TOWARDS SAMPLING DATA STRUCTURES FOR TENSOR PRODUCTS IN TURNSTILE STREAMS

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper studies the computational challenges of large-scale attention-based models in artificial intelligence by introducing innovative sampling methods in the streaming setting. Inspired by the classical definition of the ℓ_2 sampler and the recent progress of the attention scheme in Large Language Models (LLMs), we propose the definition of the attention sampler. Our approach significantly reduces the computational burden of traditional attention mechanisms. We demonstrate the effectiveness of the attention sampler from a theoretical perspective, including space and update time. Additionally, our framework exhibits scalability and broad applicability across various model architectures and domains.

1 INTRODUCTION

In recent years, the field of artificial intelligence has witnessed a significant paradigm shift with the advent of attention-based models, particularly in the domains of natural language processing and computer vision (Vaswani et al., 2017; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019; Brown et al., 2020; Zhang et al., 2022; Chowdhery et al., 2023; Touvron et al., 2023a;b; Inc., 2023; Manyika, 2023). At the heart of these models lies the attention mechanism (Vaswani et al., 2017), which is a powerful tool for enhancing the performance of deep learning networks. It enables models to focus on relevant parts of the input data, thereby facilitating context-aware processing.

However, as these models scale in size and complexity Zeng et al. (2024); Reid et al. (2024); Zhang et al. (2024); Dubey et al. (2024); Abdin et al. (2024), the computational demands of the attention mechanism increase exponentially, posing significant challenges in terms of efficiency and scalability Fu (2024). In particular, traditional attention mechanisms used in Transformer models (Vaswani et al., 2017) require computing attention weights across all elements of the input sequence, leading to a *quadratic* increase in computational complexity with respect to the sequence length (Alman & Song, 2023; Kacham et al., 2023; Han et al., 2024; Zandieh et al., 2023; Alman & Song, 2024a;b; 2025a). This computational burden becomes particularly pronounced in large-scale applications. It hinders the usage of attention-based models in resource-constrained settings and limits their real-time processing capabilities.

To deal with this problem, the core question we ask in this paper is:

Instead of computing all entries, can we recover the most important ones in efficient space and time?

Attention Samplers. We adopt the classical idea of sampling a dataset, which selects important items to represent the entire dataset. Sampling is a central and effective technique for analyzing large-scale datasets, which has broad application in the field of big data (Vitter, 1985; Gemulla et al., 2008; Cohen et al., 2011; 2014), including network traffic monitoring (Mai et al., 2006; Huang et al., 2007; Thottan et al., 2010), database management (Haas & Swami, 1992; Haas, 2016; Cohen & Geri, 2019), and data summarization (Frieze et al., 2004; Aggarwal et al., 2009; Mahabadi et al., 2019; Indyk et al., 2020; Mahabadi et al., 2020). A well-known example is the ℓ_2 sampler first asked by Cormode et al. (2005) and studied by Monemizadeh & Woodruff (2010): given a vector $x \in \mathbb{R}^n$, we sample an index $i \in [n]$ with probability $\frac{x_i^2}{\|x\|_2^2}$.

To address the challenges in implementing large-scale attention schemes, we seek to sample the most important coordinates in attention computation, reducing computational overhead and computer

storage. Inspired by the classical definition of the g-sampler on a vector, we propose the following attention sampler, which is thoroughly investigated in this paper.

Definition 1.1 (Attention sampler)**.** Given matrix $A \in \mathbb{R}^{n \times d}$, vector $x \in \mathbb{R}^d$, and a distribution function g, the attention sampler samples index $i \in [n]$ with probability $p_i = \frac{g((Ax)_i)}{\sum_{j=1}^n g((Ax)_j)}$.

The motivation of our definition lies in the internal structure of the attention mechanism. Given input matrices A_1 and A_2, the linear attention matrix is defined as $A_1 X A_2^\top$, where $X = QK^\top$ is the fused key and query matrix. For linear self-attention, A_1 and A_2 are identical. Utilizing a well-known tensor product construction (Alman & Song, 2024a), we simplify the expression of the attention matrix to a matrix-vector product. Let $A = A_1 \otimes A_2$, and let $x = \mathrm{vec}(X)$, the vectorized linear attention matrix turns out to be $\mathrm{vec}(A_1 X A_2^\top) = Ax$. Here, vec denotes the vector representation of a matrix by concatenating the rows. Therefore, our attention sampler detects the dominant entry in the linear attention matrix, providing an effective approximation of the attention scheme.

Given unlimited space and time, the sampling problem is trivial since one can compute each entry explicitly and sample an index with the corresponding probability. However, as mentioned earlier, we are not granted unlimited resource in real-world applications, for instance, in resource-constrained settings or in real-time processing. This inspires us to investigate the attention sampler in the streaming model, where the input matrix A, the weight vector x, or both A and x arrive sequentially in a data stream, and the goal is to report a valid sample *at all times* using efficient space and update time. We study the turnstile data streams where the data can either be inserted or deleted at each time.

As databases handle increasingly vast and dynamic real-time data, the streaming model has emerged as a vital framework for designing algorithms to process massive, constantly evolving datasets. Examples include real-time analysis of social media streams, sensor data for smart infrastructure, live video processing, detection of distributed denial of service (DDoS) attacks, and efficient indexing and querying in large-scale databases. In this work, we combine the streaming model with attention mechanisms and construct novel efficient attention samplers, which identifies the critical coordinates in attention computation. Our contributions can be summarized as follows:

- For the softmax distribution $\langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \exp(Ax)$, we prove an $\Omega(n)$ space streaming sampler algorithm lower bound. (See Theorem 4.4)
- As the softmax distribution has a strong lower bound, we then provide upper bounds for polynomial type samplers, i.e., L_2 sampling from Ax:
  1. For updating A and fixed x, our sampler takes $d \,\mathrm{poly}\left(\frac{1}{\epsilon}, n\right)$ bits of space and update time (see Theorem 5.3).
  2. For updating A and fixed x, our sampler takes $d \,\mathrm{poly}\left(\frac{1}{\epsilon}, n\right)$ bits of space and $O(1)$ update time (see Theorem 5.5).
  3. For updating both A and x, our sampler takes $d \,\mathrm{poly}\left(\frac{1}{\epsilon}, n\right)$ bits of space and update time (see Theorem 5.7).
- For updating both A and x, we also provide a lower bound of $\Omega(d)$ space (see Theorem 6.2).
- Toward tensor generalization, where we have updating $A_1 \in \mathbb{R}^{n \times d}$ or $A_2 \in \mathbb{R}^{n \times d}$ for $A = A_1 \otimes A_2 \in \mathbb{R}^{n^2 \times d^2}$ and fixed $x \in \mathbb{R}^{d^2}$, we sample $(i_1, i_2) = i \in [n^2]$ approximately according to the ℓ_2 sampling distribution on $Ax \in \mathbb{R}^{n^2}$ using $O(nd)$ space, $O(n)$ update time (see Theorem 7.6). Note that the trivial result takes $O(n^2)$ space.

Hardness of softmax attention. Our lower bound in the first result demonstrates the hardness for computing or approximating the softmax attention. This aligns with the lower bound in Alman & Song (2023), where they show that approximating softmax attentions up to small entry-wise error requires subquadratic time in n assuming the Strong Exponential Time Hypothesis. These challenges motivate us to explore polynomial attention mechanisms. Previous work has investigated the performance of polynomial attention from both theoretical and empirical perspectives. For instance, the PolySketchFormer (Kacham et al., 2023) demonstrates that polynomial attention achieves model quality comparable to softmax attentions with efficient low-dimensional approximations. Furthermore, polynomial attention schemes perform competitively in various vision and NLP tasks, including the linear attention in (Koohpayegani & Pirsiavash, 2024) and the polynomial attention in (Saratchandran et al., 2024). Building on these insights, we obtain efficient polynomial attention

samplers in the streaming model, whose space and update time have no dependence on n factors, effectively recovering the key components in the polynomial attention matrix.

Streaming attention mechanism. Our polynomial attention samplers work in the streaming model, which matches the core idea of streaming Large Language Model (LLM) introduced and studied by Xiao et al. (2024), and recently gained increasing focus in LLM research and big-data analysis (Strati et al., 2024; Yao et al., 2024; Shikhar et al., 2025; Xiao et al., 2025). The motivation is from long (or infinite) sequence generation, e.g., a chat bot having a day-long conversation. When we apply LLMs in these scenarios, we often encounter a efficiency-performance trade-off (Xiao et al., 2024): during the decoding stage, attention-based methods cache all Key and Value (KV) pairs, which requires excessive memory usage; in contrast, under restricted memory, the performance collapses once the sequence length exceeds the cache size. To deal with these drawbakcs, their method is to train the models with a finite attention window to work on text of infinite length. Unlike Xiao et al. (2024), our model is dynamic and data-driven, supporting both model weights and input tokens to constantly change. We note that our sampler provides a correct attention sample at all times using efficient space. Thus, we identify the important coordinates in attention computing without probing each KV pair, which has high potential in enhancing the performance of streaming LLMS.

Sparse attention mechanism. Another practical relevance of our attention sampler is sparse attention mechanism. The attention matrix has been shown to be naturally sparse empirically and theoretically (see e.g. (Deng et al., 2024b)). Based on this observation, researchers seek to reduce computation by sampling the attention layers (Child et al., 2019; Kitaev et al., 2020; Wang et al., 2020; Alman & Song, 2023; Brand et al., 2024; Deng et al., 2023c; Lai et al., 2025; Xiao et al., 2025; Zhang et al., 2025). In general, they construct a *sparse mask* that selects the importance entries in the attention multiplications while others are zeroed out. Then, they compute the partial attention corresponding to those in the sparse mask. Specifically, (Xiao et al., 2025) explores the sparse attention with streaming heads. Our attention sampler recovers large coordinates from the attention matrix given specific streamed inputs and weights. Thus, the sampler serves as an efficient subroutine in their sparse-attention sampling schemes, evaluating and enhancing the effectiveness of their construction of the sparse mask.

Streaming algorithms. In addition, our sampler can be integrated into inner product computation (see e.g. Woodruff & Zhou (2021)), which is a cornerstone for model training and attention computation. In fact, classical L_p samplers also serve as black-box subroutines in many other streaming algorithms, including finding heavy hitters, F_p moment estimation, and cascaded norm approximation Andoni et al. (2011); Jowhari et al. (2011); Jayaram & Woodruff (2021); Woodruff & Zhou (2021). Therefore, our attention sampler can be applied to discover essential properties of the attention scheme, e.g., the norm of the attention matrix.

2 Related Work

In this section, we present related work in sampling and tensor sketch.

On sampling. Given a vector $v \in \mathbb{R}^n$ and a distribution function g, recall that the classical g-sampler samples index $i \in [n]$ with probability $p_i = \frac{g(v_i)}{\sum_{j=1}^{n} g(v_j)}$. A well-known example is the L_p sampling defined by $g(z) = |z|^p$ for $p \geq 0$. The existence of such a L_p sampler algorithms first posed as a question by Cormode et al. (2005) in 2005. Monemizadeh & Woodruff (2010) partially answered this question in the affirmative by giving an L_p sampler using polylogarithmic space for $p \in [1, 2]$, although the sampling probabilities were distorted by a multiplicative $(1 + \epsilon)$ factor and an additive $\frac{1}{\text{poly}(n)}$ factor. We note that the sampler is perfect if there is no ϵ-multiplicative distortion; it is truly perfect if there is no additive distortion, i.e., the sampling probability is exact. The space requirements of the algorithm were subsequently improved (Andoni et al., 2011; Jowhari et al., 2011) and extended to other choices of index domain U and weight function W (Cohen & Geri, 2019; Mahabadi et al., 2020; 2022), while retaining a multiplicative distortion in the sampling probability. Surprisingly, Jayaram & Woodruff (2021) showed that it is possible to achieve no perfect samplers while using polylogarithmic space, while conversely Jayaram et al. (2022) showed that truly perfect samplers would require linear space, essentially closing the line of work studying the space complexity of

L_p samplers for $p \in [1, 2]$. It should be noted however, achieving such guarantees (no additive distortion) in sub-polynomial update time while retaining the space guarantees remains an intriguing open question (Jayaram et al., 2022). For the other regime of $p > 2$, recently, Woodruff et al. (2025) complemented the results by providing efficient perfect L_p samplers for $p > 2$. Swartworth et al. (2025) achieved perfect samplers with polylogarithmic update time for $p > 2$, improving on the previous update time. For a more comprehensive background on samplers, we refer to the survey by Cormode & Jowhari (2019).

On tensors. In the realm of tensor decomposition, the canonical polyadic (CP) decomposition, specifically the CANDECOMP/PARAFAC method, stands out for its unique ability to break down tensors into rank-1 tensors in a singular way, distinct from matrix decomposition (Harshman, 1970; Song et al., 2016). This method, having applications in computational neuroscience, data mining, and statistical learning (Wang et al., 2015), emphasizes the rigidity and uniqueness of tensor decomposition. Earlier studies (Tsourakakis, 2010; Phan et al., 2013; Choi & Vishwanathan, 2014; Huang et al., 2013; Kang et al., 2012; Wang et al., 2014; Bhojanapalli & Sanghavi, 2015) have delved into efficient tensor decomposition methods. Subsequent works introduced methods for fast orthogonal tensor decomposition using random linear sketching techniques (Wang et al., 2015) and explored symmetric orthogonally decomposable tensors' properties, integrating spectral theory (Robeva, 2016; Robeva & Seigal, 2017). Additionally, importance sampling for quicker decomposition was proposed (Song et al., 2016). (Deng et al., 2023a) studies the tensor cycle low rank approximation problem.

In algebraic statistics, tensor decompositions are linked to probabilistic models, particularly in determining latent variable models' identifiability through low-rank decompositions of specific moment tensors (Allman et al., 2009a;b; Rhodes & Sullivant, 2012). Kruskal's theorem (Kruskal, 1977) was pivotal in ascertaining the precision of model parameter identification. However, this approach, assuming an infinite sample size, does not provide the minimum sample size for learning model parameters within given error bounds. A more robust uniqueness guarantee is needed to ensure that the low-rank decomposition of an empirical moment tensor approximates that of an actual moment tensor, thus offering more insight into empirical moment tensors' decomposition.

Roadmap. In Section 3, we provide some standard notations and definitions in literature. In Section 4, we study the exponential sampler. In Section 5, we study the streaming upper for the ℓ_2 sampling problem, i.e., sampling coordinates from a vector Ax, where A and x may be updated across a data stream. In Section 6, we present lower bounds for the same ℓ_2 sampling problem. In Section 7, we discuss the tensor sampling problem.

3 PRELIMINARIES

For any positive integer n, we use $[n]$ to denote the set $\{1, 2, \cdots, n\}$. We use $\mathbb{E}[\cdot]$ to denote the expectation. We use $\Pr[\cdot]$ to denote the probability. We use $\mathbf{1}_n$ to denote a length-n vector where all the entries are ones. Given two length-n vectors, we use $\langle x, y \rangle$ to denote the inner product between x and y, i.e, $\langle x, y \rangle := \sum_{i=1}^n x_i y_i$. For a vector $x \in \mathbb{R}^n$, we use $\exp(x) \in \mathbb{R}^n$ to denote a vector that has length n and the i-th entry is $\exp(x_i)$. For a matrix A, we use $\exp(A)$ to denote the matrix that (i, j)-th coordinate is $\exp(A_{i,j})$. For a vector x, we use $\|x\|_2 := (\sum_{i=1}^n x_i^2)^{1/2}$. We use $\|x\|_1 := \sum_{i=1}^n |x_i|$. We use $\|x\|_0$ to denote the ℓ_0 norm of x, which is the number of nonzero entries in x. We use $\|x\|_\infty$ to denote the ℓ_∞ norm of x, which is $\max_{i \in [n]} |x_i|$.

Let n_1, n_2, d_1, d_2 be positive integers. Let $A \in \mathbb{R}^{n_1 \times d_1}$ and $B \in \mathbb{R}^{n_2 \times d_2}$. We define the Kronecker product between matrices A and B, denoted $A \otimes B \in \mathbb{R}^{n_1 n_2 \times d_1 d_2}$, as $(A \otimes B)_{(i_1-1)n_2+i_2,(j_1-1)d_2+j_2}$ is equal to $A_{i_1,j_1} B_{i_2,j_2}$, where $i_1 \in [n_1], j_1 \in [d_1], i_2 \in [n_2], j_2 \in [d_2]$.

We use $\mathrm{poly}(n)$ to denote n^C where $C > 1$ is some constant. For any function f, we use $\widetilde{O}(f)$ to denote $f \cdot \mathrm{poly}(\log f)$. For two sets A and B, we use $A \cap B$ to denote their intersection. We use $|A \cap B|$ to denote the cardinality of $A \cap B$. We use $A \cup B$ to denote the union of A and B.

TensorSketch. We next define TensorSketch (Pagh, 2013), which has been extensively used in many sketching and optimization problems (Diao et al., 2018; Song et al., 2019; Diao et al., 2019; Ahle et al., 2020; Song et al., 2021; 2024; 2022; Zhang, 2022; Song et al., 2023b). Song et al. (2022)

defined TensorSparse by composing Sparse embedding (Nelson & Nguyên, 2013; Cohen, 2016) with a tensor operation (Pagh, 2013).

Definition 3.1 (TensorSparse, see Definition 7.6 in Song et al. (2022))**.** Let $h_1, h_2 : [n] \times [s] \to [m/s]$ be $O(\log 1/\delta)$-wise independent hash functions and let $\sigma_1, \sigma_2 : [n] \times [s] \to \{\pm 1\}$ be $O(\log 1/\delta)$-wise independent random sign functions. Then, the degree two tensor sparse transform, $S : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m$ is given as:

$$R_{r,(i,j)} = \exists k \in [s] : \sigma_1(i,k)\sigma_2(j,k)/\sqrt{s} \cdot \mathbf{1}[((h_1(i,k) + h_2(j,k)) \bmod m/s) + (k-1)m/s = r]$$

For $s = 1$, the above definition becomes TensorSketch (Pagh, 2013).

4 Exponential Sampler

In this section, we define and consider exponential samplers. We then show strong space lower bounds for achieving such a data structure when the input dataset arrives in a data stream.

Let us firstly describe the offline version:

Definition 4.1 (Exponential sampler)**.** Given matrix $A \in \mathbb{R}^{n \times d}$ and $x \in \mathbb{R}^d$, the goal is to sample index $i \sim [n]$ with probability $p_i = \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \cdot \exp(Ax)_i$, where $\mathbf{1}_n$ denotes a length-n vector, $\exp(Ax) \in \mathbb{R}^n$ denotes a length-n vector with $\exp(Ax)_i = \exp((Ax)_i)$, and $\exp(z)$ is the usual exponential function.

Now, consider $y = Ax \in \mathbb{R}^n$, where either A or x, or both are arriving in a data stream, we use the following definition for each of the various cases:

Definition 4.2. Let $C > 0$ be any fixed constant and let $C_0 \in [n^{-C}, n^C]$. Let y be a vector. Then the exponential sampler outputs an index j^* such that for all $i \in [n]$, $\Pr[j^* = i] = C_0 \cdot \frac{\exp(y_i)}{\langle \exp(y), \mathbf{1}_n \rangle}$.

We first recall the (two-party) set-disjointness communication problem $\mathsf{SetDisj}_n$, in which two parties Alice and Bob have subsets A and B, respectively, of $[n]$. Note that we can equivalently view A and B as binary vectors in n-dimensional space, serving as the indicator vector for whether each index $i \in [n]$ is in the player's input subset. The task for the players is to determine whether there exists a common element in their intersection, i.e., whether there exists $i \in [n]$ such that $i \in (A \cap B)$ or equivalently, $A_i = B_i = 1$. In fact, the problem promises that either the inputs are completely disjoint, $|A \cap B| = 0$ or the inputs contain only a single coordinate in their intersection, $|A \cap B| = 1$. We recall the following standard communication complexity result of set-disjointness.

Theorem 4.3 (Kalyanasundaram & Schnitger (1992); Razborov (1992); Bar-Yossef et al. (2004))**.** *Any protocol that solves the set-disjointness problem* $\mathsf{SetDisj}_n$ *with probability at least* $\frac{3}{4}$ *requires* $\Omega(n)$ *bits of total communication.*

We show that even a sampler that relaxes the probability distribution defined in Definition 4.2 up to a factor of n^C is infeasible in the streaming model.

Theorem 4.4. *Let $y \in \mathbb{R}^n$ that arrives as a data stream and let $C > 0$ be a constant. Then any algorithm that samples an index $i \in [n]$ with probability proportional to $p_i = \frac{\exp(y_j)}{\langle \exp(y), \mathbf{1}_n \rangle}$ must use $\Omega(n)$ bits of space, even if the sampling probabilities are allowed to be distorted by as large as n^C and even if $\|y\|_\infty = O(\log n)$.*

Proof. Let $A, B \in \{0,1\}^n$ be input vectors from the set disjointness problem, so that the goal is to determine whether there exists $i \in [n]$ such that $A_i = B_i = 0$. Observe that Alice and Bob can multiply A and B by $100C \log n$ for some constant $C > 0$. Now, note that in the disjoint case, we have that $\|A + B\|_\infty = 100C \log n$ and in the non-disjoint case, we have that $\|A + B\|_\infty = 200C \log n$. In particular, in the non-disjoint case, there exists $i \in [n]$ such that $A_i + B_i = 200C \log n$ and for all $j \neq i$, we have that $A_j + B_j \leq 100C \log n$. Hence, in the non-disjoint case, any exponential sampler will output i with probability proportional to $\exp(200C \log n)$ and output $j \neq i$ with probability proportional to $n \cdot \exp(100C \log n)$. Even if the sampling probabilities are distorted by a factor of n^C, any exponential sampler would output i with probability at least $\frac{3}{4}$.

Thus, Alice and Bob can use such a data structure to sample an index i and then check whether $A_i = B_i = 1$. In particular, Alice can first create a data stream encoding the vector A, run the

sampling algorithm on the data stream, and then pass the state of the algorithm to Bob. Bob can then create another portion of the data stream encoding an addition of the vector B, take the state of the algorithm from Alice, run the sampling algorithm on the portion of the data stream, and query the algorithm for an index i. Bob can then take the index and pass it to Alice, and the two parties can finally communicate whether $A_i = B_i = 1$, thereby solving set-disjointness with probability at least $\frac{3}{4}$. Note that the communication of the protocol is the space used by the sampling algorithm. Therefore by Theorem 4.3, such a sampler must use $\Omega(n)$ bits of space. □

5 ℓ_2 SAMPLER UPPER BOUND WITH A AND x

In this section, we describe a standard data structure for ℓ_2 sampling. We start with providing the definition of ℓ_2 sampler as follows,

Definition 5.1. Let n denote a positive integer. Let $\epsilon \geq 0$ denote a parameter. In ℓ_2 sampling, we receive each coordinate of $y \in \mathbb{R}^n$ in a turnstile data stream, and the goal is to output an index $I \in [n]$ at all times such that for each $j \in [n]$, $\Pr[I = j] = (1 \pm \epsilon) \cdot \frac{|y_j|^2}{\|y\|_2^2} + 1/\operatorname{poly}(n)$.

We describe various instantiations of the ℓ_2 sampler for sampling entries from a vector $Ax \in \mathbb{R}^n$, based upon whether the matrix $A \in \mathbb{R}^{n \times d}$ is updated during the data stream, whether the vector $x \in \mathbb{R}^d$ is updated during the data stream, or both.

5.1 A IS UPDATED DURING THE STREAMING AND x IS FIXED

In this section, we describe the construction of an ℓ_2 sampler for sampling coordinates of the vector $Ax \in \mathbb{R}^n$, in the setting where the vector $x \in \mathbb{R}^d$ is fixed, but the entries of $A \in \mathbb{R}^{n \times d}$ are evolving as the data stream progresses.

Definition 5.2 (Updating A and fixed x)**.** In this setting, we assume $x \in \mathbb{R}^d$ is fixed, we receive updates to the entries of $A \in \mathbb{R}^{n \times d}$ in a turnstile data stream. Then for $y = Ax$, we want a data structure that produces the ℓ_2 sampling guarantee for y.

We remark that a turnstile data stream means that each update of the data stream can increase or decrease a single entry of A.

In this work, we are interested in the regime of $n \gg d$. Then we have the following guarantee:

Theorem 5.3. *Suppose $y = Ax$, for $x \in \mathbb{R}^n$, which is fixed, and $A \in \mathbb{R}^{n \times d}$, which is defined by a turnstile stream. There exists an ℓ_2-attention sampler that uses $d \log n + \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space and returns $I \in [n]$ such that $\Pr[I = j] = (1 \pm \epsilon) \cdot \frac{|y_j|^2}{\|y\|_2^2} + 1/\operatorname{poly}(n)$. The update time of the data structure is $d \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$.*

Proof. Recall that existing approximate ℓ_2 samplers, e.g., Algorithm 2 maintains a linear sketch Φy, where $\Phi \in \mathbb{R}^{m \times n}$, for $m = \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$. We have $y = Ax$, where $x \in \mathbb{R}^d$ is fixed but $A \in \mathbb{R}^{n \times d}$ is defined through turnstile updates. Nevertheless, we can maintain the state of ΦAx. In particular, whenever we receive an update in $A_{i,j}$ by Δ, then we can compute $\Phi e_i e_j^\top \Delta x$ to update the sketch ΦAx. To analyze the space complexity, observe that storing ΦAx requires $O(m)$ words of space and x requires d words of space, which is $d \log n + \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space in total. Moreover, each update to $A_{i,j}$ can change all entries of ΦAx, so the update time is $O(md) = d \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$.

□

5.2 x IS UPDATED DURING THE STREAMING AND A IS FIXED

We next consider the setting where the vector $x \in \mathbb{R}^d$ is updated as the data stream progresses, but the entries of $A \in \mathbb{R}^{n \times d}$ are fixed.

Definition 5.4 (Fixed A and updating x)**.** We assume $A \in \mathbb{R}^{n \times d}$ is fixed, we receive updates to $x \in \mathbb{R}^d$ in a turnstile data stream. Then for $y = Ax$, we want a data structure that produces the ℓ_2 sampling guarantee for y.

We have the following algorithmic guarantees for this setting:

Theorem 5.5. *Suppose $y = Ax$, for $A \in \mathbb{R}^{n\times d}$, which is fixed, and $x \in \mathbb{R}^n$, which is defined by a turnstile stream. There is an ℓ_2-attention sampler that uses $d \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space and returns $I \in [n]$ such that $\Pr[I = j] = (1 \pm \epsilon) \cdot \frac{|y_j|^2}{\|y\|_2^2} + 1/\operatorname{poly}(n)$. The update time of the data structure is $O(1)$.*

Proof. Again recall that existing approximate ℓ_2 samplers, e.g., Algorithm 2 maintains a linear sketch Φy, where $\Phi \in \mathbb{R}^{m\times n}$, for $m = \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$. Since $y = Ax$, but $A \in \mathbb{R}^{n\times d}$ is too large to store, while $x \in \mathbb{R}^n$ is defined through turnstile updates, we can instead maintain the sketch ΦA and the vector x and compute $\Phi Ax = \Phi y$ after the stream concludes. Note that storing ΦA requires $O(md)$ words of space and x requires d words of space, which is $d \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space in total. Moreover, each update to x changes a single entry, so the update time is $O(1)$. □

5.3 Both A and x are updated during the streaming

Finally, we consider the setting where both the vector $x \in \mathbb{R}^d$ and the entries of $A \in \mathbb{R}^{n\times d}$ can be changed by updates from the data stream.

Definition 5.6 (Updating A and updating x)**.** In this setting, we receive updates to both $A \in \mathbb{R}^{n\times d}$ and $x \in \mathbb{R}^d$ in a turnstile data stream. Then for $y = Ax$, we want a data structure that provides the ℓ_2 sampling guarantee for y.

We have the following guarantees:

Lemma 5.7 (Upper Bound)**.** *Suppose $y = Ax$, for $A \in \mathbb{R}^{n\times d}$ and $x \in \mathbb{R}^d$, which are each defined in a stream through turnstile updates. There exists an $\ell-2$-attention sampler that uses $d \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space and returns $I \in [n]$ such that $\Pr[I = j] = (1 \pm \epsilon) \cdot \frac{|y_j|^2}{\|y\|_2^2} + 1/\operatorname{poly}(n)$. The update time is $\operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$.*

Proof. As before, recall that existing approximate ℓ_2 samplers, e.g., Algorithm 2 maintains a linear sketch Φy, where $\Phi \in \mathbb{R}^{m\times n}$, for $m = \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$. Since $y = Ax$, but now both $A \in \mathbb{R}^{n\times d}$ and $x \in \mathbb{R}^n$ are defined through turnstile updates, we can instead maintain the sketch ΦA and the vector x and compute $\Phi Ax = \Phi y$ after the stream concludes. Observe that maintaining ΦA requires $O(md)$ words of space and x requires d words of space, which is $d \operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space in total. Each update to A can change all m entries of in a single column of ΦA, while each update to x changes a single entry. Hence, the update time is $\operatorname{poly}\left(\frac{1}{\epsilon}, \log n\right)$. □

6 ℓ_2 Sampler Lower Bound (with A and x)

In this section, we give lower bounds for ℓ_2 sampling from a vector $y = A^{\otimes p}x$, when either A or x are updated in a data stream. We show that in any of these cases, the general problem is substantially more difficult than the previous case where $p = 1$.

We first recall the Index problem for one-way communication. In the INDEX_n problem, Alice receives a vector $v \in \{0,1\}^n$ and Bob receives a coordinate $i \in [n]$. The goal is for Bob to compute v_i with probability at least $\frac{3}{4}$, given some message Π from Alice. We recall the following communication complexity lower bounds for Index.

Theorem 6.1 (Kremer et al. (1999))**.** *Any protocol that solves INDEX_n with probability at least $\frac{3}{4}$ requires $\Omega(n)$ bits of communication.*

Lemma 6.2 (Lower Bound)**.** *Any streaming algorithm that solves problem defined as Definition 5.6 will require $\Omega(d)$ space.*

Proof. Suppose Alice receives a vector $v \in \{0,1\}^d$. Then Alice creates the diagonal matrix $M \in \{0,1\}^{d\times d}$ so that the j-th diagonal entry of A is v_j, for all $j \in [n]$. Finally, Alice creates $A \in \mathbb{R}^{(d+1)\times d}$ by appending the row consisting of $\frac{1}{10^{10}}$ in all of its d entries to M. Suppose Bob receives the coordinate $i \in [d]$ and wants to determine v_i. Then Bob can set x to be the elementary vector

$e_i \in \mathbb{R}^d$, which has a 1 in its i-th coordinate and zeros elsewhere. Observe that by construction, Ax is the i-th column of A. If $v_i = 1$, then the i-th column of A consists of a 1 in the i-th entry, $\frac{1}{10^{10}}$ in the $(d+1)$-st entry, and zeros elsewhere. Hence, a sampler with the desired properties will output i with probability at least $\frac{3}{4}$. Similarly, if $v_i = 0$, then the i-th column of A consists of $\frac{1}{10^{10}}$ in the $(d+1)$-st entry and zeros elsewhere. Thus, the sampler with the desired properties will output $d+1$ with probability 1. Bob can therefore distinguish between these two cases with probability at least $\frac{3}{4}$, thereby solving INDEX_d with probability at least $\frac{3}{4}$. Therefore, by Theorem 6.1, such a sampler must use at least $\Omega(d)$ space. □

In fact, we show that if $y = A^{\otimes p}x$, where $A \in \mathbb{R}^{n\times n}$ so that $A^{\otimes p} \in \mathbb{R}^{n^p \times n^p}$ denotes the p-wise self-tensor and $x \in \mathbb{R}^{n^p}$, then actually L_2 sampling from y uses $\Omega(n)$ bits of space.

Lemma 6.3. *Let $A \in \mathbb{R}^{n\times n}$ and $A^{\otimes p} \in \mathbb{R}^{n^p \times n^p}$ denote the p-wise self-tensor. Let $y = A^{\otimes p}x$, so that $x \in \mathbb{R}^{n^p}$. Then even if all the entries of x arrive in a data stream followed by all the entries of A, L_2 sampling from y requires $\Omega(n)$ bits of space.*

Proof. Let $S \in \{0,1\}^n$ be an instance of INDEX_n. Suppose Alice creates the diagonal matrix A with exactly S being the entries across its diagonal, i.e., $A_{1,1} = S_1, \ldots, A_{n,n} = S_n$. Bob has an index $i \in [n]$, and sets the vector x to be the elementary vector $\mathbf{e}_j$, where $j = i \cdot n^{p-1}$. Then by construction Ax is the all zeros vector if $S_i = 0$ and otherwise there is a nonzero entry, which allows Alice and Bob to solve INDEX_n. Hence, L_2 sampling from y requires $\Omega(n)$ bits of space. □

7 The Tensor Version Problem

In this section, we further consider sampling from a tensor product. We provide the tensor notations and objects.

Definition 7.1. Let $A_1 \in \mathbb{R}^{n\times d}$, let $A_2 \in \mathbb{R}^{n\times d}$, we define $\mathsf{A} = A_1 \otimes A_2 \in \mathbb{R}^{n^2 \times d^2}$. Let $x \in \mathbb{R}^{d^2}$. Let $\mathsf{A}_i \in \mathbb{R}^{n\times d^2}$ denote the i-th block of A.

Definition 7.2 (fixed x, Streaming Sampler for one of A_1 and A_2 is updating.)**.** We assume $x \in \mathbb{R}^{d^2}$ is fixed. We assume that (1) one of A_1 and A_2 is updating, (2) one of A_1 and A_2 is fixed. Let $y = \mathsf{A}x$, we want ℓ_2 sampling guarantee for sampling one coordinate in $y_i \in \mathbb{R}^{n^2}$ for all $i \in [n^2]$.

To motive this model, recall that the tensor product $(A_1 \otimes A_2)\, x$ equals to the linear cross-attention matrix $A_1 Q K^\top A_2^\top$, where $W_Q = A_1 Q$ is the projected query matrix and $W_K = A_2 K$ is the projected key matrix. Our model addresses a practical scenario involving real-time contextual processing with a static reference dataset. In this setting, W_k is precomputed by the language model, representing a static dataset such as embeddings of a knowledge base, user profiles, or multimedia features. Then, the rows of matrix A_1 arrive as a data stream, representing real-time data queries. Thus, our attention sampler efficiently captures the important entries in the dynamic query dataset.

We use the following formulation of Nisan's pseudorandom generator to derandomize our algorithm.

Theorem 7.3 (Nisan's PRG, Nisan (1992))**.** *Suppose $\mathcal{A}$ is an algorithm that requires $S = \Omega(\log n)$ bits of space and R random bits. Then there exists a pseudorandom generator for $\mathcal{A}$ that succeeds with probability $1 - 1/\mathrm{poly}(n)$ and uses $O(S \log R)$ bits of space.*

Algorithm 1 We build on algorithm based on $S(x_1 \otimes x_2)$

1: Suppose we use $O(nd)$ space to store A_1 and A_2 (Avoid n^2 time/space)
2: Suppose we receive an update $q \in [2], i \in [n], j \in [d], \Delta$
3: Suppose we have hash function g to access uniform number
4: **if** $q = 1$ **then**
5: $\quad p \leftarrow g(i(n-1)+1, \cdots, in)$ $\{p \in \mathbb{R}^n\}$
6: $\quad y \leftarrow y + \Phi\Delta(e_{[i(n-1)+1,in]} \circ (A_2)_{*,j})/p$ $\{\Phi_1$ is decided by $h_1, \sigma_1\}$
7: **else**
8: $\quad y_2 \leftarrow y_2 + \Phi_2 e_i \Delta$ $\{\Phi_2$ is decided by $h_2, \sigma_2\}$
9: **end if**

In the following Lemma, we state a streaming algorithm to solve tensor related sampling problem. We consider the situation that one of A_1 and A_2 is fixed, and the other one is updated in streaming fashion. We show the following estimation guarantees using the standard CountSketch analysis, c.f., Charikar et al. (2004); Jowhari et al. (2011). We defer the proof to appendix C.

Lemma 7.4 (Tensor ℓ_2 Tail Estimation)**.** *Let $y = (A_1 \otimes A_2)x \in \mathbb{R}^{n^2}$. Let only one of A_1 and A_2 be updated in streaming. Let $w = \frac{y_i}{\sqrt{u_i}}$ for a constant $u_i \in [0,1]$ generated uniformly at random. There is an algorithm $\mathcal{A}$ that that uses $O(nd) + \mathrm{poly}\left(\frac{1}{\epsilon}, \log n\right)$ space, uses $O(n)$ update time, and estimates each element of w up to additive error $\epsilon \cdot \|z\|_2$, where z denotes the tail vector of w without the largest $\frac{1}{\epsilon^2}$ entries in magnitude. Specifically, for all $i \in [n^2]$, we have $|\widehat{w}_i - w_i| \leq \epsilon \cdot \|z\|_2$.*

We state the following lemma as a structural property that will allow us to achieve our tensor product sampler. We remark that the proof is a simple adaptation of existing proofs for approximate ℓ_p sampling (Jowhari et al., 2011). Thus we defer the proof to Appendix C.

Lemma 7.5. *Let $y = (A_1 \otimes A_2)x \in \mathbb{R}^{n^2}$ and let $w \in \mathbb{R}^{n^2}$ so that $w_i = \frac{y_i}{\sqrt{u_i}}$ for a constant $u_i \in [0,1]$ generated uniformly at random. Let z denote the tail vector of w without the largest $\frac{1}{\epsilon^2}$ entries in magnitude. Let $\widehat{Z}$ be a 2-approximation to $\|z\|_2$ and $\widehat{Y}$ be a 2-approximation to $\|y\|_2$, then we have $\Pr[\widehat{Z} > \sqrt{(C \log n)/\epsilon} \cdot \widehat{Y}] \leq O(\epsilon) + \frac{1}{\mathrm{poly}(n)}$.*

Finally, we describe the guarantees of our tensor-based sampler, deferring the proof to Appendix D.

Theorem 7.6. *Let $y = (A_1 \otimes A_2)x \in \mathbb{R}^{n^2}$ and let $w \in \mathbb{R}^{n^2}$ so that for each $i \in [n^2]$, $w_i = \frac{y_i}{\sqrt{u_i}}$ for a constant $u_i \in [0,1]$ generated uniformly at random. Let z denote the tail vector of w without the largest $\frac{1}{\epsilon^2}$ entries in magnitude. Suppose there exists:*

1. *An algorithm $\mathcal{A}_1$ that provides a 2-approximation to $\|y\|_2$ with probability $1 - \frac{1}{n^2}$.*
2. *An algorithm $\mathcal{A}_2$ that provides a 2-approximation to $\|z\|_2$ with probability $1 - \frac{1}{n^2}$.*
3. *An algorithm $\mathcal{A}_3$ that estimates each element of w up to additive error $\epsilon \cdot \|z\|_2$, $|\widehat{w_i} - w_i| \leq \epsilon \cdot \|z\|_2$, for all $i \in [n^2]$.*

Then there exists a data structure that uses $\mathrm{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space and outputs each index i with probability $p_i = (1 \pm \epsilon) \cdot \frac{y_i^2}{\|y\|_2^2} \pm \frac{1}{\mathrm{poly}(n)}$.

We remark that the algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ in the context of Theorem 7.6 can be achieved using the standard AMS ℓ_2 norm estimator (Alon et al., 1999). Moreover, algorithm $\mathcal{A}_3$ in the context of Theorem 7.6 can be achieved using the standard CountSketch algorithm (Charikar et al., 2004).

8 Conclusions

To achieve efficient attention mechanisms, we introduce the attention sampler and study its behavior in the streaming model. We established efficient polynomial samplers under various streaming settings, when the input matrix, the weight vector, or both evolve dynamically, and we complement the results by proving space lower bounds. Our framework identify the critical components in attention computation, offering a foundation for efficient simulations of large-scale attention schemes, which is central to modern machine learning and LLMs.

For future directions, from a theoretical perspective, given the $\Omega(n)$ lower bound on exponential samplers in general circumstances, it would be valuable to explore whether we can achieve $o(n)$ space under certain assumptions, e.g., restricting the entries in the attention matrix to $o(\log n)$. From a practical perspective, it would be beneficial to evaluate our sampler's performance by implementing it in existing sparse attention schemes and streaming attention schemes.

ETHIC STATEMENT

This paper does not involve human subjects, personally identifiable data, or sensitive applications. We do not foresee direct ethical risks. We follow the ICLR Code of Ethics and affirm that all aspects of this research comply with the principles of fairness, transparency, and integrity.

REPRODUCIBILITY STATEMENT

We ensure reproducibility of our theoretical results by including all formal assumptions, definitions, and complete proofs in the appendix. The main text states each theorem clearly and refers to the detailed proofs. No external data or software is required.

REFERENCES

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat S. Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xia Song, Masahiro Tanaka, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone. *CoRR*, abs/2404.14219, 2024. doi: 10.48550/ARXIV.2404.14219. URL https://doi.org/10.48550/arXiv.2404.14219.

Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX and 13th International Workshop, RANDOM. Proceedings*, pp. 15–28, 2009.

Thomas D Ahle, Michael Kapralov, Jakob BT Knudsen, Rasmus Pagh, Ameya Velingker, David P Woodruff, and Amir Zandieh. Oblivious sketching of high-degree polynomial kernels. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 141–160. SIAM, 2020.

Elizabeth S. Allman, Catherine Matias, and John A. Rhodes. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, 37(6A), dec 2009a. doi: 10.1214/09-aos689. URL https://doi.org/10.1214%2F09-aos689.

Elizabeth S. Allman, Sonja Petrović, John A. Rhodes, and Seth Sullivant. Identifiability of 2-tree mixtures for group-based models, 2009b. URL https://arxiv.org/abs/0909.1854.

Josh Alman and Zhao Song. Fast attention requires bounded entries. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2023.

Josh Alman and Zhao Song. How to capture higher-order correlations? generalizing matrix softmax attention to kronecker computation. In *The Twelfth International Conference on Learning Representations, ICLR*, 2024a.

Josh Alman and Zhao Song. The fine-grained complexity of gradient computation for training large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=up4tWnwRol.

Josh Alman and Zhao Song. Fast rope attention: Combining the polynomial method and fast fourier transform. *arXiv preprint arXiv:2505.11892*, 2025a.

Josh Alman and Zhao Song. Only large weights (and not skip connections) can prevent the perils of rank collapse. *arXiv preprint arXiv:2505.16284*, 2025b.

Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

Alexandr Andoni, Robert Krauthgamer, and Krzysztof Onak. Streaming algorithms via precision sampling. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS*, pp. 363–372, 2011.

Pranjal Awasthi and Anupam Gupta. Improving length-generalization in transformers via task hinting. *arXiv preprint arXiv:2310.00726*, 2023.

Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.*, 68(4):702–732, 2004.

Srinadh Bhojanapalli and Sujay Sanghavi. A new sampling technique for tensors. In *arXiv preprint*. `https://arxiv.org/pdf/1502.05023`, 2015.

Jan van den Brand, Zhao Song, and Tianyi Zhou. Algorithm and hardness for dynamic attention maintenance in large language models. In *Forty-first International Conference on Machine Learning, ICML*. OpenReview.net, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS*, pp. 1877–1901, 2020.

Moses Charikar, Kevin C. Chen, and Martin Farach-Colton. Finding frequent items in data streams. *Theor. Comput. Sci.*, 312(1):3–15, 2004.

Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. Scatterbrain: Unifying sparse and low-rank attention. *Advances in Neural Information Processing Systems (NeurIPS)*, 34: 17413–17426, 2021.

Beidi Chen, Tri Dao, Kaizhao Liang, Jiaming Yang, Zhao Song, Atri Rudra, and Christopher Re. Pixelated butterfly: Simple and efficient sparse training for neural network models. In *International Conference on Learning Representations, ICLR*, 2022.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Joon Hee Choi and S. Vishwanathan. Dfacto: Distributed factorization of tensors. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems, NeurIPS*, pp. 1296–1304, 2014.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24:240:1–240:113, 2023.

Edith Cohen and Ofir Geri. Sampling sketches for concave sublinear functions of frequencies. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, NeurIPS*, pp. 1361–1371, 2019.

Edith Cohen, Nick G. Duffield, Haim Kaplan, Carsten Lund, and Mikkel Thorup. Efficient stream sampling for variance-optimal estimation of subset sums. *SIAM J. Comput.*, 40(5):1402–1431, 2011.

Edith Cohen, Nick G. Duffield, Haim Kaplan, Carsten Lund, and Mikkel Thorup. Algorithms and estimators for summarization of unaggregated data streams. *J. Comput. Syst. Sci.*, 80(7):1214–1244, 2014.

Michael B. Cohen. Nearly tight oblivious subspace embeddings by trace inequalities. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), Arlington, VA, USA, January 10-12, 2016*, pp. 278–287, 2016.

Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing (STOC)*. `https://arxiv.org/pdf/1810.07896.pdf`, 2019.

Graham Cormode and Hossein Jowhari. l_p samplers and their applications: A survey. *ACM Comput. Surv.*, 52(1):16:1–16:31, 2019.

Graham Cormode, S. Muthukrishnan, and Irina Rozenbaum. Summarizing and mining inverse distributions on data streams via dynamic inverse sampling. In *Proceedings of the 31st International Conference on Very Large Data Bases*, pp. 25–36. ACM, 2005.

Giannis Daras, Nikita Kitaev, Augustus Odena, and Alexandros G Dimakis. Smyrf-efficient attention using asymmetric clustering. *Advances in Neural Information Processing Systems (NeurIPS)*, 33: 6476–6489, 2020.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

Yichuan Deng, Wenyu Jin, Zhao Song, Xiaorui Sun, and Omri Weinstein. Dynamic kernel sparsifiers. *arXiv preprint arXiv:2211.14825*, 2022a.

Yichuan Deng, Zhao Song, Omri Weinstein, and Ruizhe Zhang. Fast distance oracles for any symmetric norm. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS*, 2022b.

Yichuan Deng, Yeqi Gao, and Zhao Song. Solving tensor low cycle rank approximation. In *IEEE International Conference on Big Data, BigData 2023, Sorrento, Italy, December 15-18, 2023*, pp. 6–16. IEEE, 2023a.

Yichuan Deng, Zhihang Li, and Zhao Song. Attention scheme inspired softmax regression. *arXiv preprint arXiv:2304.10411*, 2023b.

Yichuan Deng, Sridhar Mahadevan, and Zhao Song. Randomized and deterministic attention sparsification algorithms for over-parameterized feature dimension. *arxiv preprint: arxiv 2304.03426*, 2023c.

Yichuan Deng, Zhao Song, and Shenghao Xie. Convergence of two-layer regression with nonlinear units. *arXiv preprint arXiv:2308.08358*, 2023d.

Yichuan Deng, Zhihang Li, Sridhar Mahadevan, and Zhao Song. Zero-th order algorithm for softmax attention optimization. In *IEEE International Conference on Big Data, BigData 2024, Washington, DC, USA, December 15-18, 2024*, pp. 24–33. IEEE, 2024a.

Yichuan Deng, Zhao Song, and Chiwun Yang. Attention is naturally sparse with gaussian distributed input. *arXiv preprint arXiv:2404.02690*, 2024b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Huaian Diao, Zhao Song, Wen Sun, and David P. Woodruff. Sketching for kronecker product regression and p-splines. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, volume 84 of *Proceedings of Machine Learning Research*, pp. 1299–1308. PMLR, 2018.

Huaian Diao, Rajesh Jayaram, Zhao Song, Wen Sun, and David Woodruff. Optimal sketching for kronecker product regression and low rank approximation. *Advances in neural information processing systems*, 32, 2019.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. URL https://doi.org/10.48550/arXiv.2407.21783.

Benjamin L. Edelman, Surbhi Goel, Sham M. Kakade, and Cyril Zhang. Inductive biases and variable creation in self-attention mechanisms. In *International Conference on Machine Learning, ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5793–5831. PMLR, 2022.

Alan M. Frieze, Ravi Kannan, and Santosh S. Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *J. ACM*, 51(6):1025–1041, 2004.

Yao Fu. Challenges in deploying long-context transformers: A theoretical peak performance analysis. *CoRR*, abs/2405.08944, 2024. doi: 10.48550/ARXIV.2405.08944. URL https://doi.org/10.48550/arXiv.2405.08944.

Yeqi Gao, Lianke Qin, Zhao Song, and Yitan Wang. A sublinear adversarial training algorithm. *arXiv preprint arXiv:2208.05395*, 2022.

Yeqi Gao, Sridhar Mahadevan, and Zhao Song. An over-parameterized exponential regression. *arXiv preprint arXiv:2303.16504*, 2023.

Yeqi Gao, Zhao Song, and Junze Yin. An iterative algorithm for rescaled hyperbolic functions regression. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 258 of *Proceedings of Machine Learning Research*, pp. 2548–2556. PMLR, 2025.

Rainer Gemulla, Wolfgang Lehner, and Peter J. Haas. Maintaining bounded-size sample synopses of evolving datasets. *VLDB J.*, 17(2):173–202, 2008.

Yuzhou Gu and Zhao Song. A faster small treewidth sdp solver. *arXiv preprint arXiv:2211.06033*, 2022.

Yuzhou Gu, Zhao Song, Junze Yin, and Lichen Zhang. Low rank matrix completion via robust alternating minimization in nearly linear time. In *The Twelfth International Conference on Learning Representations, ICLR*. OpenReview.net, 2024.

Peter J. Haas. Data-stream sampling: Basic techniques and results. In *Data Stream Management - Processing High-Speed Data Streams*, Data-Centric Systems and Applications, pp. 13–44. Springer, 2016.

Peter J. Haas and Arun N. Swami. Sequential sampling procedures for query size estimation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 341–350, 1992.

Insu Han, Rajesh Jayaram, Amin Karbasi, Vahab Mirrokni, David P. Woodruff, and Amir Zandieh. Hyperattention: Long-context attention in near-linear time. In *The Twelfth International Conference on Learning Representations, ICLR*. OpenReview.net, 2024.

Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

Furong Huang, Niranjan U. N, Mohammad Umar Hakeem, Prateek Verma, and Animashree Anandkumar. Fast detection of overlapping communities via online tensor methods on gpus. *CoRR*, abs/1309.0787, 2013.

Ling Huang, XuanLong Nguyen, Minos N. Garofalakis, Joseph M. Hellerstein, Michael I. Jordan, Anthony D. Joseph, and Nina Taft. Communication-efficient online detection of network-wide anomalies. In *INFOCOM. 26th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies*, pp. 134–142, 2007.

Adobe Inc. Adobe firefly. https://www.adobe.com/sensei/generative-ai/firefly.html, 2023.

Piotr Indyk, Sepideh Mahabadi, Shayan Oveis Gharan, and Alireza Rezaei. Composable core-sets for determinant maximization problems via spectral spanners. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA*, pp. 1675–1694, 2020.

Rajesh Jayaram and David P. Woodruff. Perfect l_p sampling in a data stream. *SIAM J. Comput.*, 50 (2):382–439, 2021.

Rajesh Jayaram, David P. Woodruff, and Samson Zhou. Truly perfect samplers for data streams and sliding windows. In *PODS '22: International Conference on Management of Data*, pp. 29–40, 2022.

Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster lps. *arXiv preprint arXiv:2004.07470*, 2021.

Hossein Jowhari, Mert Saglam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pp. 49–58, 2011.

Praneeth Kacham, Vahab Mirrokni, and Peilin Zhong. Polysketchformer: Fast transformers via sketches for polynomial kernels. *arXiv preprint arXiv:2310.01655*, 2023.

Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Discret. Math.*, 5(4):545–557, 1992.

U. Kang, Evangelos E. Papalexakis, Abhay Harpale, and Christos Faloutsos. Gigatensor: scaling tensor analysis up by 100 times - algorithms and discoveries. In *KDD*, pp. 316–324, 2012.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv: 2001.04451*, 2020.

Soroush Abbasi Koohpayegani and Hamed Pirsiavash. Sima: Simple softmax-free attention for vision transformers. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV*, pp. 2595–2605. IEEE, 2024.

Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. *Comput. Complex.*, 8(1):21–49, 1999.

Joseph B. Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95–138, 1977. ISSN 0024-3795. doi: https://doi.org/10.1016/0024-3795(77)90069-6. URL https://www.sciencedirect.com/science/article/pii/0024379577900696.

Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. Flexprefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *The Thirteenth International Conference on Learning Representations*. OpenReview.net, 2025.

Yin Tat Lee, Zhao Song, and Qiuyi Zhang. Solving empirical risk minimization in the current matrix multiplication time. In *COLT*. https://arxiv.org/pdf/1905.04447.pdf, 2019.

Zhihang Li, Zhao Song, and Tianyi Zhou. Solving regularized exp, cosh and sinh regression problems. *arXiv preprint arXiv:2303.15725*, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Sepideh Mahabadi, Piotr Indyk, Shayan Oveis Gharan, and Alireza Rezaei. Composable core-sets for determinant maximization: A simple near-optimal algorithm. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97, pp. 4254–4263, 2019.

Sepideh Mahabadi, Ilya P. Razenshteyn, David P. Woodruff, and Samson Zhou. Non-adaptive adaptive sampling on turnstile streams. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pp. 1251–1264, 2020.

Sepideh Mahabadi, David P. Woodruff, and Samson Zhou. Adaptive sketches for robust regression with importance sampling. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pp. 31:1–31:21, 2022.

Arvind V. Mahankali, David P. Woodruff, and Ziyu Zhang. Near-linear time and fixed-parameter tractable algorithms for tensor decompositions. In *15th Innovations in Theoretical Computer Science Conference, ITCS*, volume 287 of *LIPIcs*, pp. 79:1–79:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

Jianning Mai, Chen-Nee Chuah, Ashwin Sridharan, Tao Ye, and Hui Zang. Is sampled data sufficient for anomaly detection? In *Proceedings of the 6th ACM SIGCOMM Internet Measurement Conference, IMC*, pp. 165–176, 2006.

James Manyika. An overview of bard: an early experiment with generative ai. Technical report, Tech. rep., Technical report, Google AI, 2023.

Gary Marcus, Ernest Davis, and Scott Aaronson. A very preliminary analysis of dall-e 2. *arXiv preprint arXiv:2204.13807*, 2022.

Morteza Monemizadeh and David P. Woodruff. 1-pass relative-error l_p-sampling with applications. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pp. 1143–1160. SIAM, 2010.

Jelani Nelson and Huy L Nguyên. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *2013 ieee 54th annual symposium on foundations of computer science*, pp. 117–126. IEEE, 2013.

Noam Nisan. Pseudorandom generators for space-bounded computation. *Comb.*, 12(4):449–461, 1992.

Rasmus Pagh. Compressed matrix multiplication. *ACM Transactions on Computation Theory (TOCT)*, 5(3):1–17, 2013.

Anh-Huy Phan, Petr Tichavsky, and Andrzej Cichocki. Low complexity damped gauss–newton algorithms for candecomp/parafac. *SIAM Journal on Matrix Analysis and Applications*, 34(1): 126–147, 2013.

Lianke Qin, Rajesh Jayaram, Elaine Shi, Zhao Song, Danyang Zhuo, and Shumo Chu. Adore: Differentially oblivious relational database operators. *VLDB*, 2022a.

Lianke Qin, Aravind Reddy, Zhao Song, Zhaozhuo Xu, and Danyang Zhuo. Adaptive and dynamic multi-resolution hashing for pairwise summations. In *IEEE International Conference on Big Data, Big Data*, pp. 115–120. IEEE, 2022b.

Lianke Qin, Zhao Song, and Yitan Wang. Fast submodular function maximization. *arXiv preprint arXiv:2305.08367*, 2023a.

Lianke Qin, Zhao Song, Lichen Zhang, and Danyang Zhuo. An online and unified algorithm for projection matrix vector multiplication with application to empirical risk minimization. In *International Conference on Artificial Intelligence and Statistics*, pp. 101–156. PMLR, 2023b.

Lianke Qin, Zhao Song, and Ruizhe Zhang. A general algorithm for solving rank-one matrix sensing. In *International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 757–765. PMLR, 2024.

Alexander A. Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106 (2):385–390, 1992.

Aravind Reddy, Zhao Song, and Lichen Zhang. Dynamic tensor product regression. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS*, 2022.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy P. Lillicrap, Jean-Baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew M. Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, James Molloy, Jilin Chen, Michael Isard, Paul Barham, Tom Hennigan, Ross McIlroy, Melvin Johnson, Johan Schalkwyk, Eli Collins, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Clemens Meyer, Gregory Thornton, Zhen Yang, Henryk Michalewski, Zaheer Abbas, Nathan Schucher, Ankesh Anand, Richard Ives, James Keeling, Karel Lenc, Salem Haykal, Siamak Shakeri, Pranav Shyam, Aakanksha Chowdhery, Roman Ring, Stephen Spencer, Eren Sezener, and et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *CoRR*, abs/2403.05530, 2024. doi: 10.48550/ARXIV.2403.05530. URL `https://doi.org/10.48550/arXiv.2403.05530`.

John A Rhodes and Seth Sullivant. Identifiability of large phylogenetic mixture models. *Bulletin of mathematical biology*, 74:212–231, 2012.

Elina Robeva. Orthogonal decomposition of symmetric tensors. *SIAM Journal on Matrix Analysis and Applications*, 37(1):86–102, 2016.

Elina Robeva and Anna Seigal. Singular vectors of orthogonally decomposable tensors. *Linear and Multilinear Algebra*, 65(12):2457–2471, 2017.

Hemanth Saratchandran, Jianqiao Zheng, Yiping Ji, Wenbo Zhang, and Simon Lucey. Rethinking softmax: Self-attention with polynomial activations. *CoRR*, abs/2410.18613, 2024.

Sambal Shikhar, Mohammed Irfan Kurpath, Sahal Shaji Mullappilly, Jean Lahoud, Fahad Shahbaz Khan, Rao Muhammad Anwer, Salman H. Khan, and Hisham Cholakkal. Llmvox: Autoregressive streaming text-to-speech model for any LLM. In *Findings of the Association for Computational Linguistics, ACL*, pp. 20481–20493. Association for Computational Linguistics, 2025.

Charlie Snell, Ruiqi Zhong, Dan Klein, and Jacob Steinhardt. Approximating how single head attention learns. *arXiv preprint arXiv:2103.07601*, 2021.

Zhao Song, David P. Woodruff, and Huan Zhang. Sublinear time orthogonal tensor decomposition. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems (NIPS) 2016, December 5-10, 2016, Barcelona, Spain*, pp. 793–801, 2016.

Zhao Song, David P. Woodruff, and Peilin Zhong. Relative error tensor low rank approximation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pp. 2772–2789. SIAM, 2019.

Zhao Song, David Woodruff, Zheng Yu, and Lichen Zhang. Fast sketching of polynomial kernels of polynomial degree. In *International Conference on Machine Learning*, pp. 9812–9823. PMLR, 2021.

Zhao Song, Zhaozhuo Xu, and Lichen Zhang. Speeding up sparsification using inner product search data structures. *arXiv preprint arXiv:2204.03209*, 2022.

Zhao Song, Xin Yang, Yuanyuan Yang, and Lichen Zhang. Sketching meets differential privacy: Fast algorithm for dynamic kronecker projection maintenance. In *International Conference on Machine Learning, ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32418–32462. PMLR, 2023a.

Zhao Song, Mingquan Ye, and Lichen Zhang. Streaming semidefinite programs: $o(\sqrt{n})$ passes, small space and fast runtime. *Manuscript*, 2023b.

Zhao Song, Lichen Zhang, and Ruizhe Zhang. Training multi-layer over-parametrized neural network in subquadratic time. In *15th Innovations in Theoretical Computer Science Conference, ITCS*, volume 287 of *LIPIcs*, pp. 93:1–93:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

Foteini Strati, Sara McAllister, Amar Phanishayee, Jakub Tarnawski, and Ana Klimovic. Déjàvu: Kv-cache streaming for fast, fault-tolerant generative LLM serving. In *Forty-first International Conference on Machine Learning, ICML*. OpenReview.net, 2024.

William Swartworth, David P. Woodruff, and Samson Zhou. Perfect lp sampling with polylogarithmic update time. In *IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*, 2025.

Marina Thottan, Guanglei Liu, and Chuanyi Ji. Anomaly detection approaches for communication networks. In *Algorithms for Next Generation Networks*, Computer Communications and Networks, pp. 239–261. Springer, 2010.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Charalampos E. Tsourakakis. MACH: fast randomized tensor decompositions. In *SDM*, pp. 689–700, 2010.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Jeffrey Scott Vitter. Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57, 1985.

James Vuckovic, Aristide Baratin, and Remi Tachet des Combes. A mathematical theory of attention. *arXiv preprint arXiv:2007.02876*, 2020.

Chi Wang, Xueqing Liu, Yanglei Song, and Jiawei Han. Scalable moment-based inference for latent dirichlet allocation. In *ECML-PKDD*, pp. 290–305, 2014.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Yining Wang, Hsiao-Yu Tung, Alexander J Smola, and Anima Anandkumar. Fast and guaranteed tensor decomposition via sketching. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 991–999. https://arxiv.org/pdf/1506.04448, 2015.

Colin Wei, Yining Chen, and Tengyu Ma. Statistically meaningful approximation: a case study on approximating turing machines with transformers. *arXiv preprint arXiv:2107.13163*, 2021.

David P. Woodruff and Samson Zhou. Tight bounds for adversarially robust streams and sliding windows via difference estimators. In *62nd Annual Symposium on Foundations of Computer Science, FOCS*, pp. 1183–1196, 2021.

David P. Woodruff, Shenghao Xie, and Samson Zhou. Perfect sampling in turnstile streams beyond small moments. *Proc. ACM Manag. Data*, 3(2):106:1–106:27, 2025.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations, ICLR*. OpenReview.net, 2024.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. Duoattention: Efficient long-context LLM inference with retrieval and streaming heads. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

Yao Yao, Zuchao Li, and Hai Zhao. Sirllm: Streaming infinite retentive LLM. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*, pp. 2611–2624. Association for Computational Linguistics, 2024.

Guanghao Ye. Fast algorithm for solving structured convex programs. *The University of Washington, Undergraduate Thesis*, 2020.

Amir Zandieh, Insu Han, Majid Daliri, and Amin Karbasi. Kdeformer: Accelerating transformers via kernel density estimation. In *ICML*. arXiv preprint arXiv:2302.02451, 2023.

Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. Chatglm: A family of large language models from GLM-130B to GLM-4 all tools. *CoRR*, abs/2406.12793, 2024. doi: 10.48550/ARXIV.2406.12793. URL https://doi.org/10.48550/arXiv.2406.12793.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *Advances in Neural Information Processing Systems*, 33:15383–15393, 2020.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. Spargeattn: Accurate sparse attention accelerating any model inference. *CoRR*, abs/2502.18137, 2025. doi: 10.48550/ARXIV.2502.18137. URL https://doi.org/10.48550/arXiv.2502.18137.

Lichen Zhang. Speeding up optimizations via data structures: Faster search, sample and maintenance. Master's thesis, Carnegie Mellon University, 2022.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. ∞bench: Extending long context evaluation beyond 100k tokens. *CoRR*, abs/2402.13718, 2024. URL https://doi.org/10.48550/arXiv.2402.13718.

Haoyu Zhao, Abhishek Panigrahi, Rong Ge, and Sanjeev Arora. Do transformers parse while predicting the masked word? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023*, pp. 16513–16542. Association for Computational Linguistics, 2023.

Appendix

Roadmap. In Section A, we provide additional related work. In Section B, we briefly discuss the background on ℓ_2 sampler. In Section C, we show that how to use the tail bound to obtain sampling result. In Section D, we present the tensor sampling result. In Section E, we discuss the LLM usage of the paper.

A Additional Related Work

On sketching. The application of sketching and sampling techniques in numerical linear algebra has been remarkably effective, revolutionizing a broad spectrum of core tasks. These methods are crucial in linear programming (LP), as evidenced by Cohen et al. (2019); Jiang et al. (2021); Ye (2020); Gu & Song (2022), and have significantly impacted tensor approximation (Song et al., 2019; Mahankali et al., 2024; Deng et al., 2023a). Sketching and sampling techniques also have been widely applied in matrix completion (Gu et al., 2024), matrix sensing (Qin et al., 2024; Deng et al., 2023b), submodular function maximization (Qin et al., 2023a), dynamic sparsification (Deng et al., 2022a), dynamic tensor product regression (Reddy et al., 2022), and semi-definite programming (Song et al., 2023a). Additionally, sketching has been pivotal in iterative sparsification problems (Song et al., 2022), adversarial training (Gao et al., 2022), kernel density estimation (Qin et al., 2022b), solving the distance oracle problem (Deng et al., 2022b), and empirical risk minimization (Lee et al., 2019; Qin et al., 2023b). Its applications furthermore extends to relational databases (Qin et al., 2022a) and Large Language Model (LLM) research (Deng et al., 2023c;b; Gao et al., 2025; Li et al., 2023).

On theoretical attention. A comprehensive body of research, including studies (Child et al., 2019; Kitaev et al., 2020; Wang et al., 2020; Daras et al., 2020; Katharopoulos et al., 2020; Chen et al., 2021; 2022; Zandieh et al., 2023; Alman & Song, 2023; Brand et al., 2024; Deng et al., 2023c; Kacham et al., 2023; Alman & Song, 2024a; Han et al., 2024; Awasthi & Gupta, 2023; Marcus et al., 2022; Alman & Song, 2024b; 2025a;b), has progressively shed light on the complexities and optimization of attention matrix computation. This exploration has been further enriched by insights into the effectiveness of attention mechanisms in Transformers (Dehghani et al., 2018; Vuckovic et al., 2020; Zhang et al., 2020; Edelman et al., 2022; Snell et al., 2021; Wei et al., 2021; Deng et al., 2023d; 2024a). Among these, Zhao et al. (2023) revealed the adeptness of mid-scale masked language models in identifying syntactic elements, paving the way for innovations like partial parse tree reconstructions. Inspired the exponential mechanism in attention structure, Gao et al. (2023) provide an analysis which shows exponential regression within the over-parameterized neural tangent kernel framework can converge. In the over-constrained setting, several work show the convergence for attention inspired regression problem (Li et al., 2023; Deng et al., 2023b).

B ℓ_2 Sampler

We give the full details of the standard L_2 sampler from Jowhari et al. (2011); Mahabadi et al. (2020) in Algorithm 2. The proof of correctness is verbatim from Jowhari et al. (2011); Mahabadi et al. (2020). The challenge is how to implement the data structures of y, which is implicitly defined as $(A_1 \otimes A_2)x$. By comparison, in the standard setting of ℓ_2 samplers Monemizadeh & Woodruff (2010); Andoni et al. (2011); Jowhari et al. (2011); Jayaram & Woodruff (2021); Mahabadi et al. (2020), y is given as a data stream.

C From Tail to Sampling

Lemma C.1 (Restatement of Lemma 7.4)**.** *Let $y = (A_1 \otimes A_2)x \in \mathbb{R}^{n^2}$. Let only one of A_1 and A_2 be updated in streaming. Let $w = \frac{y_i}{\sqrt{u_i}}$ for a constant $u_i \in [0,1]$ generated uniformly at random. There is an algorithm $\mathcal{A}$ that that uses $O(nd) + \mathrm{poly}\left(\frac{1}{\epsilon}, \log n\right)$ space, uses $O(n)$ update time, and estimates each element of w up to additive error $\epsilon \cdot \|z\|_2$, where z denotes the tail vector of w without the largest $\frac{1}{\epsilon^2}$ entries in magnitude. Specifically, for all $i \in [n^2]$, we have $|\widehat{w}_i - w_i| \leq \epsilon \cdot \|z\|_2$.*

Algorithm 2 Standard ℓ_2 Sampler, e.g., extension of Jowhari et al. (2011) to $p = 2$

For each $i \in [n]$, let $u_i \in [0, 1]$ be chosen uniformly at random
$w_i \leftarrow \frac{y_i}{\sqrt{u_i}}$
Let z denote the tail vector of w without the largest $\frac{1}{\epsilon^2}$ entries in magnitude
Let $\widehat{Y}$ be a 2-approximation of $\|y\|_2$
Let $\widehat{Z}$ be a 2-approximation of $\|z\|_2$
$i \leftarrow \text{argmax}_{i\in[n]}|\widehat{w_i}|$
Let $C > 0$ be a large constant determined by the additive faliure probability $\frac{1}{\text{poly}(n)}$
if $\widehat{Z} > \sqrt{\frac{C \log n}{\epsilon}} \cdot \widehat{Y}$ or $|w_i| < \sqrt{\frac{C \log n}{\epsilon}} \cdot \widehat{Y}$ **then**
    Return FAIL
else
    Return i with estimate $\sqrt{u_i} \cdot \widehat{w_i}$
end if

Proof. Consider hash function $h_1, h_2 : [n] \to [b]$. Consider random sign functions $\sigma_1, \sigma_2 : [n] \to \{-1, +1\}$. We consider a fixed index $i_1, i_2 \in [n]$. Let $j = h_1(i_1) + h_2(i_2) \pmod b$. Let $h^{-1}(j)$ denote the all the pairs $(i_1, i_2) \in [n] \times [n]$ such that $h_1(i_1) + h_2(i_2) \pmod b = j$. Note that $\widehat{y}_i$ induced by h is $\widehat{w}_i = w_i + \sum_{l\in h^{-1}(j)\setminus\{i\}} s_i s_l w_{l_1} w_{l_2}$. For ease of presentation, we write $\sigma_i = \sigma_{1,i_1}\sigma_{2,i_2}$ and $\sigma_l = \sigma_{1,l_1}\sigma_{2,l_2}$.

$$\mathbb{E}[\widehat{w}_i] = \mathbb{E}\Big[w_i + \sum_{l\in h^{-1}(j)\setminus\{i\}} \sigma(i)\sigma(l)w_l\Big] = \mathbb{E}[w_i] + \sum_{l\in h^{-1}(j)\setminus\{i\}} \mathbb{E}[\sigma(i)\cdot\sigma(l)]\cdot w_l$$
$$= w_i + \sum_{l\in h^{-1}(j)\setminus\{i\}} \mathbb{E}[\sigma(i)]\cdot\mathbb{E}[\sigma(l)]\cdot w_l = w_i,$$

where the first step follows from definition, the second step follows from linearity of expectation, the third step follows from $\sigma(i)$ and $\sigma(l)$ are independent, the forth step follows from $\mathbb{E}[\sigma(l)] = 0$.

We now upper bound the variance of $\widehat{w}_i - y_i$ by analyzing $\mathbb{E}[(\widehat{y}_i)^2]$. Let $\mathcal{H}$ be the set of the top $\frac{1}{\epsilon^2}$ items and let $\mathcal{E}$ be the event that none of the items in $\mathcal{H}$ are mapped to $h(i)$, i.e., $h(a) \neq h(i)$ for all $a \in \mathcal{H}$.

Observe that for $b = \frac{100}{\epsilon^2}$, we have that $\Pr[\mathcal{E}] \geq 0.9$. Then we have:

$$\mathbb{E}[(\widehat{w}_i - w_i)^2 \mid \mathcal{E}] = \mathbb{E}[(\sum_{l\in[n]^2\setminus\mathcal{H}, l\in h^{-1}(j)} \sigma(i)\sigma(l)w_l)^2] = \mathbb{E}\left[\sum_{l\in[n]^2\setminus\mathcal{H}, l\in h^{-1}(j)} w_l^2\right]$$
$$= \frac{1}{b}\cdot \sum_{l\in[n]^2\setminus\mathcal{H}, l\in h^{-1}(j)} w_l^2 \leq \frac{1}{b}\cdot(w_1^2 + \ldots + w_{n^2}^2 - \sum_{l\in\mathcal{H}} w_l^2)$$
$$= 100\epsilon^2 \cdot \|z\|_2^2,$$

for $b = \frac{100}{\epsilon^2}$, since z is the vector corresponding to y that removes the entries in $\mathcal{H}$. By Chebyshev's inequality, we have that $\Pr[|\widehat{w}_i - w_i| \geq \epsilon \cdot \|z\|_2 \mid \mathcal{E}] \leq \frac{1}{10}$. Since $\Pr[\mathcal{E}] \geq 0.9$, then $\Pr|\widehat{w}_i - w_i| \geq \epsilon \cdot \|z\|_2 \leq 0.2$, for a fixed hash function h. By taking the median of $O(\log n)$ estimations corresponding to $O(\log n)$ different hash functions h, we have that $\Pr[|\widehat{w}_i - w_i| \geq \epsilon \cdot \|z\|_2] \leq \frac{1}{n^{10}}$. Thus by a union bound over $i \in [n] \times [n]$, we have that with probability at least $1 - \frac{1}{n^5}$, we have for all $i \in [n]$, $|\widehat{w}_i - w_i| \geq \epsilon \cdot \|z\|_2$. □

Lemma C.2 (Restatement of Lemma 7.5)**.** *Let $y = (A_1 \otimes A_2)x \in \mathbb{R}^{n^2}$ and let $w \in \mathbb{R}^{n^2}$ so that $w_i = \frac{y_i}{\sqrt{u_i}}$ for a constant $u_i \in [0, 1]$ generated uniformly at random. Let z denote the tail vector of w without the largest $\frac{1}{\epsilon^2}$ entries in magnitude. Let $\widehat{Z}$ be a 2-approximation to $\|z\|_2$ and $\widehat{Y}$ be a 2-approximation to $\|y\|_2$. Then*

$$\Pr\left[\widehat{Z} > \sqrt{\frac{C\log n}{\epsilon}}\cdot\widehat{Y}\right] \leq O(\epsilon) + \frac{1}{\text{poly}(n)}.$$

Proof. Let $\mathcal{E}_1$ denote the event that $\widehat{Z}$ is a 2-approximation to $\|z\|_2$ and $\widehat{Y}$ is a 2-approximation to $\|y\|_2$, so that

$$\Pr[\mathcal{E}_1] \geq 1 - \frac{1}{\text{poly}(n)}.$$

Conditioned on $\mathcal{E}_1$, it suffices to bound the probability that

$$4\|z\|_2 > \sqrt{\frac{C \log n}{\epsilon}} \cdot \|y\|_2.$$

Let $j \in [n^2]$ be a fixed index and let u_j be fixed.

Let $T = \sqrt{\epsilon} \cdot \|y\|_2$ and for each $i \in [n^2]$, we define the indicator random variable $W_i = 1$ if $|w_i| > T$ and $W_i = 0$ otherwise, if $|w_i| \leq T$. Note that W_i is an indicator random variable for whether the coordinate w_i in the vector w is "heavy" in magnitude.

We then define

$$Z_i = \frac{w_i^2}{T^2} \cdot (1 - W_i)$$

to be the scaled contribution of the small entries of z, and observe that $Z_i \in [0, 1]$.

Let

$$W = \sum_{i \in [n^2], i \neq j} w_i$$

denote the total number of heavy indices besides possibly index j and $Z = \sum_{i \in [n^2], i \neq j} Z_i$ denote the total scaled contribution of the light indices besides possibly index j. Let v denote the vector containing the heavy indices, so that $v_i = w_i$ for $W_i = 1$ and $v_i = 0$ otherwise for $W_i = 0$. Note that v has sparsity at most $Y + 1$ and moreover $U^2 Z = \|w - v\|_2^2$. We also have that $\|z\|_2 \leq \|w - v\|_2$ unless $W \geq \frac{2}{\epsilon^2}$.

Let $\mathcal{E}_2$ denote the event that $W \geq \frac{2}{\epsilon^2}$ and let $\mathcal{E}_3$ denote the event that $Z \geq \frac{C \log n}{16 T^2 \epsilon} \cdot \|y\|_2^2$. Observe that if neither $\mathcal{E}_2$ nor $\mathcal{E}_3$ occur, then we have $4\|z\|_2 \leq \sqrt{\frac{C \log n}{\epsilon}} \cdot \|y\|_2$, as desired. Thus it remains to bound the probability of the failure events $\mathcal{E}_2$ and $\mathcal{E}_3$.

We have $\mathbb{E}[W_i] = \frac{\|w\|_2^2}{T^2}$, so that $\mathbb{E}[W] \leq \frac{1}{\epsilon}$. By Markov's inequality, we have that $\Pr[\mathcal{E}_2] \leq \frac{\epsilon}{2}$.

We now upper bound $\Pr[\mathcal{E}_3]$. Recall that $Z_i = \frac{w_i^2}{T^2} \cdot (1 - W_i) = \frac{w_i^2}{T u_i^2} \cdot (1 - W_i)$, since $w_i = \frac{y_i}{\sqrt{u_i}}$. Observe that $Z_i > 0$ only if $|w_i| < T$, i.e., if $u_i \geq \frac{y_i^2}{\epsilon \cdot \|y\|_2^2}$, since $T = \sqrt{\epsilon} \cdot \|y\|_2$. For $\epsilon \in (0, 1)$, we thus have

$$\begin{aligned}
\mathbb{E}[Z_i] &\leq \int_{y_i^2/\|y\|_2^2}^{1} z_i \mathrm{d}u_i \\
&= \int_{y_i^2/\|y\|_2^2}^{1} \frac{y_i^2}{u_i} \frac{1}{T^2} \mathrm{d}u_i.
\end{aligned}$$

Now, let $\mathcal{E}_4$ be the event that $u_i \geq \frac{1}{n^{C/2}}$ for all $i \in [n^2]$, so that $\Pr[\mathcal{E}_4] \geq 1 - \frac{1}{n^{C/2-2}}$.

Then

$$\begin{aligned}
\mathbb{E}[Z_i \mid \mathcal{E}_4] &\leq \frac{1}{1 - \frac{1}{n^{C/2-2}}} \int_{1/n^{C/2}}^{1} \frac{y_i^2}{u_i} \frac{1}{T^2} \mathrm{d}u_i \\
&\leq \frac{C \log n}{T^2} y_i^2.
\end{aligned}$$

Thus, we have

$$\begin{aligned}\mathbb{E}[Z \mid \mathcal{E}_4] &= \sum_{i\in[n^2]} \mathbb{E}[Z_i \mid \mathcal{E}_4] \\ &= \sum_{i\in[n^2]} \frac{C\log n}{T^2} y_i^2 \\ &\le \sum_{i\in[n^2]} \frac{C\log n}{\epsilon} \frac{y_i^2}{\|y\|_2^2} \\ &= \frac{C\log n}{\epsilon}.\end{aligned}$$

Thus by Markov's inequality, the probability that Z is larger than $\frac{C\log n}{16T^2\epsilon}\cdot\|y\|_2^2 = \frac{C\log n}{16\epsilon^2}$ is at most $\frac{\epsilon}{16}$. The claim then follows from taking a union bound over the events $\mathcal{E}_1, \neg\mathcal{E}_2, \neg\mathcal{E}_3, \neg\mathcal{E}_4$. □

D TENSOR SAMPLING

Theorem D.1 (Restatement of Theorem 7.6). *Let $y = (A_1 \otimes A_2)x \in \mathbb{R}^{n^2}$ and let $w \in \mathbb{R}^{n^2}$ so that for each $i \in [n^2]$, $w_i = \frac{y_i}{\sqrt{u_i}}$ for a constant $u_i \in [0,1]$ generated uniformly at random. Let z denote the tail vector of w without the largest $\frac{1}{\epsilon^2}$ entries in magnitude. Suppose there exists:*

1. *An algorithm $\mathcal{A}_1$ that provides a 2-approximation to $\|y\|_2$ with probability $1 - \frac{1}{n^2}$.*
2. *An algorithm $\mathcal{A}_2$ that provides a 2-approximation to $\|z\|_2$ with probability $1 - \frac{1}{n^2}$.*
3. *An algorithm $\mathcal{A}_3$ that estimates each element of w up to additive error $\epsilon\cdot\|z\|_2$,*
$$|\widehat{w_i} - w_i| \le \epsilon\cdot\|z\|_2,$$
for all $i \in [n^2]$.

Then there exists a data structure that uses $\mathrm{poly}\left(\frac{1}{\epsilon}, \log n\right)$ bits of space and outputs each index i with probability p_i, such that

$$(1-\epsilon)\cdot\frac{y_i^2}{\|y\|_2^2} - \frac{1}{\mathrm{poly}(n)} \le p_i \le (1+\epsilon)\cdot\frac{y_i^2}{\|y\|_2^2} + \frac{1}{\mathrm{poly}(n)}.$$

Proof. Let i be fixed and let $\mathcal{E}$ denote the event that $u_i < \frac{\epsilon}{C\log n}\frac{y_i^2}{\widehat{Y}^2}$, so that $|w_i| > \sqrt{\frac{C\log n}{\epsilon}}\cdot\widehat{Y}$.

Let $\mathcal{E}_1$ denote the event that $\widehat{Y}$ is a 2-approximation to $\|y\|_2$, $\widehat{Z}$ is a 2-approximation to $\|z\|_2$, and $|\widehat{w_i} - w_i| \le \epsilon\cdot\|z\|_2$ for all $i \in [n]$. Let $\mathcal{E}_2$ denote the event that $\widehat{Z} > \sqrt{\frac{C\log n}{\epsilon}}\cdot\widehat{Y}$ and let $\mathcal{E}_3$ denote the event that multiple indices j satisfy $|w_j| > \sqrt{\frac{C\log n}{\epsilon}}\cdot\widehat{Y}$. Finally, let $\mathcal{E}_4$ denote the event that $|\widehat{w_i}| < \sqrt{\frac{C\log n}{\epsilon}}\cdot\widehat{Y}$.

Intuitively, $\mathcal{E}_1$ is a good event, i.e., correctness of the data structures, which we would like to hold. On the other hand, $\mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4$ are bad events that distort the sampling probabilities, which we would like to avoid.

We first note that $\mathcal{E}_1$ holds with high probability due to the correctness of the CountSketch and L_2-norm estimation data structures. We next note that by Lemma 7.5, the probability that $\mathcal{E}_2$ occurs is $O(\epsilon)$.

Next, note that the probability that for a fixed $j \in [n]$, u_j satisfies $\frac{y_j^2}{u_j} \ge \frac{C\log n}{\epsilon}\cdot\widehat{Y}$ is at most $\frac{\epsilon}{C'\log n}\frac{y_j^2}{\|y\|_2^2}$ for some constant C'. Thus summing over all $j \in [n]$, the probability that there exist an additional $j \in [n]$ for which $|w_j| > \sqrt{\frac{C\log n}{\epsilon}}\cdot\widehat{Y}$ is $O(\epsilon)$. Thus the probability that $\mathcal{E}_3$ occurs is $O(\epsilon)$.

Finally, conditioned on $\neg\mathcal{E}_2$, we have that $\widehat{Z} \leq \sqrt{\frac{C\log n}{\epsilon}} \cdot \widehat{Y}$. Then conditioning on $\mathcal{E}_1$, we have $\|z\|_2 \leq \widehat{Z}$ and thus $|\widehat{w_i} - w_i| \leq \epsilon\widehat{Z} \leq \sqrt{C\epsilon\log n}\widehat{Y}$, so that $\mathcal{E}_4$ can only occur for $\sqrt{\frac{C\log n}{\epsilon}} \cdot \widehat{Y} \leq |w_i| \leq \sqrt{\frac{C\log n}{\epsilon}} \cdot \widehat{Y}$, which is at most probability $O\left(\frac{\epsilon^2}{C\log n}\frac{y_i^2}{\widehat{Y}^2}\right)$, over the randomness of u_i.

In summary, we observe that conditioned on some value being output, the probability that item i is selected is proportional to the event that the events $\mathcal{E}$ and $\mathcal{E}_1$ occur, and none of the events $\mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4$ occur. The probability that $\mathcal{E}$ occurs is $\frac{\epsilon}{C\log n}\frac{y_i^2}{\widehat{Y}^2}$, which u_i is chosen uniformly at random. Due to the event $\mathcal{E}_1$, the sampling probability is distorted additively by $\frac{1}{\text{poly}(n)}$, while due to the events $\mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4$, the sampling probability is distorted multiplicatively by $(1+\epsilon)$. Thus conditioned on the event that some index is returned, the probability p_i that index i is returned satisfies

$$(1-\epsilon)\cdot\frac{y_i^2}{\|y\|_2^2} - \frac{1}{\text{poly}(n)} \leq p_i \leq (1+\epsilon)\cdot\frac{y_i^2}{\|y\|_2^2} + \frac{1}{\text{poly}(n)},$$

as desired. □

E LLM USAGE DISCLOSURE

LLMs were used only to polish language, such as grammar and wording. These models did not contribute to idea creation or writing, and the authors take full responsibility for this paper's content.