

# Asymmetric LSH (ALSH) for Sublinear Time Maximum Inner Product Search (MIPS)

KRISHNA PRIYA, CS14BTECH11010  
 KARTHIK YADAV, ES14BTECH11009  
 SHUBHAM MATTA, CS14BTECH11032  
 SAURABH SINGH, CS14BTECH11031

We present the results of implementation of first provably sub-linear time hashing algorithm for approximate Maximum Inner Product Search (MIPS) as presented in Prof. Anushmali and Prof. Ping Li's paper Asymmetric LSH (ALSH) for Sublinear Time Maximum Inner Product Search (MIPS). Searching with inner product as the underlying similarity measure is a known difficult problem and finding hashing schemes for MIPS was considered hard. While the existing Locality Sensitive Hashing (LSH) framework is insufficient for solving MIPS, we will look at asymmetric hashing schemes.

The idea is based on a key observation that the problem of finding maximum inner products, after independent asymmetric transformations, can be converted into the problem of approximate near neighbor search in classical settings.

## ACM Reference Format:

Krishna Priya, Karthik Yadav, Shubham Matta, and Saurabh Singh. 2010. Asymmetric LSH (ALSH) for Sublinear Time Maximum Inner Product Search (MIPS). *Proc. ACM Hum.-Comput. Interact.* 9, 4, Article 39 (March 2010), 10 pages. <https://doi.org/0000001.0000001>

## 1 INTRODUCTION

We are gonna look at the problem of Maximum Inner Product Search (MIPS). In this problem, given a query  $q \in \mathbb{R}^D$  and a giant collection  $C$  of  $N$  vectors in  $\mathbb{R}^D$ , search for  $p \in C$  s.t.,

$$p = \arg \max_{x \in C} q^T x$$

- Worst case  $O(N)$  for any query.  $N$  is huge.
- $O(N)$  quite expensive for frequent queries.

### 1.1 Applications :

#### 1.1.1 User Recommendation Systems.

- Matrix factorizations for collaborative filtering.
- Given a user  $u_i$ , the best item to recommend is a MIPS instance.

$$Item = \arg \max_j r_{i,j} = \arg \max_j u_i^T v_j$$

- Vectors  $u_i$  and  $v_j$  are learned. No control over norms.

---

Authors' addresses: Krishna Priya, CS14BTECH11010, [cs14btech11010@iith.ac.in](mailto:cs14btech11010@iith.ac.in); Karthik Yadav, ES14BTECH11009, [es14btech11009@iith.ac.in](mailto:es14btech11009@iith.ac.in); Shubham Matta, CS14BTECH11032, [cs14btech11032@iith.ac.in](mailto:cs14btech11032@iith.ac.in); Saurabh Singh, CS14BTECH11031, [cs14btech11031@iith.ac.in](mailto:cs14btech11031@iith.ac.in).

---

ACM acknowledges that this contribution was authored or co-authored by an employee, contractor, or affiliate of the United States government. As such, the United States government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for government purposes only.

© 2010 Association for Computing Machinery.  
 2573-0142/2010/3-ART39 \$15.00  
<https://doi.org/0000001.0000001>

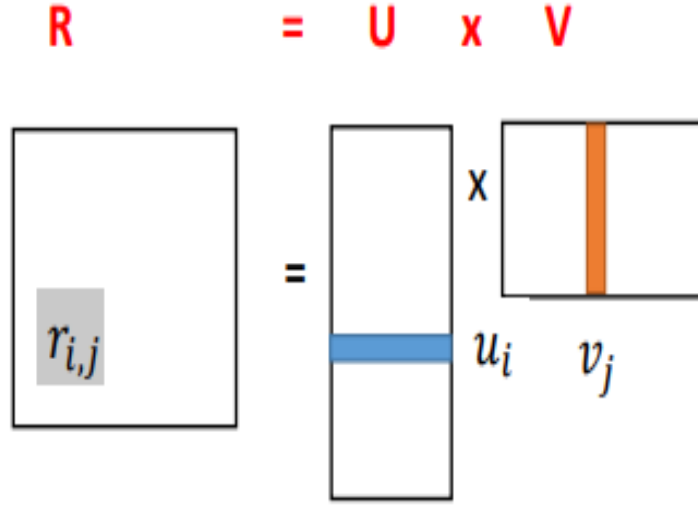


Fig. 1. User-Item Recommendation

### 1.1.2 Multi-Class Prediction.

- Standard multi-class SVM in practice learns a weight vector  $w_i$  for each of the class label  $i \in L$ .
- Predicting a new  $x_{test}$ , is a MIPS instance :

$$y_{test} = \arg \max_{i \in L} x_{test}^T w_i$$

$w_i$ s are learned and usually have varying norms.

### 1.1.3 Web-Scale Deep Architectures.

- Activation of hidden node  $i$  monotonic in  $x^T w_i$ .
- MAXOUT only requires updating hidden nodes having max activations.

$$h(x) = \max_j x^T w_j$$

- Efficient MIPS gives Fast Training and Testing of Giant Networks

## 2 LOCALITY SENSITIVE HASHING (LSH)












Can we do something better (sub-linear) than the trivial linear method?

- MIPS is different from Near Neighbor Search :

$$\arg \min_{x \in C} \|q - x\|_2^2 = \arg \min_{x \in C} (\|x\|_2^2 - 2q^T x) \neq \arg \max_{x \in C} q^T x$$

- **LSH**  $h : \mathbb{R}^D \Rightarrow [0, 1, 2, \dots, N]$  with the following property:  $Pr_h[h(x) = h(y)] = f(sim(x, y))$ , where  $f$  is monotonically increasing.
- Concatenate  $K$  independent hash functions to create a meta-hash function:  $B_l(x) = [h_1(x); h_2(x); \dots; h_K(x)]$ .
- There are  $L$  such  $B_l(x)$ ,  $l = 1, 2, \dots, L$ . (**increases recall**)
- **Pre-processing Step:** Assign  $x_i \in S$  to  $B_l(x_i)$  in the hash table  $l$ , for  $l = 1, 2, \dots, L$ .

- **Querying Step:** union of all elements from buckets  $B_l(q)$ , union is taken over all hash tables  $l$ , for  $l = 1, 2, \dots, L$ .

Table 1				Table L			
$h_1^1$	...	$h_K^1$	Buckets	...	$h_1^L$	...	$h_K^L$ Buckets
00	...	00	  ...		00	...	  ...
00	...	01	  ...		00	...	  ...
00	...	10	Empty		00	...	   ...
...	...	...	...		...	...	...
11	...	11	...		11	...	Empty

### 3 $L_2$ LSH

- Given a fixed number  $r$ , we choose a random vector  $a : a_i \sim N(0, 1)$ , and a scalar  $b$  generated uniformly at random from  $[0, r]$ .
- Hash function :  $h_{a,b}^{L_2}(x) = \left\lfloor \frac{a^T x + b}{r} \right\rfloor$
- It can be shown that  $\Pr(h_{a,b}^{L_2}(x) = h_{a,b}^{L_2}(y))$  is monotonic in  $\|x - y\|_2$ .
- Unless the given data has a constant norm,  $\|x - y\|_2 = \sqrt{\|x\|_2^2 + \|y\|_2^2 - 2x^T y}$  is not monotonic in the inner product  $x^T y$ .
- Simplifying the above result to a query and a data point:  
 $\Pr(h(q) = h(x)) = f(\|q\|_2^2 + \|x\|_2^2 - 2q^T x)$

#### LSH cannot solve MIPS

- For inner products, we can have  $x$  and  $y$ , such that  $x^T y > x^T x$ .
- Self similarity is not the highest similarity
- Under any hash function  $\Pr(h(x) = h(x)) = 1$ . But we need  $\Pr(h(x) = h(y)) > \Pr(h(x) = h(x)) = 1$
- **Solution : ALSH**

### 4 ALSH

- Use  $P(\cdot)$  for creating buckets.
- Use  $Q(\cdot)$  for querying probe buckets.  $\Pr(Q(x) = P(x)) \neq 1$
- All we need is  $\Pr(Q(q) = P(x))$  to be monotonic in  $q^T x$ .
- Recall for  $L_2$  LSH:  $\Pr(h(q) = h(x)) = f(\|q\|_2^2 + \|x\|_2^2 - 2q^T x)$   
 $\Rightarrow$  It suffices to construct  $P$  and  $Q$  such that  $\|Q(q)\|_2^2 + \|P(x)\|_2^2 - 2Q(q)^T P(x)$  is monotonic (or approximately) in  $q^T x$ .

#### 4.1 ALSH Construction

- (1) WLOG, let  $\|q\|_2 = 1$ . Also let  $\|x_i\|_2 \leq U < 1, \forall x_i \in S$ . If not divide all  $x_i$ s by  $\max_{x_i \in S} \frac{\|x_i\|_2}{U}$

$$P : \mathbb{R}^D \mapsto \mathbb{R}^{D+m} \quad P(x) = [x; \|x\|_2^2; \|x\|_2^4; \dots; \|x\|_2^{2m}]$$

$$Q : \mathbb{R}^D \mapsto \mathbb{R}^{D+m} \quad Q(x) = [x; \frac{1}{2}; \frac{1}{2}; \dots; \frac{1}{2}]$$

$$\|P(x_i)\|_2^2 = \|x_i\|_2^2 + \|x_i\|_2^4 + \dots + \|x_i\|_2^{2m} + \|x_i\|_2^{2m+1}$$

$$\|Q(q)\|_2^2 = \|q\|_2^2 + \frac{m}{4}$$

$$\|Q(q) - P(x_i)\|_2^2 = \|Q(q)\|_2^2 + \|P(x)\|_2^2 - 2Q(q)^T P(x)$$

$$= (1 + \frac{m}{4}) - 2q^T x_i + \|x_i\|_2^{2^{m+1}}$$

$$\|x_i\|_2^{2^{m+1}} \rightarrow 0 \Rightarrow \arg \max_{x \in C} q^T x \simeq \arg \min_{x \in C} \|Q(q) - P(x)\|_2$$

#### 4.1.1 Removing the condition $\|q\|_2 = 1$ :

$$(1) P : \mathbb{R}^D \mapsto \mathbb{R}^{D+2m}$$

$$P(x) = [x; \|x\|_2^2; \|x\|_2^4; \dots; \|x\|_2^{2^m}; \frac{1}{2}; \frac{1}{2}; \dots; \frac{1}{2}]$$

$$Q : \mathbb{R}^D \mapsto \mathbb{R}^{D+2m}$$

$$Q(x) = [x; \frac{1}{2}; \frac{1}{2}; \dots; \frac{1}{2}; \|x\|_2^2; \|x\|_2^4; \dots; \|x\|_2^{2^m}]$$

$$\|Q(q) - P(x_i)\|_2^2 = \frac{m}{2} + \|S(x)\|_2^{2^{m+1}} + \|S(q)\|_2^{2^{m+1}} - 2q^T x(\frac{U^2}{M^2})$$

$$\|S(x)\|_2^{2^{m+1}}, \|S(q)\|_2^{2^{m+1}} \leq U^{2^{m+1}} \rightarrow 0$$

$$(2) S(x) = \frac{U}{M}x; M = \max_{x_i \in S} \|x_i\|_2$$

#### 4.1.2 The Final Algorithm :

- **Preprocessing:** Scale  $\Rightarrow$  Append 3 Numbers  $\Rightarrow$  Usual L2-LSH
  - Scale  $x \in C$  to have norm  $\leq 0.83$
  - Append  $\|x_i\|_2$ ,  $\|x_i\|_4$ , and  $\|x_i\|_8$  to vector  $x_i$ .
  - Use standard L2 hash to create hash tables.
- **Querying:**
  - Append 0.5 three times to the query  $q$ .
  - Use standard L2 hash on the transformed query to probe buckets.

## 5 EVALUATION METRICS

- **Datasets:** Movielens and Netflix.
- Using SVD, we decompose  $R$ (Ratings matrix) into user and item latent vectors. The term  $R(i, j)$  indicates the rating of user ' $i$ ' for an item ' $j$ '.
- Latent dimension  $f = 150$  for Movielens data and  $= 300$  for Netflix data
- How the 2 hash functions correlate with the top-T inner products?  $T=10$ 
  - Compute the top-T inner prods based on the actual inner products  $u_i^T v_j$ , for every  $j$ , where  $u_i$  is a query and  $v_j$  is a vector point.
  - Rank all items based on  $Matches_j = \sum \delta(h_t(u_i) = h_t(v_j))$ , where  $\delta$  is the indicator function. The number of times its hash value matches with the hash values of query which is  $u_i$ .

### 5.1 Precision vs Recall graph

- Compute the precision and recall of the top-T items for any T obtained from the sorted list based on Matches. Start at the top of the ranked item list and walk down in order.
- If we are at the k-th ranked item, we check if this item belongs to the top-T similar vector points' list.
- If it is one of the top-T, then we increment the count of *relevant seen* by 1, else we move to  $k + 1$ . By k-th step, the total items seen is k.

- $Precision = \frac{\text{relevant seen}_k}{k}$   
 $Recall = \frac{\text{relevant seen}_T}{T}$
- For each query choose from  $K \in \{5, 6, \dots, 30\}$  and  $L \in \{1, 2, \dots, 200\}$ . Use  $m = 3$ ,  $U = 0.83$  and  $r = 2.5$ . Around 10,000 experiments

## 6 MATHEMATICAL PROOFS AND DEFINITIONS

### 6.1 c-Approximate Near Neighbor Data Structure.

- Given : a set of points in a  $\mathbb{R}^D$  and parameters  $S_0 > 0, \delta > 0$  and a query point  $q$ , with probability  $1 - \delta$ , if  $\exists S_0$ -near neighbor of  $q$  in  $P(\text{Sim}(q, p) \geq S_0)$ , it outputs some  $cS_0$ -near neighbor of  $q$  in  $P$ . Popularly c-NN is solved similar to LSH
- **(Locality Sensitive Hashing (LSH))** A family  $H$  is called  $(S_0, cS_0, p_1, p_2)$ -sensitive if,  $\forall x, y \in \mathbb{R}^D$ ,  $h$  chosen uniformly from  $H$  satisfies the following : ( $p_1 > p_2$  and  $c < 1$ )
  - if  $\text{Sim}(x, y) \geq S_0$  then  $Pr_H(h(x) = h(y)) \geq p_1$
  - if  $\text{Sim}(x, y) \leq cS_0$  then  $Pr_H(h(x) = h(y)) \leq p_2$
- 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  $\rho = \frac{\log p_1}{\log p_2} < 1$ .

### 6.2 There can not exist any LSH family for MIPS

Proof:

- Let  $\exists$  such hash function  $h$ .
- For un-normalized inner products,  $\text{Sim}(x, x) = x^T x = \|x\|_2^2$
- There may exist another points  $y$ , such that  $\text{Sim}(x, y) = y^T x > \|x\|_2^2 + M$ , for any constant  $M$ .
- Under any single randomized hash function  $h$ , the collision probability of the event  $\{h(x) = h(y)\}$  is always 1.
- So if  $h$  is an LSH for inner product then the event  $h(x) = h(y)$  should have higher probability compared to the event  $h(x) = h(x)$
- This is not possible because the probability can not be greater than 1 .
- Note that we can always choose  $y$  with  $\text{Sim}(x, y) = S_0 + \delta > S_0$  and  $cS_0 > \text{Sim}(x, x) \forall S_0$  and  $\forall c < 1$ .

### 6.3 Asymmetric Locality Sensitive Hashing (ALSH)

- A family  $H$ , along with  $P : \mathbb{R}^D \mapsto \mathbb{R}^{D'}$  (Processing transformation),  $Q : \mathbb{R}^D \mapsto \mathbb{R}^{D'}$  (Query 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  $H$  satisfies the following: ( $p_1 > p_2$  and  $c < 1$ )
  - if  $\text{Sim}(x, y) \geq S_0$  then  $Pr_H(h(Q(q)) = h(P(x))) \geq p_1$
  - if  $\text{Sim}(x, y) \leq cS_0$  then  $Pr_H(h(Q(q)) = h(P(x))) \leq p_2$
 Here  $x$  is any point in the collection  $S$ .
- Given a family of hash function  $H$  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  $\rho = \frac{\log p_1}{\log p_2}$

Proof:

- Use the Fast Similarity Search with LSH with a slight modification.
- While preprocessing, we assign  $x_i$  to bucket  $B_l(P(x_i))$  in table  $l$ .
- While querying with query  $q$ , we retrieve elements from bucket  $B_l(Q(q))$  in the hash table  $l$ .
- By definition, the probability of retrieving an element, under this modified scheme, follows the same expression as in the original LSH.

#### 6.4 $L_2$ LSH Recap

- Given a fixed number  $r$ , we choose a random vector  $a : a_i \sim N(0, 1)$ , and a scalar  $b$  generated uniformly at random from  $[0, r]$ .
- Hash function :  $h_{a,b}^{L_2}(x) = \left\lfloor \frac{a^T x + b}{r} \right\rfloor$
- It can be shown that  $Pr(h_{a,b}^{L_2}(x) = h_{a,b}^{L_2}(y))$  is monotonic in  $\|x - y\|_2$ .
- Unless the given data has a constant norm,  $\|x - y\|_2 = \sqrt{\|x\|_2^2 + \|y\|_2^2 - 2x^T y}$  is not monotonic in the inner product  $x^T y$ .
- Simplifying the above result to a query and a data point:  
 $Pr(h(q) = h(x)) = f(\|q\|_2^2 + \|x\|_2^2 - 2q^T x)$

#### 6.5 ALSH Construction Recap

- (1) WLOG, let  $\|q\|_2 = 1$ . Also let  $\|x_i\|_2 \leq U < 1, \forall x_i \in S$ . If not divide all  $x_i$ s by  $\max_{x_i \in S} \frac{\|x_i\|_2}{U}$

$$P : \mathbb{R}^D \mapsto \mathbb{R}^{D+m} \quad P(x) = [x; \|x\|_2^2; \|x\|_2^4; \dots; \|x\|_2^{2^m}]$$

$$Q : \mathbb{R}^D \mapsto \mathbb{R}^{D+m} \quad Q(x) = [x; \frac{1}{2}; \frac{1}{2}; \dots; \frac{1}{2}]$$

$$\|P(x_i)\|_2^2 = \|x_i\|_2^2 + \|x_i\|_2^4 + \dots + \|x_i\|_2^{2^m} + \|x_i\|_2^{2^{m+1}}$$

$$\|Q(q)\|_2^2 = \|q\|_2^2 + \frac{m}{4}$$

$$\|Q(q) - P(x_i)\|_2^2 = \|Q(q)\|_2^2 + \|P(x)\|_2^2 - 2Q(q)^T P(x)$$

$$= (1 + \frac{m}{4}) - 2q^T x_i + \|x_i\|_2^{2^{m+1}}$$

$$\|x_i\|_2^{2^{m+1}} \rightarrow 0 \Rightarrow \arg \max_{x \in C} q^T x \simeq \arg \min_{x \in C} \|Q(q) - P(x)\|_2$$

#### 6.6 Collision probability- $L_2$ LSH

- $Pr(h_{a,b}^{L_2}(x) = h_{a,b}^{L_2}(y)) = F_r(d)$
- $F_r(d) = 1 - 2\Phi(-\frac{r}{d}) - (\frac{2}{\sqrt{2\pi}(\frac{r}{d})})(1 - e^{-\frac{(\frac{r}{d})^2}{2}})$
- $\Phi(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$ ,  $x$  is the cumulative density function (cdf) of standard normal distribution and  $d = \|x - y\|_2$  is the Euclidean distance between the vectors  $x$  and  $y$ .

#### 6.7 Fast Algorithm for MIPS

- Given a  $c$ -approximate instance of MIPS, i.e.,  $\text{Sim}(q, x) = q^T x$ , and a query  $q$  such that  $\|q\|_2 = 1$  along with a collection  $S$  having  $\|x\|_2 \leq U < 1 \forall x \in S$ .
- We have the following two conditions for hash function  $h_{a,b}^{L_2}(x)$ :
  - if  $q^T x \geq S_0$  then  
 $Pr[h_{a,b}^{L_2}(Q(q)) = h_{a,b}^{L_2}(P(x))] \geq F_r(\sqrt{1 + \frac{m}{4} - 2S_0 + U^{2^{m+1}}})$
  - if  $q^T x \leq cS_0$  then  
 $Pr[h_{a,b}^{L_2}(Q(q)) = h_{a,b}^{L_2}(P(x))] \leq F_r(\sqrt{1 + \frac{m}{4} - 2cS_0})$

Proof :

$$\begin{aligned}
 & Pr[h_{a,b}^{L_2}(Q(q)) = h_{a,b}^{L_2}(P(x))] \\
 &= F_r(\|Q(q) - P(x)\|_2) \\
 &= F_r(\sqrt{1 + \frac{m}{4} - 2q^T x + \|x\|_2^{2m+1}}) \\
 &\geq F_r(\sqrt{1 + \frac{m}{4} - 2S_0 + U^{2m+1}})
 \end{aligned}$$

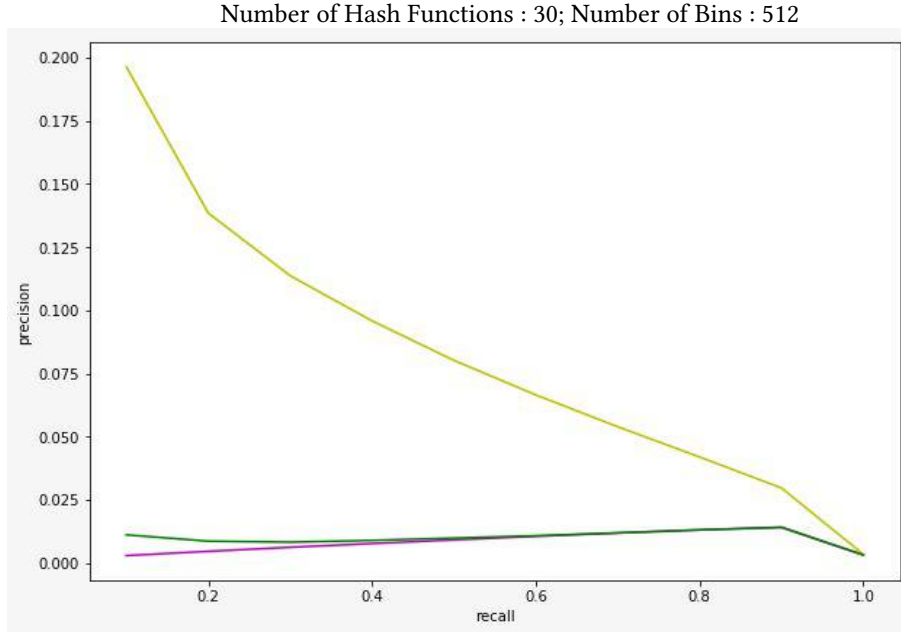
- The last step follows from the monotonically decreasing nature of  $F$  combined with inequalities  $q^T x \geq S_0$  and  $\|x\|_2 \leq U$ .
- We have also used the monotonicity of the square root function.
- The second inequality similarly follows using  $q^T x \leq cS_0$  and  $\|x\|_2 \geq 0$ .

6.8

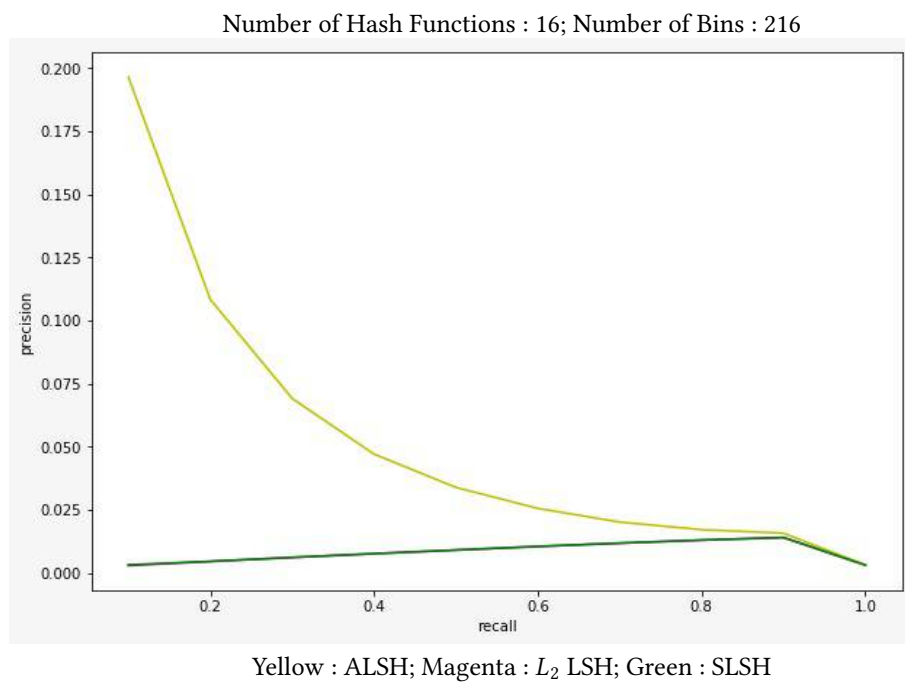
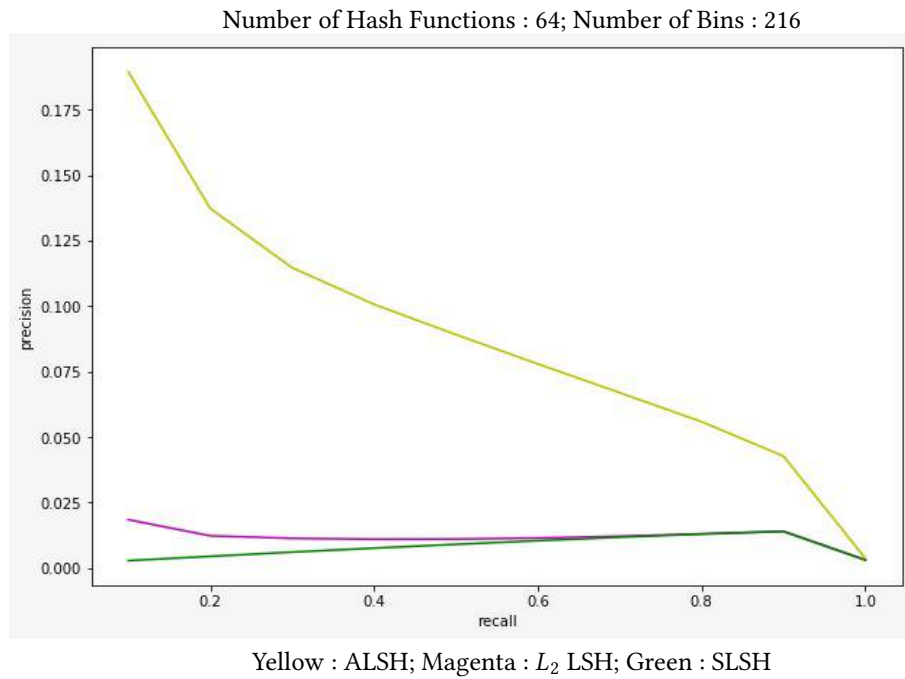
- $p_1 = F_r(\sqrt{1 + \frac{m}{4} - 2S_0 + U^{2m+1}})$
- $p_2 = F_r(\sqrt{1 + \frac{m}{4} - 2cS_0})$
- Given a  $c$ -approximate MIPS instance,  $\rho$  is a function of 3 parameters:  $U, m, r$ . The algorithm with the best query time chooses  $U, m$  and  $r$ , which minimizes the value of  $\tilde{I}A_{\rho}$ .
- Let  $\rho^* = U, m, r \frac{\log(F_r(\sqrt{1 + \frac{m}{4} - 2S_0 + U^{2m+1}}))}{\log(F_r(\sqrt{1 + \frac{m}{4} - 2cS_0}))}$  such that  $\frac{U^{2m+1}}{2S_0} < 1 - c$   $m \in \mathbb{N}^+, r > 0$  and  $U \in (0, 1)$ .

## 7 RESULTS

We look at plots of Precision vs Recall with different parameters.



Yellow : ALSH; Magenta :  $L_2$  LSH; Green : SL2SH





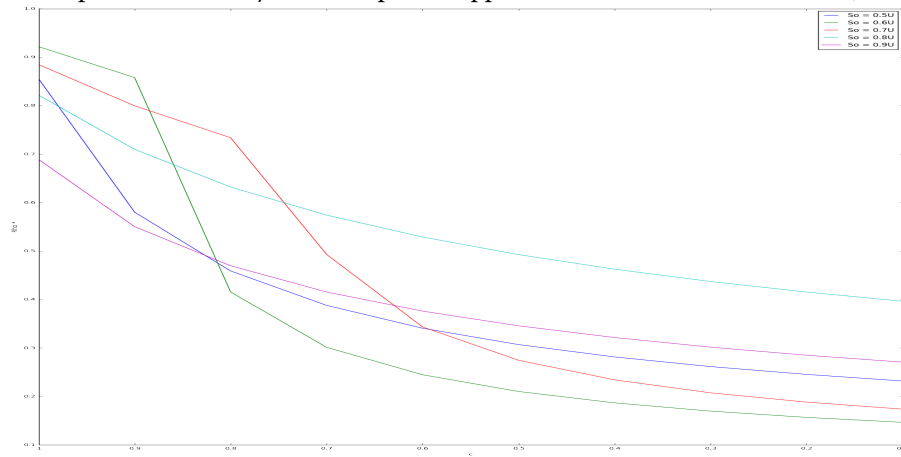
## 8 OBSERVATIONS

**These results are similar to the ones achieved by the paper. Slight deviations are observed in the results are because the data we experimented on was smaller compared to the paper since we had to run it on our local systems.**

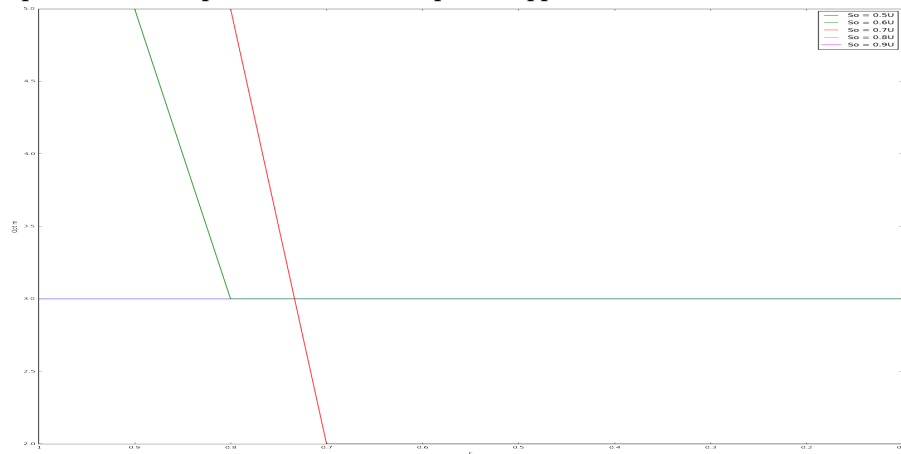
In the paper the maximum precision obtained for  $K$  is 60%. We got 20%. The query set we took is larger compared to the dataset. The dataset used by the paper is magnitudes larger than what we used. More importantly, we obtained similar trends compared to the paper. We got higher precision(much) for ALSH compared to SLSH and L2LSH. The results are so because more data points give us better scope for finer boundaries between true positives, false positives, true negatives and false negatives.

## 9 PARAMETER ESTIMATION

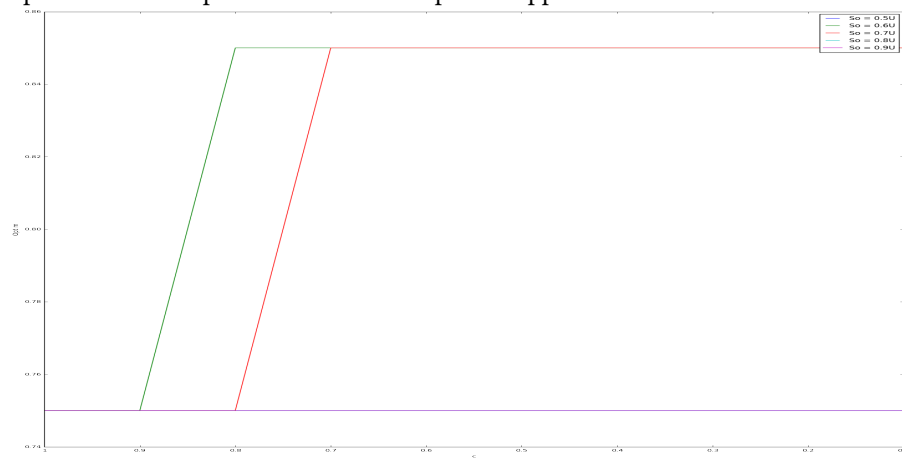
Optimal values of  $\rho^*$  with respect to approximation ratio  $c$  for different  $S_0$



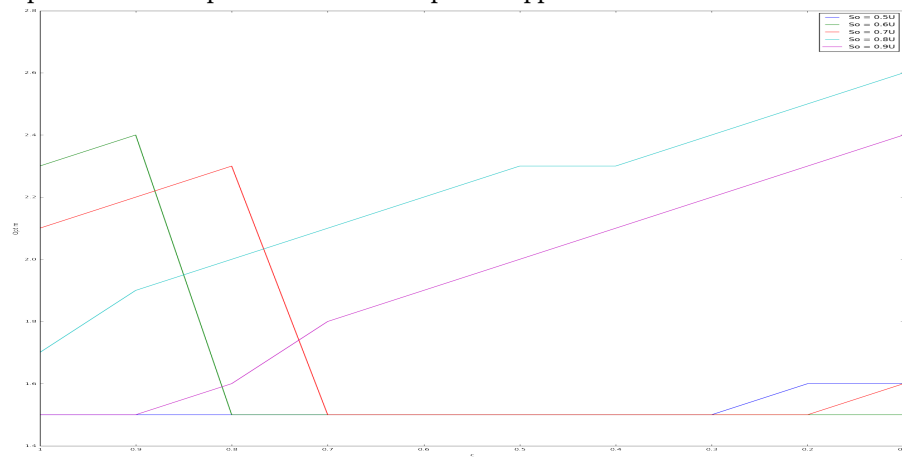
Optimal values of parameter  $m$  with respect to approximation ratio  $c$  for different  $S_0$



Optimal values of parameter  $U$  with respect to approximation ratio  $c$  for different  $S_0$



Optimal values of parameter  $r$  with respect to approximation ratio  $c$  for different  $S_0$



## 10 OBSERVATIONS

In the paper, they took a lot more points in grid search and got smoother looking graph. Regardless the trends we obtained are quite similar to the trends depicted in the paper.

Received February 2007; revised March 2009; accepted June 2009