

Bag-of-Tokens Tuning: Calibrating LLM Logits with Contextual Probabilities

Chen-Meng Qiu¹, Ze-Hao Wang¹, Deng-Bao Wang^{1,2*}, Linan Yue^{1,2}, Min-Ling Zhang^{1,2}

¹ School of Computer Science and Engineering, Southeast University, Nanjing 210096, China

² Key Laboratory of Computer Network and Information Integration (Southeast University),
Ministry of Education

{qiucm, wangzehao, wangdb, lnyue, zhangml}@seu.edu.cn

Abstract

Token probabilities play a central role in large language models. However, these probabilities often fail to reflect either answer correctness or probability distributions explicitly specified in the context. We attribute this failure to the **Context-Logit Probability Gap**: contextual probabilities are not reliably reflected in the model’s next-token logits. We show that post-hoc temperature scaling has a limited effect on addressing this gap, because it rescales logits without correcting erroneous probability rankings. Motivated by this, we propose Bag-of-Tokens (BoT) tuning, a very simple fine-tuning approach that directly calibrates the logits with the probability distribution conveyed by synthetic contexts. More specifically, we construct BoT examples by forming a bag of candidate tokens with an explicitly assigned probability distribution. This distribution is expressed in the context and used as soft supervision to align the model’s logit probabilities via KL-divergence minimization. Experiments show that BoT tuning generalizes to LLM-based sampling tasks, such as random-number generation, and improves downstream QA accuracy and calibration. These results suggest that calibrating LLM logits with contextual probabilities provides a lightweight and effective way to enhance their uncertainty awareness.

1 Introduction

Large language models (LLMs) have become a cornerstone of modern artificial intelligence, offering impressive capabilities in natural language processing tasks (Guo et al., 2025). However, the reliability of LLMs depends in part on their ability to produce confidence scores that accurately reflect output correctness (Zhang and Wang, 2026). Such calibration, which aligns a model’s confidence with its accuracy, is essential in real-world deployments

where decisions based on incorrect outputs can have significant consequences.

Although normalized token logits are commonly used as confidence scores, recent studies show that they are poorly calibrated in LMs and do not reliably reflect output correctness (Tian et al., 2023). Meister et al. (2025) further show that model log-probabilities can be misleading and remain challenging in multiple domains. These findings suggest that probabilities derived from model logits may not faithfully reflect model confidence. Besides, recent studies investigate model logits from the perspective of LLM-as-a-sampler. Hopkins et al. (2023) point out that we cannot trust LLMs to produce a target distribution. Zheng et al. (2024) show that token bias is a primary intrinsic cause of selection bias, but that this kind of bias cannot be removed by simple prompt engineering. In addition, Lovering et al. (2025) show LM outputs are not calibrated to the numeric content of their contexts and that the model behavior has systematic biases and even strong LMs are poorly calibrated in simple settings.

These observations show that poorly calibrated token logits undermine both confidence expression and probability sampling. We trace this issue to the disconnect between contextual probabilities and token logits, a discrepancy we term the *Context-Logit Probability Gap*. We first verify that the gap exists, and larger models exhibit a lower gap. Further, temperature scaling does not adequately address this misalignment, as systematic biases like incorrect ranking cannot be corrected by a single global rescaling. The limitation of post hoc scaling motivates a training-based solution. However, existing fine-tuning methods for uncertainty calibration in language models typically rely on hard label supervision (Chandu et al., 2025; Guo et al., 2024). Such supervision provides only discrete signals, rather than a full probability distribution in the context, and therefore does not directly narrow

*Corresponding author

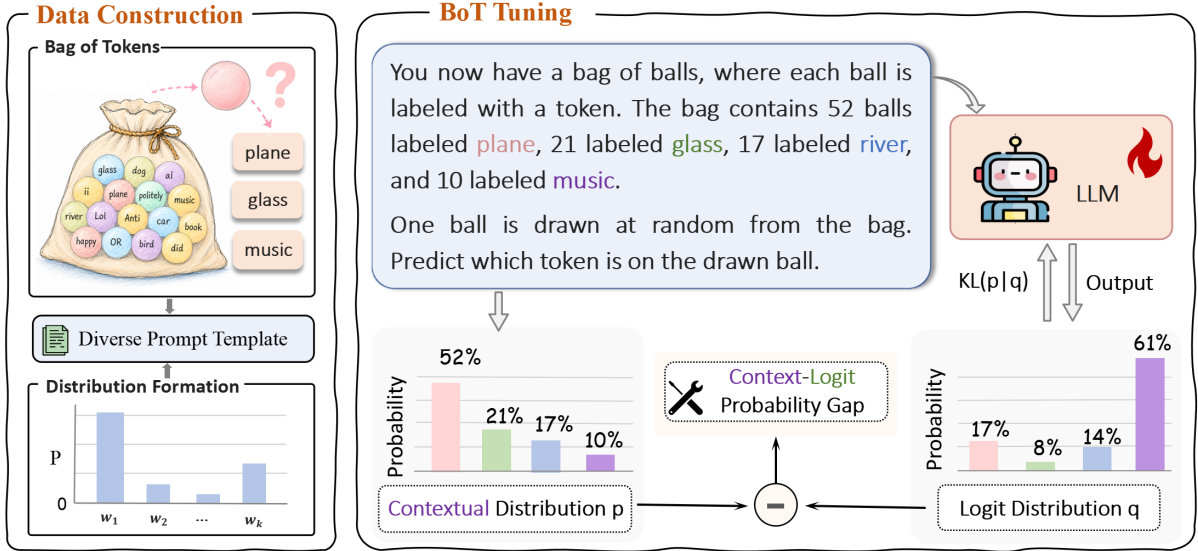


Figure 1: Overview of the Bag-of-Tokens (BoT) Tuning method. We construct a BoT dataset by combining tokens with distributions, convert these into soft-label training data, and fine-tune the model.

the context-logit gap. We therefore propose Bag-of-Tokens (BoT) tuning,¹ a simple fine-tuning approach that uses the contextual probabilities in synthetic data as soft supervision. More precisely, as illustrated in Figure 1, we first construct data by pairing tokens with probability distributions, creating contexts that explicitly specify the distribution. The tokens can be selected from frequent tokens in the tokenizer vocabulary and common words collected from HellaSwag. Then the BoT tuning minimizes the KL divergence between the predicted probability distribution and the contextual probability distribution. By directly aligning predicted probabilities with the contextual probabilities, our objective provides a more targeted way to improve token-level logit calibration.

Our method relies only on low-cost synthetic data and a simple fine-tuning procedure, yet it yields consistent gains across multiple evaluation dimensions. First, unlike standard supervision that emphasizes a single target token, this alignment task requires the model to reproduce an entire target distribution over candidate tokens, thereby directly improving its ability to sample in a probability-consistent manner. Specifically, the tuned model improves on sampling tasks where target tokens are drawn from both the training token set and a held-out test token set. The gains further transfer to more natural sampling tasks, where the LLM is used to generate samples following specified

binomial or uniform distributions. This suggests that our method does not merely memorize the constructed token distributions, but helps correct a more general failure mode in model sampling (Sec. 5.2). Furthermore, our method improves downstream QA accuracy and calibration; when combined with temperature scaling, BoT further reduces ECE (Sec. 5.3). Finally, we conduct ablation studies to isolate the effects of different components (Sec. 5.6). In summary, our contributions are: (1) We reveal the *Context-Logit Probability Gap*, showing that probability information specified in the context is not always faithfully reflected in model logits; (2) We propose Bag-of-Tokens tuning, a lightweight fine-tuning method that uses synthetic BoT data and soft-label supervision to calibrate logits with contextual probabilities. (3) We demonstrate consistent improvements on sampling and QA tasks across multiple benchmarks. Our method generalizes to held-out tokens and new sampling distributions, reduces ECE while improving QA accuracy.

2 Related Work

2.1 LLM Uncertainty Calibration

While calibration has been extensively studied in traditional classification models (Guo et al., 2017; Rudinger et al., 2018; Bai et al., 2022; Wang et al., 2021), token-level probability calibration in LLMs remains relatively underexplored. Most recent studies on LLM calibration focus on sentence-level correctness, semantic uncertainty, or verbalized

¹Our code is available at <https://github.com/qiuchenmeng/BoTTuning>.

confidence (Kuhn et al., 2023; Tian et al., 2023; Liu et al., 2024), rather than directly calibrating the token logits that govern decoding and probability-based decision making. Existing evaluation studies suggest that this issue is both prevalent and practically important. For example, Liang et al. (2023) evaluate language models across a wide range of classification tasks and find that they are often miscalibrated and overconfident. Similarly, Sorensen et al. (2024) argue that there remains limited research on explicit calibration techniques for aligning LLM confidence with actual performance. To address calibration errors in LLMs, prior work has explored post-hoc methods such as temperature scaling variants (Zheng et al., 2024; Xie et al., 2024). However, such methods mainly rescale the sharpness of the output distribution and are therefore insufficient for solving the token-level distributional alignment problem (Lovering et al., 2025; Meister et al., 2025). In other words, a single global rescaling cannot correct systematic errors.

2.2 Language Models Are Bad Samplers

Language models have strong logit bias and have been proven to be bad samplers (Pezeshkpour and Hruschka, 2024; Wei et al., 2024). Gupta et al. (2025) show that models have strong priors in probabilistic settings such as coin flips, and Meister et al. (2025) further show that models struggle to directly model distributions. Most existing work in this line shows that the model fails on simple fixed distributions (Van Koeveering and Kleinberg, 2024; Hopkins et al., 2023). Specifically, Lovering et al. (2025) show that language models have bias like pick first and pick higher even when in-context distributions are given. However, the research on improving the ability of LLMs to sample from specified probability distributions remains limited. While verbalized rejection sampling (Xiao et al., 2025) can improve sample capability in black-box settings, it is limited to the binomial distribution.

3 Key Observations

3.1 Experiment Setting

Our setup is as follows: Let $m \in M$ denote a probability task, and respondents from a fixed candidate set $w \in W$ have an in-context distribution, denoted by q . Given this task m , the model will use a distribution expression method to output a corresponding candidate distribution p_θ . We are interested in the distributional difference between

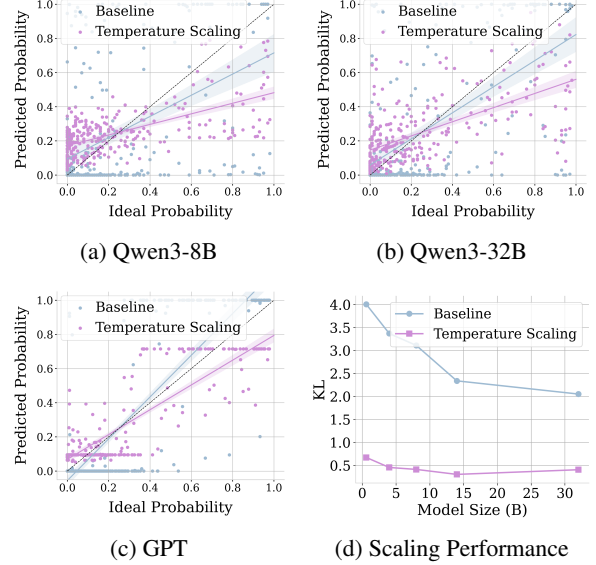


Figure 2: (a)-(c): Performance on different models to show the commonality of context-logit probability gap and the inability of temperature scaling. The x-axis represents the true probabilities in context, and the y-axis represents the predicted probabilities from model logits. (d): KL decreases with model size.

the reference distribution q in context and the model’s predicted distribution p_θ .

We adopt the Kullback–Leibler (KL) divergence to evaluate this distributional distance:

$$\mathcal{KL} = D_{\text{KL}}(q \parallel p_\theta) = \sum_{i=1}^K q^{(i)} \log \frac{q^{(i)}}{p_\theta^{(i)}}. \quad (1)$$

3.2 Token Logits Are Misaligned with the Contextual Probability Distribution

The context-logit probability gap is empirically examined using Qwen3-8B, Qwen3-32B, and GPT-4.1, the most recent GPT-series model that provides access to token-level log probabilities.² As shown in Figures 2a–2c, the predicted probabilities exhibit clear deviations from the contextual probabilities across both larger open-weight models and commercial frontier models accessed via APIs. While perfect calibration would place the points along the diagonal oracle line, the observed points are widely scattered, providing direct evidence of the context-logit probability gap. We next focus on the Qwen3 family to examine how this gap changes with model scale. As shown in Figure 2, the KL divergence between the contextual probabilities

²Unless otherwise specified, we use four candidates and 1K samples in all experiments, and report the mean results over three independent runs.

and the predicted probabilities generally decreases as model size increases. Nevertheless, the divergence remains clearly above zero even for the largest model, suggesting that scaling can calibrate predicted logit distribution but is insufficient to eliminate the gap.

3.3 Limitations of Temperature Scaling

We next examine whether temperature scaling can reduce the context-logit probability gap. Figure 2 shows that temperature scaling partially improves calibration, but the scatter points remain noticeably away from the oracle diagonal. More precisely, Figure 2d illustrates that the improvement also saturates for larger models, leaving a non-zero residual gap. The limitation of temperature scaling can be further understood from the scatter plots. The predicted probabilities are not consistently higher or lower than the true probabilities; instead, both over-estimation and under-estimation occur across different probability regions and candidate words. Temperature scaling primarily adjusts the sharpness of the predictive distribution through a global rescaling of the logits. However, because it **preserves the relative order of logits, it cannot correct miscalibration caused by incorrect probability rankings**. Therefore, although temperature scaling provides a useful baseline, the observed mixed misalignment patterns motivate a more flexible distributional calibration method.

4 Methodology

To mitigate the gap, we propose Bag-of-Tokens tuning, which leverages contextual probabilities to calibrate predicted logit probabilities. Specifically, we construct synthetic data by pairing candidate tokens with target probability distributions and expressing this mapping through natural language. The resulting synthetic datasets are then combined with QA samples for fine-tuning, encouraging logit calibration while preserving general performance.

4.1 Bag-of-Tokens Construction

Bag-of-Tokens data construction defines the candidate space, assigns a target distribution over this space, and expresses the resulting distribution in natural language. We describe how candidate tokens and target distributions are constructed as follows, while the prompt templates for combining candidate tokens with target distributions are provided in Appendix J.

Candidate Token Selection We first select a set of candidate tokens, and a naive choice is to sample candidates from the full model vocabulary. However, modern language models contain more than 30k tokens, many of which are rare, fragmented, or unnatural text units. Using the full vocabulary as candidate tokens would be computationally inefficient and could produce prompts that are difficult to interpret. Therefore, we construct a candidate pool of frequently used tokens by retaining only those with Zipf frequency scores greater than 6 (Brysbaert and New, 2009). Given a prompt x and a candidate set $\mathcal{W} = \{w_1, \dots, w_K\}$, the next-token probability of w_i is: $p_\theta(w_i | x)$.

To support a more general setting, we further consider candidates that may be tokenized into multiple tokens. We collect candidate words from the HellaSwag dataset (Zellers et al., 2019), which contains relatively long and diverse contexts. We retain words that occur more than 30 times and use them to construct the common word candidate pool. Given a prompt x , we consider a candidate word w_i that is tokenized into a sequence of tokens $(t_1, \dots, t_m)$. The probability of w_i is then computed autoregressively as

$$p_\theta(w_i | x) = \prod_{\ell=1}^m p_\theta(t_\ell | x, t_{<\ell}), \quad (2)$$

where $t_{<\ell} = (t_1, \dots, t_{\ell-1})$ denotes the previously generated tokens in the candidate sequence. We collect approximately 2k candidates and split them into training and test candidate sets with a 7:3 ratio.

Distribution Formation For each candidate set, we generate a target probability distribution with the goal of covering diverse contextual probability patterns. In our default construction, we first sample the maximum probability uniformly and assign it to a randomly selected candidate. Given K candidate tokens, we sample

$$m \sim \mathcal{U}\left(\frac{1}{K}, 1 - \epsilon\right), \quad p_j = m, \quad (3)$$

where j is randomly selected from the candidate set. The remaining probability mass $1 - m$ is then distributed among the other candidates while ensuring that no candidate receives a probability larger than m . By varying the maximum probability from close to $1/K$ to close to 1, this construction produces distributions ranging from near-uniform to highly concentrated ones.

As a natural alternative, we also consider Dirichlet sampling:

$$\mathbf{p} \sim \text{Dirichlet}(\boldsymbol{\alpha}), \quad (4)$$

where $\boldsymbol{\alpha}$ controls the concentration of the sampled distribution. Smaller concentration values tend to produce more peaked distributions, while larger values produce distributions closer to uniform.

4.2 Bag-of-Tokens Tuning

After constructing the BoT data, we propose **Bag-of-Tokens tuning**, a simple fine-tuning approach that trains the model on a hybrid dataset combining BoT data with general question-answering (QA) data. Specifically, we use the training sets of BoolQ (Clark et al., 2019), and RACE (Lai et al., 2017) as BoolQ contains naturally occurring yes/no questions derived from anonymized Google Search queries, reflecting realistic information-seeking scenarios. Besides, RACE covers diverse passages, including biographies, fiction, and argumentative essays, and requires fine-grained comprehension, multi-step reasoning, information synthesis, and analysis of authorial intent. Together, these datasets provide complementary QA supervision for maintaining downstream ability.

To optimize BoT data and QA data under a unified objective, we formulate both data types as distributional alignment tasks. For BoT examples, the candidates are tokens in a bag, and the target distribution is the contextual probability distribution specified in the prompt. For QA examples, the candidates are options, and the correct answer is represented as a one-hot target distribution.

Target Prediction Position To ensure that logits are evaluated at the correct prediction position, we adopt a prefix mechanism. In practice, a language model first generates a prefix, such as *chosen:*, *Draw ball, predict word:*, or *Randomly selected word:* to move the decoding context to the position where the sampled candidate should be generated. We then evaluate the candidate logits immediately after this prefix, so that the next-token distribution corresponds to the actual sampling position. This design ensures that all candidates are compared under the same context and that the predicted probabilities are measured at the intended position.

Given an input x and a candidate set $\mathcal{W}$, we denote the target distribution over candidates as $q(w | x)$ and the predicted distribution as $p_{\theta}(w | x)$. The training objective is defined as the KL

divergence between the target distribution and the predicted distribution:

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}(q(w | x) \| p_{\theta}(w | x)), \quad (5)$$

5 Experiment

In this section, we evaluate our method on sampling and QA tasks. We compare it with baselines from different domains across diverse benchmarks and conduct ablation studies to examine the contribution of key components in our method.

5.1 Experimental Setup

Models Our study focuses on the causal, decoder-only LLMs since this architecture has become the dominant choice for modern LLMs. We experiment with three LLM families: Qwen3-1.7B/8B/14B (Team, 2025), LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024), Mistral-7B-Instruct-v0.3 (Jiang et al., 2023). The models are all open source on HuggingFace, and we can access their output probabilities.

Benchmarks To directly evaluate whether the model can act as a reliable sampler, we evaluate it on five sampling benchmarks. ID TOKEN uses the same task formulation and candidate pool as training, but uses different target distributions. OOD TOKEN keeps the same task formulation but replaces the candidate words with different token sets, testing whether the learned calibration generalizes beyond the training candidates. We further include three task-level transfer settings. MARBLE COLOR (Lovering et al., 2025) asks the model to sample a marble color according to the color counts specified in the context. BINOMIAL DISTRIBUTION (Xiao et al., 2025) asks the model to generate integer samples whose empirical distribution should match a specified binomial distribution. UNIFORM DISTRIBUTION (Hopkins et al., 2023) asks the model to sample integers uniformly from a given interval.

Besides, we conduct experiments on five QA benchmarks: MMLU (Hendrycks et al., 2020), ARC (Clark et al., 2018), and CommonsenseQA (CSQA) (Talmor et al., 2019), hellaswag (Zellers et al., 2019), CEval (Huang et al., 2023), which are all benchmarks widely used for LLM evaluation. Our selection of benchmarks takes into account the diversity of tasks and domains. The diversity of these domains allows us to derive general observations and enable cross-domain transfer exploration.

Model	Method	ID Token	OOD Token	Marble Color	Binomial Dist.	Uniform Dist.
Qwen3-1.7B	Baseline	4.3148	4.3972	7.4190	15.3000	9.1580
	TS	0.5015	0.4999	0.2019	0.5748	0.0061
	GS	0.5210	0.5509	0.1399	0.4525	0.0055
	AdaptiveTS	0.5068	0.5115	0.7538	1.2804	0.0424
	VS	0.4867	0.4872	0.1199	1.4506	0.0653
	BoT	0.0831	0.0856	0.0129	0.2019	0.0060
Qwen3-8B	Baseline	3.6550	3.7828	4.7568	2.8541	2.6322
	TS	0.3193	0.3292	0.1335	0.2111	0.0127
	GS	0.3193	0.3292	0.0387	0.1792	0.0088
	AdaptiveTS	0.6621	0.5917	1.6973	0.5619	0.1876
	VS	0.3137	0.3242	0.0391	0.1275	0.4732
	VRS	0.3688	0.3784	0.1592	0.4178	0.2067
	BoT	0.0405	0.0555	0.0362	0.0470	0.0071
Qwen3-14B	Baseline	3.1105	3.2096	0.7248	4.5420	2.1391
	TS	0.2663	0.2879	0.0479	0.3211	0.0151
	GS	0.2877	0.3199	0.0218	0.1903	0.0161
	AdaptiveTS	0.2706	0.2907	0.0473	0.3388	0.0802
	VS	0.2602	0.2815	0.0279	0.1106	0.3377
	BoT	0.0932	0.0996	0.0039	0.0012	0.0108
LLaMA3.1-8B	Baseline	2.2192	2.1967	0.1084	2.0049	1.6495
	TS	0.4937	0.4894	0.1023	0.4289	0.0073
	GS	0.4937	0.4890	0.0753	0.4313	0.0071
	AdaptiveTS	0.5861	0.5717	0.1007	0.4289	0.0941
	VS	0.4973	0.4880	0.0612	0.4158	0.0180
	VRS	0.3509	0.3589	0.1432	0.6794	0.2067
	BoT	0.0103	0.0150	0.0028	0.0331	0.0070
Mistral-7B	Baseline	2.3977	2.2302	2.1657	0.4477	2.9984
	TS	0.5176	0.5352	0.1145	0.4289	0.0134
	GS	0.5176	0.5371	0.0363	0.4289	0.0114
	AdaptiveTS	0.5999	0.5519	0.1385	0.4289	0.3117
	VS	0.5223	0.5353	0.0295	0.4304	0.0523
	VRS	0.2021	0.2189	0.1163	0.4414	0.0166
	BoT	0.0364	0.0394	0.0068	0.0122	0.0064

Table 1: Comparison of different calibration methods across sampling tasks in terms of KL divergence. Lower KL divergence indicates better alignment with the contextual distribution, and the best result is highlighted in **bold**.

To evaluate whether the benefits of BoT extend beyond classification tasks, we further conduct experiments on three open-ended mathematical reasoning benchmarks: SVAMP (Patel et al., 2021), GSM8K (Cobbe et al., 2021), and ASDiv (Miao et al., 2020). Unlike multiple-choice QA, these tasks require the model to generate intermediate reasoning steps and final answers without selecting from a predefined candidate set.

Metrics We evaluate the model on two perspectives: sampling and QA. For the sampling task, we adopt the KL divergence (Kullback and Leibler, 1951) to evaluate the discrepancy between the contextual probability distribution and the predicted probability distribution over candidates. For general QA tasks, we report accuracy and AUROC to evaluate predictive performance and use ECE ($bin = 15$) or Brier score to assess calibration qual-

ity. Our evaluation protocol follows the mainstream LLM evaluation frameworks, such as HuggingFace LLM Leaderboard, EleutherAI lm-harness, and the original MMLU implementation. Our evaluation mainly considers the zero-shot setting, which excludes biases introduced by in-context examples.

Baselines We compare our method with three common calibration baselines: Temperature scaling (TS) (Guo et al., 2017), Adaptive Temperature Scaling (AdaptiveTS) (Xie et al., 2024), Vector scaling (VS) (Guo et al., 2017). The temperature is restricted from 0.1 to 20. We also consider baselines fit for sampling settings: Group scaling (GS) (Kumar et al., 2018) and Verbalized rejection sampling (Xiao et al., 2025) with at least 20 accepted samples. For baselines that require hyperparameter selection, we use the validation set as the calibration set when both validation and

Model	Method	ARC-E		ARC-C		CSQA		HellaSwag		MMLU		CEval	
		ACC	ECE	ACC	ECE	ACC	ECE	ACC	ECE	ACC	ECE	ACC	ECE
Qwen3-1.7B	Baseline	87.58	11.45	73.29	24.42	62.74	34.69	47.47	41.21	54.62	39.77	53.68	36.38
	BoT	90.19	9.20	78.67	19.42	69.62	27.72	53.89	38.64	59.21	34.77	56.71	36.34
	TS	87.58	3.36	73.29	5.47	63.31	6.42	47.47	4.06	54.62	6.98	53.68	5.12
	AdaptiveTS	87.58	3.50	73.29	9.02	63.31	7.14	47.47	4.10	54.62	6.90	53.68	5.22
	VS	87.46	2.17	72.78	5.44	63.06	5.38	49.46	4.12	54.39	6.93	53.53	5.43
	BoT w/ TS	90.19	1.84	78.67	3.34	69.62	3.17	53.89	3.57	59.21	4.47	56.71	3.86
Qwen3-8B	Baseline	96.46	2.87	90.19	8.37	81.00	16.73	74.48	15.83	70.10	23.85	68.47	21.60
	BoT	97.05	1.65	91.98	5.18	82.06	9.29	75.46	6.23	72.05	18.40	74.10	15.36
	TS	96.46	0.41	90.19	2.11	81.00	2.29	74.48	2.36	70.10	3.57	68.47	2.65
	AdaptiveTS	96.46	0.40	90.19	1.47	81.00	2.34	74.48	1.82	70.10	3.39	68.47	2.81
	VS	96.25	1.17	90.27	1.54	80.75	2.61	74.35	1.97	70.55	3.47	69.85	2.44
	BoT w/ TS	97.05	0.37	91.98	2.18	82.06	1.95	75.46	0.88	72.05	5.50	74.10	4.15
Qwen3-14B	Baseline	97.60	2.27	92.60	6.68	81.30	16.54	70.10	24.52	78.60	18.28	72.20	18.59
	BoT	98.10	1.94	93.30	6.42	81.70	16.13	70.70	21.85	80.40	16.89	74.70	19.91
	TS	97.60	0.80	92.60	1.65	81.30	2.47	70.10	6.86	78.60	3.22	72.20	3.52
	AdaptiveTS	97.60	0.92	92.60	2.84	81.30	2.65	70.10	6.83	78.60	3.29	72.20	3.57
	VS	97.70	1.12	93.00	1.96	81.00	3.53	71.50	7.03	78.90	3.86	73.00	3.58
	BoT w/ TS	98.10	0.71	93.30	2.15	81.70	2.47	70.70	6.61	80.40	3.19	74.70	2.77
LLaMA3.1-8B	Baseline	90.61	4.12	75.34	13.95	73.71	15.03	47.20	23.96	57.85	26.80	45.75	27.84
	BoT	90.86	7.21	79.86	15.66	76.09	14.26	58.13	21.80	60.01	24.79	46.07	14.72
	TS	90.61	1.81	75.34	4.52	73.71	3.28	47.20	2.68	57.85	5.40	45.75	4.62
	AdaptiveTS	90.61	2.07	75.34	5.06	73.71	3.49	47.20	2.80	57.85	5.90	45.75	4.36
	VS	90.66	1.40	77.39	3.88	74.93	3.55	55.17	4.91	60.00	3.00	45.99	2.38
	BoT w/ TS	90.86	3.29	79.86	3.25	76.09	4.50	58.13	2.22	60.01	7.76	46.07	2.16
Mistral-7B	Baseline	89.10	8.90	75.43	20.86	70.51	26.08	54.50	30.70	58.56	31.44	38.79	41.36
	BoT	88.17	2.80	78.58	3.63	67.81	5.31	54.00	3.42	58.61	11.53	40.62	15.64
	TS	89.10	2.60	75.43	4.74	70.51	4.74	54.50	2.66	58.56	5.99	38.79	3.46
	AdaptiveTS	89.10	2.57	75.43	4.63	70.51	4.33	54.50	2.49	58.56	5.84	38.79	3.40
	VS	88.97	2.60	75.51	3.88	70.60	4.06	58.27	0.53	58.81	5.89	38.54	4.19
	BoT w/ TS	88.17	2.38	78.58	2.62	67.81	3.31	54.00	2.79	58.61	4.82	40.62	3.10

Table 2: Accuracy and ECE (%) results across six benchmarks for different models and calibration methods. The best result is highlighted in **bold**, and shaded rows denote our methods.

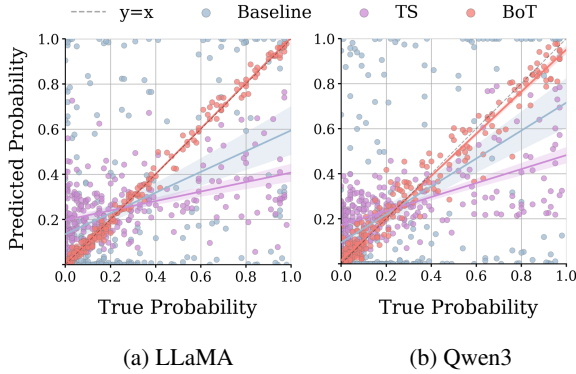


Figure 3: Sampling performance comparison on OOD Token tasks between the baseline, TS, and BoT tuning.

test sets are available, and evaluate all methods on the test set. When the test set is not publicly available, we instead sample 1k examples from the training split for calibration and use the validation set for evaluation. In the main experiments, Bag-of-Tokens tuning uses 100K synthetic samples. On

average, training is conducted for 10 hours on a single NVIDIA RTX PRO 6000 GPU (96GB). We also evaluate Dirichlet sampling as a natural alternative in Appendix E.

5.2 BoT Tuning Transfers Across Diverse Sampling Tasks

As shown in Table 1 and not surprisingly, temperature scaling and its variants fail to yield samples that faithfully reflect the target distribution across most sampling tasks. Group Scaling performs similarly to standard temperature scaling, and the learned group-level temperatures remain close to each other. As the group is split by token, we can draw that the mismatch between the predicted probabilities and the contextual probabilities is not primarily determined by token content, but is more likely related to systematic positional or structural biases in the model distribution. However, the tuned model can consistently reduce KL divergence

across all models and settings. In most cases, the reduction is up to one order of magnitude, suggesting that **BoT tuning effectively improves token-level calibration and generalizes well across distributions and tasks**. Figure 3 further illustrates the performance. The scatter points of TS still exhibit substantial deviations from the ideal calibration line, indicating that post-hoc rescaling cannot fully correct the distributional mismatch. In contrast, our method produces predictions that are closely aligned with the target distribution. Overall, these results provide strong evidence that our method does not merely memorize the synthetic BoT format but improves the model’s ability to calibrate its logits with the in-context probability distribution and sample accordingly.

5.3 BoT Tuning Improves Downstream QA Performance and Calibration

We compare our method with several calibration methods, including TS, adaptiveTS, and VS. Table 2 reports the accuracy and ECE. Table 2 shows two main trends in ECE: first, compared with the base model, our tuning method reduces ECE across six benchmarks. This indicates that tuning improves calibration regardless of the original calibration quality of the base model. Second, after applying TS to the tuned model, the resulting ECE is lower than that obtained by applying post-hoc calibration methods to the base model on a majority of benchmarks. This suggests that tuning not only improves calibration directly, but also makes the model more calibratable. For accuracy, the BoT tuning method improves performance in most cases, with the largest gain being approximately 11%. The improvement in accuracy suggests that BoT Tuning not only calibrates predictive confidence but can also alter the relative logits of candidate answers in a way that improves classification decisions. Additional results on AUROC and Brier Score are provided in Appendix F.

5.4 BoT Tuning Remains Effective across Model Scales

BoT remains effective across different model scales within the Qwen family. As shown in Table 1, Qwen3-1.7B, Qwen3-8B, and Qwen3-14B exhibit considerable contextual probability misalignment before tuning, with the smallest model showing larger initial KL divergence. BoT substantially reduces this divergence across all five sampling tasks and achieves particularly low KL divergence on

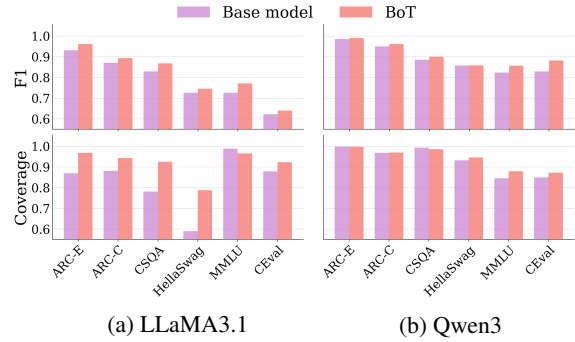


Figure 4: Selective prediction performance comparison across different benchmarks

the natural sampling tasks, including Marble Color, Binomial Dist, and Uniform Dist. The benefits of BoT also transfer consistently to downstream QA benchmarks. Table 2 shows that BoT generally improves answer accuracy across model sizes, while combining BoT with temperature scaling yields strong calibration performance. Although standalone post-hoc calibration can occasionally achieve lower ECE on individual benchmarks, it does not provide comparable accuracy improvements to those achieved by BoT.

5.5 BoT Tuning Improves Selective Prediction Performance

Because selective prediction relies on confidence derived from logits to determine whether to answer or abstain, the presence of the Context-Logit Probability Gap may lead to suboptimal decisions. To evaluate its practical impact, we consider selective prediction in which the model answers only when its confidence exceeds a threshold which is selected on the validation set and then fixed for evaluation on the test set. As shown in Figure 4, BoT consistently improves F1 across the evaluated datasets for both LLaMA and Qwen. Importantly, these gains are achieved while coverage remains comparable or substantially increases in most cases. This suggests that the F1 improvements are not simply obtained through more aggressive abstention, but instead reflect more effective selective prediction, consistent with better calibrated confidence estimates. These results provide further evidence that reducing the Context-Logit Probability Gap benefits downstream selective prediction.

5.6 Ablation Experiment

We evaluate the main components of our method on sampling tasks and downstream QA benchmarks. We compare it with four variants: (1)

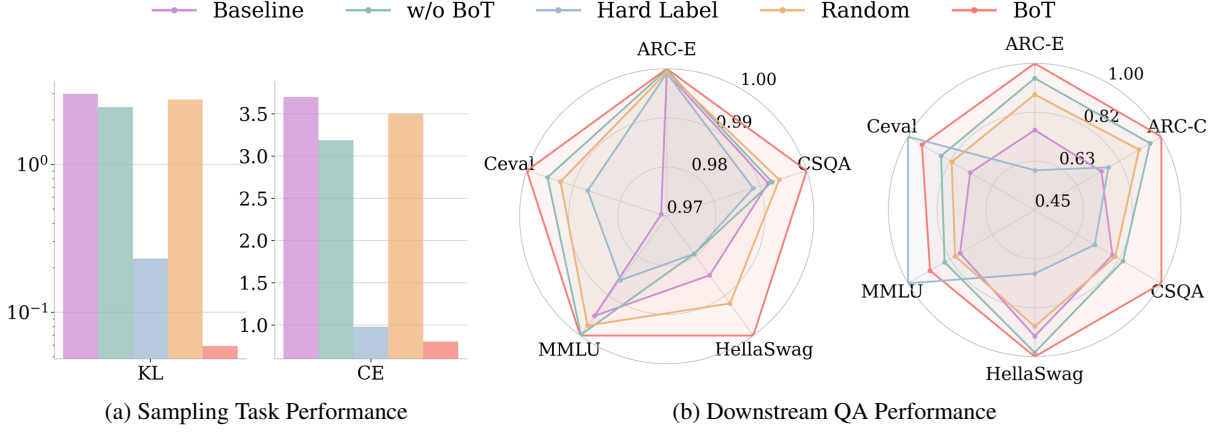


Figure 5: Ablation results on sampling and QA tasks. The left panel compares sampling behavior using KL divergence and cross-entropy (CE), while the right panel presents relative QA performance under a unified orientation, including AUROC, and reversed Brier Score.

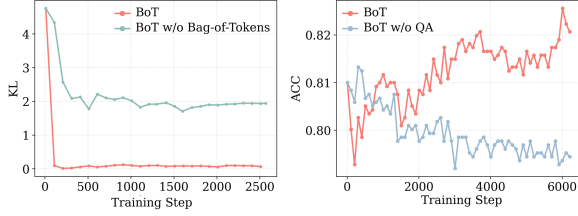


Figure 6: Training dynamics of different ablation variants. The left panel reports KL divergence on the Marble Color sampling task, while the right panel shows QA accuracy over training.

w/o QA, which removes QA supervision; (2) w/o BoT, which removes BoT supervision; (3) HARD LABEL, which replaces soft-label supervision with hard-label BoT data to examine whether the improvement simply comes from adding hard-label supervision; (4) RANDOM, which replaces grounded soft-label with random soft-label to test whether the improvement is mainly caused by flattening the logits distribution.

As shown in Figure 5, w/o BoT alone does not sufficiently improve the sampling distribution: its KL divergence remains much higher than ours and is close to the baseline. This indicates that QA supervision provides limited guidance for distribution alignment. The HARD LABEL variant further shows that the improvement does not simply come from adding hard-label supervision. Its downstream performance is worse than the baseline in most cases, suggesting that hard-label BoT supervision may over-constrain the output distribution and harm QA capability. This supports the role of QA data in preserving task-solving ability while applying distributional supervision. Besides, RANDOM performs worse than our method in both KL divergence and downstream QA metrics. Thus,

the gain is unlikely to come only from flattening the logits distribution. Instead, the results suggest that grounded soft-label provide useful signals for the logits, whereas random soft-label change the distribution without providing relevant guidance.

Figure 6 shows that BoT data and QA data play distinct roles during training. Without BoT supervision, the model fails to effectively reduce KL divergence, indicating that BoT provides the primary signal for distributional alignment. In contrast, removing QA supervision leads to a gradual degradation in accuracy, suggesting that QA data is important for preserving general ability. Together, these trends suggest that BoT supervision drives distributional calibration, while QA supervision helps maintain downstream task performance.

6 Conclusion

In this work, we identify the context-logit probability gap, showing that contextual probabilities are not always faithfully reflected in model token logits. We further show that post-hoc temperature scaling is insufficient to close this gap. To address this issue, we propose Bag-of-Tokens (BoT) Tuning, a simple fine-tuning method that uses synthetic data as soft supervision and directly calibrates predicted token logit with the contextual probabilities. Experiments show that BoT Tuning improves sampling fidelity both on different distributions and held-out tokens, transfers to new sampling settings, and reduces ECE while improving QA accuracy on downstream QA. These findings suggest that calibrating LLM logits with contextual probabilities is a lightweight and effective way to improve the sampling fidelity and calibration of LLMs.

Limitations

This work focuses only on a tuning-based method. Reinforcement learning may also be a viable alternative, as the reward can be designed to encourage low KL divergence and penalize high KL divergence. Although the ablation experiments partially explain why the tuning improves performance, the underlying mechanism is not fully understood. In particular, how the tuning changes the model in parameter space remains underexplored. Besides, a future direction is to investigate whether the model can better leverage in-context information to estimate *conditional* probability.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments and suggestions. This work was supported by the National Science Foundation of China (62225602), the Basic Research Program of Jiangsu (BK20253021, BK20251325), and the Big Data Computing Center of Southeast University.

References

- Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, and 12 others. 2022. [Training a helpful and harmless assistant with reinforcement learning from human feedback](#). *Preprint*, arXiv:2204.05862.
- Marc Brysbaert and Boris New. 2009. Moving beyond kučera and francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english. *Behavior research methods*, 41(4):977–990.
- Silvia Cappelletti, Tobia Poppi, Samuele Poppi, Zheng-Xin Yong, Diego Garcia-Olano, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. 2026. [Improving llm first-token predictions in multiple-choice question answering via output prefilling](#). *Preprint*, arXiv:2505.15323.
- Khyathi Chandu, Linjie Li, Anas Awadalla, Ximing Lu, Jae Sung Park, Jack Hessel, Lijuan Wang, and Yejin Choi. 2025. Certainlyuncertain: A benchmark and metric for multimodal epistemic and aleatoric awareness. In *International Conference on Learning Representations*.
- Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pages 2924–2936.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *Preprint*, arXiv:2110.14168.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International conference on machine learning*, pages 10764–10799. PMLR.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.
- Yangyang Guo, Fangkai Jiao, Zhiqi Shen, Liqiang Nie, and Mohan S. Kankanhalli. 2024. UNK-VQA: A dataset and a probe into the abstention ability of multi-modal large models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 46(12):10284–10296.
- Ritwik Gupta, Rodolfo Corona, Jiaxin Ge, Eric Wang, Dan Klein, Trevor Darrell, and David M Chan. 2025. Enough coin flips can make llms act bayesian. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7634–7655.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

- Aspen K Hopkins, Alex Renda, and Michael Carbin. 2023. Can llms generate random numbers? evaluating llm sampling in controlled domains. In *ICML 2023 workshop: sampling and optimization in discrete space*.
- Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, pages 62991–63010. Curran Associates, Inc.
- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2023. *Mistral 7b*. Preprint, arXiv:2310.06825.
- Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*.
- Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.
- Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. 2018. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2805–2814. PMLR.
- Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 785–794.
- Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher R  , Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. *Holistic evaluation of language models*. Preprint, arXiv:2211.09110.
- Shudong Liu, Zhaocong Li, Xuebo Liu, Runzhe Zhan, Derek F Wong, Lidia S Chao, and Min Zhang. 2024. Can llms learn uncertainty on their own? expressing uncertainty effectively in a self-training manner. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21635–21645.
- Charles Lovering, Michael Krumdick, Viet Dac Lai, Varshini Reddy, Seth Ebner, Nilesh Kumar, Rik Koncel-Kedziorski, and Chris Tanner. 2025. Language model probabilities are not calibrated in numeric contexts. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 29218–29257.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pages 46534–46594. Curran Associates, Inc.
- Nicole Meister, Carlos Guestrin, and Tatsunori B Hashimoto. 2025. Benchmarking distributional alignment of large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 24–49.
- Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, pages 975–984.
- Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 2080–2094.
- Pouya Pezeshkpour and Estevam Hruschka. 2024. Large language models sensitivity to the order of options in multiple-choice questions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2006–2017.
- Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14.
- Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell L. Gordon, Niloofar Miresghallah, Christopher Michael Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin Choi. 2024. Position: A roadmap to pluralistic alignment. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, pages 46280–46302.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference*

of the North American Chapter of the Association for Computational Linguistics: *Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Qwen Team. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442.

Katherine Van Koeveing and Jon Kleinberg. 2024. How random is random? evaluating the randomness and humanness of llms’ coin flips. *arXiv preprint arXiv:2406.00092*.

Deng-Bao Wang, Lei Feng, and Min-Ling Zhang. 2021. Rethinking calibration of deep neural networks: Do not be afraid of overconfidence. *Advances in Neural Information Processing Systems*, 34:11809–11820.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Sheng-Lun Wei, Cheng-Kuang Wu, Hen-Hsen Huang, and Hsin-Hsi Chen. 2024. Unveiling selection biases: Exploring order and token sensitivity in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5598–5621.

Tim Z Xiao, Johannes Zenn, Zhen Liu, Weiyang Liu, Robert Bamler, and Bernhard Schölkopf. 2025. Flipping against all odds: Reducing llm coin flip bias via verbalized rejection sampling. *arXiv preprint arXiv:2506.09998*.

Johnathan Xie, Annie S Chen, Yoonho Lee, Eric Mitchell, and Chelsea Finn. 2024. Calibrating language models with adaptive temperature scaling. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18128–18138.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 4791–4800.

Min-Ling Zhang and Deng-Bao Wang. 2026. Uncertainty calibration in deep learning: Methods, emerging challenges, and llm frontiers. *Journal of Computer Science and Technology*, pages 1–23.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2024. Large language models are not robust multiple choice selectors. In *International Conference on Learning Representations*, pages 19426–19454.

A Description of Other Baselines

We compare our method with four common calibration baselines. Temperature scaling (TS) (Guo et al., 2017) rescales all logits with a global temperature. Vector scaling (VS) (Guo et al., 2017) learns class-wise calibration parameters on a validation set and applies them to the test set. Adaptive Temperature Scaling (AdaptiveTS) (Xie et al., 2024) predicts temperatures from model features for every forward pass and performs post-hoc calibration under heterogeneous calibration shifts. For baselines that require hyperparameter selection, we use a separate calibration set from the training split. Specifically, we randomly sample 1k training examples for calibration and hyperparameter tuning, while keeping the held-out test set identical across all methods.

We also consider baselines fit for sampling settings. Group scaling (GS) (Kumar et al., 2018) partitions candidate words into five groups according to their token bias, defined as the discrepancy between the predicted probability and the target probability, and applies a separate scaling parameter to each group. Verbalized rejection sampling (Xiao et al., 2025) is included only for sampling tasks, since it is tailored to correcting sampled outputs.

In addition to calibration and sampling baselines, we compare our method with several representative reasoning strategies for large language models. Self-consistency (Wang et al., 2022) samples multiple reasoning trajectories and selects the final answer by majority voting. Program-aided language models (PAL) (Gao et al., 2023) prompt the model to generate executable Python programs, using program execution to reduce arithmetic and symbolic reasoning errors. Self-Refine (Madaan et al., 2023) follows an iterative refinement procedure, where the model first produces an initial answer, then critiques its own response, and finally generates a revised answer based on the feedback.

B Description of Datasets

Task Description We evaluate sampling performance under five settings. The following are

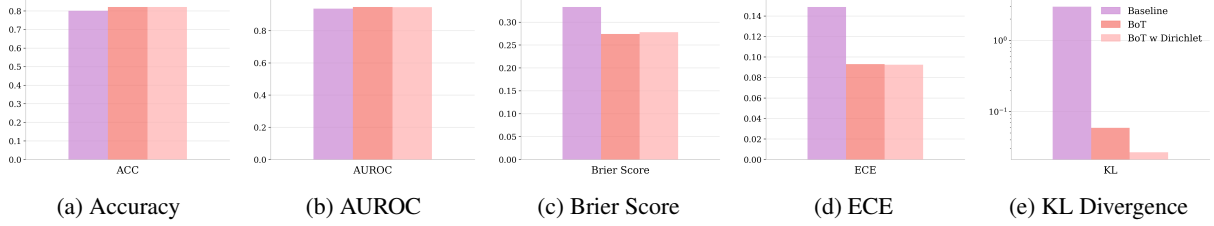


Figure 7: The effectiveness of BoT tuning does not rely on a specific distribution generator.

detailed setting descriptions or representative examples, respectively:

- ID Token uses the same task formulation and candidate pool as training, but with unseen target distributions.
- OOD Token uses the same task formulation as training, but replaces the candidates with unseen token sets.
- Marble Color (Lovering et al., 2025): From 17 red marbles and 99 blue marbles, Billy reached blindly into the bag and grabbed a marble with the color [red/blue]
- Binomial Distribution adapted from (Xiao et al., 2025): Let X follow a *Binomial distribution* with $n = 5$ and $p = 0.4$. A value of X is sampled according to this distribution. Sample the value of X from [0/1/2/3/4/5].
- Uniform Distribution (Hopkins et al., 2023): An integer is sampled *uniformly* at random from the interval $[3, 8)$, where the interval is left-closed and right-open. The integer is [3/4/5/6/7]
- Mathematical Task Example: There are two third-grade classes. Class A has 32 students and Class B has 62 students. Today, 3 students from Class A and 5 students from Class B are absent. A student who is at school today is selected uniformly at random. Which class is the selected student from? Answer with one word: [A/B].

Specifically, MMLU and ARC consist of 4-option MCQs, while CSQA consists of 5-option ones, and MMLU covers tests from 4 domain categories spanning 57 subjects.

C Reproducibility

Table 4 reports the default training hyperparameters used in our experiments. It summarizes the key

settings for sequence processing, optimization, checkpointing, data loading, mixed precision, and distributed training, thereby improving the transparency and reproducibility of our experimental setup. Besides, the code will be released after the paper is accepted.

D Method Description

Temperature scaling (TS) (Guo et al., 2017) rescales all logits with a global temperature. Adaptive Temperature Scaling (AdaptiveTS) (Xie et al., 2024) predicts temperatures from model features for every forward pass and performs post-hoc calibration under heterogeneous calibration shifts. Vector scaling (VS) (Guo et al., 2017) learns class-wise calibration parameters on a validation set and applies them to the test set.

E The effectiveness of BoT tuning does not rely on a specific distribution generator.

We design BOT W DIRICHLET, which replaces the default training distributions with samples drawn from a Dirichlet distribution with $\alpha = 0.5$, to evaluate whether BoT tuning remains effective under an alternative distribution generator. As illustrated in Figure 7, BOT W DIRICHLET achieves competitive performance compared with the default BoT construction, suggesting that the effectiveness of BoT tuning does not rely on a specific distribution generator. This result further indicates that the key factor behind BoT tuning is the explicit supervision of contextual probability distributions, rather than the particular sampling rule used to instantiate those distributions. Therefore, BoT tuning can be viewed as a flexible distributional alignment framework that remains robust under different ways of generating target probabilities.

Model	Method	ARC-E		ARC-C	CSQA		HellaSwag		MMLU		CEval	
		AUROC	Brier	Brier	AUROC	Brier	AUROC	Brier	AUROC	Brier	AUROC	Brier
Qwen3-8B	Baseline	0.9975	0.0636	0.1748	0.9596	0.3526	0.9160	0.3824	0.9075	0.5164	0.9008	0.5081
	TS	0.9979	0.0543	0.1460	0.9621	0.2829	0.9171	0.3682	0.9086	0.3931	0.9034	0.4019
	VS	0.9973	0.0589	0.1528	0.9622	0.2790	0.9177	0.3638	0.9091	0.3805	0.9041	0.3926
	AdaptiveTS	0.9978	0.0054	0.1458	0.9622	0.2824	0.9170	0.3682	0.9086	0.3824	0.9034	0.4022
	Ours	0.9983	0.0477	0.1293	0.9676	0.2778	0.9301	0.3531	0.9120	0.4420	0.9275	0.3947
LLaMA3.1-8B	Baseline	0.9886	0.1508	0.3831	0.9356	0.4794	0.7836	0.7191	0.8193	0.6549	0.7127	0.7857
	TS	0.9887	0.1467	0.3489	0.9382	0.3762	0.7874	0.6349	0.8210	0.5516	0.7150	0.6700
	VS	0.9894	0.1316	0.3242	0.9401	0.3534	0.7891	0.5863	0.8380	0.5066	0.7127	0.6541
	AdaptiveTS	0.9886	0.1472	0.3491	0.9381	0.3764	0.7881	0.6348	0.8200	0.5523	0.7146	0.6697
	Ours	0.9883	0.1616	0.3531	0.9351	0.4274	0.8228	0.6225	0.8298	0.6752	0.7164	0.8993
Mistral-7B	Baseline	0.9841	0.1917	0.4454	0.9084	0.5463	0.8045	0.7284	0.8312	0.7005	0.6529	0.9698
	TS	0.9838	0.1597	0.4275	0.9112	0.3556	0.8102	0.5841	0.8317	0.5241	0.6569	0.6989
	VS	0.9832	0.1598	0.3550	0.9121	0.4209	0.8122	0.5577	0.8307	0.5199	0.6570	0.6985
	AdaptiveTS	0.9839	0.1597	0.3578	0.9113	0.4276	0.8101	0.5841	0.8320	0.5234	0.6570	0.6989
	Ours	0.9853	0.1750	0.3078	0.9090	0.4438	0.8302	0.5870	0.8297	0.5552	0.6791	0.7255

Table 3: AUROC and Brier Score results across six benchmarks. Higher AUROC indicates better uncertainty discrimination, while lower Brier Score indicates better probabilistic calibration.

Category	Argument	Default	Description
Sequence and Tokenization	max_length	4096	Maximum input sequence length.
Sequence and Tokenization	no_force_single_token	False	Disable single-token constraint when enabled.
Optimization	per_device_train_batch_size	1	Batch size per device.
Optimization	gradient_accumulation_steps	8	Number of gradient accumulation steps.
Optimization	learning_rate	1×10^{-5}	Learning rate for optimization.
Optimization	num_train_epochs	1.0	Number of training epochs.
Logging and Checkpointing	logging_steps	10	Logging interval in training steps.
Logging and Checkpointing	save_steps	100	Checkpoint saving interval.
Logging and Checkpointing	save_total_limit	2	Maximum number of saved checkpoints.
Logging and Checkpointing	resume_from_checkpoint	None	Optional checkpoint for resuming training.
Data Loading	dataloader_num_workers	4	Number of data loading workers.
Data Loading	dataloader_pin_memory	True	Whether to use pinned memory.
Precision and Distributed Training	bf16	True	Enable bfloat16 mixed precision.
Precision and Distributed Training	fp16	False	Enable float16 mixed precision when set.
Precision and Distributed Training	tf32	True	Enable TensorFlow-32 acceleration.
Precision and Distributed Training	ddp_find_unused_parameters	False	DDP unused-parameter detection setting.
Precision and Distributed Training	deepspeed_config	None	Optional DeepSpeed configuration path.

Table 4: Training hyperparameters used for reproducibility.

F BoT Tuning Improves AUROC and Brier Score.

Table 3 reports the AUROC and Brier Score results across six benchmarks. AUROC measures the model’s ability to distinguish correct from incorrect predictions, where a higher value indicates better uncertainty discrimination. Brier Score measures the mean squared error between predicted probabilities and the ground-truth correctness labels, where a lower value indicates better probabilistic calibration. Our method achieves competitive AUROC and consistently reduces Brier Score across six QA benchmarks, suggesting that the BoT tuning improves the uncertainty discrimination of model and reliability of model confidence expression.

G BoT Tuning Outperforms Output Prefilling

Recent work has shown that output prefilling can improve first-token predictions in multiple-choice question answering (MCQA) by reducing the mismatch between the model’s generative behavior and the expected answer format (Cappelletti et al., 2026). We further compare BoT with this inference-time intervention by prepending the phrase *The correct option is:* immediately before model generation and evaluate the resulting first-token candidate probabilities. Table 5 reports the ACC and ECE across six MCQA benchmarks and three backbone models. As the table illustrates, the output prefilling yields limited and inconsistent

Model	Method	ARC-E		ARC-C		CSQA		HellaSwag		MMLU		CEval	
		ACC	ECE	ACC	ECE	ACC	ECE	ACC	ECE	ACC	ECE	ACC	ECE
Qwen3	Baseline	96.46	2.87	90.19	8.37	81.00	16.73	74.48	15.83	70.10	23.85	68.47	21.60
	Prefilling	95.66	3.60	87.88	9.43	78.95	18.01	75.71	17.16	67.12	25.27	65.87	21.22
	BoT	97.05	1.65	91.98	5.18	82.06	9.29	75.46	6.23	72.05	18.40	74.10	15.36
LLaMA3.1	Baseline	90.61	4.12	75.34	13.95	73.71	15.03	47.20	23.96	57.85	26.80	45.75	27.84
	Prefilling	89.89	4.69	75.42	14.27	73.63	15.91	49.97	22.86	57.10	26.24	45.33	28.45
	BoT	90.86	7.21	79.86	15.66	76.09	14.26	58.13	21.80	60.01	24.79	46.07	14.72
Mistral	Baseline	89.10	8.90	75.43	20.86	70.51	26.08	54.50	30.70	58.56	31.44	38.79	41.36
	Prefilling	89.60	8.60	76.45	20.07	71.17	25.54	58.43	28.98	57.76	33.65	39.28	43.53
	BoT	88.17	2.80	78.58	3.63	67.81	5.31	54.00	3.42	58.61	11.53	40.62	15.64

Table 5: Comparison of the original models, output prefilling, and BoT on multiple-choice question answering. ACC denotes answer accuracy, and ECE denotes expected calibration error. Higher ACC and lower ECE are better.

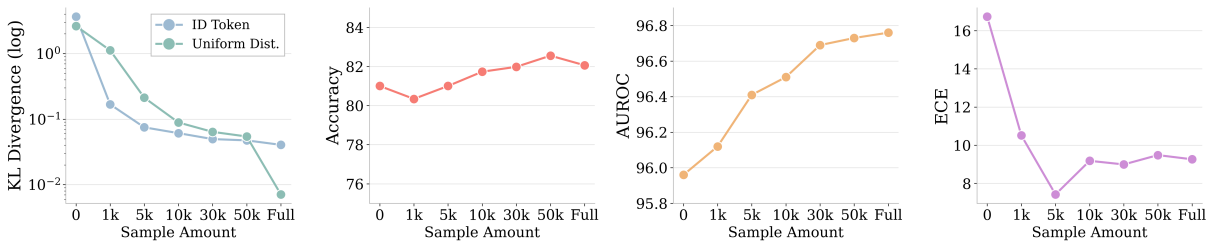


Figure 8: Sensitivity of BoT to the amount of synthetic training data on Qwen3-8B. Lower KL, ECE, and Brier score are better, while higher ACC and AUROC are better.

improvements over the original models. For Qwen, prefilling improves accuracy only on HellaSwag, while degrading accuracy on the other five benchmarks and providing little improvement in calibration. For LLaMA, the gains are similarly modest, with noticeable accuracy improvement mainly on HellaSwag. Mistral benefits more from prefilling on several benchmarks, but its calibration remains largely unchanged or even deteriorates on MMLU and CEval. In contrast, BoT generally provides substantially larger calibration improvements and more consistent accuracy gains. One possible reason for the relatively small effect of prefilling in our setting is that the original evaluation prompt already strongly constrains the output format using the instruction *Answer directly with one letter from {labels}. Answer:.* Such an explicit position constraint may already mitigate part of the format misalignment and instruction issues targeted by prefilling. BoT, by contrast, directly optimizes the token distribution during fine-tuning, and therefore addresses a broader problem.

H BoT Tuning Is Effective with Limited Synthetic Data

We further investigate the sensitivity of BoT to the amount of synthetic training data using Qwen3-

8B. Specifically, we vary the number of synthetic BoT examples from 1k to the full training set and evaluate both distribution matching on the synthetic tasks and downstream performance on CSQA. As Figure 8 shows, BoT achieves substantial improvements with only a small amount of synthetic data. With 5k examples, the KL divergence decreases and CSQA calibration improves considerably while accuracy remains at 81.00. This indicates that approximately 5k examples are already sufficient to obtain most of the practical benefits of BoT.

Increasing the training set beyond 5k primarily improves distributional fidelity and yields smaller downstream gains. The synthetic-task KL values continue to decrease with additional data, whereas CSQA accuracy, AUROC, and Brier score change more gradually and become relatively stable from approximately 30k examples onward. In contrast, ECE exhibits some non-monotonic variation, suggesting that additional synthetic data does not uniformly improve every calibration metric. Overall, these results demonstrate a favorable cost-benefit trade-off: moderate amounts of synthetic BoT data are sufficient for substantial downstream improvements, while larger datasets mainly provide further gains in strict distribution matching.

Method	SVAMP		GSM8K		ASDiv	
	Acc $\uparrow$	ECE $\downarrow$	Acc $\uparrow$	ECE $\downarrow$	Acc $\uparrow$	ECE $\downarrow$
Baseline	90.27	7.48	88.89	9.52	92.14	1.10
CoT	<u>92.13</u>	6.38	92.73	4.36	94.44	2.29
Self-Refine	91.90	4.04	93.62	7.06	<u>95.34</u>	2.41
Program-Aided Prompting	91.23	33.33	87.77	37.65	89.94	3.23
BoT w/ CoT	93.20	<u>5.92</u>	<u>92.78</u>	<u>6.97</u>	95.35	0.91

Table 6: Accuracy and ECE (%) on open-ended mathematical reasoning benchmarks using Qwen3-8B. The best and second-best results are shown in bold and underlined, respectively.

I BoT Tuning Extends to Open-Ended Reasoning Generation

We report both accuracy and ECE to evaluate whether BoT can improve not only reasoning correctness but also the reliability of model confidence. The results in Table 6 show that BoT Tuning transfers effectively to open-ended reasoning generation. Compared with the baseline, the BoT model with CoT achieves the best accuracy on SVAMP and ASDiv and the second on GSM8K. BoT also provides consistent calibration benefits over the baseline. Among all compared methods, BoT with CoT achieves the best ECE on ASDiv and the second on both SVAMP and GSM8K. In particular, on ASDiv, BoT with CoT simultaneously achieves the highest accuracy and the lowest ECE, indicating that the improvement in reasoning accuracy does not come at the expense of calibration. The comparison with other reasoning strategies further highlights that the effect of BoT is not restricted to token probabilities.

J Prompt Templates

This appendix lists the prompt templates used to instantiate Bag-of-Tokens examples. All these prompts are generated by LLMs with instructions to generate variants of the prompt in Figure 1. Placeholders such as {bag_description}, {distribution}, and {word_list} are filled during data construction.

J.1 Mathematical Templates

Categorical Random Variable

Let X be a categorical random variable with the following distribution:
{distribution}
Sample one realization of X and identify the word printed on it.
Restrictions:
- Sampling must follow the given probabilities exactly.

- Semantic meaning is irrelevant.
Return a randomly selected token from: {word_list}

J.2 Canonical Templates

Canonical Bag Sampling

Imagine a bag containing small balls, each with a word written on it. {bag_description}
You reach into the bag and randomly select one ball according to this exact probability distribution, just like drawing a physical ball from a real bag. Then predict which word is on the ball you selected.

CRITICAL:

- Select based SOLELY on the given frequency distribution.
- Ignore all personal biases, preferences, word meanings, or prior knowledge about these words.
- The selection must reflect ONLY the exact proportions provided.
- If you output anything other than exactly one candidate word, the answer is invalid.

MANDATORY OUTPUT FORMAT:

- Output ONLY the single **English** word: {word_list}
- No other text, explanations, punctuation, or characters.
- Do not translate or modify the word.
- The semantic meaning of the words in the bag is irrelevant.

Canonical Short Bag Sampling

Imagine a bag containing small balls, each with a word written on it. {bag_description}
Select a ball at random and predict its labeled word.

CRITICAL:

- Select based SOLELY on the given frequency distribution.

- Ignore all personal biases, preferences, word meanings, or prior knowledge.
- The selection must reflect ONLY the exact proportions provided.

MANDATORY OUTPUT FORMAT:

- Output ONLY the single English word: {word_list}
- No other text, explanations, punctuation, or characters.

J.3 Paraphrased Templates

Labeled Item Sampling

You have a bag of items, where each item is labeled with a word.
Each word appears with the following probability distribution:
{distribution}

Your job is to randomly draw one item according to this distribution and then state which word it displays.

Important:

- The choice must strictly follow the probabilities.
- Do not rely on word meanings or any personal judgment.
- The meaning of the word implies nothing here.

Answer format:

- Return exactly one English word from: {word_list}
- Do not add anything else.

Discrete Word Distribution

Suppose words are sampled from a discrete distribution given below:
{distribution}

One word is drawn at random following these exact probabilities.
Please report the drawn word.

Constraints:

- Use only the numerical probabilities.
- Ignore semantics and preferences.

Output:

- One word chosen from {word_list}

J.4 Minimal Templates

Minimal Bag Sampling

There is a bag containing small balls, each with a word written on it. {bag_description}

Randomly draw a ball and predict the word written on it.

Valid outputs are in {word_list}

Minimal Distribution Sampling

{distribution}

Pick ONE word randomly from the bag.
Output from {word_list}

J.5 Expanded Templates

Calibrated Model Sampling

You are a perfectly calibrated language model that produces tokens with probabilities that correspond to values in the context and prompt.
I will give you a scenario.

A set of words is placed into a container, where the likelihood of selecting each word is governed by the following probability distribution:
{distribution}

Although these words may carry meanings in natural language, you must completely ignore any semantic interpretation.
The process is purely statistical and random.

Your task is to simulate a single random draw from this distribution and report the selected word.

Output rules:

- Output exactly one word from: {word_list}
- Do not include explanations, punctuation, or formatting.

Physical Draw Forecasting

{bag_description}

Imagine performing a physical random draw that perfectly respects this distribution and forecast the word written on it.

Despite any intuition you may have, the outcome must depend ONLY on the numerical probabilities provided.

Final answer requirements:

- Return exactly one word from {word_list}
- No additional text

Discrete Outcome Sampling

Given a discrete probability distribution over outcomes:
{distribution}

Draw one sample randomly from this distribution and predict which word is on the ball you sample.

Output specification:
- One English word from {word_list}
- No other output

J.6 Procedural Templates

Input-Output Sampling

Input:
- Probability distribution (word -> probability):
{distribution}

Sample one word randomly and predict which word is on the ball you sample.

Output one word from the following list:
- {word_list}
(single token)

Task-Style Sampling

{bag_description}

Task: stochastic sampling and return the word on the selected ball.

Return exactly one of {word_list}

J.7 Instruction-Weak Templates

Weak Bag Sampling

{bag_description}

Predict which word is randomly chosen.

Answer:

Weak Distribution Sampling

Words and probabilities:
{distribution}

Respond with one word that is randomly selected.

K Data Usage Discussion

We use publicly available datasets and model artifacts for research purposes. BoolQ is released under the Creative Commons Share-Alike 3.0 license, RACE is available for non-commercial research use, and CEval is released under the Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 license. For other datasets and model artifacts, including HellaSwag and open-weight language models, we follow their official licenses and terms of use. We use these resources only for evaluation and describe the evaluated model families, training settings, and evaluation metrics to facilitate reproducibility in Table 4.

Our use of existing datasets and model artifacts is consistent with their intended research and evaluation purposes. We use these artifacts only for academic benchmarking, calibration analysis, and reproducibility, following their original licenses and terms of use. For artifacts created in this work, including the synthetic Bag-of-Tokens data and evaluation code, the intended use is research and reproducibility. We do not redistribute the original datasets or model weights; released code will only provide scripts for reproducing our experiments and will require users to obtain the corresponding artifacts from their official sources.

Personally Identifying Information and Offensive Content. Our synthetic Bag-of-Tokens data is automatically generated from candidate token sets and does not intentionally include names, private user information, demographic attributes, or offensive content. We do not collect new human-subject data or private user data. The external datasets used in our experiments are publicly available benchmark datasets and are used only for research and evaluation under their original terms.

L The Use of Large Language Models

We used large language models (LLMs) exclusively for language polishing. Only generic text that contains no experimental results was refined; no raw data, code, or sensitive information was uploaded to any commercial LLM platform. All outputs were reviewed by the authors for factual accuracy. The models' role was strictly limited to linguistic enhancement, and they are not listed as authors. We assume full responsibility for the entire content of this paper, including any content revised by LLMs.