Locality-Sensitive Hashing without false negatives for ℓ_1

COMS 6998 Algorithmic Techniques for Massive Data

Negev Shekel-Nosatzki, Chen-Yu Yang, Parthiban Loganathan

December 2015

Contents

1 Introduction

1.1 Abstract

In this article we analyze Rasmus Pagh recent publication "Locality-sensitive Hashing without False Negatives" and extend its findings to ℓ_1. To simplify our analysis - we have focused on integers bounded by range ($[N]^d$) - by defining random bit mappings to create an efficient hashing that guarantees the collision of close points always and rarely collides far points. We achieve this by mapping far points closer but farther than cr to preserve the LSH property. This results in a much faster algorithm (by expectation) than any existing deterministic ℓ_1 Nearest Neighbor Search algorithm with near-linear space, similar to the results achieved by Pagh.

1.2 Acknowledgements

We would like to thank Prof. Alexandr Andoni for assisting us in project guidelines and direction.

2 Background

2.1 Locality-Sensitive Hashing

In problems like Nearest Neighbor Search, we need a data structure to distinguish between close and far points with near-linear space and sub-linear query time. This motivates Locality-Sensitive Hashing (LSH).

Definition 1 (informal). *A* locality-sensitive hash function, LSH *is a random hash function $h : \mathbb{R}^d \to U$ (h drawn from a family $\mathcal{H}$, U some finite set) such that*

1. $d(q,p) \le r \implies \Pr[h(q) = h(p)] = P_1$ *is "not-so-small"*
2. $d(q,p) > cr \implies \Pr[h(q) = h(p)] = P_2$ *is "small"*

Generally, $P_1 < P_2$ and we use n^ρ hash tables where:

$$\rho = \frac{\log(1/P_1)}{\log(1/P_2)}.$$

For the construction of LSH for Hamming Space $\{0,1\}^d$ with distance metric $\|x-y\|_{Ham} = |\{x_i \neq y_i\}|$, we define the hash family, $\mathcal{H} : \{g : \{0,1\}^d \to \{0,1\}^k\}$ as:

$$g(p) := (h_1(p), h_2(p), \dots, h_k(p)),$$

where $h_i(p) := p_j$ for random $j \leftarrow [d]$. It essentially randomly selects k bits from d-dimensional point. For NNS, we maintain $L = n^\rho$ such Hamming-LSH tables g. We will see how we can always collide close points by choosing correlated hash functions instead of random independent ones in Pagh'16 and how this applies to our construction.

2.2 LSH without false negatives from Pagh'16

Theorem (Pagh'16). *Given $S \subseteq \{0,1\}^d, c > 1$ and $r \in \mathbf{N}$ there exists a data structure with parameter ρ such that:*

- *On query $y \in \{0,1\}^d$ the data structure us guaranteed to return $x \in S$ with $\|x-y\| < cr$ if there exists $x' \in S$ with $\|x'-y\| \le r$.*
- *The size is $O(dn^{1+\rho})$ bits where $n = |S|$.*
- *The expected query times is $O(n^\rho(1 + d/w))$, where w is the word length.*

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Figure 1: Sample $\mathcal{A}$ for $d = 7$ and $r = 2$

- *Parameter ρ is a function of n, r and c bounded by $\rho \leq \min\left(\ln 4/c + \frac{\log r}{\log n}, 1/c + \frac{r}{\log n}\right)$. If $\log(n)/cr \in \mathbf{N}$, then $\rho = 1/c$.*

The important takeaways from Pagh's approach that are relevant to our construction are below:

- It's *covering*, meaning that it guarantees collision for $\|x - y\| \leq r$. ie. $P_1 = 1$. The probability of false negatives is 0.
- Far points collide with low probability.
- It chooses correlated hash functions instead of selecting them independently to cover all ways in which two points can be r apart.
- Performance is almost as good as what we saw in class where we had constant probability of false negatives. The algorithm is efficient if $r = \Theta(\log n)$.

2.3 Hash function family from Pagh'16

$$\mathcal{H}_{\mathcal{A}} = \{x \mapsto x \wedge a | a \in \mathcal{A}\}$$

where $\mathcal{A} \subseteq \{0,1\}^d$ is a set of bit masks. Given a query q, we iterate through all $h \in \mathcal{H}_{\mathcal{A}}$ and check for collisions. To generate a covering set $\mathcal{A}$, we define it as:

$$\mathcal{A}(m) = \{a(v) | v \in \{0,1\}^{r+1} \setminus \{\mathbf{0}\}\}$$

where $a(v)_i = \langle m(i), v\rangle \; \forall \; v \in \{0,1\}^{r+1}$ and $m : \{1, \cdots, d\} \to \{0,1\}^{r+1}$ is a random uniformly selected function.

For example, this procedure produces the covering set in 1 when $m(i)$ is the binary representation of i. Each column represents an $a \in \mathcal{A}$ where 1 samples the coordinate in the point. In Fig. 1, $r = 2$ and we see that for any 2 coordinates we select, there exists a column where both coordinates are 1s. This ensures that we can always sample the 2 coordinates that differ between points which are 2-apart.

It is sufficient for any one column has the collision. We could always achieve vectors a that select all r possible coordinates by having $\binom{d}{r}$ possible values for a but Pagh's procedure produces an efficient covering set with fewer columns.

LSH is fast only when the input is in Hamming space. It is possible to extend the algorithm to the ℓ_1 norm by embedding ℓ_1 into Hamming space. However, it increases the query time and error and adds complexity to the algorithm. We will attempt to extend Pagh's work to ℓ_1 in a manner that is more efficient than simply embedding ℓ_1 to Hamming and then running Pagh's algorithm.

2.4 Other approaches

In Datur-Immorlica-Indyk-Mirrokni'04, an LSH scheme based on p-stable distributions is proposed. It works for any ℓ_p norm $p \in (0, 2]$ which includes ℓ_1 that we use in our attempt. They show that there exists an algorithm for (R, c)-NN which uses $O(dn + n^{1+\rho})$ space and $O(n^\rho log_{1/\delta} n)$ query time.

3 Main Theorem

Our main theorem for this project:

Theorem 2. *Given $S \subseteq [N]^d, c > 1$ and $r \in \mathbf{N}$ there exists a data structure with parameter ρ such that:*

- *On query $y \in [N]^d$ the data structure is guaranteed to return $x \in S$ with $\|x - y\| < cr$ if there exists $x' \in S$ with $\|x' - y\| \leq r$.*
- *The size is $\mathcal{O}(r \cdot \log N \cdot (N + d \cdot n^{1+\rho})$ bits where $n = |S|$.*
- *The expected query times is $\mathcal{O}((r \cdot \log N \cdot d)/w \cdot n^\rho)$, where w is the word length.*
- *Parameter ρ is a function of n, r and c bounded by:*

$$\rho \leq \min\left(\ln 4/c + \frac{\log r}{\log n}, 1/c + \frac{r}{\log n}\right)$$

If $\log(n)/cr \in \mathbf{N}$, then $\rho = 1/c$.

Note that this is very similar results to Pagh, with another $r \log(N)$ factor as a result from the $\ell 1$ domain and our construction. Similarly to Pagh, our algorithm will achieve similar features like finding all close points, etc. Furthermore, we will also show how we can exchange the added r factor with a d factor, with some constant factor to the denominator and the exponent (trade-off).

4 Construction

4.1 Distance-preserving Mappings

The main idea behind our extension is to create mappings from $\ell 1$ to Hamming that will distinguish between close points and far points. An example of such mapping would be unary embedding which maintains exact distance but is highly inefficient as it grows the dimension by factor N (the range of target domain). In our project we explored more efficient mappings which we called distance-preserving mappings.

Definition 3 (Distance-Preserving Mapping)**.** *Define $f : \mathbf{N} \to \{0, 1\}^\alpha$ to be β distance-preserving α-bit mapping, if $\forall x, y \in \mathbf{N}$ s.t. $|x - y| \leq \beta : \|f(x) - f(y)\|_{Ham} = |x - y|$*

As an example, the following function is α distance-preserving α-bit mapping:

$$f(x) = \begin{cases} 0^{\alpha-\beta}1^\beta, & \text{if } \beta < \alpha \\ 1^{2\alpha-\beta}0^{\beta-\alpha}, & \text{otherwise} \end{cases}$$

Where $\beta = x \pmod{2\alpha}$.

We are interested in random distance preserving mappings so we can have far points remain far on expectation.

Let $\tau : \mathbf{N} \to [\alpha]$ be the following random function that maps integers to coordinates:

$$\tau(x) = \begin{cases} x, & \text{if } x \leq \alpha/2 \\ \mathcal{U}([\alpha] \setminus \bigcup_{i=1}^{\alpha/2} \tau(x-i)), & \text{otherwise} \end{cases}$$

($\tau(x)$ simply samples a uniformly random coordinate which wasn't sampled in the previous $\alpha/2$ rounds.)

And let $f : \mathbf{N} \to \{0,1\}^\alpha$ be the following random mapping:

$$f(x) = \begin{cases} f(0) & = \vec{0} \\ f(x) & = f(x-1) \oplus e_{\tau(x)} \end{cases}$$

($f(x)$ simply "flips" the $\tau(x)$ coordinate from $f(x-1)$)

Claim 4. *f is $\alpha/2$ distance-preserving α-bit mapping*

Proof. Define $d(x,y) = \|f(x) - f(y)\|_{Ham}$.
Since $\forall x \in \mathbf{N}, i \in [\alpha/2], \tau(x) \neq \tau(x-i)$, then:

$$\forall j \in [\alpha/2], d(x+j, x) = \left\| (\sum_{i=1}^{j} e_{\tau(x+i)}) \oplus x - x \right\|_{Ham} = j$$

□

Lemma 5 (Lower bound on distance expectation). *$\forall x, y$ s.t. $|x - y| \geq \alpha/2 : \mathbb{E}_f[d(x,y)] \geq \alpha/4$*

Proof. We will proof it by induction. Base case: $|x-y| = \alpha/2 \Rightarrow d(x,y) = \alpha/2$ holds from the claim. To calculate general case, we have:

$$\begin{aligned} \mathbb{E}[d(x,y)] &= \mathbb{E}[d(x-1,y) + \Pr[f(x-1)_{\tau(x)} = f(y)_{\tau(x)}] - \Pr[f(x-1)_{\tau(x)} \neq f(y)_{\tau(x)}]] \\ &= \mathbb{E}[d(x-1,y)] + 1 - 2\Pr[f(x-1)_{\tau(x)} \neq f(y)_{\tau(x)}] \end{aligned}$$

For this we first need to calculate how many matching coordinates were flipped in the previous $\alpha/2$ rounds. Assume that in the previous $\alpha/2$ rounds, we flipped k matching coordinates: $k = |\{f(x-1-\alpha/2)_{\tau(x-i)} = f(y)_{\tau(x-i)}); i \in [\alpha/2]\}|$ and $(\alpha/2 - k)$ un-matching coordinates $(f(x-1-\alpha/2)_{\tau(x-i)} \neq f(y)_{\tau(x-i)})$. So we have $d(x-1,y) = d(x-1-\alpha/2) + k - (\alpha/2 - k)$ or $k = \frac{d(x-1,y) - d(x-1-\alpha/2,y) + \alpha/2}{2}$.

Now we can calculate $\Pr[f(x-1)_{\tau(x)} \neq f(y)_{\tau(x)}]$. From $f(x-1)$ to $f(x)$, there are $d(x-1,y)$ unmatched coordinates, with k coordinates which have been flipped in the previous $\alpha/2$ rounds. Thus:

$$\begin{aligned} \Pr[f(x-1)_{\tau(x)} \neq f(y)_{\tau(x)}] &= \frac{d(x-1,y) - k}{\alpha/2} \\ &= \frac{d(x-1,y) - \frac{d(x-1,y) - d(x-1-\alpha/2,y) + \alpha/2}{2}}{\alpha/2} \\ &= \frac{d(x-1,y) + d(x-1-\alpha/2,y) - \alpha/2}{\alpha} \end{aligned}$$

And our expectation:

$$\begin{aligned}
\mathbb{E}[d(x,y)] &= \mathbb{E}[d(x-1,y)] + 1 - 2\Pr[f(x-1)_{\tau(x)} \neq f(y)_{\tau(x)}] \\
&= \mathbb{E}[d(x-1,y)] + 1 - 2\frac{\mathbb{E}[d(x-1,y)] + \mathbb{E}[d(x-1-\alpha/2,y)] - \alpha/2}{\alpha} \\
&= \mathbb{E}[d(x-1,y)] + 2\left(1 - \frac{\mathbb{E}[d(x-1,y)] + \mathbb{E}[d(x-1-\alpha/2,y)]}{\alpha}\right) \\
&= 2 + \frac{(\alpha-2)\mathbb{E}[d(x-1,y)] - 2\mathbb{E}[d(x-1-\alpha/2,y)]}{\alpha} \\
&\geq 2 + \frac{(\alpha-2)\mathbb{E}[d(x-1,y)] - 2(\mathbb{E}[d(x-1,y)] + \alpha/2)}{\alpha} \\
&= 1 + \frac{(\alpha-4)}{\alpha}\mathbb{E}[d(x-1,y)] \\
&\geq 1 + \frac{(\alpha-4)}{\alpha} \cdot \frac{\alpha}{4} = 1 + \frac{\alpha}{4} - 1 = \alpha/4
\end{aligned}$$

Last step is by induction. □

Observation 6. *This bound is not tight - it is possible to show tighter bounds around $\alpha/2$ with a bit more work - but for our purpose it makes no asymptotic difference so $\alpha/4$ is good enough.*

4.2 Algorithm

We propose the following algorithm for our $\ell 1$ LSH:

1. Prepare $K = 100\log(N)$ random $\alpha/2$ distance-preserving α-bit mappings denoted $f^1..f^K$ (as specified above) with $\alpha = 8cr$. We define $F : [N] \to \{0,1\}^{K\alpha}$ to be the concatenation of $f^1..f^K$
2. Map each $x \in S \subseteq [N]^d$ to $x' \in S' \subseteq \{0,1\}^{\alpha K d}$ using mapping T, where $T : [N]^d \to \{0,1\}^{\alpha K d}$ is defined to be the concatenation of $F(x_1)..F(x_d)$
3. Hash all $x' \in S'$ using Pagh construction

On each NNS query $y \in [N]^d$, we will map $y' = T(y)$, look for nearest-neighbors for y' in S' using Pagh, and for each such neighbor $x' \in S'$, compare the original $\ell 1$ distance $\|x-y\|_1$.

4.3 Analysis

4.3.1 Correctness

To prove our main theorem, we would like to show that close points remain close always and all far points will remain far with constant probability. As a direct result - Pagh data-structure guarantee of finding close point always with sub-linear expected running time will hold.

Claim 7. *For all $x,y \in [N]^d$ s.t. $\|x-y\|_1 \leq r$, $\|T(x)-T(y)\|_{Ham} = K\|x-y\|_1 \leq Kr$ with probability 1*

Proof. Since $\forall i \in [d] : |x_i - y_i| \leq r < 4cr$, then $\|T(x)-T(y)\|_{Ham} = K \cdot \|x-y\|_1 \leq Kr$ □

Claim 8. *For all $x,y \in [N]^d$ s.t. $\|x-y\|_1 \geq cr$, $\|T(x)-T(y)\|_{Ham} \geq Kcr$ (with 90% probability)*

Proof. If $\forall i \in [d] : |x_i - y_i| \leq 4cr$, then $\|T(x)-T(y)\|_{Ham} = K \cdot \|x-y\|_1 \geq Kcr$.
Otherwise according to Lemma 9 (assuming i is dimension with higher distance):

$$\|T(x)-T(y)\|_{Ham} \geq \|F(x_i)-F(y_i)\|_{Ham} \geq Kcr$$

□

Lemma 9. $\Pr[\forall a, b \in [N] \ s.t. \ |a-b| \geq 4cr, \ \|F(a) - F(b)\|_{Ham} \geq Kcr] \geq 0.9$

Proof. By Lemma 5, for each $\alpha/2$ distance-preserving function $f^j, j \in [K]$, and for any a, b s.t. $|a-b| \geq 4cr$:

$$\mathbb{E}[\|f^j(a) - f^j(b)\|_{Ham}] \geq \alpha/4 = 2cr$$

And since the functions are chosen iid, by applying Chernoff:

$$\begin{aligned}\Pr[\|F(a) - F(b)\|_{Ham} < Kcr] &= \Pr[\sum_{j=1}^{K} \|f^j(a) - f^j(b)\|_{Ham}/8cr < 1/2 \cdot K/4] \\ &\leq exp\left(-\frac{K}{32}\right) < exp\left(-3\log N\right) < 0.1/N\end{aligned}$$

Therefore, by union bound on all distances $|a-b|$:

$$\Pr[\exists a, b \in [N] \text{ s.t. } |a-b| \geq 4cr \text{ and } \|F(a) - F(b)\|_{Ham} < Kcr] < N \cdot 0.1/N < 0.1$$

□

Observation 10. *We can amplify such probability to* $1 - \delta$ *with another* $\log(1/\delta)$ *factor in* K.

4.3.2 Complexity

As this is a theoretical paper, we were mainly concerned with asymptotic complexity. The final dimension is:

$$d' = \alpha K d = \mathcal{O}(r \cdot \log N \cdot d)$$

The space required is the Hashing space plus the space to store the K random functions ($\mathcal{O}(rN\log N)$):

$$S = \mathcal{O}(r \cdot \log N \cdot (N + d \cdot n^{1+\rho}) \text{ bits}$$

Note that the space to store the random functions can be absorbed asymptotically whenever $N = \mathcal{O}(d \cdot n^{1+\rho})$ which is generally the case.

To make time analysis comparable to other algorithms, we will assume word-length $w = \log N$:

$$T = \mathcal{O}((r \cdot \log N \cdot d)/w \cdot n^{\rho}) = \mathcal{O}(r \cdot d \cdot n^{\rho})$$

4.4 Replacing the r factor with d

Note that our final dimension is linear in r, which works well with Pagh's Hamming LSH which is optimal when $r = \Theta(\log n)$ or $r = \mathcal{O}(1)$.

It is possible, however, to replace the r factor with d and pay some constant factor in denominator and exponent (trade-off) dividing each coordinate by $\frac{\epsilon r}{d}$ (for some constant $\epsilon < c - 1$) and round them to the nearest integer.
As a result, we may "lose" $\frac{\epsilon r}{d}$ distance per coordinate and in total ϵr, so cr-far points might only be

$(c-\epsilon)r$-far, and our updated parameters:

$$c' = c - \epsilon$$
$$K = 100\log(\frac{Nd}{\epsilon r})$$
$$\alpha = \frac{8cd}{\epsilon}$$
$$r' = K\frac{d}{\epsilon}$$

Final dimension becomes:

$$d' = \mathcal{O}(d^2\log(\frac{Nd}{\epsilon r})/\epsilon)$$

And parameter ρ:

$$\rho' \approx \rho + \epsilon/c$$

5 Conclusion and Future Work

In this project we analyzed the Pagh construction for guaranteed LSH close-point finding with expected efficiency and extended such findings efficiently to $\ell 1$ by introducing random distance-preserving mappings.

Some of the interesting features about our construction:

- It can find all close points.
- It can be extended to $\ell 2$ by $\ell 2 \to \ell 1$ embedding.
- While it is for range-bounded $\ell 1$ ($[N]^d$), it is dynamic in the sense that if the range increases, the random functions can be adjusted for the increased range without needing to rehash all points.

There are several ways that our suggested algorithm can be further improved:

- While the space to store K random distance preserving functions can generally be absorbed by the model, it might be possible to choose pseudo-random distance-preserving functions to further improve space-efficiency in the worst-case.
- Furthermore - it might be possible to find "covering" set distance-preserving functions which would:
  1. Save space as above.
  2. Remove dependency on $\log(N)$.
  3. Remove range restriction (so would work on $\mathbf{N}^d$).
- Last - some of our bounds can be tightened to for constant improvements in final dimension.

6 Citations

- Pagh, Rasmus. "Locality-sensitive hashing without false negatives." arXiv preprint arXiv:1507.03225 (2015).
- Indyk, Piotr, and Rajeev Motwani. "Approximate nearest neighbors: towards removing the curse of dimensionality." Proceedings of the thirtieth annual ACM symposium on Theory of computing. ACM, 1998.

- Datar, Mayur, et al. "Locality-sensitive hashing scheme based on p-stable distributions." Proceedings of the twentieth annual symposium on Computational geometry. ACM, 2004.