Sketching Techniques for Hinge Loss

March 28, 2022
1. Introduction

2. Sketching for Hinge Loss
Introduction
**Problem Statement**

Given $n$ data points $x_1, \ldots, x_n \in \mathbb{R}^d$, a label vector $y \in \mathbb{R}^n$ and $f$ being a classification function

Let $x^* = \arg \min_{x \in \mathbb{R}^d} \sum_{i=1}^{n} f(\langle x_i, x \rangle \cdot y_i)$ and $F(x) = \sum_{i=1}^{n} f(\langle x_i, x \rangle \cdot y_i)$,

**Goal**: Find a subset of $x'_1, \ldots, x'_r$ points along with corresponding weights $w_1, \ldots, w_r$ s.t. for some small $k$, we have:

$$F(x') \leq k \cdot F(x^*)$$

where $x' = \arg \min_{x \in \mathbb{R}^d} \sum_{j=1}^{r} w_j \cdot f(\langle x'_j, x \rangle \cdot y'_j)$. 
**Coresets** are small subsets of data, often achieved by subsampling from a properly designed distribution.

Deficiencies of coreset constructions:

1. Rely on regularization to obtain small coresets,
2. Usually require random access to data,
3. Require at least two passes over the data (one for calculating/approximating probabilities and the other for subsampling and collecting data),
4. Usually only work in insertion streams, where the data is presented row by row.
Initialize the data matrix $A \in \mathbb{R}^{n \times d}$ to be a all-zero matrix, where $n$ is large. We have a sequence of updates $(i, j, v)$, each causing a change $A_{ij} = A_{ij} + v$. Updates $v$ of $A$ can be negative.

This is referred to as the turnstile model, which is the most flexible dynamic setting.

A linear sketch is an algorithm which computes $SA$ as $A$ is updated, where $S \in \mathbb{R}^{m \times n} (m \ll n)$.

Linear sketches support operations such as addition, subtracting and scaling of databases $A_j$ efficiently, since $SA = S \sum_j \alpha_j A_j = \sum_j \alpha_j SA_j$. 
Advantages of using Oblivious Sketching:

1. Works well with highly unstructured and arbitrarily distributed data,
2. Allows efficient applications in a single pass of data,
3. Applicable to high velocity streams, since any update can be calculated in $O(1)$ time,
4. Linear sketches support several useful operations on the data.
Oblivious Sketching for Logistic Loss

First data oblivious sketch for logistic regression [A. Munteanu, S. Omlor, D. P. Woodruff (2021)]:

• The sketch can be computed in input sparsity time in one pass over a turnstile data stream,
• It reduces the size of a $d$-dimensional data set from $n$ to $\text{poly}(\mu d \log n)$ weighted points (where $\mu$ is a parameter capturing the complexity of compressing the data),
• It obtains a $O(\log n)$ approximation to the original problem,
• Can obtain a $O(1)$ approximation with slight modifications.
OVERVIEW OF THE ALGORITHM

In logistic regression we are given a data matrix \( Q \in \mathbb{R}^{n \times d} \) and a label vector \( L \in \{-1, 1\}^n \). Let data matrix \( A \in \mathbb{R}^{n \times d} \) where each row \( a_i \) for \( i \in [n] \) is defined as \( a_i := -l_i q_i \). Our goal is to find \( x \in \mathbb{R}^d \) that minimizes the logistic loss given by

\[
f(Ax) = \sum_{i \in [n]} \ln(1 + \exp(a_i x))
\]
1. Logistic regression loss function can be approximated as 
   \[ f(Ax) \approx G^+(Ax) + f((Ax)^-) \] which can be handled separately 
   while losing only an approximation factor of 2, where we define:
   - \( G^+(y) := \sum_{y_i \geq 0} y_i \) to be the sum of positive entries of \( y \),
   - \((Ax)^-\) the vector \( Ax \) with all positive entries replaced with 0

2. The first part \( G^+(Ax) \) can be approximated by the collection of 
   sketches \((S_0, \ldots, S_{h_{\text{max}}})\)

3. The second part \( f((Ax)^-) \) can be approximated by a uniform 
   sample \((T)\)
Sketching for Hinge Loss
**Overview of the Algorithm**

In the classification problem we are given a data matrix $Q \in \mathbb{R}^{n \times d}$ and a label vector $L \in \{-1, 1\}^n$. Let data matrix $A \in \mathbb{R}^{n \times d}$ where each row $a_i$ for $i \in [n]$ is defined as $a_i := -l_i q_i$. Our goal is to find $x \in \mathbb{R}^d$ that minimizes the hinge loss given by

$$H = \sum_{i \in [n]} \max(0, 1 + a_i x) = \sum_{a_i x \geq -1} a_i x + 1 = \sum_{y(i) \geq -1} y(i)$$

![Graph](image)
We will approximate $H$ using a matrix $S$ consisting of sketching matrices $S_0, \ldots, S_{l_{\text{max}}}$ where the sketch $S_l$ on each level is presented only a fraction of all coordinates.

This approach is based on a combination of subsampling at different levels and hashing the coordinates assigned to the same level uniformly into a small number of buckets.

Collisions are handled by summing all entries that are mapped to the same level and we use a CountMin-sketch algorithm to recover large enough entries.
Let $b$ be the number of buckets in each level. We let $l_{\text{max}} = 10 \log\left(\frac{n}{\epsilon}\right)$.
So each $S_l \in \mathbb{R}^{(l_{\text{max}}b) \times n}$ for $l \in [l_{\text{max}}]$.

For entry $y(j)$ for any $j \in [n]$

1. $y(j)$ is assigned to level $l$ w.p. $\frac{1}{\beta 2^l}$ where $\beta = 2 - 2^{-l_{\text{max}}}$
2. insert $y(j)$ assigned to level $l$ in a CMS datastructure called $C_l$ with $m$ buckets and using $t$ hash functions ($b = mt$).
3. for each level compute a list of all ’recovered’ elements $R_l$ with $\frac{|y(j) - \tilde{y}(j)|}{y(j)} \leq c \cdot \epsilon$ for some small constant $c$ where $\tilde{y}(j)$ are the approximated values by CMS for all $y(j)$ in level $l$
4. if assigned to level $l$, $y(j)$ gets the weight:
   • $w_j = \beta 2^l$ if it is recovered ($y(j) \in R_l$)
   • $w_j = 0$ otherwise

**Approximate:** $H = \sum_{j=1}^{n} y(j)$ by $\tilde{H} = \frac{1}{l_{\text{max}}} \sum_{j=1}^{n} w_j \cdot \tilde{y}(j)$
**Algorithm Cont.**

\[
\begin{align*}
S\ell_0 & = \begin{bmatrix}
0 & 0 & 0 \\
0 & 2\beta & 0 \\
\vdots & \vdots & \vdots \\
0 & 0 & 2\beta \\
\end{bmatrix} \\
S\ell_1 & = \begin{bmatrix}
0 & 0 \\
2\beta & 0 \\
\vdots & \vdots \\
0 & 0 \\
\end{bmatrix} \\
S_{\text{max}} & = \begin{bmatrix}
0 & 0 \\
2\beta & 0 \\
\vdots & \vdots \\
0 & 0 \\
\end{bmatrix}
\end{align*}
\]

\[
y = (Ax + 1)
\]

\[
Sy = S(Ax + 1)
\]

\[
y = (SA)x + 5\beta
\]

Only recovered entries

\[
\begin{bmatrix}
1 \\
2\beta(y'_i + y_k) \\
\vdots \\
2\beta(y'_i + y_k) \\
\end{bmatrix}
\]
Goal of the sketch:

- preserves big entries $y(i)$
- for smaller entries it finds a set of representatives which are in buckets of appropriate weight and are large in contrast to the remaining entries

We will show this by splitting entries $y(i)$ into weight classes and deriving bounds for the contribution of each weight class.

**Weight classes**: For $q \in \mathbb{N}$ we define

$$B_q = \{y(j) \mid 2^{-q}H < y(j) \leq 2^{-q+1}H\}$$

where $q_{\text{max}} = \log\left(\frac{n}{\epsilon}\right)$.

**Count-Min Sketch guarantees**: Let $x$ denote a signal vector. By setting $m = \frac{2}{\epsilon}$, $t = \log\left(\frac{1}{\delta}\right)$ we have

$$Pr[\tilde{x}_i - x_i \geq \epsilon \cdot \|x\|_1] \leq \left(\frac{1}{m\epsilon}\right)^t \leq \delta$$

where $m$ denotes the number of buckets and $t$ the number of pairwise independent hash functions.
**Theorem:**\[ \mathbb{E}\left[ \frac{1}{l_{\max}} \sum_{j=1}^{n} w_j \cdot \tilde{y}(j) \right] = (1 + \epsilon) \mathbb{E}\left[ \sum_{j=1}^{n} y(j) \right] \]

**Proof:**\[ \mathbb{E}\left[ \frac{1}{l_{\max}} \sum_{j=1}^{n} w_j \cdot \tilde{y}(j) \right] = \]

\[ \frac{1}{l_{\max}} \sum_{j=1}^{n} \sum_{l=0}^{l_{\max}} \frac{1}{\beta^{2l}} \cdot \beta^{2l} \cdot \operatorname{Pr}[y(j) \text{ is recovered given it's assigned to level } l] \cdot \tilde{y}(j) \]

Let's compute\[ \sum_{j=1}^{n} \sum_{l=0}^{l_{\max}} \operatorname{Pr}[y(j) \text{ is recovered given it's assigned to level } l] \cdot \tilde{y}(j) \]

Let \( y(j) \in B_q \) then \( y(j) \in (2^{-q}H, 2^{-q+1}H] \). Assume \( y(j) \) is assigned to level \( l \). We know that in expectation we have \( \frac{n}{\beta^{2l}} \) other elements in this level. Let's denote the sum of these elements as \[ S_l = \sum_{y(i) \text{ in level } l} y(i) \approx \frac{1}{\beta^{2l}} \cdot H. \]
For \( y(j) \) to be recovered correctly we must have that \( \frac{|y(j) - \tilde{y}(j)|}{y(j)} \leq c \cdot \epsilon \).

Our CMS datastructure for level \( l \) allows us to approximate \( \tilde{y}(j) - y(j) \leq \epsilon \cdot S_l \) with high probability. So we have that:

\[
\frac{|y(j) - \tilde{y}(j)|}{y(j)} \leq c \cdot \epsilon \implies \frac{\epsilon \cdot S_l}{y(j)} \leq c \cdot \epsilon 
\]  

(1)

\[
y(j) \geq \frac{S_l}{c} \implies 2^{-q+1}H \geq \frac{1}{c \cdot \beta} \cdot \frac{1}{2^l} \cdot H
\]  

(2)

\[
\implies l - q \geq \log\left(\frac{1}{c \beta}\right) - 1
\]  

(3)

\[
\implies l \geq q - K
\]  

(4)

for some constant \( K \). Hence

\[
\sum_{j=1}^{n} \sum_{l=0}^{l_{\text{max}}} Pr[\text{y}(j) \text{is recovered given it's assigned to level } l] \cdot \tilde{y}(j) \approx \sum_{j=1}^{n} (l_{\text{max}} - q) \cdot \tilde{y}(j)
\]
Thus we have that

$$\mathbb{E}\left[ \frac{1}{l_{\text{max}}} \sum_{j=1}^{n} w_j \cdot \tilde{y}(j) \right] = \frac{1}{l_{\text{max}}} \sum_{j=1}^{n} (l_{\text{max}} - q) \cdot \tilde{y}(j)$$  \hspace{1cm} (5)$$

$$= \frac{l_{\text{max}} - q}{l_{\text{max}}} \sum_{j=1}^{n} \tilde{y}(j)$$  \hspace{1cm} (6)$$

$$= \frac{9 \log \frac{n}{\epsilon}}{10 \log \frac{n}{\epsilon}} \cdot \left( \sum_{j=1}^{n} y(j) + c\epsilon y(j) \right)$$  \hspace{1cm} (7)$$

$$= 0.9(1 + c\epsilon) \cdot \sum_{j=1}^{n} y(j)$$  \hspace{1cm} (8)$$

$$= 0.9(1 + c\epsilon) \cdot H$$  \hspace{1cm} (9)$$
• Oblivious Sketching for Logistic Regression [A. Munteanu, S. Omlor, D. Woodruff (2021)]
• Coresets for Classification - Simplified and Strengthened [T. Mai, C. Musco, A. B. Rao (2021)]
• On coresets for logistic regression [A. Munteanu, C. Schwiegelshohn, C. Sohler, D. P. Woodruff (2018)]
• Unconditional coresets for regularized loss minimization. [A. Samadian, K. Pruhs, B. Moseley, S. Im, R. R. Curtin (2020)]
Thank you!