LAMP: Look-Ahead Mixed-Precision Inference of Large Language Models

Stanislav Budzinskiy *
Faculty of Mathematics
University of Vienna, Austria

Marian Gloser
Faculty of Mathematics
University of Vienna, Austria

Tolunay Yilmaz
Faculty of Mathematics
University of Vienna, Austria

Ying Hong Tham
Huawei Technologies

Yuanyi Lin
Huawei Technologies

Wenyi Fang
Huawei Technologies

Fan Wu
Huawei Technologies

Philipp Petersen
Faculty of Mathematics
University of Vienna, Austria

Abstract

Mixed-precision computations are a hallmark of the current stage of AI, driving the progress in large language models towards efficient, locally deployable solutions. This article addresses the floating-point computation of compositionally-rich functions, concentrating on transformer inference. Based on the rounding error analysis of a composition $f(g(\mathbf{x}))$, we provide an adaptive strategy that selects a small subset of components of $g(\mathbf{x})$ to be computed more accurately while all other computations can be carried out with lower accuracy. We then explain how this strategy can be applied to different compositions within a transformer and illustrate its overall effect on transformer inference. We study the effectiveness of this algorithm numerically on GPT-2 models and demonstrate that already very low recomputation rates allow for improvements of up to two orders of magnitude in accuracy.

1 Introduction

Transformer deep neural networks (DNNs), originally introduced for sequence modeling in natural language processing [29], have become a standard computational paradigm across a wide range of domains, including language understanding [12] as well as vision and multimodal learning [13]. From a computational perspective, a transformer is a deep composition of simple operators, obtained by iterating attention-based mappings and pointwise nonlinearities across layers.

In practice, the evaluation of such deep compositions (*inference*) is performed in floating-point (FP) arithmetic, low-precision formats being routinely employed to improve performance and energy efficiency [19, 25, 21]. From a numerical analysis standpoint, FP evaluation introduces rounding errors at every stage of the computation; the cumulative effect of these local errors hinges on how they propagate through successive compositions of operators [20].

The bulk of operators in transformers are matrix products, and the existing approaches to mixed-precision inference largely address them based on two key principles: the input is quantized to low precision, and the output is accumulated in high precision [30]. Typically, the quantization precision is uniform across the whole input, and the accumulation precision is uniform across the whole output.

*Corresponding author: stanislav.budzinskiy@univie.ac.at

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

While there are also mixed-precision quantization strategies [11], we are not aware of developments in *mixed-precision accumulation* for transformer inference.

In this work, we address this gap by proposing a mathematically principled mixed-precision inference strategy that is explicitly aware of compositional effects. Instead of treating the output of an intermediate computation uniformly, we follow a *look-ahead strategy* by flagging and recomputing with higher accuracy those computations whose round-off errors will be most strongly amplified by the ensuing operator. Our method has a rigorous theoretical foundation and delivers strong empirical results in numerical experiments, allowing for improvements of up to two orders of magnitude in accuracy with only 1% of recomputations. We describe our contribution in detail in Subsection 1.2.

1.1 Rounding error analysis

To place our contribution in context, we briefly review how rounding error analysis has been applied to function evaluation, and why DNNs fall outside this classical setting. The primary applications are to matrix computations [20, 9] and "basic" nonlinear functions such as the *elementary functions*: powers and roots, exponentials and logarithms, trigonometric and hyperbolic functions [26].

Special functions [17] are "basic" in mathematical physics, defined as solutions to specific differential equations or integrals. The complexity of their evaluation stems from the need to discretize the differential equation or integral to sufficient (typically very high) precision, after which rounding effects of FP arithmetic become noticeable [23].

DNNs, including transformers, can be considered "basic" in the field of AI, and they differ from the aforementioned "basic" functions in two aspects: the target accuracy[2] and the source of evaluation complexity. First, the applications of DNNs do not seem to require the accuracy to be very high. Second, the evaluation complexity of DNNs is due to their extremely rich *compositional structure*, where the "building blocks" are either elementary functions or their compositions [3, 15].

Therefore, *the algorithms of DNN inference need to pay specific attention to compositions*. Rounding error analysis applied to DNNs leads to worst-case bounds that grow exponentially with depth [14, 6]. While the *global* rounding error appears to be difficult to tame, the *local* rounding error at each composition is easier to control—the core idea of our work.

1.2 Contributions, limitations, and outlook

Our key contribution is methodological: we identify *a new paradigm for mixed-precision evaluation in transformer inference*, placing it in the accumulation of matrix products and thereby complementing the standard approach to mixed-precision matrix multiplication rooted in the quantization of operands. The attainable gains within this novel framework are validated in numerical experiments, the results of which simultaneously serve as a proof of concept for our idea and a benchmark for future practical inference kernels designed in accordance with it. In addition to empirical accuracy improvements, our results highlight the importance of compositional numerical effects in deep models.

Novel computational framework: We propose an adaptive approach to FP evaluation of compositions called *look-ahead mixed precision* (LAMP). This method stems from a rigorous theoretical derivation and is formulated explicitly as an optimization problem (5), which aims to determine a sparse subset of the inner function's components that need to be recomputed more accurately to ensure the numerical stability of the composition. Find the details in Section 2.

Particularly well-suited to transformers: A theoretical investigation of the nonlinearities inherent to the transformer architecture allows us to prove that the aforementioned *selection procedure admits closed-form solutions*, overcoming the otherwise combinatorial nature of the problem and paving the way for practical implementations. See Section 3.

Proof of concept: For validation, we apply LAMP to evaluate softmax in attention. Our method demonstrates convincing performance in numerical experiments with the GPT-2 XL model. In Figure 1, the key-query (KQ) inner products are accumulated using μ mantissa bits, and 0.9% of adaptively selected inner products are recomputed in FP32 (corresponding to a threshold $\tau = 0.1$ in Sections 2-3). Notably, BF16 accumulation ($\mu = 7$) with adaptive recomputation deviates from uniform FP32 accumulation just as much as uniform TF32 accumulation ($\mu = 10$). Meanwhile, the

[2] How many unit round-offs of error are tolerable, rather than the value of the unit round-off.

effective number of mantissa bits used per KQ inner product is about 8.9 for the former against 10 for the latter.[3] In addition, Figure 1 shows that our method does indeed select critical-for-performance inner products, since the same number of random recomputations has no effect. Furthermore, the "look-ahead" nature of our approach is vital: a non-look-ahead strategy that recomputes the most *intrinsically* ill-conditioned inner products (those with a large ratio $|\mathbf{x}|^\intercal|\mathbf{y}|/|\mathbf{x}^\intercal\mathbf{y}|$) fails to deliver gains under the same recomputation budget. Find a detailed description of our experiments and more numerical results in Section 4 and Appendix C.

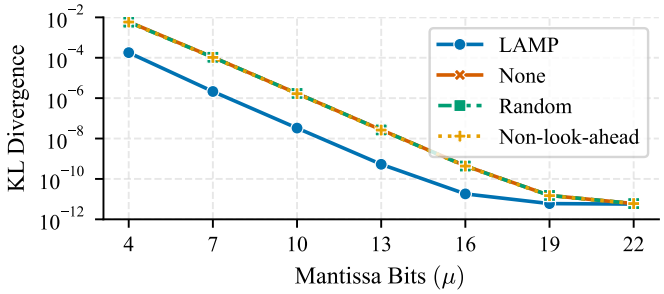

Figure 1: Performance of mixed-precision GPT-2 XL inference on the OpenWebText dataset with the proposed LAMP evaluation of KQ inner products. Only 0.9% of them are recomputed in FP32. The Kullback–Leibler divergence is measured against a reference model with uniform FP32 accumulation.

Not sparse attention: Unlike sparse attention [2, 22], which achieves efficiency by discarding low-scoring tokens without regard for the resulting probability mass shift, LAMP computes these scores in low precision. By dynamically routing high FP precision only to numerically sensitive tokens, LAMP strictly bounds the worst-case rounding error of the softmax distribution.

Benchmark: In this paper, we apply the LAMP framework to the *offline* computation of softmax, and therefore the analysis and the resulting algorithm do not directly extend to the *online*-softmax mechanism of FlashAttention [10] [4]. Consequently, our proposed algorithm, designed strictly based on the solution of the optimization problem (5) for the softmax function, falls short in long-context scenarios, because the entire vector of softmax probabilities needs to be materialized in memory. Hence, we treat the "strict" LAMP algorithm for softmax as a benchmark that achieves an optimal balance between the recomputation rate and the accuracy gain.

Practical LAMP attention: To compete with the state of the art, the "strict" LAMP needs to be relaxed in order to become compatible with the online softmax. While such modification is a topic of our future work, we propose here a crucial intermediate step in this direction in the form of *relaxed relative-threshold* LAMP (Subsection 4.4). Our preliminary experiments indicate a marginal loss in performance compared to the "strict" LAMP.

Hardware reliance: The numerical experiments carried out in this article simulate low-precision accumulation to allow for high granularity of mantissa bit-widths. In a practical setting, it would be beneficial to pack multiple low-precision KQ inner products in a single 32-bit register (e.g., four 8-bit accumulators) and thereby increase the throughput. As far as we know, current accelerators do not support native matrix-product accumulation in such packed formats. Therefore, our LAMP framework provides a mathematically grounded blueprint for next-generation AI accelerators that natively support packed low-precision accumulation. For the same reason, we do not present runtime comparisons between our method and the state-of-the-art kernels tailored to modern hardware.

Complementary idea to quantization: Let us stress that quantization is *not* addressed in our work. The proposed approach does not modify the operands of matrix products and does not reduce the memory footprint of a transformer [16]—we aim to control the accuracy of *computations* instead. Consequently, our method should be viewed as complementary to quantization techniques.

Extension to other architectures: The proposed LAMP inference strategy is derived from a theoretical analysis of the core building blocks of transformers. While the underlying idea applies

[3] $1 \cdot 7 + 0.083 \cdot 23 = 8.909$

[4] Though the LAMP framework itself is not fundamentally incompatible with FlashAttention; see a follow-up work [7]

to arbitrary DNNs, the feasibility of its implementation is architecture-dependent. Extending our approach to other DNN architectures would require a similar theoretical analysis of their constituent operations. Given the strong empirical results for transformers, we view this as a natural direction for future work (including the simultaneous LAMP evaluation of all transformer nonlinearities).

Extension to training: While the focus of this article is on inference, the same LAMP principle can be applied to compositions taking place during the backward pass. This is a topic for future work.

Validation on larger models: Our experiments focus on the family of GPT-2 transformer models. While these models are moderate in size by current standards, they already exhibit the characteristic depth and compositional structure of modern transformers. Importantly, the numerical effects targeted by our method are present at this scale, suggesting that they are likely to become more pronounced in larger architectures where softmax probabilities are expected to be more concentrated (a comparison of GPT-2 XL and GPT-2 small models in Appendix C.2 supports this claim). Consequently, this paper establishes a theoretical foundation and a behavioral proof of concept for LAMP.

2 Floating-point evaluation of compositions

Let $m, n, k \in \mathbb{N}$ and consider functions: $g : \mathbb{R}^k \to \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}^m$. Given $\mathbf{x} \in \mathbb{R}^k$, our goal is to evaluate the composition $f(g(\mathbf{x}))$ in FP arithmetic.

2.1 Baseline evaluation

Let $\mathbf{y} = g(\mathbf{x}) \in \mathbb{R}^n$ be the exact output of the inner function, and let $\hat{\mathbf{y}} \in \mathbb{R}^n$ be its computed value. Assuming that g is evaluated with a mixed forward-backward stable algorithm [20], rounding error analysis guarantees that the relative error is bounded by

$$\frac{|\hat{\mathbf{y}} - \mathbf{y}|}{|\mathbf{y}|} \leq \mathbf{c}_g \mathrm{u}_g + \mathcal{O}(\mathrm{u}_g^2), \quad \mathbf{c}_g \in \mathbb{R}^n_+, \tag{1}$$

where the absolute value, division, and comparison of vectors are componentwise, and u_g denotes the unit round-off. The nonnegative vector $\mathbf{c}_g$ describes how round-off errors are magnified during the computation of each component of $\mathbf{y}$ and depends on the rounding mode (e.g., deterministic or stochastic) and the specific algorithm used to evaluate g.

Next, $\hat{\mathbf{y}}$ is used as input for f, which is evaluated in precision u_f. Denoting by $\hat{\mathbf{z}} \in \mathbb{R}^m$ the computed value of $f(\hat{\mathbf{y}})$, we get by the same logic as in (1) that

$$\frac{|\hat{\mathbf{z}} - f(\hat{\mathbf{y}})|}{|f(\hat{\mathbf{y}})|} \leq \mathbf{c}_f \mathrm{u}_f + \mathcal{O}(\mathrm{u}_f^2), \quad \mathbf{c}_f \in \mathbb{R}^m_+.$$

However, we need to compare $\hat{\mathbf{z}}$ with the exact $\mathbf{z} = f(\mathbf{y})$. By triangle inequality,

$$\frac{|\hat{\mathbf{z}} - \mathbf{z}|}{|\mathbf{z}|} \leq \frac{|\hat{\mathbf{z}} - f(\hat{\mathbf{y}})| + |f(\hat{\mathbf{y}}) - \mathbf{z}|}{|f(\hat{\mathbf{y}})|} \Big(1 + \mathcal{O}(\mathrm{u}_f)\Big).$$

We bound the first term by $\mathbf{c}_f \mathrm{u}_f$ to first order. To bound the second term, assume that f is sufficiently regular in the neighborhood of $\hat{\mathbf{y}}$ and use Taylor's theorem to get a bound in terms of the Jacobian:

$$\frac{|f(\hat{\mathbf{y}}) - f(\mathbf{y})|}{|f(\hat{\mathbf{y}})|} \leq \frac{|\mathbf{J}_f(\hat{\mathbf{y}}) \mathrm{diag}(\hat{\mathbf{y}}) | \mathbf{c}_g}{|f(\hat{\mathbf{y}})|} \mathrm{u}_g + \mathcal{O}(\mathrm{u}_g^2),$$

where $\mathbf{J}_f(\hat{\mathbf{y}}) \in \mathbb{R}^{m \times n}$ and $\mathrm{diag}(\hat{\mathbf{y}}) \in \mathbb{R}^{n \times n}$ is diagonal. As a result, we get

$$\frac{|\hat{\mathbf{z}} - \mathbf{z}|}{|\mathbf{z}|} \leq \mathbf{c}_f \mathrm{u}_f + \frac{|\mathbf{J}_f(\hat{\mathbf{y}}) \mathrm{diag}(\hat{\mathbf{y}}) | \mathbf{c}_g}{|f(\hat{\mathbf{y}})|} \mathrm{u}_g + \mathcal{O}(\mathrm{u}_f^2 + \mathrm{u}_g^2). \tag{2}$$

2.2 Refined evaluation

The bound (2) follows from (1) and therefore depends on $\mathbf{c}_g$ (i.e., the evaluation algorithm of g) and the assumption that every component of g is computed in precision u_g. Consider a more flexible setting where we may want to compute some of the components more accurately—with a more accurate algorithm or in higher precision—to ensure that the second term in (2) is not too large. Let the nonzeros of $\mathbf{q} \in \{0, 1\}^n$ encode these components.

2.2.1 More accurate algorithm

We shall say that an evaluation algorithm is more accurate than the baseline if it leads to smaller entries in $\mathbf{c}_g$. Let us denote by $0 \leq \epsilon \leq 1$ the corresponding gain factor. Then the bound (1) becomes

$$\frac{|\hat{\mathbf{y}} - \mathbf{y}|}{|\mathbf{y}|} \leq \big(\mathbf{I} - \mathrm{diag}(\mathbf{q})\big)\mathbf{c}_g \mathrm{u}_g + \mathcal{O}(\mathrm{u}_g^2 + \epsilon \mathrm{u}_g),$$

leading to a modification of the second term in (2):

$$\frac{|\mathbf{J}_f(\hat{\mathbf{y}})\mathrm{diag}(\hat{\mathbf{y}})|\big(\mathbf{I} - \mathrm{diag}(\mathbf{q})\big)\mathbf{c}_g}{|f(\hat{\mathbf{y}})|}\mathrm{u}_g + \mathcal{O}(\mathrm{u}_g^2 + \epsilon \mathrm{u}_g).$$

For example, let $g(\mathbf{x}) = \mathbf{A}\mathbf{x}$ with $\mathbf{A} \in \mathbb{R}^{n \times k}$ and $\mathbf{x} \in \mathbb{R}^k$ be stored in precision u_g. Evaluating g in precision u_g, the basic multiplication algorithm has $\mathbf{c}_g = k\frac{|\mathbf{A}||\mathbf{x}|}{|\mathbf{A}\mathbf{x}|}$ for deterministic rounding [20] and $\mathbf{c}_g \lesssim \sqrt{k}\frac{|\mathbf{A}||\mathbf{x}|}{|\mathbf{A}\mathbf{x}|}$ with high probability for stochastic rounding [9]. Mixed-precision algorithms based on fused multiply-add can achieve $\mathbf{c}_g = \frac{|\mathbf{A}||\mathbf{x}|}{|\mathbf{A}\mathbf{x}|}$ [4], and similarly for compensated-summation algorithms. Therefore, the gain factor ϵ can be made small for large k.

2.2.2 Higher precision

As an alternative to a more accurate algorithm, we can use FP precision u_g^2 to compute and store the selected components. Then (1) turns into

$$\frac{|\hat{\mathbf{y}} - \mathbf{y}|}{|\mathbf{y}|} \leq \big(\mathbf{I} - \mathrm{diag}(\mathbf{q})\big)\mathbf{c}_g \mathrm{u}_g + \mathcal{O}(\mathrm{u}_g^2)$$

and leads to a similar modification in (2). This requires the evaluation algorithm of f to process mixed-precision inputs. Notably, mixed-precision matrix multiplication [4] achieves $\mathbf{c}_g = \mathbf{0}$ when the output is stored in precision u_g^2, since no extra rounding is done at the end.

2.3 Look-ahead mixed-precision evaluation

The two refinements in Subsection 2.2 lead to very similar componentwise rounding error bounds for the composition. The accuracy of the inner computation manifests itself in the second term, which we aim to reduce as follows. Denote

$$\mathbf{K}(f, \hat{\mathbf{y}}) = \mathbf{J}_f(\hat{\mathbf{y}})\mathrm{diag}(\hat{\mathbf{y}}), \quad \mathbf{M}(f, \hat{\mathbf{y}}) = \mathrm{diag}(f(\hat{\mathbf{y}}))^{-1}\mathbf{K}(f, \hat{\mathbf{y}}),$$

then we seek a binary vector $\mathbf{q} \in \{0, 1\}^n$ such that

$$\kappa_c(f, \hat{\mathbf{y}}; \mathbf{q}) = \big\|\mathbf{M}(f, \hat{\mathbf{y}})(\mathbf{I} - \mathrm{diag}(\mathbf{q}))\big\|_{\infty,\infty} \leq \tau \tag{3}$$

for a given threshold $\tau \geq 0$. While this objective targets the componentwise error for the composition, a similar derivation for an ℓ_p-based combination of input-output error metrics leads to

$$\kappa_p(f, \hat{\mathbf{y}}; \mathbf{q}) = \frac{\|\mathbf{K}(f, \hat{\mathbf{y}})(\mathbf{I} - \mathrm{diag}(\mathbf{q}))\|_{\mathrm{p,p}}}{\|f(\hat{\mathbf{y}})\|_{\mathrm{p}}} \leq \tau. \tag{4}$$

Let us discuss these objectives:

- Objective (3) improves the bound (2). Specifically, the second term in (2) can be bounded by $\mathbf{1}\tau\|\mathbf{c}_g\|_\infty \mathrm{u}_g$, and (4) has a similar effect on the corresponding rounding error bound.
- Both (3) and (4) can always be attained with $\mathbf{q} = \mathbf{1}$, i.e., when every component of the inner function g is computed more accurately.
- When $\mathbf{q} = \mathbf{0}$, the quantity $\kappa_c(f, \hat{\mathbf{y}}; \mathbf{q})$ is the componentwise condition number of f[18].
- The exact $f(\hat{\mathbf{y}})$ and $\mathbf{J}_f(\hat{\mathbf{y}})$ are not known in practice, so their computed values will be used.

If the baseline $\hat{\mathbf{y}}$ with $\mathbf{q} = \mathbf{0}$ satisfies (3) or (4), we deem the computation complete. Otherwise, we try a different $\mathbf{q}$, which entails the recomputation of $\hat{\mathbf{y}}$. Because such recomputations can be costly, we shall assume that the Jacobian is stable[5] with respect to small variations in $\hat{\mathbf{y}}$ and fix

$$\kappa \in \{\mathbf{q} \mapsto \kappa_c(f, \hat{\mathbf{y}}; \mathbf{q}),\ \mathbf{q} \mapsto \kappa_p(f, \hat{\mathbf{y}}; \mathbf{q})\}$$

corresponding to the baseline $\hat{\mathbf{y}}$. Having found a suitable $\mathbf{q}$ that satisfies the bound $\kappa(\mathbf{q}) \leq \tau$, we will recompute more accurately those components of $\hat{\mathbf{y}}$ that are indexed by the nonzeros of $\mathbf{q}$. To minimize the number of recomputations, we require $\mathbf{q}$ to be sparse. This leads to our *LAMP problem*:

$$\|\mathbf{q}\|_0 \to \min \quad \text{s.t.} \quad \kappa(\mathbf{q}) \leq \tau. \tag{5}$$

LAMP evaluation of a composition is summarized in Algorithm 1. We shall call the LAMP problem (5) *componentwise* or ℓ_p*-based* depending on the underlying objective.

Algorithm 1 LAMP evaluation of a composition

Input: functions f and g, variable $\mathbf{x}$, threshold $\tau \geq 0$, componentwise or normwise objective
Output: adaptively computed value $\hat{\mathbf{y}}$ of $g(\mathbf{x})$
Compute $\hat{\mathbf{y}} \approx g(\mathbf{x})$ in FP arithmetic.
Set up the function κ in FP arithmetic.
Find a solution $\mathbf{q}$ of the LAMP problem (5).
Recompute components of $\hat{\mathbf{y}}$ indexed by nonzeros of $\mathbf{q}$ more accurately.

3 Application to transformer inference

Algorithm 1 relies on the possibility to compute individual components of g separately. Yet this is not always possible. For instance, consider the softmax function

$$\mathrm{softmax}(\mathbf{x}) = \left[\frac{\exp(x_1)}{\sum_{i=1}^k \exp(x_i)} \quad \cdots \quad \frac{\exp(x_k)}{\sum_{i=1}^k \exp(x_i)}\right]^\mathsf{T}. \tag{6}$$

If at least one component needs to be computed more accurately, every component has to be as well. Meanwhile, the required property holds for matrix-vector products: it suffices to divide the matrix into two blocks and use different multiplication algorithms for each. When a bias term is present, it can be added to the accumulator in the required precision as well.

Matrix products are used in all DNN architectures. We thus apply LAMP to the following question:

How to adapt matrix multiplication to the ensuing nonlinearity during DNN inference?

What is the "ensuing nonlinearity" in question? To target the accuracy of the end result of inference, an obvious choice is to set f to be the remaining tail of the DNN. However, the computation of the associated $\kappa(\mathbf{q})$ and the solution of the LAMP problem (5) are highly demanding.

We focus on the *elementary transformer nonlinearities* (activation functions, layer normalization, softmax) and show that their LAMP problems (5) admit *closed-form solutions*.

3.1 Activation functions

Consider an activation function $\varphi : \mathbb{R} \to \mathbb{R}$. In DNNs, activation functions are applied entrywise:

$$f : \mathbb{R}^n \to \mathbb{R}^n, \quad f(\mathbf{y}) = [\varphi(y_1) \quad \cdots \quad \varphi(y_n)]^\mathsf{T}.$$

As a consequence, the matrix $\mathbf{M}$ is diagonal,

$$\mathbf{M}(f, \mathbf{y}) = \mathrm{diag}\Big(\tfrac{\varphi'(y_1)}{\varphi(y_1)} y_1, \ldots, \tfrac{\varphi'(y_n)}{\varphi(y_n)} y_n\Big),$$

and the solution $\mathbf{q}$ of the componentwise LAMP problem (5) can be written in closed form: an entry of $\mathbf{q}$ is nonzero if and only if the corresponding diagonal entry of $\mathbf{M}$ exceeds τ in absolute value. The diagonal structure of the Jacobian makes it possible to solve (5) immediately.

For activation functions, LAMP evaluation essentially repeats the mixed-precision accumulation of [14]. The novelty of LAMP is generality: it applies to arbitrary compositions, and we prove that transformer-specific compositions are particularly suitable for LAMP evaluation.

[5] If the Jacobian changes rapidly, the Hessian would need to be included in the analysis.

3.2 Layer normalization

Layer normalization aims to stabilize the training of DNNs [1] by "standardizing" its input via[6]

$$f : \mathbb{R}^n \to \mathbb{R}^n, \quad f(\mathbf{y}) = \sqrt{n}\frac{\mathbf{y} - \frac{1}{n}\mathbf{1}\mathbf{1}^\intercal\mathbf{y}}{\|\mathbf{y} - \frac{1}{n}\mathbf{1}\mathbf{1}^\intercal\mathbf{y}\|_2}.$$

Its placement within a transformer can vary across specific architectures [31], the two conventional choices being *post*normalization [29, 12] and *pre*normalization [5, 28, 8]. Yet neither places layer normalization right after a matrix product.

Another possible placement is between a (feedforward or attention) sublayer and a skip connection [24, 27], then layer normalization directly follows a matrix product and we can apply Algorithm 1.

Note that the shifting step in layer normalization is itself a matrix-vector product, and we can "attach" this matrix to the preceding matrix multiplication. The remaining function

$$f : \mathbb{R}^n \to \mathbb{R}^n, \quad f(\mathbf{y}) = \sqrt{n}\frac{\mathbf{y}}{\|\mathbf{y}\|_2} \tag{7}$$

is *root mean square* (RMS) layer normalization [32] and is often used on its own [28, 27]. We analyze its componentwise LAMP problem (5), which aims to control the relative error incurred in each individual entry of $f(\mathbf{y})$.

Proposition 3.1. *Let $\mathbf{q} \in \{0,1\}^n$ be such that $\mathbf{q} \neq \mathbf{1}$, and denote by Ω its support. For RMS layer normalization* (7)*,*

$$\kappa_c(f, \mathbf{y}; \mathbf{q}) = \begin{cases} 2\left(1 - \frac{\min_{j\notin\Omega} y_j^2}{\|\mathbf{y}\|_2^2}\right) - \frac{\sum_{i\in\Omega} y_i^2}{\|\mathbf{y}\|_2^2}, & |\Omega| \leq n-2, \\ \max\left\{\frac{y_j^2}{\|\mathbf{y}\|_2^2}, 1 - \frac{y_j^2}{\|\mathbf{y}\|_2^2}\right\}, & \{1,\dots,n\}\setminus\Omega = \{j\}. \end{cases}$$

Proof. See Appendix A.1. □

Even though Proposition 3.1 provides exact values of κ_c for each $\mathbf{q}$, finding the sparsest $\mathbf{q}$ to satisfy (3) is expensive for large n. The following proposition shows that there exist simple solutions of (5).

Proposition 3.2. *Suppose that the entries of $\mathbf{y}$ are arranged as $y_1^2 \geq \cdots \geq y_n^2$. Let $\mathbf{q} \in \{0,1\}^n$ be a solution of the componentwise LAMP problem* (5) *for RMS layer normalization* (7)*. If $\|\mathbf{q}\|_0 \leq n-3$ then a vector $\mathbf{q}' \in \{0,1\}^n$ given by*

$$\mathbf{q}' = [1 \quad \cdots \quad 1 \quad 0 \quad \cdots \quad 0]^\intercal, \quad \|\mathbf{q}'\|_0 = \|\mathbf{q}\|_0 + q_n,$$

also satisfies $\kappa_c(f, \mathbf{y}; \mathbf{q}') \leq \tau$.

Proof. See Appendix A.2. □

Proposition 3.2 guarantees that an almost-the-sparsest solution of the componentwise LAMP problem (5) has a closed-form expression and can be computed greedily: we need to sort the entries of $\mathbf{y}$ in descending order according to their squares, pick the smallest s such that

$$\sum_{i=1}^{s} y_i^2 + 2y_n^2 \geq (2-\tau)\|\mathbf{y}\|_2^2,$$

and form $\mathbf{q}$ based on the initial positions of the indices. For a spread-out vector with $y_1^2 = \cdots = y_{n-1}^2 = 1$ and $y_n = 0$, we get $s = \lceil (2-\tau)(n-1) \rceil$. On the contrary, $s = 1$ when $y_1^2 = 1$ and $y_2 = \cdots = y_n = 0$. Thus, vectors with massive outliers require a small number of recomputations.

[6] We focus on layer normalization without scale and bias, as they are an affine function applied after the nonlinearity.

3.3 Attention and softmax

The self-attention mechanism is a distinctive feature of transformers [29]. In the simplest case, attention is computed as $\mathbf{V} \cdot \mathrm{softmax}(\mathbf{K}^\intercal \mathbf{Q})$. This is a composition of three functions: the product of the key and query matrices, the softmax function (6) applied columnwise, and the product of the value matrix and the output of softmax.

When the value-softmax product is followed by layer normalization, its LAMP evaluation can be performed as in Subsection 3.2. Here, we study the LAMP problem (5) for

$$f : \mathbb{R}^n \to \mathbb{R}^n, \quad f(\mathbf{y}) = \mathrm{softmax}(\mathbf{y}).$$

There is a remarkable distinction between the componentwise LAMP problems for RMS layer normalization and softmax. Whereas the former admits a simple solution at the cost of one additional index, Appendix B shows that enlarging an optimal-sized greedy index set by any fixed or percentage-based margin is insufficient to ensure the componentwise LAMP objective (3) for softmax.

Because highly negative KQ inner products create pathological failure cases for the componentwise LAMP, we pivot to the ℓ_1-based objective (4), which naturally bounds the total probability mass shift. Recall the notation $\mathbf{z} = f(\mathbf{y})$.

Proposition 3.3. *Let $\mathbf{q} \in \{0,1\}^n$ be such that $\mathbf{q} \neq \mathbf{1}$, and denote by Ω its support. For softmax* (6)*,*

$$\kappa_1(f, \mathbf{y}; \mathbf{q}) = 2 \max_{j \notin \Omega} z_j(1 - z_j)|y_j|.$$

Proof. See Appendix A.3. □

The optimal solution of the LAMP objective is simple: the jth entry of $\mathbf{q}$ is nonzero if and only if

$$2z_j(1 - z_j)|y_j| > \tau. \tag{8}$$

This condition postulates that the tiniest probabilities are very stable (unlike in the case of componentwise LAMP). For an extremely concentrated distribution where $\mathbf{z}$ is close to a standard basis vector, no recomputations are needed. When an attention head is "confused" and produces multiple equally probable outcomes, these can necessitate more accurately computed KQ inner products.

As noted in Subsection 1.2, computing the optimal solution (8) requires a fully materialized vector of softmax probabilities. In Subsection 4.4, we bring the "strict" LAMP closer to the one-pass softmax.

4 Numerical experiments

The code used for the experiments and the raw numerical results (including results not reported in the paper) are provided in the Supplementary Material and in a public repository.[7] The computations were carried out on a cloud-based GPU equipped with 80 GB of VRAM and required approximately 60 GPU-hours.

4.1 Custom floating-point format

To focus on precision and put aside the issue of overflows, we simulate low-precision matrix products using a custom *partial single* FP format. For every $\mu \in \{1, \dots, 23\}$, we define a format $\mathrm{PS}(\mu)$ with μ mantissa bits, 8 exponent bits, and one sign bit. This format is equivalent to FP32 when $\mu = 23$, to TF32 when $\mu = 10$, and to BF16 when $\mu = 7$. In code, we implement $\mathrm{PS}(\mu)$ numbers via FP32 numbers rounded to μ mantissa bits according to the round-to-nearest-ties-to-even mode.

To multiply matrices with input-output formats $\mathrm{PS}(\mu_A) \times \mathrm{PS}(\mu_B) \to \mathrm{PS}(\mu_C)$, we accumulate inner products as $\mathrm{round}(c + a \cdot b)$, where the scalar multiplication and addition are in FP32.

[7] https://github.com/sbudzinskiy/LAMP-LLM

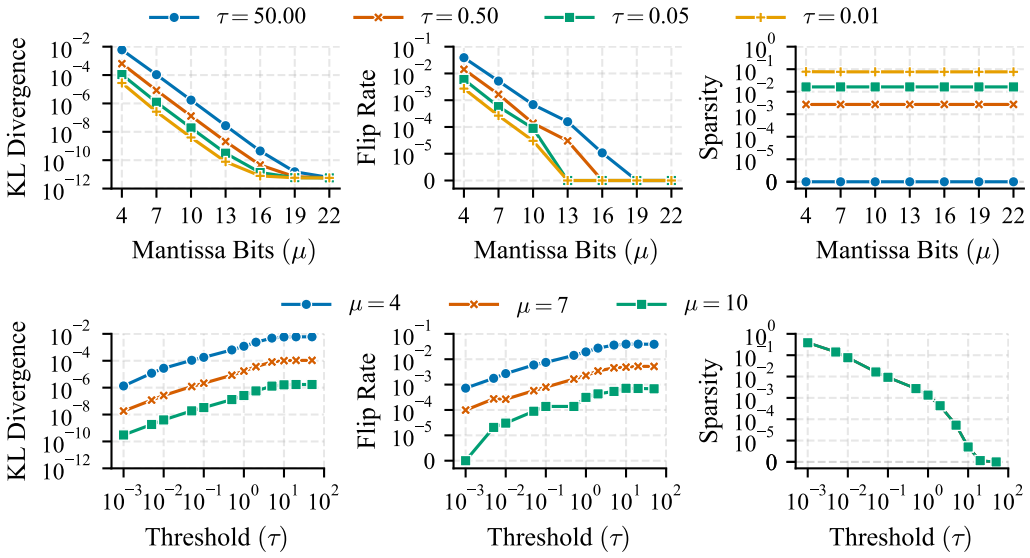

Figure 2: Performance of mixed-precision GPT-2 XL inference on the OpenWebText dataset with LAMP evaluation of KQ inner products.

4.2 Experimental setting

We validate the LAMP evaluation of attention according to (8) in experiments with the GPT-2 XL model and the OpenWebText[8] dataset. Find more experiments in Appendix C.

To assess the impact of LAMP evaluation on inference, we compute the mean Kullback–Leibler (KL) divergence between the probability distributions output by a reference model and a test model over 100 sequences of 1024 tokens each. We also look at the flip rate, i.e., how often the most probable predictions of the reference and test models differ.

Our reference model uses FP32 inference uniformly for all FP operations. The test models perform the KQ products in $\mathrm{PS}(\mu)$ and recompute those selected by the LAMP solution (8) in FP32. We keep track of how many inner products are recomputed; to get the recomputation rate, we divide by the number of KQ inner products in the "causal mask."

4.3 Numerical results: Proof of concept

The plots in Figure 2 demonstrate that our method works as intended: as the threshold τ decreases, it improves the accuracy of inference and increases the number of recomputations required. Our method achieves consistent $12\times$, $83\times$, and $385\times$ reductions in KL divergence at recomputation rates of only 0.3%, 1.6%, and 7.6%, respectively. Similar improvements hold for the flip rate, and we observe an exponential decay of the two metrics. An interesting observation is that the recomputation rate hardly depends on μ, which serves as circumstantial evidence for the stability of the Jacobian to small input variations (Subsection 2.3). We repeat that the *choice* of the components to be recomputed has a crucial impact on accuracy gains (see Figure 1 and Appendix C.4).

These results are a qualitative proof of concept for the LAMP evaluation of softmax in attention, as it reaches high inference accuracies relative to the ground truth with the majority of KQ inner products performed in low precision. The additional experiments in Appendix C strengthen this conclusion.

4.4 Relaxed relative-threshold LAMP for softmax

Consider a relaxation of the optimal LAMP solution (8) for softmax, obtained by omitting $1 - z_j$. One negligible effect incurred by this relaxation is that strictly dominant tokens with $z_j \approx 1$ can now

[8] https://huggingface.co/datasets/Skylion007/openwebtext

be considered numerically sensitive, whereas they are perfectly stable in the "strict" formulation (8). A more significant effect emerges once we switch to a relative threshold $0 \leq \tau < 1$,

$$|y_j|e^{y_j} > \tau \max_{1 \leq i \leq n} |y_i|e^{y_i}. \tag{9}$$

The defining advantage of the relaxed relative-threshold LAMP (9) is its complete independence from the normalization constant $\sum_i e^{y_i}$. This decoupling is a critical achievement for our future work on applying the LAMP framework to the one-pass design of FlashAttention.

In Figure 3, we compare the Pareto boundaries for the optimal and relaxed LAMP solutions, i.e., plotting the accuracy metrics (KL divergence and flip rate) against the efficiency metric (recomputation rate). As the theoretically optimal solution, the "strict" LAMP (8) achieves a perfect balance between accuracy and efficiency, and therefore its Pareto boundaries are lower and serve as a benchmark. Meanwhile, the Pareto boundaries of the relaxed LAMP (9) exhibit only a marginal upward shift, which is an illustration of its almost-optimality. These results validate the stability of our relaxation, paving the way for the efficient and effective integration of the LAMP framework into FlashAttention.

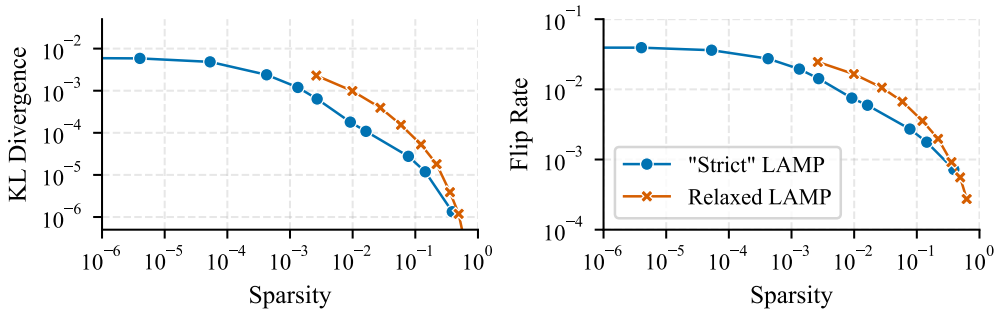

Figure 3: Comparison of Pareto boundaries of "strict" LAMP (8) and relaxed LAMP (9) applied to GPT-2 XL with $\mu = 4$ mantissa bits for KQ accumulation and validated on the OpenWebText dataset.

Author Contributions

SB conceived the approach, formulated the research problem, and carried out the formal analysis, experimentation, and implementation. SB wrote the original draft of the manuscript. MG and TY contributed to the analysis and implementation and assisted with proofreading. YHT contributed to project administration and to reviewing and editing the manuscript. YL, WF, and FW contributed to project administration. PP supervised the project as laboratory head and reviewed and edited the manuscript.

Acknowledgments

This work was carried out in the framework of a research project funded by Huawei Technologies Ltd. We are grateful to El-Mehdi El Arar, Silviu-Ioan Filip, Theo Mary, and Elisa Riccietti for fruitful discussions.

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv*, 2016. doi:10.48550/arXiv.1607.06450.

[2] I. Beltagy, M. E. Peters, and A. Cohan. Longformer: The long-document transformer. *arXiv*, 2020. doi:10.48550/arXiv.2004.05150.

[3] P. Blanchard, D. J. Higham, and N. J. Higham. Accurately computing the log-sum-exp and softmax functions. *IMA J Numer Anal*, 41(4):2311–2330, 2021. doi:10.1093/imanum/draa038.

[4] P. Blanchard, N. J. Higham, F. Lopez, T. Mary, and S. Pranesh. Mixed precision block fused multiply-add: Error analysis and application to gpu tensor cores. *SIAM J Sci Comput*, 42(3):C124–C141, 2020. doi:10.1137/19M1289546.

[5] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. In *NeurIPS*, volume 33, pages 1877–1901, 2020. doi:10.48550/arXiv.2005.14165.

[6] S. Budzinskiy, W. Fang, L. Zeng, and P. Petersen. Numerical stability analysis of large language models. *arXiv*, 2025. doi:10.48550/arXiv.2503.10251.

[7] S. Budzinskiy, M. Gloser, T. Yilmaz, Y. H. Tham, Y. Lin, W. Fang, F. Wu, and P. Petersen. LampAttention: Look-ahead mixed-precision FlashAttention for dedicated accelerators. *arXiv*, 2026. doi:10.48550/arXiv.2609.39361.

[8] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *J Mach Learn Res*, 24(240):1–113, 2023. doi:10.48550/arXiv.2204.02311.

[9] M. P. Connolly, N. J. Higham, and T. Mary. Stochastic rounding and its probabilistic backward error analysis. *SIAM J Sci Comput*, 43(1):A566–A585, 2021. doi:10.1137/20M1334796.

[10] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, volume 35, pages 16344–16359, 2022. doi:10.48550/arXiv.2205.14135.

[11] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In *NeurIPS*, volume 35, pages 30318–30332, 2022. doi:10.48550/arXiv.2208.07339.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, volume 1, pages 4171–4186, 2019. doi:10.48550/arXiv.1810.04805.

[13] A. Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. doi:10.48550/arXiv.2010.11929.

[14] E.-M. El Arar, S.-I. Filip, T. Mary, and E. Riccietti. Mixed precision accumulation for neural network inference guided by componentwise forward error analysis. *IMA J Numer Anal*, 2026. doi:10.1093/imanum/drag054/8785932.

[15] E.-M. El Arar, D. Sohier, P. de Oliveira Castro, and E. Petit. Bounds on nonlinear errors for variance computation with stochastic rounding. *SIAM J Sci Comput*, 46(5):B579–B599, 2024. doi:10.1137/23M1563001.

[16] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh. OPTQ: Accurate post-training quantization for generative pre-trained transformers. In *ICLR*, 2023. doi:10.48550/arXiv.2210.17323.

[17] A. Gil, J. Segura, and N. M. Temme. *Numerical Methods for Special Functions*. SIAM, 2007. doi:10.1137/1.9780898717822.

[18] I. Gohberg and I. Koltracht. Mixed, componentwise, and structured condition numbers. *SIAM J Matrix Anal Appl*, 14(3):688–704, 1993. doi:10.1137/0614049.

[19] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan. Deep learning with limited numerical precision. In *ICML*, volume 37, pages 1737–1746, 2015. doi:10.48550/arXiv.1502.02551.

[20] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2 edition, 2002. doi:10.1137/1.9780898718027.

[21] D. Kalamkar, D. Mudigere, N. Mellempudi, D. Das, K. Banerjee, S. Avancha, D. T. Vooturi, N. Jammalamadaka, J. Huang, H. Yuen, et al. A study of BFLOAT16 for deep learning training. *arXiv*, 2019. doi:10.48550/arXiv.1905.12322.

[22] N. Kitaev, Ł. Kaiser, and A. Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020. doi:10.48550/arXiv.2001.04451.

[23] C. Lauter and M. Mezzarobba. Semi-automatic floating-point implementation of special functions. In *ARITH 22*, pages 58–65, 2015. doi:10.1109/ARITH.2015.12.

[24] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *IEEE/CVF CVPR*, pages 12009–12019, 2022. doi:10.48550/arXiv.2111.09883.

[25] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, et al. Mixed precision training. In *ICLR*, 2018. doi:10.48550/arXiv.1710.03740.

[26] J.-M. Muller. *Elementary Functions: Algorithms and Implementation*. Springer, 3 edition, 2016. doi:10.1007/978-1-4899-7983-4.

[27] T. OLMo, P. Walsh, L. Soldaini, D. Groeneveld, K. Lo, S. Arora, A. Bhagia, Y. Gu, S. Huang, M. Jordan, et al. 2 OLMo 2 furious. *arXiv*, 2025. doi:10.48550/arXiv.2501.00656.

[28] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv*, 2023. doi:10.48550/arXiv.2302.13971.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, volume 30, pages 5998–6008, 2017. doi:10.48550/arXiv.1706.03762.

[30] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *ICML*, volume 202, pages 38087–38099, 2023. doi:10.48550/arXiv.2211.10438.

[31] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu. On layer normalization in the transformer architecture. In *ICML*, volume 119, pages 10524–10533, 2020. doi:10.48550/arXiv.2002.04745.

[32] B. Zhang and R. Sennrich. Root mean square layer normalization. In *NeurIPS*, volume 32, pages 12360–12371, 2019. doi:10.48550/arXiv.1910.07467.

A Proofs

A.1 Proof of Proposition 3.1

Direct calculation yields

$$\mathbf{J}_f(\mathbf{y}) = \mathbf{I} - \frac{\mathbf{y}\mathbf{y}^\intercal}{\|\mathbf{y}\|_2^2}, \quad \mathbf{M}(f,\mathbf{y}) = \mathbf{I} - \frac{\mathbf{1}\mathbf{y}^\intercal}{\|\mathbf{y}\|_2^2}\mathrm{diag}(\mathbf{y}).$$

Denote by s_l the lth absolute row-sum of $\mathbf{M}(f,\mathbf{y})\big(\mathbf{I} - \mathrm{diag}(\mathbf{q})\big)$:

$$s_l = \begin{cases} \frac{1}{\|\mathbf{y}\|_2^2}\sum_{j\notin\Omega} y_j^2, & l \in \Omega, \\ \frac{1}{\|\mathbf{y}\|_2^2}\sum_{j\notin\Omega\cup\{l\}} y_j^2 + 1 - \frac{y_l^2}{\|\mathbf{y}\|_2^2}, & l \notin \Omega. \end{cases}$$

Rewriting $\sum_{j\notin\Omega} y_j^2 = \|\mathbf{y}\|_2^2 - \sum_{i\in\Omega} y_i^2$ and taking the maximum of s_l over l, we get

$$\kappa_c(f,\mathbf{y};\mathbf{q}) = \max\left\{2\left(1 - \frac{\min_{j\notin\Omega} y_j^2}{\|\mathbf{y}\|_2^2}\right), 1\right\} - \frac{\sum_{i\in\Omega} y_i^2}{\|\mathbf{y}\|_2^2}.$$

Note that $1 > 2\Big(1 - \frac{\min_{j\notin\Omega} y_j^2}{\|\mathbf{y}\|_2^2}\Big)$ if and only if $\min_{j\notin\Omega} y_j^2 > \frac{1}{2}\|\mathbf{y}\|_2^2$. When $|\Omega| \le n-2$, this would imply that

$$\sum_{j\notin\Omega} y_j^2 > \frac{1}{2}(n - |\Omega|)\|\mathbf{y}\|_2^2 \ge \|\mathbf{y}\|_2^2,$$

which is impossible. This contradiction proves the first formula. The second formula follows from the general expression.

A.2 Proof of Proposition 3.2

Denote by Ω and Ω' the supports of $\mathbf{q}$ and $\mathbf{q}'$, respectively. By Proposition 3.1, as $|\Omega'| \le n-2$, we need to show that

$$2\left(1 - \frac{y_n^2}{\|\mathbf{y}\|_2^2}\right) - \frac{\sum_{i\in\Omega'} y_i^2}{\|\mathbf{y}\|_2^2} \le \tau.$$

Since $|\Omega| < n-2$ and $\mathbf{q}$ satisfies the componentwise LAMP constraint, it suffices to show that

$$\sum_{i\in\Omega'} y_i^2 + 2y_n^2 \ge \sum_{i\in\Omega} y_i^2 + 2\min_{j\notin\Omega} y_j^2.$$

Note that Ω' is constructed in such a way that for every $\tilde{\Omega} \subset \{1,\dots,n\}$ with $|\tilde{\Omega}| = |\Omega'|$, it holds that $\sum_{i\in\Omega'} y_i^2 \ge \sum_{i\in\tilde{\Omega}} y_i^2$. When $n \notin \Omega$, we have $\min_{j\notin\Omega} y_j^2 = y_n^2$, and the desired inequality follows trivially since $|\Omega| = |\Omega'|$. When $n \in \Omega$, we have

$$\sum_{i\in\Omega} y_i^2 + 2\min_{j\notin\Omega} y_j^2 \le y_n^2 + \sum_{i\in\Omega\setminus\{n\}} y_i^2 + \min_{\substack{j,l\notin\Omega \\ j\ne l}}(y_j^2 + y_l^2) \le y_n^2 + \sum_{i\in\Omega'} y_i^2 \le 2y_n^2 + \sum_{i\in\Omega'} y_i^2,$$

where in the second inequality we apply the majorizing property of Ω' to the set $\tilde{\Omega} = \Omega\setminus\{n\}\cup\{j_*, l_*\}$, with minimizing indices j_* and l_*, and note that $|\tilde{\Omega}| = |\Omega'|$.

A.3 Proof of Proposition 3.3

Direct calculation yields

$$\mathbf{J}_f(\mathbf{y}) = \mathrm{diag}(\mathbf{z}) - \mathbf{z}\mathbf{z}^\intercal, \quad \mathbf{K}(f,\mathbf{y}) = \big(\mathrm{diag}(\mathbf{z}) - \mathbf{z}\mathbf{z}^\intercal\big)\mathrm{diag}(\mathbf{y}).$$

Then

$$\begin{aligned}\kappa_1(f,\mathbf{y};\mathbf{q}) &= \|\mathbf{K}(f,\mathbf{y})(\mathbf{I} - \mathrm{diag}(\mathbf{q}))\|_{1,1} = \max_{j\notin\Omega}\sum_i |z_j y_j \delta_{i,j} - z_i z_j y_j| \\ &= \max_{j\notin\Omega}\Big[z_j(1-z_j)|y_j| + z_j|y_j|\sum_{i\ne j} z_i\Big] = \max_{j\notin\Omega}\Big[z_j(1-z_j)|y_j| + z_j|y_j|(1-z_j)\Big].\end{aligned}$$

B Componentwise LAMP for softmax: Counterexamples

Let us derive an explicit expression for $\kappa_c(f, \mathbf{y}; \mathbf{q})$. Let Ω be the support of $\mathbf{q}$, then

$$\begin{aligned}\kappa_c(f, \mathbf{y}; \mathbf{q}) &= \|\mathbf{M}(f, \mathbf{y})(\mathbf{I} - \operatorname{diag}(\mathbf{q}))\|_{\infty,\infty} \\ &= \max_{1\le i\le n} \sum_{j\notin\Omega} |\delta_{i,j} - z_j||y_j| = \sum_{j\notin\Omega} z_j|y_j| + \max_{i\notin\Omega}(1-2z_i)|y_i|.\end{aligned}$$

The formula naturally contains auxiliary vectors $\mathbf{u} \in \mathbb{R}^n$ with $u_j = z_j|y_j|$ and $\mathbf{v} \in \mathbb{R}^n$ with $v_i = (1-2z_i)|y_i|$. If either $\mathbf{u}$ or $\mathbf{v}$ were identically zero, the componentwise LAMP problem (5) would admit a simple exact solution based on thresholding or sorting, respectively. This is not the case in general, though, and we prove that the exact solution $\mathbf{q}$ of (5) cannot be approximated with a greedy surrogate based on sorting the entries of $\mathbf{u}$ or picking the largest entries of $\mathbf{v}$. Specifically, we design two families of examples that illustrate that for any $s \in \mathbb{N}$, taking a greedily constructed $\mathbf{q}'$ with support Ω' of cardinality $|\Omega'| = |\Omega| + s$ does not guarantee $\kappa_c(f, \mathbf{y}; \mathbf{q}') \le \tau$. Because the choice of s is independent of $|\Omega|$, this simultaneously shows that increasing the cardinality by any arbitrary percentage is not a viable strategy either.

In the first example, the optimal solution is to select the most negative entries of $\mathbf{y}$ to minimize the second term $\max_{i\notin\Omega} v_i$, whereas the greedy strategy based on the vector $\mathbf{u}$ ignores these entries because of their tiny probabilities.

Proposition B.1. *Let $n_0, s \in \mathbb{N}$ and set $n = 2n_0 + s$. There exists a vector $\mathbf{y} \in \mathbb{R}^n$ and a threshold $\tau > 0$ such that*

$$n_0 = \min\{\|\mathbf{q}\|_0 : \kappa_c(f, \mathbf{y}; \mathbf{q}) \le \tau\},$$

but the heuristic $\mathbf{q}' \in \{0,1\}^n$ formed greedily by selecting the largest $n_0 + s$ entries of either $\mathbf{u}$ or $\mathbf{z}$ satisfies $\kappa_c(f, \mathbf{y}; \mathbf{q}') > \tau$.

Proof. Fix $\alpha \ge 3$ and define a vector $\mathbf{y} \in \mathbb{R}^n$ with entries

$$y_i = \begin{cases} -\alpha, & 1 \le i \le n_0, \\ -1, & n_0 < i \le n. \end{cases}$$

Denote $Z = n_0 e^{-\alpha} + (n_0 + s)e^{-1}$. Then the vectors $\mathbf{z}$, $\mathbf{u}$, and $\mathbf{v}$ evaluate to

$$z_i = \begin{cases} \frac{1}{Ze^\alpha}, & 1 \le i \le n_0, \\ \frac{1}{Ze}, & n_0 < i \le n, \end{cases} \quad u_i = \begin{cases} \frac{\alpha}{Ze^\alpha}, & 1 \le i \le n_0, \\ \frac{1}{Ze}, & n_0 < i \le n, \end{cases} \quad v_i = \begin{cases} \alpha(1 - \frac{2}{Ze^\alpha}), & 1 \le i \le n_0, \\ 1 - \frac{2}{Ze}, & n_0 < i \le n. \end{cases}$$

Note that the function xe^{-x} reaches its maximum at $x = 1$. Define $\mathbf{q} \in \{0,1\}^n$ with support $\Omega = \{1, \dots, n_0\}$ and set

$$\tau = \kappa_c(f, \mathbf{y}; \mathbf{q}) = (n_0 + s)\frac{1}{Ze} + 1 - \frac{2}{Ze} < 1 + \frac{n_0 + s}{Ze} < 1 + \frac{n_0 + s}{(n_0 + s)e^{-1}e} = 2.$$

For any $\tilde{\mathbf{q}} \in \{0,1\}^n$ with $\|\tilde{\mathbf{q}}\|_0 < n_0$, the complement of $\tilde{\Omega}$ contains at least one $i \in \{1, \dots, n_0\}$, and thus

$$\kappa_c(f, \mathbf{y}; \tilde{\mathbf{q}}) > \alpha\Big(1 - \frac{2}{Ze^\alpha}\Big) \ge \alpha\Big(1 - \frac{2}{1 + 2e^{\alpha-1}}\Big) > 2 > \tau = \kappa_c(f, \mathbf{y}; \mathbf{q}),$$

proving that $\mathbf{q}$ is optimal for the chosen τ. By construction, the heuristic vector $\mathbf{q}'$ has support $\Omega' = \{n_0 + 1, \dots, n\}$, so

$$\kappa_c(f, \mathbf{y}; \mathbf{q}') = n_0 \frac{\alpha}{Ze^\alpha} + \alpha\Big(1 - \frac{2}{Ze^\alpha}\Big) > \alpha\Big(1 - \frac{2}{Ze^\alpha}\Big) > 2 > \tau. \qquad \square$$

In the second example, the optimal solution is to take the largest entries of $\mathbf{y}$ so as to minimize the first term $\sum_{j\notin\Omega} u_j$, but the greedy strategy based on the vector $\mathbf{v}$ ignores these entries because their probabilities force $1 - 2z_i$ to be small.

Proposition B.2. *Let $n_0, s \in \mathbb{N}$ and set $n = 2n_0 + s$. There exists a vector $\mathbf{y} \in \mathbb{R}^n$ and a threshold $\tau > 0$ such that*

$$n_0 = \min\{\|\mathbf{q}\|_0 : \kappa_c(f, \mathbf{y}; \mathbf{q}) \le \tau\},$$

but the heuristic $\mathbf{q}' \in \{0,1\}^n$ formed greedily by selecting the largest $n_0 + s$ entries of $\mathbf{v}$ satisfies $\kappa_c(f, \mathbf{y}; \mathbf{q}') > \tau$.

Proof. Consider a constant

$$\alpha = \frac{(n_0+s)(5n_0-4)}{4s}\log\Big(\frac{n_0+s}{n_0}\Big) > 0$$

and define a vector $\mathbf{y} \in \mathbb{R}^n$ with entries

$$y_i = \begin{cases} \alpha + \log\big(\frac{n_0+s}{n_0}\big), & 1 \le i \le n_0, \\ \alpha, & n_0 < i \le n. \end{cases}$$

Then

$$z_i = \begin{cases} \frac{1}{2n_0}, & 1 \le i \le n_0, \\ \frac{1}{2(n_0+s)}, & n_0 < i \le n, \end{cases} \qquad u_i = \begin{cases} \frac{1}{2n_0}\Big(\alpha + \log\big(\frac{n_0+s}{n_0}\big)\Big), & 1 \le i \le n_0, \\ \frac{1}{2(n_0+s)}\alpha, & n_0 < i \le n, \end{cases}$$

$$v_i = \begin{cases} \big(1-\frac{1}{n_0}\big)\Big(\alpha + \log\big(\frac{n_0+s}{n_0}\big)\Big), & 1 \le i \le n_0, \\ \big(1-\frac{1}{n_0+s}\big)\alpha, & n_0 < i \le n. \end{cases}$$

Direct calculation yields $v_i > v_j$ for $n_0 < i \le n$ and $1 \le j \le n_0$. Let $\mathbf{q} \in \{0,1\}^n$ have support $\Omega = \{1, \dots, n_0\}$ and set

$$\tau = \kappa_c(f, \mathbf{y}; \mathbf{q}) = \frac{\alpha}{2} + \Big(1 - \frac{1}{n_0+s}\Big)\alpha.$$

For any $\tilde{\mathbf{q}} \in \{0,1\}^n$ with $\|\tilde{\mathbf{q}}\|_0 < n_0$, the complement of $\tilde{\Omega}$ contains at least one index from each of the two groups, so

$$\kappa_c(f, \mathbf{y}; \tilde{\mathbf{q}}) = \sum_{j \notin \tilde{\Omega}} u_j + \Big(1 - \frac{1}{n_0+s}\Big)\alpha \ge \frac{1}{2n_0}\Big(\alpha + \log\Big(\frac{n_0+s}{n_0}\Big)\Big) + \frac{n_0+s}{2(n_0+s)}\alpha + \Big(1 - \frac{1}{n_0+s}\Big)\alpha > \tau,$$

proving that $\mathbf{q}$ is optimal for the chosen τ. By construction, the heuristic vector $\mathbf{q'}$ has support $\Omega' = \{n_0+1, \dots, n\}$, so

$$\begin{aligned} \kappa_c(f, \mathbf{y}; \mathbf{q}') &= \frac{1}{2}\Big(\alpha + \log\Big(\frac{n_0+s}{n_0}\Big)\Big) + \Big(1 - \frac{1}{n_0}\Big)\Big(\alpha + \log\Big(\frac{n_0+s}{n_0}\Big)\Big) - \tau + \tau \\ &= \Big(\frac{3}{2} - \frac{1}{n_0}\Big)\log\Big(\frac{n_0+s}{n_0}\Big) - \frac{s\alpha}{n_0(n_0+s)} + \tau \\ &= \Big(\frac{3}{2} - \frac{1}{n_0} - \frac{5n_0-4}{4n_0}\Big)\log\Big(\frac{n_0+s}{n_0}\Big) + \tau = \frac{1}{4}\log\Big(\frac{n_0+s}{n_0}\Big) + \tau > \tau. \end{aligned}$$

□

C Additional numerical experiments

C.1 Comparison of performance: Different datasets

In the main text, we presented experiments with the GPT-2 XL model and the OpenWebText[9] dataset. Here, we add two more datasets for comparison: CodeParrot[10] and ArXiv.[11] The comparison of Pareto boundaries in Figure 4 shows that the behavior of the proposed LAMP method (8) for softmax remains almost the same across different datasets, suggesting that our approach is "input-agnostic."

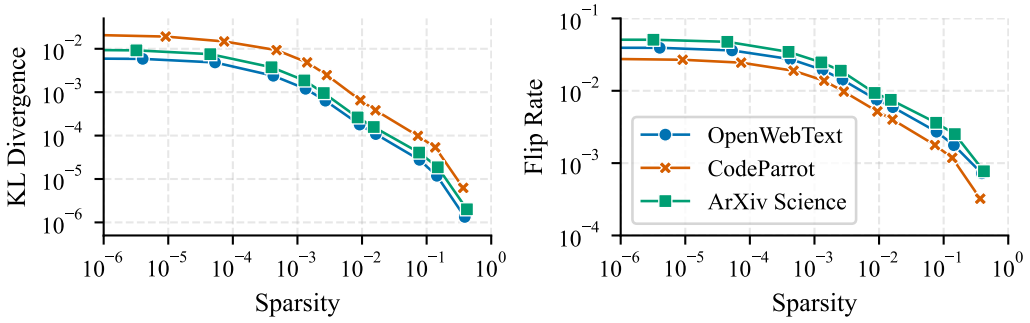

Figure 4: Comparison of Pareto boundaries of LAMP (8) applied to GPT-2 XL with $\mu = 4$ mantissa bits for KQ accumulation and validated on different datasets.

C.2 Comparison of performance: Different models

Next, we validate the proposed method on a different transformer model—the GPT-2 small model. The numerical results in Figure 5 show that LAMP evaluation of attention performs better (has a lower Pareto boundary) for the GPT-2 XL model than for the GPT-2 small model, supporting our vision that the performance of LAMP is likely to improve further for larger transformer architectures.

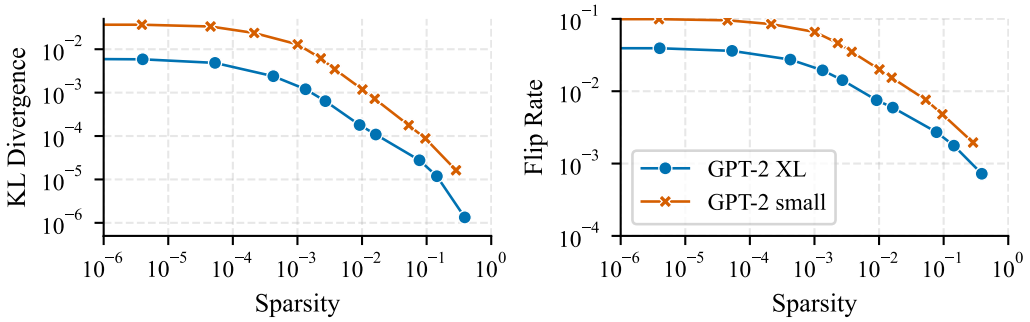

Figure 5: Comparison of Pareto boundaries of LAMP (8) applied to GPT-2 XL and GPT-2 small with $\mu = 4$ mantissa bits for KQ accumulation and validated on the OpenWebText dataset.

C.3 Comparison of performance: Permuted sequences of tokens

To isolate the effect of word order, we construct three modified datasets by permuting the tokens in each sequence at random. This eliminates all sequential dependencies while preserving the unigram distribution, i.e., the new sequences no longer correspond to a coherent text yet consist of the same tokens in a different order. Figure 6 shows that LAMP inference performs equally well on such "incoherent" data, supporting our claim that the proposed method is "input-agnostic." An interesting

[9] https://huggingface.co/datasets/Skylion007/openwebtext
[10] https://huggingface.co/datasets/codeparrot/codeparrot-clean
[11] https://huggingface.co/datasets/ccdv/arxiv-summarization

observation is that while the Pareto boundaries of the KL divergence essentially overlap, the Pareto boundary of the flip rate shifted upwards slightly for the permuted tokens.

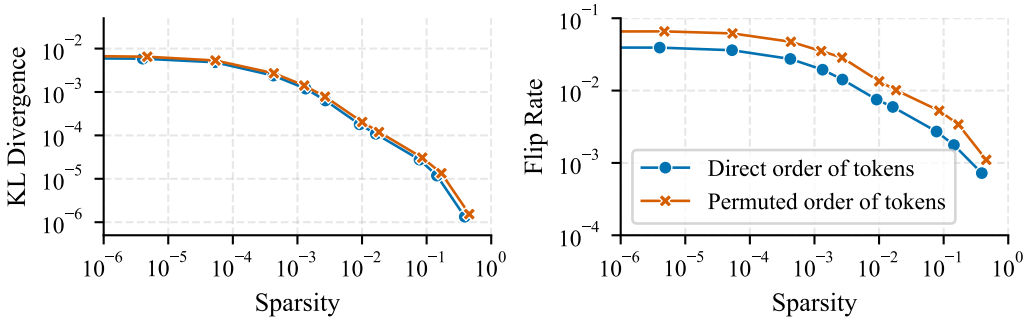

Figure 6: Comparison of Pareto boundaries of LAMP (8) applied to GPT-2 XL with $\mu = 4$ mantissa bits for KQ accumulation and validated on the OpenWebText dataset with direct and permuted tokens.

C.4 Comparison of performance: Random recomputation

The proposed LAMP inference adaptively chooses those KQ inner products that need to be recomputed more accurately. To show that it is not just the *number* of recomputations that is significant, but the actual *choice* of them, we perform experiments with *random* recomputations: the number of inner products to redo is chosen by LAMP, but the specific inner products are selected at random. An experiment of this sort was presented in Figure 1. The results in Figure 7 leave no doubt that the adaptive choice of the recomputations is the crux of our method, since random recomputations do not lead to any improvements.

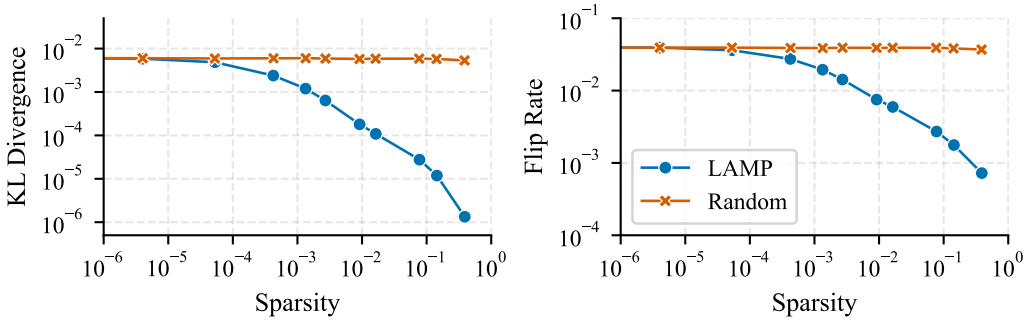

Figure 7: Comparison of Pareto boundaries of LAMP (8) and its analogue with randomly selected recomputations applied to GPT-2 XL with $\mu = 4$ mantissa bits for KQ accumulation and validated on the OpenWebText dataset.

C.5 Perplexity and length-normalized thresholding

Previously, we focused on the KL divergence and flip rate as the accuracy metrics to validate LAMP; here, we consider the perplexity attained on GSM8k Math,[12] WikiText-2,[13] and CodeParrot.[14] For each dataset, we compute the perplexity over 100 consecutive batches of 1024 tokens each. In Table 1, we compare the relaxed relative-threshold LAMP (9) with its *length-normalized* (LN) modification. The idea of the latter is following: instead of using the same relative threshold τ for each row of the "causal" KQ matrix, irrespective of its length n, we try to increase the threshold for shorter rows to

[12] https://huggingface.co/datasets/openai/gsm8k
[13] https://huggingface.co/datasets/Salesforce/wikitext
[14] https://huggingface.co/datasets/codeparrot/codeparrot-clean

Table 1: Comparison of perplexities of full-precision inference, low-precision inference, relaxed LAMP (9), and its length-normalized modification applied to GPT-2 XL with $\mu = 4$ mantissa bits for KQ accumulation and validated on several datasets.

Dataset	Method	Specification	Perplexity	Sparsity
GSM8k Math	Full precision	N/A	13.088	100%
	Low precision	N/A	13.161	0%
	LAMP	Relaxed ($\tau = 0.03$)	13.091	5.8%
		Relaxed LN ($\tau = 0.03$)	13.094	4.5%
		Relaxed ($\tau = 0.09$)	13.093	2.8%
		Relaxed LN ($\tau = 0.09$)	13.096	2.0%
WikiText-2	Full precision	N/A	17.411	100%
	Low precision	N/A	17.524	0%
	LAMP	Relaxed ($\tau = 0.03$)	17.454	5.7%
		Relaxed LN ($\tau = 0.03$)	17.456	4.3%
		Relaxed ($\tau = 0.09$)	17.458	2.7%
		Relaxed LN ($\tau = 0.09$)	17.459	1.9%
CodeParrot	Full precision	N/A	3.174	100%
	Low precision	N/A	3.221	0%
	LAMP	Relaxed ($\tau = 0.03$)	3.180	5.5%
		Relaxed LN ($\tau = 0.03$)	3.180	4.2%
		Relaxed ($\tau = 0.09$)	3.182	2.6%
		Relaxed LN ($\tau = 0.09$)	3.183	1.8%

preserve the relative significance of the selected indices. To test this idea, we scale the threshold as $\tau\sqrt{1024/n}$, where 1024 is the maximum context length the GPT-2 family of models was trained on.

First of all, the results in Table 1 show that the LAMP framework succeeds in decreasing the perplexity of purely low-precision inference using a small number of recomputed KQ inner products. Moreover, the results obtained for CodeParrot demonstrate that the LN modification of relaxed LAMP can achieve the same perplexity as the original relaxed LAMP (9) with fewer recomputations. On the contrary, the results for GSM8k Math show that the LN variant can also perform worse than the original relaxed LAMP. This suggests that length-normalized thresholding can be beneficial for LAMP, but empirical fine-tuning of the scaling law is necessary.

C.6 Zero-shot multiple-choice task

We additionally assess LAMP on a downstream task ARC-Easy[15]. The baseline accuracy achieved with full-precision GPT-2 XL inference is 0.561, and it drops to 0.558 with $\mu = 4$ mantissa bits for KQ accumulation. The use of LAMP with $\tau = 0.09$ increases the accuracy to 0.560 using 32% of recomputations. These results suggest that GPT-2 models are too small to be confident on zero-shot downstream tasks, which is likely why the accuracy drop is low and the recomputation rate is higher than in other experiments.

[15] https://huggingface.co/datasets/allenai/ai2_arc

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The introduction discusses the limitations of the proposed method and the paper in details. See Subsection 1.2.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Every theoretical result is accompanied with a full set of assumptions and a complete, rigorous proof. See Section 3, Appendix A, and Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The supplementary material provides the full code, the computational environment, and the instructions to run the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The supplementary material provides the full code, the computational environment, and the instructions to run the experiments.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setting is described in Section 4, and the full details are provided in the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The primary experiments evaluate the deterministic inference of fixed-weight models on a static, sequential subset (100 batches) of benchmark datasets; hence, there is no cross-run variance. For the experiments with random token permutation and random inner-product recomputation, we enforce a fixed random seed to guarantee exact reproducibility, though we omit cross-seed variance reporting due to computational constraints.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The type of hardware, its memory specifications, and the total compute time are provided in Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper does not raise any ethical concerns and therefore conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: Our work has no immediate negative societal impact. Its fundamental goal is to improve the efficiency of LLMs, which has the potential to reduce the energy consumption required to deploy such models.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: The specific models and datasets used in the paper are open-source. A citation is provided for the models and URLs are provided for the datasets, however the license and terms of use are not explicitly mentioned.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: The supplementary material contains an anonymized zip file with the code and computational environment used for experiments. While a readme is provided to run the code, the code is not well documented, remaining a work in progress.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research did not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.