Weighted Minwise Hashing Beats Linear Sketching for Inner Product Estimation

Aline Bessa*
New York University
aline.bessa@nyu.edu

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

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

Cameron Musco
University of Massachusetts Amherst
cmusco@cs.umass.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

We present a new approach for independently computing compact sketches that can be used to approximate the inner product between pairs of high-dimensional vectors. Based on the Weighted MinHash algorithm, our approach admits strong accuracy guarantees that improve on the guarantees of popular *linear sketching* approaches for inner product estimation, such as CountSketch and Johnson-Lindenstrauss projection. Specifically, while our method exactly matches linear sketching for dense vectors, it yields significantly *lower* error for sparse vectors with limited overlap between non-zero entries. Such vectors arise in many applications involving sparse data, as well as in increasingly popular dataset search applications, where inner products are used to estimate data covariance, conditional means, and other quantities involving columns in *unjoined tables*. We complement our theoretical results by showing that our approach empirically outperforms existing linear sketches and unweighted hashing-based sketches for sparse vectors.

CCS CONCEPTS

• **Information systems** → **Data management systems**; *Data structures*; *Join algorithms*; • **Theory of computation** → **Sketching and sampling**.

KEYWORDS

inner product estimation, vector sketching, join-size estimation

ACM Reference Format:
Aline Bessa, Majid Daliri, Juliana Freire, Cameron Musco, Christopher Musco, Aécio Santos, and Haoxiang Zhang. 2023. Weighted Minwise Hashing Beats Linear Sketching for Inner Product Estimation . In *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS '23), June 18–23, 2023, Seattle, WA, USA*. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3584372.3588679

*Author names are listed in alphabetical order.

PODS '23, June 18–23, 2023, Seattle, WA, USA

ACM ISBN 979-8-4007-0127-6/23/06…$15.00
https://doi.org/10.1145/3584372.3588679

1 INTRODUCTION

The *inner product* of two vectors $\mathbf{a}$ and $\mathbf{b}$, $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{k=1}^{n} \mathbf{a}[k]\mathbf{b}[k]$, is a ubiquitous operation. Among many other applications, inner products can be used to compute document similarities [46], to evaluate learned classification models, and to estimate join sizes [1, 4, 45]. However, in modern applications involving very high-dimensional vectors, computing exact inner products can be intractable. The computational cost is $O(n)$ and computing $\langle \mathbf{a}, \mathbf{b} \rangle$ requires loading $O(n)$ numbers from memory, or communicating $O(n)$ numbers if $\mathbf{a}$ and $\mathbf{b}$ are stored on different machines.

A common approach for resolving this issue is to pre-compute a small space compression (a sketch) of each vector, which we will denote by $\mathcal{S}(\mathbf{a})$ and $\mathcal{S}(\mathbf{b})$, respectively. An estimation function $\mathcal{F}$ is then used to approximate the inner product as $\mathcal{F}(\mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b})) \approx \langle \mathbf{a}, \mathbf{b} \rangle$. The beauty of sketching is that it simultaneously reduces storage, communication, and runtime complexity. Moreover, once computed, sketches can be reused again and again to estimate inner products with other vectors. For example, given another vector $\mathbf{c}$ we can estimate $\langle \mathbf{a}, \mathbf{c} \rangle \approx \mathcal{F}(\mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{c}))$.

Sketching methods for approximating inner products are already widely used throughout computer science. In machine learning, they can be used to accelerate the training of large-scale linear models like support vector machines or logistic regression [5, 40]. In relational databases, inner product sketches are used in query optimizers to choose optimal query plans without having to execute expensive queries that involve large joins [19]. More recently, inner product sketches have found applications in dataset search and discovery, where they are used to discover joinable tables [22] and to estimate other column statistics, such as correlation [47], without explicitly performing a join operation between two tables. We discuss these applications and others in Section 1.2.

What was Previously Known? In all of the applications above, a primary concern is optimizing the trade-off between the sketch size (which governs storage, communication, and runtime efficiency) and how accurately $\mathcal{F}(\mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b}))$ approximates $\langle \mathbf{a}, \mathbf{b} \rangle$. A large sketch size will in general lead to better approximation, but the question is by exactly how much. Currently, the only methods with strong theoretical guarantees on this tradeoff for *general vectors* (i.e., vectors without any assumed value distribution or magnitude) are based on *linear sketching* algorithms. Such algorithms include the famous "tug-of-war" sketch, a.k.a. the AMS sketch [2, 4], the CountSketch algorithm [12], and methods based on Johnson-Lindenstrauss (JL) random projection [1, 20].

All of these approaches have a similar form. We choose a random matrix $\mathbf{\Pi} \in \mathbb{R}^{m \times n}$ ($\mathbf{\Pi}$ might have i.i.d. random entries or more complex structure) and sets $\mathcal{S}(\mathbf{a}) = \mathbf{\Pi a}$ and $\mathcal{S}(\mathbf{b}) = \mathbf{\Pi b}$. Each sketch is a length m vector and is considered a *linear sketch* since $\mathcal{S}$ is a linear function. To estimate the inner product, the typical approach is to simply return the sketch inner product $\langle \mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b}) \rangle$.[1]

A textbook theoretical accuracy guarantee for inner product estimation based on linear sketching is:

Fact 1 (Linear Sketching for Inner Products [5]). *Let $\epsilon, \delta \in (0, 1)$ be accuracy and failure probability parameters respectively and let $m = O(\log(1/\delta)/\epsilon^2)$. Let $\mathbf{\Pi} \in \mathbb{R}^{m \times n}$ be a random matrix with each entry set independently to $+\sqrt{1/m}$ or $-\sqrt{1/m}$ with equal probability. For length n vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, let $\mathcal{S}(\mathbf{a}) = \mathbf{\Pi a}$ and $\mathcal{S}(\mathbf{b}) = \mathbf{\Pi b}$. With probability at least $1 - \delta$,*

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

where $\|\mathbf{x}\|$ denotes the standard Euclidean norm.

In addition to dense random matrices, analogous results to Fact 1 can be proven for sparse JL matrices, CountSketch matrices, and other linear sketches [19]. The fact provides a powerful accuracy guarantee that improves with the sketch size m and depends naturally on the norms of $\mathbf{a}$ and $\mathbf{b}$. To the best of our knowledge, linear sketching methods were previously the only known algorithms to obtain such a strong theoretical guarantee.

1.1 Our Contributions

In this paper we introduce a novel method for inner product sketching based on the Weighted MinHash sketch [25, 28, 41], which is a variant of the classic MinHash method [8, 9]. We prove that our method obtains a refined guarantee than Fact 1. In particular, it matches the result for linear sketches in the worst case when $\mathbf{a}$ and $\mathbf{b}$ are dense[2], but always obtains a *better bound* when $\mathbf{a}$ and $\mathbf{b}$ are sparse vectors with limited overlap between non-zero entries. As discussed further in Section 1.2, such pairs of vectors are the norm in many applications of inner product sketching to database problems and modern dataset search applications.

Theorem 2 (Main Result). *Let $\epsilon, \delta \in (0, 1)$ be accuracy and failure probability parameters and let $m = O(\log(1/\delta)/\epsilon^2)$. There is an algorithm $\mathcal{S}$ that produces size-m sketches (Algorithm 3), along with an estimation procedure $\mathcal{F}$ (Algorithm 5), such that for any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, with probability at least $1 - \delta$,*

$$|\mathcal{F}(\mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b})) - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \epsilon \max(\|\mathbf{a}_{\mathcal{I}}\| \|\mathbf{b}\|, \|\mathbf{a}\| \|\mathbf{b}_{\mathcal{I}}\|)$$

Above, $\mathcal{I} = \{i : \mathbf{a}[i] \neq 0 \text{ and } \mathbf{b}[i] \neq 0\}$ is the intersection of $\mathbf{a}$'s and $\mathbf{b}$'s supports. $\mathbf{a}_{\mathcal{I}}$ and $\mathbf{b}_{\mathcal{I}}$ denote $\mathbf{a}$ and $\mathbf{b}$ restricted to indices in $\mathcal{I}$.

We always have $\|\mathbf{a}_{\mathcal{I}}\| \leq \|\mathbf{a}\|$ and $\|\mathbf{b}_{\mathcal{I}}\| \leq \|\mathbf{b}\|$, so we can bound $\max(\|\mathbf{a}_{\mathcal{I}}\| \|\mathbf{b}\|, \|\mathbf{a}\| \|\mathbf{b}_{\mathcal{I}}\|) \leq \|\mathbf{a}\| \|\mathbf{b}\|$. That is, the guarantee of Theorem 2 matches that of Fact 1 in the worse-case, but can be significantly better. For example, consider $\mathbf{a}$ and $\mathbf{b}$ that have roughly the same number of non-zero entries, but only a $\gamma < 1$ fraction of those entries are non-zero in *both* $\mathbf{a}$ and $\mathbf{b}$. In this case, it is reasonable to expect that $\|\mathbf{a}_{\mathcal{I}}\|^2 \approx \gamma \|\mathbf{a}\|^2$ and $\|\mathbf{b}_{\mathcal{I}}\|^2 \approx \gamma \|\mathbf{b}\|^2$ since $\mathbf{a}_{\mathcal{I}}$ and $\mathbf{b}_{\mathcal{I}}$ contain just a γ fraction of entries from the original vectors. Our course, the actually improvement is data dependent; for example, we might have that $\|\mathbf{a}_{\mathcal{I}}\|^2$ is significantly smaller than $\gamma \|\mathbf{a}\|^2$, or that it is not much smaller than $\|\mathbf{a}\|^2$.

Nevertheless, considering the "typical case" when a γ fraction of non-zeros overlap, we might expect the bound from Theorem 2 to be better than Fact 1 by a factor of $\sqrt{\gamma}$. So, to obtain the same error as a linear sketch, our method could set m smaller by a factor of γ. In many applications, γ is very small. E.g., in Section 5 we consider a document similarity problem where $\gamma \leq .05$ for 95% of vector pairs sketched. This could equate to roughly a $20x$ improvement in sketch size required to achieve a specified level of error.

Thanks to their strong theoretical guarantees, linear sketching algorithms have become the go-to approach for generic inner product estimation [19]. Our results show for the first time that an alternative method can provide stronger bounds. We hope that this paper will serve as a starting point for further investigation into hashing-based algorithms for inner product sketching.

1.2 Motivating Application: Dataset Search

Before presenting the technical details of our results and discussing related work, we detail one application that could benefit from our proposed sketches, and helps illustrate the importance of obtaining bounds for inner product estimation that are sensitive to the number of overlapping non-zero entries in $\mathbf{a}$ and $\mathbf{b}$. Specifically, we consider the problem of *dataset search* which has received increasing attention in recent years [22, 34, 47, 48, 54–56].

Suppose that a data scientist wants to understand the reasons for fluctuations in taxi ridership in New York City in 2022. The analyst only has a table containing two columns: a *date* column and the *number of taxi rides* taken on that day. In order to carry out the analysis, she needs to find other tables, either in her organization's data lake or in public repositories like NYC Open Data (which contain thousands of datasets [15]), that would bring in other relevant variables when joined with the original table. For example, the analyst might hope to find weather data, which can impact taxi ridership. Moreover, she would like to find relevant factors that she *might not think of on her own*, in an *automatic way*.

To solve this problem, we would like methods to automatically discover tables that are both 1) joinable with the target table (i.e., also contain columns with dates from 2022) and 2) meaningfully related with the analyst's data. For example, a table containing precipitation data should be returned if taxi ridership is significantly higher or lower on days with high precipitation. To find such tables, brute force search is not infeasible – we typically cannot afford to join the analyst's table with all tables in the search set to look for good candidates. Instead, we need to efficiently estimate statistics between disparate tables *without materializing their join* [47].

Sketching has become the most popular approach for performing this sort of estimation between unjoined data tables [22, 47, 48, 54, 56]. Specifically, a small-space sketch is *precomputed* for all data tables in the search set. When the analyst issues a query to find relevant data, a sketch of her table is compared against these preexisting sketches using a fraction of the computational resources in comparison to explicitly materializing table joins [47].

[1] Other estimators involving e.g., the median of multiple approximate inner products, are also used [33]. However, theoretical guarantees are similar, typically differing in the dependence on the failure probability δ

[2] For dense vectors, Fact 1 is actually optimal up to constants: recent work implies that no sketch of size $m = o(\log(1/\delta)/\epsilon^2)$ can achieve error $\epsilon \|\mathbf{a}\| \|\mathbf{b}\|$ with probability $1 - \delta$ for all inputs [3, 32]. Our result also matches this lower bound.

Method	Error for sketches of size $O(1/\epsilon^2)$	Assumptions
JL [5], AMS [2, 4], CountSketch [12]	$\epsilon \cdot \|\mathbf{a}\|\|\mathbf{b}\|$	None
MinHash (MH) Sampling [6, 44]	$\epsilon \cdot \max(\|\mathbf{a}_{\mathcal{I}}\|\|\mathbf{b}\|, \|\mathbf{a}\|\|\mathbf{b}_{\mathcal{I}}\|)$	$\mathbf{a}, \mathbf{b}$ are binary, i.e. with $\{0, 1\}$ entries.
Weighted MinHash (WMH) Sampling (our method)	$\epsilon \cdot \max(\|\mathbf{a}_{\mathcal{I}}\|\|\mathbf{b}\|, \|\mathbf{a}\|\|\mathbf{b}_{\mathcal{I}}\|)$	None

Table 1: High-probability additive error guarantees for estimating $\langle \mathbf{a}, \mathbf{b} \rangle$ using various sketching methods. We let $\mathcal{I} = \{i : \mathbf{a}[i] \neq 0 \text{ and } \mathbf{b}[i] \neq 0\}$ denote the intersection of a's and b's supports. $\mathbf{a}_{\mathcal{I}}$ and $\mathbf{b}_{\mathcal{I}}$ are $\mathbf{a}$ and $\mathbf{b}$ restricted to indices in $\mathcal{I}$. Since $\max(\|\mathbf{a}_{\mathcal{I}}\|\|\mathbf{b}\|, \|\mathbf{a}\|\|\mathbf{b}_{\mathcal{I}}\|) \leq \|\mathbf{a}\|\|\mathbf{b}\|$, the bound for our Weighted MinHash method is better than the linear sketching methods. Unweighted MinHash only matches our bound under the strong limiting assumption that $\mathbf{a}$ and $\mathbf{b}$ are binary.

$\mathcal{T}_A$

K_A	V_A
1	6.0
3	2.0
4	6.0
5	1.0
6	4.0
7	2.0
8	2.0
9	8.0
11	3.0

$\mathcal{T}_B$

K_B	V_B
2	1.0
4	5.0
5	1.0
8	2.0
10	4.0
11	2.5
12	6.0
15	6.0
16	3.7

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

$K_{A \bowtie B}$	$V_{A \bowtie}$	$V_{B \bowtie}$
4	6.0	5.0
5	1.0	1.0
8	2.0	2.0
11	3.0	2.5

$\texttt{SIZE}(V_{A \bowtie}) = 4$
$\texttt{SUM}(V_{A \bowtie}) = 12.0$
$\texttt{SUM}(V_{B \bowtie}) = 10.5$
$\texttt{MEAN}(V_{A \bowtie}) = 12.0/4 = 3.0$

Figure 1: The table $\mathcal{T}_{A \bowtie B}$ is the output of a one-to-one join between the tables $\mathcal{T}_A$ with $\mathcal{T}_B$. We are interested in approximating post-join statistics (e.g., join size, sums, means, and covariances) of the table $\mathcal{T}_{A \bowtie B}$ using only inner products.

index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\mathbf{x}^{V_A}$	6.0	0	2.0	**6.0**	**1.0**	4.0	2.0	**2.0**	8.0	0	**3.0**	0	0	0	0	0
$\mathbf{x}^{\mathbb{1}[K_A]}$	1	0	1	**1**	**1**	1	1	**1**	1	0	**1**	0	0	0	0	0
$\mathbf{x}^{V_B}$	0	1.0	0	**5.0**	**1.0**	0	0	**2.0**	0	4.0	**2.5**	6.0	0	0	6.0	3.7
$\mathbf{x}^{\mathbb{1}[K_B]}$	0	1	0	**1**	**1**	0	0	**1**	0	1	**1**	1	0	0	1	1

Figure 2: Vector representation of tables $\mathcal{T}_A$ with $\mathcal{T}_B$ from Figure 1. The vector $\mathbf{x}^{\mathbb{1}[K_A]}$ (resp. $\mathbf{x}^{\mathbb{1}[K_B]}$) is the vector representation for the join key K_A (resp. K_B) and $\mathbf{x}^{V_A}$ (resp. $\mathbf{x}^{V_B}$) is the vector representation for the column V_A (resp. V_B). Bold numbers are entries included in the join result $\mathcal{T}_{A \bowtie B}$.

Inner product sketching for dataset search. Interestingly, in the framework discussed above, many problems of interest can be formulated precisely as inner product sketching problems. To see why this is the case, consider the example tables $\mathcal{T}_A$ and $\mathcal{T}_B$ shown in Figure 1: each contains a column of keys, K_A and K_B, and a column of values, V_A and V_B. A *join* operation between the tables on their keys generates the output table $\mathcal{T}_{A \bowtie B}$.[3]

We list in Figure 1 a number of statistics that we might hope to estimate in $\mathcal{T}_{A \bowtie B}$ when searching for relevant datasets. We claim that all of these statistics can be estimated using inner products between vector representations of the tables, which we denote $\mathbf{x}^{\mathbb{1}[K_A]}, \mathbf{x}^{K_A}$ and $\mathbf{x}^{\mathbb{1}[K_B]}, \mathbf{x}^{K_B}$ respectively and show in Figure 2.

First, it is easy to see that the size of $\mathcal{T}_{A \bowtie B}$ is equal to the intersection between the keys in K_A and K_B, i.e., $|K_A \cap K_B| = 4$. This is in turn equal to the *inner product* between $\mathbf{x}^{\mathbb{1}[K_A]}$ and $\mathbf{x}^{\mathbb{1}[K_B]}$. Similarly, the SUM aggregate of the values in V_A after join (i.e., $\texttt{SUM}(V_{A \bowtie})$) is equal to the inner product $\texttt{SUM}(V_{A \bowtie}) = \langle \mathbf{x}^{V_A}, \mathbf{x}^{\mathbb{1}[K_B]} \rangle$. To estimate a post-join mean (i.e., $\texttt{MEAN}(V_{A \bowtie})$), we can combine the join-size estimate with the SUM estimate:

$$\texttt{MEAN}(V_{A \bowtie}) = \frac{\langle \mathbf{x}^{V_A}, \mathbf{x}^{\mathbb{1}[K_B]} \rangle}{\langle \mathbf{x}^{\mathbb{1}[K_A]}, \mathbf{x}^{\mathbb{1}[K_B]} \rangle}.$$

Finally, computing a post-join inner product, $\langle \mathbf{x}^{V_A}, \mathbf{x}^{V_B} \rangle$ could be useful. In the application above, for tables containing precipitation data and taxi ridership, a high inner-product might signify that high precipitation days align with high ridership days.

Comparison of different methods. Given the above reductions, both linear sketching methods like JL projection and CountSketch, and our Weighted MinHash method, can be directly applied to the dataset search problem. We simply need to precompute $\mathcal{S}(\mathbf{x}^{\mathbb{1}[K_B]})$ and $\mathcal{S}(\mathbf{x}^{V_B})$ for all tables $\mathcal{T}_B$ in our search set. Sketching other vector transformations like $\mathcal{S}((\mathbf{x}^{V_B})^2)$ opens up the possibility of also estimating other quantities like post-join variance.

In search applications, we note that the vector length n can be very large. However, computing sketches does not require fully materializing the vectors $\mathbf{x}^{\mathbb{1}[K_A]}$ and $\mathbf{x}^{\mathbb{1}[K_B]}$: all sketching methods discussed in this paper only need to process the vectors' non-zero entries. Furthermore, it is not necessary to know the n beforehand: we can simply set n to be large enough to cover the whole domain of the keys being sketched (e.g., $n = 2^{32}$ or $n = 2^{64}$).

To compare methods, Fact 1 and Theorem 2 suggest that any asymptotic differences in performance between our WMH method and linear sketching will depend on the overlap in non-zero entries between the vectors being sketched. In dataset search, this exactly corresponds to the *Jaccard similarity* of the key sets K_A and K_B. Our method will perform better when the Jaccard similarity is small. For example, in Figure 1, only 4 out of 14 unique keys are shared in both tables, so the similarity is $\approx .29$. In the scenario discussed above, we could imagine a much smaller ratio: for example, our data analyst might only have a table containing taxi data from 2022, but compare it to a weather data table with dates from 1960 through the present day. The Jaccard similarity would be $1/63 \approx .016$. In Section 5 we consider a dataset search use case involving data from the World Bank [51] where 42% percent of table pairs had Jaccard similarity $< .1$, and 35% have Jaccard similarity $< .05$.

1.3 Paper Roadmap

In Section 2 we review related prior work. In Section 3 we outline an analysis of the standard *unweighted* MinHash method for inner product estimation. This analysis serves as a technical warm-up for our main result (Theorem 2) on Weighted MinHash, which is presented in Section 4. Finally, in Section 5 we support Theorem 2 with a detailed empirical evaluation of our method.

[3] Note that, in the example described in Figure 1, we assume a one-to-one join. Dataset search problems can involve many-to-many joins as well, although a typical approach is to use a data aggregation function to reduce to the one-to-one setting [31, 47, 48].

2 RELATED WORK

Inner Product Estimation for Binary Vectors. Beyond linear sketching methods for estimating the inner product between general real-valued vectors $\mathbf{a}$ and $\mathbf{b}$, there has been a lot of prior work on the special case of *binary* vectors with $\{0, 1\}$ entries. For such vectors, approximating the inner product amounts to approximating the size of the intersection of two sets. Concretely, any $\mathbf{a}, \mathbf{b} \in \{0, 1\}^n$ can be associated with sets $\mathcal{A}$ and $\mathcal{B}$ that contain integers from $\{1, \ldots, n\}$. We define $\mathcal{A}$ to contain all i for which $\mathbf{a}[i] = 1$, and similarly $\mathcal{B}$ to contain all i for which $\mathbf{b}[i] = 1$. Note that $\langle \mathbf{a}, \mathbf{b} \rangle = |\mathcal{A} \cap \mathcal{B}|$.

Applying Fact 1, we know that a linear sketch of size $m = O(1/\epsilon^2)$ can estimate $\langle \mathbf{a}, \mathbf{b} \rangle$ up to additive error $\epsilon \|\mathbf{a}\| \|\mathbf{b}\| = \epsilon \sqrt{|\mathcal{A}||\mathcal{B}|}$. However, a better bound can be obtained using non-linear sketching methods based on the classic MinHash sketch [8, 9, 27, 42], the k-minimum value (KMV) sketch [6], or related techniques [37, 39]. With $m = O(1/\epsilon^2)$ space, such methods are achieve error $\epsilon \sqrt{\max(|\mathcal{A}|, |\mathcal{B}|) \cdot |\mathcal{A} \cap \mathcal{B}|}$, which is always smaller than $\epsilon \sqrt{|\mathcal{A}||\mathcal{B}|}$ [6, 44]. For binary vectors, this bound was proven optimal in [44].

Our work was motivated by this pre-existing result for binary vectors. In fact, our Theorem 2, is a strict generalization of the bound to *all real-valued vectors*. When $\mathbf{a}$ and $\mathbf{b}$ are binary, we have that $\|\mathbf{a}_{\mathcal{I}}\|^2 = \|\mathbf{b}_{\mathcal{I}}\|^2 = |\mathcal{A} \cap \mathcal{B}|$. So it is not hard to see that $\epsilon \sqrt{\max(|\mathcal{A}|, |\mathcal{B}|) \cdot |\mathcal{A} \cap \mathcal{B}|} = \epsilon \cdot \max\left(\|\mathbf{a}_{\mathcal{I}}\| \|\mathbf{b}\|, \|\mathbf{a}\| \|\mathbf{b}_{\mathcal{I}}\|\right)$, which is exactly our bound from Theorem 2. We summarize how all prior inner product sketching methods compare to our result in Table 1.

Beyond Binary Vectors. There has been less work on obtaining better results for estimating inner products of vectors with non-binary entries. One recent paper [33] proves refined bounds for the CountSketch method that depend on the ℓ_1 norm of $\mathbf{a}$ and $\mathbf{b}$ (instead of the Euclidean norm). These bounds can be tighter than Fact 1 for some vectors, especially when the sketch size m is large. However, the results are not directly comparable to ours.

We take a different approach, moving beyond linear sketching entirely. Our main result is based on a class of sketches that we collectively refer to as "Weighted MinHash" methods [13, 49]. These methods include weighted versions of coordinated random sampling [17, 18], as well as the "Consistent Weighted Sampling" algorithm [25, 41] and its descendants, which are essentially equivalent, but computationally cheaper to apply [26, 28, 53]. As shown in Section 4, Weighted MinHash sketches allows us to handle vectors whose entries have *highly varying magnitude* (in contrast to binary vectors, where all non-zero entries have the same magnitude of 1).

Weighted MinHash sketches have been used in a number of applications, including for approximating weighted Jaccard similarity [53], for near-duplicate detection with weighted features [41], for approximating the distance between two vectors [28], and for sketching image histograms [49]. In many of these applications, the weighted sketches empirically outperform unweighted sketches. Weighted MinHash sketches have also been used to compute general "sum aggregate" queries, for which the inner product is a special case [18]. However, we are not aware of strong worst-case error guarantees for the above applications, let alone for the problem of general inner product estimation. Consistent Weighted Sampling has also been used to approximate inner products in [35], albeit using a different estimator than in our work. However, non-asymptotic worst-case guarantees are not provided.

Locality Sensitive Hashing. Finally, our problem of estimating inner products from sketches is closely related to cosine similarity and maximum inner product search (MIPS), where the goal is to retrieve vectors from a database with the *highest* cosine similarity (respectively, inner product) with a given query vector. One approach for solving these problems is locality sensitive hashing [24], and there are methods based on both MinHash and random projections, like SimHash [11]. It has been observed that MinHash often outperforms SimHash for binary data, which parallels what was previously known for binary inner product estimation [50].

3 WARMUP: UNWEIGHTED MINHASH

Notation. We use bold letters to denote vectors, and for a vector $\mathbf{a}$, $\mathbf{a}[k]$ denotes the k^{th} entry (indexing starts with 1). For two length n vectors, $\mathbf{a}, \mathbf{b}$, $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{k=1}^{n} \mathbf{a}[k]\mathbf{b}[k]$ denotes the inner product. $\|\mathbf{a}\| = \sqrt{\langle \mathbf{a}, \mathbf{a} \rangle}$ denotes the Euclidean norm and $\|\mathbf{a}\|_\infty = \max_{k \in \{1,\ldots,n\}} |\mathbf{a}[k]|$ denotes the infinity norm. $\|\mathbf{a}\|_1 = \sum_{k=1}^{n} |\mathbf{a}[k]|$ denotes the ℓ_1 norm. As is standard in the literature [6], we assume access to uniformly random hash functions that map to the real line. I.e., we assume that we can construct a random function h such that for any input $j \in \{1, \ldots, n\}$, $h(j)$ is distributed uniformly and independently on the interval $[0, 1]$. In practice, h can be replaced with a low-randomness function that map to a sufficient large discrete set $\{1/U, 2/U \ldots, 1\}$. Typically U is chosen to equal n^c for constant c (e.g. $c = 3$) [19]. We let $\Pr[E]$ denote the probability that a random event E occurs, and $\mathbb{1}[E]$ is the indicator random variable that evaluates to 1 if E occurs and to 0 otherwise. $\mathbb{E}[X]$ and $\mathrm{Var}[X]$ denote the expectation and variance of a random variable X.

An unweighted method. Before introducing our Weighted MinHash sketching method, we review the unweighted MinHash algorithm and prove a inner product estimation bound that can be obtained from this method. The bound closely follows prior work on binary vectors [6, 44] and only holds under strong assumptions on the sketched vectors $\mathbf{a}$ and $\mathbf{b}$ – specifically that their entries are uniformly bounded in magnitude. Nevertheless, it serves as a warmup for our main result, which is proven using a similar strategy, but eliminates the assumption by using weighted sampling.

Given a vector $\mathbf{a}$, we obtain an entry in the standard MinHash sketch (see e.g., [8]) by hashing the index of every non-zero entry in $\mathbf{a}$ to the interval $[0, 1]$. We then store the smallest hash value. This process is repeated m times with independently chosen random hash functions. For binary vectors $\mathbf{a}$ and $\mathbf{b}$ with non-zero index sets $\mathcal{A} = \{k : \mathbf{a}[k] \neq 0\}$ and $\mathcal{B} = \{k : \mathbf{b}[k] \neq 0\}$, the minimum hash value alone can be used to estimate the Jaccard similarity $|\mathcal{A} \cap \mathcal{B}|/|\mathcal{A} \cup \mathcal{B}|$ or the union size $|\mathcal{A} \cup \mathcal{B}|$ [6, 23, 30].

For non-binary vectors, is it common to *augment* the standard MinHash sketch by also storing the *value* of the index with minimum hash value. This idea is used in "coordinated sampling" or "conditional random sampling" sketches [16, 18, 36], and was recently used to extend MinHash and the closely related k-minimum values (KMV) sketch to estimate vector correlations [47]. The basic augmented MinHash sketching method is shown in Algorithm 1, which returns $H_{\mathbf{a}}^{hash}$ and $H_{\mathbf{a}}^{val}$ as vectors of minimum hashes and their corresponding vector values, respectively.

For any single vector $\mathbf{a}$, the augmented MinHash sketch $H_{\mathbf{a}}$ contains a uniform subsample (collected with replacement) of the

Algorithm 1 Unweighted MinHash Sketch

Input: Length n vector $\mathbf{a}$, sample number m, random seed s.
Output: Sketch $H_{\mathbf{a}} = \{H_{\mathbf{a}}^{hash}, H_{\mathbf{a}}^{val}\}$, where $H_{\mathbf{a}}^{hash}$ and $H_{\mathbf{a}}^{val}$ have length m and contain values in $[0,1]$ and from $\mathbf{a}$, respectively

1: Initialize random number generator with seed s.
2: **for** i = 1, …, m **do**
3: Select uniformly random hash func. $h^i : \{1, ..., n\} \to [0,1]$.
4: Compute $j^* = \arg\min_{j \in \{1,...,n\}, \mathbf{a}[j] \neq 0} h^i(j)$.
5: Set $H_{\mathbf{a}}^{hash}[i] = h^i(j^*)$ and $H_{\mathbf{a}}^{val}[i] = \mathbf{a}[j^*]$
6: **end for**
7: **return** $\{H_{\mathbf{a}}^{hash}, H_{\mathbf{a}}^{val}\}$

Algorithm 2 Unweighted MinHash Estimate

Input: Sketches $H_{\mathbf{a}} = \{H_{\mathbf{a}}^{hash}, H_{\mathbf{a}}^{val}\}, H_{\mathbf{b}} = \{H_{\mathbf{b}}^{hash}, H_{\mathbf{b}}^{val}\}$ constructed using Algorithm 1 with the same inputs m, s.
Output: Estimate of $\langle \mathbf{a}, \mathbf{b} \rangle$.

1: Set $\tilde{U} = \frac{m}{\sum_{i=1}^{m} \min\left(H_{\mathbf{a}}^{hash}[i], H_{\mathbf{b}}^{hash}[i]\right)} - 1$
2: **return** $\frac{\tilde{U}}{m} \sum_{i=1}^{m} \mathbb{1}\left[H_{\mathbf{a}}^{hash}[i] = H_{\mathbf{b}}^{hash}[i]\right] \cdot H_{\mathbf{a}}^{val}[i] \cdot H_{\mathbf{b}}^{val}[i]$

non-zero values in $\mathbf{a}$. This is because for all $i \in \{1, \ldots, m\}$ the minimum value of the i^{th} hash is equally likely to come from any of the indices with non-zero value. More importantly, sketch can be used to obtain a uniform subsample from the *intersection* of $\mathbf{a}$ and $\mathbf{b}$, i.e., from entries where both vectors are non-zero. This subsample can in turn be used to estimate the sum $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{k=1}^{n} \mathbf{a}[k]\mathbf{b}[k]$, since $\mathbf{a}[k]\mathbf{b}[k]$ only contributes to the sum if $\mathbf{a}[k]$ and $\mathbf{b}[k]$ are both non-zero. Concretely, we have the following well-known fact:

Fact 3. *Consider vectors $\mathbf{a}$ and $\mathbf{b}$ sketched using Algorithm 1 to produce sketches $H_{\mathbf{a}}$ and $H_{\mathbf{b}}$. Define the sets $\mathcal{A} = \{i : \mathbf{a}[i] \neq 0\}$ and $\mathcal{B} = \{i : \mathbf{b}[i] \neq 0\}$. Then for all $i \in \{1, \ldots, m\}$ we have:*

(1) $H_{\mathbf{a}}^{hash}[i] = H_{\mathbf{b}}^{hash}[i]$ with probability $\frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|}$.
(2) If $H_{\mathbf{a}}^{hash}[i] = H_{\mathbf{b}}^{hash}[i]$, then $H_{\mathbf{a}}^{val}[i] = \mathbf{a}[j]$ and $H_{\mathbf{b}}^{val}[i] = \mathbf{b}[j]$ for j chosen uniformly at random from $\mathcal{A} \cap \mathcal{B}$.

Fact 3 indicates that, to obtain a uniform subsample from the intersection of $\mathbf{a}$ and $\mathbf{b}$, we can simply take all entries in $H_{\mathbf{a}}^{val}$ and $H_{\mathbf{b}}^{val}$ where the corresponding entries in $H_{\mathbf{a}}^{hash}$ and $H_{\mathbf{b}}^{hash}$ are equal – and, as per (1), they *will be equal* with good probability.

With Fact 3 in place, we describe an inner product estimator based on MinHash (Algorithm 2). This estimator will serve as a template for our weighted MinHash estimator in the next section.

Consider the summation in line 2 of Algorithm 2. Using linearity of expectation and Fact 3, we can compute the expectation:

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i=1}^{m} \mathbb{1}\left[H_{\mathbf{a}}^{hash}[i] = H_{\mathbf{b}}^{hash}[i]\right] \cdot H_{\mathbf{a}}^{val}[i] \cdot H_{\mathbf{b}}^{val}[i]\right] \\
&= m \cdot \mathbb{E}\left[\mathbb{1}\left[H_{\mathbf{a}}^{hash}[1] = H_{\mathbf{b}}^{hash}[1]\right] \cdot H_{\mathbf{a}}^{val}[1] \cdot H_{\mathbf{b}}^{val}[1]\right] \\
&= m \cdot \sum_{j \in \mathcal{A} \cap \mathcal{B}} \frac{1}{|\mathcal{A} \cup \mathcal{B}|} \mathbf{a}[j]\mathbf{b}[j] = \frac{m}{|\mathcal{A} \cup \mathcal{B}|} \cdot \langle \mathbf{a}, \mathbf{b} \rangle.
\end{aligned}$$

It follows from the above that, if we multiplied the summation $\sum_{i=1}^{m} \mathbb{1}\left[H_{\mathbf{a}}^{hash}[i] = H_{\mathbf{b}}^{hash}[i]\right] \cdot H_{\mathbf{a}}^{val}[i] \cdot H_{\mathbf{b}}^{val}[i]$ by $\frac{|\mathcal{A} \cup \mathcal{B}|}{m}$, then we would have an unbiased estimate for $\langle \mathbf{a}, \mathbf{b} \rangle$, as desired. The only catch is that we do not *know* $|\mathcal{A} \cup \mathcal{B}|$. This union size cannot be computed exactly from our sketches $H_{\mathbf{a}}$ and $H_{\mathbf{b}}$. However, it can be *estimated* using the same information contained in our MinHash sketches. In particular, since h^i hashes uniformly to $[0,1]$, $\frac{m}{\sum_{i=1}^{m} \min\left(H_{\mathbf{a}}^{hash}[i], H_{\mathbf{b}}^{hash}[i]\right)} - 1$ provides a good estimate for $|\mathcal{A} \cup \mathcal{B}|$. This is actually a standard variant of the well-known Flajolet-Martin distinct elements estimator [6, 23]. In Line 1 of Algorithm 2, we set $\tilde{U}$ equal to this estimator and we multiply by $\frac{\tilde{U}}{m}$ in Line 2 as a surrogate for $\frac{|\mathcal{A} \cup \mathcal{B}|}{m}$. This gives our final estimator for $\langle \mathbf{a}, \mathbf{b} \rangle$.

Overall, we are able to prove the following concentration bound for the estimator for computing the inner product between any pair of *bounded vectors*. For binary vectors, the constant c below equals 1 and we exactly recover the bounds from prior work [44].

Theorem 4 (Intermediate Result: Inner Product Sketching with Unweighted MinHash). *Let $\epsilon, \delta \in (0,1)$ be accuracy and failure probability parameters and let $m = O(\log(1/\delta)/\epsilon^2)$. There is an algorithm $\mathcal{S}$ that produces size-m sketches (Algorithm 1), along with an estimation procedure $\mathcal{F}$, such that for any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ with entries bounded in $[-c, c]$, with probability at least $1 - \delta$,*

$$|\mathcal{F}(\mathcal{S}(\mathbf{a}), \mathcal{S}(\mathbf{b})) - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \epsilon \cdot c^2 \cdot \sqrt{\max(|\mathcal{A}|, |\mathcal{B}|) \cdot |\mathcal{A} \cap \mathcal{B}|}$$

for $\mathcal{A} = \{i : \mathbf{a}[i] \neq 0\}$ and $\mathcal{B} = \{i : \mathbf{b}[i] \neq 0\}$.

The full proof of Theorem 4 is included in Appendix A.1. It requires two technical ingredients. First, we must bound the variance of an "ideal" estimator that uses the exact value of $|\mathcal{A} \cup \mathcal{B}|$. This can be done by using the fact that $\mathbf{a}$ and $\mathbf{b}$ have entries bounded in $[-c, c]$. Second, we can bound the error introduced by replacing $|\mathcal{A} \cup \mathcal{B}|$ with an estimate for the union, as discussed above. To do so, we rely on the following standard result, which shows that MinHash sketches for $\mathbf{a}$ and $\mathbf{b}$ can be used to compute a $(1 \pm \epsilon)$ relative error approximation to the true union $|\mathcal{A} \cup \mathcal{B}|$ when $m = O(1/\epsilon^2)$:

Lemma 1 (Union Size Estimator [7]). *Let $\mathcal{A}$ and $\mathcal{B}$ be non-empty subsets of $\{1, \ldots, n\}$ and let $h^1, \ldots, h^m : \{1, \ldots, n\} \to [0,1]$ be independent, uniform random hash functions. For any $\epsilon, \delta \in (0,1)$, if $m = O\left(\frac{1}{\delta\epsilon^2}\right)$, then with prob. at least $1 - \delta$, the estimator $\tilde{U} = \frac{m}{\sum_{i=1}^{m} \min_{j \in \mathcal{A} \cup \mathcal{B}} h^i(j)} - 1$ satisfies:*

$$(1 - \epsilon)|\mathcal{A} \cup \mathcal{B}| \leq \tilde{U} \leq (1 + \epsilon)|\mathcal{A} \cup \mathcal{B}|.$$

Note that, while it is written in a slightly different way, the $\tilde{U}$ in Lemma 1 is exactly equivalent to the $\tilde{U}$ in Algorithm 2 (when $\mathcal{A}$ and $\mathcal{B}$ contain the non-zero indices of $\mathbf{a}$ and $\mathbf{b}$). To see why this is the case, note that $H_{\mathbf{a}}^{hash}[i] = \min_{j \in \mathcal{A}} h^i(j)$ and $H_{\mathbf{b}}^{hash}[i] = \min_{j \in \mathcal{B}} h^i(j)$. So $\min\left(H_{\mathbf{a}}^{hash}[i], H_{\mathbf{b}}^{hash}[i]\right) = \min_{j \in \mathcal{A} \cup \mathcal{B}} h^i(j)$.

4 MAIN RESULT: WEIGHTED MINHASH

The main technical challenge in our work is extending the results of the previous section (Theorem 4) to vectors whose entries have *highly varying magnitude*. It is not hard to see that the simple MinHash method fails for such vectors. For example, consider the extreme case when $\mathbf{a}$ and $\mathbf{b}$ both contain a very large values at some index i, so large that the term $\mathbf{a}[i]\mathbf{b}[i]$ dominates the inner product $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{k=1}^{n} \mathbf{a}[k]\mathbf{b}[k]$. To correctly approximate the inner

Algorithm 3 Weighted MinHash Sketch

Input: Length n vector $\mathbf{a}$, sample number m, random seed s, integer discretization parameter L.

Output: Sketch $W_{\mathbf{a}} = \{W_{\mathbf{a}}^{hash}, W_{\mathbf{a}}^{val}, \|\mathbf{a}\|\}$, where $W_{\mathbf{a}}^{hash}$ is a length m vector of values in $[0,1]$, $W_{\mathbf{a}}^{val}$ is a length m vector containing a subset of entries from $\mathbf{a}$, and $\|\mathbf{a}\|$ is a scalar, the Euclidean norm of $\mathbf{a}$.

1: Initialize random number generator with seed s.
2: Set $\tilde{\mathbf{a}} = \text{ROUND}(\mathbf{a}/\|\mathbf{a}\|, \text{L})$ using Algorithm 4.
3: For each $i \in \{1,\ldots,n\}$, let $\bar{\mathbf{a}}^{(i)}$ be a length L vector whose first $\tilde{\mathbf{a}}[i]^2 \cdot L$ entries are set to $\tilde{\mathbf{a}}[i]$. Set the remaining entries to 0.
4: Let $\bar{\mathbf{a}} = [\bar{\mathbf{a}}^{(1)}, \ldots, \bar{\mathbf{a}}^{(n)}]$ be a length $n \cdot L$ vector obtained by concatenating the vectors defined above.
5: **for** i = 1, …, m **do**
6: Select uniform random hash func. $h^i : \{1, ..., nL\} \to [0,1]$.
7: Compute $j^* = \arg\min_{j \in \{1,\ldots,n\cdot L\}, \bar{\mathbf{a}}[j] \neq 0} h^i(j)$.
8: Set $W_{\mathbf{a}}^{hash}[i] = h^i(j^*)$ and $W_{\mathbf{a}}^{val}[i] = \bar{\mathbf{a}}[j^*]$.
9: **end for**
10: **return** $\{W_{\mathbf{a}}^{hash}, W_{\mathbf{a}}^{val}, \|\mathbf{a}\|\}$.

Algorithm 4 Vector Rounding for Weighted MinHash

Input: Length n unit vector $\mathbf{z}$, integer discretization parameter L.

Output: Length n unit vector $\tilde{\mathbf{z}}$ with $\tilde{\mathbf{z}}[i]^2$ an integer multiple of $1/L$ for all i.

1: For all $i \in \{1,\ldots,n\}$, $\tilde{\mathbf{z}}[i] = \text{sign}(\mathbf{z}[i]) \cdot \sqrt{\frac{\lfloor \mathbf{z}[i]^2 \cdot L \rfloor}{L}}$.
2: Let $i^* = \arg\max_{i \in 1,\ldots,n} |\mathbf{z}[i]|$.
3: Fix $\delta = 1 - \|\tilde{\mathbf{z}}\|^2$, then set $\tilde{\mathbf{z}}[i^*] = \text{sign}(\mathbf{z}[i^*]) \cdot \sqrt{\tilde{\mathbf{z}}[i^*]^2 + \delta}$.
4: **return** $\tilde{\mathbf{z}}$.

product, we *need* to include $\mathbf{a}[i]$ and $\mathbf{b}[i]$ in our sketches for $\mathbf{a}$ and $\mathbf{b}$, respectively. A MinHash sketch will only do so with low probability, since it uniformly samples entries from the intersection of the vectors. Thus, it will obtain a poor estimate for $\langle \mathbf{a}, \mathbf{b} \rangle$.

To address the issue with *heavy* entries, we modify the approach of Section 3 to incorporate *non-uniform* sampling weights using a Weighted MinHash sketch [41]. This allows us to sample high magnitude entries in the vectors with higher probability. Specifically, our goal is to sample the i^{th} entry of $\mathbf{a}$ with probability proportional to the squared magnitude, $\mathbf{a}[i]^2$. The Weighted MinHash sketch achieves non-uniform sampling in a simple way: we construct an *extended* vector $\bar{\mathbf{a}}$ which has the same entries as $\mathbf{a}$, but entries are repeated multiple times, with the exact number of repetitions proportional to their magnitude. We then apply the standard MinHash sketch to $\bar{\mathbf{a}}$. This approach is detailed in Algorithm 3.

Rounding & Normalization. While Weighted MinHash allows us to sample entries with non-uniform probability, another challenge arises: since sketches for $\mathbf{a}$ and $\mathbf{b}$ are computed independently, we no longer sample with the *same probability* from both vectors. For $\mathbf{b}$, Weighted MinHash samples indices with probability proportional to $\mathbf{b}[i]^2$ instead of $\mathbf{a}[i]^2$. This mismatch can actually *reduce* the probability that we select entries from $\mathbf{a}$ and $\mathbf{b}$ with the same index.

We are able to balance this issue with a normalization strategy. In particular, line 2 in Algorithm 3 performs a simple but important preprocessing step that *scales* and *rounds* $\mathbf{a}$ to a unit vector $\tilde{\mathbf{a}}$ whose squared entries are all integer multiples of $1/L$ for some large integer

Algorithm 5 Weighted MinHash Estimate

Input: Sketches $W_{\mathbf{a}} = \{W_{\mathbf{a}}^{hash}, W_{\mathbf{a}}^{val}, \|\mathbf{a}\|\}$ and $W_{\mathbf{b}} = \{W_{\mathbf{b}}^{hash}, W_{\mathbf{b}}^{val}, \|\mathbf{b}\|\}$ constructed using Algorithm 3 with the same inputs m, s, and L.

Output: Estimate of $\langle \mathbf{a}, \mathbf{b} \rangle$.

1: For $i \in \{1,\ldots,m\}$, set $q_i = \min\left(W_{\mathbf{a}}^{val}[i]^2, W_{\mathbf{b}}^{val}[i]^2\right)$.
2: Set $\tilde{M} = \frac{1}{L} \cdot \left(\frac{m}{\sum_{i=1}^m \min\left(W_{\mathbf{a}}^{hash}[i], W_{\mathbf{b}}^{hash}[i]\right)} - 1\right)$.
3: Set $I = \frac{\tilde{M}}{m} \sum_{i=1}^m \mathbb{1}\left[W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]\right] \cdot \frac{W_{\mathbf{a}}^{val}[i] \cdot W_{\mathbf{b}}^{val}[i]}{q_i}$.
4: **return** $\|\mathbf{a}\|\|\mathbf{b}\| \cdot I$

L (to be chosen later). The rounding handles a minor issue: since we control the frequency with which each entry $\mathbf{a}[i]$ is sampled by *repetition*, we need the squared value of all entries to be integer multiples of the same fixed constant in order to sample precisely with probability proportional to $\mathbf{a}[i]^2$. As will be proven, L can be chosen so that the discretization has little impact on the accuracy of our final inner product estimate, and the parameter also has no impact on the size of the sketch returned by Algorithm 1.[4]

The scaling is what deals with the bigger issue discussed above, which is the mismatch in sampling probabilities between $\mathbf{a}$ and $\mathbf{b}$. Surprisingly, we can show that the impact of this mismatch can be controlled when $\|\mathbf{a}\| = \|\mathbf{b}\|$. So while it is possible to come up with examples where the algorithm fails if we directly sketch $\mathbf{a}$ and $\mathbf{b}$, we can obtain a worst-case bound by sketching $\mathbf{a}/\|\mathbf{a}\|$ and $\mathbf{b}/\|\mathbf{b}\|$, approximating $\langle \mathbf{a}/\|\mathbf{a}\|, \mathbf{b}/\|\mathbf{b}\| \rangle$, and then post-multiplying the result by $\|\mathbf{a}\|\|\mathbf{b}\|$ to get our final estimator.

Deriving the Inner Product Estimator. We next motivate Algorithm 2, which is the algorithm used to estimate $\langle \mathbf{a}, \mathbf{b} \rangle$ from our sketches. Note that Weighted MinHash Sketch (Algorithm 3) in fact returns an Unweighted MinHash Sketch (Algorithm 1) for the expanded vectors $\bar{\mathbf{a}}, \bar{\mathbf{b}}$. So, we can apply Fact 3 to obtain the following:

FACT 5. *Consider vectors* $\mathbf{a}$ *and* $\mathbf{b}$ *sketched using Algorithm 3 to produce* $W_{\mathbf{a}}$ *and* $W_{\mathbf{b}}$. *Define* $\mathcal{A}$ *and* $\mathcal{B}$ *as in Fact 3. For all* $i \in \{1,\ldots,m\}$ *we have:*

(1) $W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]$ *with probability equal to the weighted Jaccard similarity,* $\bar{J} = \frac{\sum_{j=1}^n \min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[J]^2)}{\sum_{j=1}^n \max(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)}$.

(2) If $W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]$, *then we have that* $W_{\mathbf{a}}^{val} = \tilde{\mathbf{a}}[j]$ *and* $W_{\mathbf{b}}^{val} = \tilde{\mathbf{b}}[j]$ *for* j *chosen from* $\mathcal{A} \cap \mathcal{B}$ *with probability equal to* $\min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2) / \sum_{i=1}^n \max(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)$.

A proof of Fact 5 is given in Appendix A.2. With the statement in place, we present our procedure for estimating $\langle \mathbf{a}, \mathbf{b} \rangle$ based the sketches computed by Algorithm 3. This procedure, shown in Algorithm 5, is reminiscent of our estimator for unweighted sketches from the previous section. The only difference is that, since we are sampling with non-uniform probabilities, we need to inversely weight samples in our sum to keep everything correct in expectation. In particular, consider the sum in line 3 of the algorithm.

[4] Note that our rounding method (Algorithm 4) is non-standard: It rounds all entries of the input vector down to smaller magnitude values, except for the *largest* magnitude entry in the vector, which gets rounded up. This scheme allows us to achieve small *relative error* when rounding and to avoid additive error depending on $1/L$.

By Fact 5 and linearity of expectation, we have that:

$$\mathbb{E}\left[\sum_{i=1}^{m} \mathbb{1}\left[W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]\right] \cdot \frac{W_{\mathbf{a}}^{val}[i] \cdot W_{\mathbf{b}}^{val}[i]}{q_i}\right]$$

$$m \cdot \mathbb{E}\left[\mathbb{1}\left[W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]\right]\right] \cdot \frac{W_{\mathbf{a}}^{val}[i] \cdot W_{\mathbf{b}}^{val}[i]}{q_i}$$

$$= m \cdot \sum_{j \in \mathcal{A} \cap \mathcal{B}} \frac{q_j}{\sum_{i=1}^{n} \max(\tilde{\mathbf{a}}[i]^2, \tilde{\mathbf{b}}[i]^2)} \frac{\tilde{\mathbf{a}}[j]\tilde{\mathbf{b}}[j]}{q_j}$$

$$= \frac{m}{\sum_{i=1}^{n} \max(\tilde{\mathbf{a}}[i]^2, \tilde{\mathbf{b}}[i]^2)} \cdot \langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle.$$

So, we have obtained an estimator that in expectation is equal to $\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle$, multiplied by m over a term $M = \sum_{i=1}^{n} \max(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)$. This term M is referred to as the *weighted* union size between the vectors. We can multiply by $\frac{M}{m}$ to obtain an unbiased estimator for $\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle$. Since $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{b}}$ were obtained by scaling $\mathbf{a}$ and $\mathbf{b}$ inversely by their Euclidean norms (ignoring the effect of rounding for now), our final estimator in Line 4 of Algorithm 5 multiplies by $\|\mathbf{a}\|\|\mathbf{b}\|$. The values of $\|\mathbf{a}\|$ and $\|\mathbf{b}\|$ are stored explicitly in the sketches for $\mathbf{a}$ and $\mathbf{b}$, respectively (as just one extra number per sketch).

The formal analysis of Algorithm 5, which yields Theorem 2, is included in Appendix A.2. It contains three parts. First, when analyzing the unweighted estimator, we do not know M exactly, so must estimate it. We can take advantage of the fact that M is exactly equal to the *unweighted* union size $|\bar{\mathcal{A}} \cup \bar{\mathcal{B}}|$ between the non-zero index sets $\bar{\mathcal{A}}$ and $\bar{\mathcal{B}}$ of the *expanded vectors* $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ constructed in Algorithm 3. We can apply Lemma 1 directly to obtain an estimator, which is denoted as $\tilde{M}$ in Algorithm 5. Second, we need to analyze the variance of the sum $\sum_{i=1}^{m} \mathbb{1}\left[W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]\right] \cdot \frac{W_{\mathbf{a}}^{val}[i] \cdot W_{\mathbf{b}}^{val}[i]}{q_i}$. This analysis uses the fact that $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{b}}$ are unit vectors. Third, we need to rigorously analyze the impact of the rounding procedure performed in Line 2 of Algorithm 3 to establish that a good estimate for $\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle$ actually yields a good estimate for $\langle \mathbf{a}/\|\mathbf{a}\|, \mathbf{b}/\|\mathbf{b}\| \rangle = \frac{1}{\|\mathbf{a}\|\|\mathbf{b}\|} \langle \mathbf{a}, \mathbf{b} \rangle$.

We conclude by noting that our final analysis of Algorithm 5 requires setting L to be on the order of n^6/ϵ^2 when sketching using Algorithm 3. This may sound large, but note that the parameter has *no impact* on the size of the sketches returned by Algorithm 3, or on the runtime of our estimation procedure Algorithm 5. L does impact the runtime of Algorithm 3, but as discussed in Section 5, prior work can be used to implement the Weighted MinHash sketching method so that it has a *logarithmic* dependence on L – i.e., on $O(\log(n/\epsilon))$.

5 EXPERIMENTS

To support the results presented in Section 4, we performed an experimental evaluation using synthetic data and real-world datasets.

Baselines. We compare our Weighted MinHash approach against 4 baseline methods, 2 linear and 2 sampling-based, with the goal of evaluating the trade-off between sketch size and accuracy in estimating inner products. Those methods are:

Johnson-Lindenstrauss Projection (JL): equivalent to the AMS sketch [1, 4]. Uses a random matrix $\mathbf{\Pi}$ with scaled ±1 entries (Fact 1).

CountSketch (CS): classic linear sketch introduced in [12], and corresponds to multiplication with a $\mathbf{\Pi}$ that has sparse random entries. We follow the implementation in [33], using 5 repetitions of the sketch and taking the median to improve performance.

MinHash Sampling (MH): method described in Algorithm 1; we use a single sketch without any median estimate.

k-Minimum Values Sampling (KMV): sampling-based sketch closely related to MinHash, but it draws samples from the vector being sketched *without replacement*. It can also be used to estimate union size. We follow the implementations from [6] and [47].

Weighted MinHash Sampling (WMH): our method described in Algorithm 3; we use a single sketch without any median estimate.

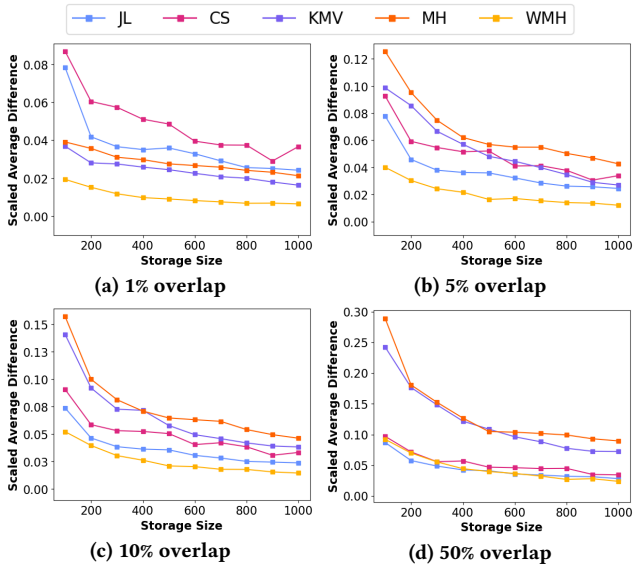
Storage Size. For linear sketches, we store the output of the matrix multiplication $\mathbf{\Pi}\mathbf{a}$ as 64-bit doubles. We also store $W_{\mathbf{a}}^{val}$ and $H_{\mathbf{a}}^{val}$ as 64-bit doubles. Since sampling-based sketches need to store hash values (which in our case are 32-bit ints), a sampling-based sketch with m samples takes $1.5x$ as much space as a JL sketch with m rows. In our experiments, we plot *storage size* which denotes the total number of bits in the sketch divided by 64, i.e., the total number of 64-bit doubles (or equivalent) used in the sketch. Standard quantization tricks could likely be used to reduce the size of numbers in all sketches (linear and sampling), but we leave the development of such methods to future work. As a starting point, we note that there has already been interesting work on quantized JL projections [29, 38], and the SimHash method for estimating cosine similarity can be viewed as a "1-bit" quantization of a JL sketch [11].

Estimation Error. For all plots, we report the absolute difference between $\langle \mathbf{a}, \mathbf{b} \rangle$ and the estimate, divided by $\|\mathbf{a}\|\|\mathbf{b}\|$. This is the term appearing on the right-hand side of the accuracy guarantee for linear sketches Fact 1, so this scaling roughly ensures that errors are between 0 and 1, making it easier to compare across different datasets. We always report average error over 10 independent trials.

Choice of L. Note that the choice of L in Algorithm 3 does not impact the size of our final sketch, so in general, it should be set as large as possible. Our bounds from Lemma 3 that suggest L should be set $\geq n^6$ are likely loose (we did not attempt to optimize polynomial factors), but we did find that it *is necessary* to at least ensure that $L > n$. Ideally it should be larger by a multiplicative factor 100 or 1000. The reason for this is that, if $\mathbf{a}$ is dense and is normalized to have unit norm, as in Algorithm 1, *most* of its entries could have squared value $< 1/n$ (as the average value of a squared entry in a unit norm vector is always $1/n$). If we set $L < 1/n$, then any entries with value $< 1/n$ would get rounded to 0, which could negatively impact the accuracy of an inner product estimate.

Efficient Weighted Hashing. When L is large, a naive implementation of Algorithm 3 would be prohibitively slow. The "extended" vector $\bar{\mathbf{a}}$ has length $n \cdot L$ and we must apply a hash function to every non-zero entry in that vector. Let $\mathcal{A} = \{i : \mathbf{a}[i] \neq 0\}$ as before, so $|\mathcal{A}|$ is equal to the number of non-zero values in $\mathbf{a}$. If each hash computation is considered unit cost, this amounts to a runtime of $O(|\mathcal{A}|m \cdot L)$, which is too large, since L is chosen larger than n.

Fortunately, it is possible to improve this cost to $O(|\mathcal{A}|m \cdot \log L) = O(|\mathcal{A}|m \cdot \log n)$ using techniques for speeding up weighted MinHash sketches. Such techniques have been heavily studied in recent years [26, 28, 49, 53]. The savings are significant, reducing the computation cost of sketching to nearly-linear in the size of the input for each of our m samples. Among faster methods, we specifically employ the simple "active index" technique, which was first introduced in [25]. The rough idea is that, when hashing non-zero entries in a particular length L block of $\bar{\mathbf{a}}$, there is no need to hash



(a) 1% overlap (b) 5% overlap

(c) 10% overlap (d) 50% overlap

Figure 3: Inner product estimation (synthetic data).

all non-zero indices in that block. We can skip over large sections of indices by observing that if z is the minimum hash value generated so far, the next index where a lower hash value will be seen is a distributed as a *geometric random variable* with parameter z. We can sample from the geometric distribution efficiently (e.g. using a built-in Python routine) and skip ahead to that index. It is possible to prove that the expected cost of this approach is just $O(\log L)$ per block. See the exposition in [41] for further details.

Since initially releasing this paper, we became aware of even faster implementations of weighted MinHash that reduce the runtime to $O(|\mathcal{A}| + m \log m)$, which is nearly linear in the number of non-zeros in the vector being sketched [14, 21]. Such methods should be able to be adapted for use in our inner product sketching application, although we leave further exploration to future work.

Choice of Hash Function. In practice we cannot obtain a truly uniform random hash function from $\{1, \ldots, n\}$ to the reals, so we must use an approximation. In our experiments, we employ a standard 2-wise independent hash function (linear function with random coefficients) that maps from $\{1, \ldots, n\}$ to $\{1, \ldots, p\}$ for a 31-bit prime p [10]. [5] We then use as our hash value $h(i)/p$, which is a number between 0 and 1. Since p is chosen to have 31 bits, we can store the value of $h(i)$ in our sketch using a standard 32-bit int.

5.1 Synthetic Data

We begin with an evaluation of our approach using synthetic data. We generate length 10000 vectors **a** and **b**, each with 2000 non-zero entries. The ratio of non-zero entries that *overlap*, i.e., are non-zero in both **a** and **b**, is adjusted to simulate different practical settings with different levels of *joinability* between tables (see Section 1.2). The non-zero entries in **a** and **b** are normal random variables with values between −1 and 1, except 10% of entries are chosen randomly as outliers and set to random values between 20 and 30.

[5] Our choice to use a 2-wise independent hash function was based on prior implementations of the weighted MinHash method [52] that do so.

	Kurtosis>0	Kurtosis>10	Kurtosis>50
Overlap<0.05	-0.018	-0.019	-0.018
Overlap<0.1	-0.013	-0.014	-0.014
Overlap<0.25	-0.006	-0.007	-0.010
Overlap<0.5	0.000	-0.001	-0.006
Overlap<0.75	0.005	0.003	-0.003
Overlap<1.0	0.006	0.004	-0.003

(a) WMH estimation error minus JL estimation error.

	Kurtosis>0	Kurtosis>10	Kurtosis>50
Overlap<0.05	-0.012	-0.015	-0.016
Overlap<0.1	-0.009	-0.012	-0.013
Overlap<0.25	-0.004	-0.007	-0.008
Overlap<0.5	-0.008	-0.013	-0.016
Overlap<0.75	-0.014	-0.020	-0.020
Overlap<1.0	-0.020	-0.029	-0.031

(b) WMH estimation error minus MH estimation error.

Figure 4: Inner product estimation (World Bank data). Different shades of blue highlight combinations for which WMH outperforms the other methods.

Results for varying amounts of overlap are reported in Figure 3. They closely align with our theoretical findings: when the overlap is small, the bounds for Weighted MinHash are significantly better than those of linear sketching methods. Accordingly, WMH outperforms all other methods for overlap ratio $\leq 10\%$. Note that unweighted sampling based sketches also outperform linear sketches for very low overlap (1%). But as the overlap increases, the advantage brought about by Theorem 2 over Fact 1 decreases. We can see this in Figure 3(d): at 50% overlap, the performance of linear sketching is comparable to that of Weighted MinHash.

5.2 Real-World Data

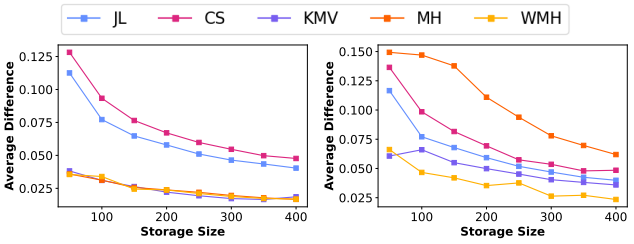
Assessing the Effect of Overlap and Outliers. Using sketches of size 400,[6] we estimate the inner product between 5000 pairs of numerical columns from 56 datasets published by the World Bank Group [51]. We normalize columns to have norm 1 so that all inner products have magnitude less than 1. We visual results using a winning table in Figure 4, filting vector pairs based on different overlap ratios (column) and kurtosis values, a measure of outliers (row). Each cell shows the average error difference (WMH estimation error minus the error of other method) for vector pairs with the specified overlap and kurtosis values.

The blue cells (negative difference) correspond to combinations in which WMH outperforms the other methods, while the red cells (positive difference) represent combinations in which the other methods win. The darker the cells, the bigger the difference. A high kurtosis often indicates the presence of outliers, which will, based on our theoretical results, present a difficulty for unweighted sampling methods like MH in comparison to JL or our WMH method. This is supported by the experiments, which show that WMH has a great improvement over MH when kurtosis is high (up to -.031 vs. at most -.020 when kurtosis is low). As predicted by Theorem 2 and shown in our synthetic experiments, WMH also has a great edge over JL for low overlap values. For large overlaps (greater than .75), JL leads to slightly smaller errors (from 0.003 to 0.006).

This suggests that *WMH provides a good compromise for applications in which the distribution of data is unknown: it provides much better estimates for many cases, and when it does not, its estimates are comparable to the best results from existing sketching methods.*

Document Similarity Estimation. We also evaluated the performance of WMH sketches for text similarity estimation using the *20*

[6] The size was chosen empirically. Our goal here is to simulate the real-world situation where a fixed parameter must be selected for a given application.



(a) All Documents (b) Documents > 700 words

Figure 5: Text similarity estimation (20 Newsgroups dataset). Note that in the left plot, the lines for MH, WMH, and KMV all lie essentially on top of one another.

newsgroups dataset [43]. We represent each document as a vector in which each entry represents a term or a combination of 2 terms (bigrams), and is associated with a value that encodes term/bigram importance using TF-IDF weights [46]. This setting is well-known for generating sparse vectors of very high dimension. As a similarity measure, we use the *cosine*, which is equal to an inner product when the vectors have are normalized. We sampled 700 documents and estimated the cosine similarity for over 200,000 pairs of documents. The results in Fig. 5 show that, similar to previous experiments, *in the worst case, the accuracy of WMH is comparable to the other methods, but it can sometimes be better by a large margin.* In this case, it performs better for documents containing more than 700 words. Note that linear projection sketches have poor performance for small sketches even when the documents are small, whereas our sampling-based methods are able to obtain significantly better accuracy for the same storage budget. Finally, also note that the Unweighted MinHash (MH) performs poorly for long documents whereas the weighted version still performs well.

Acknowledgements. This work was supported by the DARPA D3M program and NSF awards ISS-2106888 and CCF-2046235. Aline Bessa was supported by a 2021 CRA/CCC CIFellows Award. Cameron Musco was also supported by a Google Research Scholar Award. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF, DARPA, or other funding organizations.

REFERENCES

[1] Dimitris Achlioptas. 2003. Database-friendly Random Projections: Johnson-Lindenstrauss with Binary Coins. *J. Comput. Syst. Sci.* 66, 4 (2003), 671–687.
[2] Noga Alon, Phillip B. Gibbons, Yossi Matias, and Mario Szegedy. 1999. Tracking Join and Self-Join Sizes in Limited Storage. In *Proceedings of the 18th Symposium on Principles of Database Systems (PODS)*.
[3] Noga Alon and Bo'az Klartag. 2017. Optimal Compression of Approximate Inner Products and Dimension Reduction. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 639–650.
[4] Noga Alon, Yossi Matias, and Mario Szegedy. 1999. The Space Complexity of Approximating the Frequency Moments. *J. Comput. System Sci.* 58, 1 (1999).
[5] Rosa I. Arriaga and Santosh Vempala. 2006. An algorithmic theory of learning: Robust concepts and random projection. *Machine Learning* 63, 2 (2006), 161–182.
[6] Kevin Beyer, Peter J. Haas, Berthold Reinwald, Yannis Sismanis, and Rainer Gemulla. 2007. On Synopses for Distinct-Value Estimation under Multiset Operations. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*. 199–210.
[7] Avrim Blum, John Hopcroft, and Ravindran Kannan. 2020. *Foundations of Data Science*. Cambridge University Press.
[8] A.Z. Broder. 1997. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997*. 21–29.
[9] Andrei Z. Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher. 1998. Min-Wise Independent Permutations (Extended Abstract). In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC)*. 327–336.
[10] J. Lawrence Carter and Mark N. Wegman. 1979. Universal classes of hash functions. *J. Comput. System Sci.* 18, 2 (1979), 143–154.
[11] Moses Charikar. 2002. Similarity Estimation Techniques from Rounding Algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC)*. 380–388.
[12] Moses Charikar, Kevin Chen, and Martin Farach-Colton. 2002. Finding Frequent Items in Data Streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP)*. 693–703.
[13] Lianhua Chi and Xingquan Zhu. 2017. Hashing Techniques: A Survey and Taxonomy. *ACM Comput. Surv.* 50, 1 (2017).
[14] Tobias Christiani. 2020. DartMinHash: Fast Sketching for Weighted Sets. *arXiv:2005.11547* (2020).
[15] City of New York. 2022. Open Data NYC. https://opendata.cityofnewyork.us/.
[16] Edith Cohen. 2016. *Min-Hash Sketches*. Springer New York, New York, NY, 1282–1287.
[17] Edith Cohen and Haim Kaplan. 2007. Summarizing Data Using Bottom-k Sketches. In *Proceedings of the 2007 ACM Symposium on Principles of Distributed Computing (PODC)*. 225–234.
[18] Edith Cohen and Haim Kaplan. 2013. What You Can Do with Coordinated Samples. In *Proceedings of the 16th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*. 452–467.
[19] Graham Cormode, Minos Garofalakis, Peter Haas, and Chris Jermaine. 2011. *Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches*. NOW publishers.
[20] Sanjoy Dasgupta and Anupam Gupta. 2003. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms* 22, 1 (2003), 60–65.
[21] Otmar Ertl. 2018. BagMinHash - Minwise Hashing Algorithm for Weighted Sets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1368–1377.
[22] Raul Castro Fernandez, Jisoo Min, Demitri Nava, and Samuel Madden. 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)*. 1190–1201.
[23] Philippe Flajolet and G. Nigel Martin. 1985. Probabilistic Counting Algorithms for Data Base Applications. *J. Comput. Syst. Sci.* 31, 2 (1985), 182–209.
[24] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity Search in High Dimensions via Hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*. 518–529.
[25] Sreenivas Gollapudi and Rina Panigrahy. 2006. Exploiting Asymmetry in Hierarchical Topic Extraction. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM)*. 475–482.
[26] Bernhard Haeupler, Mark Manasse, and Kunal Talwar. 2014. Consistent Weighted Sampling Made Fast, Small, and Easy. *arXiv:1410.4266* (2014).
[27] Nevin Heintze. 1996. Scalable Document Fingerprinting. In *USENIX Workshop on Electronic Commerce*.
[28] Sergey Ioffe. 2010. Improved Consistent Sampling, Weighted Minhash and L1 Sketching. In *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM)*. 246–255.
[29] Laurent Jacques. 2015. A Quantized Johnson–Lindenstrauss Lemma: The Finding of Buffon's Needle. *IEEE Trans. Inf.* 61, 9 (2015), 5012–5027.
[30] Daniel M. Kane, Jelani Nelson, and David P. Woodruff. 2010. An Optimal Algorithm for the Distinct Elements Problem. In *Proceedings of the 29th Symposium on Principles of Database Systems (PODS)*. 41–52.
[31] James Max Kanter and Kalyan Veeramachaneni. 2015. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE international conference on data science and advanced analytics (DSAA)*. IEEE, 1–10.
[32] Kasper Green Larsen and Jelani Nelson. 2017. Optimality of the Johnson-Lindenstrauss Lemma. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 633–638.
[33] Kasper Green Larsen, Rasmus Pagh, and Jakub Tětek. 2021. CountSketches, Feature Hashing and the Median of Three. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*. 6011–6020.
[34] Oliver Lehmberg, Dominique Ritze, Petar Ristoski, Robert Meusel, Heiko Paulheim, and Christian Bizer. 2015. The Mannheim Search Join Engine. *Journal of Web Semantics* 35 (2015), 159 – 166.
[35] Ping Li. 2017. Linearized GMM Kernels and Normalized Random Fourier Features. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 315–324.
[36] Ping Li, Kenneth Church, and Trevor Hastie. 2006. Conditional Random Sampling: A Sketch-based Sampling Technique for Sparse Data. In *Advances in Neural Information Processing Systems 19 (NeurIPS)*, Vol. 19.
[37] Ping Li and Arnd Christian König. 2010. b-Bit Minwise Hashing. In *Proceedings of the 19th International World Wide Web Conference (WWW)*.
[38] Ping Li, Michael Mitzenmacher, and Martin Slawski. 2016. Quantized Random Projections and Non-Linear Estimation of Cosine Similarity. In *Advances in Neural Information Processing Systems 29 (NeurIPS)*, Vol. 29.
[39] Ping Li, Art Owen, and Cun-hui Zhang. 2012. One Permutation Hashing. In *Advances in Neural Information Processing Systems 25 (NeurIPS)*.

[40] Ping Li, Anshumali Shrivastava, Joshua Moore, and Arnd König. 2011. Hashing algorithms for large-scale learning. *Advances in Neural Information Processing Systems 24 (NeurIPS)* 24 (2011).
[41] Mark Manasse, Frank McSherry, and Kunal Talwar. 2010. *Consistent Weighted Sampling*. Technical Report MSR-TR-2010-73. https://www.microsoft.com/en-us/research/publication/consistent-weighted-sampling/
[42] Udi Manber. 1994. Finding Similar Files in a Large File System. In *USENIX Winter 1994 Technical Conference*.
[43] Tom Mitchell. 1997. 20 Newsgroups Dataset. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html.
[44] Rasmus Pagh, Morten Stöckel, and David P. Woodruff. 2014. Is Min-Wise Hashing Optimal for Summarizing Set Intersection?. In *Proceedings of the 33rd Symposium on Principles of Database Systems (PODS)*. 109–120.
[45] Florin Rusu and Alin Dobra. 2008. Sketches for size of join estimation. *ACM Transactions on Database Systems (TODS)* 33, 3 (2008), 1–46.
[46] Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Commun. ACM* 18, 11 (1975), 613–620.
[47] Aécio Santos, Aline Bessa, Fernando Chirigati, Christopher Musco, and Juliana Freire. 2021. Correlation Sketches for Approximate Join-Correlation Queries. In *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*. 199–210.
[48] Aécio Santos, Aline Bessa, Christopher Musco, and Juliana Freire. 2022. A sketch-based index for correlated dataset search. In *Proceedings of the 38th IEEE International Conference on Data Engineering (ICDE)*. IEEE, 2928–2941.
[49] Anshumali Shrivastava. 2016. Simple and Efficient Weighted Minwise Hashing. In *Advances in Neural Information Processing Systems 29 (NeurIPS)*. 1506–1514.
[50] Anshumali Shrivastava and Ping Li. 2014. In Defense of Minhash over Simhash. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS)*.
[51] World Bank. 2022. World Bank Group Finances. https://finances.worldbank.org/
[52] Wei Wu, Bin Li, Ling Chen, Junbin Gao, and Chengqi Zhang. 2020. A Review for Weighted MinHash Algorithms. *IEEE Trans. Knowl. Data Eng.* (2020), 1–1.
[53] Wei Wu, Bin Li, Ling Chen, Chengqi Zhang, and Philip S. Yu. 2019. Improved Consistent Weighted Sampling Revisited. *IEEE Trans. Knowl. Data Eng.* 31, 12 (2019), 2332–2345.
[54] Yang Yang, Ying Zhang, Wenjie Zhang, and Zengfeng Huang. 2019. Gb-kmv: An augmented kmv sketch for approximate containment similarity search. In *Proceedings of the 35th IEEE International Conference on Data Engineering (ICDE)*. IEEE, 458–469.
[55] Erkang Zhu, Dong Deng, Fatemeh Nargesian, and Renée J. Miller. 2019. JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes. In *Proceedings of the 2019 ACM SIGMOD International Conference on Management of Data*. 847–864.
[56] Erkang Zhu, Fatemeh Nargesian, Ken Q Pu, and Renée J. Miller. 2016. LSH Ensemble: Internet-Scale Domain Search. *Proceedings of the VLDB Endowment* 9, 12 (2016).

A ADDITIONAL PROOFS

A.1 Unweighted MinHash Analysis

In this section, we give a full proof of Theorem 4.

Proof of Theorem 4. Let $\overline{\mathcal{F}}(H_{\mathbf{a}}, H_{\mathbf{b}})$ denote the estimator from Algorithm 2. Ultimately we will set $\mathcal{F}$ in Theorem 4 to be $\overline{\mathcal{F}}$, but repeated $O(\log(1/\delta)$ times to obtain failure probability $1-\delta$.

We focus on showing first that $\overline{\mathcal{F}}(H_{\mathbf{a}}, H_{\mathbf{b}})$ achieves error $\epsilon \cdot c^2 \cdot \sqrt{\max(|\mathcal{A}|, |\mathcal{B}|) \cdot |\mathcal{A} \cap \mathcal{B}|}$ with probability $\geq 2/3$. To prove this, let $\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}})$ be an alternative idealized estimator where we replace $\tilde{U}$ in line 1 of Algorithm 2 with the true union size $U = |\mathcal{A} \cup \mathcal{B}|$:

$$\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}}) = \frac{U}{m} \sum_{i=1}^{m} \mathbb{1}\left[H_{\mathbf{a}}^{hash}[i] = H_{\mathbf{b}}^{hash}[i]\right] \cdot H_{\mathbf{a}}^{val}[i] \cdot H_{\mathbf{b}}^{val}[i].$$

We will first analyze $\mathcal{F}^*$, before showing that $\overline{\mathcal{F}}$ obtains essentially as good of an estimate. As established in Section 3, using the properties of Fact 3, we have that

$$\mathbb{E}\left[\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}})\right] = U \cdot \frac{1}{|\mathcal{A} \cup \mathcal{B}|} \cdot \langle \mathbf{a}, \mathbf{b} \rangle = \langle \mathbf{a}, \mathbf{b} \rangle.$$

So we turn to bounding the variance of the estimator. Define the random variable $Z_i = \mathbb{1}\left[H_{\mathbf{a}}^{hash}[i] = H_{\mathbf{b}}^{hash}[i]\right] \cdot H_{\mathbf{a}}^{val}[i] \cdot H_{\mathbf{b}}^{val}[i]$ and note that $\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}}) = \frac{U}{m}\sum_{i=1}^{m} Z_i$. From Fact 3 we have:

$$Z_i = \begin{cases} 0 & \text{with probability } 1 - \frac{|\mathcal{A}\cap\mathcal{B}|}{|\mathcal{A}\cup\mathcal{B}|} \\ \mathbf{a}[j]\mathbf{b}[j] & \text{with probability } \frac{1}{|\mathcal{A}\cup\mathcal{B}|} \text{ for all } j \in \mathcal{A}\cap\mathcal{B}. \end{cases}$$

Since each Z_i is independent, we can bound:

$$\text{Var}\left[\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}})\right] = \frac{U^2}{m^2} \sum_{i=1}^{m} \text{Var}\left[Z_i\right].$$

Using our assumption that $\mathbf{a}[k], \mathbf{b}[k] \leq c$ for all k, we have

$$\text{Var}\left[Z_i\right] \leq \mathbb{E}\left[Z_i^2\right] = \sum_{j \in \mathcal{A}\cap\mathcal{B}} \frac{1}{|\mathcal{A}\cup\mathcal{B}|} \cdot \mathbf{a}[j]^2\mathbf{b}[j]^2 \leq c^4 \cdot \frac{|\mathcal{A}\cap\mathcal{B}|}{|\mathcal{A}\cup\mathcal{B}|},$$

for all Z_i. So we conclude that $\text{Var}\left[\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}})\right] \leq \frac{1}{m} \cdot c^4 \cdot |\mathcal{A}\cap\mathcal{B}||\mathcal{A}\cup\mathcal{B}|$. We then plug our expectation and variance bounds into Chebyhev's inequality. If $m = O(1/\epsilon^2)$, we conclude that with probability $\geq 5/6$,

$$\left|\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}}) - \langle \mathbf{a}, \mathbf{b} \rangle\right| \leq \epsilon \cdot c^2 \sqrt{|\mathcal{A}\cap\mathcal{B}||\mathcal{A}\cup\mathcal{B}|}. \tag{1}$$

The proof is almost complete; we just need to extend this bound to the non-idealized estimator $\overline{\mathcal{F}} = \frac{\tilde{U}}{U} \cdot \mathcal{F}^*$. We do so by observing that $\tilde{U}$ is a good approximation to U. Specifically, by Lemma 1 applied with $\delta = 1/6$, we have that, when $m = O(1/\epsilon^2)$, $(1-\epsilon)U \leq \tilde{U} \leq (1+\epsilon)U$, with probability $\geq 5/6$. It follows that

$$(1-\epsilon)\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}}) \leq \overline{\mathcal{F}}(H_{\mathbf{a}}, H_{\mathbf{b}}) \leq (1+\epsilon)\mathcal{F}^*(H_{\mathbf{a}}, H_{\mathbf{b}}). \tag{2}$$

By a union bound, with probability at least 2/3, both (1) and (2) hold simultaneously. Finally, by triangle inequality and the fact that $\langle a, b \rangle \leq c^2|\mathcal{A}\cap\mathcal{B}| \leq c^2\sqrt{|\mathcal{A}\cap\mathcal{B}||\mathcal{A}\cup\mathcal{B}|}$ it follows that:

$$|\overline{\mathcal{F}}(H_{\mathbf{a}}, H_{\mathbf{b}}) - \langle \mathbf{a}, \mathbf{b} \rangle| \leq 3\epsilon \cdot c^2 \cdot \sqrt{|\mathcal{A}\cap\mathcal{B}||\mathcal{A}\cup\mathcal{B}|}.$$

Noting that $|\mathcal{A}\cap\mathcal{B}||\mathcal{A}\cup\mathcal{B}| \leq 2\max(|\mathcal{A}|, |\mathcal{B}|) \cdot |\mathcal{A}\cap\mathcal{B}|$ and adjusting ϵ by a constant factor, we thus have that when $m = O(1/\epsilon^2)$, $\overline{\mathcal{F}}(H_{\mathbf{a}}, H_{\mathbf{b}})$ satisfies the guarantee of Theorem 4 with probability at least 2/3. To boost success probability to $1-\delta$, we can use the exact same median-trick used in the proof of Theorem 2: instead of computing a single pair of sketches $H_{\mathbf{a}}, H_{\mathbf{b}}$ for inputs $\mathbf{a}, \mathbf{b}$, we concatenate $O(\log(1/\delta))$ sketches, each constructed using an independent random seed. If we apply $\overline{\mathcal{F}}$ to each pair of independent sketches and return the median estimate for $\langle \mathbf{a}, \mathbf{b} \rangle$, with probability at least $1-\delta$, it will satisfy our desired guarantee. □

A.2 Weighted MinHash Analysis

In this section we complete the analysis of Algorithm 5 introduced in Section 4, which yields our main result, Theorem 2. We start with a formal proof of Fact 5, which is the weighted analog of Fact 3.

Proof of Fact 5. Let $\bar{\mathcal{A}} = \{i : \bar{\mathbf{a}}[i] \neq 0\}$ and $\bar{\mathcal{B}} = \{i : \bar{\mathbf{b}}[i] \neq 0\}$. Since $\bar{\mathbf{a}}, \bar{\mathbf{b}}$ are each comprised of n blocks of L elements, with the first $\tilde{\mathbf{a}}[i]^2 \cdot L$ entries and $\tilde{\mathbf{b}}[i]^2 \cdot L$ entries in the i^{th} block set to be nonzero, we have the following equalities:

$$|\bar{\mathcal{A}} \cap \bar{\mathcal{B}}| = L \cdot \sum_{j=1}^{n} \min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2) \tag{3}$$

$$|\bar{\mathcal{A}} \cup \bar{\mathcal{B}}| = L \cdot \sum_{j=1}^{n} \max(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2). \tag{4}$$

Since $\mathbf{W}_{\mathbf{a}}^{hash}[i]$ and $\mathbf{W}_{\mathbf{b}}^{hash}[i]$ are constructed exactly as unweighted MinHash sketches of $\bar{\mathbf{a}}, \bar{\mathbf{b}}$, by claim (1) of Fact 3, $\mathbf{W}_{\mathbf{a}}^{hash}[i] = \mathbf{W}_{\mathbf{b}}^{hash}[i]$ with probability $\frac{|\bar{\mathcal{A}} \cap \bar{\mathcal{B}}|}{|\bar{\mathcal{A}} \cup \bar{\mathcal{B}}|} = \bar{J}$. This gives claim (1).

To prove claim (2) we note that it is equivalent to claiming that, unconditional on whether or not $\mathbf{W}_{\mathbf{a}}^{hash}[i] = \mathbf{W}_{\mathbf{b}}^{hash}[i]$, $W_{\mathbf{a}}^{val} = \tilde{\mathbf{a}}[j]$ and $W_{\mathbf{b}}^{val} = \tilde{\mathbf{b}}[j]$ for some shared $j \in \mathcal{A} \cap \mathcal{B}$ with probability $\frac{\min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)}{\sum_{i=1}^n \max(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)}$. To prove this statement, we use that, by Fact 3, for any $\ell \in \bar{\mathcal{A}} \cap \bar{\mathcal{B}}$, $W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i] = h^i(\ell)$, $\mathbf{W}_{\mathbf{a}}^{val}[i] = \bar{\mathbf{a}}[\ell]$, and $\mathbf{W}_{\mathbf{b}}^{val}[i] = \bar{\mathbf{b}}[\ell]$ with probability $\frac{1}{|\bar{\mathcal{A}} \cup \bar{\mathcal{B}}|} = \frac{1}{L \sum_{k=1}^n \max(\tilde{\mathbf{a}}[k]^2, \tilde{\mathbf{b}}[k]^2)}$. Now, by construction (line 3 of Algorithm 3), $\bar{\mathbf{a}}[\ell] = \tilde{\mathbf{a}}[j]$ and $\bar{\mathbf{b}}[\ell] = \tilde{\mathbf{b}}[j]$ whenever ℓ lies in the j^{th} length L block of entries in $\bar{\mathbf{a}}$. For a given j, the number of values of ℓ for which $\bar{\mathbf{a}}[\ell] = \tilde{\mathbf{a}}[j], \bar{\mathbf{b}}[\ell] = \tilde{\mathbf{b}}[j]$ is exactly $L \cdot \min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)$. Thus, summing over these entries, $W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]$, $\mathbf{W}_{\mathbf{a}}^{val}[i] = \tilde{\mathbf{a}}[j]$, and $\mathbf{W}_{\mathbf{b}}^{val}[i] = \tilde{\mathbf{b}}[j]$ with probability $\frac{\min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)}{\sum_{k=1}^n \max(\tilde{\mathbf{a}}[k]^2, \tilde{\mathbf{b}}[k]^2)}$. □

Analysis for Discrete Vectors. Next, as a step towards proving Theorem 2, we prove a restricted intermediate result, Lemma 2, that only applies to vectors whose entries, after scaling to be unit norm, are already integer multiples of $1/L$ for a fixed discretization parameter L. When this is the case, the Round procedure in Algorithm 3 is no-op: it simply returns $\mathbf{a}/\|\mathbf{a}\|$ unmodified. Making this assumption simplifies our analysis. Later we introduce a rounding error analysis to obtain a result for arbitrary vectors.

Lemma 2. *Consider any integer discretization parameter L, accuracy parameter $\epsilon \in (0, 1)$, and $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ such that for all i, $\frac{\mathbf{a}[i]^2}{\|\mathbf{a}\|^2}$ and $\frac{\mathbf{b}[i]^2}{\|\mathbf{b}\|^2}$ are integer multiples of $1/L$. When run with sample size $m = O\left(1/\epsilon^2\right)$ and discretization parameter L, Algorithm 3 returns sketches $W_{\mathbf{a}}$ and $W_{\mathbf{b}}$ such that, letting $\mathcal{F}$ denote the estimation procedure of Algorithm 5, with probability at least $2/3$,*

$$|\mathcal{F}(W_{\mathbf{a}}, W_{\mathbf{b}}) - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \epsilon \max\left(\|\mathbf{a}_{\mathcal{I}}\|\|\mathbf{b}\|, \|\mathbf{a}\|\|\mathbf{b}_{\mathcal{I}}\|\right).$$

Here $\mathcal{I} = \{i : \mathbf{a}[i] \neq 0$ and $\mathbf{b}[i] \neq 0\}$ is the intersection of $\mathbf{a}$'s and $\mathbf{b}$'s supports and $\mathbf{a}_{\mathcal{I}}, \mathbf{b}_{\mathcal{I}}$ denote $\mathbf{a}$ and $\mathbf{b}$ restricted to indices in $\mathcal{I}$.

Note that Lemma 2 is also weaker than Theorem 2 in that it only gives an accurate solution with *constant probability*, 2/3, instead of $1 - \delta$ probability for any chosen δ. This is again to simplify the analysis and later we show how the standard "median-trick" can be used to improve the success probability to $1 - \delta$ [19, 33].

Proof. As stated, since $\mathbf{a}/\|\mathbf{a}\|$ and $\mathbf{b}/\|\mathbf{b}\|$ have squared entries that are integer multiples of $1/L$ by assumption, in line 2 of Algorithm 3, Round$(\mathbf{a}/\|\mathbf{a}\|, L)$ simply sets $\tilde{\mathbf{a}} = \mathbf{a}/\|\mathbf{a}\|$. Analogously it sets $\tilde{\mathbf{b}} = \mathbf{b}/\|\mathbf{b}\|$. Let $\mathcal{A} = \{i : \mathbf{a}[i] \neq 0\}$ and $\mathcal{B} = \{i : \mathbf{b}[i] \neq 0\}$ denote the supports of $\mathbf{a}$ and $\mathbf{b}$ respectively. We have $\mathcal{I} = \mathcal{A} \cap \mathcal{B}$.

Reduction to Unit Vectors. We first note that, to prove the theorem, it suffices to only consider the inner product between the unit vectors $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{b}}$. Specifically, we will show that:

$$\left|\frac{\mathcal{F}(W_{\mathbf{a}}, W_{\mathbf{b}})}{\|\mathbf{a}\|\|\mathbf{b}\|} - \langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle\right| \leq \epsilon \sqrt{\sum_{i \in \mathcal{A} \cap \mathcal{B}} \max(\tilde{\mathbf{a}}[i]^2, \tilde{\mathbf{b}}[i]^2) \sum_{i=1}^n \max(\tilde{\mathbf{a}}[i]^2, \tilde{\mathbf{b}}[i]^2)}. \tag{5}$$

Using that $\|\tilde{\mathbf{a}}\|^2 + \|\tilde{\mathbf{b}}\|^2 = 2$ since $\tilde{\mathbf{a}}, \tilde{\mathbf{b}}$ are unit vectors, we have:

$$\begin{aligned}
&\sqrt{\sum_{i \in \mathcal{A} \cap \mathcal{B}} \max(\tilde{\mathbf{a}}[i]^2, \tilde{\mathbf{b}}[i]^2) \sum_{i=1}^n \max(\tilde{\mathbf{a}}[i]^2, \tilde{\mathbf{b}}[i]^2)} \\
&\quad \leq \sqrt{\left(\|\tilde{\mathbf{a}}_{\mathcal{I}}\|^2 + \|\tilde{\mathbf{b}}_{\mathcal{I}}\|^2\right)\left(\|\tilde{\mathbf{a}}\|^2 + \|\tilde{\mathbf{b}}\|^2\right)} \\
&\quad = \sqrt{2\left(\|\tilde{\mathbf{a}}_{\mathcal{I}}\|^2 + \|\tilde{\mathbf{b}}_{\mathcal{I}}\|^2\right)} = \sqrt{2\left(\frac{\|\mathbf{a}_{\mathcal{I}}\|^2}{\|\mathbf{a}\|^2} + \frac{\|\mathbf{b}_{\mathcal{I}}\|^2}{\|\mathbf{b}\|^2}\right)}.
\end{aligned}$$

Thus, multiplying (5) on both sides by $\|\mathbf{a}\|\|\mathbf{b}\|$ we have:

$$\begin{aligned}
|\mathcal{F}(W_{\mathbf{a}}, W_{\mathbf{b}}) - \langle \mathbf{a}, \mathbf{b} \rangle| &\leq \epsilon\sqrt{2}\|\mathbf{a}\|\|\mathbf{b}\| \cdot \sqrt{\frac{\|\mathbf{a}_{\mathcal{I}}\|^2}{\|\mathbf{a}\|^2} + \frac{\|\mathbf{b}_{\mathcal{I}}\|^2}{\|\mathbf{b}\|^2}} \\
&= \epsilon\sqrt{2}\sqrt{\|\mathbf{a}_{\mathcal{I}}\|^2\|\mathbf{b}\|^2 + \|\mathbf{b}_{\mathcal{I}}\|^2\|\mathbf{a}\|^2} \\
&\leq 2\epsilon \cdot \max\left(\|\mathbf{a}_{\mathcal{I}}\|\|\mathbf{b}\|, \|\mathbf{b}_{\mathcal{I}}\|\|\mathbf{a}\|\right).
\end{aligned}$$

The last inequality follows from the fact that the sum is at most two times the max. Adjusting ϵ by a constant gives the desired bound of Lemma 2. Thus, we turn our attention to proving (5).

Analysis for Unit Vectors. We start by analyzing an idealized version of the estimator computed by Algorithm 5, where M is replaced by the *exact* weighted union size $M = \sum_{i=1}^n \max(\tilde{\mathbf{a}}[i]^2, \tilde{\mathbf{b}}[i]^2)$. Specifically, define:

$$\mathcal{F}^* = \frac{M}{m} \sum_{i=1}^m \mathbb{1}\left[W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]\right] \cdot \frac{W_{\mathbf{a}}^{val}[i] \cdot W_{\mathbf{b}}^{val}[i]}{q_i}, \tag{6}$$

where $q_i = \min\left(W_{\mathbf{a}}^{val}[i]^2, W_{\mathbf{b}}^{val}[i]^2\right)$ as in line 1 of Algorithm 5.

We first show that $\mathbb{E}[\mathcal{F}^*] = \langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle$ and then bound $\mathcal{F}^*$'s variance. For each $i \in \{1, \ldots, m\}$ define the random variable Z_i as

$$Z_i = \mathbb{1}\left[W_{\mathbf{a}}^{hash}[i] = W_{\mathbf{b}}^{hash}[i]\right] \cdot \frac{W_{\mathbf{a}}^{val}[i] \cdot W_{\mathbf{b}}^{val}[i]}{q_i}.$$

Recalling that $\bar{J} = \frac{\sum_{j=1}^n \min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)}{\sum_{j=1}^n \max(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)}$ is the weighted Jaccard similarity between $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{b}}$, applying Fact 5 we have:

$$Z_i = \begin{cases} 0 & \text{with probability } 1 - \bar{J} \\ \frac{\tilde{\mathbf{a}}[j]\tilde{\mathbf{b}}[j]}{\min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)} & \text{with probability } \frac{\min(\tilde{\mathbf{a}}[j]^2, \tilde{\mathbf{b}}[j]^2)}{\sum_{k=1}^n \max(\tilde{\mathbf{a}}[k]^2, \tilde{\mathbf{b}}[k]^2)} \\ & \text{for all } j \in \mathcal{A} \cap \mathcal{B}. \end{cases}$$

Thus, $\mathbb{E}[Z_i] = \frac{\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle}{\sum_{k=1}^n \max(\tilde{\mathbf{a}}[k]^2, \tilde{\mathbf{b}}[k]^2)} = \frac{\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle}{M}$. Since $\mathcal{F}^* = \frac{M}{m} \sum_{i=1}^m Z_i$, it follows from linearity of expectation that:

$$\mathbb{E}[\mathcal{F}^*] = \frac{M}{m} \sum_{i=1}^m \mathbb{E}[Z_i] = \langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle. \tag{7}$$

We next bound the variance of $\mathcal{F}^*$. For each Z_i we have that:

$$\begin{aligned}\mathrm{Var}[Z_i] &\leq \sum_{j\in\mathcal{A}\cap\mathcal{B}} \frac{\min(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)}{\sum_{k=1}^n \max(\tilde{\mathbf{a}}[k]^2,\tilde{\mathbf{b}}[k]^2)} \cdot \frac{\tilde{\mathbf{a}}[j]^2\tilde{\mathbf{b}}[j]^2}{\min(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)^2}\\ &= \sum_{j\in\mathcal{A}\cap\mathcal{B}} \frac{\max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)}{\sum_{k=1}^n \max(\tilde{\mathbf{a}}[k]^2,\tilde{\mathbf{b}}[k]^2)}\\ &= \frac{\sum_{j\in\mathcal{A}\cap\mathcal{B}} \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)}{\sum_{k=1}^n \max(\tilde{\mathbf{a}}[k]^2,\tilde{\mathbf{b}}[k]^2)} = \frac{\sum_{j\in\mathcal{A}\cap\mathcal{B}} \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)}{M}\end{aligned}$$

Since each Z_i is independent, it follows that:

$$\begin{aligned}\mathrm{Var}[\mathcal{F}^*] &= \frac{M^2}{m^2}\sum_{i=1}^m \mathrm{Var}\,[Z_i]\\ &\leq \frac{1}{m}\sum_{j\in\mathcal{A}\cap\mathcal{B}} \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)\cdot\sum_{j=1}^n \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2). \quad (8)\end{aligned}$$

Combining (7) and (8) with Chebyshev's inequality, we can claim that when $m = O(1/\epsilon^2)$, with probability at least 5/6:

$$\left|\mathcal{F}^* - \langle\tilde{\mathbf{a}},\tilde{\mathbf{b}}\rangle\right| \leq \epsilon\sqrt{\sum_{j\in\mathcal{A}\cap\mathcal{B}} \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)\sum_{j=1}^n \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)}. \quad (9)$$

We want to extend this bound from the idealized estimator $\mathcal{F}^*$ to our true estimator $\mathcal{F}$, which equals $\frac{\tilde{M}}{M}\cdot\mathcal{F}^*$. To do so, we use that $\tilde{M}$ is a good approximation to M. As discussed in Section 4, this is because $\tilde{M}$ exactly equals $\frac{1}{L}$ times a distinct elements estimator applied to the support sets $\bar{\mathcal{A}}$ and $\bar{\mathcal{B}}$ of the extended vectors $\bar{\mathbf{a}}, \bar{\mathbf{b}}$. From (4) and Lemma 1, we have that for $m = O(1/\epsilon^2)$,

$$(1-\epsilon)M \leq \tilde{M} \leq (1+\epsilon)M,$$

with probability at least 5/6. It follows that:

$$(1-\epsilon)\mathcal{F}^* \leq \frac{\mathcal{F}(W_{\mathbf{a}}, W_{\mathbf{b}})}{\|\mathbf{a}\|\|\mathbf{b}\|} \leq (1+\epsilon)\mathcal{F}^*. \quad (10)$$

By a union bound, with probability at least 2/3, both (9) and (10) hold simultaneously. Finally, by Cauchy-Schwarz inequality,

$$\langle\tilde{\mathbf{a}},\tilde{\mathbf{b}}\rangle \leq \sqrt{\sum_{j\in\mathcal{A}\cap\mathcal{B}} \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)\sum_{j=1}^n \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)}.$$

Combining (9) and (10) with triangle inequality, it follows that

$$\begin{aligned}&\left|\frac{\mathcal{F}(W_{\mathbf{a}}, W_{\mathbf{b}})}{\|\mathbf{a}\|\|\mathbf{b}\|} - \langle\tilde{\mathbf{a}},\tilde{\mathbf{b}}\rangle\right|\\ &\qquad\leq 3\epsilon\sqrt{\sum_{j\in\mathcal{A}\cap\mathcal{B}} \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)\sum_{j=1}^n \max(\tilde{\mathbf{a}}[j]^2,\tilde{\mathbf{b}}[j]^2)}.\end{aligned}$$

Adjusting ϵ by a 1/3 factor proves Lemma 2. □

Rounding for Continuous Vectors. With Lemma 2 in place, we complete our proof of Theorem 2 by analyzing the impact of the rounding step in Algorithm 5. In Lemma 3, we show that if L is set on the order of n^6/ϵ^2, then we can bound the impact of this step on the accuracy of our inner product estimate. Formally, we have:

Lemma 3 (Rounding). *Consider any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ and discretization parameter L. Let $\tilde{\mathbf{a}} = \textsc{Round}(\mathbf{a}/\|\mathbf{a}\|, L)$ and $\tilde{\mathbf{b}} = \textsc{Round}(\mathbf{b}/\|\mathbf{b}\|, L)$, as in line 2 of Algorithm 3. Let $\mathbf{a}' = \|\mathbf{a}\|\cdot\tilde{\mathbf{a}}$ and $\mathbf{b}' = \|\mathbf{b}\|\cdot\tilde{\mathbf{b}}$, and let B denote $B = \max\left(\|\mathbf{a}_{\mathcal{I}}\|\|\mathbf{b}\|, \|\mathbf{a}\|\|\mathbf{b}_{\mathcal{I}}\|\right)$.*

1. *$\mathbf{a}', \mathbf{b}'$ satisfy the assumption of Lemma 2, that for all i, $\frac{\mathbf{a}'[i]^2}{\|\mathbf{a}'\|^2}$ and $\frac{\mathbf{b}'[i]^2}{\|\mathbf{b}'\|^2}$ are integer multiples of $1/L$.*
2. *For any discretization parameter L, sketch size m, and random seed s, Algorithm 3 yields identical outputs on $\mathbf{a}, \mathbf{b}$ and $\mathbf{a}', \mathbf{b}'$. I.e., $W_{\mathbf{a}} = W_{\mathbf{a}'}$ and $W_{\mathbf{b}} = W_{\mathbf{b}'}$.*
3. *For $L \geq 9n^6/\epsilon^2$, $|\langle\mathbf{a},\mathbf{b}\rangle - \langle\mathbf{a}',\mathbf{b}'\rangle| \leq \epsilon B$.*
4. *For $L \geq n^3$, $\max\left(\|\mathbf{a}'_{\mathcal{I}}\|\|\mathbf{b}'\|, \|\mathbf{a}'\|\|\mathbf{b}'_{\mathcal{I}}\|\right) \leq 2B$.*

Proof. We prove the four claims of the lemma in order. For the first two, we focus on $\mathbf{a}$ and $\mathbf{a}'$. Identical claims hold for $\mathbf{b}$ and $\mathbf{b}'$.

Claim 1: $\frac{\mathbf{a}'[i]^2}{\|\mathbf{a}'\|^2}$ is an integer multiple of $1/L$ for all i. First observe that $\tilde{\mathbf{a}} = \textsc{Round}(\mathbf{a}/\|\mathbf{a}\|, L)$ is a unit vector. This is ensured by line 3 of Algorithm 4. Thus, $\|\mathbf{a}'\| = \|\mathbf{a}\|\cdot\|\tilde{\mathbf{a}}\| = \|\mathbf{a}\|$ and $\frac{\mathbf{a}'[i]^2}{\|\mathbf{a}'\|^2} = \frac{\mathbf{a}'[i]^2}{\|\mathbf{a}\|^2} = \tilde{\mathbf{a}}[i]^2$. So to prove the claim, it suffices to show that $\tilde{\mathbf{a}}[i]^2$ is an integer multiple of $1/L$ for all i. This is guaranteed by Algorithm 4. After line 1, we can see that $\tilde{\mathbf{z}}[i]^2$ is an integer multiple of $1/L$ for all i. Since L is an integer, 1 is also trivially an integer multiple of $1/L$. So $\delta = 1 - \|\tilde{\mathbf{z}}\|^2$ as set in line 2 is an integer multiple of $1/L$. Finally, this ensures that $\tilde{\mathbf{z}}[i^*]^2 = \tilde{\mathbf{z}}[i^*]^2 + \delta$ as set in line 3 is an integer multiple of $1/L$, completing the claim.

Claim 2: $W_{\mathbf{a}} = W_{\mathbf{a}'}$. As shown above, $\|\mathbf{a}'\| = \|\mathbf{a}\|$. So to prove the claim, it suffices to show that $\textsc{Round}\left(\frac{\mathbf{a}}{\|\mathbf{a}\|}, L\right) = \textsc{Round}\left(\frac{\mathbf{a}'}{\|\mathbf{a}'\|}, L\right)$. This ensures that Algorithm 3 proceeds identically on inputs $\mathbf{a}$ and $\mathbf{a}'$. By Claim (1), $\textsc{Round}(\mathbf{a}'/\|\mathbf{a}'\|, L) = \mathbf{a}'/\|\mathbf{a}'\| = \mathbf{a}'/\|\mathbf{a}\| = \tilde{\mathbf{a}}$. And by definition, $\tilde{\mathbf{a}} = \textsc{Round}(\mathbf{a}/\|\mathbf{a}\|, L)$. This completes the claim.

Claim 3: For $L \geq 9n^6/\epsilon^2$, $|\langle\mathbf{a},\mathbf{b}\rangle - \langle\mathbf{a}',\mathbf{b}'\rangle| \leq \epsilon B$. Let $\hat{\mathbf{a}} = \mathbf{a}/\|\mathbf{a}\|$ and $\hat{\mathbf{b}} = \mathbf{b}/\|\mathbf{b}\|$. So $\tilde{\mathbf{a}} = \textsc{Round}(\hat{\mathbf{a}}, L)$ and $\tilde{\mathbf{b}} = \textsc{Round}(\hat{\mathbf{b}}, L)$. We will show that

$$\left|\langle\hat{\mathbf{a}},\hat{\mathbf{b}}\rangle - \langle\tilde{\mathbf{a}},\tilde{\mathbf{b}}\rangle\right| \leq \epsilon\cdot\sqrt{\|\hat{\mathbf{a}}_{\mathcal{I}}\|^2 + \|\hat{\mathbf{b}}_{\mathcal{I}}\|^2}. \quad (11)$$

Multiplying each side of (11) by $\|\mathbf{a}\|\|\mathbf{b}\|$ then gives:

$$\begin{aligned}\left|\langle\mathbf{a},\mathbf{b}\rangle - \langle\mathbf{a}',\mathbf{b}'\rangle\right| &\leq \epsilon\cdot\|\mathbf{a}\|\|\mathbf{b}\|\sqrt{\|\hat{\mathbf{a}}_{\mathcal{I}}\|^2 + \|\hat{\mathbf{b}}_{\mathcal{I}}\|^2}\\ &= \epsilon\cdot\|\mathbf{a}\|\|\mathbf{b}\|\sqrt{\left(\frac{\|\mathbf{a}_{\mathcal{I}}\|^2}{\|\mathbf{a}\|^2} + \frac{\|\mathbf{b}_{\mathcal{I}}\|^2}{\|\mathbf{b}\|^2}\right)}\\ &= \epsilon\sqrt{\left(\|\mathbf{a}_{\mathcal{I}}\|^2\|\mathbf{b}\|^2 + \|\mathbf{b}_{\mathcal{I}}\|^2\|\mathbf{a}\|^2\right)}\\ &\leq \sqrt{2}\epsilon\cdot\max\left(\|\mathbf{a}_{\mathcal{I}}\|^2\|\mathbf{b}\|^2, \|\mathbf{b}_{\mathcal{I}}\|^2\|\mathbf{a}\|^2\right),\end{aligned}$$

which completes the claim after adjusting ϵ by a constant.

We proceed to prove (11). Observe that for any $i \notin \mathcal{I}$, we have at least one of $\hat{\mathbf{a}}[i]$ or $\hat{\mathbf{b}}[i]$ equal to 0. In turn, at least one of $\tilde{\mathbf{a}}[i]$ or $\tilde{\mathbf{b}}[i]$ is also 0 since in the rounding procedure of Algorithm 4 any entry of $\mathbf{z}$ that is 0 is set to 0 in $\tilde{\mathbf{z}}$. So we can conclude that $\langle\hat{\mathbf{a}},\hat{\mathbf{b}}\rangle = \langle\hat{\mathbf{a}}_{\mathcal{I}},\hat{\mathbf{b}}_{\mathcal{I}}\rangle$ and similarly, $\langle\tilde{\mathbf{a}},\tilde{\mathbf{b}}\rangle = \langle\tilde{\mathbf{a}}_{\mathcal{I}},\tilde{\mathbf{b}}_{\mathcal{I}}\rangle$. This gives that:

$$\left|\langle\hat{\mathbf{a}},\hat{\mathbf{b}}\rangle - \langle\tilde{\mathbf{a}},\tilde{\mathbf{b}}\rangle\right| = \left|\langle\hat{\mathbf{a}}_{\mathcal{I}},\hat{\mathbf{b}}_{\mathcal{I}}\rangle - \langle\tilde{\mathbf{a}}_{\mathcal{I}},\tilde{\mathbf{b}}_{\mathcal{I}}\rangle\right|.$$

So, to prove (11), it suffices to bound the righthand side of the above equation. We consider two cases:

Case 1: $\max\left(\|\hat{\mathbf{a}}_{\mathcal{I}}\|, \|\hat{\mathbf{b}}_{\mathcal{I}}\|\right) \geq \frac{1}{\sqrt{L}}$**.** For $i \in \mathcal{I}$, if $|\hat{\mathbf{a}}[i]| < \frac{1}{\sqrt{L}}$ and $L \geq n$, then $|\hat{\mathbf{a}}[i]| < \frac{1}{\sqrt{n}}$ and so $i \neq \arg\max_{i \in 1,\ldots,n} \hat{\mathbf{a}}[i]$ since $\hat{\mathbf{a}}$ is a unit vector so has at least one entry with magnitude $\geq 1/\sqrt{n}$. Thus, $\hat{\mathbf{a}}[i]$ is rounded in line 1 of Algorithm 4, and not in line 3. We have $\lfloor \hat{\mathbf{a}}[i]^2 \cdot L \rfloor = 0$ and so $|\tilde{\mathbf{a}}[i] - \hat{\mathbf{a}}[i]| = |\hat{\mathbf{a}}[i]| < \frac{1}{\sqrt{L}}$. Alternatively, if $|\hat{\mathbf{a}}[i]| \geq \frac{1}{\sqrt{L}}$ and $i \neq \arg\max_{i \in 1,\ldots,n} \hat{\mathbf{a}}[i]$ (so $\hat{\mathbf{a}}[i]$ is rounded in line 1 but not line 3 of Algorithm 4) then:

$$|\tilde{\mathbf{a}}[i] - \hat{\mathbf{a}}[i]| \leq \frac{1}{\sqrt{L}} \cdot \left|\sqrt{\hat{\mathbf{a}}[i]^2 \cdot L} - \sqrt{\hat{\mathbf{a}}[i]^2 \cdot L - 1}\right|$$
$$= \frac{1}{\sqrt{L}} \cdot \frac{1}{\sqrt{\hat{\mathbf{a}}[i]^2 \cdot L} + \sqrt{\hat{\mathbf{a}}[i]^2 \cdot L - 1}} \leq \frac{1}{\sqrt{L}}.$$

If $i = \arg\max_{i \in 1,\ldots,n} \hat{\mathbf{a}}[i]$ then $\hat{\mathbf{a}}[i]$ is rounded in line 3 and so

$$|\tilde{\mathbf{a}}[i] - \hat{\mathbf{a}}[i]| \leq \left|\sqrt{\hat{\mathbf{a}}[i]^2 + \delta} - |\hat{\mathbf{a}}[i]|\right| \leq \frac{\delta}{2|\hat{\mathbf{a}}[i]|}, \tag{12}$$

where we use that $\sqrt{x}$ is concave with derivative $\frac{1}{2|\hat{\mathbf{a}}[i]|}$ at $\hat{\mathbf{a}}[i]^2$. In line 2 of Algorithm 4 we set $\delta = 1 - \|\tilde{\mathbf{a}}\|^2$, where $\tilde{\mathbf{a}}$ is formed by rounding down entries of $\hat{\mathbf{a}}$ in line 1. Each squared entry is rounded down by at most $1/L$, so recalling that $\hat{\mathbf{a}}$ is a unit vector, $\delta \leq n/L$. Plugging into (12), and recalling that we assume $\hat{\mathbf{a}}[i] \geq 1/\sqrt{L}$,

$$|\tilde{\mathbf{a}}[i] - \hat{\mathbf{a}}[i]| \leq \frac{n/L}{2/\sqrt{L}} \leq \frac{n}{\sqrt{L}}. \tag{13}$$

Overall, we can conclude that $\|\tilde{\mathbf{a}}_{\mathcal{I}} - \hat{\mathbf{a}}_{\mathcal{I}}\|_\infty \leq \frac{n}{\sqrt{L}}$. Similarly, we have $\|\tilde{\mathbf{b}}_{\mathcal{I}} - \hat{\mathbf{b}}_{\mathcal{I}}\|_\infty \leq \frac{n}{\sqrt{L}}$. Thus,

$$\left|\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle - \langle \hat{\mathbf{a}}, \hat{\mathbf{b}} \rangle\right| = \left|\langle \tilde{\mathbf{a}}_{\mathcal{I}}, \tilde{\mathbf{b}}_{\mathcal{I}} \rangle - \langle \hat{\mathbf{a}}_{\mathcal{I}}, \hat{\mathbf{b}}_{\mathcal{I}} \rangle\right|$$
$$\leq \frac{n}{\sqrt{L}} \left(\|\hat{\mathbf{a}}_{\mathcal{I}}\|_1 + \|\hat{\mathbf{b}}_{\mathcal{I}}\|_1\right) + \frac{|\mathcal{I}| \cdot n^2}{L}.$$

By Cauchy-Schwarz, we have $\|\hat{\mathbf{a}}_{\mathcal{I}}\|_1 \leq \sqrt{|\mathcal{I}|} \cdot \|\hat{\mathbf{a}}_{\mathcal{I}}\|$ and $\|\hat{\mathbf{b}}_{\mathcal{I}}\|_1 \leq \sqrt{|\mathcal{I}|} \cdot \|\hat{\mathbf{b}}_{\mathcal{I}}\|$. Overall, this gives:

$$\left|\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle - \langle \hat{\mathbf{a}}, \hat{\mathbf{b}} \rangle\right| \leq \frac{n\sqrt{|\mathcal{I}|}}{\sqrt{L}} \left(\|\hat{\mathbf{a}}_{\mathcal{I}}\| + \|\hat{\mathbf{b}}_{\mathcal{I}}\|\right) + \frac{|\mathcal{I}| \cdot n^2}{L}$$
$$\leq \frac{n^3}{\sqrt{L}} \cdot \left(\|\hat{\mathbf{a}}_{\mathcal{I}}\| + \|\hat{\mathbf{b}}_{\mathcal{I}}\| + \max\left(\|\hat{\mathbf{a}}_{\mathcal{I}}\|, \|\hat{\mathbf{b}}_{\mathcal{I}}\|\right)\right),$$

where in the last line we use that $|\mathcal{I}| \leq n$, along with the assumption of Case 1 that $\max\left(\|\hat{\mathbf{a}}_{\mathcal{I}}\|, \|\hat{\mathbf{b}}_{\mathcal{I}}\|\right) \geq \frac{1}{\sqrt{L}}$. Setting $L \geq \frac{9n^6}{\epsilon^2}$, we have

$$\left|\langle \tilde{\mathbf{a}}, \tilde{\mathbf{b}} \rangle - \langle \hat{\mathbf{a}}, \hat{\mathbf{b}} \rangle\right| \leq \epsilon \cdot \max\left(\|\hat{\mathbf{a}}_{\mathcal{I}}\|, \|\hat{\mathbf{b}}_{\mathcal{I}}\|\right) \leq \epsilon \cdot \sqrt{\|\hat{\mathbf{a}}_{\mathcal{I}}\|^2 + \|\hat{\mathbf{b}}_{\mathcal{I}}\|^2}.$$

This proves (11) for Case 1.

Case 2: $\max\left(\|\hat{\mathbf{a}}_{\mathcal{I}}\|, \|\hat{\mathbf{b}}_{\mathcal{I}}\|\right) < \frac{1}{\sqrt{L}}$**.** In this case, for all $i \in \mathcal{I}$, $|\hat{\mathbf{a}}[i]| < \frac{1}{\sqrt{L}}$ and $|\hat{\mathbf{b}}[i]| < \frac{1}{\sqrt{L}}$. Thus, for $L > n$, no $i \in \mathcal{I}$ satisfies $i = \arg\max_{i \in 1,\ldots,n} \hat{\mathbf{a}}[i]$ or $i = \arg\max_{i \in 1,\ldots,n} \hat{\mathbf{b}}[i]$. So for all $i \in \mathcal{I}$, $\hat{\mathbf{a}}[i]$ and $\hat{\mathbf{b}}[i]$ are rounded to 0 in line 1 of Algorithm 4. I.e., $\tilde{\mathbf{a}}_{\mathcal{I}}$ and $\tilde{\mathbf{b}}_{\mathcal{I}}$ are both all zero vectors. So, to prove (11), we must show that $\left|\langle \hat{\mathbf{a}}_{\mathcal{I}}, \hat{\mathbf{b}}_{\mathcal{I}} \rangle\right| \leq \epsilon \cdot \sqrt{\|\hat{\mathbf{a}}_{\mathcal{I}}\|^2 + \|\hat{\mathbf{b}}_{\mathcal{I}}\|^2}$. This follows from Cauchy-Schwarz and our assumption that $\|\mathbf{a}_{\mathcal{I}}\|, \|\mathbf{b}_{\mathcal{I}}\| < \frac{1}{\sqrt{L}}$

$$\left|\langle \hat{\mathbf{a}}_{\mathcal{I}}, \hat{\mathbf{b}}_{\mathcal{I}} \rangle\right| \leq \|\hat{\mathbf{a}}_{\mathcal{I}}\| \|\hat{\mathbf{b}}_{\mathcal{I}}\| \leq \frac{1}{\sqrt{L}} \max\left(\|\hat{\mathbf{a}}_{\mathcal{I}}\|, \|\hat{\mathbf{b}}_{\mathcal{I}}\|\right)$$
$$\leq \frac{1}{\sqrt{L}} \sqrt{\|\hat{\mathbf{a}}_{\mathcal{I}}\|^2 + \|\hat{\mathbf{b}}_{\mathcal{I}}\|^2}.$$

Setting $L \geq \frac{1}{\epsilon^2}$ gives (11), completing Claim (3) of the lemma.

Claim 4: For $L \geq n^3$, $\max\left(\|\mathbf{a}'_{\mathcal{I}}\| \|\mathbf{b}'\|, \|\mathbf{a}'\| \|\mathbf{b}'_{\mathcal{I}}\|\right) \leq 2B$. Recall that by construction $\|\mathbf{a}'\| = \|\mathbf{a}\|$ and $\|\mathbf{b}'\| = \|\mathbf{b}\|$. Thus, dividing each side of the inequality by $\|\mathbf{a}\|\|\mathbf{b}\|$ it suffices to show:

$$\max\left(\frac{\|\mathbf{a}'_{\mathcal{I}}\|}{\|\mathbf{a}\|}, \frac{\|\mathbf{b}'_{\mathcal{I}}\|}{\|\mathbf{b}\|}\right) \leq 2 \max\left(\frac{\|\mathbf{a}_{\mathcal{I}}\|}{\|\mathbf{a}\|}, \frac{\|\mathbf{b}_{\mathcal{I}}\|}{\|\mathbf{b}\|}\right).$$

I.e., we must show that $\max(\|\tilde{\mathbf{a}}_{\mathcal{I}}\|, \|\tilde{\mathbf{b}}_{\mathcal{I}}\|) \leq 2\max(\|\hat{\mathbf{a}}_{\mathcal{I}}\|, \|\hat{\mathbf{b}}_{\mathcal{I}}\|)$. It suffices to show that $\|\tilde{\mathbf{a}}_{\mathcal{I}}\| \leq 2\|\hat{\mathbf{a}}_{\mathcal{I}}\|$ and that $\|\tilde{\mathbf{b}}_{\mathcal{I}}\| \leq 2\|\hat{\mathbf{b}}_{\mathcal{I}}\|$. We focus on proving this for $\mathbf{a}$. The bound for $\mathbf{b}$ follows the same argument. We consider two cases. Let $i^* = \arg\max_{i \in 1,\ldots,n} |\hat{\mathbf{a}}[i]|$.

Case 1: $i^* \notin \mathcal{I}$**.** In this case, all entries in $\hat{\mathbf{a}}_{\mathcal{I}}$ are only rounded in line 1 of Algorithm 4. They are thus all rounded down and so $\|\tilde{\mathbf{a}}_{\mathcal{I}}\| \leq \|\hat{\mathbf{a}}_{\mathcal{I}}\|$, giving the claim.

Case 2: $i^* \in \mathcal{I}$**.** In this case, since $\hat{\mathbf{a}}$ is a unit vector, we have $\|\hat{\mathbf{a}}_{\mathcal{I}}\| \geq |\hat{\mathbf{a}}[i^*]| \geq 1/\sqrt{n} \geq 1/\sqrt{L}$ when $L > n$. Further, all entries in $\hat{\mathbf{a}}_{\mathcal{I}}$ are rounded down, except $\hat{\mathbf{a}}[i^*]$. But as shown via (13), $|\tilde{\mathbf{a}}[i^*]| \leq |\hat{\mathbf{a}}[i^*]| + \frac{n}{\sqrt{L}}$. Thus, $\|\tilde{\mathbf{a}}_{\mathcal{I}}\| \leq \|\hat{\mathbf{a}}_{\mathcal{I}}\| + \frac{n}{\sqrt{L}} \leq 2\|\hat{\mathbf{a}}_{\mathcal{I}}\|$, as long as $L \geq n^3$.

This completes Claim (4) and thus the lemma. □

Putting everything together. Finally, we prove our main result by combining Lemma 3 with Lemma 2.

Proof of Theorem 2. Given any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, let $\mathbf{a}'$ and $\mathbf{b}'$ be defined as in Lemma 3. Consider applying Algorithm 3 to compute sketches $W_{\mathbf{a}}, W_{\mathbf{b}}, W_{\mathbf{a}'}, W_{\mathbf{b}'}$ of size $m = O(1/\epsilon^2)$, using discretization parameter $L = O(n^6/\epsilon^2)$. Using the first claim of Lemma 3 , we can apply Lemma 2 to $\mathbf{a}', \mathbf{b}'$ to show that with probability $\geq 2/3$,

$$|\mathcal{F}(W_{\mathbf{a}'}, W_{\mathbf{b}'}) - \langle \mathbf{a}', \mathbf{b}' \rangle| \leq \epsilon \max\left(\|\mathbf{a}'_{\mathcal{I}}\| \|\mathbf{b}'\|, \|\mathbf{a}'\| \|\mathbf{b}'_{\mathcal{I}}\|\right).$$

Combining triangle inequality with Claims (2) and (4) of Lemma 3, we conclude that with probability $\geq 2/3$,

$$|\mathcal{F}(W_{\mathbf{a}}, W_{\mathbf{b}}) - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \left|\langle \mathbf{a}, \mathbf{b} \rangle - \langle \mathbf{a}', \mathbf{b}' \rangle\right| + 2\epsilon \max\left(\|\mathbf{a}_{\mathcal{I}}\| \|\mathbf{b}\|, \|\mathbf{a}\| \|\mathbf{b}_{\mathcal{I}}\|\right).$$

Finally, applying Claim (3) of Lemma 3 gives that

$$|\mathcal{F}(W_{\mathbf{a}}, W_{\mathbf{b}}) - \langle \mathbf{a}, \mathbf{b} \rangle| \leq 3\epsilon \max\left(\|\mathbf{a}_{\mathcal{I}}\| \|\mathbf{b}\|, \|\mathbf{a}\| \|\mathbf{b}_{\mathcal{I}}\|\right).$$

After adjusting ϵ by a factor of 1/3, this establishes the bound of Theorem 2. The probability of success is 2/3. Using a standard trick, we can boost the success probability by computing $t = O(\log(1/\delta))$ independent sketches of $\mathbf{a}, \mathbf{b}$ using Algorithm 3 with independent random seeds [19]. Call these sketches $W_{\mathbf{a}}^{(1)}, \ldots, W_{\mathbf{a}}^{(t)}$ and $W_{\mathbf{b}}^{(1)}, \ldots, W_{\mathbf{b}}^{(t)}$. For any i, with probability $\geq 2/3$,

$$|\mathcal{F}(W_{\mathbf{a}}^{(i)}, W_{\mathbf{b}}^{(i)}) - \langle \mathbf{a}, \mathbf{b} \rangle| \leq \epsilon \max\left(\|\mathbf{a}_{\mathcal{I}}\| \|\mathbf{b}\|, \|\mathbf{a}\| \|\mathbf{b}_{\mathcal{I}}\|\right).$$

Via a standard Chernoff bound, with probability at least $1 - \delta$, this bound holds for $> t/2$ of the independent sketches. Thus, if we take the median estimate produced by the sketches, it will satisfy the desired bound with probability $\geq 1 - \delta$. Concatenating our t independent sketches into a single sketch, we can see that the total sketch size is $t \cdot m = O(\log(1/\delta)/\epsilon^2)$, giving Theorem 2. □