Algorithms for Big Data (Fall 25)

Lecture 8

CountSketch & Sketching Applications

Ali Vakilian (vakilian@vt.edu)

Virginia Tech

Basic Hashing Idea

Heavy Hitters Problem: Find all items i such that $f_i \geq m/k$.

- Let $b_1, \ldots, b_k$ be the k heavy hitters (at most k)
- Suppose we pick a hash function $h: [n] \rightarrow [ck]$ for some $c > 1$
- h maps the heavy hitters into different buckets (k balls into ck bins)
- Then, ideally, we would like to use the count of items in each bucket as an estimate for the frequency of one heavy hitters.

Repeating this idea with independent hashes improves the estimate

CountMin Sketch [Cormode-Muthukrishnan]

- d pairwise independent hash functions $h_1, \dots, h_d$; each $[n] \to [w]$
- Equivalently, a table C with d rows and w columns.
- Store one counter per entry in the table, which keep the aggregate frequency of items mapped to the entry by the corresponding hash function. $C[\ell, s]$ is the counter for bucket s in hash function h_ℓ.
- Let $f \in \mathbb{R}^n$ be the final frequency vector. For $\ell \in [d], s \in [w]$,

$$C[\ell, s] = \sum_{i : h_\ell(i) = s} f_i$$

- For every $\ell \in [d]$, $C[\ell, h_\ell(i)]$ is an over-estimate of f_i.
- We have d such estimate, how good is the quality of best of them?

CountMin Sketch in Streaming

- Each of d estimates for f_i is overcounting its frequency
- Picking the minimum such estimate is reasonable.

CountMin Sketch (stream):

let $h_1, \ldots, h_k$ be pairwise independent hash functions from $[n] \to [w]$

foreach item $e_t = (i_t, \Delta_t)$ in the stream **do**:
  for $\ell = 1$ to d **do**:
    $C[\ell, h_\ell(i_t)] \leftarrow C[\ell, h_\ell(i_t)] + \Delta_t$

//frequency estimates
foreach $i \in [n]$, set $\tilde{f}_i = \min_{\ell \in [d]} C[\ell, h_\ell(i)]$

CountMin Sketch: Main Property

Theorem. Consider strict turnstile streaming (i.e., always $f \geq 0$). Let $d = \Omega\left(\log\frac{1}{\delta}\right)$ and $w > \frac{2}{\varepsilon}$. Then, for any fixed $i \in [n]$, $\boldsymbol{f_i \leq \tilde{f}_i}$, and

$$\mathbf{Pr}\left[\tilde{\boldsymbol{f}}_{\boldsymbol{i}} \geq \boldsymbol{f}_{\boldsymbol{i}} + \boldsymbol{\varepsilon}\|\boldsymbol{f}\|_{\mathbf{1}}\right] \leq \boldsymbol{\delta}$$

- Unlike Misra-Gries, CountMin overestimates.
- Items are not stored (can be recovered via queries).
- Handles deletion (works in strict turnstile model)
- Space complexity: $O(\frac{\log\frac{1}{\delta}}{\varepsilon} \cdot \log m)$ bits

CountMin: Analysis

- Consider an item i and fix a row ℓ.
- Define $Z_\ell = C[\ell, h_\ell(i)]$ the value of counter in row ℓ that i is hashed to.

$$\mathbb{E}[Z_\ell] = f_i + \sum_{j \neq i} \Pr[h_\ell(j) = h_\ell(i)] \cdot f_j$$

$$= f_i + \sum_{j \neq i} \frac{1}{w} \cdot f_j \qquad \text{// pairwise independence of } h_\ell$$

$$\leq f_i + \varepsilon \|f\|_1/2 \qquad \text{// } w > 2/\varepsilon$$

Applying Markov, $\Pr[Z_\ell - f_i \geq \varepsilon \|f\|_1] \leq 1/2$

Since d hash functions are independent,

$$\Pr[\min_{\ell \in [d]} Z_\ell \geq f_i + \varepsilon \|f\|_1] \leq \frac{1}{2^d} \leq \delta \qquad \text{// } d = \Omega\left(\log \tfrac{1}{\delta}\right)$$

CountMin Sketch

Space Complexity: $O(\frac{1}{\varepsilon}\log n \log m)$ bits

Theorem. Consider strict turnstile streaming (i.e., always $f \geq 0$). Let $d = \Omega\left(\log\frac{1}{\delta}\right)$ and $w > \frac{2}{\varepsilon}$. Then, for any fixed $i \in [n]$, $\boldsymbol{f_i \leq \tilde{f}_i}$, and

$$\mathbf{Pr}\left[\tilde{\boldsymbol{f}}_i \geq \boldsymbol{f}_i + \boldsymbol{\varepsilon}\|\boldsymbol{f}\|_1\right] \leq \boldsymbol{\delta}$$

- Setting $\delta = 1/n^2$, a CountMin with $O(\log n)$ rows and $O(1/\varepsilon)$ columns, for every $i \in [n]$,

$$\Pr\left[\tilde{f}_i > f_i + \varepsilon\|f\|_1\right] \leq 1/n^2$$

- By union bound over all n items, with probability $\geq 1 - 1/n$, for all $i \in [n]$

$$\tilde{f}_i \leq f_i + \varepsilon\|f\|_1$$

CountMin is a Linear Sketch

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} = \begin{bmatrix} f_4 \\ f_2 + f_3 \\ f_1 + f_5 \end{bmatrix}$$

Item **1** is hashed to bucket 3

Item **2** is hashed to bucket 2

Item **3** is hashed to bucket 2

Item **4** is hashed to bucket 1

Item **5** is hashed to bucket 3

hash function h_i as a Matrix-Vector Multiplication

$$\Pi_{w \times m} \, \boldsymbol{f}_{m \times 1}$$

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} = \begin{bmatrix} f_4 \\ f_2 + f_3 \\ f_1 + f_5 \\ f_1 + f_5 \\ f_2 + f_4 \\ f_3 \\ f_1 + f_2 \\ f_3 + f_4 \\ f_5 \end{bmatrix}$$

CountMin as a Matrix-Vector Multiplication

$$\Pi_{(k \cdot w) \times m} \, \boldsymbol{f}_{m \times 1}$$

CountSketch

- Simialr to CountMin, keeps track of a table of $k \times w$ counters
- Inspired by AMS sketch, assign u.a.r signs $\{-1, +1\}$ to items
- Counters can get even negative

CountSketch (stream):

let $h_1, \ldots, h_d$ be pairwise independent hash functions from $[n] \rightarrow [w]$

let $g_1, \ldots, g_d$ be pairwise independent hash functions from $[n] \rightarrow \{-1, +1\}$

foreach item $e_t = (i_t, \Delta_t)$ in the stream **do**:

for $\ell = 1$ to k **do**:

$$C[\ell, h_\ell(i_t)] \leftarrow C[\ell, h_\ell(i_t)] + g_\ell(i_t) \cdot \Delta_t$$

//frequency estimates

foreach $i \in [n]$, set $\tilde{f}_i = \operatorname{median}_{\ell \in [k]}\{g_\ell(i) \cdot C[\ell, h_\ell(i)]\}$

Why CountSkecth is a Linear Sketch?

$$\begin{bmatrix} 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} = \begin{bmatrix} -f_4 \\ -f_2 + f_3 \\ f_1 + f_5 \end{bmatrix}$$

Item **1** is hashed to bucket **3**

Item **2** is hashed to bucket **2** **× -1**

Item **3** is hashed to bucket **2**

Item **4** is hashed to bucket **1** **× -1**

Item **5** is hashed to bucket **3**

$$\begin{bmatrix} 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} \times \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} = \begin{bmatrix} -f_4 \\ -f_2 + f_3 \\ f_1 + f_5 \\ -f_1 + f_5 \\ f_2 - f_4 \\ -f_3 \\ f_1 - f_2 \\ -f_3 + f_4 \\ -f_5 \end{bmatrix}$$

hash function h_i as a Matrix-Vector Multiplication

$$\Pi_{w \times m} \, \boldsymbol{f}_{m \times 1}$$

CountMin as a Matrix-Vector Multiplication

$$\Pi_{(k \cdot w) \times m} \, \boldsymbol{f}_{m \times 1}$$

CountSketch: Main Property

Theorem. Consider strict turnstile streaming (i.e., always $f \geq 0$). Let $d = \Omega\left(\log \frac{1}{\delta}\right)$ and $w > \frac{3}{\varepsilon^2}$. Then, for any fixed $i \in [n]$, $\mathbb{E}\left[\tilde{f}_i\right] = f_i$, and

$$\Pr\left[\left|\tilde{f}_i - f_i\right| \geq \varepsilon \|f\|_2\right] \leq \delta$$

Comparison to CountMin

- Error is w.r.t. $\|f\|_2$ instead of $\|f\|_1$. Note $\|f\|_2 \leq \|f\|_1$, and in some cases $\|f\|_2 \ll \|f\|_1$
- **Space complexity:** $O(\frac{1}{\varepsilon^2} \cdot \log n)$ bits

CountSketch Analysis

- Consider an item i and fix a row ℓ.
- Define $Z_\ell = g_\ell(i) C[\ell, h_\ell(i)]$ the value of counter in row ℓ that i is hashed to.

For $j \in [n]$ let Y_j be the indicator r.v. that is 1 if $h_\ell(i) = h_\ell(j)$; i.e., i and j collide in h_ℓ

$\mathbb{E}[Y_j] = \mathbb{E}[Y_j^2] = 1/w$ from pairwise independence of h_ℓ

$Z_\ell = g_\ell(i) C[\ell, h_\ell(i)] = g_\ell(i) f_i + \sum_{j \neq i} g_\ell(i) f_j Y_j$

$\mathbb{E}[Z_\ell] = f_i + \sum_{j \neq i} \mathbb{E}[g_\ell(i) g_\ell(j) Y_j] \cdot f_j$

$= f_i$ // pairwise independence of g_ℓ

Since $\mathbb{E}[g_\ell(i) g_\ell(j) Y_j] = \mathbb{E}[g_\ell(i) g_\ell(j)]\, \mathbb{E}[Y_j] = 0$

CountSketch Analysis: Variance

- Define $Z_\ell = g_\ell(i) C[\ell, h_\ell(i)]$ the value of counter in row ℓ that i is hashed to.

For $j \in [n]$ let Y_j be the indicator r.v. that is 1 if $h_\ell(i) = h_\ell(j)$; i.e., i and j collide in h_ℓ

$\mathbb{E}[Y_j] = \mathbb{E}[Y_j^2] = 1/w$ from pairwise independence of h_ℓ

$$= \mathbb{E}[(Z_\ell - f_i)^2]$$

$$= \mathbb{E}\left[\left(\sum_{j \neq i} g_\ell(i) g_\ell(j) Y_j f_j\right)^2\right]$$

$$= \mathbb{E}\left[\sum_{j \neq i} g_\ell(i)^2 g_\ell(j)^2 Y_j^2 f_j^2 + \sum_{j, j' \neq i} g_\ell(i)^2 g_\ell(j) g_\ell(j') Y_j Y_{j'} f_j f_{j'}\right]$$

$$= \sum_{j \neq i} f_j^2 \, \mathbb{E}[Y_j^2]$$

$$\leq \|f\|_2^2 / w$$

Using Chebyshev, $\Pr[|Z_\ell - f_i| \geq \varepsilon \|f\|_2] \leq \frac{\mathrm{Var}(Z_\ell)}{\varepsilon^2 \|f\|_2^2} \leq \frac{1}{\varepsilon^2 w} \leq 1/3$

CountSketch: Concentration

Using Chebyshev, $\Pr[|Z_\ell - f_i| \geq \varepsilon\|f\|_2] \leq \frac{\mathrm{Var}(Z_\ell)}{\varepsilon^2\|f\|_2^2} \leq \frac{1}{\varepsilon^2 w} \leq 1/3$

Then, by Chernoff bound,

$$\Pr[|\mathrm{median}\{Z_1, \ldots, Z_d\} - f_i| \geq \varepsilon\|f\|_2] \leq e^{-\Omega(d)} \leq \delta$$

Applications of CountMin & CountSketch

Heavy Hitters: Point Queries

Heavy Hitters Problem. Find all items i such that $f_i \geq \alpha \|f\|_1$ for $\alpha \in (0,1]$.

- output: any i such that $f_i \geq (\alpha - \varepsilon) \cdot \|f\|_1$

First Attempt:

Using CountMin, go over each $i \in [n]$ and check if $\tilde{f}_i \geq (\alpha - \varepsilon) \cdot \|f\|_1$

What is the computation time?

- To compute the frequency of each item, requires $O(\log n)$ time.
- Overall, $O(n \log n)$ time.

Can we solve it in sublinear time in n?

Dyadic (Hierarchical) Search

Idea. Hierarchical data structure of CountMin sketches

- Number of levels is $L = \lceil \log n \rceil + 1$. (level 0 to level $L - 1$)
- At level $\ell \in \{0, \ldots, \ell - 1\}$
  - There are 2^ℓ disjoint intervals (or buckets), each of length $B_\ell = \lceil n/2^L \rceil$.
  - Interval (or bucket) index of item i is

$$b_\ell(i) = 1 + \left\lfloor \frac{i-1}{B_\ell} \right\rfloor \in \{1, \ldots, 2^\ell\}$$

We maintain one CountMin per each level. In each level ℓ, we have 2^ℓ **super-items** corresponding to each of the buckets in this level.

$$\forall \mathbf{b} \in [2^\ell], \quad e_{\ell,\mathbf{b}} = \{j \mid b_\ell(j) = b\} \text{ and } \mathbf{freq}(e_{\ell,\mathbf{b}}) = \sum_{j:b_\ell(j)=\mathbf{b}} f_j$$

Dyadic (Hierarchical) Search

How big is the CountMin in level ℓ?

- Set the width as $w = O(1/\varepsilon)$
- Set the #rows as $d = O\left(L \log\left(\frac{1}{\delta}\right)\right)$ (for overall $\leq \delta$ failure probability)

The overall number of counters is $\boldsymbol{L \times O\left(\frac{L}{\varepsilon} \log\left(\frac{1}{\delta}\right)\right) = O(\varepsilon^{-1} \cdot \log n \cdot \log 1/\delta)}$

How to update the CountMin (CM_ℓ) when a new item (i, Δ) arrives?

- For each level $\ell = 0, \ldots, L-1$:
  - Compute the bucket id $\mathbf{b} = b_\ell(i)$
  - Update CM_ℓ for super-item $e_{\ell,\mathbf{b}}$

CountMin updates during the stream

Dyadic (Hierarchical) Search

Heavy Hitters Problem. Find all items i such that $f_i \geq \alpha \|f\|_1$ for $\alpha \in (0,1]$.

- output: any i such that $f_i \geq (\alpha - \varepsilon) \cdot \|f\|_1$

How to find Heavy Hitters?

How large is **candidate set**?

- At any level ℓ,
  - $\leq \frac{1}{\alpha}$ super-items expand
- Similarly, $\leq \frac{2}{\alpha}$ in **candidate set**

Finally, verify the frequency of those in **candidate set** by CM_L

Hierarchical CountMin Query:

$Q = \{(0,1)\}$

while Q is non-empty

  pop $(\ell, \mathbf{b})$ from Q

  query CM_ℓ to estimate $\mathbf{freq}(\boldsymbol{e}_{\ell,\mathbf{b}})$

  if $\mathbf{freq}(\boldsymbol{e}_{\ell,\mathbf{b}}) < \alpha \|f\|_1$, **prune** (do not expand)

  elseif $\ell = L - 1$, add b to the **candidate set**

  else push its 2 children $(\ell + 1, 2b - 1)$ & $(\ell + 1, 2b)$ to Q

return candidate set

Dyadic (Hierarchical) Search

How to find Heavy Hitters?

How large is **candidate set**?

- At any level ℓ,
  - $\leq \frac{1}{\alpha}$ super-items expand
- Similarly, $\leq \frac{1}{\alpha}$ in **candidate set**

Finally, verify the frequency of those in **candidate set** by CM_L

Hierarchical CountMin Query:
$Q = \{(0,1)\}$
while Q is non-empty
    pop $(\ell, \mathbf{b})$ from Q
    query CM_ℓ to estimate $\mathbf{freq}(\boldsymbol{e}_{\ell,\mathbf{b}})$
    if $\mathbf{freq}\left(\boldsymbol{e}_{\ell,\mathbf{b}}\right) < \alpha \|f\|_1$, **prune** (do not expand)
    elseif $\ell = L - 1$, add b to the **candidate set**
    else push its 2 children $(\ell + 1, 2b - 1)$ & $(\ell + 1, 2b)$ to Q
return candidate set

How many estimate of $\mathbf{freq}(\boldsymbol{e}_{\ell,\mathbf{b}})$ is computed?

- $O(1/\alpha)$ per row; overall $O\left(\frac{L}{\alpha}\right) = O(\frac{1}{\alpha} \cdot \log n)$ many estimates is computed
- Overall runtime is $O(\frac{1}{\alpha} \cdot \log n)$ (improving upon the naïve $O(n \log n)$)

Range Queries

Range queries: given $i, j \in [n]$, output $\sum_{i \leq \ell \leq j} f_\ell$

Examples

- In networking, database, or discretization of a signal value

There are $\Omega(n^2)$ potential range queries. A naïve way requires $O(i - j)$ which could be as large as $O(n)$ queries to the CountMins Sketch.

Can we do better?

In **HW 1**: You'll answer this question.

Sparse Recovery

Sparsity is an important theme in optimization/algorithms/modeling

- Data is often explicitly sparse.

  Examples: graphs, matrices, vectors, documents (as word vectors)

- Data is often *implicitly* sparse: in a different representation the data is explicitly sparse.

  Examples: signals/images, topics, ...

Algorithmic advantage

- To improve performance (speed, quality, memory, ...)
- Find sparse representation to reveal information about data

  Examples: topics in documents, frequencies in Fourier analysis

Sparse Recovery

Problem. Given a vector/signal $x \in \mathbb{R}^n$, find a sparse vector z approximating x.

More formally, given $x \in \mathbb{R}^n$ and integer $k \geq 1$, find z s.t. z has at most k non-zeros ($\|z\|_0 \leq k$) s.t. $\|z - x\|_p$ is minimized for some $p \geq 1$.

What is the optimal offline solution?

How to solve in strict turnstile streaming for $p = 2$ using $\tilde{O}(k)$ space?