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 it was shown that the problem of Maximum Inner Product Search (MIPS) is efficient and it admits provably sub-linear hashing algorithms. Asymmetric transformations before hashing were the key in solving MIPS which was otherwise hard. In [18], the authors use asymmetric transformations which convert the problem of approximate MIPS into the problem of approximate near neighbor search which can be efficiently solved using hashing. In this work, we provide a different transformation which converts the problem of approximate MIPS into the problem of approximate cosine similarity search which can be efficiently solved using signed random projections. Theoretical analysis show that the new scheme is significantly better than the original scheme for MIPS. Experimental evaluations strongly support the theoretical findings.

1 Introduction

In this paper, we revisit the problem of *Maximum Inner Product Search (MIPS)*, which was studied in a recent technical report [18]. In this report the authors present the first provably fast algorithm for MIPS, which was considered hard [16, 11]. 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 [11]. As reviewed in [18], examples of applications of MIPS include recommender system [12, 2, 11], large-scale object detection with DPM [6, 4, 10, 10], structural SVM [4], and multi-class label prediction [16, 11, 19].

Asymmetric LSH (ALSH): Locality Sensitive Hashing (LSH) [9] is popular in practice for efficiently solving NNS. In the prior work [18], the concept of "asymmetric LSH" (ALSH) was proposed that one can transform the input query $Q(p)$ and data in the collection $P(x)$ independently, where the transformations Q and P are different. [18] developed a particular set of transformations to convert MIPS into L2-NNS and then solved the problem by standard L2-hash [3]. In this paper, we name the scheme in [18] as **L2-ALSH**. Asymmetry in hashing has become popular recently, and it has been applied for hashing higher order similarity [17], data dependent hashing [15], sketching [5] etc.

Our contribution: 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 "sign random projections" [8, 1]. We name this new scheme as **Sign-ALSH**. Our theoretical analysis and experimental study show that Sign-LSH is more advantageous than L2-ALSH for MIPS.

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 [7]. Approximate versions of the near neighbor search problem [9] 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) [9] 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

[3] 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:

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

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

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

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" [8, 1]. Again, we choose a random vector a with $a_i \sim N(0, 1)$. The hash function is defined as:

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

And collision probability is

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

This hashing scheme is also popularly known as *signed random projections (SRP)*

3 Review of ALSH for MIPS and L2-ALSH

In [18], 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 [18]: (*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 only 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 *[18] 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 $\rho = \frac{\log p_1}{\log p_2}$.*

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

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

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:

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

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:

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

Since $||x_i||_2 \le U < 1$, 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., $m \ge 3$ would suffice), we have

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

This scheme is the first connection between solving unnormalized 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. The general idea of ALSH was partially inspired by the work on three-way similarity search [17], where they applied different hashing functions for handling query and data in the repository.

3.1 Intuition for the Better Scheme

Asymmetric transformations give us enough flexibility to modify norms without changing inner products. The transformation provided in [18] used this flexibility to convert MIPS to standard near neighbor search in L_2 space for which we have standard hash functions. Signed random projections are popular hash functions widely adopted for correlation or cosine similarity. We use asymmetric transformation to convert approximate MIPS into approximate maximum correlation search. The transformations and the collision probability of the hashing functions determines the efficiency of the obtained ALSH algorithm. We show that the new transformation with SRP is better suited for ALSH compared to the existing L2-ALSH. Note that in the recent work on *coding for random projections* [13, 14], it was already shown that sign random projections (or 2-bit random projections) can outperform L2LSH.

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.
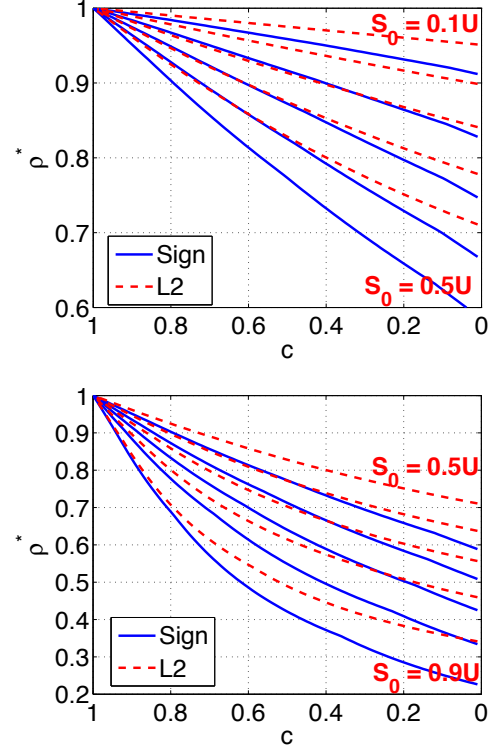

Figure 1: Optimal values of ρ^* (lower is better) with respect to approximation ratio c for different S_0, obtained by a grid search over parameters U and m, given S_0 and c. The curves show that Sign-ALSH (solid curves) is noticeably better than L2-ALSH (dashed curves) in terms of their optimal ρ^* values. The results for L2-ALSH were from the prior work [18]. For clarity, the results are in two figures.

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:

$$P(x) = [x; 1/2 - ||x||_2^2; 1/2 - ||x||_2^4; ....; 1/2 - ||x||_2^{2^m}] \quad (13)$$

$$Q(x) = [x; 0; 0; ....; 0], \quad (14)$$

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

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

we obtain the following key equality:

$$\frac{Q(q)^T P(x_i)}{||Q(q)||_2 ||P(x_i)||_2} = \frac{q^T x_i}{\sqrt{m/4 + ||x_i||_2^{2^{m+1}}}} \quad (15)$$

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

$$\arg\max_{x \in \mathcal{S}} q^T x \simeq \arg\max_{x \in \mathcal{S}} \frac{Q(q)^T P(x_i)}{||Q(q)||_2 ||P(x_i)||_2} \quad (16)$$
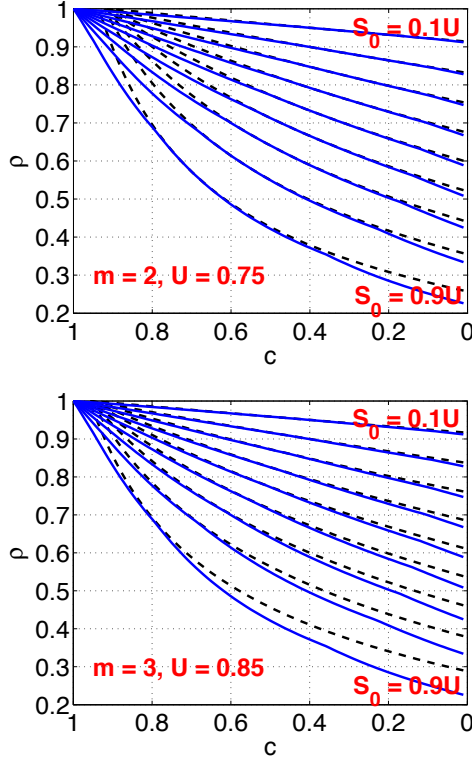
Figure 2: The solid curves are the optimal ρ values of Sign-ALSH from Figure 1. The dashed curves represent the ρ values for fixed parameters: $m = 2$ and $U = 0.75$ (left panel), and $m = 3$ and $U = 0.85$ (right panel). Even with fixed parameters, the ρ does not degrade much.

This provides another solution for solving MIPS using known methods for approximate correlation-NNS.

4.2 Fast Algorithms for MIPS Using Sign Random Projections

Eq. (16) 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 $||x||_2 \leq U < 1 \ \forall x \in \mathcal{S}$. Let P and Q be the vector transformations defined in Eq. (13) and Eq. (14), respectively. We have the following two conditions for hash function h^{Sign} (defined by Eq. (6))*

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

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

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

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

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

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

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

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

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

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

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

One can also check that $f''(z) \leq 0$ 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 $z^* \geq cS_0$, then we need to use $f(cS_0)$ as the bound. $\square$

Therefore, we have obtained, in LSH terminology,

$$p_1 = 1 - \frac{1}{\pi} \cos^{-1} \left(\frac{S_0}{\sqrt{m/4 + U^{2^{m+1}}}} \right) \quad (17)$$

$$p_2 = 1 - \frac{1}{\pi} \cos^{-1} \left(\frac{\min\{cS_0, z^*\}}{\sqrt{m/4 + (\min\{cS_0, z^*\})^{2^{m+1}}}} \right), \quad (18)$$

$$z^* = \left(\frac{m/2}{2^{m+1} - 2} \right)^{2^{-m-1}} \quad (19)$$

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

$$\rho^* = \min_{U,m} \frac{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{S_0}{\sqrt{m/4+U^{2^{m+1}}}}\right)\right)}{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{\min\{cS_0,z^*\}}{\sqrt{m/4+(\min\{cS_0,z^*\})^{2^{m+1}}}}\right)\right)} \tag{20}$$

See Figure 1 for the plots of ρ^*, which also compares the optimal ρ values for L2-ALSH in the prior work [18]. The results show that Sign-ALSH is noticeably better.

4.3 Parameter Selection

Figure 2 presents the ρ values for two sets of selected parameters: $(m,U) = (2,0.75)$ and $(m,U) = (3,0.85)$. We can see that even if we use fixed parameters, the performance would not degrade much. This essentially frees practitioners from the burden of choosing parameters.

5 Remove Dependence 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. For this given c-approximate instance we choose optimal parameters K and L. If the queries have varying norms, which is likely the case in practical scenarios, then given a c-approximate MIPS instance, normalizing the query will change the problem because it will change the threshold S_0 and also the approximation ratio c. The optimal parameters for the algorithm K and L, which are also the size of the data structure, change with S_0 and c. This will require re-doing the costly preprocessing with every change in query. Thus, the query time which is dependent on ρ should be independent of the query.

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. Realizing the fact that we are allowed asymmetry, we can use the same idea to get rid of the norm of q. 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

$$T(x) = \frac{Ux}{M} \tag{21}$$

and transformations $P, Q : \mathbb{R}^D \to \mathbb{R}^{D+m}$ are the same for the Sign-ALSH scheme as defined in Eq (13) and (14).

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

$$P(Q(T(q)))^T Q(P(T(x))) = q^T x \times \left(\frac{U^2}{M^2}\right) \tag{22}$$

$P(Q(T(q)))$ 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

$$||P(Q(T(q)))||_2 = \sqrt{\frac{m}{4} + ||T(q)||_2^{2^{m+1}}} \tag{23}$$

$$||Q(P(T(x)))||_2 = \sqrt{\frac{m}{4} + ||T(x)||_2^{2^{m+1}}} \tag{24}$$

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

$$Corr = \frac{q^T x \times \left(\frac{U^2}{M^2}\right)}{\sqrt{\frac{m}{4} + ||T(q)||_2^{2^{m+1}}}\sqrt{\frac{m}{4} + ||T(x)||_2^{2^{m+1}}}} \tag{25}$$

Note $||T(q)||_2^{2^{m+1}}, ||T(x)||_2^{2^{m+1}} \le U < 1$, 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 $Q(P(T(q)))$.

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

$$\rho_u^* = \min_{U,m,} \frac{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{S_0\times\left(\frac{U^2}{M^2}\right)}{\frac{m}{4}+U^{2^{m+1}}}\right)\right)}{\log\left(1 - \frac{1}{\pi}\cos^{-1}\left(\frac{cS_0\times\left(\frac{4U^2}{M^2}\right)}{m}\right)\right)} \tag{26}$$

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

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 (26).*

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.
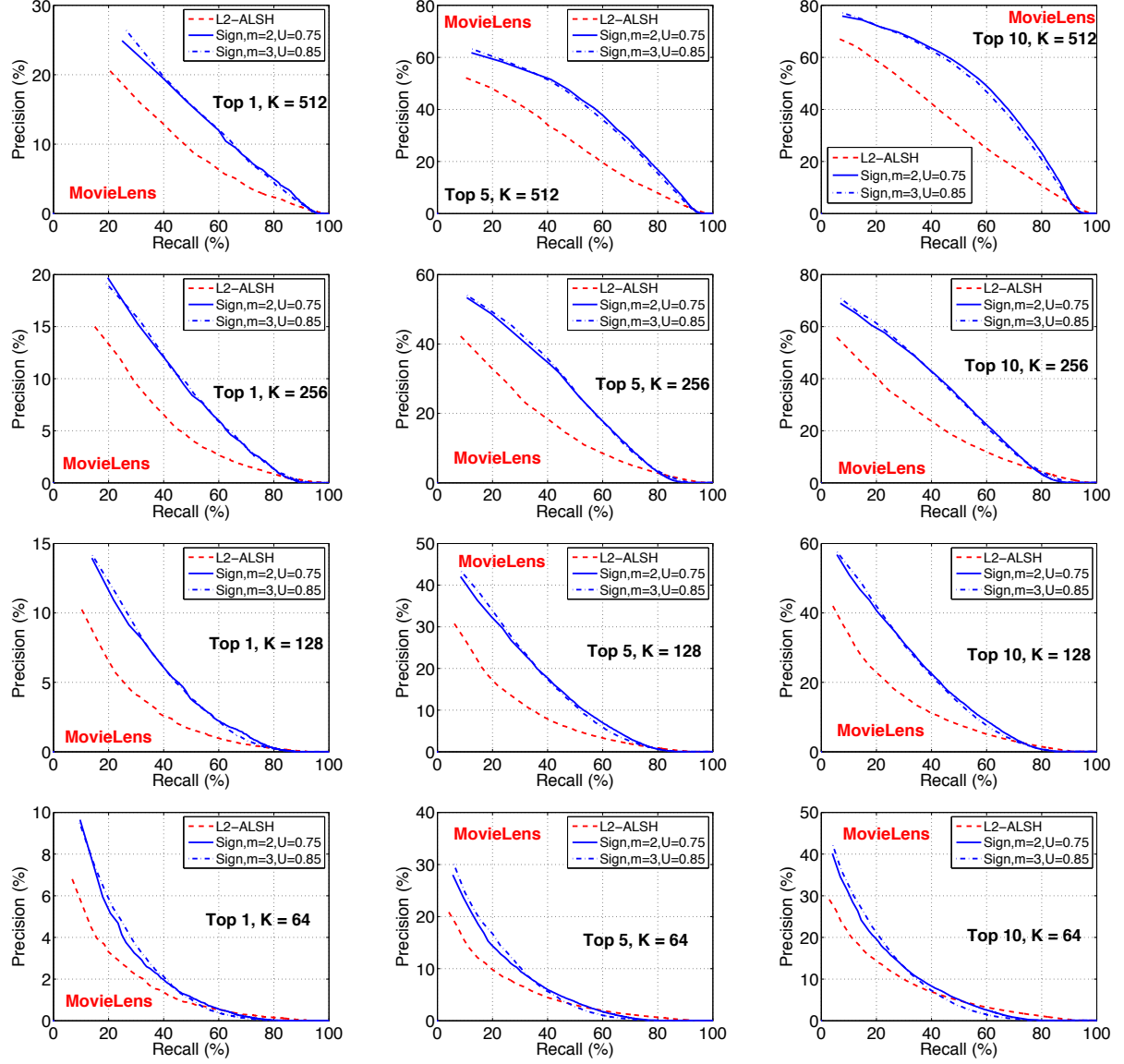
Figure 3: **Movielens**. Precision-Recall curves (higher is better), of retrieving top-T items, for $T = 1, 5, 10$. We vary the number of hashes K from 64 to 512. We compare L2-ALSH (using parameters recommended in [18]) with our proposed Sign-ALSH using two sets of parameters: $(m = 2, U = 0.75)$ and $(m = 3, U = 0.85)$. Sign-ALSH is noticeably better.

6 Ranking Evaluations

In [18], the L2-ALSH scheme was shown to outperform the LSH for L2 distance in retrieving maximum inner products. Since our proposal is an improvement over L2-ALSH, we focus on comparisons with L2-ALSH. In this section, we compare L2-ALSH with Sign-ALSH based on ranking.

6.1 Datasets

We use the two popular collaborative filtering datasets **MovieLens 10M** and **Netflix**, for the task of item recommendations. These are also the same datasets used in [18]. Each dataset is a sparse **user-item matrix** R, where $R(i, j)$ indicates the rating of user i for movie j. For getting the latent feature vectors from user item matrix, we follow the methodology of [18]. They use PureSVD procedure

described in [2] to generate user and item latent vectors, which involves computing the SVD of R

$$R = W\Sigma V^T$$

where W is $n_{users} \times f$ matrix and V is $n_{item} \times f$ matrix for some chosen rank f also known as latent dimension.

After the SVD step, the rows of matrix $U = W\Sigma$ are treated as the user characteristic vectors while rows of matrix V correspond to the item characteristic vectors. This simple procedure has been shown to outperform other popular recommendation algorithms for the task of top item recommendations in [2], on these two datasets. We use the same choices for the latent dimension f, i.e., $f = 150$ for Movielens and $f = 300$ for Netflix as [18].
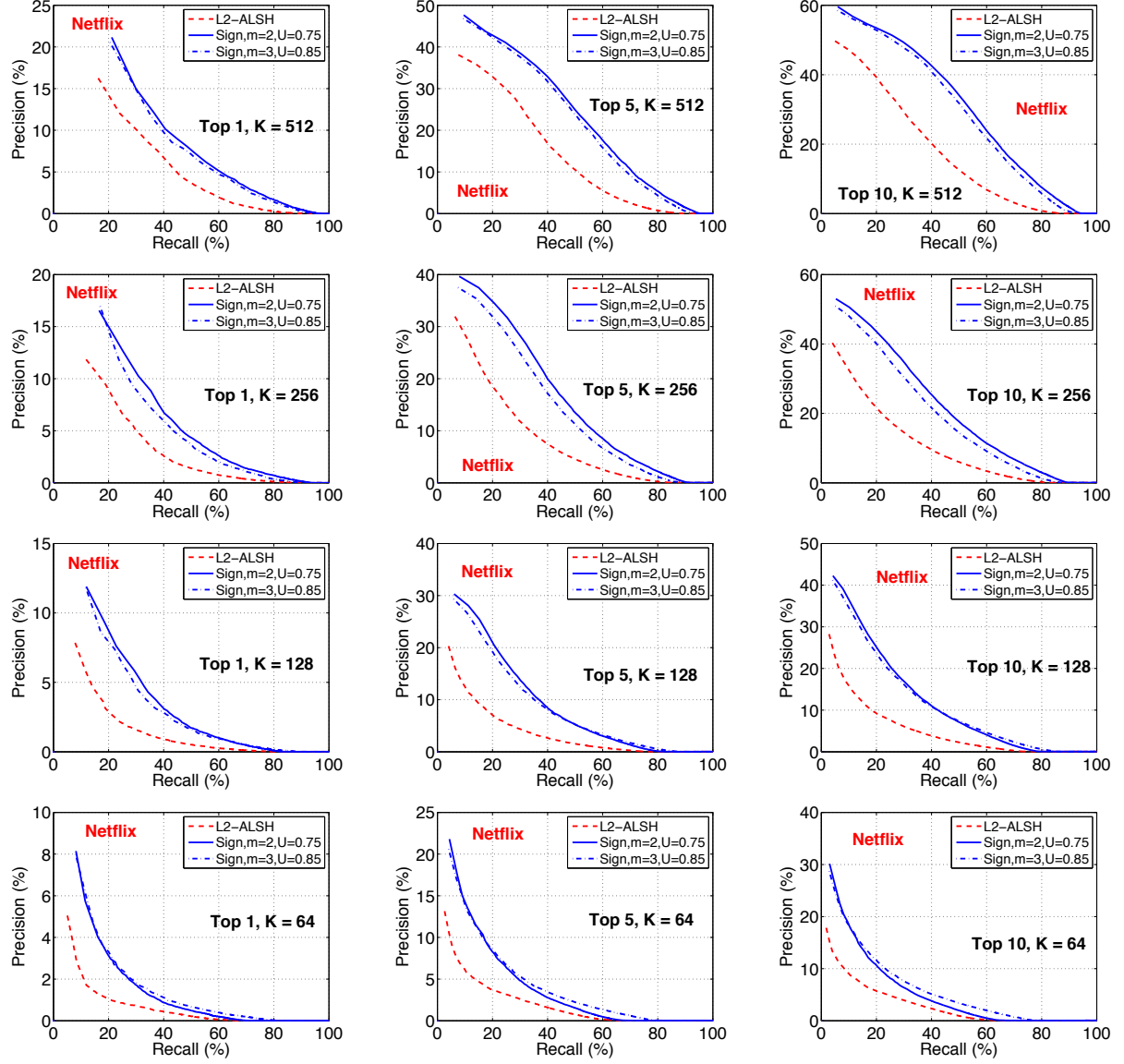
Figure 4: **Netflix**. Precision-Recall curves (higher is better), of retrieving top-T items, for $T = 1, 5, 10$. We vary the number of hashes K from 64 to 512. We compare L2-ALSH (using parameters recommended in [18]) with our proposed Sign-ALSH using two sets of parameters: $(m = 2, U = 0.75)$ and $(m = 3, U = 0.85)$. Sign-ALSH is noticeably better.

6.2 Evaluations

In this section, we show how the ranking of the two ALSH schemes, L2-ALSH and Sign-ALSH, correlates with the top-T inner products. Given a user i and its corresponding user vector u_i, we compute the top-T gold standard items based on the actual inner products $u_i^T v_j, \forall j$. We then generate K different hash codes of the vector u_i and all the item vectors v_js and then compute

$$Matches_j = \sum_{t=1}^{K} \mathbb{1}(h_t(u_i) = h_t(v_j)), \qquad (27)$$

where $\mathbb{1}$ is the indicator function and the subscript t is used to distinguish independent draws of h. Based on $Matches_j$ we rank all the items. Ideally, for a better hashing scheme, $Matches_j$ should be higher for items having higher inner products with the given user u_i. This procedure generates a sorted list of all the items for a given user vector u_i corresponding to the each hash function under consideration.

For L2-ALSH, we used the same parameters used and recommended in [18]. For Sign-ALSH, we used the two recommended choices shown in Section 4.3, which are $U = 0.75$, $m = 2$ and $U = 0.85$, $m = 3$. It should be noted that Sign-ALSH does not have the parameter r.

We compute the precision and recall of the top-T items for $T \in \{1, 5, 10\}$, obtained from the sorted list based on $Matches$. 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 kth ranked item, we check if this item belongs to the gold standard top-T list. If it is one of the top-T gold standard item, then we increment the count of
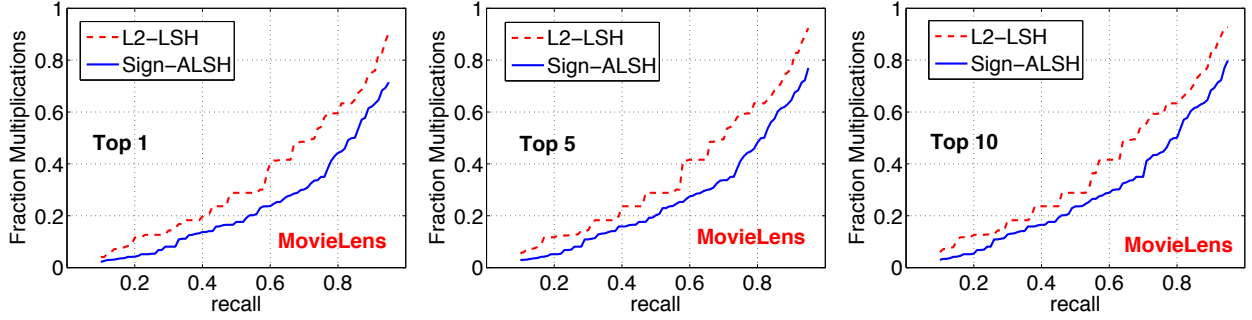
Figure 5: **MovieLens**. Recall-FIP (Fractions of Inner Products) curves for top-1, top-5, and top-10, for Sign-ALSH with L2-ALSH. We used the recommended parameters for L2-ALSH [18]. For Sign-ALSH, we used $m = 2$ and $U = 0.75$.

relevant seen by 1, else we move to $k + 1$. By k^{th} step, we have already seen k items, so the *total items seen* is k. The precision and recall at that point is then computed as:

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

We show performance for $K \in \{64, 128, 256, 512\}$. Note that it is important to balance both precision and recall. The method which obtains higher precision at a given recall is superior. Higher precision indicates higher ranking of the relevant items. We report averaged precisions and recalls over 2000 randomly chosen users.

The plots for MovieLens and Netflix datasets are shown in Figure 3 and Figure 4 respectively. 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 the top neighbors under inner products with Sign-ALSH compared to L2-ALSH. In addition, there is not much difference in the two different combinations of the parameters U and m in Sign-ALSH. The results are very consistent across both datasets.

7 LSH Bucketing Experiments

In this section, we evaluate the actual savings in the number of inner product evaluations for recommending top-T items for the MovieLens dataset. For this, we implemented the standard (K, L) algorithms in [9], where K is number of hashes in each hash table and L is the total number of tables. For each query point, the returned results are the union of matches in all L tables. To find the top-T items, we need to compute the actual inner products only on the candidate items retrieved by the bucketing procedure.

In this experiment, we choose $T \in \{1, 5, 10\}$ and compute the recall value for each combination of (T, K, L) for every query. For example, given query q and a (K, L)-LSH scheme, if $T = 10$ and only 5 of the true top-10 data points are retrieved, the recall will be 50% for this (T, K, L). At the same time, we can also compute the **FIP** (fraction of inner products):

$$\text{FIP} = \frac{(K \times L) + TotalRetrieved}{Total\ Items} \qquad (28)$$

which is basically the total number of inner products evaluation (where $K \times L$ represents the cost of hashing), normalized by the total number of items in the repository. Thus, for each q and (T, K, L), we can compute two values: recall and FIP. We also need to figure out a way to aggregate the results for all queries.

Typically the performance of bucketing algorithm is very sensitive to the choice of hashing parameters K and L. Ideally, to find best K and L, we need to know the operating threshold S_0 and the approximation ratio c in advance. Unfortunately, the data and the queries are very diverse and therefore for retrieving top-T near neighbors there is no common fixed threshold S_0 and approximation ratio c that works for different queries.

Our goal is to compare the hashing schemes, and minimize the effect of K and L on the evaluation. To get away with the effect of K and L, we perform rigorous evaluations of various K and L which includes optimal choices at various thresholds. For both the hashing schemes, we then select the best performing K and L and report the performance. This involves running the bucketing experiments for thousands of combinations and then choosing the best K and L to marginalize the effect of parameters in the comparisons. This all ensures that our evaluation is fair.

We choose the following scheme. For each (T, K, L), we compute the averaged recall and averaged FIP, over all queries. Then for each "target" recall level (and T), we can find the (K, L) which produces the best (lowest) averaged FIP. This way, for each T, we can compute a "FIP-recall" curve, which can be used to compare Sign-ALSH with L2-ALSH. We use $K \in \{4, 5, .., 20\}$ and $L \in \{1, 2, 3, ..., 200\}$.

The results are summarized in Figure 5. We can clearly see from the plots that for achieving the same recall for top-T, Sign-ALSH scheme needs to do less computations compared to L2-ALSH.

8 Conclusion

The MIPS (maximum inner product search) problem has numerous important applications in machine learning, databases, and information retrieval. [18] developed the

framework of Asymmetric LSH and provided an explicit scheme (L2-ALSH) for approximate MIPS in sublinear time. 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. Theoretical analysis and experimental study demonstrate that *Sign-ALSH* can be noticeably more advantageous than *L2-ALSH*.

Acknowledgement

References

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

[2] 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.

[3] 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.

[4] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijaya-narasimhan, 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.

[5] W. Dong, M. Charikar, and K. Li. Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In *SIGIR*, pages 123–130, 2008.

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

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

[8] 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.

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

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

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

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

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

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

[15] B. Neyshabur, N. Srebro, R. Salakhutdinov, Y. Makarychev, and P. Yadollahpour. The power of asymmetry in binary hashing. In *NIPS*, Lake Tahoe, NV, 2013.

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

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

[18] A. Shrivastava and P. Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). Technical report, arXiv:1405.5869 (To Appear in NIPS 2014. Initially submitted to KDD 2014), 2014.

[19] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *VLDB*, pages 194–205, 1998.