Data Sketches for Disaggregated Subset Sum Estimation

Daniel Ting
Tableau Software
1162 N 34th St
Seattle, Washington 98103
dting@tableau.com

ABSTRACT

We introduce and study a new data sketch for processing massive datasets. It addresses two common problems: 1) computing a sum given arbitrary filter conditions and 2) identifying the frequent items or heavy hitters in a data set. For the former, the sketch provides unbiased estimates with state of the art accuracy. It is specifically designed to handle the challenging scenario when the data is disaggregated. In this case, there is a per unit metric of interest that can only be computed as an expensive pre-aggregation of the raw, disaggregated data. For example, the metric of interest may be total clicks per user while the raw data is a click stream containing multiple rows per user. By creating a small, in-memory sketch of a massive dataset, a consumer may interactively query the data nearly instantaneously while still being able to slice or filter by almost any dimension. The sketch is suitable for use in a wide range of applications including computing historical click through rates for ad prediction, reporting user metrics from user event streams, and measuring network traffic for IP flows.

We informally prove and empirically show that the sketch has good properties for both the disaggregated subset sum estimation and frequent item problems on i.i.d. data. It not only picks out the frequent items but also gives strongly consistent estimates for the proportion of each frequent item. For subset sum estimation, it asymptotically draws a probability proportional to size sample that is optimal for estimating the sum over the data. Empirically, despite the disadvantage of operating on disaggregated data, our method matches or bests priority sampling, a state of the art method on pre-aggregated data. When compared to naive uniform sampling, it performs orders of magnitude better on skewed data. We also propose extensions to the sketch that allow it to be used in combining multiple data sets, in distributed systems, and for time decayed aggregation.

This paper is a work in progress.

CCS CONCEPTS

•Mathematics of computing →Probabilistic algorithms; •Theory of computation →Sketching and sampling;

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada
© 2017 Copyright held by the owner/author(s). .
DOI:

KEYWORDS

Data sketching, subset sum estimation, counting, frequent item, heavy hitters, sampling

1 INTRODUCTION

When analyzing massive data sets, even simple operations such as computing a sum or mean is costly and time consuming. These simple operations are frequently performed both by people investigating the data interactively and asking a series of questions about it as well as in automated systems which must monitor or collect a multitude of statistics.

Data sketching algorithms enable the information in these massive datasets to be efficiently processed, stored, and queried. This allows them to be applied, for example, in real-time systems, both for ingesting massive data streams or for interactive analysis.

In order to achieve this efficiency, sketches are designed to only answer a specific class of question, and there is typically error in the answer. In other words, it is a form of lossy compression on the original data where one must choose what to lose in the original data. A good sketch makes the most efficient use of the data so that the errors are minimized while having the flexibility to answer a broad range of questions of interest. Some sketches, such as HyperLogLog, are constrained to answer very specific questions with extremely little memory. On the other end of the spectrum, sampling based methods such as priority sampling [13] or coordinated sampling [3], [7] are able to answer almost any question on the original data but at the cost of far more space to achieve the same approximation error.

We introduce a sketch, unbiased space saving, that simultaneously addresses two common data analysis problems: the disaggregated subset sum problem and the frequent item problem. This makes the sketch more flexible than previous sketches that address one problem or the other. Furthermore, it is efficient as it provides state of the art performance on the disaggregated subset sum problem and has a stronger consistency guarantee for frequent item count estimation than previous results for i.i.d. streams.

The disaggregated subset sum estimation is a more challenging variant of the subset sum estimation problem [13], the extremely common problem of computing a sum or mean over a dataset with arbitrary filtering or grouping conditions. In the disaggregated subset sum problem [5], [17] the data is "disaggregated" so that a per item metric of interest is split across multiple rows. For example in an ad click stream, the data may arrive as a stream of single clicks that are identified with each ad while the metric of interest is the total number of clicks per ad. The frequent item problem is the problem of identifying the heavy hitters or most frequent items in a dataset. Several sketches exist for both these individual problems.

In particular, the sample and hold methods of [5], [15], [17] address the disaggregated subset sum estimation problem. Frequent item sketches include the space saving sketch [23], Misra-Gries sketch [24], and lossy counting sketch. [22].

Our sketch is an extension of the space saving frequent item sketch, and as such, has stronger frequent item estimation properties than sample and hold. In particular, unlike sample and hold, theorem 6.1 gives both that a frequent item will eventually be included in the sketch with probability 1, and that the proportion of times it appears will be consistently estimated for i.i.d. streams. In contrast to frequent item sketches which are biased, our unbiased space saving sketch gives unbiased estimates for any subset sum, including subsets containing no frequent items.

Our contributions are in three parts: 1) the development of the unbiased space saving sketch, 2) the generalizations obtained from understanding the properties of the sketch and the mechanisms by which it works, and 3) the theoretical and empirical results establishing the correctness and efficiency of the sketch for answering the problems of interest. In particular, the generalizations allow multiple sketches to be merged so that information from multiple data sets may be combined as well as allowing it to be applied in distributed system. Other generalizations include the ability to handle signed and real-valued updates as well as time-decayed aggregation. We empirically test the sketch on both synthetic and real ad prediction data. Surprisingly, we find that it even outperforms, priority sampling, a method that requires pre-aggregated data.

This paper is structured as follows. First, we describe the disaggregated subset sum problem, some of its applications, and related sketching problems. We then introduce our sketch, unbiased space saving, as a small but significant modification of the space-saving sketch. We examine its relation to other frequent item sketches, and show that they differ in a ”reduction” operation. This is used to show that any unbiased reduction operation yields an unbiased sketch for the disaggregated subset sum estimation problem. The theoretical properties of the sketch are then examined. We conjecture its consistency for the frequent item problem and for drawing a probability proportional to size sample and provide informal arguments that we believe can be made precise. Finally, we present experiments using real and synthetic data.

2 DISAGGREGATED SUBSET SUM PROBLEM

Many data analysis problems consist of a simple aggregation over some filtering and group by conditions.

```
SELECT sum(metric), dimensions
FROM table
WHERE filters
GROUP BY dimensions
```

This problem has several variations that depend on what is known about the possible queries and about the data before the sketch is constructed. For problems in which there is no group by clause and the set of possible filter conditions are known before the sketch is constructed, counting sketches such as the CountMin sketch [9] and AMS sketch [2] are appropriate. When the filters and group by dimensions are not known and arbitrary, the problem is the subset sum estimation problem. Sampling methods such as priority sampling [13] can be used to solve it. These work by exploiting a measure of importance for each row and sampling important rows with high probability. For example, when computing a sum, the rows containing large values contribute more to the sum.

The disaggregated subset sum estimation problem is a more difficult variant where there is little to no information about row importance and only a small amount of information about the queries. For example, many user metrics, such as number of clicks, are computed as aggregations over some event stream where each event has the same weight 1 and hence, the same importance. Filters and group by conditions can be arbitrary except for a small restriction that one cannot query at a granularity finer than a specified unit of analysis. In the click example, the finest granularity may be at the user level. One is allowed to query over arbitrary subsets of users but cannot query a subset of a single user's clicks. The data is ”disaggregated” since the relevant per unit metric is split across multiple rows. We will refer to something at the smallest unit of analysis as an *item* to distinguish it from one row in the data.

Since pre-aggregating to compute per unit metrics does not reduce the amount of relevant information, it follows that the best accuracy one can achieve is to first pre-aggregate and then apply a sketch for subset sum estimation. This operation, however, is extremely expensive, especially as the number of units is often large. Examples of units include users and ad id pairs for ad click prediction, source and destination IP pairs for IP flow metrics, and distinct search queries or terms. Each of these have trillions or more possible units.

Several sketches based on sampling have been proposed that address the disaggregated subset sum problem. These include the bottom-k sketch [6] which samples items uniformly at random, the class of ”NetFlow” sketches [14], and the sample and hold sketches [5], [15], [17]. Of these, the Sample-and-Hold sketches are clearly the best as they use strictly more information than the other methods to construct samples and maintain aggregate statistics. We describe them in more depth in section 4.4.

The unbiased space-saving sketch we propose throws away even less information than previous sketches. Surprisingly, this allows it to match the accuracy of priority sampling, a nearly optimal subset sum estimation algorithm [29], which uses pre-aggregated data. In some cases, our sketch achieves better accuracy despite being computed on disaggregated data.

2.1 Applications

The disaggregated subset sum problem has many applications. These include machine learning and ad prediction [28], analyzing network data [14], [5], detecting distributed denial of service attacks [27], database query optimization and join size estimation, as well as analyzing web users' activity logs or other business intelligence applications.

For example, in ad prediction the historical click-through rate and other historical data are among the most powerful features for future ad clicks [18]. Since there is no historical data for newly created ads, one may use historical click or impression data for previous ads with similar attributes such as the same advertiser or product category [26]. In join size estimation, it allows the sketch to estimate the size under the arbitrary filtering conditions that a user might impose.

Algorithm 1 Space-Saving algorithms

- Maintain an m list of $(item, count)$ pairs initialized to have count 0.
- For each new row in the stream, let x_{new} be its item and increment the corresponding counter if the item is in the list. Otherwise, find the pair $(x_{min}, \hat{N}_{min})$ with the smallest count. Increment the counter and replace the item label with x_{new} with probability p.
- For the original space-saving algorithm $p = 1$. For unbiased count estimates $p = 1/(\hat{N}_{min} + 1)$.

It also can be naturally applied to hierarchical aggregation problems. For network traffic data, IP addresses are arranged hierarchically. A network administrator may both be interested in individual nodes that receive or generate an excess of traffic or aggregated traffic statistics on a subnet. Several sketches have been developed to exploit hierarchical aggregations including [8], [25], and [30]. Since the disaggregated subset sum sketches handles arbitrary group by conditions, it can compute the next level in a hierarchy.

2.2 Frequent item problem

The frequent item or heavy hitter problem is related to the disaggregated subset sum problem. Our sketch is an extension of space saving, [23], a frequent item sketch. Like the disaggregated subset sum problem, frequent item sketches are computed with respect to a unit of analysis that requires a partial aggregation of the data. Only the most frequent items are of interest though. Most frequent item sketches are deterministic and have deterministic guarantees on both the identification of frequent items and the error in the counts of individual items. However, since counts in frequent item sketches are biased, further aggregation on the sketch can lead to large errors as bias accumulates as shown in section 6.2.

Our work is based on a frequent item sketch, but applies randomization to achieve unbiased count estimates. This allows them to be used in subset sum queries. Furthermore, it maintains good frequent item estimation properties as shown in section 6.

3 UNBIASED SPACE-SAVING

Our sketch is based on the space-saving sketch [23] used in frequent item estimation. For simplicity, we consider the case where the metric of interest is the count for each item. The space saving sketch works by maintaining a list of m bins labeled by distinct items. A new row with item i increments i's counter if it is in the sketch. Otherwise, the smallest bin is incremented, and its label is changed to i. Our sketch introduces one small modification. If $\hat{N}_{min}$ is the count for the smallest bin, then only change the label with probability $1/(\hat{N}_{min} + 1)$. This change provably yields unbiased counts as shown in theorem 3.1 More formally, the algorithms are given in algorithm 1.

Theorem 3.1. *For any item x, the randomized Space-Saving algorithm in figure 1 gives an unbiased estimate of the count of x.*

Proof. Let $\hat{N}_x(t)$ denote the estimate for the count of x at time t and $\hat{N}_{min}(t)$ be the count in the smallest bin. We show that the expected increment to $N_x(t)$ is 1 if x is the next item and 0 otherwise. Suppose x is the next item. If it is in the list of counters, then it is incremented by exactly 1. Otherwise, it incremented by $\hat{N}_{min}(t) + 1$ with probability $1/(\hat{N}_{min}(t) + 1)$ for an expected increment of 1. Now suppose x is not the next item. The estimated count $\hat{N}_x(t)$ can only be modified if x is the label for the smallest count. It is incremented with probability $\hat{N}_x(t)/(\hat{N}_x(t) + 1)$. Otherwise $\hat{N}_x(t+1)$ is updated to 0. This gives the update an expected increment of $\mathbb{E}\hat{N}_x(t+1) - \hat{N}_x(t) = (\hat{N}_x(t) + 1)\hat{N}_x(t)/(\hat{N}_x(t) + 1) - \hat{N}_x(t) = 0$ when the new item is not x. □

We note that although given any fixed item x, the estimate of its count is unbiased, each stored pair often contains an overestimate of the item's count. This occurs since any item with a positive count will receive a downward biased estimate of 0 when it is not in the sketch. Thus, conditional on an item appearing in the sketch, the count must be biased upwards to balance out the bias.

4 RELATED SKETCHES AND FURTHER GENERALIZATIONS

Although our primary goal is to demonstrate the usefulness of the unbiased space-saving sketch, we also try to understand the mechanisms by which it works and find extensions and generalizations that can be gleaned from that understanding.

In particular, we examine the relationship between unbiased space saving and existing deterministic frequent items sketches. We show that existing frequent item sketches all share the same structure as an exact increment of the count followed by a size reduction. This size reduction is implemented as an adaptive sequential thresholding operation which biases the counts. Our modification replaces the thresholding operation with a subsampling operation. This observation allows us to extend the sketch. This includes endowing it with an unbiased merge operation that can be used to combine datasets or in distributed computing environments.

The sampling design in the reduction step may also be chosen to give the sketch different properties. For example, time-decayed sampling methods may be used to weight recently occurring items more heavily. If multiple metrics are being tracked, the multi-objective sampling [4] may be used.

4.1 Probability proportional to size sampling

Our key observation in generalizing unbiased space saving is that the choice of label is a sampling operation. In particular, this sampling operation chooses the item with probability proportional to its size. We briefly review probability proportional to size sampling and priority sampling as well as the Horvitz-Thompson estimator which allows one to unbias the sum estimate from any biased sampling scheme.

For unequal probability samples, an unbiased estimator for the sum over the true population $\{x_i\}$ is given by the Horvitz-Thomson estimator $\hat{S} = \sum_i \frac{x_i Z_i}{\pi_i}$ where Z_i denotes whether x_i is in the sample and $\pi_i = P(Z_i = 1)$ is the inclusion probability. When only linear statistics of the sampled items are computed, the item values may be updated $x_i^{new} = x_i/\pi_i$.

When drawing a sample of fixed size, it is trivial to see that an optimal set of inclusion probabilities is given by $\pi_i \propto x_i$ when this is possible. In other words, it generates a probability proportional to size (PPS) sample. In this case, each term in the sum is constant, so that the estimator is exact and has zero variance. When the data is skewed, drawing a probability proportional size sample may be impossible for sample sizes greater than 1. For example, given values 1, 1, and 10, any scheme to draw 2 items with probabilities exactly proportional to size has inclusion probabilities bounded by 1/10, 1/10, and 1. The expected sample size is at most $12/10 < 2$. In this case, one often chooses inclusion probabilities $\pi_i = \min\{\alpha x_i, 1\}$ for some constant α. The inclusion probabilities are proportional to the size if the size is not too large and 1 otherwise.

Many algorithms exist for generating PPS samples. In particular, the splitting procedure of [12] provides a class of methods to generate a fixed size PPS sample with the desired inclusion probabilities. Another method which approximately generates a PPS sample is priority sampling. Instead of exact inclusion probabilities which are typically intractable to compute, priority sampling generates a set of pseudo-inclusion probabilities.

4.2 Misra-Gries and frequent item sketches

The Misra-Gries sketch [24], [11], [20] is a frequent item sketch and is isomorphic to the space saving sketch [1]. The only difference is that it decrements all counters rather than incrementing the smallest bin when processing an item that is not in the sketch. Thus, the count in the smallest bin for the space-saving sketch is equal to the total number of decrements in the Misra-Gries sketch. Given estimates $\hat{\mathbf{N}}$ from a space-saving sketch, the corresponding estimated item counts for the Misra-Gries sketch are $\hat{N}_i^{MG} = (\hat{N}_i - \hat{N}_{min})_+$ where $\hat{N}_{min}$ is the count for the smallest bin and the operation $(x)_+$ truncates negative values to be 0. In other words, the Misra-Gries estimate is the same as space saving estimate soft thresholded by $\hat{N}_{min}$. Equivalently, the space-saving estimates are obtained by adding back the total number of decrements $\hat{N}_{min}$ to any nonzero counter in the Misra-Gries sketch.

The sketch has a deterministic error guarantee. When the total number of items is N and the total estimated count for items in the sketch is $\hat{n}_{tot} = \sum_i \hat{n}_i$ then the error for any item's count is at most $(n - \hat{n}_{tot})/(m + 1)$.

Other frequent item sketches include the deterministic lossy counting and randomized sticky sampling sketches [22]. We describe only lossy counting as sticky sampling has both worse practical performance and weaker guarantees than other sketches.

A simplified version of Lossy counting applies the same decrement reduction as the Misra-Gries sketch but decrements occur at a fixed schedule rather than one which depends on the data itself. To count items with frequency $> N/m$, all counters are decremented after every m rows. Lossy counting does not provide a guarantee that the number of counters can be bounded by m. In the worst case, the size can grow to $m \log(N/m)$ counters. Similar to the isomorphism between the Misra-Gries and Space-saving sketches, the original Lossy counting algorithm is recovered by adding the number of decrements back to any nonzero counter.

4.3 Reduction operations

Existing deterministic frequent item sketches differ in only the operation to reduce the number of nonzero counters. They all have the form described in algorithm 2 and have reduction operations that can be expressed as a thresholding operation. Although it is isomorphic to the Misra-Gries sketch, space-saving's reduction operation can also be described as collapsing the two smallest bins by adding the larger bin's count to the smaller one's.

Algorithm 2 General frequent item sketching

```
• Maintain current estimates of counts $\hat{\mathbf{N}}(t)$
• Increment $\hat{N}'_{x_{t+1}}(t+1) \leftarrow \hat{N}_{x_{t+1}}(t) + 1$.
• $\hat{\mathbf{N}}(t+1) \leftarrow ReduceBins(\hat{\mathbf{N}}'(t+1), t+1)$
```

Modifying the reduction operation provides the sketch with different properties. We highlight several uses for alternative reduction operations.

The reduction operation for unbiased space saving can be seen as a PPS sample on the two smallest bins. A natural generalization is to consider a PPS sample on all the bins. We highlight three benefits of such a scheme. First, items can be added with arbitrary counts or weights. Second, the sketch size can be reduced by multiple bins in one step. Third, there is less quadratic variation added by one sampling step, so error can be reduced. The first two benefits are obvious consequences of the generalization. To see the third, consider when a new row contains an item not in the sketch, and let $\mathcal{J}$ be the set of bins equal to the size of the smallest bin $\hat{N}_{min}$. When using the thresholded PPS inclusion probabilities from section 4.1, the resulting PPS sample has inclusion probability $\alpha = |\mathcal{J}|/(1 + |\mathcal{J}|\hat{N}_{min})$ for the new row's item and $\alpha\hat{N}_{min}$ for bins in $\mathcal{J}$. Other bins have inclusion probability 1. After sampling, the Horvitz-Thompson adjusted counts are $1/|\mathcal{J}| + \hat{N}_{min}$. Unbiased space saving is thus a further randomization to convert the real valued increment $1/|\mathcal{J}|$ over $|\mathcal{J}|$ bins to an integer update on a single bin. Since unbiased space saving adds an additional randomization step, the PPS sample has smaller variance. The downside of this procedure, however, is that it requires real valued counters that require more space per bin.

Changing the sampling procedure can also provide other desirable behaviors. Applying forward decay sampling [10] allows one to obtain estimates that weight recent items more heavily. Other possible operations include adaptively varying the sketch size in order to only remove items with small estimated frequency.

Furthermore, the reduction step does not need to be limited strictly to subsampling. Theorem 4.1 gives that any unbiased reduction operation yields unbiased estimates. This generalization allows us to analyze Sample-and-Hold sketches.

Theorem 4.1. *Any reduction operation where the expected post-reduction estimates are equal to the pre-reduction estimates yields an unbiased sketch for the disaggregated subset estimation problem. More formally, if* $\mathbb{E}(\hat{\mathbf{N}}(t)|S_{pre}(t)) = \hat{\mathbf{N}}_{pre}(t)$ *where* $S_{pre}(t), \hat{\mathbf{N}}_{pre}(t)$ *are the sketch and estimated counts before reduction at time step* t *and* $\hat{\mathbf{N}}(t)$ *is the post reduction estimate, then* $\hat{\mathbf{N}}(t)$ *is an unbiased estimator.*

Proof. Since $\hat{\mathbf{N}}_{pre}(t) = \hat{\mathbf{N}}_{post}(t-1) + (\mathbf{n}(t) - \mathbf{n}(t-1))$, it follows that $\hat{\mathbf{N}}(t) - \mathbf{n}(t)$ is a martingale with respect to the filtration adapted to $S(t)$. Thus, $\mathbb{E}\hat{\mathbf{N}}(t) = \mathbf{n}(t)$, and the sketch gives unbiased estimates for the disaggregated subset sum problem. □

We also note that reduction operations can be biased. The merge operation on the Misra-Gries sketch given by [1] can be seen as performing a soft-thresholding by the size of the $(m+1)^{th}$ counter. This also allows it to reduce the size of the sketch by more than 1 bin at a time. It can be modified to handle deletions and arbitrary numeric aggregations by making the thresholding operation two-sided so that negative values are shrunk toward 0 as well. In this case, we do not provide a theoretical analysis of the properties.

Modifying the reduction operation also yields interesting applications outside of counting. In particular, a reduction operation on matrices can yield accurate low rank decompositions [21], [16].

4.4 Sample and Hold

To the author's best knowledge, the current state of the art sketches designed to answer disaggregated subset sum estimation problems are the family of sample and hold sketches [17], [15], [5]. These methods can also be described with a randomized reduction operation.

For adaptive sample and hold [5], the sketch maintains an auxiliary variable p which represents the sampling rate. Each point in the stream is assigned a $U_i \sim Uniform(0,1)$ random variable, and the items in the sketch are those with $U_i < p$. If an item remains in the sketch starting from time t_0, then the counter stores the number of times it appears in the stream after the initial time. Every time the sketch becomes too large, the sampling rate is decreased so that under the new rate p', one item is no longer in the sketch.

It can be shown that unbiased estimates can be obtained by keeping a counter value the same with probability p'/p and decrementing the counter by a random $Geometric(p')$ random variable otherwise. If a counter becomes negative, then it is set to 0 and dropped. Adding back the mean $(1-p')/p'$ of the *Geometric* random variable to the nonzero counters gives an unbiased estimator. Effectively, the sketch replaces the first time an item enters the sketch with the expected $Geometric(p')$ number of tries before it successfully enters the sketch plus it adds the actual count after the item enters the sketch. Using the memoryless property of *Geometric* random variables, it is easy to show that the sketch satisfies the conditions of theorem 4.1. It is also clear that one update step adds more error and unbiased space saving as it potentially adds $Geometric(p')$ noise with variance $(1-p')/p'^2$ to every bin. Furthermore, the eliminated bin may not even be the smallest bin. Since p' is the sampling rate, it is expected to be close to 0. By contrast, unbiased space saving has bounded increments of 1 for bins other than the smallest bin, and the only bin that can be removed is the current smallest bin.

The discrepancy is especially prominent for frequent items. A frequent item in an i.i.d. stream for unbiased space saving enters the sketch almost immediately, and the count for the item is nearly exact as shown in theorem 6.1. For adaptive sample and hold, the first $n_i(1-p')$ occurrences of item i are expected to be discarded and replaced with a high variance $Geometric(p')$ random variable. Since p' is typically small in order to keep the number of counters low, most of the information about the count is discarded.

Another sketch, step sample-and-hold, avoids the problem by maintaining counts for each "step" when the sampling rate changes. However, this is more costly both from storage perspective as well as a computational one. For each item in the sketch, computing the expected count takes time quadratic in the number of steps J_i in which the step's counter for the item is nonzero, and storage is linear in J_i.

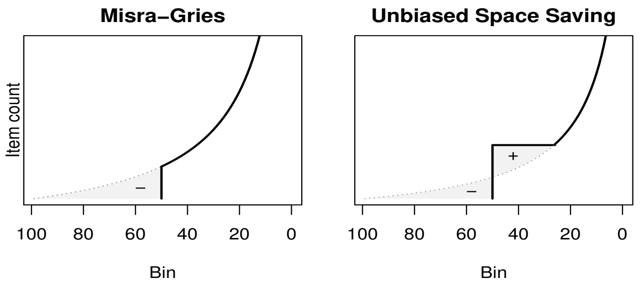

Figure 1: In a merge operation, the Misra-Gries sketch simply removes mass from the extra bins with small count. Unbiased space saving moves the mass from infrequent items to moderately frequent items. It loses the ability to pick those items as frequent items in order to provide unbiased estimates for the counts in the tail.

4.5 Merging and Distributed counting

The generalized reduction operations allow for merge operations on the sketches. Merge operations and mergeable sketches [1] are important since they allow a collection of sketches, each which answers questions about the specific data it was constructed on, to be combined to answer a question over all the data. For example, a set of frequent item sketches that give trending news for each country can be combined to give trending news for Europe as well as a multitude of other possible combinations. Another common scenario arises when sketches are aggregated across time. Sketches for clicks may be computed per day, but the final machine learning feature may combine the last 7 days. Furthermore, merges allow for simple distributed computation. In a map-reduce framework, each mapper can quickly compute a sketch, and only a set of small sketches needs to be sent over the network to perform an aggregation at the reducer.

As noted in the previous section, the Misra-Gries sketch has a simple merge operation which preserves its deterministic error guarantee. It simply soft thresholds by the $(m+1)^{th}$ largest counter so that at most m nonzero counters are left. Previously, no merge operation existed for space-saving except to first convert it to a Misra-Gries sketch. The conversion of soft-thresholds to approximate hard thresholds yields a merge operation for space-saving sketches. However, this does not preserve the total item count. Theorem 4.1 shows that by replacing the pairwise randomization with priority sampling or some other sampling procedure still allows one to obtain an unbiased space saving merge that can preserve the expected count in the sketch rather than biasing it downward.

The trade-off required for such an unbiased merge operation is that the sketch may detect fewer of the top items by frequency than the biased Misra-Gries merge. Rather than truncating and preserving more of the "head" of the distribution, it must move mass from the tail closer to the head. This is illustrated in figure 1.

5 PROPERTIES

We study the properties of the space-saving sketches here. These include asymptotic properties, empirical properties, behavior in pathological cases, and costs in time and space. In particular, we conjecture and provide an informal proof that when the data is i.i.d., the sketch eventually includes frequent items with probability 1. Other items are sampled with probability proportional to their size. This is also borne out in the experimental results where the observed inclusion probabilities match the theoretical ones and in estimation error where unbiased space saving matches or even exceeds the accuracy of priority sampling. In pathological cases, we demonstrate that deterministic space-saving fails at the subset estimation problem. Furthermore, these pathological sequences can arise naturally. Any sequence where items' arrival rates change significantly over time forms a pathological sequence.

6 ASYMPTOTIC CONSISTENCY

We conjecture and provide a proof sketch that shows that the data sketch contains all frequent items eventually on i.i.d. stream. Thus it does no worse than deterministic space-saving asymptotically for frequent item estimation on such streams while having much better aggregation behavior on pathological streams.

Assume that items are drawn from a possibly infinite, discrete distribution with probabilities $p_1 \geq p_2 \geq \ldots$ and, without loss of generality, assume they are labeled by their index into this sequence of probabilities. Let m be the number of bins and t be the number of items processed by the sketch. We will also refer to t as time. Let $\mathcal{I}(t)$ be the set of items that are in the sketch at time t and $Z_i(t) = 1(i \in \mathcal{I}(t))$ indicate if the label i is in the sketch at time t. Define an absolutely frequent item to be an item drawn with probability $> 1/m$ where m is the number of bins in the sketch. Our precise conjecture and the proof sketch of its veracity are as follows.

Conjecture 6.1. *If $p_1 > 1/m$, then as the number of items $t \to \infty$, $Z_1(t) = 1$ eventually.*

Proof. The goal is to show that label 1 will eventually become "sticky." This requires (1) that some bin gets the label 1 and (2) that the bin "escapes" from being the smallest bin before it is relabeled, and (3) that it remains that way so that the label cannot be changed. For (1), it is easy to see that there are $\Omega(\log t)$ times that the smallest bin will flip to label 1. Denote the size of the smallest bin $N_{min}(t)$ and the estimated count for label 1 as $N_1(t)$. Trivially $N_{min}(t) < t/m \leq tp_1$.For the remaining two requirements, we compare the size of the bin with label 1 to t/m. For (2), we note that the smallest bin grows at a rate of at least $\alpha = \sum_{j>m} p_j$. Hence, $t/m - N_{min}(t) \leq (1-\alpha)t/m + o_p(1)$. A bin growing at rate p_1 will take approximately $t' - t = (t/m - N_{min}(t))(p_1 - 1/m)^{-1}$ steps to catch up to t'/m. The probability that the label is overwritten during this time is bounded above by $(t' - t)/t'$ which simplifies to a constant that does not depend on the time t. Thus, every time the label flips to 1, there is at least a constant nonzero probability of "escape." Every time a bin escapes with label 1, $N_1(t) - t/m$ forms an asymmetric random walk starting at or slightly above 0. The probability of never returning to 0 and, hence, never being relabeled is at least some positive constant c. Since there is some constant positive probability a bin will become sticky after being relabeled 1, and there are infinitely many time it will acquire that label, it eventually must become sticky. □

6.1 Approximate PPS Sample

An interesting consequence of the above conjecture is that bins fall into two classes for i.i.d. streams. The first class are bins with labels that become "sticky." These are asymptotically "pure" bins where the proportion of items with the current label goes to 1. The second are bins where the label keeps changing because the rate of a labeled bin is never greater than the rate for the smallest bin, and hence, the size must remain close to the minimum bin size.

Each time an item is added to the smallest bin, the label for that bin is a probability proportional to size sample of size 1 from the items previously added to that bin. This informal argument leads to our second conjecture.

Conjecture 6.2. *The items in the sketch converge in distribution to a PPS sample on i.i.d. streams where either a label is sampled with probability 1 or with probability $\propto n_i$.*

We note, however, that the resulting PPS sample has limitations not present in PPS samples on pre-aggregated data. For pre-aggregated data, one has both the original value x_i and the Horvitz-Thompson adjusted value x_i/π_i where π_i is the inclusion probability. This allows the sample to compute non-linear statistics such as the population variance which uses the second moment estimator $\sum_i x_i^2 Z_i/\pi_i$. With the PPS samples from disaggregated subset sum sketching, only the adjusted values are observed.

6.2 Pathological sequences

Deterministic space-saving has remarkably low error when estimating the counts of frequent items [8]. In general, if the order data arrives is uniformly random or if the data stream consists of i.i.d. data, one expects the deterministic space-saving algorithm to share similar unbiasedness properties as the randomized version as in both cases the label for a bin can be treated roughly as a uniform random choice out of the items in that bin.

Pathological cases arise when an item's arrival rate changes over time rather than staying constant. Consider a sketch with 2 bins. For a sequence of c 1's, c 2's, a single 3, and a single 4, the deterministic space saving algorithm will always return 3 and 4, each with count $c + 1$. By contrast, randomized space-saving will return 1 and 2 with probability $(1 - 1/c)^2 \approx 1$ when c is large. Note that in this case, the count for each frequent item is slightly below the threshold that guarantees inclusion in the sketch, $c < n/2$. This example illustrates the behavior for the deterministic algorithm. When an item is not in the "frequent item head" of the distribution then the bins that represent the tail pick the labels of the most recent items without regard to the frequency of older items.

We note that such a pathological sequence can easily occur naturally. For instance, partially sorted data can naturally lead to such pathological sequences. Periodic bursts of an item followed by periods in which its frequency drops below the threshold of guaranteed inclusion are another example. Another pathological case for disaggregated subset sum problems arises when every item is distinct. The deterministic sketch consists of the last m

items rather than a random sample, and no meaningful subset sum estimate can be derived for deterministic space saving.

6.3 Running time and space complexity

The update operation is identical to the deterministic space saving update except that it changes the label of a bin less frequently. Thus, each update can be performed in $O(1)$ time [23] when the stream summary data structure is used. In this case the space usage is $O(m)$ in the number of bins.

7 EXPERIMENTS

We perform experiments with both simulations and real ad prediction data. For synthetic data, we draw the count for each item using a Weibull distribution that is discretized to integer values. That is $n_i \sim Round(Weibull(k, \alpha))$ for item i. The discretized Weibull distribution is a generalization of the geometric distribution that allows us to adjust the tail of the distribution to be more heavy tailed. We choose it over the Zipfian or other truly heavy tailed distributions as few real data distributions have infinite variance. Furthermore, we expect our methods to perform even better with greater data skew as shown in figure 2. For more easily reproducible behavior we applied the inverse cdf method $n_i = F^{-1}(U_i)$ so that the U_i are on a regular grid of 1000 values rather than $Uniform(0, 1)$ random variables. In each case, we draw at least 10,000 samples to estimate the root mean squared error. To simulate a variety of possible filtering conditions, we draw random subsets of 100 items. As expected, subsets which mostly pick items in the tail of the distribution have estimates with higher relative root mean squared error. Note that an algorithm with α times the root mean squared error of a baseline algorithm often requires α^2 times the space as the variance, not the standard deviation, scales linearly with size.

For real data, we use a Criteo ad click prediction dataset [1]. This dataset provides a sample of 45 million ad impressions. Each sample includes the outcome of whether or not the ad was clicked as well as multiple integer valued and categorical features. We pick a subset of 9 of these features. There are over 500 million possible tuples on these features and many more possible filtering conditions. The impressions without a click are sampled at a lower rate than those with clicks.

The Criteo dataset provides a natural application of the disaggregated subset sum problem. Historical clicks are a powerful feature in click prediction [26], [19]. While the smallest unit of analysis is the *ad* or the (*user*, *ad*) pair, the data is in a disaggregated form with one row per impression. Furthermore, since there may not be enough data for a particular ad, the relevant click prediction feature may be the historical click through rate for the advertiser or some other higher level aggregation. Past work using sketches to estimate these historical counts [28] include the CountMin counting sketch as well as the Lossy Counting frequent item sketch.

Although we do not directly compare against sample and hold methods, we note that figure 2 in [5] shows that sample and hold performs worse than priority sampling.

This added variability in the threshold and the relatively small sketch sizes for the simulations on i.i.d. streams may explain why unbiased space saving performs even better than what could be considered close to a "gold standard" on pre-aggregated data.

[1] http://labs.criteo.com/2014/02/kaggle-display-advertising-challenge-dataset/

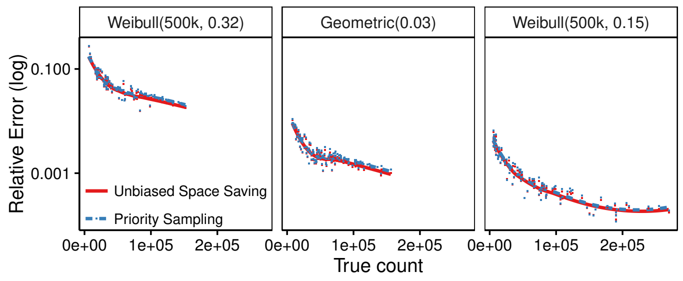

Figure 2: The empirical performance of Unbiased Space-Saving matches priority sampling, which required an expensive pre-aggregation step. The sketch accuracy improves when the skew is higher and when more and larger bins are contained in the subset. The number of bins is 200.

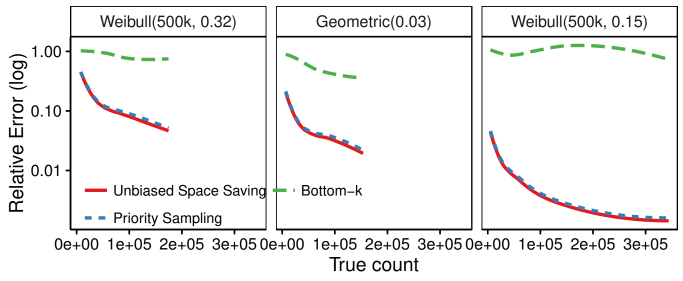

Figure 3: Unbiased space saving performs orders of magnitude better than uniform sampling of items (Bottom-k). The plots show the smoothed plot of relative error versus the true count. With 100 bins, the error is higher than with 200 bins given in figure 2 but the curve is qualitatively similar.

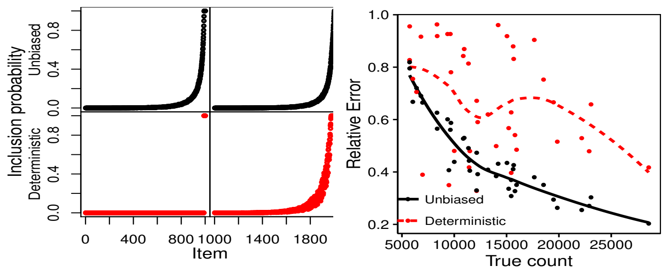

Figure 4: Deterministic Space-Saving performs poorly on pathological sequences. Left: Items 1 to 1000 only appear in the first half of the stream. The inclusion probabilities for a pathological sequence still behave like a PPS sample for unbiased space saving, but only the frequent items in the first half are sampled under deterministic space saving. Right: As a result, deterministic space saving is highly inaccurate when querying items in the first half of the stream.

8 CONCLUSION

We have introduced a novel sketch, unbiased space saving, that answers both the disaggregated subset sum and frequent item problems. Surprisingly, for the disaggregated subset sum problem, the sketch can outperform even methods that run on pre-aggregated data. We prove that asymptotically, it can answer the frequent item problem for i.i.d. sequences with probability 1 eventually. Furthermore, it gives stronger probabilistic consistency guarantees on the accuracy of the count than the deterministic space saving sketch. For infrequent items, we prove that the items selected for the sketch are sampled approximately according to a PPS sample.

We study its behavior and connections to other data sketches. In particular, we identify the primary difference between many of the frequent item sketches is a slightly different operation to reduce the number of bins. We use that understanding to provide multiple generalizations to the sketch which allow it to be applied in distributed settings, handle weight decay over time, and adaptively change its size over time. This also allows us to compare unbiased space to the family of sample and hold sketches that are also designed to answer the disaggregated subset sum problem.

REFERENCES

[1] P. K. Agarwal, G. Cormode, Z. Huang, Jeff M Phillips, Z. Wei, and K. Yi. 2013. Mergeable summaries. *ACM Transactions on Database Systems* 38, 4 (2013), 26.

[2] N. Alon, Y. Matias, and M. Szegedy. 1999. The Space Complexity of Approximating the Frequency Moments. *J. Comput. System Sci.* 58, 1 (1999), 137–147.

[3] K. R.W. Brewer, L.J. Early, and S.F. Joyce. 1972. Selecting several samples from a single population. *Australian & New Zealand Journal of Statistics* 14, 3 (1972), 231–239.

[4] Edith Cohen. 2015. Multi-objective weighted sampling. In *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on.* IEEE, 13–18.

[5] E. Cohen, N. Duffield, H. Kaplan, C. Lund, and M. Thorup. 2007. Sketching unaggregated data streams for subpopulation-size queries. In *PODS*. ACM.

[6] Edith Cohen and Haim Kaplan. 2007. Summarizing data using bottom-k sketches. In *PODC*.

[7] E. Cohen and H. Kaplan. 2013. What You Can Do with Coordinated Samples. In *RANDOM*.

[8] Graham Cormode and Marios Hadjieleftheriou. 2008. Finding frequent items in data streams. *VLDB* 1, 2 (2008), 1530–1541.

[9] G. Cormode and S. Muthukrishnan. 2005. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms* 55, 1 (2005), 58–75.

[10] G. Cormode, V. Shkapenyuk, D. Srivastava, and B. Xu. 2009. Forward decay: A practical time decay model for streaming systems. In *ICDE*. IEEE, 138–149.

[11] E. D. Demaine, A. López-Ortiz, and J. I. Munro. 2002. Frequency estimation of internet packet streams with limited space. In *European Symposium on Algorithms*. 348–360.

[12] J.C. Deville and Y. Tillé. 1998. Unequal probability sampling without replacement through a splitting method. *Biometrika* 85, 1 (1998), 89–101.

[13] Nick Duffield, Carsten Lund, and Mikkel Thorup. 2007. Priority sampling for estimation of arbitrary subset sums. *Journal of the ACM (JACM)* 54, 6 (2007), 32.

[14] C. Estan, K. Keys, D. Moore, and G. Varghese. 2004. Building a better NetFlow. In *ACM SIGCOMM Computer Communication Review*, Vol. 34. ACM, 245–256.

[15] Cristian Estan and George Varghese. 2002. *New directions in traffic measurement and accounting*. Vol. 32. ACM.

[16] Mina Ghashami, Edo Liberty, and Jeff M Phillips. Efficient Frequent Directions Algorithm for Sparse Matrices. (????).

[17] P. B. Gibbons and Y. Matias. 1998. New sampling-based summary statistics for improving approximate query answers. In *ACM SIGMOD Record*, Vol. 27. ACM, 331–342.

[18] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. Bowers, and others. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the International Workshop on Data Mining for Online Advertising*. ACM, 1–9.

[19] D. Hillard, S. Schroedl, E. Manavoglu, H. Raghavan, and C. Leggetter. 2010. Improving ad relevance in sponsored search. In *WSDM*. ACM, 361–370.

[20] R. M. Karp, S. Shenker, and C. H Papadimitriou. 2003. A simple algorithm for finding frequent elements in streams and bags. *ACM Transactions on Database Systems (TODS)* 28, 1 (2003), 51–55.

[21] Edo Liberty. 2013. Simple and deterministic matrix sketching. In *KDD*. ACM.

[22] G. Manku and R. Motwani. 2002. Approximate frequency counts over data streams. In *VLDB*.

[23] A. Metwally, D. Agrawal, and A. El Abbadi. 2005. Efficient computation of frequent and top-k elements in data streams. In *ICDT*.

[24] J. Misra and D. Gries. 1982. Finding repeated elements. *Science of computer programming* 2, 2 (1982), 143–152.

[25] M. Mitzenmacher, T. Steinke, and J. Thaler. 2012. Hierarchical heavy hitters with the space saving algorithm. In *Meeting on Algorithm Engineering & Expermiments*. 160–174.

[26] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *WWW*. ACM, 521–530.

[27] V. Sekar, N. Duffield, O. Spatscheck, J. E. van der Merwe, and H. Zhang. LADS: Large-scale Automated DDoS Detection System.

[28] A. Shrivastava, A. C. König, and M. Bilenko. 2016. Time Adaptive Sketches (Ada-Sketches) for Summarizing Data Streams. *SIGMOD* (2016).

[29] Mario Szegedy. 2006. The DLT priority sampling is essentially optimal. In *STOC*. ACM, 150–158.

[30] Y. Zhang, S. Singh, S. Sen, N. Duffield, and C. Lund. 2004. Online identification of hierarchical heavy hitters: algorithms, evaluation, and applications. In *Internet Measurement Conference (IMC)*. ACM, 101–114.