Sub-linear Memory Sketches for Near Neighbor Search on Streaming Data with RACE: Supplementary

Anonymous Authors[1]

1. Techniques

We obtain our results by combining recent advances in locality-sensitive hashing (LSH)-based estimation with standard compressed sensing techniques. This section contains a high-level overview of our strategy to solve the nearest-neighbor problem.

LSH-based kernel estimators: The array-of-counts estimator (ACE) is an unbiased estimator for kernel functions. Our first step is to use ACE to estimate arbitrary linear combinations of kernels. We get sharp estimates of these linear combinations by averaging over multiple ACEs. We call this structure a RACE because it consists of *repeated* ACEs. The number of repetitions needed for a good estimate does not depend on N, the dataset size. Once we have sharp estimates of the measurements, we apply standard compressed sensing techniques.

Compressed sensing: A central result of compressed sensing is that a v sparse vector of length N can be recovered from $O(v \log N/v)$ linear combinations of its elements. The coefficients of the linear combination are defined by the measurement matrix. In this context, our measurements are of the vector $\mathbf{s}(q) \in \mathbb{R}^N$, where the i^{th} component of $\mathbf{s}(q)$ is the kernel evaluation $k(x_i, q)$. Since we are using LSH kernels, $k(x_i, q)$ is the LSH collision probability of q and $x_i \in \mathcal{D}$. That is, $\mathbf{s}_i(q) = p(x_i, q)$. We will use the terms LSH kernel value and collision probability interchangeably.

We use one RACE structure to estimate each compressed sensing measurement. Each RACE gets a different set of linear combination coefficients, and we choose the coefficients so that they describe a valid measurement matrix. By applying sparse recovery to the set of estimated measurements, we can approximate the kernel evaluations (LSH collision probabilities) between q and each element in the dataset. As explained in the "Intuition" section of the main text, these kernel evaluations are sufficient to perform near neighbor search. Assuming sparsity, the sketch is sublinear because each RACE requires a constant amount of memory and we only need to use $O(v \log N/v)$ RACEs.

Query-dependent guarantees: To find neighbors for a query (q), we recover the kernel values ($\mathbf{s}(q)$) and return the indices with the largest values as the identities of the near-neighbors. This process will not succeed if $\mathbf{s}(q)$ is not *sparse*. Sparsity is the reason for our query-dependent assumptions. To find the nearest indices, we need $\mathbf{s}(q)$ to have few large elements. The geometric interpretation is that most elements in the dataset are not near-neighbors of q. We show that well-established notions of near-neighbor stability (Beyer et al., 1999) are equivalent to weak sparsity conditions on $\mathbf{s}(q)$, allowing us to express our algorithm in terms of near neighbor stability. This result connects sparsity - a compressed sensing idea - with the difficulty of the near-neighbor search problem. We can analyze a large class of geometric data assumptions by interpreting them as sparsity conditions.

If the dataset already satisfies our sparsity condition, then we proceed directly to recovery. If not, we can force $\mathbf{s}(q)$ to be sparse by raising the kernel function $k(x_i, q)$ to a power K. This modification decreases the bandwidth of the kernel, letting us locate near-neighbors at a finer resolution. RACE can accommodate this idea by using standard methods for amplifying a LSH family. Specifically, we construct the LSH function from K independent realizations of an LSH family. The result is a new LSH function with the collision probability $p(x, q)^K$. However, there is a price - the size of each ACE repetition grows larger.

Reduce near-neighbor to compressed sensing recovery: Using compressed sensing, we can estimate the kernel values within an ϵ additive tolerance. To solve the near-neighbor problem, we make ϵ small enough to distinguish between near-neighbors and the rest of the dataset. The value of ϵ depends on K. Increasing K makes $\mathbf{s}(q)$ sparse but also increases the amount of storage required for the sketch. Therefore, we want K to be *just large enough*. By balancing the sparsity requirement with the memory, we introduce a query-dependent multiplicative $O(N^b)$ factor for the sketch size. This term is sub-linear ($b < 1$) when $\mathbf{s}(q)$ is sufficiently sparse or, equivalently, when q is a stable query. Our sketch requires $O(N^b \log^3(N))$ bits, where b depends on the query stability.

2. Theory

In this section, we provide a detailed explanation of the theory with complete proofs.

2.1. Estimation of Compressed Sensing Measurements

In this section, our goal is to prove that the RACE algorithm can estimate the compressed sensing measurements of $\mathbf{s}(q)$, the vector of kernel evaluations. We begin by constructing a modified version of ACE that can estimate any linear combination of $\mathbf{s}(q)$ components. Then, we derive a variance bound on this estimator and apply the median of means technique.

We can estimate the linear combination by incrementing the ACE array using the linear combination coefficients. Suppose we are given a sequence of linear combination coefficients $\{r_i\}_{i=1}^N$. The original ACE estimator simply increments the array A at index $L(x_i)$ by 1. We will use the notation $\mathbb{1}_i$ to refer to the indicator function $\mathbb{1}_{L(x_i)=L(q)}$. That is, $\mathbb{1}_i$ is 1 when the query collides with element x_i from the dataset. For the original ACE algorithm, $A[L(q)] = \sum_{x_i\in\mathcal{D}} \mathbb{1}_i$. In our case, we increment $A[L(x_i)]$ by r_i and therefore we have $A[L(q)] = \sum_{x_i\in\mathcal{D}} r_i\, \mathbb{1}_i$. The expectation of this estimator is the linear combination of LSH kernels (collision probabilities).

Theorem 3. *Given a dataset $\mathcal{D}$, K independent LSH functions $l(\cdot)$ and any choice of constants $r_i \in \mathbb{R}$, RACE can estimate a linear combination of $\mathbf{s}_i(q) = p(x_i, q)^K$ with the following variance bound.*

$$\mathbb{E}[A[L(q)]] = \sum_{x_i\in\mathcal{D}} r_i p(x_i,q)^K \tag{1}$$

$$\mathrm{var}(A[l(q)]) \leq |\tilde{\mathbf{s}}(q)|_1^2 \tag{2}$$

where $L(\cdot)$ is formed by concatenating the K copies of $l(\cdot)$ and $\tilde{\mathbf{s}}_i(q) = \sqrt{\mathbf{s}_i(q)}$.

Proof. For the sake of presentation, let $Z = A[L(q)]$.

Expectation: The count in the array can be written as

$$Z = \sum_{x_i\in\mathcal{D}} r_i\, \mathbb{1}_i$$

By linearity of the expectation operator

$$\mathbb{E}[Z] \ = \sum_{x_i\in\mathcal{D}} r_i\, \mathbb{E}[\mathbb{1}_i]$$

$\mathbb{E}[\mathbb{1}_i]$ is simply the collision probability of L, thus

$$\mathbb{E}[Z] \ = \sum_{x_i\in\mathcal{D}} r_i p(x_i,q)^K$$

Variance: The variance is bounded above by the second moment. The second moment of this estimator can be written as

$$\mathbb{E}[Z^2] \ = \sum_{x_i\in\mathcal{D}}\sum_{x_j\in\mathcal{D}} r_i r_j\, \mathbb{E}[\mathbb{1}_i\, \mathbb{1}_j]$$

Use the Cauchy-Schwarz inequality to bound $\mathbb{E}[\mathbb{1}_i\, \mathbb{1}_j] \leq \sqrt{\mathbb{E}[\mathbb{1}_i]}\sqrt{\mathbb{E}[\mathbb{1}_j]}$. Thus

$$\mathbb{E}[Z^2] \ \leq \sum_{x_i\in\mathcal{D}}\sum_{x_j\in\mathcal{D}} r_i r_j \sqrt{p(x_i,q)^K}\sqrt{p(x_j,q)^K}$$

$$= \left(\sum_{x_i\in\mathcal{D}} r_i \sqrt{p(x_i,q)^K)}\right)^2$$

For our analysis, we will assume that $r_i \in [-1,1]$. This is valid because we can always scale the compressed sensing matrix so that it is true. Then the bound becomes

$$\mathrm{var}(Z) \leq \left(\sum_{x_i\in\mathcal{D}} \sqrt{p(x_i,q)^K)}\right)^2 = |\tilde{\mathbf{s}}(q)|_1^2$$

□

Using Theorem 3 and the median-of-means (MoM) technique, we can obtain an arbitrarily close estimate of each compressed sensing measurement $y_i(q)$. Suppose we independently repeat the ACE estimator and compute the MoM estimate from the repetitions. Let $\hat{y}_i(q)$ be the MoM estimate of $y_i(q)$ computed from a set of independent ACE repetitions of $A[l(q)]$. Then we have a pointwise bound on the error for each $y_i(q)$.

Lemma 1. *For any $\epsilon > 0$ and given*

$$O\Big(\frac{|\tilde{\mathbf{s}}(q)|_1^2}{\epsilon^2}\log\Big(\frac{1}{\delta}\Big)\Big)$$

independent ACE repetitions, we have the following bound for the MoM estimator

$$y_i(q) - \epsilon \leq \hat{y}_i(q) \leq y_i(q) + \epsilon \tag{3}$$

with probability $1-\delta$ for any query q.

Proof. For presentation, we will drop the index and write $\hat{y}_i(q)$ as $\hat{y}$ where the context is clear. We use a very common proof technique with the median-of-means estimator $\hat{y}$. With probability at least $1-\delta$ and n independent realizations of the random variable, we can estimate the mean with MoM so that

$$\Pr\left[|\hat{y} - y| \leq \sqrt{32\frac{\mathrm{var}(\hat{y})}{n}\log\left(\frac{1}{\delta}\right)}\right] \geq 1-\delta$$

We can substitute the variance bound from Theorem 3 in for $\mathrm{var}(\hat{y})$ without changing the validity of the inequality. To have the lemma, we need $|\hat{y} - y| \leq \epsilon$. We will choose n to be large enough that

$$\sqrt{32\frac{|\tilde{\mathbf{s}}(q)|_1^2}{n}\log\left(\frac{1}{\delta}\right)} \leq \epsilon$$

Therefore, we need n ACE repetitions, where n is

$$n = 32\frac{|\tilde{\mathbf{s}}(q)|_1^2}{\epsilon^2}\log\left(\frac{1}{\delta}\right)$$

□

Lemma 1 only works for one of the compressed sensing measurements. To ensure that all M of the measurements obey this bound with probability $1-\delta$, we apply the probability union bound to get Theorem 4. Note that the multiplicative M factor comes from the fact that we are using ACE to estimate M different measurements.

Theorem 4. *For any $\epsilon > 0$ and given $O\left(M\frac{|\tilde{\mathbf{s}}(q)|_1^2}{\epsilon^2}\log\left(\frac{M}{\delta}\right)\right)$ independent ACE repetitions, we have the following bound for each of the M measurements with probability $1-\delta$ for any query q*

$$y_i(q) - \epsilon \leq \hat{y}_i(q) \leq y_i(q) + \epsilon \quad (4)$$

Proof. We want all measurements to succeed with probability $1-\delta$. The probability union bound states that if δ_i is the failure probability for measurement i, then the overall failure probability is smaller than $\sum_{i=1}^{M}\delta_i$. We would like this probability to be smaller than δ, so we put $\delta_i = \frac{\delta}{M}$ for each RACE estimator. By Lemma 1, we need

$$32\frac{|\tilde{\mathbf{s}}(q)|_1^2}{\epsilon^2}\log\left(\frac{M}{\delta}\right)$$

repetitions for each measurement. There are M measurements, so we need

$$32M\frac{|\tilde{\mathbf{s}}(q)|_1^2}{\epsilon^2}\log\left(\frac{M}{\delta}\right)$$

repetitions in total. □

2.2. Query-Dependent Sparsity Conditions

Before we can discuss compressed recovery of $\mathbf{s}(q)$, we need to limit our analysis to vectors $\mathbf{s}(q)$ that are sparse. In this section, we introduce a permissive way to bound the sparsity of $\mathbf{s}(q)$ for our analysis. Our bounds are forgiving in the sense that we assume as little underlying sparsity as possible - with stronger assumptions, you can get better bounds. We also connect sparsity with near-neighbor stability. We analyze these conditions in the context of compressed sensing and computational geometry.

We need bounds for $|\mathbf{s}(q)|_1$ and $|\tilde{\mathbf{s}}(q)|_1$. Our vector $\mathbf{s}(q)$ has three properties that make these bounds possible. First, the collision probabilities are bounded: $p(x_i, q) \in [0, 1]$. Second, increasing K causes each element of $\mathbf{s}(q)$ to decrease, since $s(q)_i = p(x_i, q)^K$. Third we may choose K to be as large as necessary. Therefore, we can force $|\mathbf{s}(q)|_1$ to be arbitrarily small by choosing K sufficiently large. However, each ACE estimator requires $O(r^K \log N)$ memory where r is the number of hash codes that L can return. Therefore, we want K to be *just large enough* so that we do not increase the space too much.

We will analyze sparsity under the *equidistant* assumption. Under this assumption, all points other than the v nearest neighbors are equidistant to the query. This is a relatively weak way to describe sparsity, but we still get an acceptable dependence of K on N. Stronger assumptions require smaller K and therefore less space. To choose K, we need a good way to characterize the sparsity of $\mathbf{s}(q)$. We begin by defining two query-dependent values Δ and B. Δ is related to the stability of the near-neighbor query and B is related to sparsity.

Δ-Stable Queries: We want a parameter that measures the difficulty of the query. For the v-nearest neighbor problem, let x_v and x_{v+1} be the v^{th} and $(v+1)^{\text{th}}$ nearest neighbors, respectively. Using the same notation as before, let Δ be defined as

$$\Delta = \frac{p(x_{v+1}, q)}{p(x_v, q)} \quad (5)$$

Δ governs the stability of the nearest neighbor query. It is a measure of the gap between the near-neighbors and the rest of the dataset. If $\Delta = 1$, then the v^{th} and $(v+1)^{\text{th}}$ neighbors are the same distance away. In this case, it is impossible to tell the difference between them. If $\Delta \approx 0$, then it means that neighbors $v+1$, $v+2$, ... are all very far away. Our definition of Δ is similar to the definition of an ϵ-unstable query. In fact, we can express a Δ-stable query as an ϵ-unstable query by finding the distances that correspond to $p(x_{v+1}, q)$ and $p(x_v, q)$. This is possible because the collision probability is a monotone function of distance.

B-Bounded Queries: We want a flexible way to bound the sum:

$$|\mathbf{s}(q)|_1 = \sum_{x_i \in \mathcal{D}} p(x_i, q)^K$$

For convenience, we will suppose that the elements x_i are sorted based on their distance from the query. This is not necessary - it just simplifies the presentation. When we write $p(x_i, q)$, we mean that x_i is the i^{th} near neighbor of the query. We will use the notation $p_i = p(x_i, q)$. Define a constant B as

$$B = \sum_{i=v+1}^{N} \frac{\tilde{s}_i}{\tilde{s}_{v+1}} = \sum_{i=v+1}^{N} \sqrt{\frac{p_i^K}{p_{v+1}^K}} \tag{6}$$

B is a query-dependent value that measures the sparsity of $\mathbf{s}$. It bounds the size of the tail entries of $\mathbf{s}$. A bound on B implies a bound on $|\mathbf{s}|_1$ and $|\tilde{\mathbf{s}}|_1$.

Lemma 2. $|\mathbf{s}|_1 \leq |\tilde{\mathbf{s}}|_1$ *and* $|\tilde{\mathbf{s}}|_1 \leq v + B\sqrt{p_{v+1}^K}$

Proof. It is easy to see that $\mathbf{s}_i \leq \sqrt{\mathbf{s}_i}$ because $0 \leq \mathbf{s}_i \leq 1$. For the second inequality, break the summation for $|\tilde{\mathbf{s}}|_1$ into two components:

$$|\tilde{\mathbf{s}}|_1 = \sum_{i=1}^{v} \sqrt{p_i^K} + \sum_{i=v+1}^{N} \sqrt{p_i^K}$$

The first term corresponds to the nearest v points in the dataset. The second term corresponds to the rest of the dataset. For the first term, we will use the trivial bound that $\sqrt{p_i^K} \leq 1$. For the second term,

$$\sum_{i=v+1}^{N} \sqrt{p_i^K} = \sum_{i=v+1}^{N} \sqrt{p_i^K} \frac{\sqrt{p_{v+1}^K}}{\sqrt{p_{v+1}^K}} = \sqrt{p_{v+1}^K} B$$

□

Using B and Δ, we can find a value of K that bounds $|\mathbf{s}(q)|_1$ and $|\tilde{\mathbf{s}}(q)|_1$.

Theorem 5. *Given a query q and query-dependent parameters B and Δ, if*

$$K = \left\lceil 2\frac{\log B}{\log \frac{1}{\Delta}} \right\rceil$$

then

$$p(x_v, q)^K \geq \sum_{i=v+1}^{N} p(x_i, q)^K$$

and we have the bounds

$$|\mathbf{s}(q)|_1 \leq v + 1$$

$$|\tilde{\mathbf{s}}(q)|_1 \leq v + 1$$

Proof. Start with the inequality

$$\sqrt{p_v^K} \geq \sum_{i=v+1}^{N} \sqrt{p_i^K}$$

Now divide both sides by $\sqrt{p_{v+1}^K}$.

$$\left(\sqrt{\frac{p_v}{p_{v+1}}}\right)^K \geq \sum_{i=v+1}^{N} \frac{\sqrt{p_i^K}}{\sqrt{p_{v+1}^K}}$$

Observe that the left side is equal to $\Delta^{-K/2}$ and the right side to B. Thus we have

$$\Delta^{-K/2} \geq B$$

We use the smallest integer K that satisfies this inequality

$$K \geq 2\frac{\log B}{\log \frac{1}{\Delta}}$$

To bound the L1 norms, observe that $p_v \leq 1$ and that the summation $\leq p_v$. To get the final inequality in the theorem, start with the inequality in terms of p_i rather than $\sqrt{p_i}$ and follow the same steps. The result will be $K \geq \log B / -\log \Delta$. Our choice of K also satisfies this inequality (the $|\tilde{\mathbf{s}}|_1$ bound is more restrictive). □

Equidistant Assumption: If we wanted to make the bound in Lemma 2 or K in Theorem 5 as large as possible, we would set $B = N - v$. To have $B = N - v$, we need $p_{v+1} = p_{v+2} = ... = p_N$. Since the collision probability is a monotone function of distance, this condition means that all non-neighbors are equidistant from the query. The rationale behind our equidistant assumption is that it represents the worst possible Δ-stable query. We are also motivated by (Beyer et al., 1999), who also identify the equidistant case as a particularly hard instance of the near-neighbor problem. When $B = N$, the vector is minimally sparse and we rely on K to do all of the work. Theorem 5 works for any distribution of points, so we could repeat the analysis with $B < O(N)$ under stronger sparsity assumptions. However, our sketch is sublinear for stable queries even under the equidistant assumption. In the next section, we will see that the memory required by our sketch depends on p_v and Δ.

2.3. Reduce Near-Neighbor to Compressed Recovery

In this section, we will combine all of our results to create a near-neighbor sketch under the equidistant assumption. For simplicity, we restrict our attention to the CMS. The main challenge is to ensure that the kernel values recovered by our algorithm are within ϵ of the true ones.

There are two sources of error: the CMS recovery and the RACE estimator. We will use ϵ_C for the CMS error and ϵ_E for the estimator error. The value of ϵ_C is determined by the CMS recovery guarantee while ϵ_E is determined by Theorem 4. We will use $M = O\left(\frac{1}{\epsilon_C}\log\frac{N}{\delta}\right)$ measurements for the CMS. Each measurement can differ from the true value by up to ϵ_E. This situation is known as *measurement noise*. For the CMS, measurement noise propagates as-is to the recovered output values. This happens because the CMS recovery procedure returns one of the cell values as its estimate for each component of $\hat{\mathbf{s}}$. If the cell values in $\widehat{\text{CMS}}$ deviate from the true CMS values by $\leq \epsilon_E$, then the output of $\widehat{\text{CMS}}$ deviates from the true output by $\leq \epsilon_E$.

$$s_i(q) - \epsilon_E \leq \hat{s}_i(q) \leq s_i(q) + \epsilon_E + \epsilon_C |\mathbf{s}(q)|_1 \quad (7)$$

By choosing appropriate values for ϵ_C and ϵ_E, we obtain a concise statement for the pointwise recovery guarantee of our estimated CMS.

Theorem 6. *We require*

$$O\left(\frac{|\tilde{\mathbf{s}}(q)|_1^2 |\mathbf{s}(q)|_1}{\epsilon^3} \log\left(\frac{|\mathbf{s}(q)|_1}{\epsilon\delta} \log\left(\frac{N}{\delta}\right)\right) \log\left(\frac{N}{\delta}\right)\right)$$

ACE estimates to recover $\hat{\mathbf{s}}(q)$ *such that*

$$s_i(q) - \frac{\epsilon}{2} \leq \hat{s}_i(q) \leq s_i(q) + \frac{\epsilon}{2} \quad (8)$$

with probability $1 - \delta$.

Proof. Put $\epsilon_E = \epsilon/4$ and $\epsilon_C = \epsilon/4|\mathbf{s}(q)|_1$. Then the error is

$$\mathbf{s}_i(q) - \frac{\epsilon}{4} \leq \hat{\mathbf{s}}_i(q) \leq \mathbf{s}_i(q) + \frac{\epsilon}{2}$$

To have $\epsilon_C = \epsilon/4|\mathbf{s}(q)|_1$ with probability $1 - \delta_C$ we must have

$$M = O\left(\frac{|\mathbf{s}|_1}{\epsilon} \log\left(\frac{N}{\delta_C}\right)\right)$$

CMS measurements. For all measurements to have $\epsilon_E = \epsilon/4$ with probability $1 - \delta_E$, we must have

$$O\Big(M \frac{|\tilde{\mathbf{s}}(q)|_1^2}{\epsilon^2} \log\Big(\frac{M}{\delta_E}\Big)\Big)$$

ACE repetitions. The first requirement comes from the CMS guarantee. The second comes from Theorem 4.For both of these conditions to hold with probability $1 - \delta$, we use the union bound and put $\delta_C = \delta_E = \delta/2$. To obtain the result, substitute M into the second requirement. We can safely ignore the constant factors inside the logarithm because they are constant additive terms. $\square$

2.4. Near-Neighbor Sketch Size

To differentiate between the v and the $(v+1)^{\text{th}}$ elements of $\mathbf{s}$, we need to have $\epsilon < s_v - s_{v+1}$. This means that we can identify the v nearest neighbors by setting $\epsilon < p_v^K - p_{v+1}^K$.

Lemma 3. *Put* $K = \lceil 2\frac{\log N}{\log \frac{1}{\Delta}} \rceil$. *Then*

$$\epsilon = p_v^K - p_{v+1}^K = O\left(N^{2\frac{\log p_v}{\log \frac{1}{\Delta}}}\right) \quad (9)$$

Proof.

$$p_v^K - p_{v+1}^K = p_v^K (1 - \Delta^K)$$

Substitute K:

$$p_v^{2\frac{\log N}{\log \frac{1}{\Delta}}} (1 - \Delta^{2\frac{\log N}{\log \frac{1}{\Delta}}})$$

Recall the identity

$$x^{\log y / \log x} = y$$

First address the Δ^K term. Observe that

$$\Delta^{2\frac{\log N}{\log \frac{1}{\Delta}}} = \left(\Delta^{\frac{\log N}{\log \Delta}}\right)^{-2} = N^{-2}$$

Next address the p_v^K term. Observe that

$$p_v^{2\frac{\log N}{\log \frac{1}{\Delta}}} = \left(p_v^{\frac{\log N}{\log p_v}}\right)^{\frac{\log p_v}{\log \frac{1}{\Delta}}} = N^{2\frac{\log p_v}{\log \frac{1}{\Delta}}}$$

Put these together:

$$p_v^K - p_{v+1}^K = N^{2\frac{\log p_v}{\log \frac{1}{\Delta}}} (1 - N^{-2})$$

Since $(1 - N^{-2}) \to 1$ with N, we have that

$$p_v^K - p_{v+1}^K = O\left(N^{2\frac{\log p_v}{\log \frac{1}{\Delta}}}\right)$$

This result may seem strange, but remember that $p_v < 1$. Therefore, $\epsilon = p_v^K - p_{v+1}^K$ is a negative power of N. Also, we restrict Δ to the range where $2\frac{\log N}{\log \frac{1}{\Delta}} > 1$. Otherwise, $K = 1$ and the lemma is unnecessary. $\square$

We are finally ready to state our main result. We assume the equidistant case and put $K = \lceil 2\frac{\log N}{\log \frac{1}{\Delta}} \rceil$ according to Theorem 5 and we plug the result into Theorem 6.

Theorem 7. *Given a query* q, *a dataset* $\mathcal{D}$ *and an LSH function that can output* r *different values, we can construct a sketch to solve the* v*-nearest neighbor problem with probability* $1 - \delta$ *in size*

$$O\left(v^3 N^{b_1} \log\left(\frac{v}{\delta} N^{b_2} \log \frac{N}{\delta}\right) \log\left(\frac{N}{\delta}\right) \log N\right)$$

bits, where

$$b_1 = \frac{6|\log p_v| + 2\log r}{\log \frac{1}{\Delta}}$$

$$b_2 = \frac{2|\log p_v|}{\log \frac{1}{\Delta}}$$

x_v *is the* v^{th} *nearest neighbor of* q *in* $\mathcal{D}$, x_{v+1} *is the* $(v+1)^{th}$ *nearest neighbor of* q *in* $\mathcal{D}$, *and* $\Delta = \frac{p_{v+1}}{p_v}$.

Proof. Assume the equidistant case and put $K = \lceil 2\frac{\log N}{\log \frac{1}{\Delta}} \rceil$. Then Theorem 6 states that we require

$$O\left(\frac{(v+1)^3}{\epsilon^3}\log\left(\frac{v+1}{\epsilon\delta}\log\left(\frac{N}{\delta}\right)\right)\log\left(\frac{N}{\delta}\right)\right)$$

ACE repetitions. The $(v+1)^3$ terms came from the bounds in Theorem 5. Put $\epsilon = p_v^K + p_{v+1}^K$ and use Lemma 3 to get that $\epsilon^{-1} = N^{b_2}$. The requirement is now

$$O\left(v^3 N^{3b_2}\log\left(N^{b_2}\frac{v}{\delta}\log\left(\frac{N}{\delta}\right)\right)\log\left(\frac{N}{\delta}\right)\right)$$

ACE repetitions. Each ACE repetition requires $r^K \log N$ bits. Apply the same trick as in Lemma 3 to get that

$$r^K = r^{2\frac{\log N}{\log \frac{1}{\Delta}}} = N^{2\frac{\log r}{\log \frac{1}{\Delta}}}$$

The total requirement is therefore

$$O\left(v^3 N^{b_1}\log\left(N^{b_2}\frac{v}{\delta}\log\left(\frac{N}{\delta}\right)\right)\log\left(\frac{N}{\delta}\right)\right)\log N$$

bits. □

Our main theorem from the main text (Corollary 7.1) is a substantially simplified version of Theorem 7.

Corollary 7.1. *It is possible to construct a sketch that solves the exact v-nearest neighbor problem with probability $1-\delta$ using $O\left(N^b \log^3(N)\right)$ bits, where*

$$b = \frac{6|\log p_v| + 2\log r}{\log \frac{1}{\Delta}}$$

Here, r is the range of the LSH function, and p_v is the collision probability of the v^{th} nearest neighbor with the query.

Proof. The proof involves expanding Theorem 7 and dropping terms. Observe that

$$\log\left(N^{b_1}\frac{v}{\delta}\log\left(\frac{N}{\delta}\right)\right)$$

$$= b_1 \log N + \log v - \log\delta + \log(\log N - \log\delta)$$

This is multiplied by $v^3 N^{b_2}$, $(\log N - \log\delta)$ and $\log N$. The N^{b_2} term asymptotically dominates the expression. We are left with

$$O\left(v^3 N^b \log^3 N\right)$$

□

References

Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. When is "nearest neighbor" meaningful? In *International conference on database theory*, pp. 217–235. Springer, 1999.