Total Variation Distance Estimation in Autoregressive Models

Eric Price[*]
University of Texas at Austin
ecprice@cs.utexas.edu

Kevin Tian[*]
University of Texas at Austin
kjtian@cs.utexas.edu

Zhiyang Xun[*]
University of Texas at Austin
zxun@cs.utexas.edu

Yusong Zhu[*]
University of Texas at Austin
zhuys@utexas.edu

Abstract

Modern LLM deployments use a number of implementation choices and inference optimizations (e.g., batching, custom kernels, quantization) on top of fixed weights, so two engines serving "the same model" can produce meaningfully different distributions. We study the problem of estimating the total variation (TV) distance between two length-n autoregressive distributions to additive error ε, under three access models:

- Under *sample* access, we use $\widetilde{O}(\frac{n^2K}{\varepsilon^2})$ queries, where K is the maximum support of the next-token distribution.
- Under *logit* access, we use $O(\frac{n}{\varepsilon^2})$ queries, and this is tight.
- Under *noisy logit* access, we can smoothly interpolate between the above two: if probability values are given to σ relative error, we use $\widetilde{O}(\frac{n+n^2\sigma^2}{\varepsilon^2})$ queries.

We complement our theoretical results with an empirical evaluation of our algorithms, for example measuring the distance between `sglang` and `vllm` serving identical weights. Our experiments highlight the robustness and practicality of estimating the total variation distance, which remains estimable where the KL divergence is infinite. Our code is available at `https://github.com/XunZhiyang/llm-tv-estimation`.

1 Introduction

While the behavior of a large language model (LLM) is often idealized as being determined by its model weights, in practice a wide variety of choices — GPUs, batching, kernels, and low-level numerical effects — can noticeably change its output behavior [HT25]. Other inference-time optimizations, e.g., weight quantization [FAHA22, LTT$^+$24], KV quantization [LYJ$^+$24, HKM$^+$24], and lossy speculative decoding methods [ZLR$^+$24, LWZZ24], deliberately introduce approximation errors in exchange for greater efficiency. Other times, it has been documented that model providers simply serve with bugs [CTLI25]. More broadly, different providers of the same open-weight models can exhibit significant differences in latency, throughput, cost, and performance [Art26, Cla25].

These considerations motivate the problem of eliciting differences between two LLMs. We approach this issue from the perspective of *distribution testing*: under an appropriate notion of API access to two distributions μ, π over Σ^n, representing LLMs generating length-n sequences over a token alphabet Σ, can we estimate the distance between μ, π? From a practical perspective, such an

[*] Authors are ordered alphabetically.

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

estimator could then be used by a user for comparing model providers (how close is the cheap provider to the expensive one?), and conversely, a model provider can use the estimator to test how well a candidate more-efficient inference stack matches their old stack.

In this paper, to evaluate such distributional changes, we focus on measuring the *total variation (TV) distance* between two LLMs. The TV distance, whose definition is recalled in (2), is a convenient and interpretable statistical distance. For example, bounding the TV distance also directly implies bounds on the change to any distinguishing test, e.g., the loss on a fixed evaluation metric.

Prior work on evaluating distributional changes in LLMs [Fir24, DKKR24] often measure the KL divergence, whose definition is also recalled in (2). From a practical standpoint, this has several disadvantages. The KL divergence between distributions is unbounded when their supports are disjoint, so it is not possible to estimate in general to ε precision from finite samples, as this would require support recovery. Smoothed approximations to the KL can be used by e.g., introducing pseudocounts, but these strategies do not have a uniform approximation guarantee and require hyperparameter selection. In contrast, the TV distance is bounded in $[0, 1]$, and therefore it inherits an interpretable scale. Moreover, the TV distance is robust to small implementation differences, such as restricting next-token distributions to their top-k elements for $k = 20$ vs. $k = 30$, or censoring swear words. Such differences alter the support and thus cause KL to diverge. Measuring KL has thus far remained the practitioner default largely because of the chain rule: the sequence-level KL divergence is the sum of the expected per-token KL divergences, so per-step logit comparisons give an unbiased estimator. TV has no such decomposition, which is the technical obstacle this paper addresses.

For further discussion on the impact of the choice of distance metric, see Appendix E.

1.1 Our results

We describe results under three models of accessing LLMs: sample access, logit access, and noisy distribution access. These are formalized in Definitions 1, 2, and 3, but we informally describe them here. Throughout, we fix two distributions π, μ over Σ^n whose TV distance we want to estimate, and we define $m := |\Sigma|$. We also let $K \in [m]$ denote the maximum support size of any next-token distribution $\pi(x_i \mid x_{<i})$ or $\mu(x_i \mid x_{<i})$, where $x_{<i} \in \Sigma^{i-1}$ is a specified prefix. LLM APIs typically support specifying a $k \in \mathbb{N}^+$ and sampling from the top-k tokens; due to nondeterminism in which tokens reach the threshold, the total support size K is somewhat larger than k; but we find it to be well below $2k$ in practice (Figure 5 in Appendix D.5).

Sample access. The simplest and most general access model we consider is *sample access*: we can query any prefix $x_{<i}$ and get a sample from the next-token distribution $\pi(x_i \mid x_{<i})$. This oracle is explicitly studied by [AFL24, MKP25], and has an intellectual history in the recent theoretical computer science literature on conditional property testing [CFGM13, ACK14, CRS15, BC18].

Theorem 1 (Informal, cf. Corollary 1)**.** *There is an algorithm that estimates* $\mathrm{TV}(\pi, \mu)$ *up to error* ε *with probability* $\geq \frac{9}{10}$*, using* $O(\frac{n^2 K}{\varepsilon^2} \cdot \log^2(\frac{1}{\varepsilon}))$ *queries to prefix sampling oracles to both* π *and* μ*.*

The most directly-related result to Theorem 1 is by [MKP25], whose Theorem 1 uses $O(\frac{n^3 m}{\varepsilon^5})$ prefix sampling oracle queries. Theorem 1 is better in terms of sequence length n, accuracy ε, and it saves an $\frac{m}{K}$ factor when the support is sparse[1] . This $\frac{m}{K}$ factor can be quite large: for reference, Qwen has m=151,643 tokens and recommends sampling with $k = 20$ [Qwe26]; such parameters are typical.

Logit queries. Beyond providing sample access, many LLM APIs also return the logits: we can query any prefix $x_{<i}$ and see the full vector $\pi(x_i \mid x_{<i})$. This is much more powerful—for instance, it lets us compute $\pi(x)$ with n queries—leading to a better sample complexity.

Theorem 2 (Informal, cf. Theorems 4, 5)**.** *There is an algorithm that estimates* $\mathrm{TV}(\pi, \mu)$ *up to error* ε *with probability* $\geq \frac{9}{10}$*, using* $O(\frac{n}{\varepsilon^2})$ *queries to prefix logit oracles to both* π *and* μ*. Moreover, this oracle query complexity is tight up to constant factors for any algorithm.*

To our knowledge we are the first paper to formalize the corresponding TV distance estimation problem under logit access. As discussed, the logit variant of the problem is natural in practical

[1] Depending on m rather than K appears inherent to the [MKP25] strategy, which involved "taming" distributions by padding with a $\frac{1}{m}$-scaled uniform distribution to ensure their estimator converges (see discussion after their Theorem 4.8).

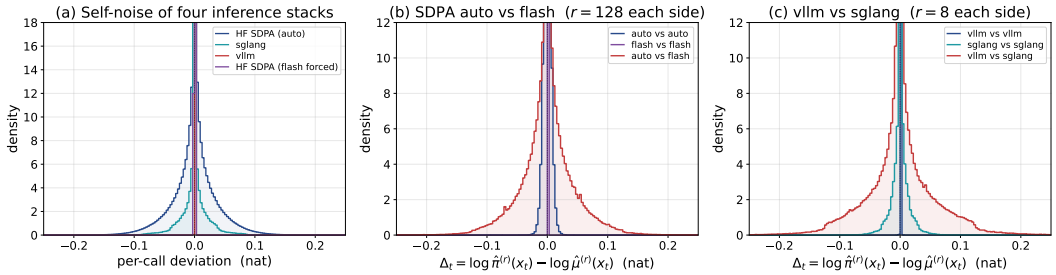


Figure 1: (a) The logits returned by popular inference engines are often not deterministic. For example, the probabilities `sglang` returns typically vary by 5%; but PyTorch SDPA with the Flash Attention 2 backend is deterministic. (b,c) After averaging over several draws, we converge toward the "true" probability; different inference engines converge to systematically different probabilities (e.g. sglang and vllm are typically 5-10% different).

settings where the entire next-token sampling distribution can be queried, and Theorem 5 provides matching upper and lower bounds for its query complexity.

Our upper bound is proven using a simple observation: $O(n)$ queries to prefix logits allows exact computation of a likelihood-ratio $\frac{\pi(x)}{\mu(x)}$, for any sequence $x \in \Sigma^n$. This allows us to repurpose a simple TV estimator from [CR14], which uses $O(\frac{1}{\varepsilon^2})$ likelihood-ratio queries. We complement this algorithm with a matching distribution over hard instances π and μ, indexed by a hidden parameter $p \in [0, 1]$. For this construction, we reduce estimating $\mathrm{TV}(\pi, \mu)$ to learning p up to additive error ε. By carefully defining the hard instance, we show that $\approx n$ prefix logit oracle queries are needed to simulate one toss $\sim \mathrm{Bern}(p)$, at which point $\approx \frac{1}{\varepsilon^2}$ tosses are necessary.

Noisy logits. Unfortunately, while LLM APIs do return logits, those logits are often not reliable. The issues (batching, kernels, precision) that motivate this paper by making LLM implementations unreliable can also make the computation of logits non-deterministic. In Figure 1 we see that some popular open-source LLM implementations return probabilities that regularly vary by 10%.

To model this, for any given prefix $x_{<i}$ we can no longer query the vector $p := \pi(x_i \mid x_{<i})$ directly. Instead, we suppose that we can sample from an unbiased estimator with bounded relative variance; that is, we can sample a random vector $\overline{p}$ such that $\mathbb{E}[\overline{p}] = p$ and

$$\mathbb{E}_{a\sim p}\left[\mathbb{V}\mathrm{ar}\left[\frac{\overline{p}_a}{p_a}\right]\right] \le \sigma^2. \tag{1}$$

This is implied, for instance, by estimating the final logits (before the softmax) to additive $\Theta(\sigma^2)$-variance Gaussian noise, see Lemma 5; and it is empirically validated in Figure 1 (a).

We believe this to be a fairly reasonable model. The challenge and non-determinism in sampling from an LLM comes from computing the logits, not in taking the final softmax; whatever process produces the logits and corresponding $\overline{p}$, the average of $\overline{p}$ really will be the true p. The main caveat is that we need independence across samples.

This "noisy prefix distribution model" *smoothly interpolates* between the previous models (cf. Lemma 4). If $\sigma = 0$, it is precisely the logit query model. With just sample access, we can simply consider the sample $a \sim p$ as the one-hot distribution e_a. This is unbiased with variance (1)

$$\sigma^2 := \sum_{a\in\mathrm{supp}(p)} \frac{1}{p_a}\mathrm{Var}[\overline{p}_a] = \sum_{a\in\mathrm{supp}(p)} (1 - p_a) = |\mathrm{supp}(p)| - 1 < K.$$

Our last main result gives an efficient query complexity bound under noisy logit access.

Theorem 3. *There is an algorithm that estimates* $\mathrm{TV}(\pi, \mu)$ *up to error* ε *with probability* $\ge \frac{9}{10}$, *using* $O(\frac{n+n^2\sigma^2}{\varepsilon^2}\log^2(\frac{1}{\varepsilon}))$ *queries to noisy prefix distribution oracles for both* π *and* μ.

Theorem 3 precisely recovers Theorem 1, and it recovers Theorem 2 up to $\log^2 \frac{1}{\varepsilon}$.

Oracle	Prior upper	Our upper	Prior lower	Our lower
Sampling	$\frac{n^3 m}{\varepsilon^5}$ [MKP25]	$\frac{n^2 K}{\varepsilon^2}$	$\frac{n}{\log n}$ [CDKS20]	$\frac{n}{\varepsilon^2}$
Logit	–	$\frac{n}{\varepsilon^2}$	–	$\frac{n}{\varepsilon^2}$
Noisy logit	–	$\frac{n+n^2\sigma^2}{\varepsilon^2}$	$\frac{n}{\log n}$ [CDKS20]	$\frac{n}{\varepsilon^2}$

Table 1: Prefix query complexity bounds for TV estimation over Σ^n with logarithmic factors omitted. Here $m := |\Sigma|$, and $K \le m$ is the maximum support size of any next-token distribution.

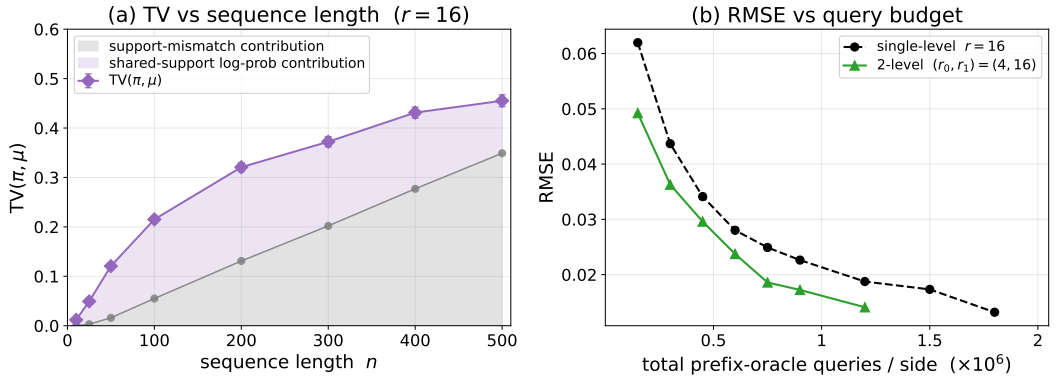


Figure 2: Two production inference engines, `vllm` and `sglang`, serving identical `Qwen3-0.6B` weights are far apart at face value: $\mathrm{TV}(\pi, \mu) \approx 0.45$ at sequence length $n = 500$. *(a)* Decomposing the estimator output against n shows that beyond $n \approx 300$ the bulk comes from *cross-engine support mismatch* — positions where one engine's r-averaged top-k assigns zero mass to a token sampled by the other — on which an empirical KL would return $+\infty$. *(b)* The multilevel schedule $(r_0, r_1) = (4, 16)$ from Algorithm 3 matches the single-level estimator at $r = 16$ at roughly half the oracle queries.

To prove Theorem 3, we first consider an alternative representation of the TV distance (cf. (6) in Lemma 1), which allows us to avoid the "distribution taming" step caused by the unbounded estimators used by the [MKP25] algorithm. As we discuss in Section C.2, this simple change already improves the query complexity of [MKP25] to $\approx \frac{n^2 K}{\varepsilon^4}$. To achieve our tightest result in Section C.3, we use a variance reduction scheme patterned off of multilevel Monte Carlo (MLMC) estimation, a classical technique in statistics [Hei01, Gil15]. Our approach is particularly inspired by an MLMC scheme from [Gil08], which trades off cheaper query complexities with larger variances at different levels of estimation, using a telescoping sum.

We summarize our main theoretical results in Table 1, which illustrates the tradeoff between Theorems 1 and 2 that is interpolated in Theorem 3: the stronger logit query model improves the query complexity by a factor of n and eliminates the dependence on K. An intriguing question which we leave open for future work is whether this separation is inherent, i.e., if superlinear-in-n lower bounds or subquadratic-in-n upper bounds can be proven for the prefix sampling oracle access model.

Experiments. We empirically evaluate Algorithm 3 in Section 4. On a synthetic autoregressive instance with known true TV, the algorithm returns the correct value across a range of noise levels, and a measurable diagnostic guides the choice of multilevel depth on each instance. We then apply it to a real measurement of independent interest: the TV between two production inference engines, `vllm` and `sglang`, serving identical `Qwen3-0.6B` weights; the result and a finer decomposition are summarized in Figure 2.

1.2 Related work

TV estimation under conditional access. TV estimation is a quantitative analogue of distribution testing: an additive estimator immediately yields tolerant and non-tolerant testers by thresholding. With i.i.d. samples, even non-tolerant identity testing over $\{0,1\}^n$ requires $\Omega(2^{n/2})$ samples [CDVV14, VV17], which motivates stronger sampling interfaces. Conditional sampling oracles have been extensively studied in theoretical computer science (TCS), from full conditioning on arbitrary events [CFGM13] to more restricted oracles such as pair [ACK14], interval [CRS15, Nar20], subcube [BC18, CCK+21, CCR+25], and prefix [BCP+24, AFL24] oracles. Subcube and prefix models are closest to the autoregressive sampling setting. However, most prior work focuses on non-tolerant testing rather than distance estimation; see [Can20] for a broader survey.

The closest theoretical works to ours study TV estimation under prefix sampler access. On the upper-bound side, [MKP25, BCP+24, Ada25] consider essentially the same sampler oracle as ours. [MKP25] show that $O(n^3m/\varepsilon^5)$ queries suffice, while [BCP+24] improve the bound to $O(n^2m/\varepsilon^4)$ when one of π, μ is known. Using a different approach, [Ada25] obtain an $O(n^2/\varepsilon^4)$ bound, but only for $m = 2$. On the lower-bound side, [CDKS20] prove an $\Omega(n/\log n)$ lower bound for non-tolerant testing, which also implies hardness for TV estimation. In addition, [Ada25] prove an $\Omega(n^2/\varepsilon^2)$ lower bound for elementwise probability mass estimation. Although this does not rule out an $o(n^2)$ TV estimator, it suggests that sharper bounds may require new techniques. A separate line of work studies TV estimation with numerical probability access: [CR14, BGMV20] assume direct queries to $P(x)$ and $Q(x)$ and show that $\Theta(1/\varepsilon^2)$ queries are necessary and sufficient. Their algorithms can be adapted to our prefix-logit setting, yielding an $O(n/\varepsilon^2)$-query algorithm, but their lower bounds do not apply because their hard instances do not capture prefix structures.

Auditing distribution shifts in LLMs. Auditing whether an LLM API serves the model it claims to provide has become a central question in trustworthy model deployment. Some works approach this problem through downstream task performance [EGD+24, CZZ24]. However, task accuracy is only an indirect proxy for model substitution: two models may achieve similar correctness while inducing substantially different output distributions, leading to changes in token usage, response style, or generation quality. A more direct approach is therefore to audit the model-induced output distribution itself, where the core question becomes whether, and by how much, the served distribution differs from the claimed one. Several recent works take this distributional perspective. [GLG24] proposes a model-equality test, corresponding to non-tolerant testing in TCS, and uses MMD as the test statistic. While empirically promising, this approach lacks theoretical guarantees and was later found to be less effective under subtle substitutions, such as lower-precision inference or partial-time model replacement [CSZS25]. [ZYQ+25] strengthens detection via rank-based nonparametric tests, but its complexity and significance analyses rely on assumptions on the log-ranks of model outputs. [AHSL24] reported changes in LLM output distributions when varying setups such as prompt formulation and sampling temperature but did not provide explicit finite-sample error bounds for their estimators. Finally, [AVC25] proposes an algorithm for estimating KL divergence between LLMs, but its analysis assumes bounded estimator variance, which can be very large or even infinite.

2 Preliminaries

In this section, we introduce preliminaries used throughout the paper. Section 2.1 fixes notation; and Section 2.2 defines the oracles used to describe autoregressive models.

2.1 Notation

Probability. For a state space Ω, we let $\mathcal{P}(\Omega)$ denote all probability measures over Ω. For an event $\mathcal{E} \subseteq \Omega$, we let $\mathbb{I}(\mathcal{E})$ denote the corresponding 0-1 indicator random variable, and $\mu(\mathcal{E})$ denote the probability of the event. We let $\mathbb{E}[\cdot]$ and $\mathrm{Var}[\cdot]$ denote the expectation and variance. We frequently use the following distances (and divergences) between two distributions $\mu, \pi \in \mathcal{P}(\Omega)$, where $\mathrm{d}\omega$ denotes a counting measure if Ω is discrete:

$$\begin{aligned} \mathrm{TV}(\mu,\pi) &:= \frac{1}{2}\int_\Omega |\mu(\omega)-\pi(\omega)|\,\mathrm{d}\omega = \sup_{\mathcal{E}\subseteq\Omega} \mu(\mathcal{E})-\pi(\mathcal{E}), \\ \mathrm{KL}(\mu\|\pi) &:= \int_\Omega \log\left(\frac{\mathrm{d}\mu}{\mathrm{d}\pi}(\omega)\right)\mu(\omega)\mathrm{d}\omega, \quad \chi^2(\mu\|\pi) := \int_\Omega \left(\frac{\mu(\omega)}{\pi(\omega)}-1\right)^2 \pi(\omega)\mathrm{d}\omega. \end{aligned} \tag{2}$$

Autoregressive models. Throughout, we let Σ be a finite alphabet with $m := |\Sigma|$, and let Σ^n denote the space of length-n sequences. In an LLM setting, Σ represents a tokenizer vocabulary, and n is a sequence length. When comparing two distributions π and μ, we reserve use of the letter $K \in [m]$ to denote the maximum support size of any (prefix-conditional) next-token distribution, i.e., $\pi(X_{i+1} = \cdot \mid X_1, \ldots, X_i)$ or $\mu(X_{i+1} = \cdot \mid X_1, \ldots, X_i)$. Next-token distributions in $\mathcal{P}(\Sigma)$ are typically implicitly specified as $\mathrm{softmax}(\ell)$ for logits $\ell \in \mathbb{R}^m$, where $\mathrm{softmax}(\ell) \propto \exp(\ell)$.

Finally we introduce some notation for manipulating sequences. For $x \in \Sigma^n$ and $0 \le i \le n$, we write $x_{1:i} = (x_1, \ldots, x_i)$, with $x_{1:0} = \emptyset$. For $\pi \in \mathcal{P}(\Sigma^n)$, let $\pi_{1:i}$ denote the marginal distribution of $X_{1:i}$. We also use $\pi_{x_{1:i}}(a)$ as shorthand for the next-token distribution $\pi(X_{i+1} = a \mid X_{1:i} = x_{1:i})$. For sequences $s \in \Sigma^m$ and $s' \in \Sigma^n$, let $s \oplus s' \in \Sigma^{m+n}$ denote their concatenation. When the domain is clear, we use e_a to denote the a^{th} basis vector; typically, this is used when $a \in \Sigma$ so that $e_a \in \mathcal{P}(\Sigma)$.

2.2 Oracle definitions

We introduce three oracle models for autoregressive distributions, corresponding to different levels of access to an autoregressive model. The strongest model is exact prefix logit access. This corresponds to a white-box setting: one can run the model forward pass at a given prefix and read off the next-token log-probabilities, as if the model weights and inference procedure were directly available.

Definition 1 (Prefix logit oracle)**.** *Let $\pi \in \mathcal{P}(\Sigma^n)$. For any prefix $x_{1:i} \in \Sigma^i$ with $0 \le i < n$, the prefix logit oracle returns the vector of next-token log-probabilities* [2]

$$\mathcal{O}^{\pi}_{\mathrm{logit}}(x_{1:i}) := \big(\log \pi(X_{i+1} = a \mid X_{1:i} = x_{1:i})\big)_{a \in \Sigma} \in \mathbb{R}^m. \tag{3}$$

We also use the notation

$$\mathcal{O}^{\pi}_{\mathrm{logit}}(x_{1:i}, a) := \log \pi(X_{i+1} = a \mid X_{1:i} = x_{1:i}), \qquad a \in \Sigma. \tag{4}$$

It is clear that one query to the vector oracle (3) simulates one scalar oracle query (4) at no additional cost. In the reverse direction, a scalar oracle simulates the full vector oracle using m scalar queries. We ignore this alphabet-size distinction in our main complexity statements under prefix logit oracle access. This convention is natural for LLMs, where a single forward pass produces all logits simultaneously, and log-probabilities differ from logits only by a softmax normalization.

The second model is prefix sampling access. This model is a special case of the more general conditional sampling oracle model that has received substantial attention in the theoretical computer science literature [CFGM13, ACK14, CRS15, BC18]. Under this model, the algorithm cannot directly inspect the model logits, but it can request a token sampled from the model's next-token conditional distribution. This is also a natural abstraction for closed-source or commercial language models that do not allow explicit access under Definition 1.

Definition 2 (Prefix sampler oracle)**.** *Let $\pi \in \mathcal{P}(\Sigma^n)$. For any prefix $x_{1:i} \in \Sigma^i$ with $0 \le i < n$, the prefix sampler oracle returns*

$$\mathcal{O}^{\pi}_{\mathrm{samp}}(x_{1:i}) \sim \pi(\cdot \mid X_{1:i} = x_{1:i}).$$

Finally, we introduce a noisy prefix logit oracle in Definition 3. This is the most general oracle considered in this paper, subsuming both exact prefix logit queries and prefix sampling queries as special cases. The oracle returns an unbiased random sketch of the next-token distribution, with a noise level parameterized by relative variance. Beyond being a convenient parameterization for analysis, this model captures self-inconsistency in LLM inference, where inference-time randomness can perturb the logits and induce different next-token distributions for the same prefix. We substantiate these interpretations in Appendix C.1, and highlight the practical relevance of modeling query noise in Section 4, when we test the distance between an autoregressive model and itself.

Definition 3 (Noisy prefix distribution oracle)**.** *Let $\pi \in \mathcal{P}(\Sigma^n)$. For any prefix $x_{1:i} \in \Sigma^i$ with $0 \le i < n$, the noisy prefix distribution oracle with relative variance σ^2 returns an unbiased vector $\mathcal{O}_{\mathrm{ndist}}(x_{1:i}) \in \mathcal{P}(\Sigma)$, such that $\mathbb{E}\left[\mathcal{O}^{\pi}_{\mathrm{ndist}}(x_{1:i})\right] = \pi_{x_{1:i}}$, and*

$$\mathbb{E}\left[\chi^2\left(\mathcal{O}^{\pi}_{\mathrm{ndist}}(x_{1:i}) \| \pi_{x_{1:i}}\right)\right] = \mathbb{E}_{a \sim \pi_{x_{1:i}}}\left[\mathrm{Var}\left[\frac{\mathcal{O}^{\pi}_{\mathrm{ndist}}(x_{1:i})(a)}{\pi_{x_{1:i}}(a)}\right]\right] \le \sigma^2,$$

where $\mathbb{E}$ on the left and Var *on the right are over the internal randomness of $\mathcal{O}_{\mathrm{ndist}}$.*

[2] If $\pi(X_{1:i} = x_{1:i}) = 0$, the conditional distribution is undefined; in this case, one may define the oracle to return a distinguished symbol $\perp$. Throughout the paper, unless explicitly specified, our analysis will not invoke it.

Lemma 4 proves that Definition 2 is a case of Definition 3 with $\sigma^2 \leq K$. Notably, Definition 2 is the hardest case of Definition 3, so we may always assume that $\sigma^2 \in [0, K]$. This is because we can always simulate Definition 2 by querying $a \sim \mathcal{O}_{\mathrm{ndist}}(x_{1:i})$ and outputting e_a, because $\mathbb{E}[\mathcal{O}_{\mathrm{ndist}}(x_{1:i})] = \pi_{x_{1:i}}$. We also give broader motivation for our relative variance parameterization in Lemma 5.

We use the following unified formulation for our main distance estimation problem under all three oracle models. The distributions π and μ are only accessed through their corresponding oracles, and the goal is to estimate their TV distance to additive accuracy ε.

Problem 1 (TV estimation)**.** *Let $\pi, \mu \in \mathcal{P}(\Sigma^n)$. Assume access to appropriate prefix oracles for μ and π, denoted $\mathcal{O}^\pi, \mathcal{O}^\mu$. Given $\varepsilon, \delta \in (0, 1)$, design an algorithm that outputs $\widehat{d}$ such that*

$$\mathbb{P}\left[\left|\widehat{d} - \mathrm{TV}\left(\mu, \pi\right)\right| \leq \varepsilon\right] \geq 1 - \delta.$$

The query complexity is the total number of calls to $\mathcal{O}^\pi$ and $\mathcal{O}^\mu$.

Across our analysis, we instantiate $\mathcal{O}^\pi$ and $\mathcal{O}^\mu$ with the access models in Definitions 1, 2, and 3.

3 Technical Overview

In this section we give an overview of our main theoretical results, particularly our upper bound Theorem 6 and our lower bound Theorem 5. For associated proofs, we refer to Appendix B and C.

3.1 Estimator construction

We first describe our estimator under noisy prefix distribution access. The starting point is a mixture representation of TV distance that we recall in (6):

$$\mathrm{TV}\left(\pi, \mu\right) = \mathbb{E}_{X \sim M}[Z(X)], \quad M = \frac{\pi + \mu}{2}, \quad Z(x) = \frac{|\pi(x) - \mu(x)|}{\pi(x) + \mu(x)}, x \in \mathrm{supp}(\pi) \cup \mathrm{supp}(\mu).$$

Thus, it suffices to estimate the local discrepancy $Z(X)$ for strings $X \sim M$. Sampling from M is straightforward: we first choose one of π, μ uniformly at random, and then roll out a string token by token using n queries to the corresponding prefix oracle.

For a fixed string x, the probability $\pi(x)$ factors into the product of its prefix conditionals. We estimate each conditional probability by averaging r independent noisy oracle outputs at the corresponding prefix, and then multiply these averaged conditionals to obtain a plug-in estimate $\widehat{\pi}_r(x)$. The estimate $\widehat{\mu}_r(x)$ is defined analogously. We then estimate $Z(x)$ by replacing $\pi(x)$ and $\mu(x)$ with $\widehat{\pi}_r(x)$ and $\widehat{\mu}_r(x)$ in the normalized discrepancy formula.

The main stability issue is whether this product estimator accumulates noise too quickly across the n prefixes. We control this through the χ^2 error, which captures the relative squared error of the trajectory probability estimate. For a trajectory x, the ratio $\widehat{\pi}_r(x)/\pi(x)$ is a product of n one-step relative estimates. Each one-step estimate has second moment $1 + O(\sigma^2/r)$, and independence across prefixes gives

$$\mathbb{E}\big[\chi^2(\widehat{\pi}_r \| \pi)\big] \lesssim (1 + O(\sigma^2/r))^n - 1 = O(n\sigma^2/r),$$

provided $r \gtrsim n\sigma^2$. Hence the product structure causes only a linear accumulation of relative squared error, rather than an exponential one.

This relative-error control is also well matched to the normalized discrepancy $Z(x)$. Although the denominator $\pi(x) + \mu(x)$ can be small, the mixture distribution $M = (\pi + \mu)/2$ downweights such strings by the same quantity. Consequently, the mean-squared error of the plug-in discrepancy is $O(n\sigma^2/r)$. Taking $r = O(n\sigma^2/\varepsilon^2)$ makes the plug-in approximation error $O(\varepsilon)$. A direct Monte Carlo estimator of $\mathbb{E}[\widehat{Z}_r(X)]$ would then require $O(1/\varepsilon^2)$ outer samples, giving suboptimal query complexity $O(n^2\sigma^2/\varepsilon^4)$, a factor $\approx \varepsilon^{-2}$ away from our final bound.

3.2 Variance reduction

We use a multilevel decomposition because directly estimating the finest-level approximation $\theta_L = \mathbb{E}[\widehat{Z}_L(X)]$ is wasteful: each sample of $\widehat{Z}_L$ is expensive, since it uses many oracle repetitions at every

prefix. Instead, we decompose θ_L into a cheap coarse estimate plus a sequence of refinement terms:

$$\theta_L = \theta_0 + \sum_{\ell=1}^{L} (\theta_\ell - \theta_{\ell-1}) = \mathbb{E}[\widehat{Z}_0(X)] + \sum_{\ell=1}^{L} \mathbb{E}\Big[\widehat{Z}_\ell(X) - \widehat{Z}_{\ell-1}(X)\Big].$$

Here $\theta_\ell = \mathbb{E}[\widehat{Z}_\ell(X)]$, and larger ℓ means a more accurate but more expensive plug-in estimator. Thus, each difference $\theta_\ell - \theta_{\ell-1}$ only measures the additional improvement gained by increasing the number of oracle repetitions from level $\ell - 1$ to level ℓ. The key point is that each refinement term has much smaller variance than $\widehat{Z}_\ell(X)$ itself (which is dominated by the intrinsic variance of Z). To see this, we estimate $\theta_\ell - \theta_{\ell-1}$ by drawing one shared sample $X \sim M$ and computing

$$Y_\ell = \widehat{Z}_\ell(X) - \widehat{Z}_{\ell-1}(X).$$

Because both terms are evaluated on the same X, the underlying signal $Z(X)$ cancels:

$$Y_\ell = \big(\widehat{Z}_\ell(X) - Z(X)\big) - \big(\widehat{Z}_{\ell-1}(X) - Z(X)\big).$$

Therefore, Y_ℓ is governed only by the approximation errors of the two adjacent levels. As the level increases, these errors shrink, and hence the variance of the coupled increment decays geometrically. This is where the telescoping scheme saves queries: coarse corrections are cheap but noisy, while fine corrections are expensive but have much smaller variance, so we can allocate samples nonuniformly across levels. Concretely, with $r_\ell = n\sigma^2 2^\ell$, one sample from level ℓ costs $O(nr_\ell) = O(n^2\sigma^2 2^\ell)$, whereas the coupled variance satisfies $\mathrm{Var}(Y_\ell) = O(2^{-\ell})$. Thus it suffices to take $O(2^{-\ell}/\varepsilon^2)$ samples at level ℓ. The total cost of this level is therefore

$$O(n^2\sigma^2 2^\ell) \cdot O\left(\frac{2^{-\ell}}{\varepsilon^2}\right) = O\left(\frac{n^2\sigma^2}{\varepsilon^2}\right).$$

The increasing per-sample cost and decreasing variance exactly offset each other, so each level contributes roughly the same amount. Summing over $L = O(\log(1/\varepsilon))$ levels gives the final query bound up to logarithmic factors, which are incurred due to the levels of the scheme.

3.3 Lower bound

We prove the lower bound in Theorem 5 by embedding a Bernoulli parameter-estimation problem into TV estimation. For simplicity, consider the binary alphabet. We split a length-n string as $x = u \oplus z$, where the first r bits u identify one of $L = 2^r$ blocks and the remaining $t = n - r$ bits specify a leaf inside that block. In each block u, we draw a hidden bit $X_u \sim \mathrm{Bern}(p)$. If $X_u = 0$, the two distributions π and μ are identical inside the block; if $X_u = 1$, they are maximally separated, but only at a hidden pair of sibling leaves. Thus the global TV distance decomposes as $\mathrm{TV}(\pi, \mu) = \frac{1}{L}\sum_{u\in\{0,1\}^r} X_u$. For $L \gtrsim 1/\varepsilon^2$, this average is concentrated around p. Therefore, an ε-accurate TV estimator can distinguish $p = p_0$ from $p = p_1$, where $p_1 - p_0 = \Theta(\varepsilon)$.

The construction hides each bit X_u behind a random suffix prefix. More concretely, within block u, the two distributions only reveal whether they are identical or separated when the algorithm queries the unique hidden parent of the sibling leaves. Before reaching this parent, oracle answers only provide limited information about where the hidden parent is; they do not reveal X_u. Hence learning one useful Bernoulli sample X_u requires locating a hidden string of length $t - 1$ inside the block. The appendix formalizes this through a transcript mutual-information argument: any adaptive algorithm making B prefix-logit queries can reveal only $O(B/t)$ such blocks in expectation.

Now suppose a TV estimator used only B oracle queries. We could simulate its interaction with the hard instance using an unknown coin of bias $p \in \{p_0, p_1\}$, lazily flipping the coin only when the transcript actually reveals some X_u. Since the transcript reveals only $O(B/t)$ blocks, this would give a distinguisher between $\mathrm{Bern}(p_0)$ and $\mathrm{Bern}(p_1)$ using only $O(B/t)$ coin samples. But distinguishing two Bernoulli biases separated by $\Theta(\varepsilon)$ requires $\Omega(1/\varepsilon^2)$ samples. Therefore $B = \Omega(t/\varepsilon^2)$. Taking $r = O(\log(1/\varepsilon))$ ensures concentration of the block average, and when n is sufficiently larger than r, we have $t = \Theta(n)$. This yields the lower bound $\Omega(n/\varepsilon^2)$.

4 Experiments

We empirically evaluate Algorithm 3 in two settings: a synthetic autoregressive instance with controllable noise, where the true TV is known by direct enumeration (Section 4.1), and a real LLM

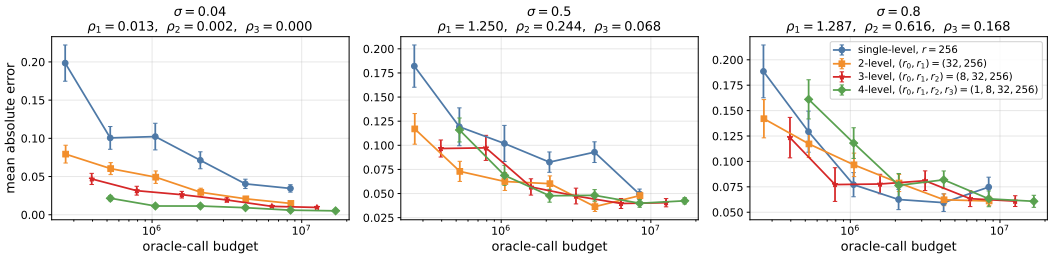


Figure 3: Mean absolute error vs. oracle-call budget for 1-, 2-, 3-, 4-level instantiations of Algorithm 3 on the synthetic hard instance, at three noise levels $\sigma \in \{0.04, 0.5, 0.8\}$. Column titles report the variance ratio ρ_ℓ at each level ℓ of the deepest (4-level) schedule.

deployment, where the TV between two production inference engines serving identical model weights is itself a question of independent interest (Section 4.2). Implementation details, instance parameters, and the methodology behind individual quantitative claims are deferred to Appendix D.

4.1 Validation on a synthetic hard instance

We instantiate the binary autoregressive hard instance of Appendix B.2 with controllable Gaussian logit noise of standard deviation σ. The true $\mathrm{TV}(\pi, \mu) = 0.388$ is known by direct enumeration, which lets us ask two questions: *(i)* does Algorithm 3 return the correct value across the noise range, and *(ii)* does its multilevel structure produce instance-wise cost savings beyond the worst-case bound of Theorem 3? All four schedules in Figure 3 converge to the true TV at every σ, settling *(i)*. Question *(ii)* reduces to choosing the right multilevel depth, which we can read off the data through a measurable diagnostic.

The cost benefit of including each level is captured by the empirical variance ratio

$$\rho_\ell := \frac{\mathrm{Var}[Y_\ell]}{\mathrm{Var}[Z_\ell]} = \frac{\mathrm{Var}(Z_\ell - Z_{\ell-1})}{\mathrm{Var}(Z_\ell)},$$

where Y_ℓ and Z_ℓ are the coupled level-ℓ increment and estimator from described formally from (12)–(16). When $\rho_\ell \ll 1$, the cheaper level $\ell - 1$ provides an essentially free variance reduction at level ℓ, and including it strictly beats the single-level estimator at r_ℓ at matched RMSE; when ρ_ℓ is order one, level $\ell - 1$ adds cost without reducing variance and should be dropped.

Figure 3 sweeps $\sigma \in \{0.04, 0.5, 0.8\}$ across four schedule depths under matched query budget, with ρ_ℓ for each level reported in the column titles. The empirical winner tracks ρ_ℓ at every σ: at $\sigma = 0.04$ all $\rho_\ell < 0.02$ and the deepest (4-level) schedule beats the single-level estimator at $r = 256$ by $\approx 7\times$ at matched RMSE; as σ grows, ρ_1 rises past 1 and the optimal depth shrinks; at $\sigma = 0.8$ the 2-level schedule wins. Since ρ_ℓ is itself measurable on a small calibration sample, a practitioner can pick the depth before paying the full estimation budget.

4.2 Application: cross-engine TV between vllm and sglang

We apply Algorithm 3 to a real measurement of independent interest: the TV between two production inference engines, `vllm` and `sglang`, serving identical `Qwen3-0.6B` weights with top-$k = 20$ truncation in bf16 precision. The two engines differ in attention-kernel choice, batching strategy, and execution scheduler, but should in principle realize the same next-token distribution at every prefix; the empirical question is how different they are at face value.

The estimator returns $\mathrm{TV}(\pi, \mu) = 0.455 \pm 0.012$ at length $n = 500$. The decomposition in Figure 2(a) localizes the bulk of this distance: at $n = 500$, 77% of the estimator output comes from *cross-engine support mismatch* — positions where one engine's r-averaged top-k assigns zero mass to a token sampled by the other — rather than from log-probability drift on the shared support. Each such event sends the trajectory's log-likelihood ratio to $+\infty$, so the empirical r-averaged KL is $+\infty$ on more than a third of trajectories at $n = 500$ and cannot distinguish a small cross-engine

disagreement from a large one. Algorithm 3 replaces each such event with the bounded contribution 1 from (12) and returns a finite estimate.

The ρ_ℓ diagnostic of Section 4.1 transfers to this real-data setting: with levels $r_0 = 4$, $r_1 = 16$ we measure $\rho_1 = 0.06$, putting this instance in the multilevel-helps regime; the corresponding two-level schedule cuts the prefix-oracle budget by roughly half at matched RMSE (Figure 2(b)).

5 Conclusion

Measuring the KL divergence between LLMs has become a basic tool in LLM research, while the TV distance, the most natural alternative to KL, remains underexplored. The KL and TV have complementary strengths: the former is unbounded and sensitive to changes in the tail, whereas the latter can saturate quickly for large distribution changes. Motivated by the strengths of TV as an estimable quantity from finite samples, in this paper, we studied how to efficiently measure the TV distance between the output distributions of two LLMs.

To capture the situation when measuring real-world deployments, we propose three levels for modeling the query access given by LLMs: *sample-only access, exact logit access, and noisy logit access*, the last of which reflects the nondeterminism of LLM inference. We designed algorithms that are more sample-efficient than prior work under all three access models, proved tightness of our exact logit access upper bound, and validated the practicality of our estimators experimentally.

Our experiments show that, in settings like distance estimation where the object of interest is the distribution change itself, the noisy-logit model is both faithful and necessary. We thus believe that our formulation of this model holds independent interest beyond TV estimation. For instance, the existing literature on measuring KL divergence generally treats the returned logits as exact, without accounting for nondeterminism.

In the last, we highlight some considerations for practitioners using our estimators.

1. The measured value is sensitive to the measurement condition. For the same model weights, serving choices such as the kernels or the batching shift the output distribution itself (Figure 1). Distances should therefore be reported with stated measurement condition.
2. Scoring the same next-token prediction repeatedly denoises the measurement. Without denoising, the estimated TV distance is systematically inflated. One caveat of such denoising is that repeated scores are not perfectly independent, since each evaluation can alter the system state seen by the next. In our experimental setup, this correlation is small enough that the repeats are approximately independent, and repeated scoring achieves its purpose.
3. All scoring must follow the generation path exactly. A sequence-scoring API, for example, invokes batching strategies and other execution details that differ from those of generation. Consequently, it produces scores different from the distribution the engine generates from.
4. The magnitude of the oracle noise differs across inference engines. A small pilot sample is enough to learn the noise of the engine at hand: on the pilot, given the target accuracy of the final measurement, one can directly select the best denoising level or multilevel schedule, and then use it when scoring at full scale.

Acknowledgments

This work is supported by the NSF AI Institute for Foundations of Machine Learning (IFML).

References

[ACK14] Jayadev Acharya, Clément L Canonne, and Gautam Kamath. A chasm between identity and equivalence testing with conditional queries. *arXiv preprint arXiv:1411.7346*, 2014.

[Ada25] Tomer Adar. Tight simulation of a distribution using conditional samples. *arXiv preprint arXiv:2506.18444*, 2025.

[AFL24] Tomer Adar, Eldar Fischer, and Amit Levi. Improved bounds for high-dimensional equivalence and product testing using subcube queries. *arXiv preprint arXiv:2408.02347*, 2024.

[AHSL24] Julian Arnold, Flemming Holtorf, Frank Schäfer, and Niels Lörch. Phase transitions in the output distribution of large language models. *arXiv preprint arXiv:2405.17088*, 2024.

[Art26] Artificial Analysis. gpt-oss-120B (high): API provider performance benchmarking & price analysis. `https://artificialanalysis.ai/models/gpt-oss-120b/providers#evaluations`, 2026. Accessed: 2026-05-04.

[AVC25] Afra Amini, Tim Vieira, and Ryan Cotterell. Better estimation of the kullback–leibler divergence between language models. *arXiv preprint arXiv:2504.10637*, 2025.

[BC18] Rishiraj Bhattacharyya and Sourav Chakraborty. Property testing of joint distributions using conditional samples. *ACM Transactions on Computation Theory (TOCT)*, 10(4):1–20, 2018.

[BCP+24] Rishiraj Bhattacharyya, Sourav Chakraborty, Yash Pote, Uddalok Sarkar, and Sayantan Sen. Testing self-reducible samplers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7952–7960, 2024.

[BGMV20] Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S Meel, and NV Vinodchandran. Efficient distance approximation for structured high-dimensional distributions via learning. *Advances in Neural Information Processing Systems*, 33:14699–14711, 2020.

[Can20] Clément L Canonne. A survey on distribution testing: Your data is big. but is it blue? *Theory of Computing*, pages 1–100, 2020.

[Can22] Clément L Canonne. A short note on an inequality between kl and tv. *arXiv preprint arXiv:2202.07198*, 2022.

[CCK+21] Clément L Canonne, Xi Chen, Gautam Kamath, Amit Levi, and Erik Waingarten. Random restrictions of high dimensional distributions and uniformity testing with subcube conditioning. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 321–336. SIAM, 2021.

[CCR+25] Deeparnab Chakrabarty, Xi Chen, Simeon Ristic, C Seshadhri, and Erik Waingarten. Monotonicity testing of high-dimensional distributions with subcube conditioning. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1019–1030, 2025.

[CDKS20] Clement Canonne, Ilias Diakonikolas, Daniel Kane, and Alistair Stewart. Testing bayesian networks, 2020.

[CDVV14] Siu-On Chan, Ilias Diakonikolas, Paul Valiant, and Gregory Valiant. Optimal algorithms for testing closeness of discrete distributions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1193–1203. SIAM, 2014.

[CFGM13] Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 561–580, 2013.

[Cla25] Chris Clark. Provider variance: Introducing exacto. `https://openrouter.ai/announcements/provider-variance-introducing-exacto`, October 2025. OpenRouter Announcement.

[CR14] Clément Canonne and Ronitt Rubinfeld. Testing probability distributions underlying aggregated data. In *International Colloquium on Automata, Languages, and Programming*, pages 283–295. Springer, 2014.

[CRS15] Clément L Canonne, Dana Ron, and Rocco A Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Computing*, 44(3):540–616, 2015.

[CSZS25] Will Cai, Tianneng Shi, Xuandong Zhao, and Dawn Song. Are you getting what you pay for? auditing model substitution in llm apis. *arXiv preprint arXiv:2504.04715*, 2025.

[CTLI25] Xiaoyu Chu, Sacheendra Talluri, Qingxian Lu, and Alexandru Iosup. An empirical characterization of outages and incidents in public services for large language models. In *Proceedings of the 16th ACM/SPEC International Conference on Performance Engineering, ICPE 2025*, pages 69–80. ACM, 2025.

[CZZ24] Lingjiao Chen, Matei Zaharia, and James Zou. How is chatgpt's behavior changing over time? *Harvard Data Science Review*, 6(2), 2024.

[DKKR24] Abhinav Dutta, Sanjeev Krishnan, Nipun Kwatra, and Ramachandran Ramjee. Accuracy is not all you need. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[EGD+24] Sabri Eyuboglu, Karan Goel, Arjun Desai, Lingjiao Chen, Mathew Monfort, Chris Ré, and James Zou. Model changelists: Characterizing updates to ml models. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, pages 2432–2453, 2024.

[FAHA22] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[Fir24] Fireworks AI. How fireworks evaluates quantization precisely and interpretably, 2024. Fireworks AI Blog.

[Gil08] Michael B Giles. Multilevel monte carlo path simulation. *Operations research*, 56(3):607–617, 2008.

[Gil15] Michael B Giles. Multilevel monte carlo methods. *Acta numerica*, 24:259–328, 2015.

[GLG24] Irena Gao, Percy Liang, and Carlos Guestrin. Model equality testing: Which model is this api serving? In *The Thirteenth International Conference on Learning Representations*, 2024.

[GS02] Alison L Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International statistical review*, 70(3):419–435, 2002.

[Hei01] Stefan Heinrich. Multilevel monte carlo methods. In *International Conference on Large-Scale Scientific Computing*, pages 58–67. Springer, 2001.

[HKM+24] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length LLM inference with KV cache quantization. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024*, 2024.

[HT25] Horace He and Thinking Machines Lab. Defeating nondeterminism in LLM inference. *Thinking Machines Lab: Connectionism*, September 2025. `https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/`.

[LTT+24] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6:87–100, 2024.

[LWZZ24] Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-2: faster inference of language models with dynamic draft trees. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024*, pages 7421–7432. Association for Computational Linguistics, 2024.

[LYJ+24] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen (Henry) Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *Forty-first International Conference on Machine Learning, ICML 2024*, Proceedings of Machine Learning Research, pages 32332–32344. PMLR / OpenReview.net, 2024.

[MKP25] Kuldeep S Meel, Gunjan Kumar, and Yash Pote. Distance estimation for high-dimensional discrete distributions. In *International Conference on Artificial Intelligence and Statistics*, pages 955–963. PMLR, 2025.

[Nar20] Shyam Narayanan. On distribution testing in the conditional sampling model. *arXiv preprint arXiv:2007.09895*, 2020.

[Qwe26] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026.

[SV16] Igal Sason and Sergio Verdú. f-divergence inequalities. *IEEE Transactions on Information Theory*, 62(11):5973–6006, 2016.

[VH14] Ramon Van Handel. Probability in high dimension. Technical report, Princeton University, 2014.

[VV17] Gregory Valiant and Paul Valiant. An automatic inequality prover and instance optimal identity testing. *SIAM Journal on Computing*, 46(1):429–455, 2017.

[ZLR+24] Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. Distillspec: Improving speculative decoding via knowledge distillation. In *The Twelfth International Conference on Learning Representations, ICLR 2024*. OpenReview.net, 2024.

[ZYQ+25] Xiaoyuan Zhu, Yaowen Ye, Tianyi Qiu, Hanlin Zhu, Sijun Tan, Ajraf Mannan, Jonathan Michala, Raluca Ada Popa, and Willie Neiswanger. Auditing black-box llm apis with a rank-based uniformity test. *arXiv preprint arXiv:2506.06975*, 2025.

A Properties of TV Distance

Here we derive some equivalent formulations of the TV distance used in our estimation algorithms. We use the notation $(x)_+ := \max(x, 0)$.

Lemma 1 (Equivalent representations of total variation)**.** *Let $\pi, \mu \in \mathcal{P}(\Sigma^n)$, then* $\mathrm{TV}(\pi, \mu)$ *admits the following equivalent representations:*

$$\mathrm{TV}(\pi, \mu) = \mathbb{E}_{X\sim\pi}\left[\left(1 - \frac{\mu(X)}{\pi(X)}\right)_+\right], \tag{5}$$

$$\mathrm{TV}(\pi, \mu) = \mathbb{E}_{X\sim\frac{\pi+\mu}{2}}\left[\frac{|\pi(X) - \mu(X)|}{\pi(X) + \mu(X)}\right]. \tag{6}$$

Proof. By definition,

$$\mathrm{TV}(\pi, \mu) = \frac{1}{2}\sum_{x\in\Sigma^n} |\mu(x) - \pi(x)| = \sum_{x\in\Sigma^n} (\pi(x) - \mu(x))_+.$$

We first prove (5) by noting that

$$\mathbb{E}_{X\sim\pi}\left[\left(1 - \frac{\mu(X)}{\pi(X)}\right)_+\right] = \sum_{x\in\mathrm{supp}(\pi)} \pi(x)\left(1 - \frac{\mu(x)}{\pi(x)}\right)_+ = \sum_{x\in\Sigma^n} (\pi(x) - \mu(x))_+ = \mathrm{TV}(\pi, \mu).$$

A similar direct calculation yields (6):

$$\mathbb{E}_{X\sim\frac{\pi+\mu}{2}}\left[\frac{|\pi(X) - \mu(X)|}{\pi(X) + \mu(X)}\right] = \frac{1}{2}\sum_{x\in\Sigma^n} |\pi(x) - \mu(x)| = \mathrm{TV}(\pi, \mu).$$

□

B TV Estimation via Prefix Logit Queries

In this section, we study Problem 1 under access to the prefix logit oracles $(\mathcal{O}^{\pi}_{\text{logit}}, \mathcal{O}^{\mu}_{\text{logit}})$. We first present our distance estimator and prove its query-complexity upper bound in Appendix B.1. The key observation is that exact prefix logit access simulates an exact log-probability oracle over full trajectories: for any $x \in \Sigma^n$,

$$\ell_\pi(x) := \log \pi(x_{1:n}) = \sum_{i=0}^{n-1} \log \pi(x_{i+1} \mid x_{1:i}), \tag{7}$$

and a similar expression holds for μ. Hence, density evaluations for π or μ cost n prefix logit oracle queries. We adapt the likelihood-ratio estimator of [CR14], which is a bounded, unbiased estimator for $\text{TV}(\pi, \mu)$ leveraging query access to $\frac{\mu}{\pi}$ via (5), achieving query complexity $O\left(\frac{n}{\varepsilon^2} \log \frac{1}{\delta}\right)$.

In Appendix B.2, we show that this reduction is tight in the worst case for any algorithm with prefix logit oracle access. In particular, our lower bound construction implies that, when $n \gg 1$, any algorithm solving Problem 1 with access to $(\mathcal{O}^{\pi}_{\text{logit}}, \mathcal{O}^{\mu}_{\text{logit}})$ requires $\Omega(\frac{n}{\varepsilon^2})$ queries.

B.1 Likelihood-ratio estimator

Our estimator in Algorithm 1 follows from the likelihood-ratio representation in equation (5): it is the empirical mean of a truncated likelihood-ratio functional evaluated on samples from one of the two distributions. To implement Algorithm 1, we need two primitives: (a) sampling a trajectory from one distribution, and (b) evaluating the probability of a given trajectory under either π or μ. Both primitives are available from accessing prefix logit oracles. Firstly, sampling can be performed sequentially by querying the prefix logit oracle, and drawing the next token appropriately. Moreover, given a trajectory $x_{1:n}$, its log-probability is computed via (7). Each primitive therefore requires n queries. Since the estimator averages bounded independent summands, standard concentration inequalities recalled in Theorem 4 give the desired high-probability guarantee.

Algorithm 1 $\mathsf{LogitAccessTV}(\mathcal{O}^{\pi}_{\text{logit}}, \mathcal{O}^{\mu}_{\text{logit}}, \varepsilon, \delta)$

$N \leftarrow \left\lceil \frac{\log \frac{2}{\delta}}{2\varepsilon^2} \right\rceil$
Generate $X_1, \ldots, X_N \overset{\text{i.i.d.}}{\sim} \pi$ via $\mathcal{O}^{\pi}_{\text{logit}}$
for $i = 1, \ldots, N$ **do**
    $L_{\pi,i} \leftarrow \log \pi(X_i)$, $L_{\mu,i} \leftarrow \log \mu(X_i)$ ▷ Evaluate using n prefix logit oracle queries via (7).
    $R_i \leftarrow (1 - \exp(L_{\mu,i} - L_{\pi,i}))_+$
end for
$\hat{d} \leftarrow \frac{1}{N} \sum_{i=1}^{N} R_i$
return $\hat{d}$

Theorem 4. *Algorithm 1 solves Problem 1 with $O(\frac{n}{\varepsilon^2} \log \frac{1}{\delta})$ queries to $(\mathcal{O}^{\pi}_{\text{logit}}, \mathcal{O}^{\mu}_{\text{logit}})$.*

Proof. Letting $\log 0 := -\infty$, we obtain that $R_i = (1 - \frac{\mu(X_i)}{\pi(X_i)})_+$. By Lemma 1, we know that $\mathbb{E}[R_i] = \text{TV}(\pi, \mu)$. Moreover, since $\{R_i\}_{i \in [N]} \in [0, 1]$ are independent, Hoeffding's inequality gives

$$\mathbb{P}\left[\left|\frac{1}{N} \sum_{i=1}^{N} R_i - \text{TV}(\pi, \mu)\right| \geq \varepsilon\right] \leq 2\exp(-2N\varepsilon^2).$$

Taking N as stated gives the desired probability bound. Each sequence log-likelihood evaluation and sample generation uses n queries, which gives us $3nN = O(\frac{n}{\varepsilon^2} \log \frac{1}{\delta})$ query complexity bound. □

B.2 Lower bound

In this subsection, we take $\Sigma = \{0, 1\}$ to be a binary alphabet. We provide a hard instance showing that, for constant δ, Theorem 4 achieves the optimal query complexity up to a constant factor.

Intuitively, this hard instance reduces to estimating a parameter $p \in (0,1)$ up to error ε, in a way such that it takes $\approx n$ prefix logit queries to reveal one sample $\sim \mathrm{Bern}(p)$.

Hard instance construction. Fix $r \in [1, n-2]$, let $t = n - r$, and set $L := |\Sigma^r| = 2^r$. We divide any $x \in \Sigma^n$ into two parts: $x = u \oplus z$, where $u \in \Sigma^r, z \in \Sigma^t$. The prefix u serves as a block index, while the suffix z specifies a leaf inside that block. In each block, we randomly choose whether $\mu_u, \pi_u \in \mathcal{P}(\Sigma^t)$ are identical or separated. If they are separated, they share the same $(t-1)$-bit prefix inside the block and differ only at the final bit, so the discrepancy takes t queries to reveal.

Formally, for $p \in [0,1]$, define $(\mu, \pi) \sim \mathcal{D}_p$ as follows. For each $u \in \Sigma^r$, draw

$$X_u \overset{\text{i.i.d.}}{\sim} \mathrm{Bern}(p), \qquad Z_u \overset{\text{i.i.d.}}{\sim} \mathrm{Unif}(\{0,1\}^{t-1}), \qquad B_u \overset{\text{i.i.d.}}{\sim} \mathrm{Unif}(\{0,1\}).$$

Let $a_u := Z_u \oplus B_u$, $b_u := Z_u \oplus (1 - B_u)$, be sibling leaves. Define distributions $\mu_u, \pi_u \in \mathcal{P}(\Sigma^t)$ as

$$(\mu_u, \pi_u) = \begin{cases} \left(\frac{1}{2}\delta_{a_u} + \frac{1}{2}\delta_{b_u}, \frac{1}{2}\delta_{a_u} + \frac{1}{2}\delta_{b_u}\right), & X_u = 0, \\ \left(\delta_{a_u}, \delta_{b_u}\right), & X_u = 1. \end{cases}$$

Finally, let $\mu(u \oplus z) \coloneqq \frac{1}{L}\mu_u(z)$, $\pi(u \oplus z) \coloneqq \frac{1}{L}\pi_u(z)$. A sample from μ is generated by drawing $u \sim \mathrm{Unif}(\Sigma^r)$ and then drawing $z \sim \mu_u$; a sample from π is generated analogously using π_u.

This construction makes the TV distance easy to compute. Since different prefixes u define disjoint blocks, the global distance decomposes as the average of the suffix distances:

$$\mathrm{TV}(\mu, \pi) = \frac{1}{L}\sum_{u \in \Sigma^r} \mathrm{TV}(\mu_u, \pi_u) = \frac{1}{L}\sum_{u \in \Sigma^r} X_u. \tag{8}$$

The above display shows that, under $(\mu, \pi) \sim \mathcal{D}_p$, $\mathrm{TV}(\mu, \pi)$ is exactly the empirical average of L i.i.d. Bernoulli variables. Hence, $\mathrm{TV}(\mu, \pi)$ is close to p with high probability. Therefore, we can show that an accurate estimate of $\mathrm{TV}(\mu, \pi)$ distinguishes the two hypotheses

$$H_0 : (\mu, \pi) \sim \mathcal{D}_{p_0}, \qquad H_1 : (\mu, \pi) \sim \mathcal{D}_{p_1},$$

whenever $p_1 - p_0$ is sufficiently large.

Lemma 2. *Let $0 \le p_0 < p_1 \le 1$ and set $\Delta \coloneqq \frac{p_1 - p_0}{4}$. Suppose $\mathcal{A}$ estimates $\mathrm{TV}(\mu, \pi)$ within additive error Δ with probability at least $\frac{2}{3}$ for every fixed pair (μ, π). Then there is a tester distinguishing $\mathcal{D}_{p_0}$ from $\mathcal{D}_{p_1}$ with success probability at least $\frac{2}{3} - 2\exp(-2L\Delta^2)$.*

Proof. By (8) and Hoeffding's inequality, we know that for $(\mu, \pi) \sim \mathcal{D}_p$,

$$\mathbb{P}\left[|\mathrm{TV}(\mu, \pi) - p| > \Delta\right] \le 2\exp(-2L\Delta^2).$$

The tester runs $\mathcal{A}$ and outputs p_1 iff $\hat{d} \ge \frac{p_0 + p_1}{2}$. Conditioned on the success of the tester and $|\mathrm{TV}(\mu, \pi) - p| \le \Delta$, we have $|\hat{d} - p| \le 2\Delta = \frac{p_1 - p_0}{2}$. Therefore, if $p = p_0$, then $\widehat{d} \le \frac{p_0 + p_1}{2}$; if $p = p_1$, then $\widehat{d} \ge \frac{p_0 + p_1}{2}$. Hence the tester succeeds on these two events. Applying a union bound gives a success probability of at least $\frac{2}{3} - 2\exp(-2L\Delta^2)$. □

Lemma 2 shows it is enough to lower bound the oracle query complexity of estimating $\mathrm{TV}(\mu, \pi)$. Intuitively, we wish to show that many (u, X_u) must be revealed to perform this estimate.

Lemma 3 shows that with only B prefix queries, any adaptive algorithm can reveal at most $O(\frac{B}{t})$ such blocks u in expectation. We formalize this through the query-answer transcript. A possibly randomized adaptive algorithm chooses each query $W_i \in \Sigma^{<n}$ as a function of its private randomness and the previous query-answer pairs. We write

$$\mathsf{Tr} = \big((W_1, \mathcal{O}(W_1)), \ldots, (W_T, \mathcal{O}(W_T))\big)$$

for the full transcript, where $\mathcal{O}(W_i)$ is the oracle answer and $T \le B$ is the stopping time. For simplicity, we assume that $\mathcal{O}(W_i)$ reveals both $\mathcal{O}^{\pi}_{\mathrm{logit}}(W_i)$ and $\mathcal{O}^{\mu}_{\mathrm{logit}}(W_i)$, which is without loss of generality up to a constant factor in the oracle query complexity.

The crux of Lemma 3's argument is that X_u can affect the transcript only if the algorithm queries the hidden parent $u \oplus Z_u$. Hence, before learning anything about X_u, the algorithm must locate the hidden string $Z_u \in \{0,1\}^{t-1}$. We capture this notion formally by bounding the mutual information between the transcript and the initially-random set of strings $\mathcal{Z} = (Z_u)_{u \in \Sigma^r}$.

Lemma 3. *Consider any adaptive algorithm interacting with the oracle in Definition 1 on an instance (μ, π) drawn from $\mathcal{D}_p$. Suppose its worst-case query complexity is B. Let $\mathcal{G} \subseteq \Sigma^r$ be the set of $u \in \Sigma^r$ for which $W_j = u \oplus Z_u$ for some $j \le T$. There is a universal constant $C > 0$ such that*

$$\mathbb{E}[|\mathcal{G}|] \le C \cdot \frac{B}{t}.$$

Proof. Throughout, let q denote a random seed specifying internal randomness, which is drawn and fixed before interactions with $\mathcal{O}$. Thus, the algorithm is deterministic conditioned on q.

Let $\ell = t - 1 \ge \frac{t}{2}$ and let $\mathcal{Z} = (Z_u)_{u \in \Sigma^r}$. Inside any block $u \in \Sigma^r$, both μ_u and π_u are supported on the two sibling leaves $Z_u \oplus B_u$ and $Z_u \oplus (1 - B_u)$. Hence, before $u \oplus Z_u$ is queried, the oracle only reveals whether the queried prefix is consistent with Z_u and, if so, reveals the next bit of Z_u.[3] When the hidden $u \oplus Z_u$ is queried, the answer is $(\frac{1}{2}(e_0 + e_1), \frac{1}{2}(e_0 + e_1))$ if $X_u = 0$, and is $(\delta_{B_u}, \delta_{1-B_u})$ if $X_u = 1$. Therefore X_u can affect the oracle answer only at the query $u \oplus Z_u$.

Let Tr be the full transcript. We first upper bound the mutual information $I(\mathcal{Z}; \mathsf{Tr} \mid s)$. As shown above, each oracle answer lies in a constant-sized set so each query contributes at most $h_0 = O(1)$ bits of mutual information about $\mathcal{Z}$. Because the algorithm makes at most B queries, we have $I(\mathcal{Z}; \mathsf{Tr} \mid q) \le h_0 B$. Averaging over the internal randomness q, we obtain $I(\mathcal{Z}; \mathsf{Tr}) \le h_0 B$.

Conversely, if $u \in \mathcal{G}$, then the transcript contains a query of the form $u \oplus Z_u$, so it determines the full string Z_u. Thus, after observing the transcript, revealed blocks u have no remaining entropy in their hidden parents, while each unrevealed block u has at most ℓ bits of remaining entropy. Hence

$$H(\mathcal{Z} \mid \mathsf{Tr}) \le \ell\big(L - \mathbb{E}[|\mathcal{G}|]\big).$$

Since the unconditional entropy $H(\mathcal{Z}) = L\ell$, we get

$$I(\mathcal{Z}; \mathsf{Tr}) = H(\mathcal{Z}) - H(\mathcal{Z} \mid \mathsf{Tr}) \ge \ell\, \mathbb{E}[|\mathcal{G}|].$$

Combining the upper and lower bounds gives

$$\mathbb{E}[|\mathcal{G}|] \le \frac{h_0 B}{\ell} \le 2h_0 \frac{B}{t},$$

where we used $\ell = t - 1 \ge \frac{t}{2}$. □

We are now ready to prove the lower bound. Lemma 2 shows that a TV estimator for our random instance can distinguish whether the underlying bias is p_0 or p_1. In the construction, this bias appears only through the variables $X_u \sim \mathrm{Bern}(p)$; conditioned on these variables, the remaining randomness in Z_u and B_u is independent of p.

Thus, given an unknown coin of bias $p \in \{p_0, p_1\}$, we can simulate the oracle for $(\mu, \pi) \sim \mathcal{D}_p$ by lazily drawing a coin sample only when the transcript reveals some X_u. By Lemma 3, a cost-B transcript reveals only $O(\frac{B}{t})$ such blocks in expectation. Hence a cost-B TV estimator would yield a coin distinguisher using only $O(\frac{B}{t})$ coin samples. Combining this result with classic sample complexity lower bounds for Bernoulli parameter estimation yields Theorem 5.

Theorem 5. *Fix any $\varepsilon \in (0, \frac{1}{8})$, and let $n = \Omega(\log \frac{1}{\varepsilon})$ for a sufficiently large constant. Any algorithm that solves Problem 1 with probability at least $\frac{2}{3}$ requires $\Omega(\frac{n}{\varepsilon^2})$ queries to $(\mathcal{O}^{\pi}_{\mathrm{logit}}, \mathcal{O}^{\mu}_{\mathrm{logit}})$.*

Proof. Set $p_0 = \frac{1}{2}, p_1 = \frac{1}{2} + 4\varepsilon$. By assumption we can set $r = \Omega(\log \frac{1}{\varepsilon})$ large enough so that $n \ge 2r$ and the error term in Lemma 2 satisfies $2\exp(-2L\varepsilon^2) \le \frac{1}{100}$. Suppose that there is an estimator $\mathcal{A}$ with worst-case prefix-query cost B. By Lemma 2, $\mathcal{A}$ yields a distinguisher between

$$(\mu, \pi) \sim \mathcal{D}_{p_0} \qquad \text{and} \qquad (\mu, \pi) \sim \mathcal{D}_{p_1}$$

with success probability at least $\frac{2}{3} - \frac{1}{100} > \frac{3}{5}$.

We now simulate this distinguisher in an equivalent way, using an unknown coin of bias $p \in \{p_0, p_1\}$. The simulator samples all of the hidden Z_u and B_u using its own randomness, but does not sample X_u in advance. Whenever the distinguishing algorithm reveals an X_u, equivalently when some

[3] If not, the model returns the distinguished symbol $(\perp, \perp)$, which is included in the transcript and carries only $O(1)$ bits.

$W_i = u \oplus Z_u$ in the transcript, the simulator flips the unknown coin and stores it as X_u. By lazy sampling, the simulated transcript has exactly the same law as the true transcript under $\mathcal{D}_p$.

Let N be the number of coin samples used by the simulator. Since each coin sample corresponds to one revealed block, Lemma 3 implies $\mathbb{E}[N] \le C\frac{B}{t}$. By Markov's inequality, we may truncate the simulation so that it uses at most $100C\frac{B}{t}$ coin samples while losing only $\frac{1}{100}$ of success probability. Thus, we obtain a tester distinguishing $\mathrm{Bern}(p_0)$ from $\mathrm{Bern}(p_1)$ with success probability $> \frac{3}{5}$ using $O(\frac{B}{t})$ coin samples. Classical results, e.g., [Can22], imply that distinguishing $\mathrm{Bern}(p_0)$ from $\mathrm{Bern}(p_1)$ requires $\Omega\left(\frac{1}{\varepsilon^2}\right)$ samples, which implies $B = \Omega\left(\frac{t}{\varepsilon^2}\right) = \Omega\left(\frac{n}{\varepsilon^2}\right)$. □

Remark 1 (Extension to weaker prefix oracles)**.** *The same lower bound also applies to weaker access models, such as noisy prefix distribution or prefix sampler oracles. The only oracle-specific part of the proof is Lemma 3, where we show that a transcript of B prefix queries can reveal at most $O(B/t)$ hidden blocks in expectation. This step is information-theoretic: the bottleneck is locating the hidden parent $u \oplus Z_u$, before which the transcript contains no information about the block variable X_u. Since noisy distribution access and sampler access are no more informative than exact prefix-logit access, the same mutual-information bound on the revealed set $\mathcal{G}$ continues to hold. The remaining reduction from TV estimation to distinguishing $\mathrm{Bern}(p_0)$ from $\mathrm{Bern}(p_1)$ is unchanged, and therefore these weaker oracle models inherit the same $\Omega(n/\varepsilon^2)$ lower bound.*

C TV Estimation via Noisy Prefix Distribution Queries

In this section, we study Problem 1 under access to the noisy prefix distribution oracles $(\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}})$. We begin by motivating the noisy prefix distribution oracle definition in Definition 3 and justifying the claims from Section 2.2, particularly, that our results for Definition 3 with $\sigma^2 = k$ capture the prefix sampling oracle in Definition 2. We then present our basic estimator and its approximation guarantee in Appendix C.2, followed by an improved variance reduction argument based on multilevel Monte Carlo in Appendix C.3 that yields our final query complexity bound.

C.1 Motivation for the relative variance model

Here we give two basic justifications for the relative variance parameterization used in our noisy prefix distribution oracle (Definition 3). First, under this parameterization, Definition 3 subsumes the two oracle models introduced earlier: exact logit queries correspond to zero-noise sketches, while prefix-conditional samples correspond to random one-hot probability vectors with $\sigma^2 \le K$.

Lemma 4. *A prefix logit oracle (Definition 1) is a noisy prefix distribution oracle (Definition 3) with $\sigma^2 = 0$. Moreover, drawing $a \in \Sigma$ from a prefix sampler oracle (Definition 2) and outputting $e_a \in \mathcal{P}(\Sigma)$ is a noisy prefix distribution oracle with $\sigma^2 = K$.*

Proof. The first statement is trivial. For the second, consider any $\pi \in \mathcal{P}(\Sigma^n)$ and any prefix $s \in \Sigma^i$ with $i \in [n-1]$. Then letting $a \sim \mathcal{O}^{\pi}_{\mathrm{samp}}(s)$, we have $\mathbb{E}e_a = \pi_s$ by definition, and

$$\begin{aligned}\mathbb{E}_{a\sim\pi_s}\left[\chi^2\left(e_a\|\pi_s\right)\right] &= \mathbb{E}_{a\sim\pi_s}\left[\sum_{b\in\Sigma:[\pi_s]_b>0}\frac{(\mathbb{I}_{a=b}-\pi_s(b))^2}{\pi_s(b)}\right]\\ &= \mathbb{E}_{a\sim\pi_s}\left[\frac{(1-\pi_s(a))^2}{\pi_s(a)}+1-\pi_s(a)\right] = \mathbb{E}_{a\sim\pi_s}\left[\frac{1-\pi_s(a)}{\pi_s(a)}\right] \le K.\end{aligned}$$

□

Second, Definition 3 also captures a simple form of noisy logit access with linear noise. Since language models produce logits before applying the softmax, a natural abstraction of inference-time instability is as an additive perturbation to the logits. Lemma 5 shows that small (additive) Gaussian logit perturbations induce bounded relative variance at the distributional level.

Lemma 5. *Fix $\ell \in \mathbb{R}^m$, let $s(\xi) := \mathrm{softmax}(\ell+\xi) \in \mathcal{P}(\Sigma)$, and let $s_a(\xi) = \mathrm{softmax}(\ell+\xi)_a$, for all $a \in \Sigma$. Let $p := \mathbb{E}_{\xi}[s(\xi)]$, where $\xi \sim N(0, \gamma^2 I_m)$ is an isotropic Gaussian. If $\gamma^2 < \frac{1}{2}$, then*

$$V := \mathbb{E}_{\xi\sim N(0,\gamma^2 I_m)}\left[\sum_{a\in\Sigma}\frac{(s_a(\xi)-p_a)^2}{p_a}\right] \le \frac{2\gamma^2}{1-2\gamma^2}.$$

Proof. By the Gaussian Poincaré inequality (Corollary 2.17, [VH14]), for any smooth (infinitely-differentiable) function f on $\mathbb{R}^m$, if $\xi \sim N(0, \gamma^2 I_m)$,

$$\mathrm{Var}[f(\xi)] \leq \gamma^2 \mathbb{E}\|\nabla f(\xi)\|_2^2.$$

Applying this fact to the scalar function $f(\xi) = s_a(\xi)$ for each $a \in \Sigma$, and using the fact that $\nabla_\xi s_a(\xi) = s_a(e_a - s)$ with $\|e_a - s\|_2^2 \leq 2$, gives $\mathrm{Var}[s_a(\xi)] \leq 2\gamma^2 \mathbb{E}\left[s_a^2(\xi)\right]$. Thus,

$$V = \sum_{a \in \Sigma} \frac{\mathrm{Var}[s_a(\xi)]}{p(a)} \leq 2\gamma^2 \sum_{a \in \Sigma} \frac{\mathbb{E}\left[s_a^2(\xi)\right]}{p(a)} = 2\gamma^2(V+1).$$

We conclude the proof by rearranging the above inequality. □

C.2 Estimator construction

In this section, we develop a basic estimator for solving Problem 1 under the noisy access afforded by Definition 3. A naïve application of this estimator yields a suboptimal $\approx \frac{n^2\sigma^2}{\varepsilon^4}$ query complexity, which is improved by a variance reduction scheme in Section C.3.

To begin, we recall the mixture representation of the TV distance (6) from Lemma 1:

$$\begin{aligned} \mathrm{TV}(\pi, \mu) &= \mathbb{E}_{X \sim M}[Z(X)], \qquad M = \frac{\pi + \mu}{2}, \\ Z_{\pi,\mu}(x) &:= \begin{cases} \dfrac{|\pi(x) - \mu(x)|}{\pi(x) + \mu(x)}, & \pi(x) + \mu(x) > 0, \\ 0, & \pi(x) = \mu(x) = 0. \end{cases} \end{aligned} \tag{9}$$

When (π, μ) is clear from context, we write $Z(x)$ instead of $Z_{\pi,\mu}(x)$. Our strategy is to construct an empirical approximation $\widehat{Z}$ to Z and then bound the expected squared error $\mathbb{E}_{X \sim M}[(\widehat{Z}(X) - Z(X))^2]$.

As a first step, we explain how to construct an approximation to each of the quantities $\pi(x)$ and $\mu(x)$ required by Z. Fix some $r \in \mathbb{N}$ and prefix $s \in \Sigma^{<n}$. We define a random estimator

$$\widehat{p}_{s,r} := \frac{1}{r} \sum_{j=1}^{r} Y_{s,j}^{\pi} \in \mathcal{P}(\Sigma), \qquad Y_{s,j}^{\pi} \overset{\text{i.i.d.}}{\sim} \mathcal{O}_{\mathrm{ndist}}^{\pi}(s). \tag{10}$$

In other words, $\widehat{p}_{s,r}$ simply queries the noisy prefix distribution oracle $\mathcal{O}_{\mathrm{ndist}}^{\pi}(s)$ for r times in total, and averages the outputs. For any distribution $\pi \in \mathcal{P}(\Sigma^n)$ and any $r \in \mathbb{N}$, we can therefore define a corresponding empirical distribution $\widehat{\pi}_r$ as follows: for a string $x \in \Sigma^n$, we let

$$\widehat{\pi}_r(x) := \prod_{i=0}^{n-1} \widehat{p}_{x_{1:i},r}(x_{i+1}) \tag{11}$$

be a random estimator for $\pi(x)$, where each of the $\widehat{p}_{x_{1:i},r}(x_{i+1})$ is estimated via (10) using r independent queries to $\mathcal{O}_{\mathrm{ndist}}(x_{1:i})$. Thus, one evaluation of $\widehat{\pi}_r$ requires nr queries to $\mathcal{O}_{\mathrm{ndist}}$.

We first bound the expected χ^2 divergence between $\pi \in \mathcal{P}(\Sigma^n)$ and its empirical counterpart $\widehat{\pi}_r$, where expectations are over the randomness defining $\widehat{\pi}_r$ (e.g., in the oracle queries).

Lemma 6. *For any $\pi \in \mathcal{P}(\Sigma^n)$, let $\widehat{\pi}_r$ be defined in* (11)*, where for each prefix $s \in \Sigma^{<n}$, we generate an estimate $\widehat{p}_{s,r}$ as in* (10) *independently. Then if $\mathcal{O}_{\mathrm{ndist}}$ is a noisy prefix distribution oracle (Definition 3) with relative variance σ^2, there is a universal constant C such that for any $r \geq Cn\sigma^2$,*

$$\mathbb{E}\left[\chi^2(\widehat{\pi}_r \| \pi)\right] \leq \frac{Cn\sigma^2}{r}.$$

Proof. We observe that almost surely $\widehat{\pi}_r \ll \pi$, since any token with zero conditional probability under any $\pi(\cdot \mid s)$ receives zero mass under $\widehat{p}_{s,r}$. Hence

$$1 + \chi^2(\widehat{\pi}_r \| \pi) = \mathbb{E}_{X \sim \pi}\left[\left(\frac{\widehat{\pi}_r(X)}{\pi(X)}\right)^2\right],$$

where the expectation also includes the randomness of querying $\mathcal{O}_{\mathrm{ndist}}$. For any fixed prefix $s \in \Sigma^{<n}$,

$$\mathbb{E}_{a\sim\pi_s}\left[\left(\frac{\widehat{p}_{s,r}(a)}{\pi_s(a)}\right)^2\right] = \sum_{a:\pi_s(a)>0}\frac{\mathbb{E}[\widehat{p}_{s,r}(a)^2]}{\pi_s(a)} = \sum_{a:\pi_s(a)>0}\left(\pi_s(a) + \frac{\mathrm{Var}[\widehat{p}_{s,r}(a)]}{\pi_s(a)}\right) \le 1 + \frac{\sigma^2}{r}.$$

Applying this one-step estimate sequentially along the path $X_1, \ldots, X_n$, and using independence of the estimators $\widehat{p}_{s,r}$ across prefixes s, gives the claim:

$$\mathbb{E}_{X\sim\pi}\left[\left(\frac{\widehat{\pi}_r(X)}{\pi(X)}\right)^2\right] \le \left(1 + \frac{\sigma^2}{r}\right)^n \le 1 + \frac{Cn\sigma^2}{r}.$$

□

In implementing the estimator $\widehat{\pi}_r$ in (11), it is helpful to assume that in the background, a data structure has queried $\mathcal{O}_{\mathrm{ndist}}(s)$ for r times for *every prefix* $s \in \Sigma^{<n}$. When our algorithm requires an estimate $\widehat{\pi}_r(x)$, the data structure lazily provides the estimates $\widehat{p}_{s,r}$ for each $s = x_{1:i}$, and we use these estimates to compute $\widehat{\pi}_r(x)$. We similarly define an estimator $\widehat{\mu}_r$ for each $r \in \mathbb{N}$.

We are now ready to define our empirical approximation $\widehat{Z}$ to Z in (9). For fixed $r \in \mathbb{N}$, define $\widehat{\pi}_r$ and $\widehat{\mu}_r$ as in (11), which we have query access to. We then let

$$\widehat{Z}(x) := \frac{|\widehat{\pi}_r(x) - \widehat{\mu}_r(x)|}{\widehat{\pi}_r(x) + \widehat{\mu}_r(x)} \tag{12}$$

for any x where $\pi(x) + \mu(x) > 0$. If we observe $\widehat{\pi}_r(x) + \widehat{\mu}_r(x) = 0$, we set the corresponding $\widehat{Z}(x) = 0$.

In Lemma 7 and 8, we reduce bounding the squared error between $\widehat{Z}$ and Z to understanding the χ^2 divergences between $(\widehat{\pi}_r, \pi)$ and $(\widehat{\mu}_r, \mu)$.

Lemma 7. *For $u, v \ge 0$, let $\phi(u,v) := \frac{|u-v|}{u+v}, \forall u, v \ge 0, u+v > 0$, and $\phi(0,0) = 0$. Then for all $a, b, c, d \ge 0$ with $a + b > 0$,*

$$|\phi(c,d) - \phi(a,b)| \le 4\left(\frac{|c-a| + |d-b|}{a+b}\right).$$

Proof. Let $s := a + b$, $t := c + d$, and $\Delta := |c-a| + |d-b|$. If $t \ge \frac{s}{2}$, then

$$\begin{aligned}
|\phi(c,d) - \phi(a,b)| &\le \left|\frac{c-d}{t} - \frac{a-b}{s}\right| \\
&\le \frac{|(c-d)-(a-b)|}{t} + |a-b|\frac{|t-s|}{st} \\
&\le \frac{\Delta}{t} + \frac{\Delta}{t} \le \frac{4\Delta}{s}.
\end{aligned}$$

We conclude the proof by noting that when $t < \frac{s}{2}$, $\Delta > \frac{s}{2}$, while $|\phi(c,d) - \phi(a,b)| \le 1$. □

Lemma 8. *Let $\mathcal{O}^{\pi}_{\mathrm{ndist}}$, $\mathcal{O}^{\mu}_{\mathrm{ndist}}$ respectively be noisy prefix distribution oracles with relative variance σ^2 for $\pi, \mu \in \mathcal{P}(\Sigma^n)$. There is a universal constant $C > 0$ such that, for every $r \ge Cn\sigma^2$, defining M, Z as in* (9) *and $\widehat{Z}$ as in* (12)*,*

$$\mathbb{E}_{X\sim M}\left[\left(\widehat{Z}(X) - Z(X)\right)^2\right] \le \frac{Cn\sigma^2}{r}.$$

Proof. Observe that for $X \sim M$, we must have $\pi(X) + \mu(X) > 0$ by definition. Applying Lemma 7 with $(a,b,c,d) \leftarrow (\pi(X), \mu(X), \widehat{\pi}_r(X), \widehat{\mu}_r(X))$ shows that

$$|\widehat{Z}(X) - Z(X)| \le 4\left(\frac{|\widehat{\pi}_r(X) - \pi(X)| + |\widehat{\mu}_r(X) - \mu(X)|}{\pi(X) + \mu(X)}\right).$$

Therefore, using $(a+b)^2 \le 2a^2 + 2b^2$ and $M = \frac{\pi+\mu}{2}$,

$$\mathbb{E}_{X\sim M}\left[(\widehat{Z}(X) - Z(X))^2\right] \le 16 \sum_{x\in\Sigma^n} \frac{(\widehat{\pi}_r(x) - \pi(x))^2 + (\widehat{\mu}_r(x) - \mu(x))^2}{\pi(x) + \mu(x)}$$
$$\le 16\chi^2(\widehat{\pi}_r\|\pi) + 16\chi^2(\widehat{\mu}_r\|\mu).$$

Taking expectations over the randomness of $\widehat{\pi}_r$, $\widehat{\mu}_r$ and applying Lemma 6 proves the claim. □

Lemma 8 already implies a nontrivial algorithm for solving Problem 1. In particular, taking $r = O(\frac{n\sigma^2}{\varepsilon^2})$ in Lemma 8 implies that $\widehat{Z}$ has bias ε with respect to Z, and each evaluation of $\widehat{Z}$ requires $O(nr)$ queries. Combining this calculation with the $O(\frac{1}{\varepsilon^2})$ evaluations of $\widehat{Z}$ required to estimate its expectation to ε additive error gives a total query complexity of

$$O\left(\frac{n^2\sigma^2}{\varepsilon^2}\right) \cdot O\left(\frac{1}{\varepsilon^2}\right) = O\left(\frac{n^2\sigma^2}{\varepsilon^4}\right).$$

This bound already improves upon the $O(\frac{n^3}{\varepsilon^5})$ queries required by the prior work of [MKP25] in the regime $m = |\Sigma| = O(1)$. In Section C.3, we show how to use the squared error bound in Lemma 8 in a black-box way within a variance reduction scheme to improve this complexity bound.

Remark 2 (Asymmetric noise)**.** *Lemma 8 generalizes straightforwardly to settings where there is asymmetry in the relative variance bounds for $\mathcal{O}^{\pi}_{\mathrm{ndist}}$, $\mathcal{O}^{\mu}_{\mathrm{ndist}}$. In particular, if these bounds are respectively σ_π^2, σ_μ^2, and we instead use $r_\pi \in \mathbb{N}$ oracle queries at each prefix to define $\widehat{\pi}_{r_\pi}$ (similarly, $r_\mu \in \mathbb{N}$ queries to define $\widehat{\mu}_{r_\mu}$), it is simple to extend Lemma 8 to yield an expected squared error of*

$$\frac{Cn\sigma_\mu^2}{r_\mu} + \frac{Cn\sigma_\pi^2}{r_\pi}.$$

This asymmetry could be useful in saving samples when, e.g., we have exact query access to one of the autoregressive models (π, μ), but we have noisy or sample access to the other. To ease notational burden, we assume $\sigma_\mu = \sigma_\pi$ for simplicity henceforth.

C.3 Variance reduction through multilevel Monte Carlo

In this subsection, we improve upon a naïve application of Lemma 8 by estimating a telescoping sequence of increasingly-accurate approximations in a *multilevel Monte Carlo* (MLMC) variance reduction scheme. The analysis of our variance-reduced scheme uses Lemma 8 in a black-box way.

Fix an integer $L = O(\log \frac{1}{\varepsilon})$ in the following discussion, and for all levels $\ell = 0, 1, \ldots, L$, define

$$r_\ell := 1 + \left\lceil cn\sigma^2\, 2^\ell \right\rceil, \tag{13}$$

where c is a large constant and σ parameterizes the relative variance of $\mathcal{O}^{\pi}_{\mathrm{ndist}}$, $\mathcal{O}^{\mu}_{\mathrm{ndist}}$. The key to our variance reduction is to couple the external draw $X \sim M$ in adjacent levels, so that the correction from level $\ell - 1$ to level ℓ has variance proportional to its approximation error.

More concretely, for all $\ell = 0, 1, \ldots, L$, let Z_ℓ denote a random independent copy of the estimator $\widehat{Z}$ in (12), with $r \leftarrow r_\ell$ as set in (13). Also, define

$$\theta_\ell := \mathbb{E}[Z_\ell(X)], \qquad \theta := \mathrm{TV}\,(\pi, \mu) = \mathbb{E}_{X\sim M}\left[Z(X)\right],$$

where whenever we take expectations over Z_ℓ, we also include the randomness from the noisy queries to $\mathcal{O}_{\mathrm{ndist}}$. By Lemma 8, choosing a large enough c yields

$$\mathbb{E}\left[(Z_\ell(X) - Z(X))^2\right] \le C2^{-\ell}.$$

Therefore, by the Cauchy-Schwarz inequality,

$$|\theta_\ell - \theta| = |\mathbb{E}[Z_\ell(X) - Z(X)]| \le \sqrt{\mathbb{E}[(Z_\ell(X) - Z(X))^2]} \le C2^{-\frac{\ell}{2}}, \tag{14}$$

so that if $L = O(\log \frac{1}{\varepsilon})$ is large enough, $|\theta_L - \theta| \le \frac{\varepsilon}{2}$. Our goal is thus to estimate θ_L to $\frac{\varepsilon}{2}$ additive error. To do so, we write it as a telescoping sum:

$$\theta_L = \theta_0 + \sum_{\ell=1}^{L} (\theta_\ell - \theta_{\ell-1}). \tag{15}$$

Our goal is now to design estimators for each element $\theta_\ell - \theta_{\ell-1}$ within the sum (15). To do so, we couple the outer draw $X \sim M$ used to define these estimators. In particular, define $Z_{-1}(\cdot) \equiv 0$ for simplicity, and for each $\ell = 0, 1, \ldots, L$, define an estimator

$$Y_\ell := Z_\ell(X) - Z_{\ell-1}(X), \text{ where } X \sim M. \tag{16}$$

It is clear that this construction yields

$$\mathbb{E}[Y_0] = \theta_0, \qquad \mathbb{E}[Y_\ell] = \theta_\ell - \theta_{\ell-1} \text{ for } \ell = 0, 1, \ldots, L.$$

We now bound the variance of each estimator Y_ℓ.

Lemma 9. *For every $\ell = 0, \ldots, L$, we have $\mathrm{Var}[Y_\ell] \le \min(4, C2^{-\ell})$ for a universal constant $C > 0$. Moreover, we can obtain an i.i.d. copy of Y_ℓ using $O(n + n^2\sigma^2 2^\ell)$ queries to $\mathcal{O}^{\pi}_{\mathrm{ndist}}$ and $\mathcal{O}^{\mu}_{\mathrm{ndist}}$.*

Proof. The bound $\mathrm{Var}[Y_\ell] \le 4$ is immediate because each $Z_\ell(X)$ lies in $[0, 1]$ by definition, so each Y_ℓ lies in $[-1, 1]$. For the remaining bound, letting $\ell \ge 1$,

$$Y_\ell = Z_\ell(X) - Z_{\ell-1}(X) = (Z_\ell(X) - Z(X)) + (Z(X) - Z_{\ell-1}(X)), \tag{17}$$

where $X \sim M$. Thus, applying $(a+b)^2 \le 2a^2 + 2b^2$,

$$\mathrm{Var}[Y_\ell] \le \mathbb{E}[Y_\ell^2] \le 2\mathbb{E}[(Z_\ell(X) - Z(X))^2] + 2\mathbb{E}[(Z_{\ell-1}(X) - Z(X))^2] \le C2^{-\ell},$$

where the last step follows from adjusting the constant C in Lemma 8. The query complexity is immediate from the definition of $\widehat{Z}$ and our sample bound r_ℓ in (13). □

The proof of Lemma 9 reveals why it is important that the same $X \sim M$ is used to define Z_ℓ and $Z_{\ell-1}$ in the coupled increment Y_ℓ. Indeed, if different samples were used, then the identity (17) would not hold and we would not inherit the separate squared error bounds from Lemma 8.

Lemma 9 shows that the coupled increments Y_ℓ have geometrically-decaying variance across levels. This suggests a non-uniform sampling strategy, using more i.i.d. draws of Y_ℓ on coarse levels (small ℓ), where the variance is larger, and fewer samples on fine levels (large ℓ), where each sample is more expensive. By balancing the per-level variance contribution against the per-level sampling cost, we obtain an MLMC estimator that reduces the total query complexity.

We now explain how to allocate samples across levels. Let

$$N_\ell := \left\lceil A \cdot \frac{L+1}{\varepsilon^2} \cdot 2^{-\ell} \right\rceil, \qquad \ell = 0, 1, \ldots, L, \tag{18}$$

where $A > 0$ is a universal constant. Let $Y_{\ell,1}, \ldots, Y_{\ell,N_\ell}$ be independent copies of Y_ℓ, and define

$$T := \frac{1}{N_0} \sum_{j=1}^{N_0} Y_{0,j} + \sum_{\ell=1}^{L} \frac{1}{N_\ell} \sum_{j=1}^{N_\ell} Y_{\ell,j}. \tag{19}$$

Our final estimator simply takes the median of $\log(\frac{1}{\delta})$ copies of T defined in (19). We provide pseudocode for the overall method in Algorithms 2 and 3, and analyze its correctness in Theorem 6.

Algorithm 2 CoupledIncrement$(\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}}, \ell, \sigma, n)$

$X \sim M = \frac{1}{2}(\pi + \mu)$ ▷ Uses n queries to $\mathcal{O}^{\nu}_{\mathrm{ndist}}$ for $\nu \sim_{\text{unif.}} \{\mu, \pi\}$, as $\mathcal{O}^{\nu}_{\mathrm{ndist}}$ simulates $\mathcal{O}^{\nu}_{\mathrm{samp}}$
$Z_\ell(X) \leftarrow$ copy of $\widehat{Z}$ in (12) with $r \leftarrow r_\ell$ in (13) ▷ Uses nr_ℓ queries to $(\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}})$
if $\ell = 0$ **then**
    $Z_{\ell-1}(X) \leftarrow 0$
else
    $Z_{\ell-1}(X) \leftarrow$ copy of $\widehat{Z}$ in (12) with $r \leftarrow r_{\ell-1}$ in (13) ▷ Uses $nr_{\ell-1}$ queries to $(\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}})$
end if
return $Z_\ell(X) - Z_{\ell-1}(X)$

We are now ready to prove Theorem 3, stated slightly more formally here:

Algorithm 3 NoisyDistAccessTV($\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}}, \varepsilon, \delta, \sigma, n$)

$R \leftarrow \lceil 10 \log \frac{1}{\delta} \rceil$
$L \leftarrow \lceil C + 2 \log_2 \frac{1}{\varepsilon} \rceil$
for $s = 1, \ldots, R$ **do**
  $T_s \leftarrow 0$ ▷ Running sum for s^{th} evaluation of T in (19)
  for $\ell = 0, \ldots, L$ **do**
    $S_\ell \leftarrow 0$
    for $j = 1, \ldots, N_\ell$ **do**
      $Y_{\ell,j} \leftarrow$ CoupledIncrement($\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}}, \ell, \sigma, n$)
      $S_\ell \leftarrow S_\ell + Y_{\ell,j}$
    end for
    $T_s \leftarrow T_s + \frac{S_\ell}{N_\ell}$ where N_ℓ is as in (18)
  end for
end for
return $\mathrm{median}\{T_1, \ldots, T_R\}$

Theorem 6. *If $(\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}})$ are respectively noisy prefix distribution oracles with relative variance σ^2 for (π, μ), Algorithm 3 solves Problem 1 with $O(\frac{n+n^2\sigma^2}{\varepsilon^2} \log^2(\frac{1}{\varepsilon}) \log(\frac{1}{\delta}))$ queries to $(\mathcal{O}^{\pi}_{\mathrm{ndist}}, \mathcal{O}^{\mu}_{\mathrm{ndist}})$.*

Proof. We follow the notation in Algorithm 3. Observe that for any $s = 1, \ldots, R$, we have $\mathbb{E}[T_s] = \theta_L$ by using the calculation (15). Thus, for a large enough constant in the choice of L, (14) implies $|\mathbb{E}[T_s] - \theta| \le \frac{\varepsilon}{2}$. Moreover, by independence across calls to CoupledIncrement, we have by applying the variance bound in Lemma 9 that

$$\mathrm{Var}[T_s] = \sum_{\ell=0}^{L} \frac{\mathrm{Var}[Y_\ell]}{N_\ell} \le \frac{4}{N_0} + \sum_{\ell=1}^{L} \frac{C 2^{-\ell}}{N_\ell}.$$

Using N_ℓ as defined in (18), each summand is at most $O(\frac{\varepsilon^2}{A(L+1)})$. Hence, by choosing A sufficiently large, we obtain that $\mathrm{Var}[T_s] \le \frac{\varepsilon^2}{12}$. Chebyshev's inequality then gives the single-estimate bound

$$\mathbb{P}\left[|T_s - \mathrm{TV}(\pi, \mu)| > \varepsilon\right] \le \mathbb{P}\left[|T_s - \theta_L| > \frac{\varepsilon}{2}\right] \le \frac{4\mathrm{Var}[T_s]}{\varepsilon^2} \le \frac{1}{3}.$$

Finally, a standard application of Hoeffding's inequality shows that the median of R independent copies of an estimator T obeying the above bound lies within ε of $\theta = \mathrm{TV}(\pi, \mu)$ with probability at least $1 - \delta$. The query complexity follows from Lemma 9 with our definition (18):

$$\sum_{\ell=0}^{L} R N_\ell \cdot O(n + n^2\sigma^2 2^\ell) = O\left(\frac{(n + n^2\sigma^2)(L+1)^2 \log \frac{1}{\delta}}{\varepsilon^2}\right) + O\left((n + n^2\sigma^2) 2^L \log \frac{1}{\delta}\right),$$

where we conclude by using that $2^L = O(\frac{1}{\varepsilon^2})$ by our choice of L. □

As a corollary of Theorem 6 and the oracle reduction in Lemma 4, we obtain a query complexity bound for Problem 1 under prefix sampling oracle access.

Corollary 1. *Algorithm 3 solves Problem 1 with $O(\frac{n^2 K}{\varepsilon^2} \log^2(\frac{1}{\varepsilon}) \log \frac{1}{\delta})$ queries to $(\mathcal{O}^{\pi}_{\mathrm{samp}}, \mathcal{O}^{\mu}_{\mathrm{samp}})$.*

D Experimental details

This appendix gives the reproducibility setup, an exact-logit sanity check on the synthetic instance, and the methodology behind two quantitative claims of Section 4.2: the support-mismatch decomposition of Figure 2(a) and the variance-ratio diagnostic $\rho_1 = 0.06$ used to select the multilevel schedule.

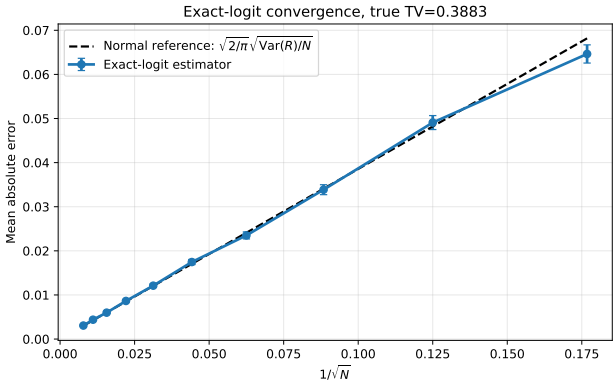


Figure 4: Mean absolute error of Algorithm 1 on the synthetic hard instance at $\sigma = 0$ (exact logits), against $1/\sqrt{N}$, with $\mathrm{TV}(\pi, \mu) = 0.388$ known by enumeration. Dashed line is the Gaussian-asymptotic reference $\sqrt{2/\pi}\sqrt{\mathrm{Var}(R)/N}$. The fitted slope is -0.493 vs the theoretical -0.500.

D.1 Setup and instance parameters

Synthetic hard instance. We use the binary autoregressive lower-bound construction of Appendix B.2: each sequence factors as $x = u \oplus z$ with $u \in \Sigma^r$ a block index and $z \in \Sigma^{n-r}$ a within-block leaf, and within each active block the discrepancy between π and μ is concentrated at the final bit. Throughout the synthetic experiments we fix $n = 128$, $r = 12$, $p_{\mathrm{active}} = 0.4$, $\alpha = 0.49$, giving $\mathrm{TV}(\pi, \mu) = 0.388$ by direct enumeration. The noisy oracle of Definition 3 is realized by adding zero-mean Gaussian noise of standard deviation σ to each realized next-token probability before taking logarithms; at $\sigma = 0$ this reduces to the exact prefix logit oracle of Definition 1.

Cross-engine sampling protocol. We use `vllm 0.19.1` with `enforce_eager=True` and prefix caching disabled, and `sglang 0.5.10.post1` with CUDA graph capture disabled, both running `Qwen3-0.6B` at bf16, top-$k = 20$, temperature 1, with EOS suppression for fixed-length sampling. The shared input is the system prompt `"You are a helpful assistant."` followed by the user prompt `"Tell me a story about a robot learning to paint."`, which the Qwen3 chat template tokenizes to a 30-token prefix. For each side we sample $N = 500$ length-$n = 500$ continuations and score each prefix $r = 16$ times per engine; the π-side uses base seed 0 and the μ-side base seed 1, with each repeat using an independent per-call seed offset. Sampling and scoring of one engine occur inside a single engine session to avoid kernel-reload byte-identical artifacts on the r-axis.

Hardware and software. All experiments were run on a single NVIDIA GH200 Grace Hopper node with 96 GB HBM3e memory on a shared SLURM-managed cluster. The toolchain is `gcc 14.2.0` / `cuda 12.8` / `python 3.11.8`. The two engines use isolated virtual environments: vllm pairs with `torch 2.10.0+cu128` and `transformers 5.7.0`; sglang pairs with `torch 2.9.1+cu128` and `transformers 5.3.0`. Synthetic experiments run on a single CPU using NumPy with master seed 10.

D.2 Exact-logit sanity check

At $\sigma = 0$, Algorithm 1 reduces to a standard outer-trajectory Monte Carlo on the bounded estimator $R = (1 - \exp(L_\mu - L_\pi))_+$, with $\mathbb{E}[R] = \mathrm{TV}(\pi, \mu)$ by Lemma 1. Figure 4 plots the empirical mean absolute error against $1/\sqrt{N}$ and overlays the Gaussian-asymptotic reference $\sqrt{2/\pi}\sqrt{\mathrm{Var}(R)/N}$. The fitted slope is -0.493 against the theoretical -0.500, and the empirical curve sits slightly below the reference because $R \in [0, 1]$ has shorter-than-Gaussian tails. This confirms that the synthetic instance, the true-TV computation, and the estimator are mutually consistent before any noise is added.

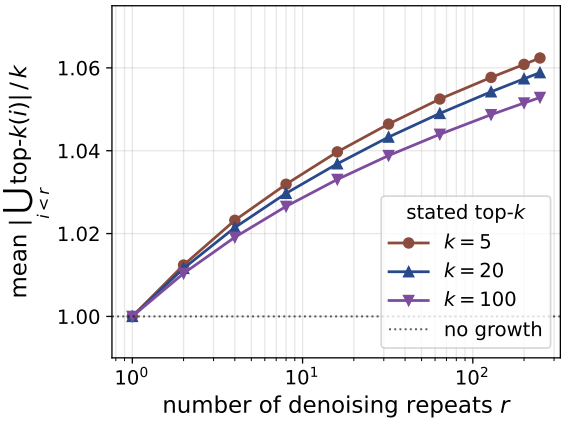


Figure 5: Mean empirical top-k support union size $|\bigcup_{i<r} \text{top-}k(i)|$ relative to the stated k, as a function of the number of repeated queries r at a fixed prefix on `Qwen3-0.6B`. Even at $r = 248$ the union exceeds the stated k by at most $\approx 7\%$, well below $2k$.

D.3 Decomposition of the cross-engine estimator

The decomposition in Figure 2(a) splits the estimator output at each truncated length n' into two parts. The *support-mismatch contribution* at n' is the fraction of trajectories that contain at least one such cell within their first n' positions, where a *support-mismatch cell* is one in which, after r-averaging, one engine's empirical top-k assigns positive mass to the sampled token while the other does not. By construction, each such trajectory contributes the bounded value 1 from (12): the formula $|\widehat{\pi}_r(x) - \widehat{\mu}_r(x)|/(\widehat{\pi}_r(x) + \widehat{\mu}_r(x))$ evaluates to 1 whenever exactly one of $\widehat{\pi}_r, \widehat{\mu}_r$ is zero on the trajectory. The *shared-support log-probability-difference contribution* is the residual after subtracting the support-mismatch term, computed directly on cells where both engines emit positive r-averaged probability and accumulating $\log \widehat{\pi}_r(x_t \mid x_{<t}) - \log \widehat{\mu}_r(x_t \mid x_{<t})$ across t. The support-mismatch term grows from 0% at $n' = 10$ to 35% at $n' = 500$; the shared-support term saturates near $n' \approx 200$. The two cross over at $n' \approx 300$, and at $n' = 500$ the support-mismatch term contributes 77% of the total.

D.4 Variance-ratio diagnostic on real data

Computing $\rho_\ell = \text{Var}[Y_\ell]/\text{Var}[Z_\ell]$ from real-engine traces requires the r replicate calls at a fixed prefix to be exchangeable. Both serving stacks group queries into kernel-internal batches, so consecutive calls in the orchestrator's call order can share scheduling-induced state and exhibit short-range correlations along the r axis. Naively splitting the r-axis into level-ℓ and level-$(\ell - 1)$ halves under such correlations can underestimate $\text{Var}[Z_\ell]$ and bias ρ_ℓ downward. We therefore evaluate ρ_ℓ after applying an independent uniform permutation of the r-axis at each $(\text{trajectory}, \text{position})$ cell. Under this iid-r baseline, with levels $r_0 = 4$, $r_1 = 16$ on the cross-engine cache we measure $\rho_1 = 0.06$, which places the instance in the regime where Figure 3 shows 2-level MLMC strictly improves over the single-level estimator at $r_1 = 16$. The matched-budget RMSE comparison in Figure 2(b) is computed on the same per-cell-permuted cache.

D.5 Empirical top-k support size

The query complexity bounds in the introduction are stated in terms of the support size K of the next-token distribution. When the API exposes a stated top-k parameter, the relevant K is the union of per-call top-k sets across the r repeated queries used by Algorithm 3 at a fixed prefix; nondeterminism in which tokens reach the threshold can make this union strictly larger than k. Figure 5 measures this union size on `Qwen3-0.6B` at the prefix of Appendix D.1, sweeping the stated top-$k \in \{5, 20, 100\}$ over r up to 248. Across all three settings the mean union remains within 7% of the stated k, so the practical K is comparable to the API-exposed value.

E Comparison between TV and KL

The comparison between TV and KL has a long history in probability, statistics, and information theory [GS02, SV16, Can22]. Here we sketch the aspects most relevant to our setting. We first recall several standard comparisons between total variation distance and KL divergence. A classical relation is Pinsker's inequality,

$$\mathrm{TV}\left(P,Q\right) \leq \sqrt{\frac{1}{2}\mathrm{KL}\left(P\|Q\right)},$$

which shows that small KL divergence implies small TV distance. However, this relation is only one-sided and can be loose in several regimes. For example, two distributions may be arbitrarily close in TV while having infinite KL divergence if their supports do not match. Concretely, considering $P_\varepsilon = \mathrm{Bern}(\varepsilon), Q = \mathrm{Bern}(0)$, we have

$$\mathrm{TV}\left(P_\varepsilon,Q\right) = \varepsilon, \qquad \mathrm{KL}\left(P_\varepsilon\|Q\right) = +\infty.$$

This illustrates that KL is highly sensitive to support mismatch, whereas TV only charges the amount of probability mass on which the two distributions differ. Conversely, when two distributions are already very far apart, TV saturates at its maximum value 1, while KL, being unbounded, can still reflect different degrees of separation.

The two quantities also differ in their operational meanings. TV admits the optimal coupling characterization

$$\mathrm{TV}\left(P,Q\right) = \inf_{\gamma\in\Gamma(P,Q)} \mathbb{P}_{(X,Y)\sim\gamma}[X \neq Y],$$

where $\Gamma(P,Q)$ denotes the set of all couplings of P and Q. Equivalently, there exists a maximal coupling such that

$$\mathbb{P}[X = Y] = 1 - \mathrm{TV}\left(P,Q\right).$$

Thus, a statement such as $\mathrm{TV}\left(P,Q\right) = 0.01$ has a clear sample-level meaning: the two distributions can be coupled so that their samples are exactly identical with probability 0.99. This makes TV particularly natural for measuring observable output-level discrepancy between two generative distributions.

TV also has a direct decision-theoretic interpretation. Consider the binary Bayesian testing problem where one first samples a hidden label $H \in \{0,1\}$ uniformly at random, and then observes $X \sim P$ if $H = 0$ and $X \sim Q$ if $H = 1$. The minimum possible probability of error over all decision rules is

$$P^\star_{\mathrm{err}} = \frac{1 - \mathrm{TV}\left(P,Q\right)}{2}.$$

Thus, TV directly measures the distinguishability of two distributions. In other words, if two APIs are treated as black-box generators and we want to detect whether their output distributions differ, TV gives the fundamental statistical limit of distinguishability. A small TV implies that no black-box detector can reliably tell the two APIs apart from a single output sample, whereas a large TV implies that the two APIs are distinguishable with small Bayesian error.

Compared with TV, KL divergence and its variants are more widely used in empirical machine learning. For example, the cross-entropy loss used to train language models satisfies

$$\mathbb{E}_{X\sim P}[-\log Q_\theta(X)] = H(P) + \mathrm{KL}\left(P\|Q_\theta\right),$$

and perplexity is an exponential transformation of this average log-loss. Thus, standard language-model training can be viewed as minimizing a KL-type objective.

This popularity is partly due to two practical advantages. First, KL admits an exact chain rule under autoregressive factorization:

$$\mathrm{KL}\left(P\|Q\right) = \sum_{t=1}^{n} \mathbb{E}_{X_{<t}\sim P}\left[\mathrm{KL}\left(P_t(\cdot \mid X_{<t})\|Q_t(\cdot \mid X_{<t})\right)\right].$$

Thus, a sequence-level KL decomposes into token-level terms that are directly accessible from prefix probabilities. Second, KL-based objectives are smooth and naturally compatible with gradient-based training. In contrast, TV is bounded and operationally meaningful, but it lacks an analogous chain rule and is less convenient as a direct differentiable loss.

F Declaration of LLM Usage

The initial idea for the variance reduction argument in Appendix C.3 was suggested by the GPT-5.5-pro model. All formal statements, proofs, and technical arguments were developed, verified, and reviewed by the authors, who take sole responsibility for their correctness.

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: Our main contributions are giving query complexity bounds for estimating the total variation (TV) distance between two length-n autoregressive distributions to additive error ε, under three access models defined in Section 2.2. All claims and statements are proved in the paper.

   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: We mentioned in our paper that our current bound is not tight under sampler access, which is an interesting open question.

   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: We provide complete proofs for all theorems and lemmas in Appendix B and C.

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: We describe the experiment setting in detail in Section 4 and provide code implementation in the supplementary files.

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 code implementation is in the supplementary files.

   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: Please refer to Section 4.

   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: [Yes]

   Justification: Please refer to Section 4.

   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: Please refer to Appendix D.

   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: We confirm that the paper conforms to 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: [N/A]

    Justification: The authors do not believe there is a potential societal impact of the work performed that is worth highlighting, as it is primarily introducing a general tool for assessing LLM distance.

    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: [Yes]

    Justification: We provide the license information in our README of the code base.

    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: [N/A]

    Justification: The paper does not release new assets.

    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: [Yes]

Justification: Please refer to Appendix F.

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.