# Flipping Against All Odds: Reducing LLM Coin Flip Bias via Verbalized Rejection Sampling

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Large language models (LLMs) can often accurately describe probability distributions using natural language, yet they still struggle to generate faithful samples from them. This mismatch limits their use in tasks requiring reliable stochasticity, such as Monte Carlo methods, agent-based simulations, and randomized decision-making. We investigate this gap between knowledge and sampling in the context of Bernoulli distributions. We introduce Verbalized Rejection Sampling (VRS), a natural-language adaptation of classical rejection sampling that prompts the LLM to reason about and accept or reject proposed samples. Despite relying on the same Bernoulli mechanism internally, VRS substantially reduces sampling bias across models. We provide theoretical analysis showing that, under mild assumptions, VRS improves over direct sampling, with gains attributable to both the algorithm and prompt design. More broadly, our results show how classical probabilistic tools can be verbalized and embedded into LLM workflows to improve reliability, without requiring access to model internals or heavy prompt engineering.

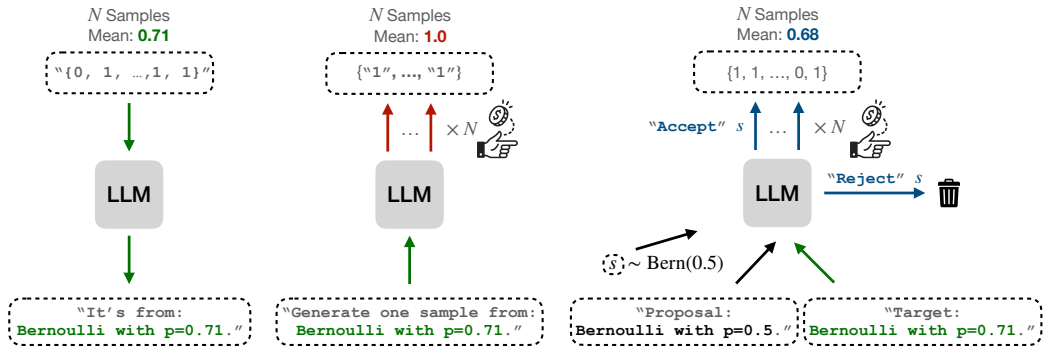
## 1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities in generating coherent text and even performing reasoning tasks. An emerging question is whether LLMs can understand and reproduce probabilistic processes when prompted in natural language. In particular, if we ask an LLM to behave like a random sampler for a known distribution (e.g., produce coin flip outcomes with a given probability), will it faithfully do so? Reliable sampling underpins Monte Carlo algorithms [13, 19], probabilistic programming [4], agent-based simulations [11, 3], and randomized decision making [16, 15]; yet, despite randomness being central to modern computation, the extent to which contemporary LLMs can generate faithful i.i.d. samples remains largely unexplored.

Recent work has begun to study LLMs not just as next-word predictors but as generators of random outcomes drawn from specified distributions. Empirical evidence shows that, while LLMs can infer probability distributions [6] and do Bayesian updates to approximately infer a coin's bias when given data [7], their own samples from a distribution remain biased [11]. Figure 1(a;b) illustrate this gap for Bernoulli distributions. Hence, LLMs know what a fair coin is, but they struggle to behave like one.

This mismatch poses concrete risks from a user's perspective. A user who sees an LLM accurately reasoning about a distribution might trust it to sample from that distribution; hidden bias can then contaminate downstream workflows, skew survey simulators, or introduce unfairness in stochastic tie-breakers. If an LLM cannot flip a fair coin, could it be trusted to sample from more complex distributions? This raises safety, reliability, and fairness concerns across the stack.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.




(a) Infer the distribution of the samples (b) Direct Sampling (c) Verbalized Rejection Sampling

Figure 1: Illustrations of the knowledge-sampling gap and two different sampling methods.

In the setting of Bernoulli distributions, we present a comprehensive study of correcting LLM sampling bias via a language-adapted rejection-sampling framework, and uncover surprising interactions between prompt design and algorithmic guarantees. Our contributions include:

- **Sampling Faithfulness Study (Section 4).** We measure how faithfully LLMs generate i.i.d. Bernoulli samples when prompted directly. Across four models, sampling bias varies significantly with the phrasing of the distribution. Chain-of-thought only helps in some cases. We also quantify the gap between a model's ability to identify a distribution and its ability to simulate it.
- **Verbalized Rejection Sampling (VRS) (Section 5).** We adapt the classical rejection sampling method through natural language into LLMs. VRS is model-agnostic (for both open-source and proprietary LLMs), requires no access to the model weights, and keeps the LLM in a black-box. Given a fixed prompt with textual descriptions of the target and proposal distributions alongside a candidate sample, the LLM is instructed to perform the accept/reject step. Our empirical study shows a significant reduction of the bias for the samples.
- **Empirical and Theoretical Insights (Section 6).** Effectively, VRS draws a Bernoulli random variable to decide whether to accept a proposed sample. Counter-intuitively, this indirection produces less sampling bias than prompting the model to output a sample directly. We analyze this phenomenon theoretically, proving—under mild assumptions—that VRS can generate samples with less bias than direct sampling and separating the gains attributable to the prompt phrasing from those guaranteed by the algorithm itself.

Beyond correcting the specific failure mode of Bernoulli sampling, our study opens a broader path towards integrating principled randomness into LLM-based systems. Faithful Bernoulli generation is a basic requirement for reliable LLM-driven simulations and stochastic reasoning. Our results show that a lightweight, theoretically sound wrapper—without model access or hyper-parameter tuning—substantially narrows the knowledge-sampling gap. More broadly, our work illustrates how classical statistical tools can be verbalized and paired with LLMs to deliver reliability without resorting to opaque prompt engineering.

## 2 Related Work

**Sampling and flipping coins with LLMs.** Recent work shows that LLMs often exhibit a gap between knowing and sampling from a distribution. [6] find that LLMs can describe the target probabilities, yet when asked to "roll a die" or "flip a coin" their outputs exhibit large bias. They show that incorporating code generation with Python tool use can alleviate the problem. In contrast, we focus on improving sampling within the natural language space, leveraging LLMs' inherent probabilistic reasoning capabilities. While one could bypass the model to obtain true samples from a target distribution, enabling LLMs to faithfully perform such tasks themselves is both practically useful and scientifically insightful. [14] explore how LLMs "flip a fair coin" and "flip 20 fair coins". They find that current LLMs not only replicate human biases but often amplify them. [7] probe the online learning setting of Bernoulli distribution from a Bayesian inference angle. They show that with sufficient in-context examples, LLMs update their estimate of a coin's bias roughly following Bayes' rule. Unlike their focus on online learning and belief updating, we do not assume sequential

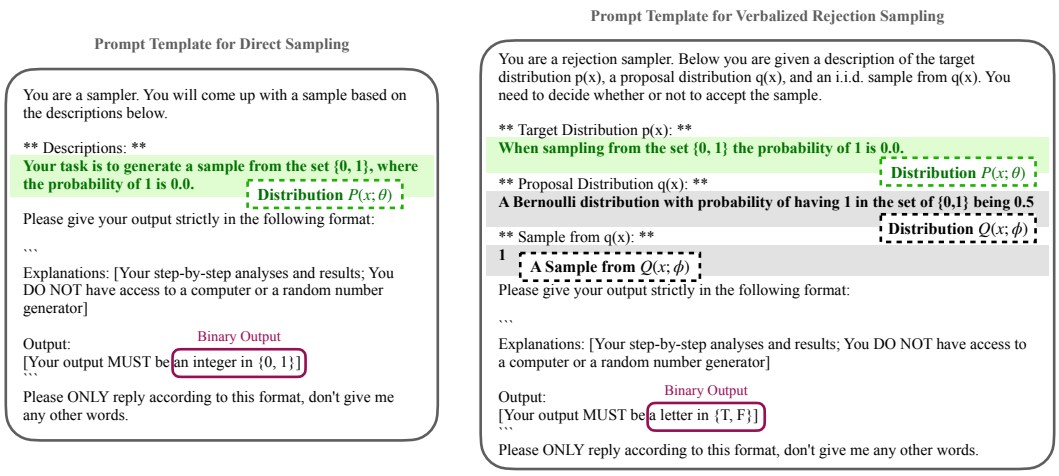



Figure 2: Prompt templates for direct sampling and Verbalized Rejection Sampling.

access to data and instead concentrate on the generation of i.i.d. samples from a fixed Bernoulli distribution. [11] and [8] find similar gaps in settings beyond Bernoulli (e.g., poll simulation, categorical distribution), showing that LLMs can summarize distributions but fail to sample from them reliably, echoing the Bernoulli findings on a higher-dimensional setup. Together, these studies reveal a recurring pattern: LLMs know the right distributions but struggle to sample from them faithfully. Our work aims to reduce this mismatch by adapting the rejection sampling algorithm to LLMs, leveraging their internal probabilistic behavior to guide natural language based sampling.

**Natural language and text based parameterization.** Recent work explores using natural language to parameterize models, treating LLMs as inference engines that interpret and evaluate these descriptions. This makes model specification more accessible and interpretable. [12] introduce LLM Processes, where LLMs generate predictive distributions conditioned on natural language inputs and in-context data. Their method operates in an in-context, non-parametric style and requires access to token logits. In contrast, we treat language as a parametric description of a fixed distribution, without past data or logit access. [18] propose Verbalized Machine Learning (VML), where prompts act as natural language parameters for deterministic functions. Our work instead focuses on probabilistic distributions and faithful sampling. [2] presents a theoretical framework demonstrating that a finite set of function compositions, analogous to a vocabulary, can approximate any continuous mapping, drawing parallels between linguistic compositionality and function approximation. These studies underscore the potential of natural language as a medium for specifying probabilistic models. In our work, we focus on the Bernoulli distribution as a fundamental case study, demonstrating how LLMs can be guided to generate faithful samples from a simple yet foundational probabilistic model.

# 3 Problem Setup

Our investigation focuses on the ability of LLMs to generate faithful i.i.d. samples from distributions described purely in natural language. Focusing on Bernoulli distributions, defined by a single numerical parameter $p \in [0, 1]$, we treat LLMs as samplers accessed solely through text interaction.

## 3.1 Parameterizing Distributions in Natural Language

In our setting, the distribution is parameterized by a textual prompt. Formally, we denote this natural language parameterized distribution as $P(x;\theta)$, where $\theta$ captures both the underlying numerical parameter $p$ and the linguistic phrasing of the prompt. Figure 2(left) shows an example, where

$P(x;\theta)$ = *"Your task is to generate a sample from the set {0, 1}, where the probability of 1 is 0.0."*.

For the same $p$, different phrasings may lead to different sampling behaviors. We test several ways of phrasing a Bernoulli distribution, and write $P(x;p)$ for a fix phrasing. For each phrasing, we test 101 values of $p \in \{0.0, 0.01, 0.02, \ldots, 1.0\}$. For each $p$, we query the LLM 100 times independently with the same prompt, and extract the binary output (i.e., '0' or '1') to form the resulting i.i.d. samples.

### 3.2 LLMs as Black-Box Samplers

We treat LLMs as black-box samplers, accessed solely via APIs. The only controllable input is the prompt; the only observable output is text. For open-source models, we use vLLM [9], but we assume no access to internals such as weights, activations, or token-level logits. This contrasts with prior work [7, 8, 12] that uses output token logits to estimate sampling probabilities.

This API-only setup allows consistent evaluation across both open-source and proprietary models, reflecting realistic usage where internals are inaccessible. It also better supports techniques like chain-of-thought (CoT; [17]) prompting, which can distort token-level probabilities by conditioning on generated reasoning: with CoT, logits reflect $p(x \mid \text{reasoning for } x)$ instead of the intended $p(x)$. We also fix all decoding hyperparameters (e.g., temperature, top-k) to their default values given in the API, since most real world users do not adjust them, and often do not have the ability to do so.

## 4 How Reliable is Direct Sampling?

This section examines the reliability of direct sampling from LLMs. We first compare their ability to generate samples to their ability to recognize distributions, then explore how prompt phrasing affects sampling bias, and finally test whether chain-of-thought reasoning improves sample quality.

### 4.1 Measuring the Knowledge-Sampling Gap

To assess the gap between an LLM's understanding of a Bernoulli distribution and its ability to sample from it, we compare its evaluative and generative performance in a controlled setup, using Llama-3.1-70B-Instruct [5]. We first test the model's ability to identify the correct Bernoulli distribution from data. For 11 equally spaced probabilities $p_0, ..., p_{10}$, s.t. $p_i \in [0, 1]$, we generate 100 i.i.d. samples using Python, forming datasets $S_i$. For each pair $(i, j)$, we prompt the LLM to decide whether $S_i$ was drawn from $\mathrm{Bern}(p_j)$, producing an $11 \times 11$ response matrix. Diagonal entries should be "Yes", off-diagonals "No". We repeat this process five times and report average accuracies in Figure 3(a). We then test the model's sampling behavior by prompting it to generate 100 samples for each $p_i$, using the template in Figure 2(left). The resulting sets $\hat{S}_i$ are evaluated using the same method as before. The average accuracies over five runs are reported in in Figure 3(b).

The left panel shows high off-diagonal accuracy for Python generated data (i.e., confidently rejecting incorrect hypotheses), with minor errors along the diagonal due to natural sample variation (e.g., 48 ones out of 100 for $p = 0.5$ may lead to confusion with $p = 0.48$, hence, rejecting the correct hypotheses). In contrast, the right panel shows major degradation for LLM-generated samples. Diagonal accuracy drops significantly for all $p_i$, except the edge cases when $p = 0.0$ and $p = 1.0$. Moreover, we observe an asymmetry in the off-diagonal entries: the lower triangle of the matrix exhibits much worse accuracy than the upper triangle. This indicates that samples from $p_i$ are often misclassified as having come from $p_j$ with $j > i$, suggesting that the LLM-generated samples are consistently biased toward ones. These results reveal a clear knowledge–sampling gap: LLMs can evaluate distributions well but fail to sample from them faithfully. Unlike question answering, where each input has a correct target, i.i.d. sampling lacks per-instance ground truth, making it a fundamentally different and underexplored capability.

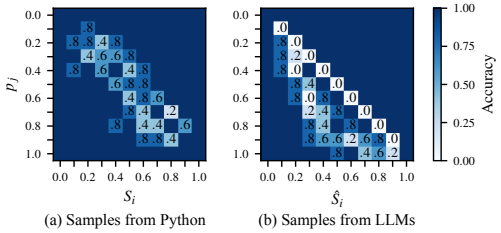
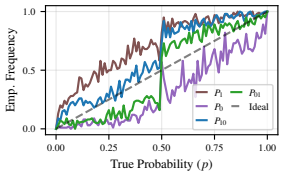


Figure 3: Recognition accuracy matrix.

### 4.2 How Much Can Prompt Phrasing Reduce Sampling Bias?

The previous section used a single fixed phrasing to describe the Bernoulli distribution (see Figure 2, left). Yet, natural language allows many equivalent ways to express the same distribution, raising the question: how much can phrasing affect sampling bias? In the prior setup, the prompt emphasized the probability of generating a 1, denoted $P_1(x; p)$, as illustrated in Figure 4(b). Notably, this formulation focuses solely on the probability of generating a 1, which may partly explain the tendency of the model to produce more 1s than 0s in the in the sampled outputs.




(a) Calibration Plot

$P_1(x; p)$ = "Your task is to generate a sample from the set {0, 1}, where the probability of 1 is $p$."

$P_0(x; p)$ = "Your task is to generate a sample from the set {0, 1}, where the probability of 0 is $1 - p$."

$P_{10}(x; p)$ = "Your task is to generate a sample from the set {0, 1}, where the probability of 1 is $p$, and the probability of 0 is $1 - p$."

$P_{01}(x; p)$ = "Your task is to generate a sample from the set {0, 1}, where the probability of 0 is $1 - p$, and the probability of 1 is $p$."

(b) Four different phrasings for a Bernoulli with $p$

Figure 4: Calibration plots for direct sampling and the four different phrasings.

To explore this, we test three alternative phrasings that shift or balance the focus across outcomes, as shown in Figure 4(b). For each, we sample across a range of $p$ values using Llama-3.1 and plot the empirical frequency of 1s against the ground truth, yielding calibration curves shown in Figure 4(a). The calibration curves show that the balanced descriptions, i.e., those stating both probabilities, yield samples that are better calibrated. Nevertheless, all four phrasings result in noticeable bias. This result resonates with [1], where they found that when prompting humans to imagine a coin flip, mentioning only 'heads' or mentioning only 'tails' will lead to a similar sampling bias.

**Quantitative comparison using Sum of TV Distance (STVD).** To quantify the calibration performance of different phrasings, we compute the area between each calibration curve and the ideal diagonal reference line. Specifically, for each $p_i$, we calculate the absolute difference between the empirical sampling frequency $\tilde{p}_i$ and the true value $p_i$, and sum these over all 101 values, i.e., $\text{STVD} = \sum_{i=0}^{100} |\tilde{p}_i - p_i|$. Since this absolute difference corresponds to the total variation (TV) distance between two Bernoulli distributions, we refer to the resulting metric as the Sum of TV Distances (STVD) where smaller is better. See Appendix A.1 for more details about the TV distance.

Table 1 presents the STVD values for the four phrasings under direct sampling. For Llama 3.1, the best-performing phrasing $P_{01}$ achieves an STVD of 11.08, nearly half that of the baseline $P_1$, which scores 21.80. We also include results for other LLMs, including GPT-4.1-nano, DeepSeekV3 [10], and Qwen-2.5 72B [20]. Interestingly, the best-performing phrasing varies across models, as highlighted by the underlined entries. The calibration plots for the other models can be found in Appendix B.1.

These findings suggest that while prompt design can influence sampling bias, relying solely on prompt engineering to eliminate bias can be difficult and inconsistent across model family, and additional mechanisms are likely needed for more systematic approaches to correct sampling bias.

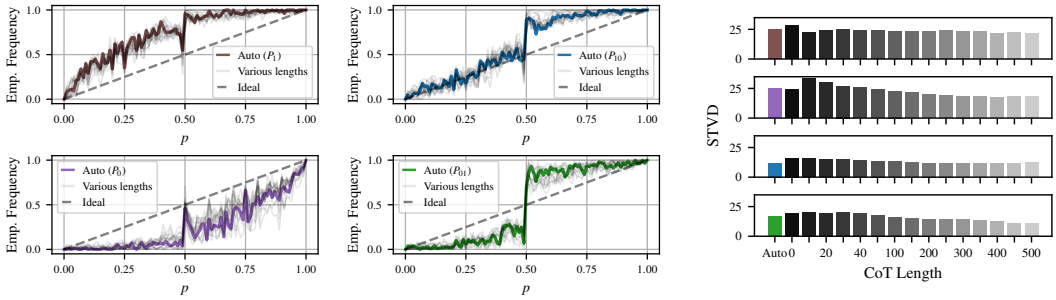
Table 1: Quantitative comparison between Direct Sampling and VRS in STVD (↓).

| Method | Llama-3.1 70B | | | | | GPT-4.1-nano | | | | | DeepSeekV3 | | | | | Qwen-2.5 72B | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_1$ | $P_0$ | $P_{10}$ | $P_{01}$ | mean | $P_1$ | $P_0$ | $P_{10}$ | $P_{01}$ | mean | $P_1$ | $P_0$ | $P_{10}$ | $P_{01}$ | mean | $P_1$ | $P_0$ | $P_{10}$ | $P_{01}$ | mean |
| Direct | 21.80 | 17.95 | 12.10 | 11.08 | 15.73 | 17.87 | 30.23 | 16.63 | 19.24 | 21.00 | 17.76 | 19.39 | 20.78 | 23.26 | 20.30 | 20.73 | 18.72 | 19.00 | 22.64 | 20.27 |
| VRS | 5.91 | 7.63 | 5.52 | 5.56 | **6.20** | 12.96 | 13.06 | 9.50 | 8.46 | **11.00** | 5.34 | 9.06 | 5.29 | 6.94 | **6.66** | 5.93 | 6.35 | 4.49 | 5.12 | **5.47** |

### 4.3 Does Chain-of-Thought (CoT) Help Sampling?

Since phrasing alone does not eliminate sampling bias, we explore whether modifying the instruction for the output can help. Prior work [14, 7, 8, 12] often asks LLMs to output the sample immediately, enabling access to token logits for estimating predictive distributions. However, this approach is constrained to open-source models and treats LLMs more as likelihood models than samplers. In our setting, we only use LLMs for sampling and do not require access to logits or early output. This allows us to apply CoT [17] prompting, where the model first generates reasoning before giving its final answer. While sampling differs from question answering, CoT may increase output variability by encouraging diverse reasoning paths, potentially reducing bias.

To test this, we instruct the model to produce reasoning of varying lengths $N$ (ranging from 0 to 500 words) before answering, along to an 'Auto' setting where no length constraint is imposed. The 'Auto' is the default setting for experiments in previous sections, which uses the template in Figure 2(left). For different $N$, we modify the '*Explanations*' instruction in the prompt template to include a sentence saying that '*Your analysis must have around $N$ words*'.




(a) Calibration Plots

(b) STVD For Various CoT Length Constraints

Figure 5: Calibration plots and STVD trend for various reasoning length constraints.

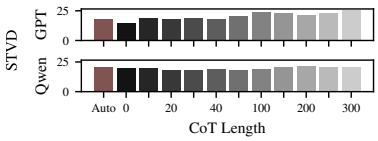



Figure 6: STVD vs CoT Length.

Figure 5 presents the calibration plots (left) and STVD scores (right) for Llama-3.1 under different CoT length constraints. Overall, reasoning length has limited effect on bias, though longer CoT slightly improves calibration. Direct output without reasoning often performs worse than the 'Auto' setting. However, this pattern does not hold across models. As shown in Figure 6, GPT-4.1 and Qwen2.5 show no consistent improvement with longer CoT; in some cases, STVD increases as reasoning length grows. These mixed results suggest that, unlike in question answering, CoT is not a reliable method for reducing sampling bias, and its effect is model-dependent. For consistency, we use 'Auto' in all remaining experiments.

# 5 Verbalized Rejection Sampling

In the previous section, we explored ways to reduce sampling bias through prompt phrasing and instruction design. While these strategies do influence the behavior of LLMs, the results suggest that prompt-only interventions are insufficient for reliably eliminating bias. If direct sampling cannot be fully corrected through language alone, we may instead embrace the bias and mitigate it using algorithmic techniques. In probabilistic methods, several algorithms exist to transform biased proposals into unbiased samples. One such method is rejection sampling, which generates candidate samples from a proposal distribution and selectively accepts them to match a desired target distribution. In the remainder of this section, we adapt rejection sampling to operate entirely within the language interface of LLMs, and we refer to this method as verbalized rejection sampling (VRS).

## 5.1 Rejection Sampling

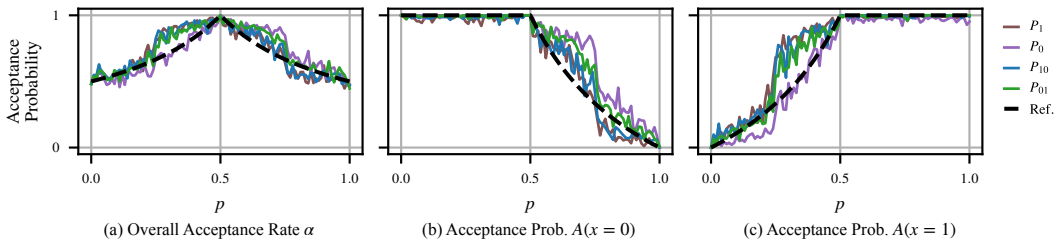
Rejection sampling is a sampling technique to generate samples from a target distribution $P$ while only having access to samples from a (typically simpler) proposal distribution $Q$. We assume that both $P$ and $Q$ can be evaluated (but only $Q$ can be directly sampled from). The general idea is that we can generate a sample from $P$ by instead sampling from $Q$ and accepting the sample with probability $P(x)/(MQ(x))$ where $M < \infty$ is a bound on the ratio $P(x)/Q(x)$. We assume that both $P$ and $Q$ are Bernoulli distributions with parameters $p$ and $q$. In this case, we can compute $M$ analytically as: $M = \max\{p/q, (1-p)/(1-q)\}$. Let $A(x)$ denote the acceptance probability of $x \sim Q$ which is

$$A(x) = \begin{cases} \frac{P(x)}{MQ(x)} = \frac{p}{Mq} & \text{if } x = 1 \\ \frac{P(x)}{MQ(x)} = \frac{1-p}{M(1-q)} & \text{if } x = 0 \end{cases}. \tag{1}$$

The accept/reject step effectively draws a sample from $\mathrm{Bern}(A(x))$. The overall acceptance rate is $\alpha = \sum_{x\in\{0,1\}} Q(x)A(x) = 1/M$. See Appendix A.2 for more details about rejection sampling.

## 5.2 Adapting Rejection Sampling to LLMs

Figure 1(c) illustrates the overall idea behind VRS. Classical rejection sampling requires three inputs: the target distribution $P$, the proposal distribution $Q$, and a sample $x \sim Q$. The algorithm evaluates




(a) Overall Acceptance Rate $\alpha$ (b) Acceptance Prob. $A(x = 0)$ (c) Acceptance Prob. $A(x = 1)$

Figure 8: Empirical acceptance rates for VRS.

whether to accept or reject $x$ based on these inputs, returning a binary decision. To implement this in the LLM setting, we design a prompt template (Figure 2, right) that verbalizes all three components, i.e., descriptions of $P, Q$, and the proposed sample $x$, as natural language. These are inserted into fixed slots in the template. The model is instructed to reason through its decision and then output a single letter from $\{\mathrm{T}, \mathrm{F}\}$, indicating whether to accept (T) or reject (F) the sample. We send the completed prompt to the LLM and parse its response. If the response indicates acceptance, we retain the sample; otherwise, we generate a new proposed sample and repeat the process. This loop continues until we collect the required number of accepted samples.

### 5.3 Experiments

We evaluate VRS on four different LLMs: Llama-3.1, GPT-4.1-nano, DeepSeekV3, and Qwen-2.5. For each model, we run VRS until it accepts 100 samples for each of the 101 values of $p \in [0.0, 1.0]$, following the same setup as in the direct sampling experiments. As the proposal distribution $Q$, we fix it to a uniform Bernoulli with $q = 0.5$ across all values of $p$. The resulting calibration plot for Llama-3.1 is shown in Figure 7, and the corresponding STVD scores across all models are included in Table 1. The calibration plots for other three LLMs can be found in Appendix B.2.

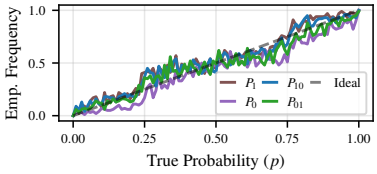



Figure 7: Calibration plot for VRS

Comparing the calibration plot for VRS (Figure 7) with that of direct sampling (Figure 4a), we observe a significant reduction in sampling bias. Across all four prompt phrasings, the calibration curves under VRS closely align with the ideal diagonal reference, indicating much improved fidelity to the target Bernoulli distributions. Figure 8 shows the corresponding empirical acceptance probabilities, which seem to align well with the analytical targets. The improvement is also reflected quantitatively in Table 1: the STVD scores for VRS are substantially lower than those for direct sampling, with most cases showing a reduction of over 50%. In some instances, STVD drops to nearly 25% of the original value. Crucially, this improvement holds across all four LLMs tested (i.e., Llama-3.1, GPT-4.1-nano, DeepSeekV3, and Qwen-2.5), demonstrating that VRS consistently mitigates bias and does so independently of the underlying model.

## 6 Why Does Verbalized Rejection Sampling Work?

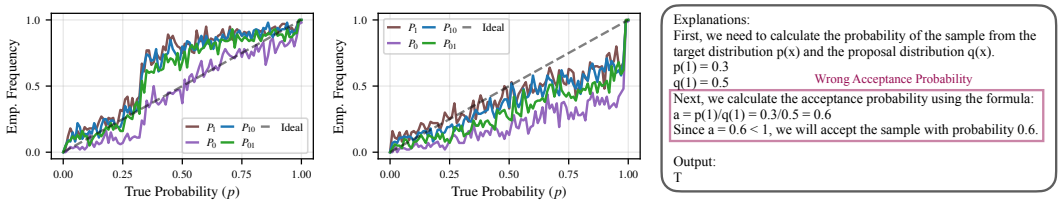
The effectiveness of VRS in reducing sampling bias is surprising at first glance since, internally, VRS still relies on the LLM to perform a Bernoulli trial, i.e., deciding whether to accept or reject a sample, which is precisely the type of stochastic behavior we have shown LLMs to struggle with.

*If LLMs are biased in direct sampling, why does wrapping the decision in rejection sampling help?*

Is the improved calibration a result of the specific prompt design used in VRS? Or does the rejection sampling algorithm itself introduce structural guarantees that correct bias, even when implemented via a biased LLM? The remainder of this section explores these possibilities empirically and theoretically.

### 6.1 Is It the Magic in the Prompt?

To investigate whether VRS's improvement stems purely from prompt design, we remove external randomness by fixing the proposed sample to a constant, i.e., $x = 1$. In this case, a faithful LLM




(a) Calibration for VRS-simple (b) Calibration for VRS-simple-M (c) Example Output for VRS-simple

Figure 9: Calibration plots for two ablations and an example LLMs output for VRS-simple.

should accept with probability $A(1)$, as defined in Equation (1). We compare this with the empirical acceptance probability $\tilde{A}(1)$, estimated from the LLM's responses. Figure 8(c) shows $\tilde{A}(1)$ for various $p$, using a fixed proposal $Q = \text{Bern}(0.5)$. For the trivial case $p > 0.5$, the alignment is strong. For $p < 0.5$, the results appear reasonable overall but show a consistent bias, particularly in the range $p \in [0.2, 0.5]$. To compare more directly with direct sampling, we evaluate $\tilde{A}(1)$ over 101 equally spaced values of $A(1)$, using the inverse of Equation (1) to recover the corresponding $p$. For each, we generate a VRS prompt with the computed $p$, a fixed $Q = \text{Bern}(0.5)$, and a fixed sample $x = 1$. We refer to this setup with fixed proposal and no introduced randomness as *VRS-simple*. If prompt design alone explains the improvement, VRS-simple should outperform direct sampling in calibration.

Figure 9(a) shows the calibration plot for VRS-simple using Llama-3.1. Compared to direct sampling (Figure 4a), the results are slightly more calibrated. Table 2 confirms this, with the mean STVD dropping from 15.73 to 11.85. This suggests the VRS prompt helps reduce bias for direct sampling. However, VRS-simple relies on explicitly computing the inverse of Equation (1) to tailor the prompt to each target $p$, and the improvement remains modest compared to full VRS.

Table 2: Ablation STVD (↓)

| Method | $P_1$ | $P_0$ | $P_{10}$ | $P_{01}$ | mean |
|---|---|---|---|---|---|
| Direct | 21.80 | 17.95 | 12.10 | 11.08 | 15.73 |
| VRS | 5.91 | 7.63 | 5.52 | 5.56 | 6.20 |
| VRS-simple | 15.83 | 6.55 | 13.97 | 11.05 | 11.85 |
| VRS-simple-M | 11.43 | 29.08 | 13.45 | 19.52 | 18.37 |
| VRS-M | 4.97 | 11.74 | 5.91 | 7.03 | 7.41 |

**Magic or Mirage?** To further understand why the VRS prompt improves sampling, we examine whether its structure encourages the model to reason differently. One hypothesis is that phrasing the sampling task in the context of rejection sampling prompts the LLM to internally compute acceptance probabilities, potentially disrupting its default biases learned during pretraining. To test this, we manually analyzed the model's reasoning outputs from VRS-simple (see Figure 9(c)). We found that, while the model often tries to derive the acceptance probability, it frequently does so incorrectly. In the non-trivial cases where $A(x) \neq 1$, the model tends to compute only the ratio $P(x)/Q(x)$, omitting the constant $M$ in the denominator.

*Could this incorrect derivation be the reason behind the improvement?* To test that, we designed variants of VRS-simple and VRS where we explicitly instruct the model to compute and use $M$ correctly. We refer to these as VRS-simple-M and VRS-M, respectively. The calibration plot for VRS-simple-M is shown in Figure 9(b), with corresponding STVD scores in Table 2. Through output inspection, we verified that the LLM now correctly computes the constant $M$ in its reasoning. However, this correction leads to worse performance: the mean STVD increases to 18.37, higher than in direct sampling. For the full VRS setup, adding the $M$-instruction also results in a slight degradation, with STVD rising from 6.20 to 7.41, though still outperforming direct sampling.

These results suggest that the improvement from the VRS prompt is not due to accurate computation of the acceptance probability. Instead, the prompt seems to help in an unexpected way, but it alone cannot explain the full benefit. The remaining gains likely come from the rejection sampling mechanism itself, rather than prompt phrasing alone.

### 6.2 Is the Improvement from the Algorithm?

Prompt design alone cannot fully explain the gains from VRS. To analyze the role of the algorithm itself, we model the LLM as a biased Bernoulli sampler. In VRS, this means the acceptance decision is not sampled from the true probability $A(x)$, but from a perturbed version $\tilde{A}(x) = A(x) + e(x)$, where $e(x)$ represents the model's bias. Based on this, we can derive the following proposition.

**Proposition 1** (Informal, see Proposition 1 in Appendix A.3.)**.** *Let $P(x;p), Q(x;q)$ be Bernoulli distributions (target and proposal, respectively). Let $\tilde{P}$ denote the resulting distribution after rejection sampling, and assume a bound on the model's bias $|e(x)| \leq c \in \mathbb{R}$. Then, with $M$ defined in Section 5.1,*

$$\mathrm{TV}(\tilde{P}, P) \leq \frac{Mc}{1 - Mc}. \tag{2}$$

From empirical observations, particularly in Figure 8(b;c), we note that the LLM appears well calibrated when $A(x) = 1$. Therefore, we can further derive the following result.

**Proposition 2** (Informal, see Proposition 2 in Appendix A.4.)**.** *Following Proposition 1, with the additional assumption that $e(x) = 0$ if $A(x) = 1$, i.e., we have $\tilde{A}(x) = A(x) + e(x)$ if $A(x) < 1$, and $\tilde{A}(x) = A(x)$ if $A(x) = 1$. Then, with $\hat{x}$ being chosen such that $A(\hat{x}) < 1$,*

$$\mathrm{TV}(\tilde{P}, P) \leq \frac{Q(\hat{x})Mc}{(1 - Q(\hat{x})Mc)}. \tag{3}$$

This gives a bound for the TV distance between the resulting distribution from VRS ($\tilde{P}$) and the ideal target ($P$). Now, we want to see when VRS is better than direct sampling. We can denote the resulting distribution from direct sampling as $\bar{P}$, and assume it has the same sampling bias $e(x)$. Then, VRS is better than direct sampling if $\mathrm{TV}(\tilde{P}, P) < \mathrm{TV}(\bar{P}, P)$. We can derive the following.

**Corollary 1** (Informal, see Corollary 1 in Appendix A.5.)**.** *Following Proposition 2, and assuming that $\bar{P}$ has the same bias as $\tilde{A}(x)$, i.e., $\bar{P}(x) = P(x) + e(x)$. Then,*

$$\mathrm{TV}(\tilde{P}, P) < \mathrm{TV}(\bar{P}, P) \iff Q(\hat{x}) < \frac{1}{M(1+c)}. \tag{4}$$

Intuitively, the corollary says that if the proposal $Q$ puts little enough mass, i.e., less than $1/(M(1+c))$, on the state that sometimes gets rejected ($\hat{x}$), the error introduced inside the rejection step hurts less than applying the same error directly to every draw from $\bar{P}$. In our experiments we fix the proposal to $q = 0.5$. This allow us, for each target $p$, to compute the corresponding $M$ and derive the maximum allowable bias $c$ under which VRS still outperforms direct sampling. In Figure 10, we visualize this by shading the box defined by $\mathrm{clip}(p \pm c, 0.0, 1.0)$ on the top of the calibration plot Figure 4(a). The result shows that in most cases, the empirical frequencies from direct sampling fall well within this box, satisfying the theoretical condition. This provides strong evidence that the primary source of VRS's improvement comes from the rejection sampling algorithm itself, not just prompt effects.

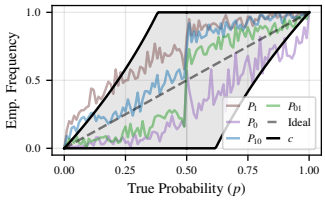


Figure 10: Calibration plots with error bounds $\pm c$ overlaid.

## 7 Conclusion and Limitation

We examined the ability of LLMs to sample from natural-language-described distributions, using Bernoulli as a test case. While LLMs can evaluate whether data matches a distribution, they struggle to generate unbiased samples, revealing a clear knowledge–sampling gap. This highlights that *sampling is a fundamentally distinct ability from question answering*: evaluation tasks have clear supervision, while i.i.d. sampling lacks per-instance ground truth and is only verifiable at the distribution level. We tested whether prompt phrasing or chain-of-thought reasoning could reduce bias. While both influence behavior, neither reliably closes the gap. To address this, we proposed Verbalized Rejection Sampling (VRS), a lightweight adaptation of classical rejection sampling expressed entirely in natural language. VRS improves calibration across models without accessing logits or tuning decoding parameters, and our analysis shows that the algorithm, not just prompt design, is key to its success. Beyond correcting this specific failure mode, our work points to a broader path: integrating principled randomness into LLM-based systems. Faithful Bernoulli sampling is a basic requirement for LLM-driven simulations and probabilistic reasoning. VRS illustrates how probabilistic tools can be verbalized and paired with LLMs to improve reliability—without resorting to opaque prompt engineering. This study is limited to the Bernoulli case; our theoretical results do not directly generalize to more complex distributions. Extending this framework to broader families remains an important direction for future work.

# References


[1] Maya Bar-Hillel, Eyal Peer, and Alessandro Acquisti. "heads or tails?"—a reachability bias in binary choice. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 40(6):1656, 2014. 5

[2] Yongqiang Cai. Vocabulary for universal approximation: A linguistic perspective of mapping compositions. *arXiv preprint arXiv:2305.12205*, 2023. 3

[3] Yong Cao, Haijiang Liu, Arnav Arora, Isabelle Augenstein, Paul Röttger, and Daniel Hershcovich. Specializing large language models to simulate survey response distributions for global populations. *arXiv preprint arXiv:2502.07068*, 2025. 1

[4] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76:1–32, 2017. 1

[5] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 4

[6] Jia Gu, Liang Pang, Huawei Shen, and Xueqi Cheng. Do llms play dice? exploring probability distribution sampling in large language models for behavioral simulation. *arXiv preprint arXiv:2404.09043*, 2024. 1, 2

[7] Ritwik Gupta, Rodolfo Corona, Jiaxin Ge, Eric Wang, Dan Klein, Trevor Darrell, and David M Chan. Enough coin flips can make llms act bayesian. *arXiv preprint arXiv:2503.04722*, 2025. 1, 2, 4, 5

[8] Aspen K Hopkins and Alex Renda. Can llms generate random numbers? evaluating llm sampling in controlled domains. Sampling and Optimization in Discrete Space (SODS) ICML 2023 Workshop, 2023. 3, 4, 5

[9] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023. 4, 26

[10] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024. 5

[11] Nicole Meister, Carlos Guestrin, and Tatsunori Hashimoto. Benchmarking distributional alignment of large language models. *arXiv preprint arXiv:2411.05403*, 2024. 1, 3

[12] James Requeima, John Bronskill, Dami Choi, Richard Turner, and David K Duvenaud. Llm processes: Numerical predictive distributions conditioned on natural language. *Advances in Neural Information Processing Systems*, 37:109609–109671, 2024. 3, 4, 5

[13] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999. 1

[14] Katherine Van Koevering and Jon Kleinberg. How random is random? evaluating the randomness and humaness of llms' coin flips. *arXiv preprint arXiv:2406.00092*, 2024. 2, 5

[15] Alicia Vidler and Toby Walsh. Evaluating binary decision biases in large language models: Implications for fair agent-based financial simulations. *arXiv preprint arXiv:2501.16356*, 2025. 1

[16] Abraham Wald. Statistical decision functions. *The Annals of Mathematical Statistics*, pages 165–205, 1949. 1

[17] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 4, 5

[18] Tim Z Xiao, Robert Bamler, Bernhard Schölkopf, and Weiyang Liu. Verbalized machine learning: Revisiting machine learning with language models. *arXiv preprint arXiv:2406.04344*, 2024. 3

[19] Yongjian Xu, Akash Nandi, and Evangelos Markopoulos. Application of large language models in stochastic sampling algorithms for predictive modeling of population behavior. *Artificial Intelligence and Social Computing*, 122:10–20, 2024. 1

[20] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. 5

# 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: All claims are listed in the abstract and introduction. The introduction includes a list of contributions.

   Guidelines:

   - The answer NA 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 NA 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 discuss limitations in Section 7.

   Guidelines:

   - The answer NA 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 an informal version of the theoretical statements in the main text which we rigorously prove in Appendix A (Appendix A.3, Appendix A.4, Appendix A.5).

Guidelines:

- The answer NA 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 provide all information necessary to reproduce the experiments in Section 3.2.

Guidelines:

- The answer NA 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: We include sufficient instructions in the main text to reproduce the experiment results. We will also release the code after.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify all details needed to understand the results in Section 3.2 and at the corresponding places in the experiments in Section 4 and in Section 5.

Guidelines:

- The answer NA 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: While error bars are not meaningful in our setup (due to the fact that we are sampling Bernoulli variables), we draw $100$ samples for every Bernoulli parameter $p$ that we investigate and ensure statistical significance in this way.

Guidelines:

- The answer NA 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: We host the open-source model ourselves on 4 Nvidia H100 GPUs. Generating 100 samples takes on average 25 seconds (see Appendix C).

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conform, in every aspect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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: [Yes]

    Justification: The scope of the research is very narrowly focusing on sampling from Bernoulli distributions utilizing LLMs, but we do mention the broader implication in the introduction, since faithful sampling in LLMs is fundamental for safety and fairness.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: There is no data or models released with this research.

    Guidelines:

    - The answer NA 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: For the open-source models, we use Llama-3.1 under the LLAMA 3.1 COMMUNITY LICENSE AGREEMENT, DeepSeekV3 under the DEEPSEEK LICENSE AGREEMENT, and Qwen-2.5 under the Qwen LICENSE AGREEMENT. We buy the service from OpenAI to use GPT-4.1-nano. We included this info in Appendix C.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA 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: [NA]

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

    Guidelines:

    - The answer NA 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: [NA]

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

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We only use LLMs for editing the paper. However, this paper is concerned with analyzing and explaining sampling from Bernoulli distributions with LLMs. The setup on how LLMs are used is described clearly in Section 3.

    Guidelines:

    - The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.