Sampling Methods for Inner Product Sketching

Majid Daliri
New York University
daliri.majid@nyu.edu

Juliana Freire
New York University
juliana.freire@nyu.edu

Christopher Musco
New York University
cmusco@nyu.edu

Aécio Santos
New York University
aecio.santos@nyu.edu

Haoxiang Zhang
New York University
haoxiang.zhang@nyu.edu

ABSTRACT

Recently, Bessa et al. (PODS 2023) showed that sketches based on coordinated weighted sampling theoretically and empirically outperform popular linear sketching methods like Johnson-Lindentrauss projection and CountSketch for the ubiquitous problem of inner product estimation. We further develop this finding by introducing and analyzing two alternative sampling-based methods. In contrast to the computationally expensive algorithm in Bessa et al., our methods run in linear time (to compute the sketch) and perform better in practice, significantly beating linear sketching on a variety of tasks. For example, they provide state-of-the-art results for estimating the correlation between columns in unjoined tables, a problem that we show how to reduce to inner product estimation in a black-box way. While based on known sampling techniques (threshold and priority sampling) we introduce significant new theoretical analysis to prove approximation guarantees for our methods.

PVLDB Reference Format:
Majid Daliri, Juliana Freire, Christopher Musco, Aécio Santos,
and Haoxiang Zhang. Sampling Methods for Inner Product Sketching.
PVLDB, 17(9): 2185 - 2197, 2024.
doi:10.14778/3665844.3665850

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at
https://github.com/VIDA-NYU/SamplingMethodsForInnerProductSketching.

1 INTRODUCTION

We study methods for approximating the inner product $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i$ between two length n vectors $\mathbf{a}$ and $\mathbf{b}$. We are interested in algorithms that independently compute compact *sketches* $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ of $\mathbf{a}$ and $\mathbf{b}$, and approximate $\langle \mathbf{a}, \mathbf{b} \rangle$ using only the information in these sketches. $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ should take much less than n space to store, allowing them to be quickly retrieved from disk or transferred over a network. Additionally, both the sketching procedure $\mathbf{a} \rightarrow \mathcal{S}(\mathbf{a})$ and the estimation procedure that returns an approximation to $\langle \mathbf{a}, \mathbf{b} \rangle$ should be computationally efficient, ideally running in linear time. We note that computing an inner product between two length n vectors naively takes just $O(n)$ time. As such, the goal of sketching methods is not to speed up a single inner product but rather to speed up many. The methods we study can compute sketches of size m for a collection of D length-n vectors in $O(nD)$ time. We can then estimate all pairwise inner products between those vectors in $O(D^2 m)$ time, significantly faster than the baseline $O(D^2 n)$ time when $m \ll n$. Sketching methods for the inner product have been studied for decades and find applications throughout data science and databases. They can be used to quickly compute document similarity, to speed up evaluation of machine learning models, and to estimate quantities like join size [1, 4, 25, 50, 51]. Recently, inner product sketching has found applications in scalable dataset search and augmentation, where sketches can be used to estimate correlations between columns in unjoined tables [52]. In such applications, we have a large repository of D vectors that we wish to compare against a query vector using inner products. By sketching the database, we can evaluate new queries much more efficiently than the naive $O(Dn)$ time.

1.1 Prior Work

Inner Product Estimation via Linear Sketching. Until recently, all sketching algorithms with strong worst-case accuracy guarantees for approximating the inner product between arbitrary inputs were based on *linear sketching*. Such methods include Johnson-Lindenstrauss random projection (JL) [1, 28], the closely related AMS sketch [3, 4], and the CountSketch algorithm [11, 23]. These methods are considered "linear" because the sketching operation $\mathbf{a} \rightarrow \mathcal{S}(\mathbf{a})$ is a linear map, meaning that $\mathcal{S}(\mathbf{a}) = \mathbf{\Pi a}$ for a matrix $\mathbf{\Pi} \in \mathbb{R}^{m \times n}$. $\mathbf{\Pi}$ is typically chosen at random and its row count m is equal to the size of the sketch $\mathcal{S}(\mathbf{a})$. To estimate the inner product between $\mathbf{a}$ and $\mathbf{b}$, the standard approach is to simply return $\langle \mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b}) \rangle = \langle \mathbf{\Pi a}, \mathbf{\Pi b} \rangle$. For all common linear sketching methods (including those listed above), it can be shown (see e.g., [5]) that, if we choose the sketch size $m = O\left(1/\epsilon^2\right)$, then with high probability:

$$|\langle \mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b}) \rangle - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \epsilon \|\mathbf{a}\|_2 \|\mathbf{b}\|_2. \tag{1}$$

Here $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} \mathbf{x}_i^2}$ denotes the Euclidean norm of a vector $\mathbf{x}$.

Better Accuracy via Weighted MinHash. While (1) is a strong guarantee, it was recently improved by Bessa et al. [6], who introduce a method based on the popular Weighted MinHash (WMH) algorithm [14, 35, 46, 54]. Like unweighted MinHash and techniques such as conditional random sampling [8, 43], the WMH sketch contains a subsample of entries from $\mathbf{a}$ and $\mathbf{b}$ that can be used to approximate the inner product. Importantly, entries with higher absolute value are sampled with higher probability, since they can contribute more to the inner product sum $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i$. Using sketches of size $O(1/\epsilon^2)$, WMH achieves accuracy:

$$|\langle \mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b}) \rangle - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \epsilon \max\left(\|\mathbf{a}_{\mathcal{I}}\|_2 \|\mathbf{b}\|_2, \|\mathbf{a}\|_2 \|\mathbf{b}_{\mathcal{I}}\|_2\right). \tag{2}$$

Proceedings of the VLDB Endowment, Vol. 17, No. 9 ISSN 2150-8097.
doi:10.14778/3665844.3665850

Table 1: Comparison of error guarantees and computational cost for sketching methods when used to estimate the inner product between vectors a and b. Note that $\epsilon \cdot \max(\|\mathbf{a}_{\mathcal{I}}\|_2\|\mathbf{b}\|_2, \|\mathbf{a}\|_2\|\mathbf{b}_{\mathcal{I}}\|_2)$ is always a better guarantee than $\epsilon \cdot \|\mathbf{a}\|_2\|\mathbf{b}\|_2$, and often significantly so when a and b are sparse with limited overlap between non-zero entries. Our Threshold and Priority Sampling methods obtain this better bound while matching or nearly matching the fast runtime of the less accurate CountSketch method.

Method	**High probability error guarantee for sketch of size $m = O(1/\epsilon^2)$**	**Time to compute sketch for length n vector with N non-zero entries**	**Strict bound on sketch size?**
JL Projection/AMS Sketch [4, 5]	$\epsilon \cdot \|\mathbf{a}\|_2\|\mathbf{b}\|_2$	$O(Nm)$	✓
CountSketch/Fast-AGMS [11, 23]	$\epsilon \cdot \|\mathbf{a}\|_2\|\mathbf{b}\|_2$	$O(N)$	✓
Weighted MinHash (WMH) [6]	$\epsilon \cdot \max(\|\mathbf{a}_{\mathcal{I}}\|_2\|\mathbf{b}\|_2, \|\mathbf{a}\|_2\|\mathbf{b}_{\mathcal{I}}\|_2)$	$O(Nm \log n)$	✓
Threshold Sampling	$\epsilon \cdot \max(\|\mathbf{a}_{\mathcal{I}}\|_2\|\mathbf{b}\|_2, \|\mathbf{a}\|_2\|\mathbf{b}_{\mathcal{I}}\|_2)$	$O(N)$	✗
Priority Sampling	$\epsilon \cdot \max(\|\mathbf{a}_{\mathcal{I}}\|_2\|\mathbf{b}\|_2, \|\mathbf{a}\|_2\|\mathbf{b}_{\mathcal{I}}\|_2)$	$O(N \log m)$	✓

Here $\mathcal{I} = \{i : \mathbf{a}[i] \neq 0 \text{ and } \mathbf{b}[i] \neq 0\}$ is the set of all indices in the *intersection* of the supports of $\mathbf{a}$ and $\mathbf{b}$, and $\mathbf{a}_{\mathcal{I}}$ and $\mathbf{b}_{\mathcal{I}}$ denote the vectors restricted to the indices in $\mathcal{I}$.[1] Since we always have $\|\mathbf{a}_{\mathcal{I}}\|_2 \leq \|\mathbf{a}\|_2$ and $\|\mathbf{b}_{\mathcal{I}}\|_2 \leq \|\mathbf{b}\|_2$, the error in (2) is always less or equal to the error in (1) for the linear sketching methods.

As confirmed by experiments in [6], the improvement over linear sketching can be significant in applications where $\mathbf{a}$ and $\mathbf{b}$ are sparse and their non-zero entries only overlap at a small fraction of indices. I.e., when $|\mathcal{I}|$ is much smaller than the number of non-zeros in $\mathbf{a}$ and $\mathbf{b}$. This is common when inner product sketches are used for data discovery, either to estimate join-sizes or correlations between unjoined tables [9, 59, 61]. In these applications, overlap between non-zeros in $\mathbf{a}$ and $\mathbf{b}$ corresponds to overlap between the keys of the tables being joined, which is often small. For example, consider a setting where we want to find additional data for use in taxi demand prediction. Given a table of 2022-2023 taxi trip data, we would like to augment it using weather information available in a table of historical weather data from the last 50 years; this leads to just a 4% overlap in keys. More examples are discussed in Section 4.

Limitations of WMH sketches. While WMH provides better accuracy than linear sketching, it has important limitations. Notably, the method has high computational complexity, requiring $O(Nm \log n)$ time to produce a sketch of size m from a length n vector $\mathbf{a}$ with $N \leq n$ non-zero entries. While this nearly matches the $O(Nm)$ complexity of a JL projection or AMS sketch (which require multiplying $\mathbf{a}$ by a dense matrix), it is far slower than methods like CountSketch or the k-minimum values (KMV) sketch [7], which can be applied in $O(N)$ or $O(N \log m)$ time, respectively. It is possible to reduce the complexity of WMH to $O(N + m \log m)$ using recent work [15, 31]. However, as shown in Section 5, even these theoretically faster methods are orders of magnitude slower in practice than the simpler sketches introduced in our work.

Beyond computational cost, another disadvantage of WMH is that it is complex, both to implement and analyze. For example, while a high probability bound is obtained in [6], they are unable to analyze the variance of the method. This makes it difficult, for example, to compute confidence intervals. Also, while it does not effect the Big O claim that a sketch of size $O(1/\epsilon^2)$ achieves error guarantee (2), the practical accuracy of WMH is negatively impacted by the fact that it samples entries from $\mathbf{a}$ and $\mathbf{b}$ *with replacement*, which can lead to redundancy in the sketches $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$.

1.2 Our Contributions

Methods and Theory. In this paper, we present and analyze two algorithms for inner product sketching that eliminate the limitations of WMH sketches, while maintaining the same strong theoretical guarantees. Both are based on existing methods for weighted sampling of vectors *without replacement*, but our choice of sampling probabilities, estimation procedure, and theoretical analysis are new, and tailored to the problem of inner product estimation.

The first method we study is based on *Threshold Sampling* [30, 34]. We show that, when used to sample vector entries with probability proportional to their squared value, this method produces inner product sketches that yield the same accuracy guarantee as WMH sketches. At the same time, the method is extremely simple to implement and can be applied to a vector with N non-zero entries in linear $O(N)$ time. Moreover, unlike WMH, the analysis of the method is straightforward. Its only disadvantage is that Threshold Sampling produces sketches that *randomly vary* in size. The user can specify a parameter m and is guaranteed that the sketch has size m in *expectation*, and will not exceed $m + O(\sqrt{m})$ with high-probability. However, there is no hard bound.

We address this drawback with an alternative method based on Priority Sampling, which has been widely studied in the sketching and statistics literature [29, 48, 56]. Priority Sampling offers a hard sketch size bound and can construct a size m sketch in near-linear $O(N \log m)$ time. While significantly more challenging to analyze than Threshold Sampling, by introducing a new estimation procedure and building on a recent analysis of Priority Sampling for a different problem (subset sum estimation) [27], we are able to show that it enjoys the same guarantees as WMH. Our analysis of Priority Sampling is the main theoretical contribution of this paper.

Experimental Results. In addition to theoretical analysis, we experimentally compare Threshold and Priority Sampling with linear sketching algorithms like JL random projections and CountSketch, as well as sampling-based sketches like k-minimum values (KMV)[2],

[1] Prior to the work of [6], the stronger guarantee of (2) was known to be obtainable for the special case of inner product of binary vectors, which corresponds to the set intersection problem [49].

[2] The KMV sketch is not typically thought of as a sketch for estimating inner products between arbitrary vectors, but can be modified to do so. See [6] for details.

MinHash, and WMH. We evaluate these on a variety of applications, including join size estimation and correlation estimation between unjoined tables. We introduce an approach to perform *join-correlation estimation* [52] using *any* inner product sketching method (Section 4) that we believe may be of independent interest.

Our Threshold and Priority Sampling methods offer much better accuracy than the baselines, beating both linear sketches and WMH sketches. An optimized version of our sketches tailored to join-correlation estimation outperforms the recent Correlation Sketches method from [52], which is based on KMV. We also test the run-time efficiency of our method for sketch construction. Even when WMH is implemented using the efficient DartMinHash algorithm [15], our methods are faster by more than an order of magnitude.

Our Approach. As in [6], sketches consist of samples from $\mathbf{a}$ and $\mathbf{b}$. We estimate the inner product $\sum_{i=1}^{n} \mathbf{a}_i\mathbf{b}_i$ using only a subset of terms in the sum. Specifically, our estimators are of the form $\sum_{j\in\mathcal{T}} w_j \cdot \mathbf{a}_j\mathbf{b}_j$, where $\mathcal{T}$ is a subset of $\{1,\dots,n\}$ and $\{w_j, j \in \mathcal{T}\}$ are appropriately chosen positive weights. To compute this estimate, we need to store *both* $\mathbf{a}_j$ in $\mathcal{S}(\mathbf{a})$ and $\mathbf{b}_j$ and $\mathcal{S}(\mathbf{b})$. If $\mathbf{a}$ and $\mathbf{b}$ are sampled independently at random, the probability of obtaining matching indices in both sketches would be small, thus leading to a small number of usable samples, and a poor inner product estimate. Our Threshold and Priority Sampling methods avoid this issue by using shared random seeds to sample from the vectors in a *coordinated way*, which ensures that if entry $\mathbf{a}_j$ is sampled from $\mathbf{a}$, it is more likely that the corresponding $\mathbf{b}_j$ is sampled from $\mathbf{b}$. This idea is not new: coordinated variants of Threshold and Priority Sampling have been studied in prior work on different problems, as have coordinated variants of related methods like PPSWOR sampling [17, 19]. What *is new* is how we apply and analyze such methods for the problem of inner product estimation.

Besides WMH [6], we are only aware of one prior paper that addresses the inner product estimation problem using coordinated sampling: the "End-Biased Sampling" algorithm of [33] can be viewed as a variant of Threshold Sampling where the i^{th} entry of $\mathbf{a}$ is sampled with probability proportional to the magnitude $|\mathbf{a}_i|$. We instead use the squared magnitude $|\mathbf{a}_i|^2$. While variance bounds are shown in [33], due to this choice of sampling probability, they fall short of improving on results for linear sketches, i.e., on Eq. (1). Additionally, unlike our work, [33] does not address the issue of how to obtain a fixed-size sketch. We discuss End-Biased Sampling further in Section 5 and fully review related work in Section 6.

Paper Roadmap. Our contributions can be summarized as follows:

- We show how to apply two coordinated sampling methods, Threshold and Priority Sampling, to the inner product sketching problem, invoking these methods with a specific choice of sampling probabilities and estimation procedures.
- We prove that these methods enjoy better theoretical accuracy guarantees than linear sketches, and match the best-known guarantees provided by WMH [6] (Section 2 and Section 3).
- We perform an empirical evaluation, showing that Threshold and Priority Sampling outperform state-of-the-art sketches in both accuracy and run-time on a variety of applications (Section 5).
- We show a black-box reduction from one such application, join-correlation estimation, to inner product estimation (Section 4).

2 THRESHOLD SAMPLING

We begin by introducing an inner product sketch based on Threshold Sampling, which is a method popularized in computer science by [30], but long studied in statistics under the name "Poisson Sampling".[3] Our algorithm based on Threshold Sampling is straightforward to implement and analyze, but still matches the strong theoretical guarantees of WMH sketches [6], while improving on runtime and performance. Its presentation serves as a warm-up for our Priority Sampling method (Section 3), which is more difficult to analyze, but has the advantage of a deterministic sketch size.

Sketching. As discussed, the goal of our sketching methods (and of WMH) is to randomly sample entries from $\mathbf{a}$ and $\mathbf{b}$, and to use those samples to estimate the inner product sum $\langle \mathbf{a}, \mathbf{b}\rangle = \sum_{i=1}^{n} \mathbf{a}_i\mathbf{b}_i$. To obtain strong guarantees, we need the samples to be both *coordinated* and *weighted.* In particular, since they contribute more to the inner product, entries with larger magnitude should be sampled with higher probability. Moreover, coordination requires that $\mathbf{b}_j$ is more likely to be sampled if $\mathbf{a}_j$ is. Ensuring coordination is not obvious because, in the sketching setting we consider, $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ need to be computed completely independently from each other: when we sample entries from $\mathbf{b}$ to form $\mathcal{S}(\mathbf{b})$, we have no knowledge about what entries were sampled from $\mathbf{a}$ to form $\mathcal{S}(\mathbf{a})$.

Algorithm 1 Threshold Sampling

```
Input: Length $n$ vector $\mathbf{a}$, random seed $s$, target sketch size $m$.
Output: Sketch $\mathcal{S}(\mathbf{a}) = \{K_{\mathbf{a}}, V_{\mathbf{a}}, \tau_{\mathbf{a}}\}$, where $K_{\mathbf{a}}$ is a subset of indices
    from $\{1,\dots,n\}$ and $V_{\mathbf{a}}$ contains $\mathbf{a}_i$ for all $i \in K_{\mathbf{a}}$.
1: Use random seed $s$ to select a uniformly random hash function
   $h : \{1,...,n\} \to [0,1]$. Initialize $K_{\mathbf{a}}$ and $V_{\mathbf{a}}$ to be empty lists.
2: for $i$ such that $\mathbf{a}[i] \neq 0$ do
3:     Set threshold $\tau_i = m \cdot \frac{\mathbf{a}_i^2}{\|\mathbf{a}\|_2^2}$.
4:     if $h(i) \leq \tau_i$ then
5:         Append $i$ to $K_{\mathbf{a}}$, append $\mathbf{a}_i$ to $V_{\mathbf{a}}$.
6: return $\mathcal{S}(\mathbf{a}) = \{K_{\mathbf{a}}, V_{\mathbf{a}}, \tau_{\mathbf{a}}\}$ where $\tau_{\mathbf{a}} = m/\|\mathbf{a}\|_2^2$.
```

Threshold Sampling achieves sampling that is both weighted and coordinated using a simple technique. We first assume access to a hash function $h : \{1,\dots,n\} \to [0,1]$ that maps indices to uniformly random real numbers in the interval $[0,1]$. Assuming access to such a function is standard in the literature, and we note that, in practice, h can be replaced with a pseudorandom function that maps to a sufficiently large discrete set, e.g., to $\{1/U, 2/U \dots, 1\}$ for $U = 2^{32}$ or some other large integer [7, 25]. As shown in Algorithm 1 and illustrated in Figure 1, we sketch the vector $\mathbf{a}$ by selecting a threshold, τ_i for each index (Line 3). We then hash all indices i for which $\mathbf{a}[i] \neq 0$ to the interval $[0,1]$, and keep as a sample all entries of $\mathbf{a}$ for which the hash value $h(i)$ is below the threshold (Line 4,5).

Concretely, we choose the threshold $\tau_i = m \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2$. Here m is a fixed parameter that controls the size of the final sketch, $\mathcal{S}(\mathbf{a})$, returned by Algorithm 1. So, we see that the threshold τ_i is higher for indices i where $\mathbf{a}_i^2$ is larger. Thus, larger entries in the vector are sampled with higher probability. Note that this is in contrast to

[3] A variant of Threshold Sampling with *uniform* probabilities was also studied under the name "adaptive sampling" by Wegman in 1984 and later by Flajolet [34].

index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
a	0	0	2.5	0	0	2.3	0	4	0	0	0.5	0	3	0	0	-3.7
b	0	0	-3.1	0	0	0	0.4	-4.2	0	1.5	1	0	-2.6	-5.9	0	0

(a) Vectors $\mathbf{a}, \mathbf{b}$ to be sketched. Their inner product is $\langle \mathbf{a}, \mathbf{b} \rangle = -31.85$.

i	$h(i)$	$\tau_i(\mathbf{a})$	$\tau_i(\mathbf{b})$
3	0.11	0.495	0.532
6	0.39	0.419	✗
7	0.92	✗	0.009
8	0.14	1.268	0.977
10	0.42	✗	0.125
11	0.8	0.020	0.055
13	0.43	0.713	0.374
14	0.07	✗	1.928
16	0.23	1.085	✗

$K_\mathbf{a}$	$V_\mathbf{a}$
3	2.5
8	4
13	3
16	-3.7

$$\underbrace{\tau_\mathbf{a} = .079}_{\mathcal{S}(\mathbf{a})}$$

$K_\mathbf{b}$	$V_\mathbf{b}$
3	-3.1
8	-4.2
14	-5.9

$$\underbrace{\tau_\mathbf{b} = .055}_{\mathcal{S}(\mathbf{b})}$$

(b) Example sketches $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ obtained using Algorithm 1 with target sketch size $m = 4$. Since the size of the sketch returned by the method is random, $\mathcal{S}(\mathbf{a})$ has size 4, but $\mathcal{S}(\mathbf{b})$ is slightly smaller. The columns $\tau_i(\mathbf{a}) = m \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2$ and $\tau_i(\mathbf{b}) = m \cdot \mathbf{b}_i^2/\|\mathbf{b}\|_2^2$ contain the thresholds computed in Line 3 of Algorithm 1. Thresholds are only computed for non-zero entries since we never sample entries with value 0. The highlighted thresholds correspond to items that are included in the sketch, i.e., the threshold is larger than the hash value $h(i)$. If the sketches $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ above are used used in our estimator from Algorithm 2, we obtain an approximate inner product of -32.85, which is close to the true inner product of -31.85.

Figure 1: Sketching with Threshold Sampling (Algorithm 1).

"End-Biased Sampling" [33], which sets $\tau_i = m \cdot \frac{|\mathbf{a}_i|}{\|\mathbf{a}\|_1}$, where $\|\mathbf{a}\|_1 = \sum_{i=1}^n |\mathbf{a}_i|$ is the ℓ_1 norm. While this choice also aligns with the goal that larger entries should be sampled with higher probability, it does not lead to the same strong theoretical guarantees.

In addition to collecting a weighted sample, since the *same hash function h* is used when sampling from both $\mathbf{a}$ and $\mathbf{b}$, the samples are coordinated. If $h(i)$ is small, we are more likely sample *both* $\mathbf{a}_i$ and $\mathbf{b}_i$. The same idea is present in common methods for unweighted coordinated sampling like MinHash or the KMV sketch [7, 8].

Finally, we note that the sketch procedure in Algorithm 1 runs in $O(N)$ time when $\mathbf{a}$ has N non-zero entries, at least when the vector is stored in a standard sparse-vector format (e.g., a key/value store) which allows iteration over the non-zero entries in $O(N)$ time.[4]

Estimation. Once our sketches $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ are computed, to estimate the inner product between $\mathbf{a}$ and $\mathbf{b}$, we simply compute a weighted sum between entries that are sampled in both $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ (see Algorithm 2). To ensure the sum equals the true inner product $\langle \mathbf{a}, \mathbf{b} \rangle$ in expectation, the weight for index i in the sum is the inverse of the probability that *both* $\mathbf{a}_i$ and $\mathbf{b}_i$ were included in the sketches $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$. We can check that this probability is equal to $\min\left(1, m \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2, m \cdot \mathbf{b}_i^2/\|\mathbf{b}\|_2^2\right)$. This can be computed in $O(1)$ time, so overall the estimator can be computed in time linear in the sketch size. Note that the estimator requires knowledge of the scaling parameters $m/\|\mathbf{a}\|_2^2$ and $m/\|\mathbf{b}\|_2^2$, so we include these numbers in our sketches $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ as $\tau_\mathbf{a}$ and $\tau_\mathbf{b}$.

Algorithm 2 Inner Product Estimator

```
Input: Sketches $\mathcal{S}(\mathbf{a}) = \{K_\mathbf{a}, V_\mathbf{a}, \tau_\mathbf{a}\}$, $\mathcal{S}(\mathbf{b}) = \{K_\mathbf{b}, V_\mathbf{b}, \tau_\mathbf{b}\}$ constructed by Algorithm 1 or Algorithm 3 with the same seed $s$.
Output: Estimate $w$ of $\langle \mathbf{a}, \mathbf{b} \rangle$.
Compute $\mathcal{T} = K_\mathbf{a} \cap K_\mathbf{b}$. Note that for all $i \in \mathcal{T}$, $V_\mathbf{a}$ and $V_\mathbf{b}$ contain $\mathbf{a}_i$ and $\mathbf{b}_i$.
return $W = \sum_{i \in \mathcal{T}} \frac{\mathbf{a}_i \mathbf{b}_i}{\min(1, \mathbf{a}_i^2 \cdot \tau_\mathbf{a}, \mathbf{b}_i^2 \cdot \tau_\mathbf{b})}$.
```

Comparison to WMH. While both WMH and Threshold Sampling use coordinated weighted sampling, WMH does so in a less efficient way. It creates a variable number of copies of every entry in $\mathbf{a}$ to ensure that larger entries are selected with higher probability. Only an integer number of copies is possible, so this step requires careful discretization of $\mathbf{a}$'s entries. Our method, in contrast, encodes weight information more efficiently through the threshold τ_i. Furthermore, to compute a sketch with m samples, WMH requires applying m independent hash functions to every index i where $\mathbf{a}$ is non-zero. This accounts for its run-time dependence on $O(Nm)$. Threshold Sampling uses one hash function, so runs in $O(N)$ time.

Another difference between Threshold Sampling and WMH is that, when run with parameter m, Threshold Sampling returns a sketch whose size is at most m *in expectation* (see Theorem 1). However, since entries of $\mathbf{a}$ are sampled independently, the actual size of the sketch will vary randomly around its expectation. In contrast, WMH allows the user to set an exact sketch size. This issue motivates our Priority Sampling method (Section 3), which is similar to Threshold Sampling but has a fixed sketch size.

Theoretical Guarantees. Our main theoretical result on Threshold Sampling is as follows:

Theorem 1. *For vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ and target sketch size m, let $\mathcal{S}(\mathbf{a}) = \{K_\mathbf{a}, V_\mathbf{a}, \tau_\mathbf{a}\}$ and $\mathcal{S}(\mathbf{b}) = \{K_\mathbf{b}, V_\mathbf{b}, \tau_\mathbf{b}\}$ be sketches returned by Algorithm 1. Let W be the inner product estimate returned by Algorithm 2 applied to these sketches. We have $\mathbb{E}[W] = \langle \mathbf{a}, \mathbf{b} \rangle$ and*

$$\mathrm{Var}[W] \leq \frac{2}{m} \max\left(\|\mathbf{a}_\mathcal{I}\|_2^2 \|\mathbf{b}\|_2^2, \|\mathbf{a}\|_2^2 \|\mathbf{b}_\mathcal{I}\|_2^2\right).$$

Moreover, let $|K_\mathbf{a}|$ and $|K_\mathbf{b}|$ be the number of index/values pairs stored in $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$. We have $\mathbb{E}[|K_\mathbf{a}|] \leq m$ and $\mathbb{E}[|K_\mathbf{b}|] \leq m$.

Above, $\mathbb{E}[\cdot]$ denotes expected value and $\mathrm{Var}[\cdot]$ denotes variance. Recall that $\mathcal{I} = \{i : \mathbf{a}[i] \neq 0 \text{ and } \mathbf{b}[i] \neq 0\}$ and $\mathbf{a}_\mathcal{I}$ and $\mathbf{b}_\mathcal{I}$ denote the vectors restricted to the indices in $\mathcal{I}$. Theorem 1 shows that the inner product estimate obtained using Threshold Sampling is correct in expectation and has bounded variance. Moreover, if the sketches are constructed with parameter m, the expected number of samples collected is always $\leq m$. Since the sketch needs to store two numbers for each sample (an index and a value), as well as the scalar value $\tau_\mathbf{a}$, the expected storage size is thus $O(m)$.

Given the expectation and variance bound in Theorem 1, we can apply Chebyshev's Inequality to obtain the following corollary:

Corollary 2. *For any given values of $\epsilon, \delta \in (0, 1)$ and vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, when run with target sketch m, Threshold Sampling returns*

[4] One computational disadvantage of sampling methods like Threshold Sampling in comparison to linear sketching is that they cannot be immediately implemented in a streaming setting where entries in $\mathbf{a}$ and $\mathbf{b}$ are updated incrementally; we need to know the magnitude of each entry in advance to perform sampling. We believe it is possible to resolve this issue using streaming ℓ_2 sampling algorithms (see e.g., [40] or [21]). We leave the details of how to do so most effectively to future work.

an inner product estimate W satisfying, with probability $1 - \delta$,

$$|W - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \sqrt{\frac{2/\delta}{m}} \max\left(\|\mathbf{a}_{\mathcal{I}}\|_2 \|\mathbf{b}\|_2, \|\mathbf{a}\|_2 \|\mathbf{b}_{\mathcal{I}}\|_2\right).$$

Setting $m = \frac{2/\delta}{\epsilon^2}$, *the error is* $\epsilon \cdot \max\left(\|\mathbf{a}_{\mathcal{I}}\|_2 \|\mathbf{b}\|_2, \|\mathbf{a}\|_2 \|\mathbf{b}_{\mathcal{I}}\|_2\right)$.

This corollary matches the asymptotic guarantee of WMH [6], improving on the bounds known for linear sketches like JL and CountSketch [5]. At the same time, as we show in Section 5, Threshold Sampling tends to perform better than WMH in practice. We believe there are a number of reasons for this, including the fact that Threshold Sampling selects vector entries without replacement, and the fact that the variance bound in Theorem 1 has a small constant factor of 2. We prove Theorem 1 below:

Proof of Theorem 1. Let $\mathcal{I}$ denote the set of all indices i for which $\mathbf{a}_i \neq 0$ and $\mathbf{b}i \neq 0$. For any $i \in \mathcal{I}$, let $\mathbb{1}_i$ denote the indicator random variable for the event that i is included in *both* $K_{\mathbf{a}}$ and $K_{\mathbf{b}}$. $\mathbb{1}_i = 1$ if this event occurs and 0 if it does not. Note that, for $i \neq j$, $\mathbb{1}_i$ is independent from $\mathbb{1}_j$, since the hash values $h(i)$ and $h(j)$ are drawn uniformly and independently from $[0, 1]$. Moreover, we claim that $\mathbb{1}_i$ is equal to 1 with probability:

$$p_i = \min\left(1, \frac{m \cdot \mathbf{a}_i^2}{\|\mathbf{a}\|_2^2}, \frac{m \cdot \mathbf{b}_i^2}{\|\mathbf{b}\|_2^2}\right) = \min(1, \tau_{\mathbf{a}} \cdot \mathbf{a}_i^2, \tau_{\mathbf{b}} \cdot \mathbf{b}_i^2). \tag{3}$$

To see why this is the case, assume without loss of generality that $\mathbf{a}_i^2 \leq \mathbf{b}_i^2$. Then, by examining Line 3 of Algorithm 1, we can see that i is included in $K_{\mathbf{a}}$ with probability $\min\left(1, m \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2\right)$. Moreover, if i is included in $K_{\mathbf{a}}$, it is *guaranteed* to be included in $K_{\mathbf{b}}$ since the threshold $m \cdot \mathbf{b}_i^2/\|\mathbf{b}\|_2^2$ is at least as large as $m \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2$. It follows that, when $\mathbf{a}_i^2 \leq \mathbf{b}_i^2$, we have that $p_i = \min\left(1, m \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2\right)$. The analysis is identical for the case $\mathbf{b}_i^2 < \mathbf{a}_i^2$, in which case $p_i = \min\left(1, m \cdot \mathbf{b}_i^2/\|\mathbf{b}\|_2^2\right)$. Combining the two cases establishes (3).

Let W be the estimate returned by Algorithm 2. We can write $W = \sum_{i \in \mathcal{I}} \mathbb{1}_i \cdot \frac{\mathbf{a}_i \mathbf{b}_i}{p_i}$, and applying linearity of expectation, we have:

$$\mathbb{E}[W] = \sum_{i \in \mathcal{I}} p_i \cdot \frac{\mathbf{a}_i \mathbf{b}_i}{p_i} = \sum_{i \in \mathcal{I}} \mathbf{a}_i \mathbf{b}_i = \langle \mathbf{a}, \mathbf{b} \rangle. \tag{4}$$

Next, since each term in the sum $W = \sum_{i \in \mathcal{I}} \mathbb{1}_i \cdot \frac{\mathbf{a}_i \mathbf{b}_i}{p_i}$ is independent,

$$\mathrm{Var}[W] = \sum_{i \in \mathcal{I}} \mathrm{Var}\left[\mathbb{1}_i \cdot \frac{\mathbf{a}_i \mathbf{b}_i}{p_i}\right] = \sum_{i \in \mathcal{I}} \frac{(\mathbf{a}_i \mathbf{b}_i)^2}{p_i^2} \mathrm{Var}[\mathbb{1}_i].$$

$\mathrm{Var}[\mathbb{1}_i] = p_i - p_i^2$, which is 0 when p_i equals 1. If $p_i \neq 1$, then $\mathrm{Var}[\mathbb{1}_i] \leq p_i = m \cdot \min\left(\mathbf{a}_i^2/\|\mathbf{a}\|_2^2, \mathbf{b}_i^2/\|\mathbf{b}\|_2^2\right)$. So we have:

$$\begin{aligned}
\mathrm{Var}[W] &\leq \sum_{i \in \mathcal{I}, p_i \neq 1} \|\mathbf{a}\|_2^2 \|\mathbf{b}\|_2^2 \frac{(\mathbf{a}_i^2/\|\mathbf{a}\|_2^2)(\mathbf{b}_i^2/\|\mathbf{b}\|_2^2)}{m \cdot \min(\mathbf{a}_i^2/\|\mathbf{a}\|_2^2, \mathbf{b}_i^2/\|\mathbf{b}\|_2^2)} \\
&= \sum_{i \in \mathcal{I}, p_i \neq 1} \|\mathbf{a}\|_2^2 \|\mathbf{b}\|_2^2 \frac{\max(\mathbf{a}_i^2/\|\mathbf{a}\|_2^2, \mathbf{b}_i^2/\|\mathbf{b}\|_2^2)}{m} \\
&\leq \frac{\|\mathbf{a}\|_2^2 \|\mathbf{b}\|_2^2}{m} \sum_{i \in \mathcal{I}} \frac{\mathbf{a}_i^2}{\|\mathbf{a}\|_2^2} + \frac{\mathbf{b}_i^2}{\|\mathbf{b}\|_2^2}.
\end{aligned}$$

Rearranging, this bound is equal to $\frac{1}{m}\left(\|\mathbf{b}\|_2^2 \|\mathbf{a}_{\mathcal{I}}\|_2^2 + \|\mathbf{a}\|_2^2 \|\mathbf{b}_{\mathcal{I}}\|_2^2\right)$, and we obtain our final bound on $\mathrm{Var}[W]$ by upper bounding the sum by 2x the maximum. Finally, we prove our claim on the expected sketch size. We have that $|K_{\mathbf{a}}| = \sum_{i=1}^n \mathbb{1}[i \in K_{\mathbf{a}}]$, where $\mathbb{1}[i \in K_{\mathbf{a}}]$ is an indicator random variable that is 1 if i is included in $K_{\mathbf{a}}$ and zero otherwise. By linearity of expectation, we have that:

$$\mathbb{E}\left[|K_{\mathbf{a}}|\right] = \sum_{i=1}^n \mathbb{E}\left[\mathbb{1}[i \in K_{\mathbf{a}}]\right] = \sum_{i=1}^n \min(1, m \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2) \leq m. \tag{5}$$

An identical analysis shows that $\mathbb{E}\left[|K_{\mathbf{b}}|\right] \leq m$, which completes the proof. In the extended version of this paper [26], we further prove that $|K_{\mathbf{a}}|$ and $|K_{\mathbf{b}}|$ are less than $m + O(\sqrt{m})$ with high probability. □

Practical Implementation. In Theorem 1, we show that the expected sketch size is *upper bounded* by m. As apparent from (5), it will be less than m if there are entries in $\mathbf{a}$ for which $\mathbf{a}_i^2/\|\mathbf{a}\|_2^2 > 1/m$. This is not ideal: we would like a sketch whose size is as close to our budget m as possible. Fortunately, Threshold Sampling can be modified so that the expected sketch size is *exactly m*. We simply use binary search to compute m' such that $\sum_{i=1}^n \min\left(1, m' \cdot \mathbf{a}_i^2/\|\mathbf{a}\|_2^2\right) = m$. Then, we replace m in Lines 3 and 6 of Algorithm 1 with m'. Doing so does not increase our estimator's variance. Further details are provided in the extended version of this paper [26].

3 PRIORITY SAMPLING

While a simple and effective method for inner product sketching, one limitation of Threshold Sampling is that the user cannot exactly control the size of the sketch $\mathcal{S}(\mathbf{a})$. We address this issue by analyzing an alternative algorithm based on Priority Sampling, a technique introduced in computer science by [29], and studied in statistics under the name "Sequential Poisson Sampling" [48].

Sketching. To motivate the method, observe from rearranging Lines 3 and 4 in Algorithm 1, that Threshold Sampling selects all entries from $\mathbf{a}$ for which $h(i)/a_i^2$ falls below a fixed "global threshold", $\tau_{\mathbf{a}} = m/\|\mathbf{a}\|_2^2$. There will be at most m such values in expectation, but there could be more or less depending on the randomness in h. Priority Sampling (Algorithm 3) removes this variability by simply selecting the m *smallest* values of $h(i)/a_i^2$. It then treats the $(m+1)^{\text{st}}$ smallest value as the global threshold $\tau_{\mathbf{a}}$.

Estimation. Given sketches $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$ computed using Priority Sampling, we can actually use the exact same estimator for $\langle \mathbf{a}, \mathbf{b} \rangle$ as Threshold Sampling (Algorithm 2). In particular,

$$W = \sum_{i \in K_{\mathbf{a}} \cap K_{\mathbf{b}}} \frac{\mathbf{a}_i \mathbf{b}_i}{\min(1, \mathbf{a}_i^2 \cdot \tau_{\mathbf{a}}, \mathbf{b}_i^2 \cdot \tau_{\mathbf{b}})} \tag{6}$$

(computed on Line 2 of Algorithm 2) remains an unbiased estimate for the inner product. However, analyzing the variance of the estimator is a lot trickier. Notably, we no longer have that the summation terms in (6) are independent; they all depend on the *same* random numbers $\tau_{\mathbf{a}}$ and $\tau_{\mathbf{b}}$, which were previously fixed quantities for Threshold Sampling. Moreover, bounding the variance of each term in the sum is complicated by the presence of random variables in the denominator. These issues arise in earlier applications of Priority Sampling, like subset-sum estimation [29]. For this problem, an optimal variance analysis proved elusive, until finally being given in a tour de force result by Szegedy [2, 55].

Theoretical Analysis. Building on a new analysis of Priority Sampling for subset sums [27], we are able to overcome these obstacles for inner product estimation as well, proving the following:

Algorithm 3 Priority Sampling

```
Input: Length $n$ vector $\mathbf{a}$, random seed $s$, target sketch size $m$.
Output: Sketch $\mathcal{S}(\mathbf{a}) = \{K_\mathbf{a}, V_\mathbf{a}, \tau_\mathbf{a}\}$, where $K_\mathbf{a}$ is a subset of indices
    from $\{1, \dots, n\}$ and $V_\mathbf{a}$ contains $\mathbf{a}_i$ for all $i \in K_\mathbf{a}$.
Use random seed $s$ to select a uniformly random hash function
    $h : \{1, ..., n\} \to [0, 1]$. Initialize $K_\mathbf{a}$ and $V_\mathbf{a}$ to be empty lists.
Compute rank $R_i = h(i)/\mathbf{a}_i^2$ for all $i$ such that $\mathbf{a}_i \neq 0$.
Set $\tau_\mathbf{a}$ equal to the $(m+1)^{\text{st}}$ smallest value $R_i$, or set $\tau_\mathbf{a} = \infty$ if
    $\mathbf{a}$ has less than $m+1$ non-zero values.
for $i$ such that $\mathbf{a}_i \neq 0$ do
    if $R_i < \tau_\mathbf{a}$ then
        Append $i$ to $K_\mathbf{a}$, append $\mathbf{a}_i$ to $V_\mathbf{a}$.
return $\mathcal{S}(\mathbf{a}) = \{K_\mathbf{a}, V_\mathbf{a}, \tau_\mathbf{a}\}$
```

Theorem 3. *For vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ and sketch size m, let $\mathcal{S}(\mathbf{a}) = \{K_\mathbf{a}, V_\mathbf{a}, \tau_\mathbf{a}\}$ and $\mathcal{S}(\mathbf{b}) = \{K_\mathbf{b}, V_\mathbf{b}, \tau_\mathbf{b}\}$ be sketches returned by Algorithm 3. Let W be the inner product estimate returned by Algorithm 2 applied to these sketches. We have that $\mathbb{E}[W] = \langle \mathbf{a}, \mathbf{b} \rangle$ and*

$$\operatorname{Var}[W] \leq \frac{2}{m-1} \max\left(\|\mathbf{a}_\mathcal{I}\|_2^2 \|\mathbf{b}\|_2^2, \|\mathbf{a}\|_2^2 \|\mathbf{b}_\mathcal{I}\|_2^2 \right)$$

Moreover, let $|K_\mathbf{a}|$ and $|K_\mathbf{b}|$ be the number of index/values pairs stored in $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$. We have $|K_\mathbf{a}| \leq m$ and $|K_\mathbf{b}| \leq m$, with equality in the typical case when $\mathbf{a}$ and $\mathbf{b}$ have at least m non-zero entries.

Theorem 3 almost exactly matches our Theorem 1 for Threshold Sampling, except that the leading constant on the variance is $\frac{2}{m-1}$ instead of $\frac{2}{m}$. Again, we can apply Chebyshev's inequality to conclude that if we set $m = \frac{2/\delta}{\epsilon^2} + 1$, then $|W - \langle \mathbf{a}, \mathbf{b} \rangle|$ is bounded by $\epsilon \cdot \max(\|\mathbf{a}_\mathcal{I}\|_2 \|\mathbf{b}\|_2, \|\mathbf{a}\|_2 \|\mathbf{b}_\mathcal{I}\|_2)$ with probability $\geq 1 - \delta$. The matching theoretical results align with experiments: as seen in Section 5, Priority Sampling performs almost identically to Threshold Sampling, albeit with the added benefit of a fixed sketch size bound.

Proof of Theorem 3. We start by introducing additional notation. Let $\mathcal{A} = \{i : \mathbf{a}_i \neq 0\}$ denote the set of indices where $\mathbf{a}$ is non-zero and let $\mathcal{B} = \{i : \mathbf{b}_i \neq 0\}$ denote the set of indices where $\mathbf{b}$ is non-zero. Recall that $\tau_\mathbf{a}$ as computed in Algorithm 3 is the $(m+1)^{\text{st}}$ smallest value of $h(i)/a_i^2$ over all $i \in \mathcal{A}$. For any $i \in \mathcal{A}$, let $\tau_\mathbf{a}^i$ denote the m^{th} smallest of $h(j)/a_j^2$ over all $j \in \mathcal{A} \setminus \{i\}$. If $\mathcal{A} \setminus \{i\}$ has fewer than m values, define $\tau_\mathbf{a}^i = \infty$. Define $\tau_\mathbf{b}^i$ analogously for all $i \in \mathcal{B}$. Let $\mathcal{T} = K_\mathbf{a} \cap K_\mathbf{b}$ be as in Algorithm 2. Later on we will use the easily checked fact that, for all $i \in \mathcal{T}$, $\tau_\mathbf{a}^i = \tau_\mathbf{a}$ and $\tau_\mathbf{b}^i = \tau_\mathbf{b}$.

The estimate W returned by Algorithm 2 can be rewritten as:

$$W = \sum_{i \in \mathcal{A} \cap \mathcal{B}} \hat{w}_i \quad \text{where} \quad \hat{w}_i = \begin{cases} \frac{\mathbf{a}_i \mathbf{b}_i}{\min(1, \mathbf{a}_i^2 \tau_\mathbf{a}, \mathbf{b}_i^2 \tau_\mathbf{b})} & i \in \mathcal{T} \\ 0 & i \notin \mathcal{T}. \end{cases} \tag{7}$$

From (7), we can see that, to prove $\mathbb{E}[W] = \langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i \in \mathcal{A} \cap \mathcal{B}} \mathbf{a}_i \mathbf{b}_i$, it suffices to prove that, for all $i \in \mathcal{A} \cap \mathcal{B}$, $\mathbb{E}[\hat{w}_i] = \mathbf{a}_i \mathbf{b}_i$. To establish this equality, first observe that for i to be in $\mathcal{T}$, it must be that both $h(i)/\mathbf{a}_i^2$ and $h(i)/\mathbf{b}_i^2$ are among the m^{th} smallest ranks computed when sketching $\mathbf{a}$ and $\mathbf{b}$, respectively. In other words, it must be that $h(i)/\mathbf{a}_i^2 < \tau_\mathbf{a}^i$ and $h(i)/\mathbf{b}_i^2 < \tau_\mathbf{b}^i$. So, conditioning on $\tau_\mathbf{a}^i$ and $\tau_\mathbf{b}^i$,

$$\begin{aligned} \Pr\left[i \in \mathcal{T} \mid \tau_\mathbf{a}^i, \tau_\mathbf{b}^i\right] &= \Pr\left[h(i)/\mathbf{a}_i^2 < \tau_\mathbf{a}^i \cap h(i)/\mathbf{b}_i^2 < \tau_\mathbf{b}^i\right] \\ &= \min(1, \mathbf{a}_i^2 \tau_\mathbf{a}^i, \mathbf{a}_i^2 \tau_\mathbf{b}^i). \end{aligned}$$

Combined with the fact discussed earlier that, conditioned on $i \in \mathcal{T}$, $\tau_\mathbf{a} = \tau_\mathbf{a}^i$ and $\tau_\mathbf{b} = \tau_\mathbf{b}^i$, we have:

$$\mathbb{E}[\hat{w}_i] = \mathop{\mathbb{E}}_{\tau_\mathbf{a}^i, \tau_\mathbf{b}^i} \left[\frac{\mathbf{a}_i \mathbf{b}_i}{\min(1, \mathbf{a}_i^2 \tau_\mathbf{a}, \mathbf{a}_i^2 \tau_\mathbf{b})} \min(1, \mathbf{a}_i^2 \tau_\mathbf{a}^i, \mathbf{a}_i^2 \tau_\mathbf{b}^i) \right] = \mathbf{a}_i \mathbf{b}_i.$$

As desired, $\mathbb{E}[W] = \langle \mathbf{a}, \mathbf{b} \rangle$ follows by linearity of expectation.

Next, we turn our attention to bounding the variance of W. As discussed, this is complicated by the fact that $\hat{w}_i$ and $\hat{w}_j$ are non-independent. However, it is possible to show that the random variables are *pairwise uncorrelated*, which will allow us to apply linearity of variance to the sum in (7). I.e., we want to show that, for all i, j, $\mathbb{E}[\hat{w}_i \hat{w}_j] = \mathbb{E}[\hat{w}_i]\,\mathbb{E}[\hat{w}_j]$. For any $i, j \in \mathcal{A}$ define $\tau_\mathbf{a}^{i,j}$ to equal the $(m-1)^{\text{st}}$ smallest of $h(k)/a_k^2$ over all $k \in \mathcal{A} \setminus \{i, j\}$, or ∞ if there are not $m-1$ values in $\mathcal{A} \setminus \{i, j\}$. Define $\tau_\mathbf{b}^{i,j}$ analogously for $i, j \in \mathcal{B}$. As in our expression for $\Pr[i \in \mathcal{T}]$, it can be seen that $\Pr[i, j \in \mathcal{T} \mid \tau_\mathbf{a}^{i,j}, \tau_\mathbf{b}^{i,j}] = \min(1, \mathbf{a}_i^2 \tau_\mathbf{a}^{i,j}, \mathbf{b}_i^2 \tau_\mathbf{b}^{i,j}) \cdot \min(1, \mathbf{a}_j^2 \tau_\mathbf{a}^{i,j}, \mathbf{b}_j^2 \tau_\mathbf{b}^{i,j})$. Furthermore, conditioned on $i, j \in \mathcal{T}$, $\tau_\mathbf{a}^{i,j} = \tau_\mathbf{a}$ and $\tau_\mathbf{b}^{i,j} = \tau_\mathbf{b}$. So,

$$\begin{aligned} \mathbb{E}[\hat{w}_i \hat{w}_j] = \mathop{\mathbb{E}}_{\tau_\mathbf{a}^{i,j}, \tau_\mathbf{b}^{i,j}} &\left[\frac{\mathbf{a}_i \mathbf{b}_i}{\min(1, \mathbf{a}_i^2 \tau_\mathbf{a}^{i,j}, \mathbf{b}_i^2 \tau_\mathbf{b}^{i,j})} \frac{\mathbf{a}_j \mathbf{b}_j}{\min(1, \mathbf{a}_j^2 \tau_\mathbf{a}^{i,j}, \mathbf{b}_j^2 \tau_\mathbf{b}^{i,j})} \cdot \ldots \right. \\ &\left. \ldots \Pr\left[i, j \in \mathcal{T} \mid \tau_\mathbf{a}^{i,j}, \tau_\mathbf{b}^{i,j}\right] \right] = \mathbf{a}_i \mathbf{b}_i \mathbf{a}_j \mathbf{b}_j = \mathbb{E}[\hat{w}_i]\,\mathbb{E}[\hat{w}_j], \end{aligned}$$

as desired. Since $\mathbb{E}[\hat{w}_i \hat{w}_j] = \mathbb{E}[\hat{w}_i]\,\mathbb{E}[\hat{w}_j]$ for all i, j we can apply linearity of variance to conclude that $\operatorname{Var}[W] = \sum_{i \in \mathcal{A} \cap \mathcal{B}} \operatorname{Var}[\hat{w}_i]$.

So, it suffices to establish individual bounds on $\operatorname{Var}[\hat{w}_i]$ for $i \in \mathcal{A} \cap \mathcal{B}$. To do so, first observe that, conditioned on $\tau_\mathbf{a}^i$ and $\tau_\mathbf{b}^i$,

$$\begin{aligned} \mathbb{E}\left[\hat{w}_i^2 \mid \tau_\mathbf{a}^i, \tau_\mathbf{b}^i\right] &= \left(\frac{\mathbf{a}_i \mathbf{b}_i}{\min(1, \mathbf{a}_i^2 \tau_\mathbf{a}^i, \mathbf{b}_i^2 \tau_\mathbf{b}^i)} \right)^2 \cdot \Pr\left[i \in \mathcal{T} \mid \tau_\mathbf{a}^i, \tau_\mathbf{b}^i\right] \\ &= \frac{\mathbf{a}_i^2 \mathbf{b}_i^2}{\min(1, \mathbf{a}_i^2 \tau_\mathbf{a}^i, \mathbf{b}_i^2 \tau_\mathbf{b}^i)} = \mathbf{a}_i^2 \mathbf{b}_i^2 \max\left(1, \frac{1}{\mathbf{a}_i^2 \tau_\mathbf{a}^i}, \frac{1}{\mathbf{b}_i^2 \tau_\mathbf{b}^i}\right). \end{aligned}$$

We can thus rewrite $\operatorname{Var}[\hat{w}_i] = \mathbb{E}\left[\hat{w}_i^2\right] - \mathbb{E}[\hat{w}_i]^2 = \mathbb{E}\left[\hat{w}_i^2\right] - \mathbf{a}_i^2 \mathbf{b}_i^2$:

$$\begin{aligned} \operatorname{Var}[\hat{w}_i] &= \mathbf{a}_i^2 \mathbf{b}_i^2\, \mathbb{E}\left[\max\left(1, \frac{1}{\mathbf{a}_i^2 \tau_\mathbf{a}^i}, \frac{1}{\mathbf{b}_i^2 \tau_\mathbf{b}^i}\right) \right] - \mathbf{a}_i^2 \mathbf{b}_i^2 \\ &= \mathbf{a}_i^2 \mathbf{b}_i^2\, \mathbb{E}\left[\max\left(0, \frac{1}{\mathbf{a}_i^2 \tau_\mathbf{a}^i} - 1, \frac{1}{\mathbf{b}_i^2 \tau_\mathbf{b}^i} - 1\right) \right] \\ &\leq \mathbf{a}_i^2 \mathbf{b}_i^2\, \mathbb{E}\left[\frac{1}{\mathbf{a}_i^2 \tau_\mathbf{a}^i} + \frac{1}{\mathbf{b}_i^2 \tau_\mathbf{b}^i} \right] = \mathbf{a}_i^2\, \mathbb{E}\left[\frac{1}{\tau_\mathbf{b}^i}\right] + \mathbf{b}_i^2\, \mathbb{E}\left[\frac{1}{\tau_\mathbf{a}^i}\right]. \end{aligned}$$

So, we have reduced the problem to bounding the expected inverse of $\tau_\mathbf{a}^i$ and $\tau_\mathbf{b}^i$. Doing so is not straightforward: these are complex random variables that depend on all entries in $\mathbf{a}$ and $\mathbf{b}$, respectively. However, it was recently shown in [27] (Claim 5) that $\mathbb{E}[1/\tau_\mathbf{a}^i] \leq \|\mathbf{a}\|_2^2/(m-1)$ and $\mathbb{E}[1/\tau_\mathbf{b}^i] \leq \|\mathbf{b}\|_2^2/(m-1)$. Finally, we have:

$$\begin{aligned} \operatorname{Var}[W] = \sum_{i \in A \cap B} \operatorname{Var}[\hat{w}_i] &\leq \sum_{i \in A \cap B} \mathbf{a}_i^2\, \mathbb{E}\left[1/\tau_\mathbf{b}^i\right] + \mathbf{b}_i^2\, \mathbb{E}\left[1/\tau_\mathbf{a}^i\right] \\ &\leq \sum_{i \in A \cap B} \mathbf{a}_i^2 \frac{\|\mathbf{b}\|^2}{m-1} + \mathbf{b}_i^2 \frac{\|\mathbf{a}\|^2}{m-1} \\ &= \frac{1}{m-1} \left(\|\mathbf{a}_\mathcal{I}\|_2^2 \|\mathbf{b}\|_2^2 + \|\mathbf{a}\|_2^2 \|\mathbf{b}_\mathcal{I}\|_2^2 \right). \end{aligned}$$

Noting that for any c, d, $c+d \leq 2\max(c, d)$ completes the proof. □

4 JOIN-CORRELATION ESTIMATION

In addition to our theoretical results, we perform an empirical evaluation of Threshold and Priority Sampling for inner product sketching. One of our main motivating applications is *join-correlation estimation* [32, 52]. This problem has previously been addressed using (unweighted) consistent sampling methods, like the KMV sketch [52, 53]. In this section, we show how it can be solved using *any* inner product sketching algorithm in a black-box way, expanding the toolkit of methods that can be applied to the task.

Problem Statement. The join-correlation problem consists of computing the Pearson's correlation coefficient between two data columns that originally reside in different data tables. Specifically, we are interested in the correlation between values that would appear in the columns *after* performing an (inner) join on the tables, i.e., values for which the same key appears in both tables. We call this quantity the *post-join correlation*, or simply the *join-correlation.* As a concrete illustration, consider the example tables in Figure 2(a). The goal of join-correlation estimation is to approximate the correlation $\rho_{\mathbf{x},\mathbf{y}}$ between the vectors $\mathbf{x}$ and $\mathbf{y}$ from $\mathcal{T}_{A\bowtie B}$.

The join-correlation problem arises in dataset search applications, where the goal is to discover new data to augment a query dataset, e.g., to improve predictive models [13, 38, 45]. In such applications, we typically want to estimate join-correlation for columns in a query table and those in a large collection of other data tables. Accordingly, the brute-force approach that explicitly joins tables and computes the correlation between attributes is infeasible.

Prior work proposes to use sketching as an efficient alternative. The idea is to pre-process (i.e., sketch) the collection of tables in advance, so that join-correlation between columns in any two tables $\mathcal{T}_A$ and $\mathcal{T}_B$ can be evaluated *without explicitly materializing the join $A \bowtie B$*. Specifically, Santos et al. [52] propose an extension of KMV sketches that uniformly samples entries from each table, and then uses the join between the sketches to estimate correlation. Unfortunately, just like inner product estimation, this approach can suffer when $\mathcal{T}_A$ and $\mathcal{T}_B$ contain entries with widely varying magnitude: larger entries often contribute more to the correlation, but are not selected with higher probability by the KMV sketch.

Join-Correlation via Inner Product Sketching. We show an alternative approach for attacking the join-correlation problem by reducing it to inner product estimation. The reduction allows us to take advantage of sketches like WMH, Threshold Sampling, and Priority Sampling, which naturally make use of weighted sampling.

Referring again to Figure 2(a), consider the vectors $\mathbf{x}$ and $\mathbf{y}$ from $\mathcal{T}_{A\bowtie B}$. Let $\bar{x}$ (resp. $\bar{y}$) denote the mean of $\mathbf{x}$ (resp. $\mathbf{y}$), n denote the length of the vectors (number of rows in $\mathcal{T}_{A\bowtie B}$), $\Sigma_\mathbf{x}$ (resp. $\Sigma_\mathbf{y}$) denote the summation of all values in $\mathbf{x}$ (resp. $\mathbf{y}$), and $\Sigma_{\mathbf{x}^2}$ (resp. $\Sigma_{\mathbf{y}^2}$) denote the summation of all squared values of $\mathbf{x}$ (resp. $\mathbf{y}$). It can be verified that correlation coefficient between $\mathbf{x}$ and $\mathbf{y}$ can be rewritten as:

$$\rho_{\mathbf{x},\mathbf{y}} = \frac{\langle \mathbf{x}-\bar{x}, \mathbf{y}-\bar{y}\rangle}{\|\mathbf{x}-\bar{x}\|_2\|\mathbf{y}-\bar{y}\|_2} = \frac{n\langle \mathbf{x},\mathbf{y}\rangle - \Sigma_\mathbf{x}\Sigma_\mathbf{y}}{\sqrt{n\Sigma_{\mathbf{x}^2}-\Sigma_\mathbf{x}^2}\sqrt{n\Sigma_{\mathbf{y}^2}-\Sigma_\mathbf{y}^2}}. \tag{8}$$

$\mathcal{T}_A$

$\mathbf{k_a}$	**a**
3	2.5
6	2.3
8	4
11	0.5
13	3
16	-3.7

$\mathcal{T}_B$

$\mathbf{k_b}$	**b**
3	-3.1
7	0.4
8	-4.2
10	1.5
11	1
13	-2.6
14	-5.9

$\mathcal{T}_{A\bowtie B}$

$\mathbf{k}_{a\bowtie b}$	**x**	**y**
3	2.5	-3.1
8	4	-4.2
11	0.5	1
13	3	-2.6

(a) The table $\mathcal{T}_{A\bowtie B}$ is the output of a join between tables $\mathcal{T}_A$ and $\mathcal{T}_B$. The goal of join-correlation estimation is to approximate the Pearson's correlation between the second two columns in $\mathcal{T}_{A\bowtie B}$.

index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
a	0	0	2.5	0	0	2.3	0	4	0	0	0.5	0	3	0	0	-3.7
$\mathbf{a}^2$	0	0	6.25	0	0	5.29	0	16	0	0	.25	0	9	0	0	13.69
$\mathbf{1_a}$	0	0	1	0	0	1	0	1	0	0	1	0	1	0	0	1
b	0	0	-3.1	0	0	0	0.4	-4.2	0	1.5	1	0	-2.6	-5.9	0	0
$\mathbf{b}^2$	0	0	9.61	0	0	0	.16	17.64	0	2.25	1	0	6.76	34.81	0	0
$\mathbf{1_b}$	0	0	1	0	0	0	1	1	0	1	1	0	1	1	0	0

(b) We define six sparse vectors a, a^2, 1$_a$, b, b^2, and 1$_b$ that encode the information in $\mathcal{T}_A$ with $\mathcal{T}_B$. In Eq. (9), we show how to express the join-correlation as a combination of inner products involving these vectors, which can be estimated with a sketching method.

Figure 2: Join-Correlation via inner product sketching.

Our observation is that all of the values in Eq. (8) can be computed using only inner product operations over vectors derived from tables $\mathcal{T}_A$ and $\mathcal{T}_B$ independently. The vectors are shown in Figure 2(b): vectors $\mathbf{a}$ and $\mathbf{b}$ contain the values, with $\mathbf{a}_i$ (resp. $\mathbf{b}_i$) set to zero if key i was not present in table $\mathcal{T}_A$ (resp. table $\mathcal{T}_B$). Vectors $\mathbf{1_a}$ and $\mathbf{1_b}$ are indicator vectors for the corresponding join keys in each table. Finally, $\mathbf{a}^2$ and $\mathbf{b}^2$ are equal to $\mathbf{a}$ and $\mathbf{b}$ with an entrywise square applied. Using these vectors, we can compute all components of the correlation formula as inner products:

$$n = \langle \mathbf{1_a}, \mathbf{1_b}\rangle, \qquad \Sigma_\mathbf{x} = \langle \mathbf{a}, \mathbf{1_b}\rangle, \qquad \Sigma_\mathbf{y} = \langle \mathbf{1_a}, \mathbf{b}\rangle,$$
$$\langle \mathbf{x}, \mathbf{y}\rangle = \langle \mathbf{a}, \mathbf{b}\rangle, \qquad \Sigma_{\mathbf{x}^2} = \langle \mathbf{a}^2, \mathbf{1_b}\rangle, \qquad \Sigma_{\mathbf{y}^2} = \langle \mathbf{1_a}, \mathbf{b}^2\rangle.$$

In particular, we can rewrite $\rho_{\mathbf{x},\mathbf{y}}$ equivalently as:

$$\frac{\langle \mathbf{a}, \mathbf{b}\rangle\langle \mathbf{1_a}, \mathbf{1_b}\rangle - \langle \mathbf{a}, \mathbf{1_b}\rangle\langle \mathbf{1_a}, \mathbf{b}\rangle}{\sqrt{\left(\langle \mathbf{1_a}, \mathbf{1_b}\rangle\langle \mathbf{a}^2, \mathbf{1_b}\rangle - \langle \mathbf{a}, \mathbf{1_b}\rangle^2\right)\left(\langle \mathbf{1_a}, \mathbf{1_b}\rangle\langle \mathbf{b}^2, \mathbf{1_a}\rangle - \langle \mathbf{b}, \mathbf{1_a}\rangle^2\right)}}. \tag{9}$$

Given this formula, we can use any inner product sketching method to approximate join-correlation. In particular, given $\mathcal{T}_A$, we compute three separate sketches, one for each of $\mathbf{a}$, $\mathbf{a}^2$, $\mathbf{1_a}$. When combined with sketches for $\mathbf{b}$, $\mathbf{b}^2$, $\mathbf{1_b}$, we can estimate all of the inner products in (9) separately, and combine them to obtain an estimate for $\rho_{\mathbf{x},\mathbf{y}}$.

For data discovery, the vectors described above are often extremely sparse with limited overlap between non-zero entries. Therefore, they are amenable to the sampling-based sketches studied in this paper, and benefit from our improvements over (1). In particular, the length of $\mathbf{a}$, $\mathbf{a}^2$, and $\mathbf{1_a}$ equals the total universe of possible keys, while the number of non-zeros in these vectors equals the number of keys in $\mathcal{T}_A$. The overlap between the non-zeros in $\mathbf{a}$, $\mathbf{a}^2$, and $\mathbf{1_a}$, and those in $\mathbf{b}$, $\mathbf{b}^2$, and $\mathbf{1_b}$ is equal to the number of keys in common between $\mathcal{T}_A$ and $\mathcal{T}_B$, which can be very small. As an example, consider a data augmentation task where were wish to join a query data table, $\mathcal{T}_A$, with keys that are addresses in a single neighborhood to a statewide database of addresses in $\mathcal{T}_B$.

Optimization for Sampling-Based Sketches. In Section 5, we use the approach above to estimate correlation using linear sketching methods like CountSketch and JL. Given sketch size budget m, we allocate $m/3$ space to sketching each of the three vectors $\mathbf{a}$, $\mathbf{a}^2$, and $\mathbf{1_a}$. Our final join-correlation sketch is then the concatenation of the equally sized sketches $\mathcal{S}(\mathbf{a})$, $\mathcal{S}(\mathbf{a}^2)$, and $\mathcal{S}(\mathbf{1_a})$. We take roughly the same approach for Threshold and Priority Sampling. However, in a sampling-based sketch, if we select index i when sketching *any* of the three vectors $\mathbf{1_a}$, $\mathbf{a}$, and $\mathbf{a}^2$, then we might as well use the index in estimating inner products involving *all* three. In particular, by storing the single key/value pair $(i, \mathbf{a}_i)$, we can compute the information $(i, 1)$, $(i, \mathbf{a}_i)$, and $(i, \mathbf{a}_i^2)$ needed to estimate all inner products. We take advantage of this fact to squeeze additional information out of our sketches. Details of the resulting optimized approach are included in the extended version of this paper [26].

5 EXPERIMENTS

Baselines. We assess the performance of our methods by comparing them to representative baselines, all of which were implemented in Python. We include both sampling and linear sketching methods for inner product estimation:

- ***Johnson-Lindenstrauss Projection (JL):*** For this *linear sketch*, we use a dense random matrix $\mathbf{\Pi}$ with scaled ±1 entries, which is equivalent to the AMS sketch [1, 4].
- ***CountSketch (CS):*** The classic *linear sketch* introduced in [11], and also studied under the name Fast-AGMS sketch in [23]. We use one repetition of the sketch.[5]
- ***Weighted MinHash Sampling (MH-weighted):*** The method described in [6], which is the first sketch with tighter theoretical bounds than linear sketching for inner product estimation.
- ***MinHash Sampling (MH):*** Also described in [6], MH is similar to Weighted MinHash, but indices are sampled uniformly at random from $\mathbf{a}$, not with probability proportional to $\mathbf{a}_i^2$.
- ***Uniform Priority Sampling (PS-uniform):*** The same as our Priority Sampling method, but the rank of index i in Algorithm 3 is chosen without taking the squared magnitude $\mathbf{a}_i^2$ into account, so indices are sampled uniformly. This method is equivalent to the KMV-based inner product sketch implemented in [6].
- ***Uniform Threshold Sampling (TS-uniform):*** The same as our Threshold Sampling method, but $\mathbf{a}_i^2$ is not taken into account when computing τ_i, so indices are sampled uniformly.

To distinguish from the uniform sampling versions, our proposed Threshold and Priority Sampling methods are called **TS-weighted** and **PS-weighted** in the remainder of the section. In addition to the baselines above, we implemented and performed initial experiments using the End-Biased sampling method from [33], which is equivalent to Threshold Sampling (Algorithm 1), but with probability proportional to $|\mathbf{a}_i|/\|\mathbf{a}\|_1$. More details on how to implement this method, as well as TS-uniform and PS-uniform are included in the extended version of this paper [26]. As shown in Section 5.1, End-Biased sampling performed slightly worse than our version of Threshold Sampling, which also enjoys stronger theoretical guarantees. So, we excluded End-Biased sampling from the majority of our experiments for conciseness and plot clarity. We also note that there are other versions of linear sketching designed to speed up computation time in comparison to the classic JL/AMS sketch [1, 50]. We focus on CountSketch/Fast-AGMS because it is one of the most widely studied of these methods, and runs in $O(n)$ time with a small constant factor. As such, it offers a challenging baseline for our sampling methods in terms of computational efficiency.

Storage Size. For linear sketches, we store the output of the matrix multiplication $\mathbf{\Pi a}$ as 64-bit doubles. For sampling-based sketches, both samples (64-bit doubles) and hash values (32-bit ints) need to be stored. As a result, a sampling sketch with m samples takes $1.5x$ as much space as a linear sketch with m entries. In our experiments, *storage size* denotes the total number of bits in the sketch divided by 64, i.e., the total number of doubles that the sketch equates to. Storage size is fixed for all methods except Threshold Sampling, for which we report the expected storage size. We note that there are variants of linear sketching that further compress $\mathbf{\Pi a}$ by thresholding or rounding its entries, e.g., SimHash [10] and quantized JL methods [39]. While an interesting topic for future study, we do not evaluate these methods because quantization can be used to reduce the sketch size of *all methods*. For instance, for sampling-based sketches, we do not need to store full 64-bit doubles. Evaluating optimal quantization strategies is beyond the scope of this work.

Estimation Error. To make it easier to compare across different datasets, when estimating inner products, we define the following error measure: the absolute difference between ground truth inner product $\langle \mathbf{a}, \mathbf{b} \rangle$ and the estimate, scaled by $1/\|\mathbf{a}\|_2\|\mathbf{b}\|_2$. Given that most methods tested (except the uniform sampling methods) achieve an error guarantee at least as good as Eq. (1), this scaling roughly ensures that reported errors lie between 0 and 1.

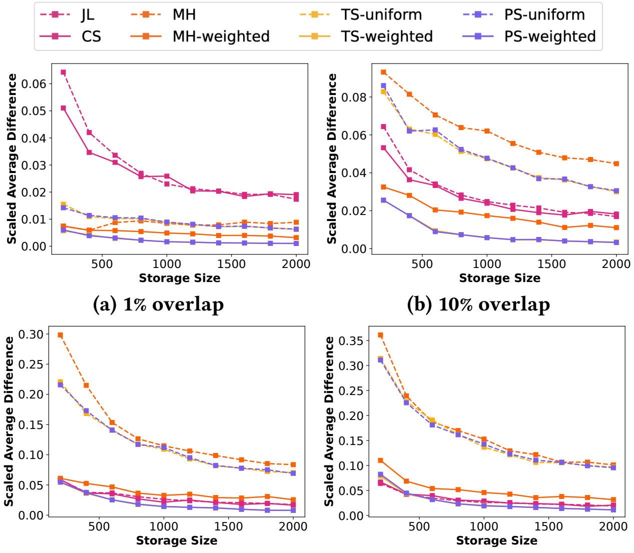
5.1 Estimation Accuracy for Synthetic Data

Synthetic Data. We ran experiments on synthetic data to validate the performance of our methods in a controlled setting. To contrast the behavior of linear sketching and weighted sampling methods like MH-weighted, TS-weighted, and PS-weighted, we generate vector pairs $\mathbf{a}, \mathbf{b}$ with varying amounts of overlap, $\mathcal{I}$, between their non-zero entries (1% to 100%). This allows us to verify our theoretical results: when $|\mathcal{I}|$ is large, we expect linear sketching and sampling to perform similarly since the linear sketching error bound of $\epsilon\|\mathbf{a}\|_2\|\mathbf{b}\|_2$ is closer to our bound of $\epsilon \cdot \max(\|\mathbf{a}\|_2\|\mathbf{b}_{\mathcal{I}}\|_2, \|\mathbf{a}\|_2\|\mathbf{b}_{\mathcal{I}}\|_2)$. When $|\mathcal{I}|$ is small, we expect a bigger difference.

We generate 100 pairs of synthetic vectors, each with 100,000 entries, 20,000 of which are non-zero. The locations of non-zero entries are randomly selected with a specific overlap $\mathcal{I}$, and their values are uniformly drawn from the interval $[-1, 1]$. Then, 2% of entries are chosen randomly as outliers. We include outliers to differentiate the performance of weighted sampling methods from their uniform counterparts (MH, TS-uniform and PS-uniform). If all entries have similar magnitude, weighted and uniform sampling are essentially the same. Outliers are chosen to be uniform random numbers between 0 and 10, which are fairly moderate values. For datasets with even larger outliers, we expect an even more pronounced difference between weighted and unweighted sampling.

5.1.1 Inner Product Estimation. Figure 3 shows the scaled average difference between the actual and estimated inner product for the different techniques. The plot is consistent with our theoretical findings: TS-weighted and PS-weighted are more accurate than all other methods for all levels of overlap. They consistently outperform the prior state-of-the-art sampling sketch, MH-weighted. For very low

[5] While prior work suggests partitioning the sketch and taking the median of multiple independent estimators [42], we found that doing so slightly decreased accuracy.



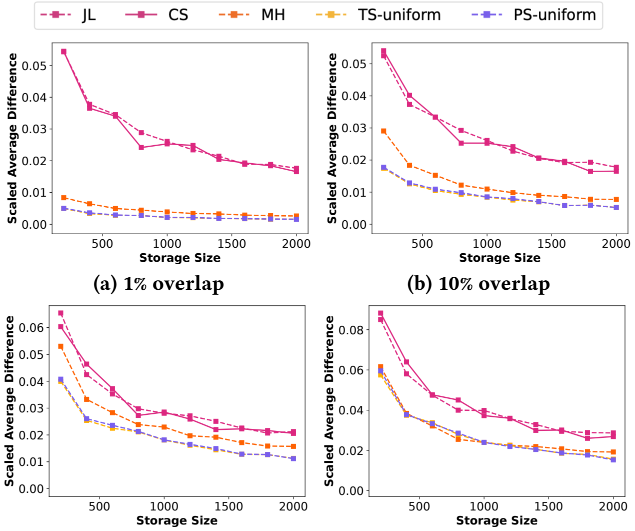
(a) 1% overlap

(b) 10% overlap

(c) 50% overlap

(d) 100% overlap

Figure 3: Inner product estimation for real-valued synthetic data. The lines for PS-uniform and TS-uniform overlap, as do the lines for our PS-weighted and TS-weighted methods. As predicted by our theoretical results, PS-weighted and TS-weighted consistently outperform all other baselines.



(a) 1% overlap

(b) 10% overlap

(c) 50% overlap

(d) 100% overlap

Figure 4: Inner product estimation for synthetic binary data. Weighted sampling methods are excluded since they are equivalent to their unweighted counterparts for binary vectors. Our PS-uniform and TS-uniform methods outperform both linear sketches and MH for computing inner products.

overlap, even unweighted sampling methods (MH, TS-uniform, and PS-uniform) outperform linear sketches (JL, CS), but this advantage decreases as overlap increases. Note that when overlap is above 50%, the performance of linear sketching is comparable to MH-weighted. However, our proposed methods, TS-weighted and PS-weighted, continue to outperform linear sketching, even in this regime.

5.1.2 Binary Inner Product Estimation. We also evaluate inner product estimation for binary $\{0, 1\}$ vectors, which can be applied to problems like join size estimation for tables with unique keys [24] and set intersection estimation. Set intersection has been studied e.g., for applications like document similarity estimation [8, 44, 49]. We use the same synthetic data as before, except that all non-zero entries are set to 1. Results are presented in Figure 4. Weighted sampling methods (WMH, TS-weighted, and PS-weighted) are not included because they are exactly equivalent to the unweighted methods for binary vectors. All of the sampling methods clearly outperform linear sketching, and the gap is most significant when the overlap is small, as predicted by our theoretical results.

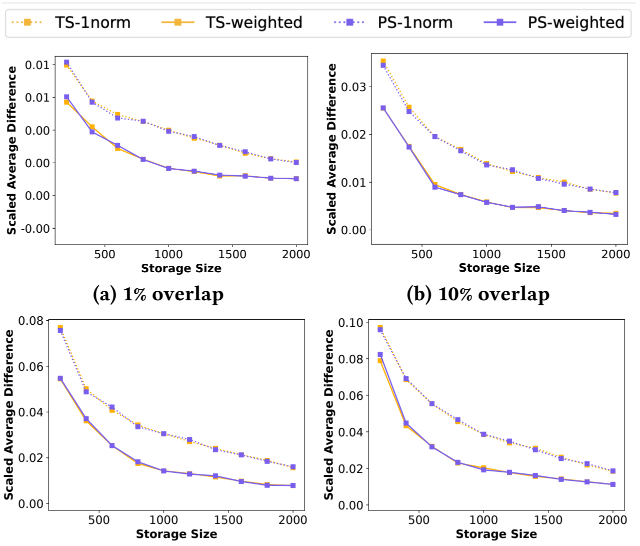
5.1.3 Join-Correlation Estimation. As discussed in Section 4, post-join correlation estimation can be cast as an inner product estimation problem involving three vectors derived from a data column, which we denote $\mathbf{a}$, $\mathbf{a}^2$, and $\mathbf{1_a}$. We do not explicitly construct synthetic database columns but instead generate vectors $\mathbf{a}$ and $\mathbf{b}$ as before, and derive $\mathbf{a}^2$, $\mathbf{1_a}$, $\mathbf{b}^2$, and $\mathbf{1_b}$ based on them. We set the overlap between vector pairs to 10% and control the correlation between the vectors (which are generated randomly) using a standard regression-based method for adjusting correlation [36]. For the linear sketching methods, we split the storage size evenly among the sketches for all three vectors and estimate correlation as discussed in Section 4. For the uniform sampling methods (MH, TS-uniform, and PS-uniform), we instead follow the approach from [52], computing a single sketch for each of $\mathbf{a}$ and $\mathbf{b}$ and then estimating the empirical correlation of the sampled entries. For TS-weighted and PS-weighted, we use our new method described in Section 4.

As Figure 6 shows, MH, TS-uniform, and PS-uniform perform well despite the lack of weighted sampling. This is consistent with observations in prior work [52]. Even without weighting, these sketches benefit from the advantage of data sparsity. Nonetheless, our TS-weighted and PS-weighted outperform all other approaches in terms of accuracy vs. sketch size. We use the optimized variants of these methods discussed in Section 4.

5.1.4 Comparison to End-Biased Sampling. As mentioned, we also considered adding End-Biased Sampling [33] as a baseline. This method is equivalent to Threshold Sampling, but samples vector entries with probability proportional to their magnitude, normalized by the vector ℓ_1 norm. We refer to this as ℓ_1 sampling to highlight the difference between our methods, which sample based on *squared magnitude* normalized by the ℓ_2 norm. A variant of Priority Sampling can also be implemented using ℓ_1 sampling. We found that End-Biased Sampling performed similarly, but never significantly better than, Threshold Sampling. This is shown in Figure 5, which uses the same experimental setting as Figure 3.

5.2 Runtime Performance

As discussed in Section 1, it is also important to consider the time required to compute inner product sketches. Threshold and Priority Sampling compute a sketch of size m in time $O(N)$ and $O(N \log m)$, respectively, for a vector with N non-zero entries, matching the complexity of the fastest methods like CountSketch, and improving on the $O(Nm)$ complexity of WMH [6]. To see how this theoretical



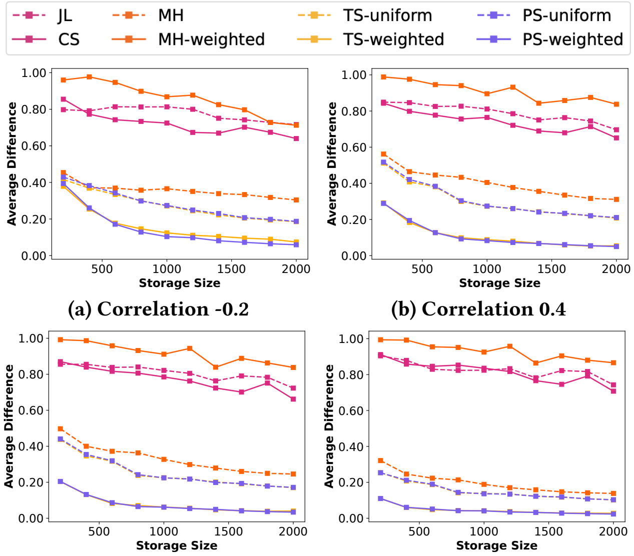
(a) 1% overlap

(b) 10% overlap

(c) 50% overlap

(d) 100% overlap

Figure 5: Comparison of End-Biased Sampling (TS-1norm) and its Priority Sampling counterpart (PS-1norm) against our TS-weighted and PS-weighted methods.



(a) Correlation -0.2

(b) Correlation 0.4

(c) Correlation -0.6

(d) Correlation 0.8

Figure 6: Join-Correlation estimation for synthetic data. The lines for PS-weighted and TS-weighted overlap, as do the lines for our PS-uniform and TS-uniform methods, which outperform all other baselines.

improvement translates to practice, we assess the run-time efficiency of these methods using high-dimensional synthetic vectors with 250,000 entries, 50,000 of which are non-zero. As above, non-zero entries are random values in $[-1, 1]$, except 10% are chosen as outliers. However, for all methods considered, the precise values of entries should have little to no impact on run-time.

In addition to our standard baselines, to evaluate runtime, we considered more efficient implementations of the WMH algorithm from [6]. That paper uses a sampling method studied in [46] and [37] that

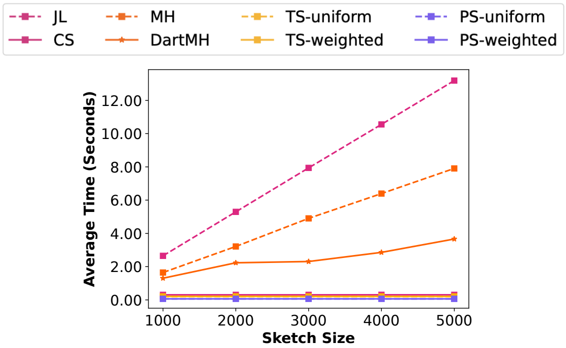


Figure 7: Sketch construction time. We omit MH-weighted since its slow time would make it difficult to see the other lines. We see a clear linear dependence on the sketch size for JL and MH, and a milder dependence for DartMH. The run-time of CountSketch, Threshold Sampling, and Priority Sampling does not noticeably scale with the sketch size.

1) requires $O(Nm)$ hash evaluations, and 2) requires an expensive discretization step. Several papers attempt to eliminate these limitations [31, 54]. We implement a recent, improved method called DartMinHash (DartMH) from [15], which runs in $O(N + m \log m)$. Details on the method are discussed in the extended version [26].

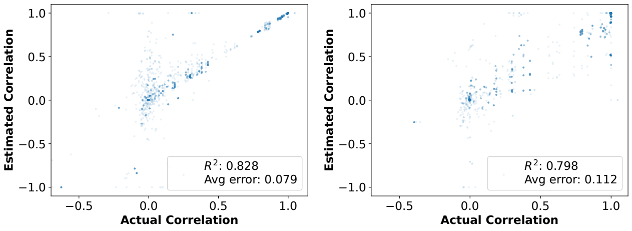
The times required by different methods to create sketches of varying sizes are shown in Fig. 7. As expected, both our weighted and unweighted Threshold and Priority Sampling methods are significantly faster than the $O(Nm)$ time methods like WMH, unweighted MinHash (MH) and Johnson-Lindenstrauss (JL). With an average runtime of .06 seconds across all sketch sizes, Priority Sampling is competitive with the less accurate CountSketch, whose average runtime is .05 seconds. Threshold Sampling was slightly slower, with an average time of .21 seconds. While this method has better asymptotic complexity than Priority Sampling (since there is no need to sort ranks), its slower empirical performance is due to the algorithm used to adaptively adjust the expected sketch size to exactly equal m (discussed in Section 2). However, we emphasize that our results are primarily meant to illustrate coarse differences in runtime. Evaluating small differences between CountSketch, Priority Sampling, and Threshold Sampling would require more careful implementation in a low-level language, an effort we leave to future work. In any case, all algorithms offer extremely good performance, with no dependence on the size of the sketch.

The WMH method from [6] is not competitive with any of the other methods, requiring 43 seconds to produce a sketch of size 1000, and 213 seconds to produce a sketch of size 5000. As such, it was omitted from Fig. 7. DartMH succeeds in speeding up the method, but even this optimized algorithm is between 20x and 60x more expensive than our Priority Sampling method.

Finally, for completeness, we evaluated the estimation time for all sketches. As expected there are no significant differences, since the estimation procedure for both sampling and linear sketches amounts to a simple sum over the entries in the sketch. For sketches of size 5000, estimation times ranged between 0.014ms and 0.052ms.

5.3 Estimation Accuracy for Real-World Data

In addition to synthetic data, we carry out experiments on real-world datasets for practical applications. We use World Bank Group



(a) PS-weighted

(b) CountSketch (CS)

Figure 8: Join-correlation Estimation on World Bank data. The best sampling-based sketch (our PS-weighted method) captures correlations significantly more accurately than the best linear sketching method we tested (CS).

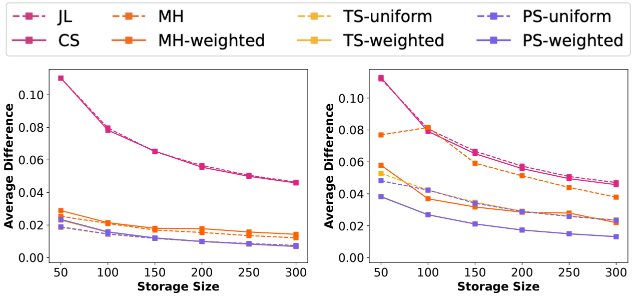
Finances data [58] to assess sketching methods for inner product and join-correlation estimation. We also evaluate the performance of Threshold and Priority Sampling for text similarity estimation on the 20 Newsgroups dataset [47], and for join-size estimation on the World Bank, Twitter [41], and TPC-H datasets [60].

5.3.1 *World Bank Finances Data.*

This collection consists of 56 tables [58], from which we randomly sampled 3,000 column pairs using the following approach (adapted from [52]). A column pair is represented as $(\langle K_A, V_A\rangle, \langle K_B, V_B\rangle)$, where K_A and K_B are join keys with temporal data, and V_A and V_B are columns with numerical values from tables A and B. Since there can be repeated keys in K_A and K_B, we pre-aggregate the values in V_A and V_B associated with repeated keys into a single value by summing them. This ensures that each key is associated with a single vector index

Inner Product Estimation. We first evaluate Threshold and Priority Sampling on the basic task of computing inner products between the data columns. We normalize all columns to have unit Euclidean norm, which ensures the inner products have a consistent scale (and are upper bounded by 1). Then we construct sketches of size 400 for all methods, which are used to estimate inner products. Table 2 shows the inner product estimation results ranked by the average error over all pairs of columns (a single trial each). We also include the R^2 score, which measures the goodness of fit of the estimated inner products to the actual inner products. The best methods are our TS-weighted and PS-weighted, followed by WMH and JL, which have average error roughly 3x larger. These results underscore the effectiveness of the weighted sampling methods.

Join-Correlation Estimation. We also evaluate Threshold and Priority Sampling for join-correlation using the estimators described in Section 4. We consider the same vectors used for evaluating inner products, and again use sketches of size 400. Table 2 shows the average error and R^2 score for all methods. PS-weighted and TS-weighted have the lowest average errors and PS-weighted has the highest R^2 score. They outperform the KMV-based sketch from [52], which is the current state-of-the-art method for join-correlation. In the table we refer to this method as PS-uniform since it is identical to Priority Sampling with uniform weights. Figure 8 shows scatter plots of correlation estimates for our PS-weighted method (the best sampling-based method) and CS (the best linear sketching method). We note that there are a large number of points around zero; this is expected since many of the datasets are not correlated.



(a) All Documents

(b) Documents > 500 words

Figure 9: Average error for text similarity estimation using the 20 Newsgroups data. The lines for PS-weighted, PS-uniform, TS-weighted, and TS-uniform overlap in (a), as do the lines for PS-weighted and TS-weighted in (b). PS-weighted and TS-weighted outperform all baselines for documents with more than 500 words.

Join Size Estimation. Finally, we evaluate our methods on the task of join size estimation using the same World Bank data, but without aggregating keys. We use the standard reduction between join size estimation and inner product computation with vectors containing key frequencies [22]. Results are presented in Table 2. Since key frequencies vary, our weighted sampling methods, TS-weighted and PS-weighted, produce more accurate results. Linear sketching methods like CountSketch and JL perform worst.

5.3.2 *20 Newsgroups Dataset.*

We also assess the effectiveness of Threshold and Priority Sampling for estimating *document similarity* using the 20 Newsgroups Dataset [47]. We generate a feature vector for each document that includes both unigrams (single words) and bigrams (pairs of words). We use standard TF-IDF weights to scale the entries of the vector [51] and then measure similarity using the cosine similarity metric, which is equivalent to the inner product when the vectors are normalized to have unit norm.

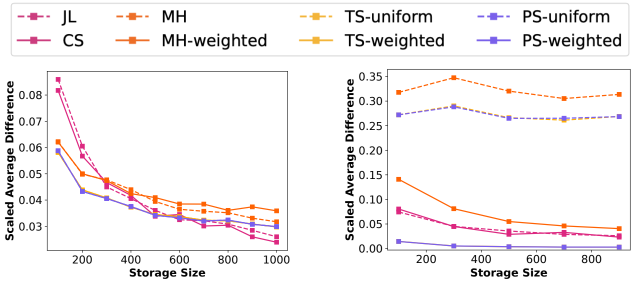
We sample 200,000 document pairs from the dataset and plot average error. As Figure 9a shows, the linear sketching methods (JL and CountSketch) perform worst. Threshold and Priority Sampling obtain the best accuracy for all sketch sizes, although the difference between the unweighted and weighted methods is negligible. As shown in Figure 9b, this difference becomes more pronounced when only considering documents with more than 500 words. For longer documents, our TS-weighted and PS-weighted perform notably better than their uniform-sampling counterparts. The larger performance gap could be due to more variability in TF-IDF weights in longer documents (which benefits the weighted sampling methods).

5.3.3 *TPC-H Benchmark and Twitter Data.*

Finally, we evaluate Threshold and Priority Sampling on two join size estimation tasks. The first is the standard TPC-H benchmark [60]. TPC-H data was generated with a scale factor of 1 and skew parameter $z = 2$. The join was performed between *LINEITEM* and *PARTSUPP* tables on the key *SUPPKEY*. Average relative error for 200 trials are presented in Figure 10a. For moderate sketch sizes (up to ∼ 600) our sampling based methods outperform linear sketching, and they always outperform MH and WMH. However, there is little difference between the weighted and unweighted sampling versions of our methods. We believe this is due to the fact that, even with skew, the

Table 2: Inner product, correlation, and join size estimations for the World Bank data, ranked by average error. Our new TS-weighted and PS-weighted methods (underlined) have both the least average error and the best R^2 score for all three problems, although differences are more pronounced for inner product and correlation estimation.

Inner Product			Join-Correlation			Join Size		
Method	Average Error	R^2 Score	Method	Average Error	R^2 Score	Method	Average Error	R^2 Score
TS-weighted	0.012	0.992	**PS-weighted**	0.079	0.828	**TS-weighted**	0.016	0.940
PS-weighted	0.012	0.991	**TS-weighted**	0.095	0.712	**PS-weighted**	0.021	0.867
CS	0.027	0.991	**CS**	0.112	0.798	**TS-uniform**	0.023	0.877
WMH	0.030	0.985	**PS-uniform**	0.124	0.613	**PS-uniform**	0.025	0.858
JL	0.033	0.988	**TS-uniform**	0.130	0.621	**MH**	0.030	0.432
TS-uniform	0.099	0.069	**MH**	0.142	0.448	**WMH**	0.030	0.833
PS-uniform	0.121	0.083	**WMH**	0.197	0.322	**CS**	0.039	0.775
MH	0.129	-0.275	**JL**	0.246	0.216	**JL**	0.040	0.775



(a) TPC-H Benchmark

(b) Twitter Self-Join

Figure 10: Join size estimation for the Twitter and TPC-H datasets. The lines for PS-weighted and TS-weighted overlap, as do the lines for PS-uniform and TS-uniform. Our PS-weighted and TS-weighted methods are most reliable, performing well in both experiments, the second of which involves two tables with highly non-uniform key distributions.

TPC-H benchmark only has a non-uniform key distribution in the LINEITEM table. The key distribution of the larger *PARTSUPP* table remains uniform. Difference between the methods is much more pronounced in our second experiment on estimating join sizes using the Twitter dataset from [41]. This data consists of a list of tuples (user, follower), representing the follower-followee relationship. We sampled 14,000,000 (user, follower) tuples from the dataset, which include approximately 420,000 users. Following the example in [12], we perform a self-join of the table to identify all the 2-hop "follows" relationships. Results are shown in Figure 10b. For all sketch sizes, TS-weighted and PS-weighted have the smallest errors, followed by the linear sketching methods, and then by WMH. The unweighted sampling methods (MH, TS-uniform, PS-uniform) perform poorly, since in this dataset there is a lot of variability in key frequencies.

6 ADDITIONAL RELATED WORK

As discussed in Section 1, we are only aware of two previous papers that directly address the inner product estimation problem using sampling-based sketches: the WMH work of [6] and the End-Biased Sampling work of [33]. Some follow-up work on End-Biased Sampling, such as Correlated Sampling [57] and Two-level Sampling [12], can also be used to estimate inner products. However, the goal of these works is to handle the more general problem of approximating data operations (such as SUM, COUNT, MEAN) with SQL predicates (WHERE clauses). In our setting, the methods from [57] and [12] degenerate to uniform sampling methods (i.e., KMV or Threshold Sampling with uniform weights), as they do not take into account the vector entries (i.e., $\mathbf{a}_i$ and $\mathbf{b}_i$) when selecting samples.

We also note that inner product estimation can be seen as a special case of the predicate aggregation problem studied in [20]. While that work gives unbiased estimators based on Threshold and Priority Sampling, inner product estimation is not considered specifically, so there is no guidance on how probabilities should be chosen or variance analyzed. Follow-up work in [16] can be used to analyze variance given a choice of probabilities. However, in our setting, the work leads to loose bounds that depend on $\max_i |\mathbf{a}_i\mathbf{b}_i|/\min(\mathbf{a}_i^2, \mathbf{b}_i^2)$. This value can be arbitrarily large in comparison to $\|\mathbf{a}\|_2\|\mathbf{b}\|_2$, so unlike our analysis, this prior work cannot be used to beat the linear sketching guarantee of (1) for inner product estimation.

Beyond the problem of inner product estimation, our work is more broadly related to the large body of work on coordinated random sampling methods, which use shared randomness (e.g., a shared hash function or random permutation) to collect samples of two vectors $\mathbf{a}$ and $\mathbf{b}$. Threshold and Priority Sampling are both examples of coordinated sampling, as is MinHash and the k-minimum values (KMV) sketch. However, there are other methods, including the coordinated random sampling method [18], conditional random sampling [43], and coordinated variants of PPSWOR sampling [17].

7 CONCLUSION

We propose two simple and efficient sampling-based sketches for inner product estimation. We prove theoretical accuracy guarantees for both methods that are stronger than the guarantees of popular linear sketching methods, and that match the best-known guarantees of the state-of-the-art hashing-based WMH sketch [6]. At the same time, our methods run in near-linear time, so are much faster than WMH. They also perform better in our empirical evaluation. In particular, our fixed-size Priority Sampling method provides a new state-of-the-art for inner product estimation and related applications, including join-correlation estimation.

ACKNOWLEDGMENTS

We thank Otmar Ertl, Jonathan Weare, and Xiaoou Cheng for helpful conversations. This research was supported by NSF awards IIS-2106888, CMMI-2146306, and CCF-2046235; and DARPA D3M and ASKEM programs. Opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not reflect the views of NSF or DARPA.

REFERENCES

[1] Achlioptas, D. (2003). Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4). Preliminary version in the 20th Symposium on Principles of Database Systems (PODS).

[2] Alon, N., Duffield, N., Lund, C., and Thorup, M. (2005). Estimating arbitrary subset sums with few probes. In *Proceedings of the 24th Symposium on Principles of Database Systems (PODS)*.

[3] Alon, N., Gibbons, P. B., Matias, Y., and Szegedy, M. (1999a). Tracking join and self-join sizes in limited storage. In *Proceedings of the 18th Symposium on Principles of Database Systems (PODS)*.

[4] Alon, N., Matias, Y., and Szegedy, M. (1999b). The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1).

[5] Arriaga, R. I. and Vempala, S. (2006). An algorithmic theory of learning: Robust concepts and random projection. *Machine Learning*, 63(2).

[6] Bessa, A., Daliri, M., Freire, J., Musco, C., Musco, C., Santos, A., and Zhang, H. (2023). Weighted minwise hashing beats linear sketching for inner product estimation. In *Proceedings of the 42nd Symposium on Principles of Database Systems (PODS)*.

[7] Beyer, K., Haas, P. J., Reinwald, B., Sismanis, Y., and Gemulla, R. (2007). On synopses for distinct-value estimation under multiset operations. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*.

[8] Broder, A. (1997). On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997*.

[9] Castro Fernandez, R., Min, J., Nava, D., and Madden, S. (2019). Lazo: A cardinality-based method for coupled estimation of jaccard similarity and containment. In *Proceedings of the 35th IEEE International Conference on Data Engineering (ICDE)*.

[10] Charikar, M. (2002). Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC)*.

[11] Charikar, M., Chen, K., and Farach-Colton, M. (2002). Finding frequent items in data streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP)*.

[12] Chen, Y. and Yi, K. (2017). Two-level sampling for join size estimation. In *Proceedings of the 2017 ACM International Conference on Management of Data*.

[13] Chepurko, N., Marcus, R., Zgraggen, E., Fernandez, R. C., Kraska, T., and Karger, D. (2020). Arda: automatic relational data augmentation for machine learning. *Proc. VLDB Endow.*, 13(9).

[14] Chi, L. and Zhu, X. (2017). Hashing techniques: A survey and taxonomy. *ACM Comput. Surv.*, 50(1).

[15] Christiani, T. (2020). Dartminhash: Fast sketching for weighted sets. *arXiv:2005.11547*.

[16] Cohen, E. (2015). Multi-objective weighted sampling. In *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*.

[17] Cohen, E. (2023). Sampling big ideas in query optimization. In *Proceedings of the 42nd Symposium on Principles of Database Systems (PODS)*.

[18] Cohen, E. and Kaplan, H. (2007). Summarizing data using bottom-k sketches. In *Proceedings of the 2007 ACM Symposium on Principles of Distributed Computing (PODC)*.

[19] Cohen, E. and Kaplan, H. (2013). What you can do with coordinated samples. In *Proceedings of the 16th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*.

[20] Cohen, E., Kaplan, H., and Sen, S. (2009). Coordinated weighted sampling for estimating aggregates over multiple weight assignments. *Proc. VLDB Endow.*, 2(1).

[21] Cohen, E., Pagh, R., and Woodruff, D. (2020). Wor and p's: Sketches for \ell_p-sampling without replacement. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21092–21104. Curran Associates, Inc.

[22] Cormode, G. (2011). Sketch techniques for approximate query processing. *Foundations and Trends in Databases. NOW publishers*.

[23] Cormode, G. and Garofalakis, M. (2005). Sketching streams through the net: Distributed approximate query tracking. *Proc. VLDB Endow.*

[24] Cormode, G. and Garofalakis, M. (2016). *Join Sizes, Frequency Moments, and Applications*. Springer Berlin Heidelberg.

[25] Cormode, G., Garofalakis, M., Haas, P., and Jermaine, C. (2011). *Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches*. Foundations and Trends in Databases. NOW publishers.

[26] Daliri, M., Freire, J., Musco, C., Santos, A., and Zhang, H. (2023a). Sampling methods for inner product sketching. *arXiv:2309.16157*.

[27] Daliri, M., Freire, J., Musco, C., Santos, A., and Zhang, H. (2023b). Simple analysis of priority sampling. *SIAM Symposium on Simplicity in Algorithms (SOSA24)*.

[28] Dasgupta, S. and Gupta, A. (2003). An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1).

[29] Duffield, N., Lund, C., and Thorup, M. (2004). Flow sampling under hard resource constraints. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2004)*.

[30] Duffield, N., Lund, C., and Thorup, M. (2005). Learn more, sample less: control of volume and variance in network measurement. *IEEE Transactions on Information Theory*, 51(5).

[31] Ertl, O. (2018). Bagminhash - minwise hashing algorithm for weighted sets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

[32] Esmailoghli, M., Quiané-Ruiz, J.-A., and Abedjan, Z. (2021). Cocoa: Correlation coefficient-aware data augmentation. In *24th International Conference on Extending Database Technology (EDBT)*.

[33] Estan, C. and Naughton, J. (2006). End-biased samples for join cardinality estimation. In *Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE)*.

[34] Flajolet, P. (1990). On adaptive sampling. *Computing*, 43(4).

[35] Gollapudi, S. and Panigrahy, R. (2006). Exploiting asymmetry in hierarchical topic extraction. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM)*.

[36] Howell, D. (2018). Generating correlated data. *Outline of the Statistical Pages Folder*.

[37] Ioffe, S. (2010). Improved consistent sampling, weighted minhash and l1 sketching. In *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM)*.

[38] Ionescu, A., Hai, R., Fragkoulis, M., and Katsifodimos, A. (2022). Join path-based data augmentation for decision trees. In *2022 IEEE 38th International Conference on Data Engineering Workshops (ICDEW)*. IEEE.

[39] Jacques, L. (2015). A quantized johnson–lindenstrauss lemma: The finding of buffon's needle. *IEEE Transactions on Information Theory*, 61(9).

[40] Jayaram, R. and Woodruff, D. P. (2018). Perfect l_p sampling in a data stream. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 544–555.

[41] Kwak, H., Lee, C., Park, H., and Moon, S. (2010). What is Twitter, a social network or a news media? In *Proceedings of the 19th International World Wide Web Conference (WWW)*, New York, NY, USA. ACM.

[42] Larsen, K. G., Pagh, R., and Tětek, J. (2021). Countsketches, feature hashing and the median of three. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*. PMLR.

[43] Li, P., Church, K., and Hastie, T. (2006). Conditional random sampling: A sketch-based sampling technique for sparse data. In *Advances in Neural Information Processing Systems 19 (NeurIPS)*, volume 19.

[44] Li, P. and König, A. C. (2011). Theory and applications of b-bit minwise hashing. *Commun. ACM*, 54(8).

[45] Liu, J., Chai, C., Luo, Y., Lou, Y., Feng, J., and Tang, N. (2022). Feature augmentation with reinforcement learning. In *Proceedings of the 38th IEEE International Conference on Data Engineering (ICDE)*.

[46] Manasse, M., McSherry, F., and Talwar, K. (2010). Consistent weighted sampling. Technical Report MSR-TR-2010-73, Microsoft Research.

[47] Mitchell, T. (1997). 20 newsgroups dataset. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html.

[48] Ohlsson, E. (1998). Sequential poisson sampling. *Journal of Official Statistics*, 14(2).

[49] Pagh, R., Stöckel, M., and Woodruff, D. P. (2014). Is min-wise hashing optimal for summarizing set intersection? In *Proceedings of the 33rd Symposium on Principles of Database Systems (PODS)*.

[50] Rusu, F. and Dobra, A. (2008). Sketches for size of join estimation. *ACM Transactions on Database Systems (TODS)*, 33(3).

[51] Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11).

[52] Santos, A., Bessa, A., Chirigati, F., Musco, C., and Freire, J. (2021). Correlation sketches for approximate join-correlation queries. In *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*.

[53] Santos, A., Bessa, A., Musco, C., and Freire, J. (2022). A sketch-based index for correlated dataset search. In *Proceedings of the 38th IEEE International Conference on Data Engineering (ICDE)*.

[54] Shrivastava, A. (2016). Simple and efficient weighted minwise hashing. In *Advances in Neural Information Processing Systems 29 (NeurIPS)*.

[55] Szegedy, M. (2006). The DLT priority sampling is essentially optimal. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*.

[56] Szegedy, M. and Thorup, M. (2007). On the variance of subset sum estimation. In *Proceedings of the 15th European Symposium on Algorithms (ESA)*. Springer Berlin Heidelberg.

[57] Vengerov, D., Menck, A. C., Zait, M., and Chakkappen, S. P. (2015). Join size estimation subject to filter conditions. *Proc. VLDB Endow.*, 8(12).

[58] World Bank (2023). World bank group finances. https://finances.worldbank.org/.

[59] Yang, Y., Zhang, Y., Zhang, W., and Huang, Z. (2019). Gb-kmv: An augmented kmv sketch for approximate containment similarity search. In *Proceedings of the 35th IEEE International Conference on Data Engineering (ICDE)*.

[60] Yu, F. (2022). Tpch skew. https://github.com/YSU-Data-Lab/TPC-H-Skew.

[61] Zhu, E., Nargesian, F., Pu, K. Q., and Miller, R. J. (2016). LSH ensemble: Internet-scale domain search. *Proc. VLDB Endow.*, 9(12).