The SpaceSaving± Family of Algorithms for Data Streams with Bounded Deletions

Fuheng Zhao
UC Santa Barbara
fuheng_zhao@ucsb.edu

Divyakant Agrawal
UC Santa Barbara
agrawal@cs.ucsb.edu

Amr El Abbadi
UC Santa Barbara
amr@cs.ucsb.edu

Claire Mathieu
CNRS and IRIF
clairemmathieu@gmail.com

Ahmed Metwally
Uber Inc.
ametwally@uber.com

Michel de Rougemont
University Paris II and IRIF
m.derougemont@gmail.com

ABSTRACT

In this paper, we present an advanced analysis of near optimal algorithms that use limited space to solve the frequency estimation, heavy hitters, frequent items, and top-k approximation in the bounded deletion model. We define the family of SpaceSaving± algorithms and explain why the original SpaceSaving± algorithm only works when insertions and deletions are not interleaved. Next, we propose the new Double SpaceSaving±, Unbiased Double SpaceSaving±, and Integrated SpaceSaving± and prove their correctness. The three proposed algorithms represent different trade-offs, in which Double SpaceSaving± can be extended to provide unbiased estimations while Integrated SpaceSaving± uses less space. Since data streams are often skewed, we present an improved analysis of these algorithms and show that errors do not depend on the hot items. We also demonstrate how to achieve relative error guarantees under mild assumptions. Moreover, we establish that the important mergeability property is satisfied by all three algorithms, which is essential for running the algorithms in distributed settings.

CCS CONCEPTS

• **Information systems → Data management systems**; **Data mining**; • **Networks → Network monitoring**; • **Theory of computation → Streaming, sublinear and near linear time algorithms**.

KEYWORDS

Data Mining, Streaming Algorithms, Data Summary, Data Sketch, Frequency Estimation, Heavy Hitters, Frequent Items, Top-K.

ACM Reference Format:
Fuheng Zhao, Divyakant Agrawal, Amr El Abbadi, Claire Mathieu, Ahmed Metwally, and Michel de Rougemont. 2018. The SpaceSaving± Family of Algorithms for Data Streams with Bounded Deletions. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Streaming algorithms [22, 28] aim to process a stream of data in a single pass with small space. In particular, for a stream of updates of items, these algorithms often only store a compact summary which uses space at most logarithmic in the data stream length or in the cardinality of the input domain. The streaming paradigm provides essential analysis and statistical measurements with strong accuracy guarantees for many big data applications [3]. For instance, Hyperloglog [26] answers queries about cardinality; Bloom Filters [9] answer membership queries; KLL [30, 33, 54] gives accurate quantile approximations such as finding the median. These data summaries (sketches) are now the cornerstones for many real-world systems and applications including network monitoring [48, 49, 53], storage engine [23, 56], data mining [36, 46], federated learning [32, 44], privacy-preserving analytic [14, 35, 43, 47, 55], vector search [11, 12], and many more. We refer the reader to a recent paper [18] for a more thorough discussion on the applications.

In this paper, we focus on the frequency estimation and heavy hitters problems with the presence of deletions. Given a data stream σ from some universe U, the **Frequency Estimation** problem constructs a summary providing, for every item x in U, an estimate $\hat{f}(x)$ of the true frequency $f(x)$. Since resources are limited (small memory footprint), the data summary cannot track every item and hence there is an inherent trade-off between the space used and the accuracy of the estimate. In the closely related approximate **Heavy Hitters** (**Frequent Items**) problem, there is an additional input frequency threshold T, and the goal is to identify all items with frequency larger than T. This applies to settings where data scientists and system operators are interested in knowing what items are hot (appear very often). Frequency estimation, heavy hitters and approximate top-k have found many applications in big data systems [5, 15, 38, 50].

1.1 Related Work

There is a large body of algorithms proposed to solve frequency estimation and heavy hitters problems [10, 13, 21, 24, 31, 34, 39, 41, 42, 51, 55]. The existing algorithms can be classified into three models: insertion-only, turnstile, and bounded-deletion. In the insertion-only model, all arriving stream items are insertions and the frequency $f(x)$ of x is the number of times x has been inserted. SpaceSaving [41] and MG [42] are two popular algorithms that operate in the insertion-only model; they provide strong deterministic guarantees with minimal space. Moreover, since these algorithms are deterministic, they are inherently adversarial robust [6, 16, 29]. In

Algorithm	Space	Model	Error Bound	Note
SpaceSaving & MG [7, 40–42]	$O(\frac{1}{\epsilon})$	Insertion-Only	$\lvert f_i - \hat{f_i}\rvert \leq \epsilon F_1$	Lemma 2.3
	$O(\frac{k}{\epsilon})$		$\lvert f_i - \hat{f_i}\rvert \leq \frac{\epsilon}{k} F_1^{res(k)}$	Lemma 2.4
	$O(\frac{k}{\epsilon}\frac{2-\gamma}{2(\gamma-1)})$	(γ-decreasing)	$\lvert f_i - \hat{f_i}\rvert \leq \epsilon f_i, \forall i \leq k$	Lemma 2.6
Count-Min [21]	$O(\frac{k}{\epsilon}\log n)$	Turnstile	$\lvert f_i - \hat{f_i}\rvert \leq \frac{\epsilon}{k}\epsilon F_1^{res(k)}$	Never underestimate
CountSketch [13]	$O(\frac{k}{\epsilon}\log n)$	Turnstile	$(f_i - \hat{f_i})^2 \leq \frac{\epsilon}{k} F_2^{res(k)}$	unbiased
DoubleSS± & IntegratedSS±	$O(\frac{\alpha}{\epsilon})$	Bounded Deletion	$\lvert f_i - \hat{f_i}\rvert \leq \epsilon F_1$	Theorem 3.1 & 3.9
Unbiased DoubleSS±	$O(\frac{\alpha}{\epsilon})$		unbiased & Var $< \epsilon^2 F_1^2$	Theorem 3.3
DoubleSS± & IntegratedSS±	$O(\frac{\alpha k}{\epsilon})$		$\lvert f_i - \hat{f_i}\rvert \leq \frac{\epsilon}{k} F_{1.\alpha}^{res(k)}$	Theorem 4.1 & 4.3
DoubleSS± & IntegratedSS±	$O(\frac{\alpha k}{\epsilon}\frac{(2-\gamma)k^\beta}{2(\gamma-1)2^{log_\gamma k}})$	(γ-decreasing and β-zipf)	$\lvert f_i - \hat{f_i}\rvert \leq \epsilon f_i, \forall i \leq k$	Theorem 4.6 & 4.7

Table 1: Comparison between different frequency estimation and approximate heavy hitters algorithms.

the turnstile model, arriving stream items can be either insertions or deletions and the (net) frequency $f(x)$ of x is the difference between the number of times x has been inserted and deleted, with the proviso that no item's frequency is negative. Count-Min [21] and CountSketch [13] are two popular algorithms that operate in the turnstile model. Supporting deletions is a harder task and as a result Count-Min and Count-Sketch require more space with poly-log dependence on the input domain size to offer strong error bounds with high probability [4]. The bounded deletion model [31, 51, 54] assumes the number of deletions are bounded by a fraction of the number of insertions. For instance, in standard testings, if students are only allowed to submit one regrade request, then the number of deletions is at most half of the number of insertions in which updating a student's grade is a deletion followed by an insertion.

Recently, SpaceSaving± [51] proposed an algorithm extending the original SpaceSaving [41] to handle bounded deletions efficiently, (implicitly) assuming [51] that the data stream has no interleaving between insertions and deletions: all insertions precede all deletions [41] [52]. In addition, Berinde et al. [7] showcased that data summaries in the insertion-only model can achieve an even stronger error bound using residual error analysis. The difference among these algorithms and their corresponding error guarantees is listed in Table 1. As a result, in this work, we aim to investigate the SpaceSaving± family of algorithms to support interleavings between insertions and deletions and provide tighter error analysis.

1.2 Our Contributions

In this paper, we present a detailed analysis of the SpaceSaving± family of algorithms with bounded deletions. In particular, we propose three new algorithms: Double SpaceSaving±, Unbiased Double SpaceSaving±, and Integrated SpaceSaving±, which operate in the general bounded deletion model (with interleavings of insertions and deletions) and provide strong theoretical guarantees with small space, while having distinct characteristics. In addition, we introduce the residual error bound guarantee in the bounded deletion model and prove that the proposed algorithms satisfy the stronger residual error guarantee. Moreover, we demonstrate that the proposed algorithms also achieve the relative error bound [20, 40] under mild assumptions, assuming skewness in the data stream. Finally, we provide a merge algorithm for our proposed algorithms and prove that all proposed algorithms are mergeable in the bounded deletion model. The mergeability is critical for distributed applications [1].

2 BACKGROUND

2.1 Preliminaries

Consider a data stream consisting of a sequence of N operations (insertions or deletions) on items drawn from a universe U of size $|U| = n$. We denote the stream by $\sigma = \{(item_t, op_t)\}_{t=\{1,2...N\}}$ where $item_t \in U$ is an element of the universe and $op_t \in \{insertion, deletion\}$. Let I denote the total number of insertions and D denote the total number of deletions in the stream, so that $I + D = N$. In the *bounded deletion model* [31, 54], it is assumed that, for some parameter $\alpha \geq 1$, the stream is constrained to satisfy $D \leq (1 - 1/\alpha)I$. Observe that when $\alpha = 1$, there can be no deletions, and so the model includes the insertion-only model as a special case. Let f denote the frequency function: the frequency of an item x is $f(x) = I(x) - D(x)$ where $I(x)$ denotes the number of insertions of x in the stream and $D(x)$ denotes the number of deletions of x in the stream. In this paper, we use a *summary* data structure from which we infer an estimation of the frequency function. We use $\hat{f}(x)$ to denote the estimated frequency of x.

Let $\{f_i\}_1^n$ denote the frequencies of the n items, sorted in non-increasing order, so that $f_1 \geq f_2, ... \geq f_n$. Similarly, sorting $\{I(x) : x \in U\}$ in non-increasing order leads to $\{I_i\}_1^n$ where $I_1 \geq I_2, ... \geq I_n$, and sorting $\{D(x) : x \in U\}$ in non-increasing order leads to $\{D_i\}_1^n$ where $D_1 \geq D_2, ... \geq D_n$. Notice that, for the same i, the item corresponding to f_i may differ from the item corresponding to I_i and from the item corresponding to D_i. For the special case of the insertion-only model, the frequency vector and insertion vector are identical. Moreover, we use F_1 to denote the total number of items resulting from the stream operations, such that $F_1 = I - D = \sum_{i=1}^n f_i$. In this paper, we focus on the unit updates model (an operation only inserts or deletes one count of an item) and assume that at any point of time, no item frequency is below zero (i.e, items cannot be deleted if they were not previously inserted).

Definition 2.1 (Deterministic Frequency Estimation Problem). Given a stream of operations (insertions or deletions) of items from a universe U, the Frequency Estimation Problem asks for an (implicitly defined) function $\hat{f}$ that when queried for any element $x \in U$,

returns a value $\hat{f}(x)$ such that

$$\forall x \in U, |f(x) - \hat{f}(x)| \leq \epsilon F_1,$$

where $F_1 = \sum_{x \in U} f(x)$.

Definition 2.2 (Heavy Hitters problem). Given a parameter $\epsilon > 0$, a *heavy hitter* is an item with frequency at least ϵF_1. Given a stream of operations (insertions or deletions) of items from a universe U, the *heavy hitters problem* asks for an (explicitly defined) subset of U that contains all heavy hitters.

All of our streaming algorithms use a *summary* of the information seen in the stream so far, consisting of a collection of items and some counts for each of them. The size of the summary is much smaller than $|U|$ (equivalent to n), and we say that an item is *monitored* if it is present in the summary.

2.2 The SpaceSaving± Family of Algorithms

Algorithm 1: SpaceSaving Update Algorithm (insertion-only, one count per item)

Input: Insertion-only data stream σ and SpaceSaving summary S
Output: SpaceSaving summary S

```
for (item e, insertion) from σ do
    if e ∈ S then
        count_e += 1 ;
    else if S not full then
        add e to S with count_e = 1 ;
    else
        minItem = item x ∈ S minimizing count_x;
        w = count_minItem;
        evict minItem from summary;
        add e to summary with count_e = w + 1;
end
```

The SpaceSaving± family of algorithms are data summaries that build on top of the original SpaceSaving [41] algorithm. In 2005, Metwally, Agrawal, and El Abbadi [41] proposed the popular **SpaceSaving** algorithm that provides highly accurate estimates on item frequency and heavy hitters among many competitors [19]. The SpaceSaving algorithm operates in the insertion-only model (so that the stream is simply a sequence of items from U) and uses the optimal space [7]. The update algorithm, as shown in Algorithm 1 [41], uses m counters to store a monitored item's identity and its estimated frequency. When a new $item$ arrives: if $item$ is monitored, then increment its counter; if $item$ is not monitored and the summary has unused counters, then start to monitor $item$, and set $count_{item}$ to 1; otherwise, i) find $minItem$, the item with minimum counter ($minCount$), ii) evicts this $minItem$ from the summary, and iii) monitor the new $item$ and set $count_{item}$ to $minCount + 1$. See Algorithm 1.

After the stream has been read, the SpaceSaving Update algorithm has produced a summary. To solve the Frequency Estimation Problem, the algorithm then needs to estimate the frequency $f(e)$ of an item e being queried: the SpaceSaving Query algorithm returns $\hat{f}(e) = count_e$ if e is monitored and $\hat{f}(e) = 0$ otherwise, as shown in Algorithm 2. (To solve the heavy hitters problem, the algorithm outputs all items in the summary whose count is greater than or equal to ϵ times the sum of the counts.)

When the number of counters (size of the summary) $m = \frac{1}{\epsilon}$, SpaceSaving solves both frequency estimation and heavy hitters problems in the insertion-only model, as shown in Lemma 2.3. Moreover, SpaceSaving satisfies a more refined residual error bound as shown in Lemma 2.4 in which *"hot"* items do not contribute error and the error bound only depends on the *"cold"* and *"warm"* items [7].

The ϵ **error guarantee** is shown in Lemma 2.3. When $m = \frac{1}{\epsilon}$, SpaceSaving ensures $\forall x \in U, |f(x) - \hat{f}(x)| \leq \epsilon F_1$.

Lemma 2.3. *[41] After processing insertion-only data stream σ with the SpaceSaving Update algorithm (Algorithm 1) with m counters, the SpaceSaving Query algorithm (algorithm 2) provides an estimate $\hat{f}$ such that $\forall x \in U, |f(x) - \hat{f}(x)| \leq \frac{F_1}{m}$, where $F_1 = \sum_{i=1}^{n} f_i$.*

The **residual error** guarantee is shown in Lemma 2.4.

Lemma 2.4. *[7] After processing insertion-only data stream σ, SpaceSaving with $O(\frac{k}{\epsilon})$ counters ensure $\forall x \in U, |f(x) - \hat{f}(x)| \leq \frac{\epsilon}{k} F_1^{res(k)}$ where $F_1^{res(k)} = F_1 - \sum_{i=1}^{k} f_i$.*

The **relative error** guarantee is shown in Lemma 2.6.

Definition 2.5 (γ-decreasing). Let $\gamma > 1$, then a non-decreasing function with domain $\{1, 2...n\}$ is γ-decreasing if for all t such that $1 \leq \gamma t \leq n$, $f_{\lceil \gamma t \rceil} \leq f_t/2$

Lemma 2.6. *[40] After processing insertion-only data stream σ and assume the exact frequencies follow the γ-decreasing function (Definition 2.5), SpaceSaving with $O(\frac{k}{\epsilon} \frac{2-\gamma}{2(\gamma-1)})$ counters ensure $|f_i - \hat{f}_i| \leq \epsilon f_i, \forall i \leq k$.*

Algorithm 2: Query Algorithm

Input: SpaceSaving summary S and an item e
Output: e's estimated frequency

```
if e ∈ S then
    return count_e;
return 0;
```

In 2021, Zhao et al. [51] proposed SpaceSaving± to extend the SpaceSaving algorithm from the insertion-only model to the bounded deletion model. In that model, they established a space lower-bound of at least $\frac{\alpha}{\epsilon}$ counters required to solve the frequency estimation and heavy hitters problems, and proposed an algorithm, SpaceSaving± (see Algorithm 3), that used space matching the lower bound. The update algorithm is shown in Algorithm 3 in which insertions are dealt with as in Algorithm 1, deletions of monitored items decrement the corresponding counters, and deletions of unmonitored items are ignored. The query algorithm is identical to Algorithm 2[1].

However, the SpaceSaving± algorithm is only correct if an implicit assumption holds [52], namely, that deletions can only happen

[1]There are a couple of variants of Algorithm 2. We adopt the most commonly used approach.

Algorithm 3: SpaceSaving± Update Algorithm (bounded deletion)

Input: Bounded deletion data stream σ, and SpaceSaving± summary S
Output: SpaceSaving± Summary S

```
1 for (item e, operation op) from σ do
2     if op is an insertion then
3         follow Algorithm 1 to process e;
4     else
5         if e ∈ S then
6             count_e− = 1;
7 end
```

after all the insertions, so that interleaving insertions and deletions are not allowed, as shown in Lemma 2.7. In this paper, we present new algorithms in the general bounded deletions model, that allow interleavings between insertions and deletions while still using only $O(\frac{\alpha}{\epsilon})$ space.

This Lemma correspond to Theorem 2 in [51]. Consider the bounded deletions model and assume, in addition, that the stream consists of an insertion phase with I insertions followed by a deletion phase with D deletions (no interleaving).

Lemma 2.7. *After processing data stream σ with the SpaceSaving± Update algorithm (Algorithm 3) with $m = \frac{\alpha}{\epsilon}$ counters, the SpaceSaving Query algorithm (algorithm 2) provides an estimate $\hat{f}$ such that $\forall x \in U, |f(x) - \hat{f}(x)| \leq \frac{F_1}{m}$, where $F_1 = \sum_{i=1}^{n} f_i$.*

3 NEW ALGORITHMS

In this section, we propose new algorithms supporting the bounded deletion model (with interleavings between insertions and deletions, while an item's frequency is never negative). Double SpaceSaving± (DoubleSS±) and IntegratedSS± (IntegratedSS±) both use $O(\frac{\alpha}{\epsilon})$ space to solve the deterministic frequency estimation and heavy hitters problems in the bounded deletion model. We extended DoubleSS± to support unbiased estimations. Also, we establish the correctness proof for these algorithms. While these algorithm share similar characteristics, they represent different trade-offs. In particular, DoubleSS± uses two disjoint summaries, one to track insertions and the other to track deletions, while IntegratedSS± uses a single summary to track both deletions and insertions.

3.1 Double SpaceSaving±

In DoubleSS±, we employ two SpaceSaving summaries to track insertions (S_{insert}) and deletions (S_{delete}) separately, using m_I and m_D counters respectively (the algorithms is parameterized by the choice of m_I and m_D). As shown in Algorithm 4, DoubleSS± completely partitions the update operations to two independent summaries. To query for the frequency of an item e, both summaries need to be queried to estimate the frequency of item e as shown in Algorithm 5.

Theorem 3.1. *After processing a bounded deletion data stream σ, DoubleSS± following Update Algorithm 4 with $m_I = O(\frac{\alpha}{\epsilon})$ and $m_D = O(\frac{\alpha-1}{\epsilon})$ and Query Algorithm 5 provides an estimate $\hat{f}$ such that $\forall x \in U, |f(x) - \hat{f}(x)| < \epsilon F_1$.*

Algorithm 4: Double SpaceSaving± Update Algorithm

Input: Bounded deletion data stream σ, and Double SpaceSaving± summary (S_{insert}, S_{delete})
Output: Double SpaceSaving± Summary $S = (S_{insert}, S_{delete})$

```
1 for (item e, operation op) from σ do
2     if op is an insertion then
3         Apply Algorithm 1 to insert e in S_insert;
4     else
5         Apply Algorithm 1 to insert e in S_delete;
6 end
```

Algorithm 5: Double SpaceSaving± Query Algorithm

Input: Double SpaceSaving± summary S and an item e
Output: e's estimated frequency.

```
1 inserts = Query S_insert for e with Algorithm 2;
2 deletes = Query S_delete for e with Algorithm 2;
3 return max(inserts - deletes, 0);
```

Proof. For every items in the universe their frequency estimation error is upper bounded by $\frac{I}{m_I} + \frac{D}{m_D}$ due to the property of SpaceSaving as shown in Lemma 2.3 and triangle inequality. In addition, from the definition of bounded deletion, we can derive the relationship between I, D, and F_1. Namely, $I \leq \alpha F_1$ and $D \leq (\alpha-1)F_1$. Set $m_I = \frac{2\alpha}{\epsilon}$ and $m_D = \frac{2(\alpha-1)}{\epsilon}$, then $\frac{I}{m_I} + \frac{D}{m_D} \leq \epsilon F_1$. Hence, DoubleSS± solve the deterministic frequency estimation problem in the bounded deletion model with $O(\frac{\alpha}{\epsilon})$ space. □

Theorem 3.2. *DoubleSS± solves the heavy hitters problem in the bounded deletion model using $O(\frac{\alpha}{\epsilon})$ space by reporting all items monitored in S_1.*

Proof. Assume a heavy hitter x is not monitored in the summary. By definition of heavy hitters, $f(x) \geq \epsilon F_1$. Since x is not monitored, its estimated frequency 0. As a result, the estimation error for item x is larger than ϵF_1 which contradicts Theorem 3.1.

Hence, by contradiction DoubleSS± solves the heavy hitters problem. □

Being more careful and using variants of Algorithm 2 would lead to improvements but would not change the theoretical bound of Theorem 3.2.

3.1.1 Unbiased DoubleSS±. Unbiased estimation is a desirable property for robust estimation [13, 25]. Research has shown that SpaceSaving can be extend to provide unbiased estimation (i.e., $E[\hat{f}(x)] = f(x)$) [45]. The extension is obtained from Algorithm 1 by the following simple modification: after line 8 of the code, with probability $1/(w+1)$ lines 9 and 10 are executed; with the complementary probability $1 - 1/(w+1)$, instead of lines 9 and 10 the algorithm simply increments $count_{minItem}$. This indicates that we can replace the two independent deterministic SpaceSaving summaries with two independent unbiased SpaceSaving summaries in Algorithm 5 to support unbiased frequency estimation in the bounded deletion model.

Theorem 3.3. *After processing a bounded deletion data stream σ, Unbiased DoubleSS± of space $O\frac{\alpha}{\epsilon}$, with two independent unbiased SpaceSaving summaries in Algorithm 5, provides unbiased frequency estimation and the estimation variance is upper bounded by $\epsilon^2 F_1^2$.*

Proof. The variance of an Unbiased SpaceSaving's estimation is upper bounded by $O(\frac{F_1}{m}^2)$ [45]. We know that $Var[X - Y] = Var[X] + Var[Y] - 2(Cov[X, Y])$, where X is the estimation of inserts and Y is the estimation of deletes. Since two summaries are independent, the covariance is 0; Hence, the variance upper bounded by the sum of the $O(\frac{I}{m_I}^2 + \frac{D}{m_D}^2)$. In Theorem 3.1, we proved that $\frac{I}{m_I} + \frac{D}{m_D} \leq \epsilon F_1$. As a result, the variance of the unbiased estimation for any item's frequency is upper bounded by $\epsilon^2 F_1^2$. □

3.2 Integrated SpaceSaving±

Algorithm 6: Integrated SpaceSaving± Update Algorithm

```
Input: Bounded deletion data stream σ, and
       IntergratedSpaceSaving± summary S
Output: IntergratedSpaceSaving± summary S
for (item e, operation op) from σ do
    if e ∈ S then
        if op is an insertion then
            insert_e += 1;
        else
            delete_e += 1;
    else
        if S not full then
            insert_e = 1;
            delete_e = 0;
        else if op is an insertion then
            minItem = item x ∈ S minimizing insert_x;
            w = insert_minItem;
            evict minItem from summary;
            add e to summary and set insert_e = w + 1 and
              delete_e = 0;
end
```

IntegratedSS± completely integrates deletions and insertions in one data structure as shown in Algorithm 6. The main difference between IntegratedSS± and original SpaceSaving± is that IntegratedSS± tracks the insert and delete count separately. This ensures that the minimum insert count monotonically increases and never decrease; however, the original SpaceSaving± uses one single count to track inserts and deletes together. When insertions and deletions are interleaved, then the minimum count in SpaceSaving± may decreases. This can lead to severe underestimation of a frequent item [52].

More specifically, the IntegratedSS± data structure maintains a record estimating the number of insertions and deletions for each monitored item. When a new item arrives, if it is already being monitored, then depending on the type of operation, either the insertion or the deletion count of that item is incremented. If the summary is not full, then no item could have every been evicted from the summary, and since no delete operation arrives before the corresponding item is inserted, then any new item in the stream must be an insertion. Hence, when a new item arrives, and if the summary is not full, the new item is added to the summary and initialized with one insertions and 0 deletions. If a delete arrives and the summary is full, the delete is simply ignored. The interesting case is when an insert arrives and the summary is full. In this case, IntegratedSS± mimics the behavior of the original SpaceSaving± by evicting the element with the least number of inserts and replacing it with the new element. The insertion count for the new element is overestimated by initializing it to the insertion count for the evicted element. However, the deletion count is underestimated by initializing the deletion count to zero.

Algorithm 7: Integrated SpaceSaving± Query Algorithm

```
Input: Integrated SpaceSaving± summary S and an item e
Output: e's estimated frequency.
if e ∈ S then
    return insert_e − delete_e;
return 0;
```

To estimate an item's frequency, if the item is monitored, it subtracts the item's delete count from the item's insert count, otherwise the estimated frequency is 0, as shown in Algorithm 7.

We now establish the correctness of IntegratedSS±. First, we establish three lemmas about IntegratedSS± to help us prove the correctness of the algorithm.

Lemma 3.4. *After processing a bounded deletion data stream σ, Integrated SpaceSaving± following update Algorithm 6 ensures that the sum of all insert counts equals I.*

Proof. This holds by induction over time, as verified by inspection of the algorithm. □

Lemma 3.5. *After processing a bounded deletion data stream σ, IntegratedSS± with m counters following update Algorithm 6 ensures that the minimum insert count, $insert_{minItem}$, is less than or equal to $\frac{I}{m}$.*

Proof. Since deletions never increment any insert counts, insert counts is only affected by insertions. We observe that the sum of all insert counts in summary is exactly equal to I, since the sum of all insert counts increment by 1 after processing one insertion. As a result, $insert_{minItem}$ is maximized when all insert counts are the same. Hence, $insert_{minItem}$ is less than or equal to $\frac{I}{m}$. □

Lemma 3.6. *All monitored items in IntegratedSS± are never underestimated following query Algorithm 7.*

Proof. For the substream of σ consisting only of the insertion operations, these operations are handled the same as the SpaceSaving algorithm. By the analysis of SpaceSaving, it is known that for an item e in the summary, $insert_e$ is an overestimate of the true number of insertions of e since the beginning of the stream.

The $delete_e$ count of element e in the summary is equal to the number of deletions of item e since the last time that it was inserted into the summary, so it is an underestimate of the number of deletions of e since the beginning of the stream.

Thus, for any e in summary, $insert_e - delete_e$ is an overestimate. □

Lemma 3.7. *In IntegratedSS±, any item that has been inserted more than $insert_{minItem}$ must be monitored in the summary; moreover, for any item e in the summary, its count $insert_e$ exceeds its true number of insertions by at most $insert_{minItem}$.*

Proof. The $insert_e$ counts of the summary are dealt with exactly like in the SpaceSaving algorithm, for the substream of σ consisting of the insertion operations only, and the properties are well-known to hold for SpaceSaving. □

Lemma 3.8. *In IntegratedSS± following update Algorithm 6 and query Algorithm 7, the estimation error for any item is upper bounded by $insert_{minItem}$. We can then write: $\forall x \in U, |f(x) - \hat{f}(x)| \leq insert_{minItem}$.*

Proof. First consider the case when e is unmonitored, so the estimate is 0. Its true frequency is at most its number of insertions, and by Lemma 3.7 that is bounded by $insert_{minItem}$, hence the Lemma holds for e.

Now, consider the case when e is in the summary, and consider the last time t at which e was inserted in the Summary. Let $I_{\geq t}(e)$ and $D_{\geq t}(e)$ denote the number of insertions and of deletions of e since time t, and let min denote the value of $insert_{minItem}$ at time t. Since the algorithm updates $insert_e$ and $delete_e$ correctly while e is in the summary, the query algorithm outputs the estimate $\hat{f}(e) = insert_e - delete_e = min + I_{\geq t}(e) - D_{\geq t}(e)$. On the other hand, the frequency $f(e)$ of e equals the net number of occurences of e prior to time t, plus $I_{\geq t}(e) - D_{\geq t}(e)$. By Lemma 3.7 the net number of occurences of e prior to time t is at most min, so $\hat{f}(e) - min \leq f(e) \leq \hat{f}(e)$, implying the Lemma. □

Theorem 3.9. *After processing a bounded deletion data stream σ, IntegratedSpaceSaving$^{\pm}$ with $O(\frac{\alpha}{\epsilon})$ space following update Algorithm 6 and query Algorithm 7 provide an estimate $\hat{f}$ such that $\forall i, |f(i) - \hat{f}(i)| < \epsilon F_1$.*

Proof. By Lemma 3.5 and Lemma 3.8, we known that $\forall x \in U, |f(x) - \hat{f}(x)| \leq insert_{minItem} \leq \frac{I}{m}$. We also know that $I \leq \alpha F_1$. Let's set $m = \frac{\alpha}{\epsilon}$, then we have $\frac{I}{m} \leq \epsilon F_1$. As a result, using $O(\frac{\alpha}{\epsilon})$ space ensures $\forall i, |f(i) - \hat{f}(i)| < \epsilon F_1$. □

IntegratedSS± also solves the heavy hitters problem. Since it never underestimates monitored items (Lemma 3.6), then if we report all the items with frequency estimations greater than or equal to ϵF_1, then all heavy hitters will be identified as shown in Theorem 3.10.

Theorem 3.10. *IntegratedSpaceSaving$^{\pm}$ solves the heavy hitters problem in the bounded deletion model using $O(\frac{\alpha}{\epsilon})$ space.*

Proof. Assume a heavy hitter x is not contained if the set of all items with estimated frequency larger than or equal to ϵF_1. Recall, by definition of heavy hitters, $f(x) \geq \epsilon F_1$. Since x is not contained, x's frequency estimation, $\hat{f}(x)$, must be less than ϵF_1. However, by Lemma 3.6, IntegratedSS± never underestimates the frequency of monitored items.

Hence, by contradiction IntegratedSS± solves the deterministic heavy hitters problem. □

3.3 Comparisons between Double SpaceSaving± and Integrated SpaceSaving±

We assume each field uses the same number of bits (32-bit in the implementation). Following the proofs of Theorem 3.9 and that each entry in the summary has three fields $(i, insert_i, delete_i)$, IntegratedSpaceSaving± uses $3\frac{\alpha}{\epsilon}$. On the other hand, following the proofs of Theorem 3.1 and that each entry in the summary uses two fields, DoubleSS± uses $2\frac{2\alpha}{\epsilon} + 2\frac{2(\alpha-1)}{\epsilon} = \frac{8\alpha-2}{\epsilon}$. IntegratedSS± requires less memory footprint. DoubleSS± also has its own advantage and it can be extended supporting unbiased estimations (Unbiased Double SpaceSaving±).

4 TIGHTER ANALYSIS ON ERROR BOUNDS

In this section, we present tighter analysis of the proposed algorithms. In Section 3, we showcased that the proposed algorithms achieve the desired error bound ϵF_1, in which the error bound depends on all items in data stream σ. In this section, we first prove that both algorithms guarantee the residual error bound and then prove that they both also ensure the relative error bound under relatively mild assumptions. We take inspirations from [7, 40, 41, 51].

4.1 Residual Error Bound

We define $F_{1,\alpha}^{res(k)} = F_1 - \frac{1}{\alpha}\sum_{i=1}^{k} f_i$. Observe that $F_{1,\alpha}^{res(k)}$ is equivalent to the residual error bound shown in Lemma 2.4, $F_1^{res(k)}$, for the insertion-only model, as α is set to 1. The residual error bound ($F_{1,\alpha}^{res(k)}$) is much tighter than the original error bound (F_1), since the residual error depends less on the heavy hitters.

Theorem 4.1. *DoubleSS± with $m_I + m_D = O(\alpha k/\epsilon)$ space achieves the residual error bound guarantee in which $\forall x \in U, |f(x) - \hat{f}(x)| \leq \frac{\epsilon}{k} F_{1,\alpha}^{res(k)}$, where $k < min(m_I, m_D)$.*

Proof. First, let's define the maximum error δ as $\delta = max(\forall x \in U, |f(x) - \hat{f}(x)|)$ for DoubleSS±. We know that $\delta \leq \frac{I - \sum_{i=1}^{k} I_i}{m_I - k} + \frac{D - \sum_{i=1}^{k} D_i}{m_D - k}$ by Lemma 2.4 and triangle inequality. By definition of bounded deletion model, $I \leq \alpha F_1$ and $D \leq (\alpha - 1)F_1$. Hence, we have $\delta \leq \alpha\frac{F_1 - 1/\alpha\sum_{i=1}^{k} I_i}{m_I - k} + (\alpha - 1)\frac{F_1 - 1/(\alpha-1)\sum_{i=1}^{k} D_i}{m_D - k}$. We know that $\sum_{i=1}^{k} I_i \geq \sum_{i=1}^{k} f_i$ (this is because sum of k largest insert count has to be larger than or equal to the sum of k largest frequencies) and $\sum_{i=1}^{k} D_i \geq 0$. As a result, $\delta < \alpha\frac{F_1 - 1/\alpha\sum_{i=1}^{k} f_i}{m_I - k} + (\alpha - 1)\frac{F_1}{m_D - k}$. Let $m_I = k(\frac{2\alpha}{\epsilon} + 1)$ and $m_D = k(\frac{2(\alpha-1)}{\epsilon} + 1)$. Then, $\delta < \frac{\epsilon}{k}(F_1 - 1/(2\alpha)\sum_{i=1}^{k} f_i) \approx \frac{\epsilon}{k} F_{1,\alpha}^{res(k)}$. □

We present the residual error bound for IntegratedSS±. Before presenting the proof, we first construct a helpful Lemma.

Lemma 4.2. *The minimum insert count, in IntegratedSS± with m counters, is less than or equal to $\alpha\frac{F_1 - 1/\alpha\sum_{i=1}^{k} f_i}{m - k}$ where $k < m$.*

Proof. The sum of all insert count equals to I, as the sum of all insert count always increase by 1 after processing an insertion. If we zoom into the k largest insert count in the summary, the sum of their insert counts is no less than $\sum_{i=1}^{k} f_i$, since no monitored items are underestimated as shown in Lemma 3.6. As a result, $insert_{minItem} \leq \frac{I-\sum_{i=1}^{k} f_i}{m-k}$. We know that by definition of bounded deletion, $I \leq \alpha F_1$. Hence, $insert_{minItem} \leq \alpha \frac{F_1 - 1/\alpha \sum_{i=1}^{k} f_i}{m-k}$. □

Theorem 4.3. *IntegratedSS± with $O(\alpha k/\epsilon)$ space achieves the residual error bound guarantee in which $\forall x \in U, |f(x) - \hat{f}(x)| \leq \frac{\epsilon}{k} F_{1,\alpha}^{res(k)}$, where $k < m$.*

Proof. From Lemma 3.8 and Lemma 4.2, we know the estimation error for all items is upper bounded by $insert_{minItem}$ and $insert_{minItem} \leq \alpha \frac{F_1 - 1/\alpha \sum_{i=1}^{k} f_i}{m-k}$. Let $m = k(\frac{\alpha}{\epsilon} + 1)$. We achieve the desired residual error bound: $\forall x \in U, |f(x) - \hat{f}(x)| \leq \frac{\epsilon}{k} F_{1,\alpha}^{res(k)}$. □

4.2 Relative Error Bound

The ϵ **relative error guarantee** is defined such that $\forall x \in U, |f(x) - \hat{f}(x)| \leq \epsilon f(x)$. The relative error has been shown to be much harder to achieve, but with practical importance [20, 27]. In a recent work, Mathieu and de Rougemont [40] showcased that the original SpaceSaving algorithm achieves relative error guarantees under mild assumptions. As a result, we demonstrate that the proposed algorithms in the bounded deletion model also achieve relative error guarantees under similar assumptions.

Lemma 4.4. *Let f be γ decreasing with $1 < \gamma < 2$. Then $\sum_{j=i}^{n} f_j \leq (i-1) f_{i-1} \frac{2(\gamma-1)}{\gamma-1}$ and hence $F_1 = \sum_{j=1}^{n} f_j \leq f_1 (1 + \frac{2(\gamma-1)}{2-\gamma}) = f_1 \frac{\gamma}{2-\gamma}$.*

Proof. We omit the proof. (This is proven in Lemma 5 at [40]) □

Definition 4.5 (Zipfian Distribution [57]). Let a function f with domain $\{1, 2...n\}$ follow the Zipf distribution with parameter β, then $f_i = F_1 \frac{1}{i^\beta \xi(\beta)}$ where $F_1 = \sum_{i=1}^{n} f_i$ and $\xi(\beta) = \sum_{i=1}^{n} i^{-\beta}$.

By the definition of Zipf distribution, we know that $\xi(\beta)$ converges to a small constant when $\beta > 1$ and $f_1 = F_1/\xi(\beta)$.

Theorem 4.6. *Assuming insertions and deletions in the α-bounded deletion model follow the γ-decreasing property ($1 < \gamma < 2$) and the exact frequencies follow the Zipf distribution with parameter β, then DoubleSS± with $O(\frac{\alpha k^{\beta+1}}{\epsilon 2^{log_\gamma k}})$ space achieves the relative error bound guarantee such that $|f_i - \hat{f}_i| \leq \epsilon f_i$, for $i \leq k$.*

Proof. Recall, the maximum estimation error δ is upper bounded by $\frac{I-\sum_{i=1}^{k} I_i}{m_I - k} + \frac{D-\sum_{i=1}^{k} D_i}{m_D - k}$. Since insertions and deletions follow γ-decreasing property, then $\sum_{i=k+1}^{n} I_i \leq k I_k \frac{2(\gamma-1)}{2-\gamma}$ and $\sum_{i=k+1}^{n} D_i \leq k D_k \frac{2(\gamma-1)}{2-\gamma}$. Hence, $\delta \leq \frac{k I_k \frac{2(\gamma-1)}{2-\gamma}}{m_I - k} + \frac{k D_k \frac{2(\gamma-1)}{2-\gamma}}{m_D - k}$.

For insertions, we know i) $I_1 \leq I \leq \alpha F_1 = \alpha \xi(\beta) f_1$, ii) $f_i = \frac{f_1}{i^\beta}$, and iii) $I_i \leq I_1 / 2^{log_\gamma(i)}$; For deletions, we know i) $D_1 \leq D \leq (\alpha - 1) F_1 = (\alpha - 1) \xi(\beta) f_1$, ii) $f_i = \frac{f_1}{i^\beta}$, and iii) $D_i \leq D_1 / 2^{log_\gamma(i)}$.

Hence, $I_k \leq \frac{I_1}{2^{log_\gamma(k)}} \leq \alpha \xi(\beta) f_k k^\beta / 2^{log_\gamma(k)}$, and $D_k \leq D_1 / 2^{log_\gamma(k)} \leq (\alpha - 1) \xi(\beta) f_k k^\beta / 2^{log_\gamma(k)}$. Let $m_I = m_D = k + \frac{2(\gamma-1)}{2-\gamma} \frac{k^{\beta+1}}{2^{log_\gamma k}} \frac{(2\alpha-1)}{\epsilon}$, then $\delta \leq \epsilon f_k$ and hence for all $i < k$, $|f_i - \hat{f}_i| \leq \epsilon f_i$. □

Theorem 4.7. *Assuming insertions in the α-bounded deletion model follows the γ-decreasing property ($1 < \gamma < 2$) and the exact frequencies follow the Zipf distribution with parameter β, then IntegratedSS± with $O(\frac{\alpha k^{\beta+1}}{\epsilon 2^{log_\gamma k}})$ space achieves the relative error bound guarantee such that $|f_i - \hat{f}_i| \leq \epsilon f_i$, for $i \leq k$.*

Proof. We know that estimation error at most $insert_{minItem}$ and $insert_{minItem} \leq \frac{I-\sum_{i=1}^{k} I_i}{m-k}$. Since insertions follow γ-decreasing property, then $\sum_{i=k+1}^{n} I_i \leq k I_k \frac{2(\gamma-1)}{2-\gamma}$. Hence, $insert_{minItem} \leq \frac{\sum_{i=k+1}^{n} I_i}{m-k} \leq \frac{k I_k \frac{2(\gamma-1)}{2-\gamma}}{m-k}$. Since we know i) $I_1 \leq I \leq \alpha F_1 = \alpha \xi(\beta) f_1$, ii) $f_i = \frac{f_1}{i^\beta}$, and iii) $I_g \leq I_1 / 2^{log_\gamma(g)}$. Hence, $I_k \leq I_1 / 2^{log_\gamma(k)} \leq \alpha \xi(\beta) f_1 / 2^{log_\gamma(k)} = \alpha \xi(\beta) f_k k^\beta / 2^{log_\gamma(k)}$. Let $m = k + \frac{2(\gamma-1)}{2-\gamma} \frac{k^{\beta+1}}{2^{log_\gamma k}} \frac{\alpha}{\epsilon}$, then $insert_{minItem} \leq \epsilon f_k$ and hence for all $i < k$, $|f_i - \hat{f}_i| \leq \epsilon f_i$. □

5 MERGEABILITY

Mergeability is a desirable property in distributed settings in which two summaries on two data sets are given, after merging between these two summaries, the merged summary has error and size guarantees equivalent to a single summary which processed the union of two data sets.

Definition 5.1 (Mergeable [2]). A summary S is mergeable if there exists a merge algorithm A that produces a summary $S(\sigma_1 \cup \sigma_2, \epsilon)$ from two input summaries $S(\sigma_1, \epsilon)$ and $S(\sigma_2, \epsilon)$. All three summaries have the same size.

Both the original SpaceSaving and the Unbiased SpaceSaving have been shown to be mergable [2, 45]. Since DoubleSS± relies on two independent summaries, Double SS± is also mergeable.

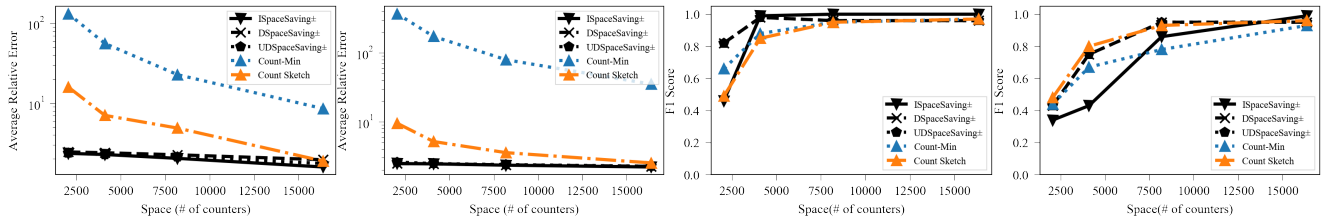
The merge algorithm for IntegratedSS± is shown in Algorithm 8. Given two IntegratedSS± summaries of m counters, S_1 and S_2, we first union the two summaries (i,e, if both summaries monitor item x, then x's insert and delete counts in the merge summary are the sum of x's insert and delete counts from S_1 and S_2). After the union, the merged summary contain at most $2m$ counters. The merged summary consists of the m largest items among these $2m$ counters based on the insert counts.

Algorithm 8: Integrated SpaceSaving± Merge Algorithm

Input: Two Integrated SpaceSaving± summaries of size m: S_1 and S_2
Output: Merged summary S_{merge} with size m

```
S3 = S1 UNION S2;
select m largest items from S3 based on insert counts;
add these m items into S_merge and keep insert/delete counts;
```

Theorem 5.2. *The IntegratedSpaceSaving± summaries are mergeable for α bounded deletion data stream following the Algorithm 8 with $m = \frac{\alpha}{\epsilon}$ counters.*



(a) Zipf(1.0), Frequency Estimation **(b) YCSB, Frequency Estimation** **(c) Zipf(1.0), Top 100** **(d) YCSB, Top 100**

Figure 1: Comparison between our proposed algorithms with linear sketches over synthetic and real world dataset.

Proof. Assume the given *IntegratedSpaceSaving*± summaries with $m = \frac{\alpha}{\epsilon}$ counters, S_1 and S_2, have processed two α bounded deletion data stream σ^1 and σ^2 with I^1 and I^2 insertions and F_1^1 and F_1^2 total frequencies respectively. Let $F_1^{final} = F_1^1 + F_1^2$. First, we observe that The union step do not introduce any additional error. As a result, the error for these items is at most $\epsilon F_1^1 + \epsilon F_1^2 \leq \epsilon F_1^{final}$.

Since we don't want to underestimate any monitored items, Algorithm 8 keeps the largest m items from the union-ed summary based on insert counts. We need to showcase that evicting all items except the largest m items will not lead to error estimation larger than $\frac{I^1+I^2}{m}$, and we know that $\frac{I^1+I^2}{m} \leq \epsilon F_1^{final}$. This can be shown by contradiction. Assuming there exists an item x in the union of S_1 and S_2 with true frequency larger or equal to $\frac{I^1+I^2}{m}$ and x is not included than the m largest items. Note, x's insert count must be larger than its true frequency by Lemma 3.6. Then this must imply the sum of the insert counts of m largest item and x is at least $(m+1) \cdot \frac{I^1+I^2}{m} > (I^1 + I^2)$. However, by Lemma 3.4, we know the sum of insert counts cannot exceed $I^1 + I^2$. As a result, evicting all items except the largest m items from the union-ed summary based on insert counts does not lead to estimation error larger than ϵF_1^{final}. Therefore, IntegratedSS± is mergeable.

□

6 EVALUATION

This section evaluates and compares the performance of our proposed algorithms with linear sketch from the turnstile model [37]. Our proposed counter based summaries have no assumption on the input universe but operates in the bounded deletion model. whereas linear sketch have dependency on the universe size but allow the entire data set to be deleted. The experiments aim to better understand the advantage and disadvantages of our proposed algorithms. The linear sketches that we compared with are:

- **Count Min** [21]: Item's frequency is never underestimated.
- **Count Sketch** [13]: Provides an unbiased estimation, such that $E[\hat{f}(x)] = f(x)$ where E is the expected value.

Experiment Setup. We implemented all of the algorithms in Python. Through all experiments, we set $\delta = U^{-1}$ for linear sketches to align the experiments with the theoretical literature [8].

Data Sets. We use both synthetic and real world data consisting of items that are inserted and deleted. **Zipf Stream**: The insertions (10^5) are drawn from the Zipf distribution [57] and deletions ($10^5/2$) are uniformly chosen from the insertions. Deletions happen after all the insertions. Cardinality is 22k. **YCSB**: We use 60% insertions and 40% updates (delete followed by insert) with request distribution set to *zipf* in YCSB benchmark [17] with interleaved 116645 insertions and 39825 deletions. Cardinality is 65k.

Metrics. We use average relative error (ARE) as the metrics for frequency estimation. Let the final data set $D = \{x | f(x) > 0\}$. ARE is calculated as $1/|D|(\sum_{x \in D} \frac{|f(x) - \hat{f}(x)|}{f(x)})$. Lower ARE indicates more accurate approximations. For identifying the top 100 heavy hitters, we query all items in D and report a set of 100 items with the largest frequency. Metrics is the F1 score.

6.1 Main Results

Our proposed algorithms perform very well on frequency estimation task, as shown in Figure 1(a)-(b). The y-axis is the average relative error and the x-axis the the number of counter used. For fair comparisons, we let IntergratedSpaceSaving± uses three counters per entry, SpaceSaving and unbiased SpaceSaving use two counters, and linear sketch uses one counter per entry. We find that Integrated SpaceSaving± is always the best, followed by Double SpaceSaving± (DSS±) or Unbiased Double SpaceSaving± ((UDSS±)), and Count Sketch always ranks at the fourth. For instance, when using 16384 counters over YCSB dataset, average relative error for IntergratedSpaceSaving± is 2.20, DSS± is 2.25, UDSS± is 2.26, Count Sketch is 2.29, and Count-Min is 35.

In identifying the top 100 items, linear sketches are more competitive to our proposed algorithms. As shown in Figure 1(c)-(d), the y-axis is the F1 score and x-axis is the number of counters. When more memory budget is allowed, Integrated SpaceSaving± is always the best. When the space budget up to 8192 counters, DSS± and UDSS± always outperform linear sketches. For instance, using 8192 counters over YCSB, DSS± and UDSS± both achieve 0.95 F1 score, whereas Count Sketch score 0.91 and Count Min scores 0.78.

7 CONCLUSION

This paper presents a detailed analysis of SpaceSaving± family of algorithms with bounded deletion. We proposed new algorithms built on top of the original SpaceSaving and SpaceSaving±. They exhibit many desirable properties such as no underestimation, unbiased estimation, residual/ relative error guarantees and mergability. We believe these unique characteristics make our proposed algorithms a strong candidate for real-world applications and systems.

REFERENCES

[1] Pankaj K Agarwal, Graham Cormode, Zengfeng Huang, Jeff Phillips, Zhewei Wei, and Ke Yi. 2012. Mergeable summaries. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*. 23–34.

[2] Pankaj K Agarwal, Graham Cormode, Zengfeng Huang, Jeff M Phillips, Zhewei Wei, and Ke Yi. 2013. Mergeable summaries. *ACM Transactions on Database Systems (TODS)* 38, 4 (2013), 1–28.

[3] Divyakant Agrawal, Philip Bernstein, Elisa Bertino, Susan Davidson, Umeshwas Dayal, Michael Franklin, Johannes Gehrke, Laura Haas, Alon Halevy, Jiawei Han, et al. 2011. Challenges and opportunities with Big Data 2011-1. (2011).

[4] Noga Alon, Yossi Matias, and Mario Szegedy. 1996. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 20–29.

[5] Peter Bailis, Edward Gan, Samuel Madden, Deepak Narayanan, Kexin Rong, and Sahaana Suri. 2017. Macrobase: Prioritizing attention in fast data. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 541–556.

[6] Omri Ben-Eliezer, Rajesh Jayaram, David P Woodruff, and Eylon Yogev. 2022. A framework for adversarially robust streaming algorithms. *ACM Journal of the ACM (JACM)* 69, 2 (2022), 1–33.

[7] Radu Berinde, Piotr Indyk, Graham Cormode, and Martin J Strauss. 2010. Space-optimal heavy hitters with strong error bounds. *ACM Transactions on Database Systems (TODS)* 35, 4 (2010), 1–28.

[8] Arnab Bhattacharyya, Palash Dey, and David P Woodruff. 2018. An optimal algorithm for l1-heavy hitters in insertion streams and related problems. *ACM Transactions on Algorithms (TALG)* 15, 1 (2018), 1–27.

[9] Burton H Bloom. 1970. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* 13, 7 (1970), 422–426.

[10] Vladimir Braverman, Stephen R Chestnut, Nikita Ivkin, Jelani Nelson, Zhengyu Wang, and David P Woodruff. 2017. BPTree: an l2 heavy hitters algorithm using constant memory. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 361–376.

[11] Andrei Z Broder. 1997. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*. IEEE, 21–29.

[12] Sebastian Bruch, Franco Maria Nardini, Amir Ingber, and Edo Liberty. 2023. An Approximate Algorithm for Maximum Inner Product Search over Streaming Sparse Vectors. *arXiv preprint arXiv:2301.10622* (2023).

[13] Moses Charikar, Kevin Chen, and Martin Farach-Colton. 2002. Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming*. Springer, 693–703.

[14] Wei-Ning Chen, Ayfer Ozgur, Graham Cormode, and Akash Bharadwaj. 2023. The communication cost of security and privacy in federated frequency estimation. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 4247–4274.

[15] Monica Chiosa, Thomas B Preußer, and Gustavo Alonso. 2021. Skt: A one-pass multi-sketch data analytics accelerator. *Proceedings of the VLDB Endowment* 14, 11 (2021), 2369–2382.

[16] Edith Cohen, Xin Lyu, Jelani Nelson, Tamás Sarlós, Moshe Shechner, and Uri Stemmer. 2022. On the robustness of countsketch to adaptive inputs. In *International Conference on Machine Learning*. PMLR, 4112–4140.

[17] Brian F Cooper, Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, and Russell Sears. 2010. Benchmarking cloud serving systems with YCSB. In *Proceedings of the 1st ACM symposium on Cloud computing*. 143–154.

[18] Graham Cormode. 2023. Applications of Sketching and Pathways to Impact. In *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems* (Seattle, WA, USA) *(PODS '23)*. Association for Computing Machinery, New York, NY, USA, 5–10. https://doi.org/10.1145/3584372.3589937

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

[20] Graham Cormode, Zohar Karnin, Edo Liberty, Justin Thaler, and Pavel Veselỳ. 2021. Relative error streaming quantiles. In *Proceedings of the 40th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 96–108.

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

[22] Graham Cormode and Ke Yi. 2020. *Small summaries for big data*. Cambridge University Press.

[23] Niv Dayan, Manos Athanassoulis, and Stratos Idreos. 2017. Monkey: Optimal navigable key-value store. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 79–94.

[24] Erik D Demaine, Alejandro López-Ortiz, and J Ian Munro. 2002. Frequency estimation of internet packet streams with limited space. In *European Symposium on Algorithms*. Springer, 348–360.

[25] 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–es.

[26] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. 2007. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. *Discrete mathematics & theoretical computer science* Proceedings (2007).

[27] Anupam Gupta and Francis Zane. 2003. Counting inversions in lists. In *SODA*, Vol. 3. 253–254.

[28] Jiawei Han, Jian Pei, and Hanghang Tong. 2006. Data mining: Concepts and techniques. *Morgan Kaufinann* 10, 559-569 (2006), 4.

[29] Avinatan Hasidim, Haim Kaplan, Yishay Mansour, Yossi Matias, and Uri Stemmer. 2020. Adversarially robust streaming algorithms via differential privacy. *Advances in Neural Information Processing Systems* 33 (2020), 147–158.

[30] Nikita Ivkin, Edo Liberty, Kevin Lang, Zohar Karnin, and Vladimir Braverman. 2019. Streaming quantiles algorithms with small space and update time. *arXiv preprint arXiv:1907.00236* (2019).

[31] Rajesh Jayaram and David P Woodruff. 2018. Data streams with bounded deletions. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 341–354.

[32] Jiawei Jiang, Fangcheng Fu, Tong Yang, and Bin Cui. 2018. Sketchml: Accelerating distributed machine learning with data sketches. In *Proceedings of the 2018 International Conference on Management of Data*. 1269–1284.

[33] Zohar Karnin, Kevin Lang, and Edo Liberty. 2016. Optimal quantile approximation in streams. In *2016 ieee 57th annual symposium on foundations of computer science (focs)*. IEEE, 71–78.

[34] Richard M Karp, Scott Shenker, and Christos 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.

[35] Christian Janos Lebeda and Jakub Tetek. 2023. Better differentially private approximate histograms and heavy hitters using the Misra-Gries sketch. In *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 79–88.

[36] Jizhou Li, Zikun Li, Yifei Xu, Shiqi Jiang, Tong Yang, Bin Cui, Yafei Dai, and Gong Zhang. 2020. Wavingsketch: An unbiased and generic sketch for finding top-k items in data streams. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1574–1584.

[37] Yi Li, Huy L Nguyen, and David P Woodruff. 2014. Turnstile streaming algorithms might as well be linear sketches. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 174–183.

[38] Zirui Liu, Yixin Zhang, Yifan Zhu, Ruwen Zhang, Tong Yang, Kun Xie, Sha Wang, Tao Li, and Bin Cui. 2023. TreeSensing: Linearly Compressing Sketches with Flexibility. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–28.

[39] Gurmeet Singh Manku and Rajeev Motwani. 2002. Approximate frequency counts over data streams. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Elsevier, 346–357.

[40] Claire Mathieu and Michel de Rougemont. 2023. Testing frequency distributions in a stream. *arXiv preprint arXiv:2309.11175* (2023).

[41] Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. 2005. Efficient computation of frequent and top-k elements in data streams. In *Database Theory-ICDT 2005: 10th International Conference, Edinburgh, UK, January 5-7, 2005. Proceedings 10*. Springer, 398–412.

[42] Jayadev Misra and David Gries. 1982. Finding repeated elements. *Science of computer programming* 2, 2 (1982), 143–152.

[43] Rasmus Pagh and Mikkel Thorup. 2022. Improved utility analysis of private countsketch. *Advances in Neural Information Processing Systems* 35 (2022), 25631–25643.

[44] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. 2020. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*. PMLR, 8253–8265.

[45] Daniel Ting. 2018. Data sketches for disaggregated subset sum and frequent item estimation. In *Proceedings of the 2018 International Conference on Management of Data*. 1129–1140.

[46] Pinghui Wang, Yiyan Qi, Yuanming Zhang, Qiaozhu Zhai, Chenxu Wang, John CS Lui, and Xiaohong Guan. 2019. A memory-efficient sketch method for estimating high similarities in streaming sets. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 25–33.

[47] Yiping Wang, Yanhao Wang, and Cen Chen. 2024. DPSW-Sketch: A Differentially Private Sketch Framework for Frequency Estimation over Sliding Windows (Technical Report). *arXiv preprint arXiv:2406.07953* (2024).

[48] Yuchen Xu, Wenfei Wu, Bohan Zhao, Tong Yang, and Yikai Zhao. 2023. MimoSketch: A Framework to Mine Item Frequency on Multiple Nodes with Sketches. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2838–2849.

[49] Tong Yang, Jie Jiang, Peng Liu, Qun Huang, Junzhi Gong, Yang Zhou, Rui Miao, Xiaoming Li, and Steve Uhlig. 2018. Elastic sketch: Adaptive and fast network-wide measurements. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 561–575.

[50] Victor Zakhary, Lawrence Lim, Divy Agrawal, and Amr El Abbadi. 2021. Cache on Track (CoT): Decentralized Elastic Caches for Cloud Environments. In *International Conference on Extending Database Technology*.

[51] Fuheng Zhao, Divyakant Agrawal, Amr El Abbadi, and Ahmed Metwally. 2021. SpaceSaving±: An Optimal Algorithm for Frequency Estimation and Frequent

items in the Bounded Deletion Model. *arXiv preprint arXiv:2112.03462* (2021).

[52] Fuheng Zhao, Divyakant Agrawal, Amr El Abbadi, Ahmed Metwally, Claire Mathieu, and Michel de Rougemont. 2023. Errata for" SpaceSaving±: An Optimal Algorithm for Frequency Estimation and Frequent Items in the Bounded-Deletion Model". *Proceedings of the VLDB Endowment* 17, 4 (2023), 643–643.

[53] Fuheng Zhao, Punnal Ismail Khan, Divyakant Agrawal, Amr El Abbadi, Arpit Gupta, and Zaoxing Liu. 2023. Panakos: Chasing the Tails for Multidimensional Data Streams. *Proceedings of the VLDB Endowment* 16, 6 (2023), 1291–1304.

[54] Fuheng Zhao, Sujaya Maiyya, Ryan Wiener, Divyakant Agrawal, and Amr El Abbadi. 2021. Kll± Approximate Quantile Sketches over Dynamic Datasets. *Proceedings of the VLDB Endowment* 14, 7 (2021), 1215–1227.

[55] Fuheng Zhao, Dan Qiao, Rachel Redberg, Divyakant Agrawal, Amr El Abbadi, and Yu-Xiang Wang. 2022. Differentially private linear sketches: Efficient implementations and applications. *Advances in Neural Information Processing Systems* 35 (2022), 12691–12704.

[56] Fuheng Zhao, Leron Reznikov, Divyakant Agrawal, and Amr El Abbadi. 2023. Autumn: A Scalable Read Optimized LSM-tree based Key-Value Stores with Fast Point and Range Read Speed. *arXiv preprint arXiv:2305.05074* (2023).

[57] George Kingsley Zipf. 2016. *Human behavior and the principle of least effort: An introduction to human ecology.* Ravenio Books.

Differentially Private Linear Sketches: Efficient Implementations and Applications

Fuheng Zhao[*†] fuheng_zhao@ucsb.edu
Dan Qiao[*†] danqiao@ucsb.edu
Rachel Redberg[*] rredberg@ucsb.edu
Divyakant Agrawal[*] agrawal@cs.ucsb.edu
Amr El Abbadi[*] amr@cs.ucsb.edu
Yu-Xiang Wang[*] yuxiangw@ucsb.edu

Abstract

Linear sketches have been widely adopted to process fast data streams, and they can be used to accurately answer frequency estimation, approximate top K items, and summarize data distributions. When data are sensitive, it is desirable to provide privacy guarantees for linear sketches to preserve private information while delivering useful results with theoretical bounds. We show that linear sketches can ensure privacy and maintain their unique properties with a small amount of noise added at initialization. From the differentially private linear sketches, we showcase that the state-of-the-art quantile sketch in the turnstile model can also be private and maintain high performance. Experiments further demonstrate that our proposed differentially private sketches are quantitatively and qualitatively similar to noise-free sketches with high utilization on synthetic and real datasets.

1 Introduction

Data sketches are fundamental tools for data analysis, statistics, and machine learning [Cormode and Yi, 2020]. Two of the most widely studied problems in data summaries are frequency estimation and quantile approximation. Many real world applications need to estimate the frequency of each item in the database and understand the overall distribution of the database. These applications include stream processing [Das et al., 2009, Bailis et al., 2017], database management [Misra and Gries, 1982, Metwally et al., 2005, Zhao et al., 2022], caching [Zakhary et al., 2020], system monitoring [Gupta et al., 2016, Ivkin et al., 2019, Zhao et al., 2021], federated learning [Rothchild et al., 2020], among others.

On one hand, the motivation for data sketch algorithms is to efficiently process a large database and extract useful knowledge, since computing the exact information for a large amount of data is both time and memory intensive. For instance, Munro and Paterson [1980] proved that to find the true median of a database with n items using p sequential passes requires at least $\Omega(n^{1/p})$ memory. On the other hand, to protect user-level privacy, privacy-preserving algorithms limit the disclosure of private information in the database so that an observer cannot infer much about an individual. Recent works have shown that data sketches can be integrated with privacy-enhancing technologies to provide insightful information and preserve individual privacy at the same time [Cormode, 2022].

Differential privacy [Dwork et al., 2006] is a widely-accepted definition of privacy. Recently, researchers have observed that some data sketches are inherently differentially private [Blocki et al., 2012, Smith et al., 2020], while many other data sketches need modifications to the algorithm to be

[*]Department of Computer Science, UC Santa Barbara.
[†]The first two authors contributed equally.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

differentially private. In particular, a substantial amount of literature has focused on differentially private data sketches for tasks such as linear algebra [Upadhyay, 2014, Arora et al., 2018], cardinality estimation [Mir et al., 2011, Pagh and Stausholm, 2021, Dickens et al., 2022] and quantile approximation [Tzamos et al., 2020, Gillenwater et al., 2021, Alabi et al., 2022].

In this paper, we introduce new differentially private algorithms that support both insertions and deletions for frequency, top k, and quantile approximation. While many data sketches assume an insertion-only model [Greenwald and Khanna, 2001, Shrivastava et al., 2004, Karnin et al., 2016] or a bounded-deletion model [Jayaram and Woodruff, 2018, Zhao et al., 2022, 2021], our algorithms build on top of linear sketches [Charikar et al., 2002, Cormode and Muthukrishnan, 2005] and operate in the turnstile model, which allows an arbitrary number of insertions and deletions into the database. Earlier, researchers attempted to prove CountSketch [Charikar et al., 2002] itself preserves differential privacy, but the authors acknowledged that there are issues in the proof [Li et al., 2019]. Instead of proving that linear sketches, i.e., both Count-Min and CountSketch, are inherently differentially private, we add a small amount of Gaussian noise at their initialization to provide a privacy guarantee, while maintaining linear sketches' original properties, providing high utility for frequency and top K estimations, and keeping update and query algorithms unchanged. We also demonstrate that our analysis provides the tight uniform bound (Section 3.1 and Appendix E). In addition, we propose the first differentially private quantile sketch in the turnstile model by leveraging the differentially private linear sketch. Our differentially private sketches can be queried an arbitrary number of times without affecting privacy guarantees based on the post-processing immunity. Following prior works [Choi et al., 2020, Smith et al., 2020], we assume ideal random hash functions exist, and this assumption can be replaced in practice by cryptographic hash functions [Dickens et al., 2022].

2 Preliminaries

Consider a database $X = \{i_t\}_{t\in[N]}$ of N items that are drawn from a large $universe$ of size U, such as IPv4 address of size 2^{32}, and for each insert or delete operations, one item can be inserted into or deleted from the database X. To support ordered statistic such as quantile, we assume that the $universe$ is some finite totally ordered data universe.

Definition 2.1. *Given a database X, the frequency of an item x is $f(x) = \sum_{t=1}^{N} \pi(i_t = x)$ where π returns 1 if i_t is x, and 0 otherwise.*

Definition 2.2. *Given a database X of items drawn from an ordered universe, the rank of an item x is $R(x) = \sum_{t=1}^{N} \pi(i_t \leq x)$ where π returns 1 if i_t is less or equal to x and 0 otherwise.*

Given the large size of N, calculating the actual statistics, such as frequency and quantile, is often hard, and hence most applications are satisfied with an *approximation*. The *randomized frequency estimation problem* takes an accuracy parameter γ and a failure probability β such that, for any item x, $|\widehat{f}(x) - f(x)| \leq \gamma \cdot N$ with high probability $1-\beta$, where $\widehat{f}(x)$ is the estimated frequency and $f(x)$ is the true frequency [Cormode and Hadjieleftheriou, 2008]. In addition, the *randomized quantile approximation problem* also takes an accuracy parameter γ and a failure probability β such that, for any item x, $|\widehat{R}(x) - R(x)| \leq \gamma \cdot N$ with high probability $1-\beta$ where $\widehat{R}(x)$ is the estimated rank and $R(x)$ is the actual rank [Karnin et al., 2016].

2.1 Differential Privacy

Definition 2.3. *Databases X and X' are neighbors ($X \sim X'$), if they differ in at most one element.*

Through this paper, we assume the $update/replace$ definition of differential privacy instead of $add/remove$ definition of differential privacy, in which one item in X is updated or replaced by another item in X' [Vadhan, 2017].

Definition 2.4 (Differential Privacy [Dwork et al., 2006])**.** *A randomized algorithm M satisfies (ϵ, δ)-differential privacy ((ϵ, δ)-DP) if for all neighboring databases X, X' and for all possible events E in the output range of M, we have*

$$\mathbb{P}(M(X) \in E) \leq e^{\epsilon} \cdot \mathbb{P}(M(X') \in E) + \delta.$$

When $\delta = 0$, ϵ-DP is known as pure DP, and when $\delta > 0$, (ϵ, δ)-DP is known as approximate DP.

Definition 2.5 (Gaussian Mechanism [Dwork et al., 2006]). *Define the ℓ_2 sensitivity of a function $f : \mathbb{N}^{\mathcal{X}} \mapsto \mathbb{R}^d$ as*

$$\Delta_2(f) = \sup_{\text{neighboring } X, X'} \|f(X) - f(X')\|_2.$$

The Gaussian mechanism $\mathcal{M}$ with noise level σ is then given by

$$\mathcal{M}(X) = f(X) + \mathcal{N}(0, \sigma^2 I_d).$$

Specifically, the Gaussian mechanism is known to satisfy a stronger notion of privacy known as zero-concentrated differential privacy (zCDP, defined below); zCDP lies between pure and approximate DP and can be translated into standard DP notations, as shown in Lemma 2.9. Moreover, zCDP satisfies cleaner composition theorems, as shown in Lemma 2.7.

Definition 2.6 (zCDP [Dwork and Rothblum, 2016, Bun and Steinke, 2016]). *A randomized mechanism M satisfies ρ-Zero-Concentrated Differential Privacy (ρ-zCDP), if for all neighboring databases X, X' and all $\alpha \in (1, \infty)$,*

$$D_\alpha(M(X) \| M(X')) \leq \rho\alpha,$$

where D_α is the Renyi divergence [Van Erven and Harremos, 2014].

Lemma 2.7 (Adaptive composition and Post Processing of zCDP [Bun and Steinke, 2016]). *Let $M : \mathcal{X}^n \to \mathcal{Y}$ and $M' : \mathcal{X}^n \times \mathcal{Y} \to \mathcal{Z}$. Suppose M satisfies ρ-zCDP and M' satisfies ρ'-zCDP (as a function of its first argument). Define $M'' : \mathcal{X}^n \to \mathcal{Z}$ by $M''(x) = M'(x, M(x))$. Then M'' satisfies $(\rho + \rho')$-zCDP.*

Lemma 2.8 (Privacy Guarantee of Gaussian mechanism [Dwork et al., 2014, Bun and Steinke, 2016]). *Let $f : \mathbb{N}^{\mathcal{X}} \mapsto \mathbb{R}^d$ be an arbitrary d-dimensional function with ℓ_2 sensitivity $\Delta_2 = \sup_{\text{neighboring } X, X'} \|f(X) - f(X')\|_2$. Then for any $\rho > 0$, Gaussian Mechanism with parameter $\sigma^2 = \frac{\Delta_2^2}{2\rho}$ satisfies ρ-zCDP.*

Lemma 2.9 (Converting zCDP to DP [Bun and Steinke, 2016]). *If M satisfies ρ-zCDP then M satisfies $(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta)$-DP.*

As we use exclusively Gaussian mechanisms and their composition in our proposed algorithms, our method actually satisfies (ϵ, δ)-DP guarantees with stronger parameters than what is implied by zCDP via techniques from [Balle and Wang, 2018, Dong et al., 2019], which reduces the ϵ parameter by a sizable fraction in typical parameter regimes. We stick to zCDP for clarity and generality, because all our results would apply without changes if we modify the noise into other mechanisms satisfying zCDP, e.g., the Discrete Gaussian Mechanism [Canonne et al., 2020].

2.2 Revisiting Linear Sketches

Charikar et al. [2002] proposed the **CountSketch**, a randomized algorithm that summarizes a database and solves the frequency estimation problem. The CountSketch uses a $d \times w$ array of counters, i.e, C[d, w], where all the counters are initialized to **zero**, and has two sets of independent hash functions h and g. For each row r, the hash function h_r maps input items uniformly onto $\{1, \ldots, w\}$ and the hash function g_r maps input items uniformly onto $\{-1, +1\}$. For item x with value $v \in \{-1, +1\}$, CountSketch updates d counters, one per each row, based on the hash values such that for a particular row r, $g_r(x)$ will be added or subtracted to the counter at the $h_r(x)^{th}$ index depending on whether x is being inserted or deleted respectively, as shown in Algorithm 1. Hence, the update time is $O(d)$. To estimate the frequency of item x, CountSketch will output the $\text{median}_{1 \leq r \leq d}\, g_r(x) \cdot C[r, h_r(x)]$, as shown in Algorithm 2. By updating each row's counter based on the hashed value of either 1 or -1 and reporting the median for query, CountSketch provides an unbiased estimate. To reduce the failure probability of bad estimations, d is set to $O(\log(1/\beta))$ and it uses $O(\frac{1}{\gamma}\log(\frac{1}{\beta}))$ space to solve the frequency estimation problem.

Algorithm 1 Linear Sketch Update(x, v)

1: **Input:** Item x with value $v \in \{-1, +1\}$, counter arrays C, and two sets of hash functions $\{h_1, \ldots, h_{C.rows}\}$ and $\{g_1, \ldots, g_{C.rows}\}$.
2: **for** $r \leftarrow 1, 2, \ldots, C.rows$ **do**
3: $\quad C[r, h_r(x)] \leftarrow C[r, h_r(x)] + v \cdot g_r(x)$
4: **end for**
5: **Output:** C.

Cormode and Muthukrishnan [2005] proposed the **Count-Min** sketch that shares the same initialization, update, and data structure as CountSketch. Count-Min sketch also uses $O(\frac{1}{\gamma}\log(\frac{1}{\beta}))$ space to solve the frequency estimation problem. A major difference is that Count-Min sketch makes all hash functions in set g return positive 1. As a result, to estimate the frequency of item x, Count-Min sketch returns $\min_{1\le r\le d} C[r, h_r(x)]$ instead of the median, as shown in Algorithm 2. In addition, it has the nice property of never underestimating item's frequency. Since linear sketches can approximate an item's frequency accurately, they also solves the top K approximation problem by returning the K items associated with the highest estimated frequency.

Algorithm 2 Linear Sketch Query(x)

Input: Item x, counter arrays C, and two sets of hash functions $\{h_1, \ldots, h_{C.rows}\}$ and $\{g_1, \ldots, g_{C.rows}\}$.
arr $\leftarrow$ [ ]
for $r \leftarrow 1, 2, \ldots, C.rows$ **do**
    arr.append$(g_r(x) \cdot C[r, h_r(x)])$
end for
Output: $\min(\text{arr})$ **for Count-Min or** $\text{median}(\text{arr})$ **for CountSketch.**

Gilbert et al. [2002] made the connection between frequency and quantiles, in which the quantile range can be decomposed into at most $\log U$ dyadic intervals [Cormode et al., 2019] and the sum of the estimated frequencies for these intervals gives the estimated rank. Wang et al. [2013] leveraged the unbiased property of CountSketch and proposed the Dyadic CountSketch (DCS) to estimate the frequencies of each dyadic interval. For more specific details, Appendix B and [Cormode and Yi, 2020] provide a comprehensive analysis of quantile sketches.

3 Private Linear Sketches

In this section, we present new algorithms for differentially private linear sketches. We highlight that our Private Count-Min and CountSketch only require a different initialization while they share the same update (Algorithm 1) and query (Algorithm 2) with the original Count-Min and CountSketch. Therefore, the implementation of our algorithms is efficient. Below we show our private initialization.

Algorithm 3 DP Linear Sketch Initialization with Gaussian Noise

Input: Desired accuracy parameter γ, failure probability β, and budget for zCDP ρ.
Initialize Counter Arrays
$\sigma \leftarrow \sqrt{\log(2/\beta)/\rho}$
$E \leftarrow \sqrt{\frac{2\log\frac{2}{\beta}}{\rho}} \cdot \sqrt{\log\frac{\frac{4}{\gamma}\log(\frac{2}{\beta})}{\beta}}$
for $r \leftarrow 1, 2, \ldots, \log(2/\beta)$ **do**
    for $c \leftarrow 1, 2, \ldots, 1/\gamma$ **do**
        $C[r, c] \leftarrow \mathcal{N}(0, \sigma^2)$ if Private CountSketch
        $C[r, c] \leftarrow E + \mathcal{N}(0, \sigma^2)$ if Private Count-Min
    end for
end for
Output: C.

In Algorithm 3, the set of arrays we use is C which consists of $\log(2/\beta)$ arrays with length $1/\gamma$, which has the same space complexity as original Count-Min and CountSketch. Recall that two neighboring databases X and X' differ by at most one item. Therefore, after updating all the items respectively, for each corresponding array in $C(X)$ and $C(X')$, they differ by at most two elements and the difference is at most 1. Then the ℓ_2-sensitivity of the set of arrays C is bounded by

$$\Delta_2 = \sqrt{2\log(2/\beta)}. \tag{1}$$

By applying the Gaussian Mechanism (Definition 2.5), we can add *independent* Gaussian noises $\mathcal{N}(0, \sigma^2)$ to each counter in C, where $\sigma = \sqrt{\frac{\log(2/\beta)}{\rho}}$. Due to the privacy guarantee of Gaussian Mechanism (Lemma 2.8), it satisfies $\frac{\Delta_2^2}{2\sigma^2} = \rho$-zCDP.

Define $E(\beta, \gamma, \rho) = \sqrt{\frac{2\log\frac{2}{\beta}}{\rho}} \cdot \sqrt{\log\frac{\frac{4}{\gamma}\log(\frac{2}{\beta})}{\beta}}$, for simplicity, we will use E in Algorithm 3 and the proof in Appendix A. The private version of Count-Min can be derived by adding *independent* Gaussian noises $\mathcal{N}(E, \sigma^2)$ to each counter of C, while the private version of CountSketch can be derived by adding *independent* Gaussian noises $\mathcal{N}(0, \sigma^2)$ to each counter of C. The private versions of Count-Min and CountSketch are derived by combining Algorithm 3, Algorithm 1, and Algorithm 2.

3.1 Main results about Private Count-Min and CountSketch

We present the privacy guarantee and utility analysis of our Private Count-Min and CountSketch below. Recall that for each item x, we perform update as in Algorithm 1 and query as in Algorithm 2. In addition, $\widehat{f}(x)$ is the output estimated frequency and $f(x)$ is the actual frequency. To provide a bound for the additional error due to DP, we define $\widetilde{f}(x)$ to be the non-private estimated frequency (the output of the original Count-Min and CountSketch with the same set of hash functions). We begin with the properties of Private Count-Min. Note that all the proofs are deferred to Appendix A.

Theorem 3.1. *Private Count-Min satisfies ρ-zCDP regardless of the number of queries. Furthermore, with probability $1-\beta$, the output $\widehat{f}(x)$ satisfies that*

$$\forall\, x, 0 \leq \widehat{f}(x) - \widetilde{f}(x) \leq 2E.$$

In addition, for each item x, with probability $1-\beta$,

$$0 \leq \widehat{f}(x) - f(x) \leq \gamma \cdot N + 2E.$$

Comparison to Count-Min. Comparing our Theorem 3.1 with the conclusion in [Cormode and Muthukrishnan, 2005], our Private Count-Min preserves the nice property that the output will not underestimate the frequency with high probability. Furthermore, within the most popular regime where the privacy budget ρ is a constant, the additional error bound due to differential privacy is independent of the size of database N, therefore it will become negligible as N goes large.

Justification of our Gaussian noise. Note that with high probability, all the noises we add ($E + \sigma_{i,j}$, $\sigma_{i,j} \sim \mathcal{N}(0, \sigma^2)$) will be non-negative. Therefore, the noise we add and the original error induced by Count-Min will directly sum up and lead to larger error in evaluation. However, we claim that the additional E ensures that with high probability, for all item x, the output will not underestimate the actual frequency. This nice property enables the good performance of our Private Count-Min when used in approximate top k task, as shown in Section 5.

Next, Theorem 3.2 shows the properties of Private CountSketch.

Theorem 3.2. *Private CountSketch satisfies ρ-zCDP regardless of the number of queries. Furthermore, the frequency query from Private CountSketch is unbiased and with probability $1-\beta$,*

$$\forall\, x, |\widehat{f}(x) - \widetilde{f}(x)| \leq E.$$

In addition, for each item x, with probability $1-\beta$,

$$|\widehat{f}(x) - f(x)| \leq \gamma \cdot N + E.$$

Comparison to CountSketch. Comparing our Theorem 3.2 with the conclusion in [Charikar et al., 2002], our Private CountSketch preserves the nice property that the output will be an unbiased estimate of the frequency. This property enables our use of Private CountSketch in quantile estimation below (Section 4). Furthermore, within the most popular regime where the privacy budget ρ is a constant, the additional error bound due to differential privacy is independent of the size of database N, thus it will become negligible as N goes large.

3.2 The Uniform Bound of Additional Error

Theorem 3.1 and Theorem 3.2 show a uniform bound $\sup_x |\widehat{f}(x) - \widetilde{f}(x)| \leq O(E)$ for linear sketches, which upper bounds the additional error imposed on the estimated frequency due to Differential Privacy guarantees. To derive the point-wise bound for $|\widehat{f}(x) - f(x)|$, we combine our result with the point-wise bound for $|\widetilde{f}(x) - f(x)|$ [Cormode and Muthukrishnan, 2005, Charikar et al., 2002] (note it is straightforward to apply other analyses on the point-wise bound [Minton and Price, 2014, Larsen et al., 2021] due to the triangle inequality). Moreover, in Appendix E, we demonstrate that our analysis provides the tight uniform bound when items are drawn from a large universe.

4 Private Quantile Sketches

In this section, we apply our Private CountSketch to state of the art quantile sketches in the turnstile model. Our private Dyadic CountSketch can estimate all the quantiles accurately at the same time while ensuring differential privacy.

4.1 Revisiting DCS

In [Wang et al., 2013], it is shown that DCS can return all γ-approximate quantiles with constant probability using space $O\left(\frac{1}{\gamma}\log^{1.5} U \log^{1.5}(\frac{\log U}{\gamma})\right)$. More specifically, the sketch structure here consists of $\log U$ CountSketches, each CountSketch uses a counter arrays C, which is $d \times w$ counters. The choice of d, w follows $d = \Theta\left(\log(\frac{\log U}{\gamma})\right)$ and $w = O\left(\sqrt{\log U \log(\frac{\log U}{\gamma})}/\gamma\right)$.

4.2 Private DCS

In this work, we aim to estimate the quantiles accurately while preserving privacy. We do this by replacing CountSketch with PrivateCountSketch, which bases on the same structure as CountSketch discussed above. Given the privacy budget ρ, the privacy budget of each Private CountSketch is thus $\rho_0 = \frac{\rho}{\log U}$, due to composition of zCDP (Lemma 2.7). The ℓ_2-sensitivity of each Private CountSketch is

$$\Delta_2 = O(\sqrt{2d}) = O\left(\sqrt{\log\left(\frac{\log U}{\gamma}\right)}\right).$$

To keep the whole algorithm ρ-zCDP, it suffices to keep each CountSketch ρ_0-zCDP (Lemma 2.7). Therefore, Gaussian Mechanism (Definition 2.5) with $\sigma^2 = O\left(\log U \log\left(\frac{\log U}{\gamma}\right)/\rho\right)$ ensures ρ-zCDP (Lemma 2.8). Similar to Lemma A.1, define $E(\gamma, U) = O\left(\sqrt{\frac{\log U \log(\frac{\log U}{\gamma})}{\rho}} \cdot \sqrt{\log \frac{\log U \log(\frac{\log U}{\gamma})}{\gamma}}\right)$, we can prove that with constant probability, all the Gaussian noises we add to all $\log U$ CountSketches are bounded by E (for simplicity, we use E to represent $E(\gamma, U)$).

Conditioned on the high probability event above, we prove that for a fixed quantile, the estimated quantile will be accurate with high probability. As has been proven in Theorem 3.2, the output estimated frequency is unbiased for any item. Therefore, similar to [Wang et al., 2013], for any item x (corresponding to a fixed CountSketch), we have the output $\widehat{f}(x)$ of that CountSketch satisfies

$$\mathbb{P}\left[\left|\widehat{f}(x) - f(x)\right| > \frac{1}{w} \cdot N + E\right] < \exp\left(-O(d)\right) = O\left(\frac{\gamma}{\log U}\right).$$

By a union bound, with probability $1 - \log U \times O\left(\frac{\gamma}{\log U}\right) = 1 - O(\gamma)$, for any item corresponding to this fixed quantile, the error of CountSketch is bounded by $\frac{1}{w} \cdot N + E$. Conditioned upon this event, by Hoeffding's inequality, with probability $1 - O\left(\frac{\gamma}{\log U}\right)$, the sum of $\log U$ such independent errors is bounded by

$$\sqrt{\log U \log\left(\frac{\log U}{\gamma}\right)} \cdot \left(\frac{N}{w} + E\right) = \gamma \cdot N + E', \tag{2}$$

where $E' = O\left(\frac{\log U \log(\frac{\log U}{\gamma})}{\sqrt{\rho}} \cdot \sqrt{\log \frac{\log U \log(\frac{\log U}{\gamma})}{\gamma}}\right)$. To sum up, for a fixed quantile, with probability $1 - O(\gamma)$, the estimating error is bounded by $\gamma \cdot N + E'$.

Finally, apply another union bound on the $\frac{1}{\gamma}$ different quantiles, with constant probability, all the quantiles are estimated accurately (within the error bound (2)). Note that similar to [Wang et al., 2013], the failure probability here is a constant. For any failure probability β, we can further increase d by a factor of $\log \frac{1}{\beta}$ to reduce this failure probability to β.

Take-away of Private DCS. First, our Private DCS has a same space complexity as the original DCS. In addition, according to (2), the additional error bound is proportional to $\frac{\log U \log \frac{1}{\gamma}}{\sqrt{\rho}}$ (ignoring $\log \log$ terms), and independent to the size of database N. In the most popular regime where the privacy budget ρ is a constant, the additional error bound only appears as lower order terms, which will become negligible as N goes large.

5 Evaluation

We have implemented DP linear sketches and DP DCS, and conducted extensive experiments to evaluated the privacy-utility trade-off of our proposed private sketches. The implementations are written in Python with the advantage of fast prototyping and good readability. The code for the following experiments can be found on Github [3].

5.1 Data Sets

The experimental evaluation is conducted using both synthetic and real world data sets. We consider the synthetic Zipf dataset Zipf [2016] with universe size of 2^{16} and the source IP addresses from CAIDA Anonymized Internet Trace 2015 dataset pas with universe size of 2^{32}. For each independent run in the experiments, we use an input database size $N = 10^5$.

5.2 Metrics

In all experiments, we average the various metrics over 5 independent runs to minimize the measurement variance. The metrics used in the experiments are:

Average Relative Error: Let the set Ψ denotes all unique items in the database. Average Relative Error (ARE) is computed based on Ψ in which $\frac{1}{\Psi}\sum_{e\in\Psi}\frac{|f(e)-\widehat{f}(e)|}{f(e)}$.

F1 Score: F1 score is the harmonic mean of the precision and recall $(2 \cdot \frac{precision \cdot recall}{precision+recall})$.

Average Rank Error: For each evenly spaced quantile and its associated item, we average the distance between the true rank and estimated rank.

We use ARE to evaluate sketch performance on frequency estimation and F1 score to evaluate the sketch's performance in identifying the top 10 items. For quantile approximation, we consider the m evenly spaced quantiles and items. For instance, if $m = 1$, we consider the rank error for the median item; if $m = 2$, we consider then average rank error for the 33^{rd} and 67^{th} percentile items. Lower ARE and average rank error, and higher F1 score indicate better approximation.

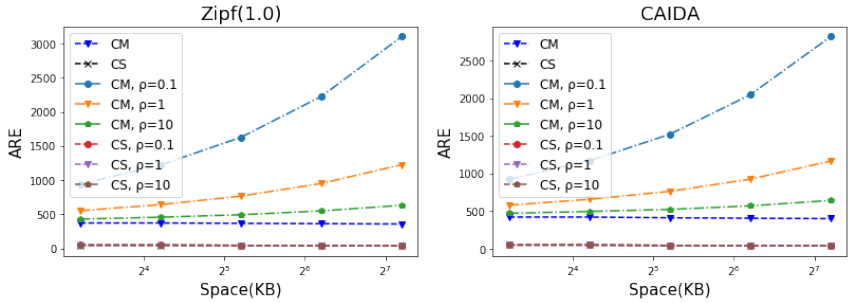
5.3 Private Linear Sketches Experiments

To evaluate the utility of DP linear sketches, we compare the average relative error (ARE) and F1 score for frequency estimation and identify the top 10 items, respectively. As shown in Figure 1, the x-axis represents the space budget for each sketch (from 9.2 KB to 147.3 KB), and the y-axis denotes ARE or F1 score. The DP linear sketches use $\rho \in \{0.1, 1, 10\}$ in which lower ρ value indicates more noise need to be added, and all sketches assume $\beta = 1\%$.

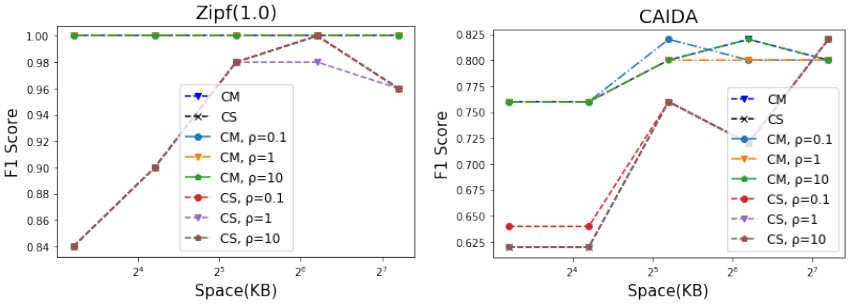
For frequency estimation, the performance of our private CountSketch with various privacy budgets is basically equivalent to the performance of the non-private CountSketch. Under different space and privacy budgets, they have minimal difference in ARE for both Zipf and CAIDA datasets, meaning that, while providing strong privacy guarantee, the estimated frequencies are still very accurate. The accurate estimation of private CountSketch is primarily due to the unbiased nature of CountSketch in which, by adding Gaussian noise, the private CountSketch still provides unbiased estimation for an item's frequency as proved in Theorem 3.2. As shown in both Figure 1(a) and Figure 1(b), the performance of private Count-Min degrades when the space budget increases or the privacy budget decreases. This behavior is expected as the upper bound on the frequency error in Theorem 3.1 has a dependency on both γ and ρ. In order to preserve the property of not underestimating an item's

[3] https://github.com/ZhaoFuheng/Differentially-Private-Linear-Sketches

frequency, the private Count-Min sketch needs to add larger noise to each counter when the number of counters increases. As a result, the estimated frequencies for low-frequency items become inflated and this in turn decreases the overall accuracy.



(a) Frequency Estimation with Zipf (b) Frequency Estimation with CAIDA

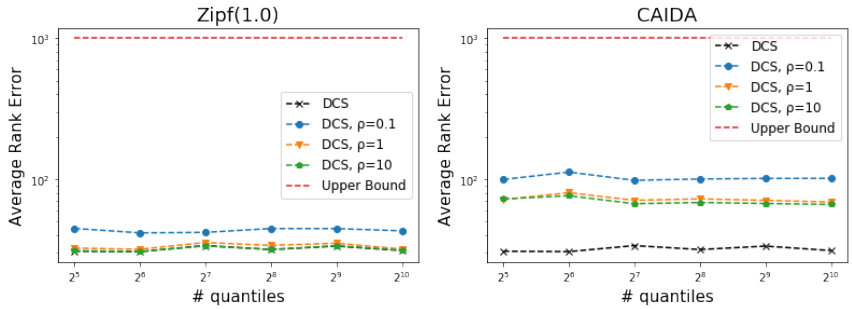


(c) Top 10 with Zipf (d) Top 10 with CAIDA dataset

Figure 1: Comparison of non-private linear sketches and DP linear sketches with various privacy budget under synthetic and real world datasets. The experiments assume $\beta = 1\%$ and $N = 10^5$.

For approximate top 10 items, private CountSketch has similar performance to CountSketch. Since both non-private and private CountSketch are unbiased, they may underestimate the frequency of true top K items and decrease the recall. On the other hand, the property of no underestimation is desirable for approximate top K items. In particular, non-private and private Count-Min sketch score high F1 scores for all datasets. While providing privacy guarantees, private Count-Min achieve 1.0 F1 scores for all space and all privacy budgets in Zipf dataset, as shown in Figure 1(c).

5.4 Private Quantile Sketch Experiments



(a) Quantile Approximation with Zipf (b) Quantile Approximation with CAIDA

Figure 2: Compare DCS and DP DCS with various privacy budget under synthetic and real world datasets. The experiments assume $\gamma = 1\%$, $N = 10^5$, and the desired error upper bound is 10^3 (γN).

To evaluate the utility of DP DCS, we compare the average rank error. As shown in Figure 2, the x-axis represents the number of evenly spaced quantiles, and the y-axis denotes the average rank error. The DP DCS use privacy budget $\rho \in \{0.1, 1, 10\}$ and all sketches assume $\gamma = 1\%$.

For the quantile approximation, we observe that the increase in the number of evenly spaced quantiles does not impact the average rank error, as shown in both Figure 2(a) and Figure 2(b). Since the CAIDA dataset universe size (2^{32}) is larger than Zipf dataset universe size (2^{16}), the average rank error in the CAIDA dataset is larger than the average rank error in the Zipf dataset. As shown in Equation (2), the error bound has a term depending on the universe size in which a large universe size leads to more error. When the privacy budget decreases, the average rank error increases as more noise needs to be added. Comparing DP DCS with strong privacy ($\rho = 0.1$) to DCS, the increase in rank error is relatively small compared to the database size of 10^5. In addition, the desired rank error upper bound is $\gamma \cdot N = 10^3$ and all the rank errors are one order of magnitude lower.

6 Related Works

In this section, we discuss and compare our results to previous literature on Private Count-Min Sketch [Mir et al., 2011, Melis et al., 2015, Ghazi et al., 2019], and the concurrent work on Private CountSketch [Pagh and Thorup, 2022]. In fact, Pagh and Thorup [2022] and us both independently discovered the same algorithm for Private CountSketch with differences in the theoretical analysis. To the best of our knowledge, we are the first to present a DP quantile sketch in the turnstile model.

Private Count-Min. Mir et al. [2011] proposed to add Laplace noise into the Count-Min Sketch estimator to derive the number of heavy hitters with Pan-Privacy [Dwork et al., 2010]. Similarly, Melis et al. [2015] add independent Laplace noise to each counter of the sketch instead of the estimator. However, adding Laplace noise breaks the nice property of never underestimation in Count-Min. In contrast, our private Count-Min guarantees no underestimation with high probability. [Ghazi et al., 2019] added one-sided binomial noise into each counter of the sketch to preserve the property of no underestimation. However, using the Binomial mechanism inherently implies approximate differential privacy [Canonne et al., 2020]. In contrast, by using the Gaussian mechanism, our Private Count-Min provides the stronger concentrated differential privacy guarantee.

Private CountSketch. Pagh and Thorup [2022] and us both independently discovered the same algorithm for private CountSketch. There is a major difference in the analysis and we believe both analyses are valuable, in which Pagh and Thorup [2022] focused on deriving a tight point-wise bound for $|\widehat{f}(x) - f(x)|$, while we focused on deriving a uniform bound for $\sup_x |\widehat{f}(x) - \widetilde{f}(x)|$. Our uniform bound for $\sup_x |\widehat{f}(x) - \widetilde{f}(x)|$ can derive the point-wise bound for $|\widehat{f}(x) - f(x)|$, by combining our result with any point-wise bound for $|\widetilde{f}(x) - f(x)|$ due to the triangle inequality. [Pagh and Thorup, 2022] obtains a tighter point-wise bound by using concentration of median instead of triangle inequality. However, Pagh and Thorup [2022]'s analysis can not imply the point-wise bound for $|\widehat{f}(x) - \widetilde{f}(x)|$. More detailed comparisons are included in Appendix C.

7 Conclusion

In this work, we demonstrate that linear sketches can be made differentially private and provide useful information while maintaining their original properties by adding a small amount of Gaussian noise at initialization. In addition, leveraging the private CountSketch, we propose the DP DCS for quantile approximation in the turnstile model. DP DCS achieves low rank errors even for a large data universe. Moreover, for all the proposed algorithms, when the privacy budget is constant, the additional error due to privacy is independent of the database size and the error will become negligible when the database grows larger. Moreover, private linear sketches bring new opportunities for other statistical questions such as the private euclidean distance estimation [Stausholm, 2021] which can be calculate as the dot product of two private linear sketches. As a result, we believe our proposed algorithms are efficient and practical for real-world systems and enable these systems to perform data analysis and machine learning tasks privately.

Acknowledgments and Disclosure of Funding

This work is partially supported by gifts from Snowflake Inc, and NSF grants CNS-1703560, CNS-1815733 and CNS-2048091. The authors thank Rasmus Pagh for a helpful discussion regarding their concurrent work [Pagh and Thorup, 2022]. The authors also thank Adam Smith for clarifying the mergeability in the inherently private Flajolet-Martin Sketch [Smith et al., 2020].

References

Anonymized internet traces 2015. https://catalog.caida.org/details/dataset/passive_2015_pcap. Accessed: 2022-5-10.

Daniel Alabi, Omri Ben-Eliezer, and Anamay Chaturvedi. Bounded space differentially private quantiles. *arXiv preprint arXiv:2201.03380*, 2022.

Raman Arora, Jalaj Upadhyay, et al. Differentially private robust low-rank approximation. *Advances in neural information processing systems*, 31, 2018.

Peter Bailis, Edward Gan, Samuel Madden, Deepak Narayanan, Kexin Rong, and Sahaana Suri. Macrobase: Prioritizing attention in fast data. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 541–556, 2017.

Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403. PMLR, 2018.

Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. The johnson-lindenstrauss transform itself preserves differential privacy. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 410–419. IEEE, 2012.

Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

Clément L Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. *Advances in Neural Information Processing Systems*, 33:15676–15688, 2020.

Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming*, pages 693–703. Springer, 2002.

Seung Geol Choi, Dana Dachman-Soled, Mukul Kulkarni, and Arkady Yerukhimovich. Differentially-private multi-party sketching for large-scale statistics. *Cryptology ePrint Archive*, 2020.

Graham Cormode. Current trends in data summaries. *ACM SIGMOD Record*, 50(4):6–15, 2022.

Graham Cormode and Marios Hadjieleftheriou. Finding frequent items in data streams. *Proceedings of the VLDB Endowment*, 1(2):1530–1541, 2008.

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

Graham Cormode and Ke Yi. *Small summaries for big data*. Cambridge University Press, 2020.

Graham Cormode, Tejas Kulkarni, and Divesh Srivastava. Answering range queries under local differential privacy. *Proceedings of the VLDB Endowment*, 12(10):1126–1138, 2019.

Sudipto Das, Shyam Antony, Divyakant Agrawal, and Amr El Abbadi. Thread cooperation in multicore architectures for frequency counting over multiple data streams. *Proceedings of the VLDB Endowment*, 2(1):217–228, 2009.

Charlie Dickens, Justin Thaler, and Daniel Ting. (nearly) all cardinality estimators are differentially private. *arXiv preprint arXiv:2203.15400*, 2022.

Jinshuo Dong, Aaron Roth, and Weijie J Su. Gaussian differential privacy. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2019.

Cynthia Dwork and Guy N Rothblum. Concentrated differential privacy. *arXiv preprint arXiv:1603.01887*, 2016.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

Cynthia Dwork, Moni Naor, Toniann Pitassi, Guy N Rothblum, and Sergey Yekhanin. Pan-private streaming algorithms. In *ics*, pages 66–80, 2010.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

Badih Ghazi, Noah Golowich, Ravi Kumar, Rasmus Pagh, and Ameya Velingker. On the power of multiple anonymous messages. *arXiv preprint arXiv:1908.11358*, 2019.

Anna C Gilbert, Yannis Kotidis, S Muthukrishnan, and Martin J Strauss. How to summarize the universe: Dynamic maintenance of quantiles. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 454–465. Elsevier, 2002.

Jennifer Gillenwater, Matthew Joseph, and Alex Kulesza. Differentially private quantiles. In *International Conference on Machine Learning*, pages 3713–3722. PMLR, 2021.

Michael Greenwald and Sanjeev Khanna. Space-efficient online computation of quantile summaries. *ACM SIGMOD Record*, 30(2):58–66, 2001.

Arpit Gupta, Rüdiger Birkner, Marco Canini, Nick Feamster, Chris Mac-Stoker, and Walter Willinger. Network monitoring as a streaming analytics problem. In *Proceedings of the 15th ACM workshop on hot topics in networks*, pages 106–112, 2016.

Nikita Ivkin, Zhuolong Yu, Vladimir Braverman, and Xin Jin. Qpipe: Quantiles sketch fully in the data plane. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, pages 285–291, 2019.

Rajesh Jayaram and David P Woodruff. Data streams with bounded deletions. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 341–354, 2018.

Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In *International Conference on Machine Learning*, pages 5213–5225. PMLR, 2021.

Zohar Karnin, Kevin Lang, and Edo Liberty. Optimal quantile approximation in streams. In *2016 ieee 57th annual symposium on foundations of computer science (focs)*, pages 71–78. IEEE, 2016.

Kasper Green Larsen, Rasmus Pagh, and Jakub Tětek. Countsketches, feature hashing and the median of three. In *International Conference on Machine Learning*, pages 6011–6020. PMLR, 2021.

Tian Li, Zaoxing Liu, Vyas Sekar, and Virginia Smith. Privacy for free: Communication-efficient learning with differential privacy using sketches. *arXiv preprint arXiv:1911.00972*, 2019.

Luca Melis, George Danezis, and Emiliano De Cristofaro. Efficient private statistics with succinct sketches. *arXiv preprint arXiv:1508.06110*, 2015.

Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. Efficient computation of frequent and top-k elements in data streams. In *International conference on database theory*, pages 398–412. Springer, 2005.

Gregory T Minton and Eric Price. Improved concentration bounds for count-sketch. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 669–686. SIAM, 2014.

Darakhshan Mir, Shan Muthukrishnan, Aleksandar Nikolov, and Rebecca N Wright. Pan-private algorithms via statistics on sketches. In *Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 37–48, 2011.

Jayadev Misra and David Gries. Finding repeated elements. *Science of computer programming*, 2(2): 143–152, 1982.

J Ian Munro and Mike S Paterson. Selection and sorting with limited storage. *Theoretical computer science*, 12(3):315–323, 1980.

Rasmus Pagh and Nina Mesing Stausholm. Efficient differentially private f0 linear sketching. In *24rd International Conference on Database Theory*, 2021.

Rasmus Pagh and Mikkel Thorup. Improved utility analysis of private countsketch. 2022.

Dan Qiao and Yu-Xiang Wang. Near-optimal differentially private reinforcement learning. *arXiv preprint arXiv:2212.04680*, 2022a.

Dan Qiao and Yu-Xiang Wang. Offline reinforcement learning with differential privacy. *arXiv preprint arXiv:2206.00810*, 2022b.

Dan Qiao, Ming Yin, Ming Min, and Yu-Xiang Wang. Sample-efficient reinforcement learning with loglog(T) switching cost. In *Proceedings of the 39th International Conference on Machine Learning*, pages 18031–18061. PMLR, 2022.

Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pages 8253–8265. PMLR, 2020.

Nisheeth Shrivastava, Chiranjeeb Buragohain, Divyakant Agrawal, and Subhash Suri. Medians and beyond: new aggregation techniques for sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 239–249, 2004.

Adam Smith, Shuang Song, and Abhradeep Guha Thakurta. The flajolet-martin sketch itself preserves differential privacy: Private counting with minimal space. *Advances in Neural Information Processing Systems*, 33:19561–19572, 2020.

Nina Mesing Stausholm. Improved differentially private euclidean distance approximation. In *Proceedings of the 40th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 42–56, 2021.

Christos Tzamos, Emmanouil-Vasileios Vlatakis-Gkaragkounis, and Ilias Zadik. Optimal private median estimation under minimal distributional assumptions. *Advances in Neural Information Processing Systems*, 33:3301–3311, 2020.

Jalaj Upadhyay. Differentially private linear algebra in the streaming model. *arXiv preprint arXiv:1409.5414*, 2014.

Salil Vadhan. The complexity of differential privacy. In *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer, 2017.

Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

Lu Wang, Ge Luo, Ke Yi, and Graham Cormode. Quantiles over data streams: an experimental study. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 737–748, 2013.

Victor Zakhary, Lawrence Lim, Divyakant Agrawal, and Amr El Abbadi. Cot: Decentralized elastic caches for cloud environments. *arXiv preprint arXiv:2006.08067*, 2020.

Fuheng Zhao, Sujaya Maiyya, Ryan Wiener, Divyakant Agrawal, and Amr El Abbadi. Kll± approximate quantile sketches over dynamic datasets. *Proc. VLDB Endow.*, 14(7):1215–1227, mar 2021. ISSN 2150-8097. doi: 10.14778/3450980.3450990. URL https://doi.org/10.14778/3450980.3450990.

Fuheng Zhao, Divyakant Agrawal, Amr El Abbadi, and Ahmed Metwally. Spacesaving±: An optimal algorithm for frequency estimation and frequent items in the bounded-deletion model. *Proc. VLDB Endow.*, 15(6):1215–1227, feb 2022. ISSN 2150-8097. doi: 10.14778/3514061.3514068. URL https://doi.org/10.14778/3514061.3514068.

George Kingsley Zipf. *Human behavior and the principle of least effort: An introduction to human ecology*. Ravenio Books, 2016.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Claims in abstract and introduction reflect our contributions.
   (b) Did you describe the limitations of your work? [Yes] In Theorem 3.1, the private Count-Min frequency estimation's additional error has a dependency of $\sqrt{\log \frac{1}{\gamma}}$ in order to keep the property of not underestimating item's frequency. Assume database size is fixed, decreasing gamma will increase the average error. Note the error is independent from the database size, and when database size grows, the error will become negligible.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] Apply our proposed algorithms to current systems will protect user privacy.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] We put some proves in Appendix A due to space limits
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We will provide an url to our Github Repo once the paper is accepted.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We provide the parameters used for the experiments.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] We didn't use random seed
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A] we didn't use any external resources beside a macbook pro.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A] We did not use existing assets.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We did not use crowdsourcing or conducted research with human subjects.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Missing proofs

In this section, we present the missing proofs. Recall that for each item x, we perform query as in Algorithm 2. In the proof below, we use arr' to denote the arr in Algorithm 2 (with private initialization) while arr denotes the arr in Algorithm 2 under non-private initialization (all zero initialization) and the same set of hash functions. In addition, $\widehat{f}(x)$ is the output estimated frequency, $f(x)$ is the actual frequency and $\widetilde{f}(x)$ is the non-private estimated frequency (the output of the original Count-Min and CountSketch with the same set of hash functions). We first present the following Lemma A.1, which gives a high probability bound for the Gaussian noises we add.

Lemma A.1 (Utility analysis)**.** *If there are $\frac{1}{\gamma}\log(\frac{2}{\beta})$ independent Gaussian noises sampled from $\mathcal{N}(0,\sigma^2)$ (where $\sigma = \sqrt{\frac{\log(2/\beta)}{\rho}}$), denoted as σ_{ij} where $i \in [\log(2/\beta)]$, $j \in [1/\gamma]$, then with probability $1-\frac{\beta}{2}$, for any $i \in [\log(2/\beta)]$, $j \in [1/\gamma]$,*

$$|\sigma_{ij}| \leq \sqrt{2}\sigma \cdot \sqrt{\log \frac{\frac{4}{\gamma}\log(\frac{2}{\beta})}{\beta}} = \sqrt{\frac{2\log\frac{2}{\beta}}{\rho}} \cdot \sqrt{\log \frac{\frac{4}{\gamma}\log(\frac{2}{\beta})}{\beta}}. \tag{3}$$

Proof of Lemma A.1. The lemma directly results from the concentration inequality of Gaussian distribution and a union bound. □

Theorem A.2 (Restate Theorem 3.1)**.** *Private Count-Min satisfies ρ-zCDP regardless of the number of queries. Furthermore, with probability $1-\beta$, the output $\widehat{f}(x)$ satisfies that*

$$\forall\, x, 0 \leq \widehat{f}(x) - \widetilde{f}(x) \leq 2E.$$

In addition, for each item x, with probability $1-\beta$,

$$0 \leq \widehat{f}(x) - f(x) \leq \gamma \cdot N + 2E.$$

Proof of Theorem A.2. Differential privacy directly results from the DP guarantee of Gaussian Mechanism (Lemma 2.8) and post processing (Lemma 2.7).

According to Lemma A.1, we have with probability $1-\frac{\beta}{2}$, for all noises $E+\sigma_{ij}$, where $\sigma_{ij} \sim \mathcal{N}(0,\sigma^2)$, $i \in [\log(2/\beta)]$, $j \in [1/\gamma]$, it holds that

$$0 \leq E + \sigma_{ij} \leq 2E.$$

Under the above case that will happen with probability $1-\frac{\beta}{2}$, for all x and the corresponding arr, arr', it holds that

$$\min(\mathrm{arr}) \leq \min(\mathrm{arr}') \leq \min(\mathrm{arr} + 2E \cdot \mathbb{1}) \leq \min(\mathrm{arr}) + 2E.$$

Therefore, for all x,

$$\widetilde{f}(x) \leq \widehat{f}(x) \leq \widetilde{f}(x) + 2E.$$

For the last conclusion (point-wise bound), by the property of Count-Min [Cormode and Muthukrishnan, 2005], we have for any item x, with probability $1-\frac{\beta}{2}$,

$$0 \leq \min(\mathrm{arr}) - f(x) \leq \gamma \cdot N.$$

Therefore, for any item x, with probability $1-\beta$, we have

$$\widehat{f}(x) = \min(\mathrm{arr}') \geq \min(\mathrm{arr}) \geq f(x)$$

and

$$\widehat{f}(x) = \min(\mathrm{arr}') \leq \min(\mathrm{arr}) + 2E \leq f(x) + \gamma \cdot N + 2E.$$

Then the proof is completed by plugging in the definition of E. □

Theorem A.3 (Restate Theorem 3.2)**.** *Private CountSketch satisfies ρ-zCDP regardless of the number of queries. Furthermore, the frequency query from Private CountSketch is unbiased and with probability $1-\beta$,*

$$\forall x, |\widehat{f}(x) - \widetilde{f}(x)| \leq E.$$

In addition, for each item x, with probability $1-\beta$,

$$|\widehat{f}(x) - f(x)| \leq \gamma \cdot N + E.$$

Proof of Theorem A.3. First of all, differential privacy directly results from the DP guarantee of Gaussian Mechanism (Lemma 2.8) and post processing (Lemma 2.7).

Next we claim that the conclusion that $\mathbb{E}\widehat{f}(x) = f(x)$ arises from symmetry. If we replace $\{h_i\}_{i\in[\log(2/\beta)]}$, $\{g_i\}_{i\in[\log(2/\beta)]}$, $\{\sigma_{ij}\}_{i\in[\log(2/\beta)],j\in[1/\gamma]}$ with $\{h_i\}_{i\in[\log(2/\beta)]}$, $\{g'_i\}_{i\in[\log(2/\beta)]}$, $\{-\sigma_{ij}\}_{i\in[\log(2/\beta)],j\in[1/\gamma]}$ where $g'_i(x) = g_i(x)$ and $g'_i(x') = -g_i(x')$, $\forall x' \neq x$, then the outputs under these two cases will be symmetric around $f(x)$ and the probability distribution function at these two cases are identical. Therefore, we have

$$\mathbb{E}\widehat{f}(x) = f(x). \tag{4}$$

According to Lemma A.1, we have with probability $1-\frac{\beta}{2}$, for all noises $\sigma_{ij} \sim \mathcal{N}(0,\sigma^2)$, $i \in [\log(2/\beta)]$, $j \in [1/\gamma]$, it holds that

$$|\sigma_{ij}| \leq E.$$

Under the above case that will happen with probability $1-\frac{\beta}{2}$, we will prove that for all x,

$$|\widehat{f}(x) - \widetilde{f}(x)| \leq E.$$

Without loss of generality, we can assume that $\log\frac{2}{\beta} = 2k+1$, where k is a positive integer (we can choose k to be the minimum integer such that $2k+1 \geq \log\frac{2}{\beta}$).
Suppose that $\mathrm{median}(\mathrm{arr}') > \mathrm{mediam}(\mathrm{arr}) + E$, then it holds that there are at least $k+1$ elements in arr' that is larger than $\mathrm{mediam}(\mathrm{arr}) + E$ due to the definition of median. Because $|\sigma_{ij}| \leq E$, for all i, j, there are at least $k+1$ elements in arr that is larger than $\mathrm{mediam}(\mathrm{arr})$, which leads to contradiction. As a result, we have

$$\widehat{f}(x) - \widetilde{f}(x) = \mathrm{median}(\mathrm{arr}') - \mathrm{mediam}(\mathrm{arr}) \leq E.$$

Similarly, $\widehat{f}(x) - \widetilde{f}(x) \geq -E$. Combining these two results, for all x, it holds that

$$|\widehat{f}(x) - \widetilde{f}(x)| \leq E.$$

Finally, we prove the point-wise bound. By the property of CountSketch [Charikar et al., 2002], we have for any item x, with probability $1-\frac{\beta}{2}$,

$$|\widetilde{f}(x) - f(x)| \leq \gamma \cdot N.$$

Therefore, according to triangle inequality, for each item x, with probability $1-\beta$,

$$|\widehat{f}(x) - f(x)| \leq \gamma \cdot N + E.$$

Then the proof is completed by plugging in the definition of E. □

B Missing Quantile Algorithms

Gilbert et al. [2002] made the connection between frequency and quantiles to propose the first universe based RSS quantile sketch in the turnstile model. Observe, that the relationship between frequency and rank is that one can sum up all items' frequency in the range of 0 to the item itself to estimate the rank. However, this naive approach requires summing all items' frequencies in the range, and the error quickly escalates. A better approach is to break the range from 0 to item x into at most $\log U$ dyadic intervals [Cormode et al., 2019] and then sum all frequencies for each dyadic interval to obtain the estimation of the rank(x). Cormode and Muthukrishnan [2005] proposed the Dyadic Count-Min

sketch (DCM) which uses Count-Min sketches for estimating the frequencies of each dyadic interval with space complexity $O(\frac{1}{\gamma}\log^2 U \log\frac{\log U}{\gamma})$ and update time $O(\log U \log\frac{\log U}{\gamma})$. Later, Wang et al. [2013] leveraged the unbiased property of CountSketch and proposed the Dyadic CountSketch (DCS) which replaces the Count-Min sketch with the CountSketch [Charikar et al., 2002] to further improve the space complexity to $O(\frac{1}{\gamma}\log^{1.5} U \log^{1.5}(\frac{\log U}{\gamma}))$ while maintaining the same update time. DCS and DCM share the same update and query algorithms as shown below. DCM uses $O(\frac{1}{\gamma}\log U \log\frac{\log U}{\gamma})$ space for each Count-Min sketch and DCS uses $O(\frac{1}{\gamma}\log^{0.5} U \log^{1.5}(\frac{\log U}{\gamma}))$ space for each CountSketch, where both of them use $O(\log\frac{\log U}{\gamma})$ rows. For more specific details, [Cormode and Yi, 2020] provide a comprehensive analysis of linear and quantile sketches.

Algorithm 4 DCS/DCM Update(x, v)

Input: Item x with value $v \in \{-1, +1\}$, and an array of linear sketches $\{\mathrm{LS}_0, \ldots, \mathrm{LS}_{\log U}\}$.
for $j \leftarrow 0, \ldots, \log U$ **do**
  $\mathrm{LS}_j.\mathrm{update}(x, v)$
  $x \leftarrow \lfloor x/2 \rfloor$
end for
Output: $\{\mathrm{LS}_0, \ldots, \mathrm{LS}_{\log U}\}$.

Algorithm 5 DCS/DCM Query(x)

Input: Item x, and an array of linear sketches $\{\mathrm{LS}_0, \ldots, \mathrm{LS}_{\log U}\}$.
$R \leftarrow 0$
for $i \leftarrow 0, \ldots, \log U$ **do**
  if x is odd **then**
    $R \leftarrow R + \mathrm{LS}_j.\mathrm{query}(x)$
  end if
  $x \leftarrow \lfloor x/2 \rfloor$
end for
Output: R.

Based on the observations, DCS and DCM quantile sketches keeps $\log U$ number of linear sketches, one for each dyadic interval. As a result, to update an item x with value $v \in \{-1, +1\}$, DCS and DCM need to update $\log U$ levels: they first map item x to a dyadic interval for the level and then update the corresponding linear sketch, as shown in Algorithm 4. To estimate the rank of an item, DCS and DCM first break the range into at most $\log U$ dyadic intervals and then query the frequency for each interval from the corresponding linear sketch, as shown in Algorithm 5.

C Detailed Comparison for Private CountSketch

Recall that for some item x, $\widehat{f}(x)$ is the output estimated frequency, $f(x)$ is the actual frequency and $\widetilde{f}(x)$ is the non-private estimated frequency (the output of the original Count-Min and CountSketch with the same set of hash functions). Let k denote the number of rows in our counter, and we choose $k = \log\frac{2}{\beta}$ to bound the failure probability by β. We first state both Pagh and Thorup [2022]'s results and ours.

Our uniform bound (Theorem 3.2): $\sup_x |\widehat{f}(x) - \widetilde{f}(x)| \leq E = \widetilde{O}(\sqrt{\frac{k}{\rho}})$.

Our lower bound (Theorem E.1): $\sup_x |\widehat{f}(x) - \widetilde{f}(x)| \geq \Omega(\sqrt{\frac{k}{\rho}})$.

Pagh and Thorup [2022]'s point-wise bound[4]: $|\widehat{f}(x) - f(x)| \leq$ non-private error bound $+ \widetilde{O}(\sqrt{\frac{1}{\rho}})$.

There is a major difference between the analysis in [Pagh and Thorup, 2022] and ours. While Pagh and Thorup [2022] focused on the point-wise bound for $|\widehat{f}(x) - f(x)|$, we focused on the uniform bound for $\sup_x |\widehat{f}(x) - \widetilde{f}(x)|$. We are interested in the trade-off between privacy and accuracy for the

[4]We reformulate the bound in [Pagh and Thorup, 2022] for comparison.

CountSketch. In particular, we want to answer the question of what additional error will be imposed on the estimated frequency due to the Differential Privacy guarantee. As a result, our result shows an uniform bound $\sup_x |\widehat{f}(x) - \widetilde{f}(x)| \leq E$ where E is a function of the desired accuracy, failure probability, and privacy guarantee. To derive the point-wise bound for $|\widehat{f}(x) - f(x)|$, we can simply combine our result with any point-wise bound for $|\widetilde{f}(x) - f(x)|$ (like the one in our work or [Pagh and Thorup, 2022]) due to triangular inequality. However, Pagh and Thorup [2022]'s analysis can not imply even point-wise bound for $|\widehat{f}(x) - \widetilde{f}(x)|$.

We agree that Pagh and Thorup [2022] has a tight point-wise bound for $|\widehat{f}(x) - f(x)|$ by using the concentration of median. By comparing our analysis and [Pagh and Thorup, 2022] for $|\widehat{f}(x) - f(x)|$, Pagh and Thorup [2022]'s point-wise bound removes the $\sqrt{\log \frac{2}{\beta}}$ (β denotes failure probability) in the lower order term E which is added to the original estimation error $|\widetilde{f}(x) - f(x)|$. However, the difference may not be substantial. When the database is large (which is the usual case for why we need to perform approximations), E is small compared to the $|\widetilde{f}(x) - f(x)|$. Even for the extreme case of setting $\beta = 1e-10$, the amplification factor for calculating E is $\sqrt{\log \frac{2}{\beta}} \approx 5.8$, which E is still very likely to be small compare to $|\widetilde{f}(x) - f(x)|$.

We believe that none of the two results dominate each other. Both Pagh and Thorup [2022]'s point-wise bound for $|\widehat{f}(x) - f(x)|$ and our uniform bound for $|\widehat{f}(x) - \widetilde{f}(x)|$ are useful analysis for understanding the Differentially Private CountSketch with Gaussian noise.

D Extension to the Data Stream Setting

Our Differentially Private Linear Sketches only guarantee Differential Privacy for the query of database, i.e., the adversary is only allowed to query after the whole database passes our algorithm. However, queries for data stream is also quite practical in real-life applications, i.e., the adversary can query at any time. Take reinforcement learning (RL) as an instance, we can use differentially private linear sketches to estimate the visitation number of all (state,action) pairs while preserving privacy. If we only have linear sketches for database, we can only handle offline RL [Qiao and Wang, 2022b], while with linear sketches for data stream, we can deal with the more challenging online RL [Qiao and Wang, 2022a, Qiao et al., 2022].

Our algorithms can be extended to the data stream setting with moderate modifications. Different from our approach of adding noise at the beginning (Algorithm 3), we need to add noise after each item passes our algorithm. To guarantee Differential Privacy under data stream, we can apply the tree-based algorithm (as shown in Kairouz et al. [2021]) to add Gaussian noises to continuous data. In this way, the algorithm is Differentially Private no matter how many times the adversary queries the data stream and the additional error bound is the same scale as E in our main theorems, with some extra multiplicative logarithmic terms.

E Lower Bound for the Additional Error due to Privacy

In this section, we provide a lower bound for CountSketch (the counterpart for Count-Min is similar and we omit it here) showing that our analysis of $\sup_x |\widehat{f}(x) - \widetilde{f}(x)|$ is tight. For simplicity, we assume the counter arrays C we use has shape $k \times d$ and k is odd, which means our upper bound E for $\sup_x |\widehat{f}(x) - \widetilde{f}(x)|$ is $\widetilde{O}(\sqrt{\frac{k}{\rho}})$. Then according to our Algorithm 3, the noise we add has scale (standard variance) $\sqrt{\frac{k}{\rho}}$. The following theorem shows that if the universe is large enough, with constant probability, $\sup_x |\widehat{f}(x) - \widetilde{f}(x)| \geq \Omega(\sqrt{\frac{k}{\rho}})$, which matches our upper bound in Theorem 3.2 up to logarithmic terms.

Theorem E.1. *There exists constants c, p, such that if the size of universe satisfies $U \geq ckd(1 + \frac{1}{d-1})^{k-1}$, for our Private CountSketch, there exists a database, with probability at least p,*

$$\sup_x |\widehat{f}(x) - \widetilde{f}(x)| \geq \sqrt{\frac{k}{\rho}}.$$

Proof of Theorem E.1. Fix some item x with its corresponding $\{h_i(x), g_i(x)\}_{i\in[k]}$. For each $i \in [k]$, we aim to find y_i in universe such that $h_i(y_i) = h_i(x)$, $h_j(y_i) \neq h_j(x)$, $\forall\, j \neq i$ and $g_i(y_i) = g_i(x)$ (this is replaced with $g_i(y_i) = -g_i(x)$ if $i \geq \frac{k+1}{2}$). For any item in the universe, due to the uniform randomness of its hash functions, the probability it satisfies such conditions is $Pr = \frac{1}{2d}(1 - \frac{1}{d})^{k-1}$. Therefore, when $U \geq ckd(1 + \frac{1}{d-1})^{k-1}$ for some constant c, with constant probability, we can find $\{y_i\}_{i\in[k]}$ satisfying the previous conditions.

Next we can construct a database with only $\{x, y_1, \cdots, y_{k-1}\}$, where the frequency of x is some large n_x, and the frequencies of all y_i's are n_y that satisfies that n_y is much larger than E. Therefore, the arr of x (without adding noise) consists of one n_x, $\frac{k-1}{2}$ numbers much larger than $n_x + E$ and $\frac{k-1}{2}$ numbers much smaller than $n_x - E$. Finally, with high probability, the change from $\widetilde{f}(x)$ to $\widehat{f}(x)$ is from n_x to $n_x + \mathcal{N}(0, \frac{k}{\rho})$, which finishes our proof. □

Hybrid Querying Over Relational Databases and Large Language Models

Fuheng Zhao
UC Santa Barbara
fuheng_zhao@ucsb.edu

Divyakant Agrawal
UC Santa Barbara
agrawal@cs.ucsb.edu

Amr El Abbadi
UC Santa Barbara
amr@cs.ucsb.edu

ABSTRACT

Database queries traditionally operate under the closed-world assumption, providing no answers to questions that require information beyond the data stored in the database. Hybrid querying using SQL offers an alternative by integrating relational databases with large language models (LLMs) to answer beyond-database questions. In this paper, we present the first cross-domain benchmark, SWAN, containing 120 beyond-database questions over four real-world databases. To leverage state-of-the-art language models in addressing these complex questions in SWAN, we present two solutions: one based on schema expansion and the other based on user defined functions. We also discuss optimization opportunities and potential future directions. Our evaluation demonstrates that using GPT-4 Turbo with few-shot prompts, one can achieves up to 40.0% in execution accuracy and 48.2% in data factuality. These results highlights both the potential and challenges for hybrid querying. We believe that our work will inspire further research in creating more efficient and accurate data systems that seamlessly integrate relational databases and large language models to address beyond-database questions.

CCS CONCEPTS

• **Information systems** → **Query languages**; **Information retrieval**; **Information integration**.

KEYWORDS

Hybrid Query, Relational Databases and Large Language Models

1 INTRODUCTION

The Relational model and SQL have achieved widespread acceptance and usage in data management systems. In particular, SQL continues to evolve by adding new syntax and features, enhancing its capabilities to meet the growing demands of modern data systems [32]. It is well known that database queries are evaluated under a closed domain assumption, meaning that the queries address aspects of the real world based solely on the data stored in the relational database management system [30]. While the direct approach is to say NO to beyond-database questions, researchers in our community have also investigated alternatives such as providing answers based on incomplete data with heuristic algorithms [14] and crowdsourcing [8]. In this paper, we explore the use of large language models to address these questions.

 15th Annual Conference on Innovative Data Systems Research (CIDR '25). January 19-22, Amsterdam, The Netherlands

In the natural language processing community, open-domain question answering has been extensively studied, often encompassing a broader range of inquiries. This long-standing task aims to provide factual answers to natural language questions by drawing from large, unstructured collections of texts and documents, such as Wikipedia. By pre-training on large corpus of knowledge, large language models (LLMs) have demonstrated significant potential in providing world knowledge and performing complex reasoning [12, 38]. In this paper, we are specifically interested in beyond-database questions which have partial information grounded in the relational database. Unlike open-domain questions, beyond-database questions require integrating and reasoning with structured data from relational databases as well as drawing from large, unstructured collections of texts.

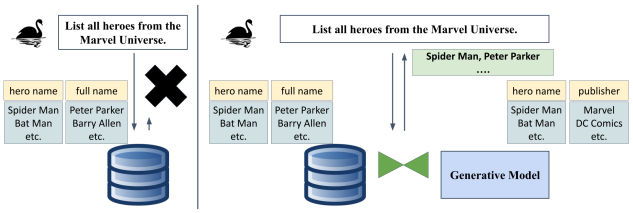


Figure 1: An illustrative example contrasting the answering of a beyond-database question solely using a database (left) versus hybrid querying over both databases and large language models (right).

A motivating example: Consider a simple database with a single table containing superhero information, as shown in Figure 1. The schema for the database is: *superhero(hero_name, full_name)*. Suppose the user wants to list all the hero names from the Marvel Universe within the database. This would be considered as a beyond-database question, since the database contains relevant or partial information to the request but cannot directly provide the answer (i.e., which heroes are Marvel universe). On the other hand, large language models such as ChatGPT can be used to identify the publisher for each superhero characters. Assume we treat LLMs as a table containing the hero_name and the publisher. Then, a SQL query like: "SELECT hero_name, full_name FROM LLM JOIN superhero ON LLM.hero_name = superhero.hero_name WHERE llm.publisher = 'Marvel';" Hence, hybrid querying by integrating relational databases and large language models offers a powerful approach for addressing beyond-database questions.

In this work, we propose **SWAN**, **S**olving beyond-database queries **W**ith generative **AI** a**N**d relational databases, the first hybrid query benchmark. SWAN comprises 120 beyond-database questions and spans four big databases, covering four diverse domains. The original databases and questions are from the recent Bird benchmark[15],

which is a benchmark for evaluating natural language to SQL translation. The data in these databases are collected from open-source relational databases in platforms such as Kaggle. Also, the proposed SWAN benchmark challenges large language models to both select values from a given list (e.g., choose publisher name from a list of predefined publishers) and generate free-form outputs (e.g., determine the city based on a street address). In addition to proposing the SWAN benchmark, we also introduce HQDL, a preliminary solution for answering beyond-database questions. We evaluate HQDL over the state-of-the-art large language models such as ChatGPT (gpt-3.5-turbo) [22] and the GTP-4 turbo [1, 20]. Our experimental results indicate significant challenges for current state-of-the-art models in generating factual data and accurately answering beyond-database questions with hybrid queries. With in-context learning (ICL), GPT-4 Turbo achieves 40.0% execution accuracy and 48.2% of its generated data is factually correct. Proposing a novel benchmark for beyond-database queries will focus attention and drive more research in this critical area. It will also encourage researchers to develop innovative solutions that are founded on solid well understood foundations. The evaluation results based on HQDL further highlight the need for more advancements in improving the accuracy and reliability of answering beyond-database questions with hybrid queries.

The paper is organized as follows: In Section 2, we discuss the background and related works on hybrid querying. Section 3 introduces the SWAN benchmark and provides details on the construction of the databases and the corresponding beyond-database questions. In Section 4, we present HQDL a preliminary solution for utilizing large language models to solve these complex questions and discuss potential areas for improvement. Section 5 showcases our evaluation of HQDL on the SWAN benchmark. Finally, Section 6 concludes the paper.

2 BACKGROUND AND PROBLEM STATEMENT

In this section, we provide some basic background of large language models (LLMs) and introduce related work from both the database and the natural language communities for answering beyond-database questions.

2.1 Large Language Models Basics

Large language models are trained on vast datasets to produce high-quality responses based on input prompts. They have shown remarkable capabilities in addressing complex tasks across various domains, including logic reasoning [38], natural language to SQL translations [7, 9, 27], and data system tuning [10, 39].

Large language models are frequently used to retrieve domain-specific knowledge and to simplify the information retrieval process by providing direct natural language answers. However, these models may exhibit hallucinations and factuality errors. Consequently, recently many researchers have focused on enhancing the capabilities of LLMs to provide factual information [34]. In-context learning (ICL) is a widely adopted strategy to enhance the factual accuracy of generated content. ICL enables a language model to learn from example demonstrations within its context to improve performance [2].

2.2 Related Work

Answering beyond-database questions has been investigated in a variety of research efforts. CrowdDB [8], Qurk [18], Deco [25], and hQuery [23] introduce crowdsourced query processing systems that address the closed-world assumption in traditional query processing. However, the cost of incorporating human input can be significant, impacting both time and resources. Inspired by the capabilities of LLMs, researchers have investigated whether LLMs can be used for declarative prompting [24] and data cleaning tasks such as data imputation where LLMs repair dirty or missing values in data entries [3, 19]. While this is closely related to hybrid queries, hybrid querying over relational databases and LLMs presents its own set of challenges in combining structured and unstructured data sources, ensuring the correctness of generated data, and materializing the data for future uses. Furthermore, two recent studies [31, 33] laid out the vision for augmenting relational databases with data generated from LLMs. However, due to the absence of an evaluation benchmark, both studies are limited to preliminary case studies. For instance, Galois [31] executed 46 SQL queries (drawn from the Spider benchmark [36]) solely on LLMs, without involving any relational databases. Also, they manually verified the generation results, comparing the output table statistics (e.g., cardinality) and verifying content accuracy with the ground truth. Our proposed SWAN benchmark is built on top of databases collected in the Bird benchmark [15], which has 270x (on average) more rows compared to the databases in the Spider benchmark [36]. Moreover, in addition to content accuracy, we also compare the execution accuracy among hybrid queries (see more explanations in Section 5)

3 SWAN CONSTRUCTION

3.1 Databases in SWAN

We constructed the SWAN benchmark[1] based on the Bird benchmark [15]. In the Bird benchmark, there are eleven diverse database domains. However, we have identified that many of these databases are overly narrow, and the questions are too specific to be answered effectively by a general intelligence model. For instance, the financial database includes detailed tables on bank accounts, credit card information, loans, and trading transactions. This level of specificity are out of the scope of a general AI's capabilities. As a result, we selected four diverse databases: European Football, Formula One, California Schools, and Superhero. These databases cover a broad range of topics, from sports statistics and history to educational trends and fictional characters.

3.2 Schema Curation

The main challenges in evaluating hybrid queries are: i) generating beyond-database questions, and ii) ensuring the availability of ground truth answers for these questions. Fortunately, the Bird benchmark provides valuable assets: natural language questions, SQL queries, and the databases. Leveraging these resources, we can modify the databases to create questions that the databases can not answer based on the database content. For the four selected databases, we removed specific columns or entire tables to generate

[1] see https://github.com/ZhaoFuheng/SWAN/

beyond-database questions. For example, in the Superhero database, the *superhero* table contains a *publisher_id* field, which is a foreign key used to identify the *publisher_name* in the *publisher* table. By dropping the *publisher_id* column, all questions related to finding the name of the publisher become unanswerable based on the newly curated database. While the entire publisher table can be directly dropped, we kept the distinct values of *publisher_name* to assist LLMs in correctly formatting the output related to publishers (see more explanation in the next section). After schema curation, the statistics of the selected databases are shown in Table 1.

Database	Tables	Rows/Table	Cols Dropped
European Football	7	31828	12
Formula One	13	39561	12
California Schools	3	9980	12
Superhero	10	1061	11

Table 1: Statistics of databases in SWAN.

3.3 Free Form Response and Value Selection

In SWAN, the challenges for LLMs to generate factual data can be broken down into two categories: i) free form response and ii) value selection. The free form response requires LLMs to generate data when some context is provided. For instance, in the California Schools database, the tables originally contained both the school name and the school url. We removed the school url column, and as a result, we expect LLMs to generate short-form urls for the schools. Often, the school url is closely related to the school name and often ends with edu. Value selection involves choosing data values from a predefined list (e.g., a list of unique publisher names). For instance, after we removed the *publisher_id* field from the *superhero* table and the entire *publisher* table from the Superhero database, we retained the unique values of *publisher_names*, which contains the names of all publishers for the superheroes in the database. Consequently, the list of all publisher names can be provided to the LLMs, allowing them to select the appropriate publisher for each superhero.

3.4 Keys for Tables from LLMs

In relational databases, a foreign key column is often represented as an integer linked to the primary key column in another table. However, integers do not provide any meaningful insights for LLMs to generate useful data values. According to SQL standards, a foreign key must reference a unique key in the foreign table. Therefore, we have curated the databases to include meaningful foreign keys for the data generated by LLMs. For example, in the Superhero database, we assume the combination of *superhero_name* and $full\_name$ of a superhero serves as the key to finding the publisher information. Also, we have ensured that there are no duplicate pairs of (*superhero_name*, $full\_name$) in the table. Our approach of designing meaningful keys for LLMs to generate data aligns with the data model in crowd-sourcing systems. For example, Deco's Fetch/Resolution rules [25] use meaningful keys as input and ask crowd-source workers to generate a group of attributes based on the given keys.

3.5 Beyond-Database Questions

For each database in SWAN, we provide 30 beyond-database questions, resulting in a total of 120 questions across all databases. For each question, we also supply i) a hybrid SQL query that joins the tables in the relational database with the tables generated by LLMs (assume the values generated by LLMs are materialized as tables), ii) a hybrid SQL query that directly invokes LLM calls based on BlendSQL [11] functions, and iii) the corresponding gold SQL query from Bird, such that the expected answer is the execution results of the gold SQL query on the original Bird databases. Notably, these hybrid queries are manually crafted and fully executable. They are provided to assess the current capabilities of combining LLMs with databases to answer beyond-database questions. Automating the translation of beyond-database questions into hybrid queries is left as future work.

4 ANSWERING BEYOND-DATABASE QUESTIONS

In this section, we discuss two different approaches to answer beyond-database questions in SWAN. One is based on schema expansion and the other is based on SQL user defined functions. At the end of this section, we discuss the promising optimization opportunities of these solutions and outline potential directions for future research.

4.1 HQDL

First, we introduce Hybrid Query Database and LLM (HQDL), a preliminary solution for solving beyond-database questions based on schema expansion. Given a beyond-database NL question, one can expand the schema of the database by including new columns or new tables such that NL question becomes answerable based on the new schema. Then, LLMs can be used to fill in all the missing data entries after schema expansion. Based on the newly updated schema, one can write a regular SQL query to answer question NL directly.

4.1.1 Data Generation. For each database, SWAN provides a list of missing columns and tables that need to be generated by LLMs to answer the provided beyond-database questions. SWAN has also provided the keys consisting of the minimal number of attributes that represent the primary-key/foreign-key (PK-FK) relationships between the existing tables in the relational database and the tables generated by LLMs. This ensures that the necessary keys can be used as input for the LLMs, enabling them to accurately generate the missing data entries. We would also like to note that this information (e.g., missing columns, keys) can be helpful, though its use is optional. In HQDL, we choose to leverage this metadata directly. Looking ahead, we envision future work to be fully automated, with capabilities such as directly discovering join keys and automatically creating new columns or tables.

The following is an example of a zero-shot prompt that has been utilized to address and generate missing data values within the Super Hero database. This structured prompt instructs LLMs to infer and fill in missing data entries by supplying guidance, column names, and example values for certain columns (e.g., publishers and colors).

```
Your task is to fill in the missing values
    in the target entry from the `superhero`
     database.
Return a single row with no explanation.

The columns are: `superhero_name`,`full_name
    `,`eye_color`,`hair_color`,`skin_color
    `,`publisher_name`,`race`,`gender`,`
    moral_alignment`,`powers`

The possible values for `publisher_name` are
    [Dark Horse Comics', 'DC Comics', Marvel
     Comics', ...]

Value list of colors, power names, etc.

Target Entry:'{superhero_name}','{full_name}
    ',?,?,?,?,?,?,?,?
The output should consist of a single row
    containing 10 fields.
Answer:
```

HQDL needs to instruct the LLMs to fill in the missing values in the target data entry. HQDL also adopts the widely accepted 'No Explanation' rule introduced by OpenAI [21], which consistently improves the quality of generated answers for semantic parsing [9]. Furthermore, HQDL provides value lists, such as publishers and colors, for LLMs to select from. Since only the id fields are removed (e.g., *publisher_id*), HQDL can directly retrieve all these predefined data values. The goal is to avoid ambiguous data values such as 'Marvel' v.s. 'Marvel Comics' in which both values represent the same publisher but pose challenges for automatic evaluation.

In addition to zero-shot prompts, we also conduct investigations on few-shot prompts. A one-shot prompt for generating the missing data values for the Super Hero database is provided below:

```
Prefix (instructions and value lists)

/*An example is provided before the target
    data entry*/
Example Entry:'3-D Man','Charles Chandler'
    ,?,?,?,?,?,?,?,?
Example Answer:'3-D Man','Charles Chandler',
    'Brown','Grey','No Colour','Marvel
    Comics','-','Male','Good','Agility,Super
     Strength,Stamina,Super Speed'

Target Entry:'{superhero_name}','{full_name}
    ',?,?,?,?,?,?,?,?

Answer:
```

As shown above, an example data entry and the corresponding answer are provided to the LLM for the constructed record corresponding to '3-D Man' and Charles Chandler. In the evaluation section (Section 5), we will show that few-shot demonstrations significantly improve the quality of the generated data entries."

Data Extraction. After collecting all data entries generated by the LLMs, HQDL materializes these entries into tables. HQDL uses the Python csv module's reader to process these entries, converting them into a structured format, and inserting them into new tables in the underlying SQLite database. Moreover, in SWAN, there are both one-to-one and one-to-many relationships. When one-to-many relationships occur, HQDL condenses the tuples in the "many" side of the relationship into a long text. For example, each superhero may be associated with many powers. HQDL would condense all the powers into a long string separated by commas (e.g., "Agility, Super Strength, Super Speed").

4.2 Hybrid Query UDFs

We observe that industry has started integrating LLM calls directly into SQL syntax through user defined functions, such as DucksDB [28] and Google BigQuery [13]. For instance, finding all hero names from the Marvel universe within the database can be rewritten as follows in Google BigQuery:

```
SELECT T1.full_name, T1.hero_name
FROM superhero AS T1
JOIN ( SELECT publisher
    FROM ML.GENERATE_TEXT(
        MODEL `a_generative_model`,
        (SELECT CONCAT(prompt, superhero.
            hero_name) AS input_text
          FROM superhero ),
        STRUCT(0 AS temperature)
      ) ) AS T2 ON T1.hero_name = T2.
          publisher
WHERE
  T2.publisher = 'Marvel';
```

Hybrid querying through UDFs offer more control for the database to optimize the hybrid query, build materialized views, and potentially reduce the amount of data generated by LLMs.

Since all four databases utilize SQLite, we can directly leverage BlendSQL [26], an extended version of the SQLite relational database management system that supports LLM functions. In SWAN, we provide 120 hybrid queries using the BlendSQL syntax, enabling SWAN to evaluate current systems in querying both relational databases and LLMs.

4.3 Optimization Opportunities

While these two solutions (HQDL and Hybrid Query UDFs) can be used to solve the challenges presented in our proposed SWAN benchmark, we believe that there are opportunities to improve upon these two solutions.

First, to answer these beyond database questions, what contexts, other than the necessary keys and the predefined value lists, should be presented in the prompt to reduce LLMs hallucination? There are other attributes inside the relational database that may be relevant and it remains an open question on how to select the best context. One possible approach is to build a vector index on the database values or rows and then fetch the relevant information based on embedding similarity [5, 37]. Second, the prompts and

Table 2: HQDL Execution Accuracy results on the SWAN benchmark using different number of demonstrations. The numbers in brackets report the accuracy improvement compared to the zero shot method.

Model	Demonstrations	California Schools	Super Hero	Formula One	European Football	Overall
GPT-3.5 Turbo	0-shot	50.0%	13.3%	16.7%	16.7%	**24.2%**
	1-shot	50.0%	23.3%	46.7%	26.7%	**36.7%** (+12.5%)
	3-shot	46.7%	20.0%	46.7%	33.3%	**36.7%** (+12.5%)
	5-shot	53.3%	20.0%	46.7%	33.3%	**38.3%** (+14.1%)
GPT-4 Turbo	0-shot	50.0%	23.3%	36.7%	16.7%	**31.6%**
	1-shot	43.3%	23.3%	50.0%	23.3%	**35.0%** (+3.3%)
	3-shot	50.0%	26.7%	50.0%	26.7%	**38.3%** (+6.7%)
	5-shot	56.7%	23.3%	50.0%	30.0%	**40.0%** (+8.4%)

static examples used in HQDL and Hybrid Query UDFs are handcrafted. It would be more convenient for users if the data system may automatically generate prompts and examples based on the specific context and query requirements. A promising direction is to develop a principled declarative prompt engineering toolkit [24]. HQDL requires LLMs to generate and materialize all missing data, while Hybrid Query UDFs, through BlendSQL, optimize queries by pushing down predicates to avoid generating unnecessary data entries. Additionally, reusing previously generated data in HQDL is straightforward. In BlendSQL, generated data is cached by mapping the LLM input prompt to its output data. However, prompts with similar meanings (e.g., "Is the superhero from the Marvel Universe?" versus "Does the hero come from Marvel?") cannot directly reuse previous results. A promising approach to address this is incorporating query rewriting within Hybrid Query UDFs to fully leverage all cached LLM-generated data [38]. Query optimization and caching are essential for reducing costs and increasing throughput, making hybrid queries more accessible and efficient. BlendSQL currently implemented batching—retrieving data values for multiple rows in a single LLM call—and plans to support parallelized LLM calls in the future to further minimize query latency.

5 EVALUATION

5.1 Evaluation Metrics

In the context of evaluating hybrid queries, we propose three metrics: execution accuracy (EX), data factuality, and the number of input/output tokens used by the LLMs.

Execution Accuracy (EX). EX is a well accepted metric in the domain of semantic parsing [7]. EX measures the percentage of hybrid queries that produce identical results to the ground truth (execution results from the Gold, correct, SQL). Since producing identical results is the end goal of hybrid querying, we adopt the EX metric.

Data Factuality. We use exact string match to verify the data factuality for each data cell value. Because of the one-to-many relationships (the key from a table maps to many values generated by LLMs), we use the widely accepted F1 score, which is a harmonic mean of precision and recall, to measure the overall factuality of generate data entries for each database [35].

Input and Output Tokens. We report the number of input and output tokens (i.e., words, sub-words) used in HQDL and Hybrid Query UDFs, which determine the monetary cost. For instance, GPT 3.5 Turbo priced at $3 per million input tokens and $6 per million output tokens.

5.2 Experiment Configurations

We evaluate HQDL and Hybrid Query UDFs on several OpenAI models (i.e., GPT-3.5 Turbo and GPT-4 Turbo) via OpenAI api calls. In all requests, we set the temperature to 0.

Few Shots Demonstrations. In the few-shot prompts, we provide static examples randomly selected from the original database. For HQDL, the few shots demonstrations are organized as static rows. In Hybrid Query UDFs, the few shots demonstrations are organized as a natural language question, an example database key, and the answer to the natural language question on the example database key (e.g., question: What is the driver code, key: Lewis Hamilton, and answer: HAM).

5.3 HQDL Results

In this subsection, we report and analyze the EX scores and the generated data factuality using F1 score.

Zero Shot. As shown in Table 2, in terms of overall accuracy over all databases using 0-Shot demonstrations, GPT-4 Turbo achieves 31.6% accuracy on the proposed SWAN benchmark, surpassing GPT-3.5 Turbo by 7.4%. One major challenge in using zero-shot prompts to generate data entries lies in ensuring that the output format is consistent, as this significantly impacts the ease of data extraction. Despite specifying the number of fields need to be returned in the input prompt, LLMs sometimes return too few or too many fields and may occasionally return an empty string for a field. Moreover, we can observe from Table 2 that for the California Schools database, GPT-4 Turbo achieved the same execution accuracy as GPT-3.5 Turbo. Questions in the California Schools frequently ask for the top schools. Consequently, queries answering these questions often include a LIMIT clause to retrieve only the top results. Hence, generating more accurate content for irrelevant entries does not necessarily lead to improvements in query execution accuracy. This observation motivated us to further examine the data factuality using F1 scores.

Few Shot. The distinction between the few-shots and zero-shot experiments is the inclusion of several static examples. We know

Table 3: HQ UDFs evaluation results on the SWAN benchmark. The numbers in brackets report the accuracy improvement compared to the zero shot method.

Model	Demonstrations	California Schools	Super Hero	Formula One	European Football	Overall
GPT-3.5 Turbo	0-shot	10.0%	23.3%	30.0%	10.0%	**18.3%**
	5-shot	13.3%	23.3%	43.3%	3.3%	**20.8%** (+2.5%)

Table 4: The average F1 score for measuring the factuality of the generated data using HQDL.

Model	Demonstrations	Average
GPT-3.5 Turbo	0-shot	**20.9%**
	1-shot	**37.3%**
	3-shot	**41.4%**
	5-shot	**42.7%**
GPT-4 Turbo	0-shot	**29.3%**
	1-shot	**47.0%**
	3-shot	**47.1%**
	5-shot	**48.2%**

that the capabilities of language models can be increased with examples provided for in-context learning [2]. Hence, we expect LLMs to generate more accurate data with few shots, and the execution accuracy should improve as more examples are provided. As shown in Table 2, in general the execution accuracy improves for both models as more examples are provided. When the prompt contains 5 static examples, GPT-3.5 Turbo achieves 38.3% and GPT-4 Turbo achieves 40.0% execution accuracy, 14.1% and 8.4% accuracy improvements compared to the zero shot method.

It is also interesting to note that both models achieve the highest execution accuracy on the California Schools and also the lowest execution accuracy on Super Hero database. One-third of the queries in California Schools database contain a LIMIT clause, retrieving the top schools. In contrast, many questions in the Super Hero database seek specific superheroes (e.g., heroes from Marvel or those with blue eyes), and only about one-tenth of the queries for this database include a LIMIT clause. One explanation for execution accuracy difference between questions in California Schools databse and questions in Super Hero database is that LLMs may exhibit biases, as previous research has shown that they tend to favor higher socioeconomic entities [17]. For instance, while LLMs can accurately identify schools with the highest standardized testing scores, they may struggle to identify schools with average or below-average grades. Because many queries in the California Schools database contain a LIMIT clause, even when an LLM provides inaccurate answers for many schools, the top results may still appear correct, masking potential errors in the model's full response.

Data Factuality To measure data factuality (using the F1 score), we use exact string matching to compare the generated data with the ground truth for each cell. Also, we compute the average F1 score over all cells for each database. When the generated content is identical to the ground truth, then it scores a 100% F1 score. As shown in Table 4, GPT-4 Turbo consistently generates more factual information than GPT-3.5 Turbo using the same prompt. For instance, with the 5-shot prompt, GPT-4 Turbo scores 5.5% higher than GPT-3.5 Turbo. Also, the results clearly showcase that providing more examples in the input prompt increases the factuality of the generated output, which also leads to higher execution accuracy when executing the hybrid queries.

5.4 Hybrid Query UDFs Results

We evaluated the performance of BlendSQL [26] on the SWAN dataset to assess its effectiveness with hybrid query UDFs. Notably, on GPT-3.5 Turbo, the execution accuracy for 0-shot and 5-shot settings reached 18.3% and 20.8% (see Table 3), which are lower compared to HQDL's results of 24.2% and 38.3%. In our evaluation of hybrid query UDFs, we provided the keys and instructed the LLM to predict only the necessary information (most of the time a single cell value). This approach contrasts significantly with HQDL, where the LLM is given the key and tasked with predicting all column values for the corresponding row. Predicting all column values may be more advantageous than predicting a single column value, as it mirrors a chain-of-thought process that enables the model to leverage inter-dependencies between columns, thereby enhancing accuracy and coherence in its predictions. In HQDL, each LLM call generates a single row. In contrast, BlendSQL uses a default batch size of 5, where each request combines five keys into a list, prompting the LLM to return a list of five data entries corresponding to the five keys. Although batching reduce the number of LLM calls, it also increases the potential for errors, as processing multiple entries in a single call may lead to inaccuracies in the returned data [4].

Another noteworthy difference between HQDL and HQ UDFS is in the format of the few-shot examples. In HQDL, we included static rows in the prompt as few-shot demonstrations, and the goal is to showcase the model how to complete a row of data based on the given keys. In contrast, for HQ UDFS (e.g., BlendSQL), we curated a list of question-answer pairs for each databases, and then BlendSQL selects relevant examples based on similarity metrics (e.g., cosine similarity using a sentence transformer) to find the most similar questions. For example, a demonstration question in the HQ UDFS system might be: 'Provide the city name based on the address.' Along with this question, given the address '5328 Brann Street', the expected city name is 'Oakland'.

5.5 Evaluation Costs

The monetary costs and the system's performance (e.g., latency and throughout) are implicitly determined by the number of input and output tokens. Here, we report the total number of input and output tokens for HQDL and HQ UDFs.

Table 5: Total tokens used for HQDL and HQ UDFs for zero shot experiments.

Algorithm	Input Tokens	Output Tokens
HQDL	6.3 M	1.5 M
HQ UDFs	23 M	2 M

HQDL generates data entries for all the missing columns. As shown in Table 5, using zero-shot prompt, a total of 6.3 million tokens are used as inputs for LLMs, and 1.5 million tokens are generated by LLMs. On average, about 52k input tokens and 12k output tokens are used per beyond-database questions. If the number of beyond-database questions increases, the cost per question will decrease.

For HQ UDFS, we expected it to use less tokens compared to HQDL, because HQ UDFS give more control to the database query optimizer. This allows the system to intelligently minimize token usage by pushing down predicates, meaning it generates tokens only for the specific data cells needed to answer the query. Surprisingly, for zero-shot prompt, the total input and output tokens are 23 million and 2 million respectively, as shown in Table 5. Compared to HQDL, HQ UDFs uses 3.6x more input tokens and 1.3x more output tokens.

The increased costs of HQ UDFs can be attributed to its limited use of cached results. For instance, to answer the beyond-database question, 'What is the height of the tallest player?', HQ UDFS used the LLMs to generate heights for all players, as the database lacks this information. Later, another question asks, 'Please list player names who are taller than 180cm.' The corresponding hybrid query created for this question prompts the LLMs to answer 'Is the player taller than 180cm?' However, it is evident that the previously generated heights could be directly reused to answer this question, rather than generating new responses. In HQ UDFS, LLM-generated content is cached as a mapping from input prompts to LLM output answers, making it challenging for the system to efficiently reuse cached outputs. In contrast, HQDL stores LLM-generated outputs directly as entities within relationships (schema expansion), simplifying reuse for users.

6 CONCLUSION AND FUTURE WORK

In this paper, we present the first benchmark, **SWAN**, for evaluating hybrid queries that answer beyond-database questions using relational databases and large language models. In addition to the benchmark, we also introduce HQDL, a preliminary solution for answering questions in SWAN based on schema expansion. We also provide queries to evaluate current Hybrid Query UDFs systems (e.g., BlendSQL). Our evaluation demonstrates that there are still many opportunities for improving the execution accuracy and also increasing the overall efficiency. To improve the execution accuracy and ensure high data fidelity, retrieve-augmented generation (RAG) [16] and supervised fine-tuning [6, 29] are two promising direction to be integrated in hybrid querying systems. Moreover, there are numerous opportunities to optimize the pipeline for executing hybrid queries, increasing throughput and lowering monetary costs. For example, to further reduce costs and improve system throughput, it is essential to focus on: (i) implementing asynchronous and parallel hybrid query execution, (ii) designing improved caching mechanisms, and (iii) fully utilizing cached content.

Additionally, in the current benchmark, we provided the missing columns for schema expansion and pre-written queries for both HQDL and HQ UDFS. In future work, the process of answering beyond-database questions should be fully automated. Given a natural language question, LLMs should first evaluate whether it can be answered using the existing schema. For questions requiring information beyond the current database, LLMs could be designed to automatically expand the schema, populate missing values, and generate a SQL query (similar to HQDL) or construct a SQL query with user-defined functions to directly prompt LLMs for required information in real time.

We envision that our benchmark, the two baseline solutions, and the discussions on future optimization opportunities will spark interests within the community to develop comprehensive data systems that leverage the full potential of relational databases and large language models.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable feedback and Parker Glenn for assistance with running BlendSQL on SWAN. Fuheng Zhao was partially funded by Microsoft PhD Fellowship.

REFERENCES

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[3] Zui Chen, Lei Cao, Sam Madden, Ju Fan, Nan Tang, Zihui Gu, Zeyuan Shang, Chunwei Liu, Michael Cafarella, and Tim Kraska. 2023. Seed: Simple, efficient, and effective data management via large language models. *arXiv:2310.00749* (2023).

[4] Zhoujun Cheng, Jungo Kasai, and Tao Yu. 2023. Batch prompting: Efficient inference with large language model apis. *arXiv preprint arXiv:2301.08721* (2023).

[5] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130* (2024).

[6] Mohamed Elaraby, Mengyin Lu, Jacob Dunn, Xueying Zhang, Yu Wang, Shizhu Liu, Pingchuan Tian, Yuping Wang, and Yuxuan Wang. 2023. Halo: Estimation and reduction of hallucinations in open-source weak large language models. *arXiv preprint arXiv:2308.11764* (2023).

[7] Avrilia Floratou, Fotis Psallidas, Fuheng Zhao, Shaleen Deep, and et. al. 2024. NL2SQL is a solved problem... Not!. In *Conference on Innovative Data Systems Research*. https://api.semanticscholar.org/CorpusID:266729311

[8] Michael J Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. 2011. CrowdDB: answering queries with crowdsourcing. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. 61–72.

[9] Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363* (2023).

[10] Victor Giannakouris and Immanuel Trummer. 2024. Demonstrating λ-Tune: Exploiting Large Language Models for Workload-Adaptive Database System Tuning. In *Companion of the International Conference on Management of Data*.

[11] Parker Glenn, Parag Pravin Dakle, Liang Wang, and Preethi Raghavan. 2024. BlendSQL: A Scalable Dialect for Unifying Hybrid Question Answering in Relational Algebra. *arXiv preprint arXiv:2402.17882* (2024).

[12] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300* (2020).

[13] Ravi Kashyap. 2023. Machine Learning in Google Cloud Big Query using SQL. *SSRG International Journal of Computer Science and Engineering* 10, 5 (2023).

[14] Alon Y Levy. 1996. Obtaining complete answers from incomplete databases. In *VLDB*, Vol. 96. Citeseer, 402–412.

[15] Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. 2024. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems* 36 (2024).

[16] Jiarui Li, Ye Yuan, and Zehua Zhang. 2024. Enhancing llm factual accuracy with rag to counter hallucinations: A case study on domain-specific queries in private knowledge-bases. *arXiv preprint arXiv:2403.10446* (2024).

[17] Rohin Manvi, Samar Khanna, Marshall Burke, et al. 2024. Large language models are geographically biased. *arXiv:2402.02680* (2024).

[18] Adam Marcus, Eugene Wu, David R Karger, Samuel Madden, and Robert C Miller. 2011. Crowdsourced databases: Query processing with people. Cidr.

[19] Avanika Narayan, Ines Chami, Laurel Orr, Simran Arora, and Christopher Ré. 2022. Can foundation models wrangle your data? *arXiv:2205.09911* (2022).

[20] AI Open. 2023. New models and developer products announced at DevDay.

[21] OpenAI. 2024. OpenAI Platform. https://platform.openai.com/examples/default-sql-translate. Accessed: 30 April 2024.

[22] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[23] Aditya Parameswaran and Neoklis Polyzotis. 2011. Answering queries using humans, algorithms and databases. (2011).

[24] Aditya G Parameswaran, Shreya Shankar, Parth Asawa, Naman Jain, and Yujie Wang. 2023. Revisiting prompt engineering via declarative crowdsourcing. *arXiv preprint arXiv:2308.03854* (2023).

[25] Hyunjung Park, Richard Pang, Aditya Parameswaran, Hector Garcia-Molina, Neoklis Polyzotis, and Jennifer Widom. 2012. Deco: A system for declarative crowdsourcing. (2012).

[26] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–22.

[27] Mohammadreza Pourreza and Davood Rafiei. 2024. Din-sql: Decomposed in-context learning of text-to-sql with self-correction. *Advances in Neural Information Processing Systems* 36 (2024).

[28] Mark Raasveldt and Hannes Mühleisen. 2019. Duckdb: an embeddable analytical database. In *Proceedings of the 2019 International Conference on Management of Data*. 1981–1984.

[29] Evgeniia Razumovskaia, Ivan Vulić, Pavle Marković, Tomasz Cichy, Qian Zheng, Tsung-Hsien Wen, and Paweł Budzianowski. 2024. Dial beinfo for faithfulness: Improving factuality of information-seeking dialogue via behavioural fine-tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. 17139–17152.

[30] Raymond Reiter. 1988. What should a database know?. In *Proceedings of the seventh ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*.

[31] Mohammed Saeed, Nicola De Cao, and Paolo Papotti. 2023. Querying large language models with SQL. *arXiv preprint arXiv:2304.00472* (2023).

[32] Michaek Stonebraker and Andrew Pavlo. 2024. What Goes Around Comes Around… And Around…. *ACM Sigmod Record* 53, 2 (2024), 21–37.

[33] Matthias Urban, Duc Dat Nguyen, and Carsten Binnig. 2023. OmniscientDB: a large language model-augmented DBMS that knows what other DBMSs do not know. In *Proceedings of the Sixth International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*. 1–7.

[34] Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, et al. 2023. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *arXiv preprint arXiv:2310.07521* (2023).

[35] Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, Aleksandr Rubashevskii, et al. 2023. Factcheck-GPT: End-to-End Fine-Grained Document-Level Fact-Checking and Correction of LLM Output. *arXiv preprint arXiv:2311.09000* (2023).

[36] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887* (2018).

[37] Haochen Zhang, Yuyang Dong, Chuan Xiao, and Masafumi Oyamada. 2023. Large language models as data preprocessors. *arXiv preprint arXiv:2308.16361* (2023).

[38] Fuheng Zhao, Lawrence Lim, Ishtiyaque Ahmad, Divyakant Agrawal, and Amr El Abbadi. 2023. LLM-SQL-Solver: Can LLMs Determine SQL Equivalence? *arXiv preprint arXiv:2312.10321* (2023).

[39] Xuanhe Zhou, Zhaoyan Sun, and Guoliang Li. 2024. Db-gpt: Large language model meets database. *Data Science and Engineering* 9, 1 (2024), 102–111.