Improved Asymmetric Locality Sensitive Hashing (ALSH) for Maximum Inner Product Search (MIPS)

Anshumali Shrivastava
Department of Computer Science
Computing and Information Science
Cornell University
Ithaca, NY 14853, USA
anshu@cs.cornell.edu

Ping Li
Department of Statistics and Biostatistics
Department of Computer Science
Rutgers University
Piscataway, NJ 08854, USA
pingli@stat.rutgers.edu

Abstract

Recently we showed that the problem of Maximum Inner Product Search (MIPS) is efficient and it admits provably sub-linear hashing algorithms. In [23], we used asymmetric transformations to convert the problem of approximate MIPS into the problem of approximate near neighbor search which can be efficiently solved using L2-LSH. In this paper, we revisit the problem of MIPS and argue that the quantizations used in L2-LSH is suboptimal for MIPS compared to signed random projections (SRP) which is another popular hashing scheme for cosine similarity (or correlations). Based on this observation, we provide different asymmetric transformations which convert the problem of approximate MIPS into the problem amenable to SRP instead of L2-LSH. An additional advantage of our scheme is that we also obtain LSH type space partitioning which is not possible with the existing scheme. Our theoretical analysis shows that the new scheme is significantly better than the original scheme for MIPS. Experimental evaluations strongly support the theoretical findings. In addition, we also provide the first empirical comparison that shows the superiority of hashing over tree based methods [21] for MIPS.

1 Introduction

In this paper, we revisit the problem of *Maximum Inner Product Search (MIPS)*, which was studied in our recent work [23]. In this work we present the first provably fast algorithm for MIPS, which was considered hard [21, 15]. Given an input query point $q \in \mathbb{R}^D$, the task of MIPS is to find $p \in \mathcal{S}$, where $\mathcal{S}$ is a giant collection of size N, which maximizes (approximately) the **inner product** $q^T p$:

$$p = \arg\max_{x \in \mathcal{S}} \quad q^T x \tag{1}$$

The MIPS problem is related to the problem of *near neighbor search (NNS)*. For example, L2-NNS

$$p = \arg\min_{x \in \mathcal{S}} ||q - x||_2^2 = \arg\min_{x \in \mathcal{S}} (||x||_2^2 - 2q^T x) \tag{2}$$

or, correlation-NNS

$$p = \arg\max_{x \in \mathcal{S}} \frac{q^T x}{\|q\|\|x\|} = \arg\max_{x \in \mathcal{S}} \frac{q^T x}{\|x\|} \tag{3}$$

These three problems are equivalent if the norm of every element $x \in \mathcal{S}$ is constant. Clearly, the value of the norm $||q||_2$ has no effect for the argmax. In many scenarios, MIPS arises naturally at places where the norms of the elements in $\mathcal{S}$ have significant variations [15]. As reviewed in our prior work [23], examples of applications of MIPS include recommender system [16, 5, 15], large-scale object detection with DPM [9, 7, 14, 14], structural SVM [7], and multi-class label prediction [21, 15, 25].

Asymmetric LSH (ALSH): Locality Sensitive Hashing (LSH) [13] is popular in practice for efficiently solving NNS. In our prior work [23], the concept of "asymmetric LSH" (ALSH) was formalized and one can transform the input query $Q(p)$ and data in the collection $P(x)$ independently, where the transformations Q and P are different. In [23] we developed a particular set of transformations to convert MIPS into L2-NNS and then solved the problem by standard hashing i.e. L2-LSH [6]. In this paper, we name the scheme in [23] as **L2-ALSH**. Later we showed in [24] the flexibility and the power of the asymmetric framework developed in [23] by constructing a provably superior scheme for binary data. Prior to our work, asymmetry was applied for hashing higher order similarity [22], sketching [8], hashing different subspaces [3], and data dependent hashing [20] which unlike locality sensitive hashing do not come with provable runtime guarantees. Explicitly constructing asymmetric transformation tailored for a particular similarity, given an existing LSH, was the first observation made in [23] due to which MIPS, a sought after problem, became provably fast and practical.

It was argued in [17] that the quantizations used in traditional L2-LSH is suboptimal and it hurts the variance of the hashes. This raises a natural question that L2-ALSH which uses L2-LSH as a subroutine for solving MIPS could be suboptimal and there may be a better hashing scheme. We provide such a scheme in this work.

Our contribution: Based on the observation that the quantizations used in traditional L2-LSH is suboptimal, in this study, we propose another scheme for ALSH, by developing a new set of asymmetric transformations to convert MIPS into a problem of correlation-NNS, which is solved by "signed random projections" (SRP) [11, 4]. The new scheme thus avoids the use of L2-LSH. We name this new scheme as **Sign-ALSH**. Our theoretical analysis and experimental study show that Sign-ALSH is more advantageous than L2-ALSH for MIPS.

For inner products asymmetry is unavoidable. In case of L2-ALSH, due to asymmetry, we loose the capability to generate LSH like random data partitions for efficient clustering [12]. We show that for inner products with Sign-ALSH there is a novel formulation that allows us to generate such partitions for inner products. With existing L2-ALSH such formulation does not work.

Apart from providing a better hashing scheme, we also provide comparisons of the Sign-ALSH with cone trees [21]. Our empirical evaluations on three real datasets show that hashing based methods are superior over the tree based space partitioning methods. Since there is no existing comparison of hashing based methods with tree based methods for the problem of MIPS, we believe that the results shown in this work will be very valuable for practitioners.

2 Review: Locality Sensitive Hashing (LSH)

The problem of efficiently finding nearest neighbors has been an active research since the very early days of computer science [10]. Approximate versions of the near neighbor search problem [13] were proposed to break the linear query time bottleneck. The following formulation for approximate near neighbor search is often adopted.

Definition: (c-Approximate Near Neighbor or c-NN) *Given a set of points in a D-dimensional space $\mathbb{R}^D$, and parameters $S_0 > 0$, $\delta > 0$, construct a data structure which, given any query point q, does the following with probability $1 - \delta$: if there exists an S_0-near neighbor of q in $\mathcal{S}$, it reports some cS_0-near neighbor of q in $\mathcal{S}$.*

Locality Sensitive Hashing (LSH) [13] is a family of functions, with the property that more similar items have a higher collision probability. LSH trades off query time with extra (one time) preprocessing cost and space. Existence of an LSH family translates into provably sublinear query time algorithm for c-NN problems.

Definition: (Locality Sensitive Hashing (LSH)) *A family $\mathcal{H}$ is called (S_0, cS_0, p_1, p_2)-sensitive if, for any two points $x, y \in \mathbb{R}^D$, h chosen uniformly from $\mathcal{H}$ satisfies:*

- *if $Sim(x, y) \geq S_0$ then $Pr_{\mathcal{H}}(h(x) = h(y)) \geq p_1$*
- *if $Sim(x, y) \leq cS_0$ then $Pr_{\mathcal{H}}(h(x) = h(y)) \leq p_2$*

For efficient approximate nearest neighbor search, $p_1 > p_2$ and $c < 1$ is needed.

Fact 1: Given a family of (S_0, cS_0, p_1, p_2) -sensitive hash functions, one can construct a data structure for c-NN with $O(n^\rho \log n)$ query time and space $O(n^{1+\rho})$, where $\rho = \frac{\log p_1}{\log p_2} < 1$.

LSH is a generic framework and an implementation of LSH requires a concrete hash function.

2.1 LSH for L2 distance

[6] presented an LSH family for L_2 distances. Formally, given a fixed window size r, we sample a random vector a with each component from i.i.d. normal, i.e., $a_i \sim N(0, 1)$, and a scalar b generated uniformly at random from $[0, r]$. The hash function is defined as:

$$h^{L2}_{a,b}(x) = \left\lfloor \frac{a^T x + b}{r} \right\rfloor \tag{4}$$

where $\lfloor\rfloor$ is the floor operation. The collision probability under this scheme can be shown to be

$$Pr(h^{L2}_{a,b}(x) = h^{L2}_{a,b}(y)) \tag{5}$$
$$= 1 - 2\Phi(-r/d) - \frac{2}{\sqrt{2\pi}(r/d)}\left(1 - e^{-(r/d)^2/2}\right) = F_r(d)$$

where $\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$ and $d = ||x - y||_2$ is the Euclidean distance between the vectors x and y.

2.2 LSH for correlation

Another popular LSH family is the so-called "sign random projections" [11, 4]. Again, we choose a random vector a with $a_i \sim N(0, 1)$. The hash function is defined as:

$$h^{Sign}(x) = sign(a^T x) \tag{6}$$

And collision probability is

$$Pr(h^{Sign}(x) = h^{Sign}(y)) = 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{x^T y}{\|x\|\|y\|}\right) \tag{7}$$

This scheme is known as *signed random projections (SRP)*.

3 Review of ALSH for MIPS and L2-ALSH

In [23], it was shown that the framework of locality sensitive hashing is restrictive for solving MIPS. The inherent assumption of the same hash function for both the transformation as well as the query was unnecessary in the classical LSH framework and it was the main hurdle in finding provable sub-linear algorithms for MIPS with LSH. For the theoretical guarantees of LSH to work there was no requirement of symmetry. Incorporating asymmetry in the hashing schemes was the key in solving MIPS efficiently.

Definition [23]: (*Asymmetric* Locality Sensitive Hashing (ALSH)) A family $\mathcal{H}$, along with the two vector functions $Q : \mathbb{R}^D \mapsto \mathbb{R}^{D'}$ (**Query Transformation**) and P :

$\mathbb{R}^D \mapsto \mathbb{R}^{D'}$ (**Preprocessing Transformation**), is called (S_0, cS_0, p_1, p_2)-sensitive if for a given c-NN instance with query q, and the hash function h chosen uniformly from $\mathcal{H}$ satisfies the following:

- if $Sim(q,x) \geq S_0$ then $Pr_{\mathcal{H}}(h(Q(q))) = h(P(x))) \geq p_1$
- if $Sim(q,x) \leq cS_0$ then $Pr_{\mathcal{H}}(h(Q(q)) = h(P(x))) \leq p_2$

Here x is any point in the collection $\mathcal{S}$.

Note that the query transformation Q is only applied on the query and the pre-processing transformation P is applied to $x \in \mathcal{S}$ while creating hash tables. By letting $Q(x) = P(x) = x$, we can recover the vanilla LSH. Using different transformations (i.e., $Q \neq P$), it is possible to counter the fact that self similarity is not highest with inner products which is the main argument of failure of LSH. We just need the probability of the new collision event $\{h(Q(q)) = h(P(y))\}$ to satisfy the conditions of definition of ALSH for $Sim(q,y) = q^T y$.

Theorem 1 *[23] Given a family of hash function $\mathcal{H}$ and the associated query and preprocessing transformations P and Q, which is (S_0, cS_0, p_1, p_2) -sensitive, one can construct a data structure for c-NN with $O(n^\rho \log n)$ query time and space $O(n^{1+\rho})$, where $\rho = \frac{\log p_1}{\log p_2}$.*

[23] provided an explicit construction of ALSH, which we call **L2-ALSH**. Without loss of generality, one can assume

$$||x_i||_2 \leq U < 1, \quad \forall x_i \in \mathcal{S} \tag{8}$$

for some $U < 1$. If this is not the case, then we can always scale down the norms without altering the $\arg\max$. Since the norm of the query does not affect the $\arg\max$ in MIPS, for simplicity it was assumed $||q||_2 = 1$. This condition can be removed easily (see Section 5 for details). In L2-ALSH, two vector transformations $P : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$ and $Q : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$ are defined as follows:

$$P(x) = [x; ||x||_2^2; ||x||_2^4; ....; ||x||_2^{2^m}] \tag{9}$$
$$Q(x) = [x; 1/2; 1/2; ....; 1/2], \tag{10}$$

where [;] is the concatenation. $P(x)$ appends m scalers of the form $||x||_2^{2^i}$ at the end of the vector x, while Q(x) simply appends m "1/2" to the end of the vector x. By observing

$$||P(x_i)||_2^2 = ||x_i||_2^2 + ||x_i||_2^4 + ... + ||x_i||_2^{2^m} + ||x_i||_2^{2^{m+1}}$$
$$||Q(q)||_2^2 = ||q||_2^2 + m/4 = 1 + m/4$$
$$Q(q)^T P(x_i) = q^T x_i + \frac{1}{2}(||x_i||_2^2 + ||x_i||_2^4 + ... + ||x_i||_2^{2^m})$$

one can obtain the following key equality:

$$||Q(q) - P(x_i)||_2^2 = (1 + m/4) - 2q^T x_i + ||x_i||_2^{2^{m+1}} \tag{11}$$

Since $||x_i||_2 \leq U < 1$, we have $||x_i||^{2^{m+1}} \to 0$ at the tower rate (exponential to exponential). Thus, as long as m is not too small (e.g., $m \geq 3$ would suffice), we have

$$\arg\max_{x \in \mathcal{S}} q^T x \simeq \arg\min_{x \in \mathcal{S}} ||Q(q) - P(x)||_2 \tag{12}$$

This scheme is the first connection between solving un-normalized MIPS and approximate near neighbor search. Transformations P and Q, when norms are less than 1, provide correction to the L2 distance $||Q(q) - P(x_i)||_2$ making it rank correlate with the (un-normalized) inner product.

3.1 Intuition for the Better Scheme : Why Signed Random Projections (SRP)?

Recently in [17, 18], it was observed that the quantization of random projections used by traditional L2-LSH scheme is not desirable when the data is normalized and in fact the shift b in Eq. (4) hurts the variance leading to less informative hashes. The sub-optimality of L2-LSH hints towards existence of better hashing functions for MIPS.

As previously argued, when the data are normalized then both L2-NNS and correlation-NNS are equivalent to MIPS. Therefore, for normalized data we can use either L2-LSH which is popular LSH for L2 distance or SRP which is popular LSH for correlation to solve MIPS directly. This raises a natural question "Which will perform better ?".

If we assume that the data are normalized, i.e., all the norms are equal to 1, then both SRP and L2-LSH are monotonic in the inner product and their corresponding ρ values for retrieving max inner product can be computed as

$$\rho_{SRP} = \frac{\log\left(1 - \frac{1}{\pi}\cos^{-1}(S_0)\right)}{\log\left(1 - \frac{1}{\pi}\cos^{-1}(cS_0)\right)} \tag{13}$$

$$\rho_{L2-LSH} = \frac{\log\left(F_r(\sqrt{2-2S_0})\right)}{\log\left(F_r(\sqrt{2-2cS_0})\right)} \tag{14}$$

where the function $F_r(.)$ is given by Eq. (5). The values of ρ_{SRP} and ρ_{L2-LSH} for different $S_0 = \{0.1, 0.2, .., 0.9, 0.95\}$ with respect to approximation ratio c is shown in Figure 1. We use standard recommendation of $r = 2.5$ for L2-LSH. We can clearly see that ρ_{SRP} is consistently better than ρ_{L2-LSH} given any S_0 and c. Thus, for MIPS with normalized data L2-LSH type of quantization given by equation 5 seems suboptimal. It is clear that when the data is normalized then SRP is always a better choice for MIPS as compared to L2-LSH. This motivates us to explore the possibility of better hashing algorithm for general (unnormalized) instance of MIPS using SRP, which will have impact in many applications as pointed out in [23].

Asymmetric transformations give us enough flexibility to modify norms without changing inner products. The transformations provided in [23] used this flexibility to convert MIPS to standard near neighbor search in L_2 space for which we have standard hash functions. For binary data, [24] showed a strictly superior construction, the asymmetric minwise hashing, which outperforms all ALSHs made for general MIPS.

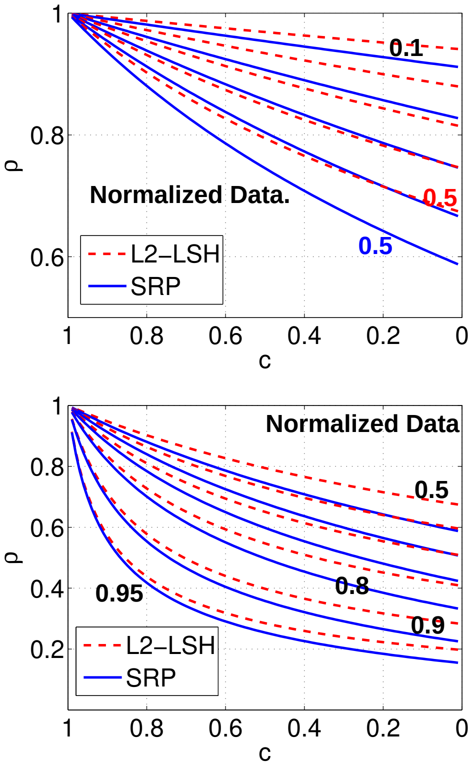


Figure 1: Values of ρ_{SRP} and ρ_{L2-LSH} (Lower is better) for normalized data. It is clear that SRP is more suited for retrieving inner products when the data is normalized

Signed random projections are popular hash functions widely adopted for correlation or cosine similarity. We use asymmetric transformations to convert approximate MIPS into approximate maximum correlation search and thus we avoid the use of sub-optimal L2-LSH. The collision probability of the hash functions is one of the key constituents which determine the efficiency of the obtained ALSH algorithm. We show that our proposed transformation with SRP is better suited for ALSH compared to the existing L2-ALSH for solving general MIPS instance.

4 The New Proposal: Sign-ALSH

4.1 From MIPS to Correlation-NNS

We assume for simplicity that $||q||_2 = 1$ as the norm of the query does not change the ordering, we show in the next section how to get rid of this assumption. Without loss of generality let $||x_i||_2 \le U < 1, \ \ \forall x_i \in \mathcal{S}$ as it can always be achieved by scaling the data by large enough number. We define two vector transformations $P : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$ and $Q : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$ as follows:

$$P(x) = [x; 1/2 - ||x||_2^2; 1/2 - ||x||_2^4; ....; 1/2 - ||x||_2^{2^m}] \tag{15}$$

$$Q(x) = [x; 0; 0; ....; 0], \tag{16}$$

Using $||Q(q)||_2^2 = ||q||_2^2 = 1$, $Q(q)^T P(x_i) = q^T x_i$, and

$$\begin{aligned} &||P(x_i)||_2^2 \\ &= ||x_i||_2^2 + 1/4 + ||x_i||_2^4 - ||x_i||_2^2 + 1/4 + ||x_i||_2^8 - ||x_i||_2^4 + ... \\ &+ 1/4 + ||x_i||_2^{2^{m+1}} - ||x_i||_2^{2^m} \\ &= m/4 + ||x_i||_2^{2^{m+1}} \end{aligned}$$

we obtain the following key equality:

$$\frac{Q(q)^T P(x_i)}{\|Q(q)\|_2 \|P(x_i)\|_2} = \frac{q^T x_i}{\sqrt{m/4 + ||x_i||_2^{2^{m+1}}}} \tag{17}$$

The term $||x_i||^{2^{m+1}} \to 0$, again vanishes at the tower rate. This means we have approximately

$$\arg\max_{x \in \mathcal{S}} q^T x \simeq \arg\max_{x \in \mathcal{S}} \frac{Q(q)^T P(x_i)}{\|Q(q)\|_2 \|P(x_i)\|_2} \tag{18}$$

This provides another solution for solving MIPS using known methods for approximate correlation-NNS. Asymmetric transformations P and Q provide a lot of flexibility. Note that transformations P and Q are not unique for this task and there are other possibilities [2, 19]. It should be further noted that even scaling data and query differently is asymmetry in a strict sense because it changes the distribution of the hashes. Flexibility in choosing the transformations P and Q allow us to use signed random projections and thereby making possible to avoid suboptimal L2-LSH.

4.2 Fast MIPS Using Sign Random Projections

Eq. (18) shows that MIPS reduces to the standard approximate near neighbor search problem which can be efficiently solved by sign random projections, i.e., h^{Sign} (defined by Eq. (6)). Formally, we can state the following theorem.

Theorem 2 *Given a c-approximate instance of MIPS, i.e., $Sim(q, x) = q^T x$, and a query q such that $||q||_2 = 1$ along with a collection $\mathcal{S}$ having $||x||_2 \le U < 1 \ \forall x \in \mathcal{S}$. Let P and Q be the vector transformations defined in Eq. (15) and Eq. (16), respectively. We have the following two conditions for hash function h^{Sign} (defined by Eq. (6))*

- *if $q^T x \ge S_0$ then*

$$\begin{aligned} &Pr[h^{Sign}(Q(q)) = h^{Sign}(P(x))] \\ &\ge 1 - \frac{1}{\pi} \cos^{-1}\left(\frac{S_0}{\sqrt{m/4 + U^{2^{m+1}}}}\right) \end{aligned}$$

- *if $q^T x \le cS_0$ then*

$$\begin{aligned} &Pr[h^{Sign}(Q(q)) = h^{Sign}(P(x))] \\ &\le 1 - \frac{1}{\pi} \cos^{-1}\left(\frac{\min\{cS_0, z^*\}}{\sqrt{m/4 + (\min\{cS_0, z^*\})^{2^{m+1}}}}\right) \end{aligned}$$

where $z^* = \left(\frac{m/2}{2^{m+1}-2}\right)^{2^{-m-1}}$.

Proof: *When $q^T x \ge S_0$, we have, according to Eq. (7)*

$$\begin{aligned}&Pr[h^{Sign}(Q(q)) = h^{Sign}(P(x))]\\ &= 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{q^T x}{\sqrt{m/4 + \|x\|_2^{2^{m+1}}}}\right)\\ &\ge 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{q^T x}{\sqrt{m/4 + U^{2^{m+1}}}}\right)\end{aligned}$$

When $q^T x \le cS_0$, by noting that $q^T x \le \|x\|_2$, we have

$$\begin{aligned}&Pr[h^{Sign}(Q(q)) = h^{Sign}(P(x))]\\ &= 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{q^T x}{\sqrt{m/4 + \|x\|_2^{2^{m+1}}}}\right)\\ &\le 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{q^T x}{\sqrt{m/4 + (q^T x)^{2^{m+1}}}}\right)\end{aligned}$$

For this one-dimensional function $f(z) = \frac{z}{\sqrt{a+z^b}}$, where $z = q^T x$, $a = m/4$ and $b = 2^{m+1} \ge 2$, we know

$$f'(z) = \frac{a - z^b\left(b/2 - 1\right)}{(a + z^b)^{3/2}}$$

One can also check that $f''(z) \le 0$ for $0 < z < 1$, i.e., $f(z)$ is a concave function. The maximum of $f(z)$ is attained at $z^* = \left(\frac{2a}{b-2}\right)^{1/b} = \left(\frac{m/2}{2^{m+1}-2}\right)^{2^{-m-1}}$ If $z^* \ge cS_0$, then we need to use $f(cS_0)$ as the bound. $\square$

Therefore, we have obtained, in LSH terminology,

$$p_1 = 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{S_0}{\sqrt{m/4 + U^{2^{m+1}}}}\right) \tag{19}$$

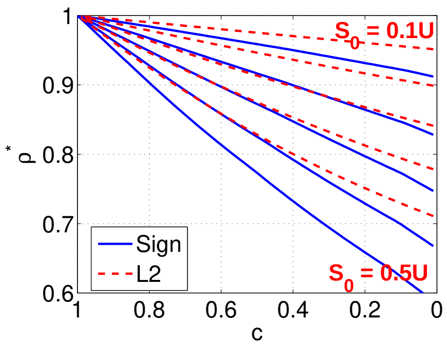
$$p_2 = 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{\min\{cS_0, z^*\}}{\sqrt{m/4 + \left(\min\{cS_0, z^*\}\right)^{2^{m+1}}}}\right), \tag{20}$$

$$z^* = \left(\frac{m/2}{2^{m+1} - 2}\right)^{2^{-m-1}} \tag{21}$$

Theorem 1 allows us to construct data structures with worst case $O(n^\rho \log n)$ query time guarantees for c-approximate MIPS, where $\rho = \frac{\log p_1}{\log p_2}$. For any given $c < 1$, there always exist $U < 1$ and m such that $\rho < 1$. This way, we obtain a sublinear query time algorithm for MIPS. Because ρ is a function of 2 parameters, the best query time chooses U and m, which minimizes the value of ρ. For convenience, we define

$$\rho^* = \min_{U,m} \frac{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{S_0}{\sqrt{m/4+U^{2^{m+1}}}}\right)\right)}{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{\min\{cS_0,z^*\}}{\sqrt{m/4+(\min\{cS_0,z^*\})^{2^{m+1}}}}\right)\right)} \tag{22}$$

See Figure 2 for the plots of ρ^*, which also compares the optimal ρ values for L2-ALSH in the prior work [23]. The results show that Sign-ALSH is noticeably better.



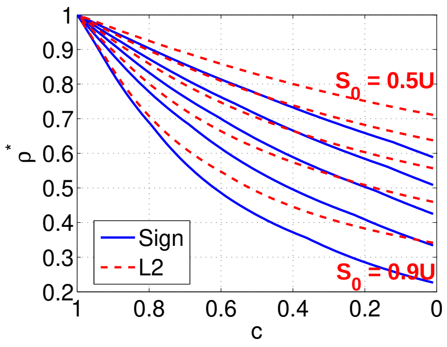


Figure 2: Optimal values of ρ^* (lower is better) with respect to approximation ratio c for different S_0, obtained by a grid search over parameters U and m, given S_0 and c. The curves show that Sign-ALSH (solid curves) is noticeably better than L2-ALSH (dashed curves) in terms of their optimal ρ^* values. The results for L2-ALSH were from the prior work [23]. For clarity, the results are in two figures.

4.3 Parameter Selection

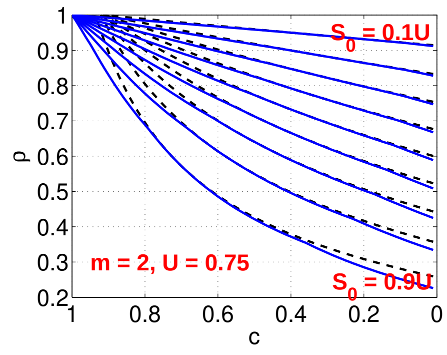


Figure 3: The solid curves are the optimal ρ values of Sign-ALSH from Figure 2. The dashed curves represent the ρ values for fixed parameters: $m = 2$ and $U = 0.75$ (left panel). Even with fixed parameters, the ρ does not degrade.

Figure 3 presents the ρ values for $(m,\ U) = (2,\ 0.75)$ We can see that even if we use fixed parameters, the performance would only degrade little. This essentially frees practitioners from the burden of choosing parameters.

5 Removing Dependency on Norm of Query

Changing norms of the query does not affect the $\arg\max_{x\in\mathcal{C}} q^T x$, and hence, in practice for retrieving top-k, normalizing the query should not affect the performance. But for theoretical purposes, we want the runtime guarantee to be independent of $||q||_2$. Note, both LSH and ALSH schemes solve the c-approximate instance of the problem, which requires a threshold $S_0 = q^T x$ and an approximation ratio c. These quantities change if we change the norms. We can use the same idea used in [23] to get rid of the norm of q. Transformations P and Q were precisely meant to remove the dependency of correlation on the norms of x but at the same time keeping the inner products same. Let M be the upper bound on all the norms i.e. $M = max_{x\in\mathcal{C}}||x||_2$. In other words M is the radius of the space.

Let $U < 1$, define the transformations, $T : \mathbb{R}^D \to \mathbb{R}^D$ as

$$T(x) = \frac{Ux}{M} \tag{23}$$

and transformations $P, Q : \mathbb{R}^D \to \mathbb{R}^{D+m}$ are the same for the Sign-ALSH scheme as defined in Eq (15) and (16).

Given the query q and any data point x, observe that the inner products between $P(Q(T(q)))$ and $Q(P(T(x)))$ is

$$P(Q(T(q)))^T Q(P(T(x))) = q^T x \times \left(\frac{U^2}{M^2}\right) \tag{24}$$

$P(Q(T(q)))$ appends first m zeros components to $T(q)$ and then m components of the form $1/2 - ||q||^{2^i}$. $Q(P(T(q)))$ does the same thing but in a different order. Now we are working in $D + 2m$ dimensions. It is not difficult to see that the norms of $P(Q(T(q)))$ and $Q(P(T(q)))$ is given by

$$||P(Q(T(q)))||_2 = \sqrt{\frac{m}{4} + ||T(q)||_2^{2^{m+1}}} \tag{25}$$

$$||Q(P(T(x)))||_2 = \sqrt{\frac{m}{4} + ||T(x)||_2^{2^{m+1}}} \tag{26}$$

The transformations are very asymmetric but we know that it is necessary.

Therefore the correlation or the cosine similarity between $P(Q(T(q)))$ and $Q(P(T(x)))$ is

$$Corr = \frac{q^T x \times \left(\frac{U^2}{M^2}\right)}{\sqrt{\frac{m}{4} + ||T(q)||_2^{2^{m+1}}}\sqrt{\frac{m}{4} + ||T(x)||_2^{2^{m+1}}}} \tag{27}$$

Note $||T(q)||_2^{2^{m+1}}, ||T(x)||_2^{2^{m+1}} \leq U < 1$, therefore both $||T(q)||_2^{2^{m+1}}$ and $||T(x)||_2^{2^{m+1}}$ converge to zero at a tower rate and we get approximate monotonicity of correlation with the inner products. We can apply sign random projections to hash $P(Q(T(q)))$ and $Q(P(T(q)))$.

As $0 \leq ||T(q)||_2^{2^{m+1}} \leq U$ and $0 \leq ||T(x)||_2^{2^{m+1}} \leq U$, it is not difficult to get p_1 and p_2 for Sign-ALSH, without conditions on any norms. Simplifying the expression, we get the following value of optimal ρ_u (u for unrestricted).

$$\rho_u^* = \min_{U,m,} \frac{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{S_0\times\left(\frac{U^2}{M^2}\right)}{\frac{m}{4}+U^{2^{m+1}}}\right)\right)}{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{cS_0\times\left(\frac{4U^2}{M^2}\right)}{m}\right)\right)} \tag{28}$$

$$s.t.\quad U^{2^{m+1}} < \frac{m(1-c)}{4c},\ m \in \mathbb{N}^+, \text{ and } 0 < U < 1.$$

With this value of ρ_u^*, we can state our main theorem.

Theorem 3 *For the problem of c-approximate MIPS in a bounded space, one can construct a data structure having $O(n^{\rho_u^*}\log n)$ query time and space $O(n^{1+\rho_u^*})$, where $\rho_u^* < 1$ is the solution to constraint optimization (28).*

Note, for all $c < 1$, we always have $\rho_u^* < 1$ because the constraint $U^{2^{m+1}} < \frac{m(1-c)}{4c}$ is always true for big enough m. The only assumption for efficiently solving MIPS that we need is that the space is bounded, which is always satisfied for any finite dataset. ρ_u^* depends on M, the radius of the space, which is expected.

6 Random Space Partitioning for Inner Product

In this section, we show that due to the nature of the new transformations P and Q there is one subtle but surprising advantage of Sign-ALSH over L2-ALSH.

One popular application of LSH (Locality Sensitive Hashing) is random partitioning of the data for large scale clustering, where similar points map to the same partition (or bucket). Such partitions are very useful in many applications [12]. With classical LSH, we simply use $h(x)$ to generate partition for x. Since $Pr_{\mathcal{H}}(h(x) = h(y))$ is high if $sim(x, y)$ is high, similar points are likely to go into the same partition under the usual LSH mapping. For general ALSH, this property is lost because of asymmetry.

In case of ALSH, we only know that $Pr(h(P(x)) = h(Q(y))$ is high if $sim(x, y)$ is high. Therefore, given x we cannot determine whether to assign partition using $h(P(.))$ or $h(Q(.))$. Neither $Pr(h(P(x)) = h(P(y))$ nor $Pr_{\mathcal{H}}(h(Q(x)) = h(Q(y))$ strictly indicates high value of $sim(x, y)$ in general. Therefore, partitioning property of classical LSH does not hold anymore with general ALSHs. However for the case of inner products using Sign-ALSH, there is a subtle observation which allows us to construct the required assignment function, where pairs of points with high inner products are more likely to get mapped in the same partition while pairs with low inner products are more likely to map into different partitions.

In case of Sign-ALSH for MIPS, we have the transformations $P(Q(T(x)))$ and $Q(P(T(x)))$ given by

$$P(Q(T(x))) = [x; 1/2 - ||T(x)||_2^2; ....; 1/2 - ||T(x)||_2^{2^m}, 0, ..., 0]$$

$$Q(P(T(x))) = [x; 0, ..., 0, 1/2 - ||T(x)||_2^2; ....; 1/2 - ||T(x)||_2^{2^m}].$$

After this transformation, we multiply the generated $D + 2m$ dimensional vector by a random vector $a \in \mathbb{R}^{D+2m}$ whose entries are i.i.d. Gaussian followed by taking the sign. For illustration let $a = [w; s_1, ...s_m, t_1, ...t_m]$ where $w \in \mathbb{R}^D$ b_i and c_i are numbers. All components of a are i.i.d. from $N(0,1)$. With this notation, we can write the final Sign-ALSH as

$$h^{Sign}(P(Q(T(x)))) = Sign(w^T T(x) + \sum_{i=1}^{m} s_i(1/2 - ||T(x)||_2^{2^i}))$$

$$h^{Sign}(Q(P(T(x)))) = Sign(w^T T(x) + \sum_{i=1}^{m} t_i(1/2 - ||T(x)||_2^{2^i}))$$

The key observation here is that $h^{Sign}(P(Q(T(x))))$ does not depend on t_i and $h^{Sign}(Q(P(T(x))))$ does not depend on s_i. If we define

$$h_w(x) = Sign(w^T T(x) + \sum_{i=1}^{m} \alpha_i(1/2 - ||T(x)||_2^{2^i})) \quad (29)$$

where α_i are sampled i.i.d. from $N(0,1)$ for every x independently of everything else. Then, **under the randomization of** w, it is not difficult to show that

$$Pr_w(h_w(x) = h_w(y)) = Pr(h^{Sign}(P(x)) = h^{Sign}(Q(y)))$$

for any x, y. The term $Pr(h^{Sign}(P(x)) = h^{Sign}(Q(y)))$ satisfies the LSH like property and therefore, in any partitions using h_w, points with high inner products are more likely to be together. Thus, $h_w(x)$ is the required assignment. Note, h_w is not technically an LSH because we are randomly sampling α_i for all x independently. The construction of h_w using independent randomizations could be of separate interest. To the best of our knowledge, this is the first example of LSH like partition using hash function with independent randomization for every data point.

The function h_w is little subtle here, we sample w i.i.d from Gaussian and use the same w for all x, but while computing h_w we use α_i independent of everything for every x. The probability is under the randomization of w and independence of all α_i ensures the asymmetry. We are not sure if such construction is possible with L2-ALSH. For LSH partitions with binary data, the idea used here can be applied on asymmetric minwise hashing [24].

7 Ranking Evaluations

In [23], the L2-ALSH scheme was shown to outperform other reasonable heuristics in retrieving maximum inner products. Since our proposal is an improvement over L2-ALSH, in this section we first present comparisons with L2-ALSH, in particular on ranking experiments.

7.1 Datasets

We use three publicly available dataset MNIST, WEBSPAM and RCV1 for evaluations. For each of the three dataset we generate two independent partitions, the query set and the train set. Each element in the query set is used for querying, while the training set serves as the collection $\mathcal{C}$ that will be searched for MIPS. The statistics of the dataset and the partitions are summarized in Table 1

Dataset	Dimension	Query size	Train size
MNIST	784	10,000	60,000
WEBSPAM	16,609,143	5,000	100,000
RCV1	47,236	5,000	100,000

Table 1: Datasets used for evaluations.

7.2 Evaluations

In this section, we show how the ranking of the two ALSH schemes, L2-ALSH and Sign-ALSH, correlates with inner products. Given a query vector q, we compute the top-10 gold standard elements based on the actual inner products $q^T x$, $\forall x \in \mathcal{C}$, here our collection is the train set. We then generate K different hash codes of the query q and all the elements $x \in \mathcal{C}$ and then compute

$$Matches_x = \sum_{t=1}^{K} \mathbf{1}(h_t(Q(q)) = h_t(P(x))), \quad (30)$$

where $\mathbf{1}$ is the indicator function and the subscript t is used to distinguish independent draws of h. Based on $Matches_x$ we rank all the elements x. Ideally, for a better hashing scheme, $Matches_x$ should be higher for element x having higher inner products with the given query q. This procedure generates a sorted list of all the items for a given query vector q corresponding to the each of the two asymmetric hash functions under consideration.

For L2-ALSH, we used the same parameters used and recommended in [23]. For Sign-ALSH, we used the recommended choice shown in Section 4.3, which is $U = 0.75$, $m = 2$. Note that Sign-ALSH does not have parameter r.

We compute precision and recall of the top-10 gold standard elements, obtained from the sorted list based on $Matches_x$. To compute this precision and recall, we start at the top of the ranked item list and walk down in order. Suppose we are at the k^{th} ranked item, we check if this element belongs to the gold standard top-10 list. If it is one of the top-10 gold standard elements, then we increment the count of *relevant seen* by 1, else we move to $k+1$. By k^{th} step, we have already seen k elements, so the *total items seen* is k. The precision and recall at that point are

$$Precision = \frac{\text{relevant seen}}{k}, \qquad Recall = \frac{\text{relevant seen}}{10}$$

We show performance for $K \in \{64, 128, 256, 512\}$. Note that it is important to balance both precision and recall. The

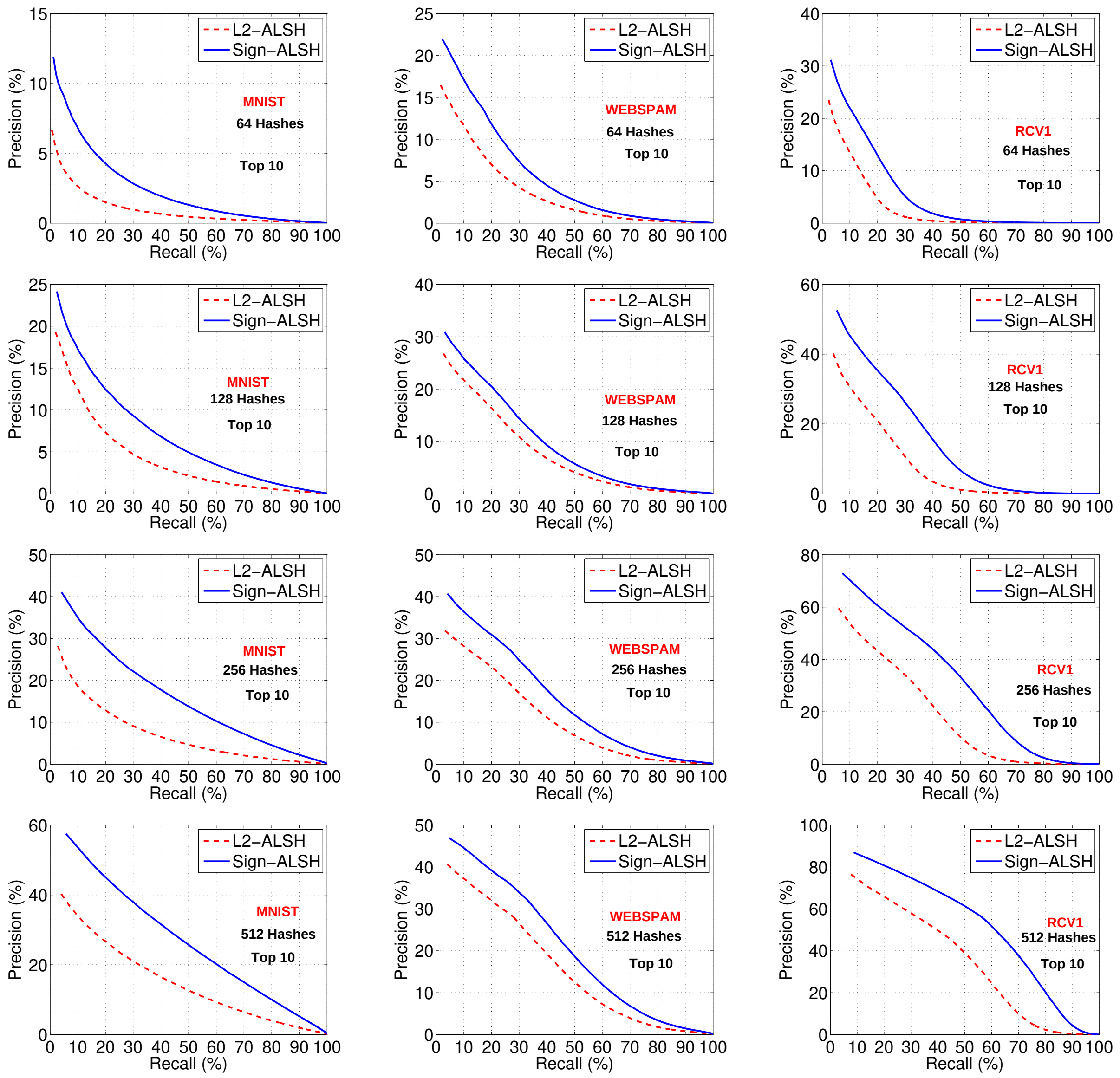


Figure 4: Precision-Recall curves (higher is better). We compare L2-ALSH (using parameters recommended in [23]) with our proposed Sign-ALSH using $(m = 2, U = 0.75)$ for retrieving top-10 elements. Sign-ALSH is noticeably better.

method which obtains higher precision at a given recall is superior. Higher precision indicates higher ranking of the top-10 inner products which is desirable. We report averaged precisions and recalls.

The plots for all the three datasets are shown in Figure 4. We can clearly see, that our proposed Sign-ALSH scheme gives significantly higher precision recall curves than the L2-ALSH scheme, indicating better correlation of top inner products with Sign-ALSH compared to L2-ALSH. The results are consistent across datasets.

8 Comparisons of Hashing Based and Tree Based Methods for MIPS

We have shown in the previous Section that Sign-ALSH outperforms L2-ALSH in ranking evaluations. In this Section, we consider the actual task of finding the maximum inner product. Our aim is to estimate the computational saving, in finding the maximum inner product, with Sign-ALSH compared to the existing scheme L2-ALSH. In addition to L2-ALSH which is a hashing scheme, there is an another tree based space partitioning method [21] for solving MIPS. Although, in theory, it is know that tree based methods perform poorly [25] due to their exponential dependence on the dimensionality, it is still important to understand the impact of such dependency in practice. Unfortunately no empirical comparison between hashing and tree based methods exists for the problem of MIPS in the literature. To provide such a comparison, we also consider tree based space partitioning method [21] for evaluations. We use the same three datasets as described in Section 7.1.

Tree based and hashing based methodologies are very different in nature. The major difference is in the stopping criteria. Hashing based methods create buckets and stop the search once they find a good enough point, they may not succeed with some probability. On the other hand, tree based methods use branch and bound criteria to stop exploring further. So it is possible that a tree based algorithm finds the optimal point but continues to explore further requiring more computations. The usual stopping criteria thus makes tree based methods unnecessarily expensive compared to hashing based methods where the criteria is to stop after finding a good point. Therefore, to ensure fair comparisons, we allow the tree based method to stop the evaluations immediately once the algorithm finds the maximum inner product and prevent it from exploring further. Also, in case when hashing based algorithm fails to find the best inner product we resort to the full linear scan and penalize the hashing based algorithm for not succeeding. All this is required to ensure that tree based algorithm is not at any disadvantage compare to hashing methods.

We implemented the bucketing scheme with Sign-ALSH and L2-ALSH. The bucketing scheme requires creating many hash tables during the preprocessing stage. During query phase, given a query, we compute many hashes of the query and probe appropriate buckets in each table. Please refer [1] for more details on the process. We use the same fixed parameters for all the evaluations, i.e., (m=2, U=0.75) for Sign-ALSH and (m=3, U=0.83, r=2.5) for L2-ALSH as recommended in [23]. The total number of inner products evaluated by a hashing scheme, for a given query, is the total number of hash computation for the query plus the total number of points retrieved from the hash tables. In rare cases, with very small probability, if the hash tables are unable to retrieve the gold standard maximum inner product, we resort to linear scan and also include the total number of inner products computed during the linear scan. We stop as soon as we reach the gold standard point.

We implemented Algorithm 5 from [21], which is the best performing algorithm as shown in the evaluations. For this algorithm, we need to select one parameter which is the minimum number of elements in the node required for splitting. We found that on all the three datasets the value of 100 for this parameter works the best among $\{500, 200, 100, 50\}$. Therefore, we use 100 in all our experiments. The total number of inner products evaluated by tree based algorithm is the total number of points reported plus the total number of nodes visited, where we compute the branch and bound constraint. Again we stop the search process as soon as we reach the point with gold standard maximum inner product. As argued, we need this common stopping condition to compare with hashing based methods, where we do not have any other stopping criteria [13].

For every query we compute the number of inner products evaluated by different methods for MIPS. We report the mean of the total number of inner products evaluated per query in Table 2. We can clearly see that hashing based methods are always better than the tree based algorithm. Except on MNIST dataset, hashing based methods are significantly superior, which is also not surprising because MNIST is an image dataset having low intrinsic dimensionality. Among the two hashing schemes Sign-ALSH is always better than L2-ALSH, which verifies our theoretical findings and supports our arguments in favor of Sign-ALSH over L2-ALSH for MIPS.

	Sign-ALSH	L2-ALSH	Cone Trees
MNIST	**7,944**	9,971	11,202
WEBSPAM	**2,866**	3,813	22,467
RCV1	**9,951**	11,883	38,162

Table 2: Average number of inner products evaluated per query by different MIPS algorithms. Both Sign-ALSH and L2-ALSH [23] outperform cone trees [21]. Sign-ALSH is always superior compared to L2-ALSH for MIPS.

9 Conclusion

The MIPS (maximum inner product search) problem has numerous important applications in machine learning, databases, and information retrieval. [23] developed the framework of Asymmetric LSH and provided an explicit scheme (L2-ALSH) for approximate MIPS in sublinear time. L2-ALSH uses L2-LSH as a subroutine which uses suboptimal quantizations. In this study, we present another asymmetric transformation scheme (Sign-ALSH) which converts the problem of maximum inner products into the problem of maximum correlation search, which is subsequently solved by sign random projections, thereby avoiding the use of L2-LSH.

Theoretical analysis and experimental study demonstrate that **Sign-ALSH** can be noticeably more advantageous than **L2-ALSH**. The new transformations with Sign-ALSH can be adapted to generate LSH like random data partitions which is very useful for large scale clustering. Such an adaptation is not possible with existing L2-ALSH. This was a rather unexpected advantage of the proposed Sign-ALSH over L2-ALSH. We also establish by experiments that hashing based algorithms are superior to tree based space partitioning methods for MIPS.

It should be noted that for MIPS over binary data our recent work asymmetric minwise hashing [24] should be used. We showed that for binary domain asymmetric minwise hashing is both empirically and provably superior, please see [24] for more details.

10 Acknowledgement

The work is partially supported by NSF-III-1360971, NSF-Bigdata-1419210, ONR-N00014-13-1-0764, and AFOSR-FA9550-13-1-0137. We would like to thank the reviewers of AISTATS 2015 and UAI 2015. We also thank Sanjiv Kumar and Hadi Daneshmand for pleasant discussions.

References

[1] A. Andoni and P. Indyk. E2lsh: Exact euclidean locality sensitive hashing. Technical report, 2004.

[2] Y. Bachrach, Y. Finkelstein, R. Gilad-Bachrach, L. Katzir, N. Koenigstein, N. Nice, and U. Paquet. Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, 2014.

[3] R. Basri, T. Hassner, and L. Zelnik-Manor. Approximate nearest subspace search with applications to pattern recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[4] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, Montreal, Quebec, Canada, 2002.

[5] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46. ACM, 2010.

[6] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokn. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG*, pages 253 – 262, Brooklyn, NY, 2004.

[7] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1814–1821. IEEE, 2013.

[8] W. Dong, M. Charikar, and K. Li. Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 123–130. ACM, 2008.

[9] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.

[10] J. H. Friedman and J. W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, 23(9):881–890, 1974.

[11] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of ACM*, 42(6):1115–1145, 1995.

[12] T. H. Haveliwala, A. Gionis, and P. Indyk. Scalable techniques for clustering the web. In *WebDB*, pages 129–134, 2000.

[13] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*, pages 604–613, Dallas, TX, 1998.

[14] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009.

[15] N. Koenigstein, P. Ram, and Y. Shavitt. Efficient retrieval of recommendations in a matrix factorization framework. In *CIKM*, pages 535–544, 2012.

[16] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems.

[17] P. Li, M. Mitzenmacher, and A. Shrivastava. Coding for random projections. In *ICML*, 2014.

[18] P. Li, M. Mitzenmacher, and A. Shrivastava. Coding for random projections and approximate near neighbor search. Technical report, arXiv:1403.8144, 2014.

[19] B. Neyshabur and N. Srebro. On symmetric and asymmetric lshs for inner product search. Technical report, arXiv:1410.5518, 2014.

[20] B. Neyshabur, N. Srebro, R. R. Salakhutdinov, Y. Makarychev, and P. Yadollahpour. The power of asymmetry in binary hashing. In *Advances in Neural Information Processing Systems*, pages 2823–2831, 2013.

[21] P. Ram and A. G. Gray. Maximum inner-product search using cone trees. In *KDD*, pages 931–939, 2012.

[22] A. Shrivastava and P. Li. Beyond pairwise: Provably fast algorithms for approximate k-way similarity search. In *NIPS*, Lake Tahoe, NV, 2013.

[23] A. Shrivastava and P. Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *NIPS*, Montreal, CA, 2014.

[24] A. Shrivastava and P. Li. Asymmetric minwise hashing for indexing binary inner products and set containment. In *WWW*, 2015.

[25] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, VLDB '98, pages 194–205, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.