming distances. The computation can be simplified by inverting each bit of the query's hash code, as previously used in H-Hash [9].

3.3 Theoretical Analysis

Our analysis starts from the fact that Gaussian random projections of fixed vectors are jointly Gaussian:

Proposition 1. *Given two ℓ_2-normalized vectors $x_1, x_2 \in \mathbb{R}^d$, project them onto specific random matrix $R \in \mathbb{R}^{n\times d}$ via $u_1 = \frac{1}{\sqrt{n}}Rx_1$, $u_2 = \frac{1}{\sqrt{n}}Rx_2$, where the elements of R are independent standard normal random variables. The following*

holds (Lemma 1.3 of [21]):

$$E\left(\|u_1\|^2\right) = E\left(\|u_2\|^2\right) = 1, \quad E\left(u_1^T u_2\right) = x_1^T x_2 = \rho, \tag{7}$$

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma = n^{-1} \cdot \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right), \tag{8}$$

where ρ is the inner product between normalized vectors x_1, x_2.

Proposition 1 indicates that the projected variables are subject to the following bivariate Gaussian distribution after appropriate scaling[3], *i.e.*,

$$f(x,y) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2} \cdot \frac{x^2 - 2\rho xy + y^2}{1-\rho^2} \right). \tag{9}$$

We can express any $\Pi_{r,s}^n$ in terms of $f(x,y)$. For example,

$$\Pi_{1,1}^n = n \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (\lambda(x,y))^{n-1} f(x,y)\, dx\, dy, \tag{10}$$

where $\lambda(x,y) = \int_x^{\infty} \int_y^{\infty} f(u,v)\, du\, dv$ represents the marginal probability that the pair (x,y) satisfies $x_k \geq x, y_k \geq y$ for $k = 1, \ldots, n$.

Recall that the collision of two vectors is identified via the same bi-value hash index set. Since we ignore the orders of index set elements, for two ℓ_2-normalized vectors $x_1, x_2 \in \mathbb{R}^d$, there is two disjoint events that make them collide, *i.e.*, $h_1(x_1) = h_1(x_2), h_2(x_1) = h_2(x_2)$ or $h_1(x_1) = h_2(x_2), h_2(x_1) = h_1(x_2)$. Therefore the overall collision probability is the sum of two terms (define ρ_{x_1,x_2} to be the inner product between x_1 and x_2):

$$\begin{aligned} & Pr\left(h(x_1) = h(x_2)\right) \\ &= \Pi_{1,1}(\rho_{x_1,x_2}) \cdot \Pi_{n,n}(\rho_{x_1,x_2}) + \Pi_{1,n}(\rho_{x_1,x_2}) \cdot \Pi_{n,1}(\rho_{x_1,x_2}) \\ &= (\Pi_{1,1}(\rho_{x_1,x_2}))^2 + (\Pi_{1,n}(\rho_{x_1,x_2}))^2 \qquad (11) \\ &= (\Pi_{1,1}(\rho_{x_1,x_2}))^2 + (\Pi_{1,1}(-\rho_{x_1,x_2}))^2, \qquad (12) \end{aligned}$$

where Equations (11) and (12) hold following the facts $\Pi_{r,s}^n = \Pi_{n-1-r,n-1-s}^n$, $\Pi_{1,1}^n(\rho) = \Pi_{1,n}^n(-\rho)$ respectively (see supplementary material).

The following proposition shows that the larger $|x_1^T x_2|$ is, the more likely x_1 and x_2 are to collide. (see supplementary material for proof):

Proposition 2. *For any pair of normalized vectors (x_1, x_2), the collision probability $P[h(x_1) = h(x_2)]$ depends only on their inner product: $P[h(x_1) = h(x_2)] = g(x_1^T x_2)$. The function $g(\rho)$ is symmetric about $\rho = 0$, and monotonically increasing on $(0, 1)$.*

In Fig. 2, we plot the collision probabilities of several hashing schemes. Note that given more projection bases (*i.e.*, larger n), it is observed the collision

[3] The scaling operation does not affect the theoretic results presented in this paper since it does not change the sorting results in the proposed hashing scheme.

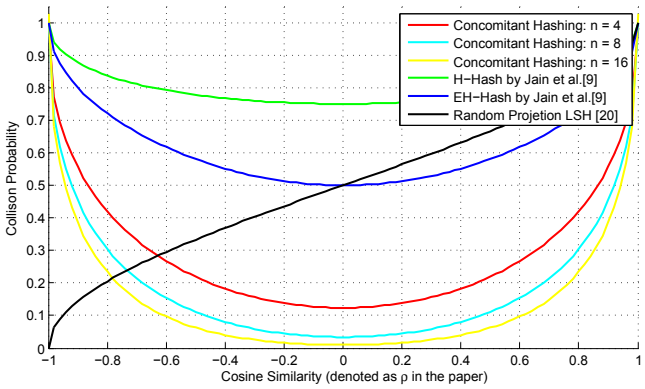

Fig. 2. Comparison of collision probabilities for different hashing schemes under cosine similarity. Better viewing in color.

rate at $\rho = 0$ continuously drops towards zero. In fact, from the simple fact that $\Pi_{1,1}(0) = 1/n$, Equation (12) further tells us that the collision rate of our proposed scheme at $\rho = 0$ has the closed form $2/n^2$, which is significantly lower than that of H-Hash (0.25) or EH-Hash (0.5) [9] and therefore suggests better ability per hashing function for identifying near-perpendicular vectors.

More importantly, note that the hashing schemes use different numbers of bits to store the result of a single hashing function. For example, our proposed scheme requires $k_n = 2 \times \lceil \log(n) \rceil$ bits per function (for the max and min indices), while H-Hash needs 2 bits. It is meaningful to investigate the bitwise property. Hereafter we only concern $\rho = 0$, since it represents the perfect matching (perpendicular vectors) for Min-Product operation and is of great interest. For the k_n bits used by concomitant hashing, denote the bitwise collision rates by $p_1, \ldots, p_{k_n}$ respectively. Recall that some bits are statistically dependent on others (*e.g.*, the last half bits denote the max index, whose decimal value is supposed to exceed that of the first half bits), therefore $\forall\ i, j, i \neq j$, in general $p_i \neq p_j$. Let $\bar{p}_n$ be the averaged collision rate. From Jensen's Inequality,

$$\bar{p}_n = \frac{1}{k_n}\left(p_1 + p_2 + \ldots + p_{k_n}\right) \geq \left(p_1 p_2 \cdots p_{k_n}\right)^{1/k_n} \tag{13}$$

It is easily verified that when $\log(n)$ is an integer, we have that

$$\lim_{n\to+\infty} \bar{p}_n \geq \lim_{n\to+\infty} \left(p_1 p_2 \cdots p_{k_n}\right)^{1/k_n} = \lim_{n\to+\infty} 2^{\frac{1}{2\log n}} \cdot 2^{-1} = \frac{1}{2}, \tag{14}$$

where we use the fact that $p_1 p_2 \cdots p_{k_n} = 2/n^2$ obtained from Equation (12).

Likewise, it can be proved that the bitwise collision rate for H-Hash at $\rho = 0$ is $\sqrt{3}/2$ and 0.5 for EH-Hash. As mentioned in Section 3.2, inversion of query's bits is required for solving Min-Product problem. Consequently a lower collision rate at $\rho = 0$ is highly desired. In this sense, our proposed scheme is superior

to H-Hash and comparable to EH-Hash. Nonetheless, note that EH-Hash first uplifts the original feature vectors of d dimension into d^2 (on million scale for a typical feature used in computer vision), which is computationally unaffordable. Due to space limit, more theoretic analysis is presented in the supplementary material.

3.4 Complexity Analysis and Practical Issues

Complexity: We lose the appealing sub-linear complexity since we choose the Hamming ranking method for retrieval. The overall computational complexity of the proposed framework is linear to the size of the database. In fact, the sub-linear complexity for Min-Product remains an open problem[4]. In supplemental material, we provide a discussion on the complexity of the proposed method based on the data structure of hash tables as in conventional LSH techniques.

Shared Hash Bases: The proposed hashing scheme has a complexity of $\mathcal{O}(Bnd)$ when calculating B hashing functions during the indexing phase, which is relatively higher than $\mathcal{O}(Bd)$ of conventional LSH [20]. The drawback can be mitigated by sacrificing the independence among different random projection matrix in $\mathbb{R}^{n\times d}$. Particularly, in practice we build a large pool of P random projection bases, and randomly select n bases each time. At most a large set of $\binom{P}{n}$ unique random matrices can be rendered from the pool. In this way, the random projections are used multiple times once computed. An empirical study about the choice of P is presented in Section 4.1.

4 Experiments

The effectiveness of the proposed method is demonstrated by three applications on five large-scale computer vision benchmarks (MNIST-digit, Tiny-1M, CIFAR-10, NUS-WIDE, and ImageNet). All features are ℓ_2-normalized. Extension to un-normalized data will be presented in a longer version of this paper.

4.1 Large-Scale Correlative Image Ranking

The first experiment aims to investigate the performance in large-scale correlation-based image ranking for Min-Product or Max-Product. Given a query image, we sort the images in the database according to their Hamming distances to the query, and contrast the obtained ranks to the ground truth. We choose two standard image benchmarks, including MNIST-digit[5] and Tiny-1M. The former benchmark is comprised of 70,000 handwritten digits, including 7K images for digits $0-9$. We select 1K images per digit as queries. Each 28×28-pixel image is converted into 786-d vector by matrix-to-vector concatenation and afterwards

[4] Jain et al. [9] present a complexity analysis yet the derived bound therein is dependent on parameter r.

[5] `http://cs.nyu.edu/~roweis/data.html`

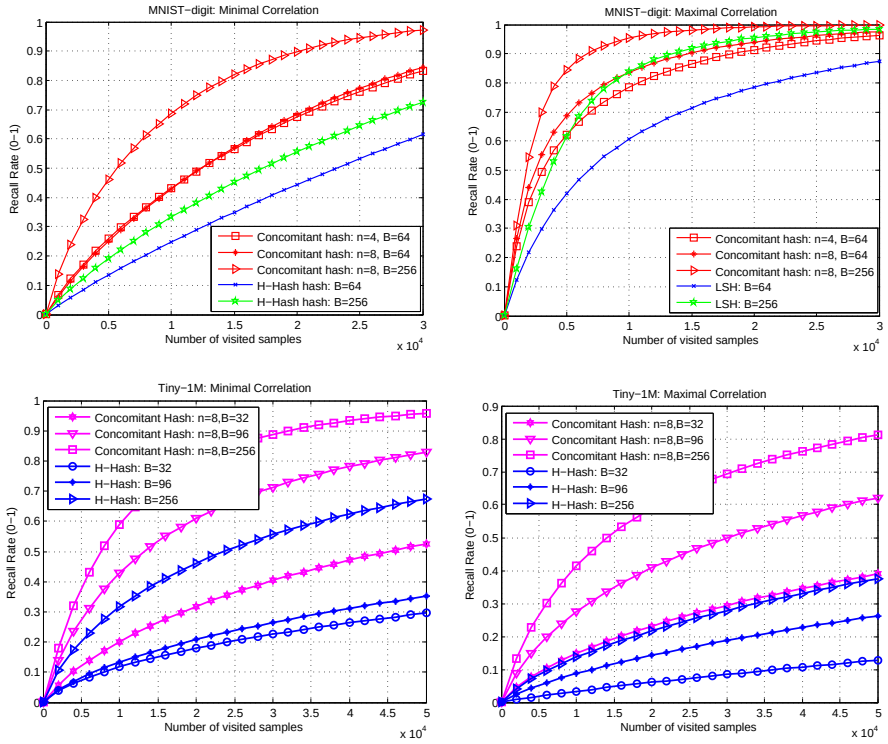

Fig. 3. Illustration of correlative image ranking on MNIST-digit and Tiny-1M. The left column presents the performances for Min-Product, and the right column shows the performance for Max-Product on both datasets. See text for details.

ℓ_2-normalized. The Tiny-1M benchmark is a 1M-cardinality random subset from 80M Tiny Images[6], which are Web-crawled images based on all the nouns in the English language. We extract 384-d GIST features from these images and choose 1K images as queries. Since the goal is mainly to evaluate correlation-based ranking, we ignore the associated labels and generate ground truth from m most (or least) correlated samples ($m = 3,000$ for MNIST-digit and $m = 10,000$ for Tiny-1M).

The performances are shown in Fig. 3. The results for both Min-Product and Max-Product cases are reported for MNIST-digit and Tiny-1M, contrasting baseline algorithms (*i.e.*, H-Hash [9] and conventional LSH [20] respectively). Another Min-Product algorithm, EH-Hash [9] is not included since it first uplifts the original features to a d^2-dimensional space and thus the complexity is beyond the scope of any practitioners. The performances of concomitant hashing are consistently superior to others. For example, in minimal-correlation based ranking, concomitant hashing returns nearly all 3,000 ground truth among first

[6] `http://horatio.cs.nyu.edu/mit/tiny/data/index.html`

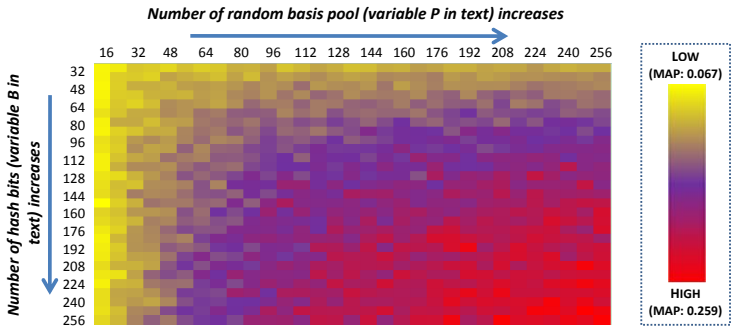

Fig. 4. Sensitivity study about hash bits B and hash pool size P. Results are statistical average over multiple runs. We fix $n = 8$ in all runs. Better viewed in color mode.

30,000 visited samples ($n = 8$ and $B = 256$) while H-Hash only has a roughly 72% recall rate ($B = 256$).

We also perform a parameter sensitivity investigation on MNIST-digit with respective to the hash bit number B and the hash pool size P. The performances are measured by mean-average-precision (MAP) and averaged over multiple independent runs. Fig. 4 is its visualization. Two observations are made: 1) the performance is stable when the global hash pool size exceeds a reasonable value (*e.g.*, $P = 128$), which empirically corroborates the idea of "global pool" in Section 3.4, and 2) larger hash bit number B consistently improve the accuracy.

4.2 Large-Scale Active Learning SVM

Another interesting Min-Product application is large-scale active learning [10]. We conduct experiment on CIFAR-10[7], which contains 60,000 images from ten categories (*e.g.*, bird, truck etc.). We extract 384-d GIST features from each image and follow the evaluation settings in [9]: for all classes, an initial linear SVM is trained in one-vs-all manner using randomly-chosen 5 samples and afterwards it runs active selection for 300 iterations. The performances are reported in Fig. 5, which contrasts the proposed method with H-Hash, random selection, linear scan, and sequential feeding (*i.e.*, randomly add one more positive and negative samples respectively in each iteration. It departs from active learning yet serves as a contrastive baseline to the active learning strategy). It is observed that active learning algorithms outruns both sequential feeding and random selection. Both concomitant hash and H-hash approach the performance of linear scan method with largely reduced number of visited samples. The accuracies in term of $|w^T x|$ for chosen sample slightly drop in contrast to exhaustive search. Although not as salient as in the correlative ranking experiment, concomitant hash still outperforms H-hash in terms of both MAP and the quality of selected samples.

[7] `http://www.cs.toronto.edu/~kriz/cifar.html`

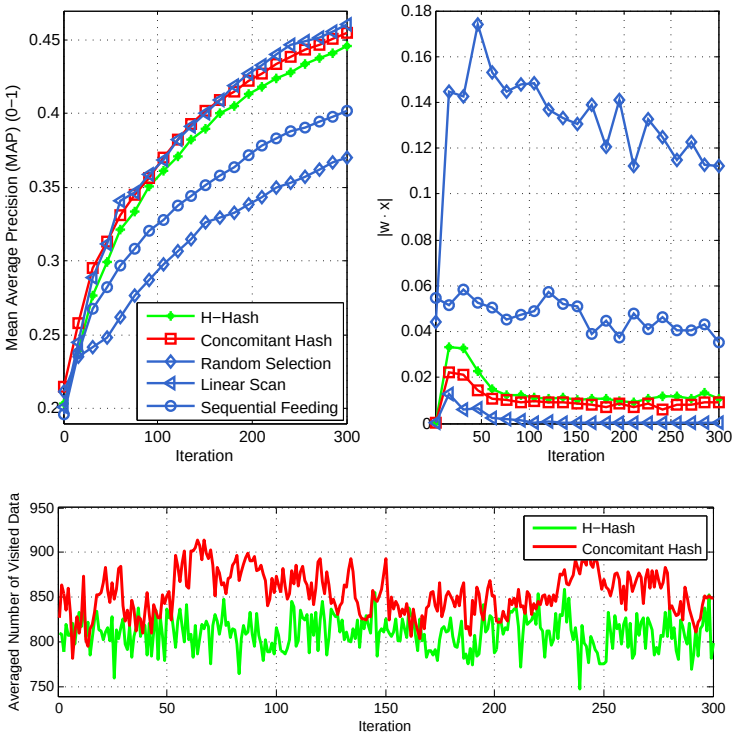

Fig. 5. Illustration of active learning SVM on CIFAR-10 dataset. *Top left*: mean-average-precision (MAP) values based on trained SVM models in first 300 iterations for different hashing schemes. We set $B = 96$ for both concomitant hash and H-hash. *Top right*: averaged values of $\left|w^T x\right|$ for the chosen samples (the legend of the figure is identical to the top-left one). *Bottom*: averaged visited samples per query in first 300 iterations. The "linear scan" method visits all rest samples in each iteration.

4.3 Sparse Coding with Million-Scale Dictionary

One open problem for sparse methods is the development of scalable solvers. In this section we apply the proposed method to accelerate two sparse algorithms (Lasso and OMP, see Section 2.2) on two large-scale image datasets: NUS-WIDE[8] and ImageNet[9]. Table 1 describes the benchmarks, both of which are split into a dictionary set and query set. We assume that the tags or labels are known on the dictionary set. The goal is to infer the semantic information of the query set without any training effort. Following the classical sparse method [14], each image from the query set is sparsely reconstructed from the dictionary set based on the proximity of low-level features. Under the assumption that low-level features and semantic labels share the same reconstruction coefficients, they are

[8] `http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm`
[9] `http://www.image-net.org/challenges/LSVRC/2010`

Table 1. Data set description and experimental results on NUS-WIDE and ImageNet. For baseline k-NN algorithm we choose $k = 200$. The parameter k in Problem (4) for all OMP-related algorithms are set as 20 after empirical tuning. $\lambda = 0.1$ for Lasso.

	Dictionary Set	Query Set	Label	Feature
NUS-WIDE	162,642	107,859	81	853
ImageNet	1,261,406	50,000	1,000	1,000

Method	NUS-WIDE MAP Value (0-1)	ImageNet Accuracy (0-100)
k-NN	0.1346	7.47%
Linear SVM	0.1382	8.95%
Lasso-Random	0.0601	0.56%
Lasso-Local	0.1654	8.96%
Lasso-Kmeans	0.1255	5.69%
Lasso-Grafting	**0.1977**	**10.34%**
OMP-Local	0.1343	8.17%
OMP-Kmeans	0.0952	6.01%
OMP-All	0.1259	3.27%

then used to predict the labels (or tags) for each query image. Performance on NUS-WIDE are reported in terms of mean-average-precision (MAP) across 81 tags, and for ImageNet we adopt mean accuracy over all classes.

With such million-scale dictionaries, exact Lasso solvers such as LARS/homotopy or proximal gradient methods are too slow to be practical. For example, on NUS-WIDE, Lasso solver like LARS is estimated to take around one-week computation for processing all queries on our computer equipped with quad-core CPU and 18GB memory. We show two ways of accelerating these algorithms using hashing. While our approach does not guarantee a good approximation to the Lasso solution, in practice it is significantly more scalable.

Scheme-1 (query-adaptive local dictionary): namely, for specific query, k dictionary items with largest correlation magnitudes are retrieved based on our proposed hashing method in sub-linear complexity. Afterwards, we run the standard Lasso solver on the reduced query-adaptive local dictionary. This scheme is inspired from the observation in [22], where data are models as mixtures and each mixture shares a local dictionary. It corresponds to the "Lasso-Local" algorithm in Table 1 with $k = 200$.

Scheme-2 (hashing-accelerated Grafting): it corresponds to the "Lasso-Grafting" in Table 1. Full dictionary set is used as described in Section 2.2.

Table 1 summarizes the performance on two datasets. We also compare with other algorithms: "Lasso-Random" performs sparse reconstruction on 200 randomly-selected dictionary items. We also group dictionary into 1,000 clusters ("Lasso-Kmeans"), which are used as reduced dictionary set universally employed for all queries. They both serve as contrastive baselines to "Lasso-Local" and "Lasso-Grafting". The mechanisms for OMP algorithms are similar. The peak performance on NUS-WIDE is achieved by "Lasso-Grafting", superior to

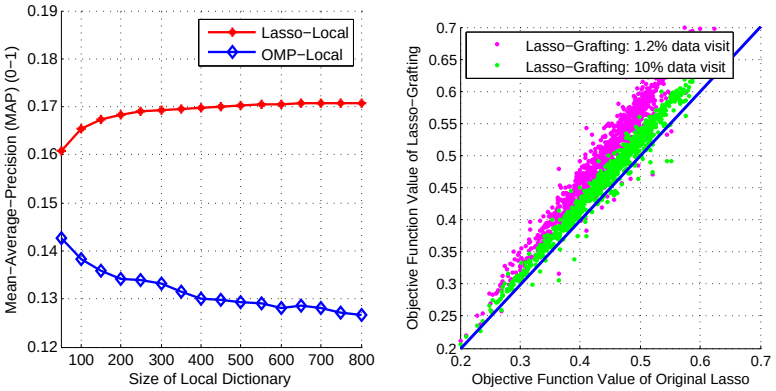

Fig. 6. Investigation of "Lasso-Local" (left) and "Lasso-Grafting" (right) on NUS-WIDE. The left figure illustrates how the choice of local dictionary size affects the performance for "Lasso-Local" and "OMP-Local". Using 2D scatters, the right figure highlights the proximity of achieved solutions via "Lasso-Grafting" (vertical axis) and original Lasso solver (horizontal axis).

state-of-the-art results based on sophisticated graph-oriented tag diffusion [23]. Moreover, we empirically study how the size of local dictionary affects final accuracies, as shown in Fig. 6. The OMP-based methods are saliently inferior. Fig. 6 also contrast the achieved minimum of objective function in Problem (3), which shows high fidelity even only 1.2% dictionary set is visited.

Besides, the computational speedup is notable. On NUS-WIDE, conventional LARS solver takes around 13 seconds for a query, which is reduced to 0.26 second in "Lasso-Grafting". The time on ImageNet for each query is slightly higher ($\sim$ 0.8 second) due to the increased dictionary size. Recall that the grafting method is guaranteed to converge to the global optimum [16]. Empirically we observe that the "Lasso-Grafting" method returns a solution whose objective function is only within a moderate multiplicative factor of the optimal. Rigorous justification for general input data is left for future work.

5 Conclusions

In this paper we first analyze the common computational bottlenecks in a large variety of large-scale optimization problems that are widely used in computer vision tasks. Two typical sub-routines are summarized as Min-Product and Max-Product operations, and a novel general hashing scheme is proposed based on the concomitant order statistics theory. It shows superior performance compared with prior works and we further develop an efficient practical implementation based on the idea of global hash pool. Comprehensive experiments are provided to validate the superior performance of the proposed methods. For future work, we plan to exploit more of its applications.

References

1. Wang, G., Hoiem, D., Forsyth, D.: Learning image similarity from flickr groups using stochastic intersection kernel machines. In: ICCV. (2009)
2. Mu, Y., Sun, J., Han, T.X., Cheong, L.F., Yan, S.: Randomized locality sensitive vocabularies for bag-of-features model. In: ECCV. (2010)
3. Talwalkar, A.: Matrix Approximation for Large-scale Learning. PhD thesis, New York University (2010)
4. Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Commun. ACM **51**(1) (2008) 117–122
5. Basri, R., Hassner, T., Zelnik-Manor, L.: Approximate nearest subspace search. IEEE Trans. Pattern Anal. Mach. Intell. **33**(2) (2011) 266–278
6. Kulis, B., Grauman, K.: Kernelized locality-sensitive hashing for scalable image search. In: ICCV. (2009)
7. Mu, Y., Yan, S.: Non-metric locality-sensitive hashing. In: AAAI. (2010)
8. Liu, W., Wang, J., Kumar, S., Chang, S.F.: Hashing with graphs. In: ICML. (2011)
9. Jain, P., Vijayanarasimhan, S., Grauman, K.: Hashing hyperplane queries to near points with applications to large-scale active learning. In: NIPS. (2010)
10. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. In: ICML. (2000)
11. David, H., O'Connell, M., Yang, S.: Distribution and expected value of the rank of a concomitant of an order statistic. The Annals of Statistics **5**(1) (1977) 216–223
12. Eshghi, K., Rajaram, S.: Locality sensitive hash functions based on concomitant rank order statistics. In: ACM SIGKDD. (2008)
13. Zhao, B., Wang, F., Zhang, C.: Efficient maximum margin clustering via cutting plane algorithm. In: SDM. (2008)
14. Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y.: Robust face recognition via sparse representation. IEEE Trans. Pattern Anal. Mach. Intell. **31** (2009) 210–227
15. Perkins, S., Lacker, K., Theiler, J.: Grafting: fast, incremental feature selection by gradient descent in function space. J. Mach. Learn. Res. **3** (2003) 1333–1356
16. Zhu, J., Lao, N., Xing, E.P.: Grafting-light: fast, incremental feature selection and structure learning of markov random fields. In: SIGKDD. (2010)
17. Bhattacharya, P.: Convergence of sample paths of normalized sums of induced order statistics. The Annals of Statistics **2** (1974) 1034–1039
18. Sen, P.: A note on invariance principles for induced order statistics. Annuals of Probability **4** (1976) 474–479
19. David, H., Galambos, J.: The asymptotic theory of concomitants of order statistics. Journal of Applied Probability **11** (1974) 762–770
20. Charikar, M.: Similarity estimation techniques from rounding algorithms. In: STOC. (2002)
21. Vempala, S.: The Random Projection Method. American Mathematical Society (2004)
22. Yang, J., Yu, K., Huang, T.: Efficient highly over-complete sparse coding using a mixture model. In: ECCV. (2010)
23. Chen, X., Mu, Y., Yan, S., Chua, T.S.: Efficient large-scale image annotation by probabilistic collaborative multi-label propagation. In: ACM Multimedia. (2010)