ECS 189A Sublinear Algorithms for Big Data **Fall 2024**

Lecture 17: Count-Min Sketch & Count Sketch

Lecturer: Jasper Lee *Scribe: Xiaohang Cheng*

1 Streaming Definition Review

Each stream token pair (j, c) denotes that we intend to increment the frequency of domain element f_j by the count c. Let $\mathbf{f}$ be the frequency vector over universe $[n]$ and assume $|f_j| \leq m$ space (analogous to "stream length" bounded by m).

Models:

- "Cash register": $c > 0$
- "Turnstile": unrestricted c (some of the frequencies can be negative)

Today's Problem: Estimate f_j. Related to finding "heavy hitters" or frequent elements in the stream. The details are left as an exercise to the readers. (HW7 Problem 6)

Today's Algorithm:

Count-Min Sketch

- Works for the "Cash Register model"
- Weaker approximation guarantee
- Lower Space Usage

Count Sketch

- Works for the "Turnstile model"
- Stronger approximation guarantee
- Higher Space Usage

2 Count-Min Sketch

Algorithm 17.1

1. $C[1, \cdots, t][1, \cdots, k] \leftarrow 0$; $k = \frac{2}{\epsilon}, t = \log_2 \frac{1}{\delta}$
2. Sample t i.i.d. functions $h_1, \cdots, h_t : [n] \to [k]$ from a 2-wise independent hash family.
3. Repeat for each token (j, c):
   for each $i = 1$ to t:
   $C[i][h_i(j)]$ += c;
4. Query $\hat{f}_j$, return $\min_i C[i][h_i(j)]$.

Note that the space usage of the 2-D array C is $O\left(\frac{1}{\epsilon}(\log \frac{1}{\delta})(\log m)\right)$. Because the dimension of C is $t \times k$. We also assume that each f_j is bounded by m, which needs $\log m$ bits. The

space usage of the hash functions $h_1, \cdots, h_t$ is $O\left(\log \frac{1}{\delta} \cdot (\log n + \log \frac{1}{\epsilon})\right)$. Because there are t many of them. Assume each of them is from a 2-wise independent hash family from $[n] \to [k]$. Therefore, it requires $\max\{\log n, \log \frac{1}{\epsilon}\}$ or simply $O(\log n + \log \frac{1}{\epsilon})$ space. Thus, the total space usage is $O\left(\log \frac{1}{\delta}(\frac{1}{\epsilon}(\log m) + \log n)\right)$. Note that the $\log \frac{1}{\epsilon}$ term can be dismissed since other terms will be dominant in the product due to $\frac{1}{\epsilon} \gg \log \frac{1}{\epsilon}$ and $\log m > 1$.

Intuition for the Algorithm:
Consider a $t \times k$ matrix. We have t copies of sketches. For each sketch, draw a hash function randomly from the hash family and get its j-th entry. Then, retrieve this particular element as the count.

Observation:
If $c > 0$ the "Cash Register" model, for every $i \in [t]$, $C[i][h_i(j)] \geq f_j$.

As we can see, we can only overcount due to possible collisions. Therefore, it makes sense to take the minimum of all counts, and we attempt to bound the probability that the minimum overcounts (possibly by a lot).

Theorem 17.2. *Consider an arbitrary stream in the "Cash Register" model and an arbitrary $j \in [n]$ query made at the end. Run Algorithm 17.1 w.p. $\geq 1 - \delta$, we have $f_j \leq \hat{f}_k \leq f_j + \epsilon \|\mathbf{f}_{-j}\|_1$, where $\mathbf{f}_{-j}$ (dimension $n-1$) is f dropping the j-th coordinate.*

Proof. Consider the event that the i-th hash function makes a and j collide. Denote this event by the indicator variable $Y_{ia} = \mathbb{1}_{\{h_i(a)=h_i(j)\}}$. Let the error of $C[i][h_i(j)]$ for f_j be $X_i = \sum_{a \neq j} f_a \cdot Y_{ia}$.
We first analyze the expectation of X_i. By linearity,

$$\mathbb{E}(X_i) = \sum_{a \neq j} f_a \cdot \mathbb{E}(Y_{ia})$$

Note that $\mathbb{E}(Y_{ia})$ is the probability of the i-th hash function make a and j collide. By the universality of hash family (Lecture 16), we have $\mathbb{E}(Y_{ia}) \leq \frac{1}{k}$. Therefore, we have

$$\sum_{a \neq j} f_a \cdot \mathbb{E}(Y_{ia}) \leq \sum_{a \neq j} \frac{f_a}{k} = \frac{\|\mathbf{f}_{-j}\|_1}{k} = \frac{\epsilon \|\mathbf{f}_{-j}\|_1}{2}$$

By Markov's Inequality, we get

$$\mathbb{P}(X_i \geq \epsilon \|\mathbf{f}_{-j}\|_1) \leq \frac{\mathbb{E}(X_i)}{\epsilon \|\mathbf{f}_{-j}\|_1} \leq \frac{1}{2}$$

Observe that if $\hat{f}_j$ is large, then $C[i][h_i(j)]$ is large for all i. Therefore,

$$\begin{aligned}
\mathbb{P}\left(\hat{f}_j = \min_i C[i][h_i(j)] \geq f_j + \epsilon \|\mathbf{f}_{-j}\|_1\right) &= \mathbb{P}\left(\min_i X_i \geq \epsilon \|\mathbf{f}_{-j}\|_1\right) \\
&= \mathbb{P}\left(\forall i, X_i \geq \epsilon \|\mathbf{f}_{-j}\|_1\right) \\
&\leq \frac{1}{2^t} \\
&= \delta
\end{aligned}$$

□

One can also do a Count-median sketch by replacing min in the query with median operation for "Turnstile" streaming model with the same guarantee except for the lower bound. The analysis is left as an exercise to the readers.

3 Count Sketch

In this section, we will stick with a constant success probability $\frac{2}{3}$ for each fixed query. The success probability can actually be boosted to $1-\delta$. The analysis is left as an exercise to the readers. (HW7 Problem 1(b))

Algorithm 17.3

1. $C[1, \cdots, k] \leftarrow 0$; $k = \frac{3}{\epsilon^2}$
2. Sample t i.i.d. functions $h_1, \cdots, h_t : [n] \to [k]$ from a 2-wise independent hash family.
3. Choose a random $g \colon [n] \to \{\pm 1\}$ from a 2-wise independent hash family.
4. Repeat for each token (j, c) :
   $C[h(j)] \mathrel{+}= c \cdot g(j)$;
5. Query j, return $\hat{f}_j = C[h(j)] \cdot g(j)$.

Now we analyze the space usage. Since the dimension of the array C is k, it requires $O(\frac{1}{\epsilon^2} \log m)$ space. The space usage of the hash functions is $O(\log n + \log \frac{1}{\epsilon})$ for a similar argument of the space usage of the Count-min Sketch algorithm. Therefore, the total space usage is $O(\frac{1}{\epsilon^2} \log m + \log n)$ for a constant probability of success. If we want to boost it to $1-\delta$, it requires $O(\log \frac{1}{\delta})$ multiplicative overhead in space and time complexity. The analysis is left as an exercise to the readers. (HW7 Problem 1(b))

Theorem 17.4. *Consider an arbitrary stream in the "Turnstile" model and an arbitrary $j \in [n]$ query made at the end. Run Algorithm 17.3 w.p. $\geq 1-\delta$, we have $|\hat{f}_j - f_j| \leq \epsilon \|\mathbf{f}_{-j}\|_2$, where $\mathbf{f}_{-j}$ (dimension $n-1$) is f dropping the j-th coordinate.*

Proof. Consider the event that the i-th hash function makes a and j collide. Denote this event by the indicator variable $Y_a = \mathbb{1}_{\{h(a)=h(j)\}}$. Let $\hat{f}_j = g(j) \cdot C[h(j)] = g(j) \cdot \sum_a f_a g(a) Y_a$, where f_a is the frequency of a and $g(a)$ is a random sign. Pulling out the case where $i = j$, we have

$$\sum_a f_a g(a) Y_a = f_j + \sum_{a \neq j} f_a g(j) g(a) Y_a$$

We first analyze the expectation of $\hat{f}_j$. We have

$$\mathbb{E}(\hat{f}_j) = f_j + \sum_{a \neq j} f_a \mathbb{E}\left[g(j) g(a) Y_a\right]$$

Since g and h are independent,

$$\mathbb{E}\left[g(j) g(a) Y_a\right] = \mathbb{E}\left[g(j) g(a)\right] \mathbb{E}(Y_a)$$

Since g is drawn from a 2-wise independent hash family,

$$\begin{aligned} &= \underbrace{\mathbb{E}[g(j)]}_{0}\underbrace{\mathbb{E}[g(a)]}_{0}\mathbb{E}(Y_a) \\ &= 0 \end{aligned}$$

Therefore, we have $\mathbb{E}(\hat{f}_j) = f_j$.
Next, we analyze the variance of $\hat{f}_j$. By definition,

$$\mathrm{Var}(\hat{f}_j) = \mathbb{E}[(\hat{f}_j - \mathbb{E}(\hat{f}_j)^2] = \mathbb{E}[(\hat{f}_j - f_j)^2] = \mathbb{E}\left[\sum_{a\neq j} f_a \cancel{g(j)} g(a) Y_a\right]^2$$

$$= \mathbb{E}\left[\sum_{a\neq j} f_a^2 \cancel{g^2(a)} Y_a^2 + \sum_{\substack{a\neq b \\ a\neq j \\ b\neq j}} f_a f_b g(a) g(b) Y_a Y_b\right]$$

Note that $\mathbb{E}(Y_a) \leq \frac{1}{k}$ and $\mathbb{E}[g(a)g(b)Y_aY_b] = \underbrace{\mathbb{E}[g(a)]\mathbb{E}[g(b)]}_{0}\mathbb{E}[Y_aY_b] = 0$.

Therefore,

$$\mathrm{Var}(\hat{f}_j) \leq \frac{1}{k} \cdot \sum_{a\neq j} f_a^2 = \frac{\|\mathbf{f}_{-j}\|_2^2}{k}$$

Finally, by Chebyshev's inequality,

$$\mathbb{P}\left(|\hat{f}_j - f_j| \geq \epsilon \|\mathbf{f}_{-j}\|_2\right) \leq \frac{\mathrm{Var}(\hat{f}_j)}{\epsilon^2 \|\mathbf{f}_{-j}\|^2} \leq \frac{1}{3}$$

□

Remark: For all $x \in \mathbb{R}^n$, we have $\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\|x\|_2$.

Proof.

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2} \leq \sum_{i=1}^{n} \sqrt{|x_i|^2} = \sum_{i=1}^{n} |x_i| = \|x\|_1$$

□

The other inequality is left as an exercise to the readers. As a result, the Count Sketch gives us a better guarantee than Count-Min Sketch.