

## Lecture 10: Local Sensitive Hash

2025.3.27

Lecturer: 丁虎

Scribe: 王浩宇, 莫官霖

这是一类近似近邻查询 (Approx-NN) 的方法, 其主要思想是设计 Hash 函数, 将点集  $P$  中的点放入不同的 Bucket 中, 同时使得距离越近的点, 其哈希值碰撞的概率越大。最后通过增加 hash 的次数来提升成功的概率 [3]。

[3] 是最早提出 LSH 框架的论文, 首次实现了高维空间下的亚线性近似最近邻搜索, 为后续研究奠定基础。此后, [2] 一文针对欧几里得距离 (L2) 设计了一种基于 p-stable 分布的 LSH 方案, 是最常用的实用 LSH 算法之一, 适用于图像、音频等连续向量数据。本篇讲义聚焦于 [1] 一文, 这是 LSH 理论的里程碑式论文, 提出了  $(r, R, \alpha, \beta)$ -Sensitive Hash 的形式定义, 并构造出几乎最优的 LSH 结构, 显著改善了查询复杂度。该文同时是 LSH 与高维数据结构研究的基础。

## 1 近似近邻查询

**Definition 1.1** ( $(r, R)$ -近似近邻查询). 输入集合  $P \subset \mathbb{R}^d$  和一个点  $q \in \mathbb{R}^d$ , 我们记  $\text{dist}(q, p) = \min_{p \in P} \|q - p\|$ 。  $(r, R)$ -近似近邻查询要求

1. 如果  $\text{dist}(q, P) \leq r$ , 返回  $u \in P$  使得  $\|q - u\| \leq R$ .
2. 如果  $\text{dist}(q, P) > R$ , 输出 “ $\text{dist}(q, P) > r$ ” .
3. 如果  $r < \text{dist}(q, P) \leq R$ , 返回上面两者中任意一种。

如果  $R = r$  那即为精确的  $NS(r)$ . 通常我们会考虑  $R = (1 + \epsilon)r$  的情况。这个时候我们可以称上面的问题为  $(1 + \epsilon)$ -近似近邻查询。

在已知  $\text{dist}(q, P)$  的上下界为  $[a, b]$  的情况下, 建立集合  $U = [a, (1 + \epsilon)a, \dots, (1 + \epsilon)^{\log_{1 + \epsilon} \frac{b}{a}} a]$ , 然后对于  $r$  的选择, 我们在  $U$  上作 Binary Search, 这样我们可以对通过至多  $\log_{1 + \epsilon} \frac{b}{a}$  次 NS(近邻查询) 来实现  $(1 + \epsilon)$ -近似近邻查询。

我们用  $B(q, r)$  表示以  $q$  为球心,  $r$  为半径的欧几里得距离的球体。用  $N(\vec{0}, I_d)$  表示  $d$  维标准正态分布。  $U([a, b])$  表示区间  $[a, b]$  上的均匀随机分布。

**Definition 1.2.**  $(r, R, \alpha, \beta)$ -Sensitive Hash [1] 给定  $r < R, 1 > \alpha > \beta > 0$ , 我们称  $F$  是一个  $(r, R, \alpha, \beta)$ -Sensitive Hash 映射集合, 如果对于  $\forall u, q \in \mathbb{R}^d$ , 随机取  $h \in F$  满足

1. 如果  $u \in B(q, r)$ , 那么  $\Pr[h(u) = h(q)] \geq \alpha$ .
2. 如果  $u \notin B(q, R)$ , 那么  $\Pr[h(u) = h(q)] \leq \beta$ .

下面我们给一个  $F$  的构造方法。考虑这样的  $F_T = \{h | h(u) = \lfloor \frac{\langle u, \vec{v}_h \rangle + t_h}{T} \rfloor\}$ , 其中  $T$  是一个待确定的值,  $\vec{v}_h \sim N(\vec{0}, I_d), t_h \sim U([0, T])$ . 我们有如下定理

**Theorem 1.3.** 任给  $r, \epsilon > 0, 1 > \alpha > \beta > 0$ , 且  $\frac{\log \frac{1}{\alpha}}{\log \frac{1}{\beta}} \leq \frac{1}{1+\epsilon}$ , 则一定存在  $T > 0$ , 使得  $F_T$  是  $(r, (1+\epsilon)r, \alpha, \beta)$ -Sensitive 的。

注意到上面的  $h$  是将  $\mathbb{R}^d \rightarrow \mathbb{Z}$ , 下面我们讨论如果将该 Hash 过程的成功率提高。为此我们考虑一个新的哈希函数  $g : \mathbb{R}^d \rightarrow \mathbb{Z}^k$ , 其中  $g = (h_1, h_2, \dots, h_k), h_i \in F$ . 这样构成的函数族我们称为  $G(F, k) = g = (h_1, \dots, h_k | h_i \in F)$ .

那么我们有下面的结论

**Theorem 1.4.** 令  $\rho = \frac{\log \frac{1}{\alpha}}{\log \frac{1}{\beta}} \leq \frac{1}{1+\epsilon}$ ,  $k = \log_{\frac{1}{\beta}} n$ ,  $\tau = 2n^\rho = o(n)$ . 随机从  $G(F, k)$  中取出  $g_1, g_2, \dots, g_\tau$  其对应  $\tau$  个哈希 bucket  $H_1, \dots, H_\tau$ , 我们有对于  $\forall q \in \mathbb{R}^d$ , 满足下列两个条件的概率  $\geq \frac{3}{5}$ :

1. 如果  $\exists u \in P$ , 使得  $\|u - q\| \leq r$ , 则  $\exists j$  使得  $g_j(u) = g_j(q)$ .
2. 如果  $\forall u \in P, \|u - q\| > (1+\epsilon)r$ , 则在  $H_1, H_2, \dots, H_\tau$  中与  $q$  发生冲突的点  $\leq 4\tau$ .

*Proof.* 先证明 2.

$$\begin{aligned}
& \forall u \in P, \|u - q\| > (1+\epsilon)r \\
& \Rightarrow \forall g \in G(F, k), \Pr[g(u) = g(q)] \leq \beta^k = \frac{1}{n} \\
& \Rightarrow \forall H_j, \mathbb{E}[\text{发生冲突个数}] \leq 1 \\
& \Rightarrow \text{在 } H_1 \sim H_\tau \text{ 中发生冲突总数期望} \leq \tau \\
& \Rightarrow \Pr[\text{发生冲突个数} \leq 4\tau] \geq \frac{3}{4}
\end{aligned}$$

再证明 1.

$$\begin{aligned}
& \|u - q\| \leq r \\
& \Rightarrow \forall q, \Pr[g(u) = g(q)] \geq \alpha^k = n^{-\rho} \\
& \Rightarrow H_1 \sim H_\tau \text{ 至少发生一次冲突的概率} \geq 1 - (1 - n^{-\rho})^\tau = 1 - (1 - n^{-\rho})^{2n^\rho} \geq 1 - \frac{1}{e^2} > \frac{4}{5}.
\end{aligned}$$

□

由该定理，我们可以将  $P$  中的点分为 3 个类：

1.  $\|u - q\| \leq r$ , 此时发生冲突个数  $\geq 1$
2.  $\|u - q\| > (1 + \epsilon)r$ , 此时发生冲突个数  $\leq 4\tau$
3.  $r < \|u - q\| \leq (1 + \epsilon)r$ , 此时发生冲突个数可能很多也可能很少，所以返回任意情况

这样我们只需要检查前面  $\leq 4\tau + 1$  个冲突即可。

该算法复杂的分析：

1. Construction Time  $O(\tau \cdot n \cdot k \cdot d) = O(n^{1+\frac{1}{1+\epsilon}} d \log n) \leq O(n^2 d)$ .
2. Space  $O(nd + \tau \cdot k \cdot n) = O(nd + n^{1+\frac{1}{1+\epsilon}} \log n)$ .
3. Query Time  $O(\tau \cdot k \cdot d + (4\tau + 1)d) = O(n^{\frac{1}{1+\epsilon}} \log n \cdot d) < nd$ .

## References

- [1] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 459–468. IEEE, 2006.
- [2] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG '04, page 253–262, New York, NY, USA, 2004. Association for Computing Machinery.
- [3] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA, 1998. Association for Computing Machinery.