USING PROBABILISTIC PROGRAMS TO TRAIN INDUCTIVE REASONING IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training Large Language Models (LLMs) for reasoning typically focuses on deductive tasks such as mathematics and coding where correctness is verifiable. Yet, many real-world reasoning problems are inductive: agents must infer uncertain beliefs from sparse, ambiguous observations. For standard fine-tuning methods, it is often challenging to curate large-scale, high-quality labeled datasets and to handle targets that are inherently distributional. In this work, we introduce a novel approach, called Program-based Posterior Training (PPT), to address these limitations: we use an LLM to generate diverse open-world scenarios as probabilistic programs, run probabilistic inference to produce distributional target responses to queries, and then fine-tune on these probabilistic labels. Using this approach, we fine-tune LLMs on 10,000 programmatically generated scenarios and evaluate on held-out motifs, human-labeled judgments, and external benchmarks. Overall, PPT substantially improves estimation accuracy on held-out inductive tasks, increases alignment with human judgments, and transfers to external benchmarks for estimation and calibration. Additionally, the gains in raw calibration are not subsumed by post-hoc temperature scaling or chain-of-thought, showing that the models have more deeply internalized uncertainty compared to output rescaling. Together, these results suggest that probabilistic-program-mediated fine-tuning is a promising approach for post-training LLMs to perform stronger approximate inductive inference.

1 INTRODUCTION

Large language models (LLMs) have made rapid progress in reasoning tasks, especially in mathematics and coding (Wei et al., 2022; Chen et al., 2021). These domains largely involve deductive problems: the model is expected to derive a correct answer through a sequence of logical, deterministic steps. However, many real-world reasoning problems are inductive rather than deductive (Griffiths et al., 2008; Tenenbaum et al., 2011). People routinely infer broader patterns from limited evidence and make predictions in situations where the relevant latent factors are only partially observed (Griffiths et al., 2008). Scientific reasoning, forecasting, diagnosis, and everyday decision-making require drawing uncertain conclusions from sparse and noisy data. This requires inferring latent quantities from noisy observations and making calibrated predictions (Blei, 2014; Bishop, 2006).

Training LLMs to perform inductive inference is challenging because high-quality labeled data is difficult to obtain and the underlying problem is often inherently probabilistic, leading to answers that require uncertainty representation. Human judgments can be collected, but they are expensive and hard to scale across the diverse open-world scenarios where induction is needed. Directly asking stronger LLMs to label such scenarios is scalable (Jagadish et al., 2024; 2025), but can inherit their biases and errors (Chhikara, 2025). In addition, real-world probabilistic reasoning tasks often lack known latent variables or calibrated posterior targets. To better train LLMs to perform inductive inference, we need natural-language problems that are rich enough to resemble open-world inductive reasoning but have labels that are grounded in a well-defined probabilistic model.

Here, we address this gap by developing a method for curating large-scale natural language inductive inference data. Probabilistic programs can provide a bridge between open-world natural language

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

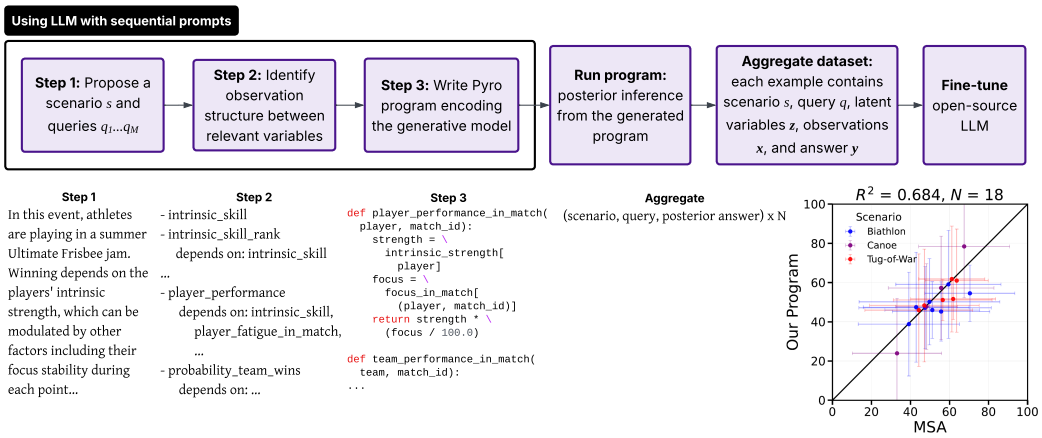


Figure 1: Workflow from data generation to LLM fine-tuning. A sequential prompting process with three steps results in probabilistic programs in Pyro for synthesized natural-language scenarios. These programs are run to generate (scenario, query, posterior answer) tuples. These tuples are used to fine-tune LLMs. The plot in the bottom right shows that the resulting pipeline programs produce aligned results with open-source programs generated by MSA on the same scenarios.

and precise assumptions and supervision. LLMs are useful for proposing diverse scenarios and translating them into structured generative models (Rmus et al., 2025), while probabilistic inference supplies inference results that are more grounded than direct LLM labels. We introduce a pipeline for scalable generation of inductive reasoning datasets for fine-tuning LLMs (Figure 1). Inspired by the Model Synthesis Architecture (MSA; Wong et al., 2025), our approach generates finetuning data from diverse open-world problems with a sequential prompt procedure: (1) synthesizing scenarios from a strong closed-source LLM; (2) generating probabilistic programs to capture these scenarios with an LLM; and (3) running probabilistic inference given these programs (or forward simulation) to calculate posterior distributions over relevant variables. We then use those posterior distributions as targets for fine-tuning.

Our approach allows us to train models on natural-language inductive reasoning problems whose latent structure and uncertainty are explicitly represented. We find that this approach improves estimation accuracy on held-out inferential structures, increases agreement with human judgments, improves performance on several transfer benchmarks, and yields better raw calibration while remaining complementary to post-hoc temperature scaling. These results suggest that data curation with probabilistic program mediation is a promising direction for training LLMs to approximate inductive inference.

In summary, we make the following contributions:

- We study whether large language models can be fine-tuned to improve *inductive reasoning*—inferring uncertain latent properties and predictions from sparse natural-language observations
- We create a dataset with over 10,000 natural-language inductive inference scenarios with probabilistically grounded supervision. To do so, we use a scalable extension of MSA that generates open-world scenarios, synthesizes probabilistic programs for these scenarios, and conducts probabilistic inference to produce targets for fine-tuning.
- We fine-tune LLMs on the resulting distributional labels and evaluate on held-out motifs and human judgments, showing improved estimation accuracy and stronger agreement with human judgments on unseen inferential structures. This procedure outperforms 1) directly calling stronger, closed-source LLM; 2) fine-tuning only on the mean of the same distribution.
- We evaluate external transfer and calibration, showing improvements on two inference-based benchmarks (Renda et al., 2026; Magnusson et al., 2025) and six multiple-choice benchmarks (Sakaguchi et al., 2021; Hendrycks et al., 2021; Lin et al., 2022; Zellers et al., 2019; Clark et al., 2018; Qiu et al., 2026), as well as improved calibration complementary to post-hoc methods such as temperature scaling (Guo et al., 2017) and chain-of-thought (Wang et al., 2023).

2 Related Work

Inductive reasoning in LLMs. Inductive reasoning has long been studied as probabilistic inference over structured generative models (Griffiths et al., 2008; Lake et al., 2015), and a growing body of work asks whether LLMs can perform such inferences. Wong et al. (2023) explore rational meaning construction, in which an LLM translates natural language into probabilistic programs in the language-of-thought tradition (Tenenbaum et al., 2011; Goodman et al., 2014). Wang et al. (2024) formulate inductive reasoning as hypothesis search, in which an LLM proposes natural-language hypotheses and translates them into deterministic Python programs that are verified against examples. Yang et al. (2024b) evaluate LLMs as inductive reasoners over natural language facts and rules.

Probabilistic programs and LLMs. Several recent works combine LLMs with probabilistic programs at inference time. Wu & Goodman (2022) train neural amortized inference networks for probabilistic programs. The Model Synthesis Architecture (MSA) proposed by Wong et al. (2025) addresses open-world inductive reasoning by using an LLM to synthesize a task-specific probabilistic model written in WebPPL (Goodman & Stuhlmüller, 2014), given a prompt with a reasoning scenario specified in natural language. Given sparse observations $\boldsymbol{x}$, the goal is to answer a query $\boldsymbol{y}$ by inferring latent variables $\boldsymbol{z}$. A probabilistic model specifies the distribution,

$$p_\theta(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) = p_\theta(\boldsymbol{z})\, p_\theta(\boldsymbol{x} \mid \boldsymbol{z})\, p_\theta(\boldsymbol{y} \mid \boldsymbol{z}, \boldsymbol{x}), \tag{1}$$

and inference requires computing posterior quantities such as,

$$p_\theta(\boldsymbol{z} \mid \boldsymbol{x}) \propto p_\theta(\boldsymbol{z})\, p_\theta(\boldsymbol{x} \mid \boldsymbol{z}) \tag{2}$$

$$p_\theta(\boldsymbol{y} \mid \boldsymbol{x}) = \int p_\theta(\boldsymbol{y} \mid \boldsymbol{z}, \boldsymbol{x})\, p_\theta(\boldsymbol{z} \mid \boldsymbol{x})\, d\boldsymbol{z}. \tag{3}$$

Given a scenario, a series of prompts leads an LLM to represent the scenario with a probabilistic program. The LLM first identifies relevant latent factors, observations, and query variables, then writes a probabilistic program in WebPPL encoding their generative relationships. The probabilistic program is run to obtain posterior estimates over $\boldsymbol{z}$ and $\boldsymbol{y}$. Thus, the LLM supplies open-world knowledge and the probabilistic program formalizes the precise meaning, facilitating coherent posterior inference.

Fine-tuning LLMs with probabilistic models. A line of work suggests that autoregressive modeling can support implicit Bayesian inference (Xie et al., 2022; Zhang et al., 2025), motivating the use of probabilistic models as training signals for LLMs. Recent work has begun to explore this direction. Hu et al. (2024) fine-tune LLMs via GFlowNets to sample from intractable posteriors over latent chains of thought. Hollmann et al. (2025) use probabilistic models to train transformers to fill in missing values in tabular data. Qiu et al. (2026) fine-tune LLMs to mimic a hand-coded Bayesian assistant. Our approach extends this significantly: it synthesizes a distinct probabilistic program for each open-world scenario, uses the full posterior as a soft label, and evaluates transfer across heterogeneous inferential motifs and external benchmarks.

Uncertainty calibration in LLMs. Calibration of LLM outputs has been studied through both prompting and post-hoc adjustment. Chhikara (2025) document persistent overconfidence and distractor effects in current LLMs. Tian et al. (2023) establish that models can be prompted to produce well-calibrated verbalized confidences; Shen et al. (2024) propose Thermometer, a post-hoc method that improves calibration across tasks. Post-hoc temperature scaling (Guo et al., 2017) remains the standard reference baseline. These methods supervise or adjust meta-confidence over an existing answer distribution. We take a different approach: rather than tuning meta-confidence, we supervise the answer distribution with the Bayesian posterior. The resulting calibration improvements compose with temperature scaling, which suggests that it shifts the model's raw uncertainty representation rather than rescaling its outputs.

3 Program-based Posterior Training

Our goal is to construct natural-language inductive inference data with probabilistically grounded supervision, and use this data to fine-tune LLMs. We propose the Program-based Posterior Training (PPT) pipeline (Figure 1). The data-generation stage proceeds in three steps. First, it uses an LLM to generate diverse natural-language scenarios and queries. Second, an LLM is called to translate each

Table 1: Dataset coverage: count of scenarios and queries.

	Total	Sports	Healthcare	General
Scenarios	14,912	4,866	5,013	5,033
Queries	59,633	19,451	20,050	20,132

scenario into a probabilistic program in Pyro (Bingham et al., 2019). Third, probabilistic inference or forward simulation is run in the resulting program to produce supervised training targets. We then fine-tune open-source LLMs on these scenario–query–answer examples.

3.1 Data generation

Our data-generation pipeline (Figure 1) builds on MSA (Wong et al., 2025), although it uses MSA as a scalable data curation mechanism rather than as an inference-time cognitive model. In the original MSA setting, human-written scenarios are translated into WebPPL programs and evaluated against human judgments. We extend this procedure in two ways. First, the LLM is also used to generate the natural-language scenarios themselves, allowing the pipeline to produce many scenario–query pairs across domains. Second, we translate the synthesized probabilistic programs into Pyro, which allows us to use Python-based probabilistic programming and PyTorch infrastructure. Modern LLMs are generally stronger at writing and editing Python code (Twist et al., 2025) than WebPPL code, which improves the reliability and expressivity of program synthesis and debugging.

Each example consists of a natural-language scenario s, a query q, latent variables $\boldsymbol{z}$, observations $\boldsymbol{x}$, and an answer variable $\boldsymbol{y}$. The scenario describes an open-world inductive setting with observed and latent properties (which are probed in the queries). Figure 4 in the Appendix shows an example scenario and inferred posteriors on the queries. The LLM is sequentially prompted with in-context examples to:

1. Propose a scenario s and queries $q_1, ..., q_M$,
2. Identify the relevant latent variables $\boldsymbol{z}$ and observation structure $p_\theta(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y})$,
3. Write a Pyro program encoding the corresponding generative model.

The program defines the joint distribution $p_\theta(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y})$ from Equation 1. To answer the queries in scenarios, the program is run to conduct posterior inference using MCMC or Rejection Sampling, giving the posterior of the quantity being queried given observations, $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ from Equation 3.

Table 1 counts scenarios and queries with posterior answers by domain. The domains we choose are *Sports* (that MSA also uses), *Healthcare*, *General*. These domains use the identical pipeline, except each domain uses a separate prompt in the Step 1 scenario generation stage. They also share the structure in terms of background, conditions (between 3 and 4), and queries on an entity-intrinsic property, an observation-episodic variable, and two group comparisons. For specific scenario examples, see Figure 4 in the Appendix.

Data validation. As a sanity check, we compared the Pyro programs we generated with the publicly released MSA programs on matched scenarios and queries. We found that the posterior estimates were in strong alignment across scenarios (Figure 1 bottom right). Additionally, we compare programs generated using our pipeline with ground truth programs on ForestDB (Goodman & Stuhlmüller, 2014), testbed datasets used to verify probabilistic programs (Appendix Section A.1). Finally, our empirical results will also utilize a small-set human response, which shows strong alignments.

3.2 Fine-Tuning Format

Posterior sampling. We use two types of supervision from the synthesized programs. The first type is based on *posterior inference*. The posterior $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ is acquired by running the Pyro program described above. It then becomes training labels for the answer to the query in the scenario in one of two ways: 1) a discretized posterior distribution; 2) a single representative answer, which is the posterior mean. In 1), the discretized posterior is realized as probabilities on answer-tokens and treated as the fine-tuning target for the LLM cross-entropy loss.

Specifically, let the posterior $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ have a finite domain, and let $\{b_0, ..., b_K\}$ be boundaries of this domain evenly partitioned into K bins. We define p_k as the posterior probability over bin k: $p_k = \int_{b_k}^{b_{k+1}} p_\theta(\boldsymbol{y}|\boldsymbol{x})d\boldsymbol{y}$. Then, the LLM cross-entropy loss is:

$$\mathcal{L}_{\text{Dist}} = -\sum_{k=0}^{K-1} p_k \log \hat{p}_k,$$

where $\hat{p}_k$ is the LLM-predicted probability for "the token of p_k". In practice, we scale dataset query answers to the 0 to 100 range, and p_k represents the density around an integer, say 36. Then $\hat{p}_k$ refers to LLM-predicted softmaxed logit on the token of "36".

We also explore traditional fine-tuning: training on the posterior mean with probability 1, so the LLM performs regular next-token prediction for the posterior mean rounded to the nearest integer.

Forward sampling. As an alternative to posterior sampling, a supervision can also use *forward sampling*. Instead of conditioning on observations and running posterior inference, we sample latent variables and outcomes directly from the generative model. Forward sampling is noisier because each sampled example reflects one draw from the model rather than an inferred posterior. However, it is cheaper: it avoids iterative posterior inference and can generate additional scenario–query–answer pairs with fewer inference-time costs, leading to hundreds of times more datapoints generated. We therefore treat forward-sampled data as an alternative supervision source.

3.3 Data splits and held-out motifs

To evaluate generalization beyond repeated surface forms, we split data by *motifs* rather than only by individual examples. A motif denotes an underlying inferential structure, implemented as patterns in the observation process. For example, one motif is "one person consistently wins but loses one match". Our motifs are inspired by MSA (Wong et al., 2025), detailed in Appendix Section A.2. Training and validation examples may share broad domains, but held-out motif evaluation requires the model to generalize to new latent structures. This is stricter than a random split over paraphrases or entities, because the model cannot solve the evaluation set by memorizing a fixed scenario template.

4 Empirical Evaluations

We evaluate whether PPT improves inductive reasoning in LLMs by testing agreement with Pyro posteriors on held-out scenarios, comparisons with strong base LLMs on these scenarios, agreement with human judgments, and transfer/calibration on external benchmarks.

Implementational details. We use low-rank adaptation (LoRA; Hu et al., 2022) to fine-tune Llama-3-8B-Instruct and Qwen-2-7B-Instruct (Dubey et al., 2024; Yang et al., 2024a). For more details, see Appendix Section A.2.[1]

4.1 Evaluating fine-tuned LLMs against MSA and human labels

4.1.1 In-domain: sports scenarios

We first evaluate models on sports-domain scenarios similar to those that were used to evaluate MSA. These scenarios describe novel sports competitions with sparse observations about athletes or teams and query latent properties or future outcomes. Our validation split uses held-out scenarios with unseen motifs, so evaluation requires generalization to new inferential structures rather than memorization of templates.

Fine-tuned models represent uncertainty structures. Figure 4 shows a representative held-out scenario. We compare the answer distributions produced by the base LLM, the LLM fine-tuned by PPT, and the Pyro program. The base model produces uncalibrated responses, often placing probability mass in uneven and idiosyncratic ways (e.g., on multiples of ten; McCoy et al. 2023), while the fine-tuned model produces a posterior closer to one computed with probabilistic inference.

[1] Code is available at `temporarily-anonymized-for-review`.

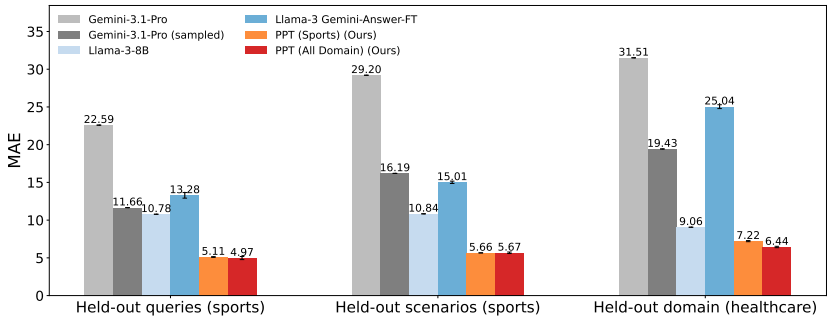


Figure 2: Mean absolute error (MAE) where colors indicate different models. Axis markers indicate the evaluation data: Held-out scenarios have unseen motifs, and held-out domain (healthcare) is where PPT (Sports) has not trained on but PPT (All Domain) has seen. For open-source models, we take the mean response across integers 0 to 100. For closed-source models, we use greedy sampling (GEMINI-3.1-PRO) or average across 100 random seeds with temperature = 1 (GEMINI-3.1-PRO sampled). LLAMA-3-GEMINI-Answer-FT is LLAMA-3 trained on targets generated by GEMINI rather than by probabilistic programs. While seen scenarios are easiest for fine-tuned models, they generalize to new motifs and new domains, significantly outperforming the base LLM. Diversifying training domains gives a further improvement on LLM performance (All-Domain vs Sports).

We further verify whether the LLM constructs a consistent model behind scenarios by checking several similar scenarios with known progressively increasing answers on one query. We manually edit one scenario by performing one action at a time (flip a win or loss or to swap two players), creating 8 scenarios labeled S1-S8, where a player named Jamal progressively goes from weaker to stronger. We found that all methods correctly show a progressive increase in its mean estimate of Jamal's strength. However, the base LLM plateaus more than the PPT-LLM, which continues to increase by 1-2% each step for scenarios S7 and S8 and, as a result, maintains closer estimates with the program; see Figure 5 in the Appendix Section A.1.

Finally, Figure 2 shows the overall quantitative comparison on large-scale held-out scenarios from held-out domains. LLM fine-tuned with PPT (red and orange) has substantially lower mean-absolute error on its average answer compared to the base LLM.

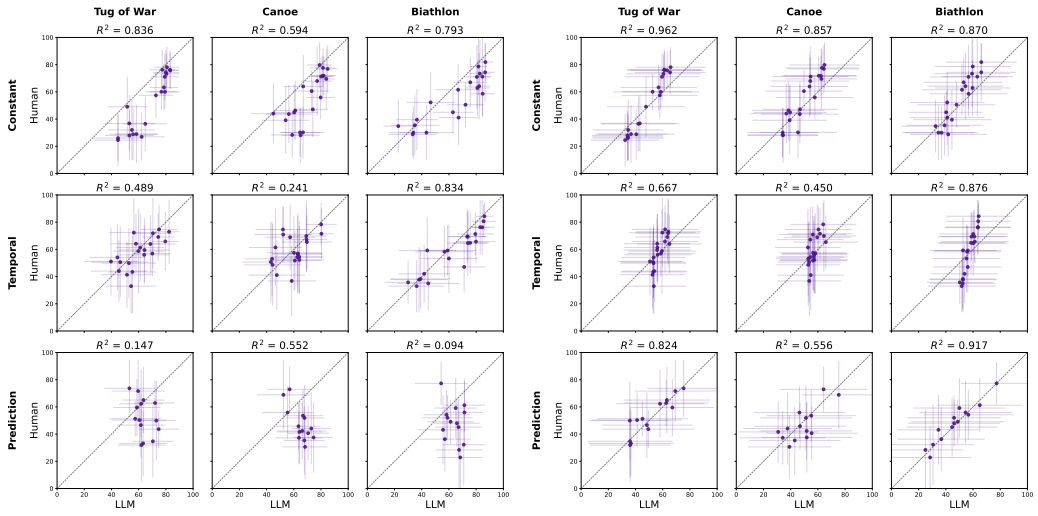
Strong closed-source LLM also fails to answer probabilistic queries well. These problems are also challenging to strong closed-source models like GEMINI-3.1-PRO (Figure 2), which, surprisingly, gives worse performance than LLAMA-3-8B. Each model's mean response over the possible answer domain is more accurate than its argmax response. For GEMINI, its mean response is approximated by averaging over 100 samples.

Fine-tuned models match human results more closely. Figure 3 shows that our approach substantially improves alignment with human judgments across held-out sports scenarios. Each point represents a model response and a human response, mean estimates with standard deviation. After fine-tuning, the model's responses move closer to the diagonal, implying lower error and stronger correlation. On an aggregate level, we find that LLM fine-tuned with PPT (0.775) had the highest overall correlation (R^2) with humans compared to base LLM (0.509) and MSA-generated program (0.769), respectively. This provides an external verification of our pipeline and the fine-tuned LLMs.

4.1.2 OUT-OF-DOMAIN: HEALTHCARE SCENARIOS

We then test whether the gains from our approach transfer beyond the domain used for fine-tuning. We evaluate on healthcare-domain scenarios generated by the same pipeline, where the model must infer latent patient- or treatment-relevant quantities from sparse observations (for an example, see Appendix Section A.2). This provides an out-of-domain test for the sports-fine-tuned model, since the surface domain, entities, and scenario semantics differ from the sports training data.

Fine-tuning reduces MAE. Figure 2 reports mean absolute error (MAE) on in-domain and out-of-domain evaluation sets. Sports-only and all-domain fine-tuning using generated targets generated using probabilistic programs substantially reduce MAE on all sets.



(a) Base pretrained LLM.

(b) Fine-tuned LLM.

Figure 3: Per-query mean response and standard-deviation between humans and LLMs on held-out sports scenarios. Fine-tuned LLMs align much more closely with human responses. The overall correlation R^2 with humans for base LLM, fine-tuned LLM, and MSA results from Wong et al. (2025) are: 0.509, 0.775, 0.769, showing that both our fine-tuned LLMs and original probabilistic programs achieve decent human alignment.

The healthcare validation set provides a stronger test of domain transfer. The sports-fine-tuned model improves over the base model despite not being trained on healthcare scenarios, suggesting that our approach teaches some domain-general inductive behavior.

4.2 Fine-tuned LLMs transfer to common external benchmarks

We finally evaluate whether training on data generated from probabilistic programs results in generalizable inductive and probabilistic reasoning capability. We consider PosteriorDB (PDB; Magnusson et al., 2025) as it is a Bayesian inference benchmark on problems with different inference structures from our training ones; and OpenEstimate (OE; Renda et al., 2026) as it measures LLM's prior elicitation. We evaluate on the Bayesian Teaching dataset (BT; Qiu et al., 2026) since it features multiple-choice problems that involve uncertainty and an underlying optimal Bayesian model. Finally, for a wider performance and calibration evaluation, we use MMLU (Hendrycks et al., 2021), TruthfulQA (Lin et al., 2022), HellaSwag (Zellers et al., 2019), ARC-Challenge (Clark et al., 2018), and Winogrande (Sakaguchi et al., 2021), which involve challenging multiple-choice problems that are not purely deterministic.

Metrics. We use accuracy or mean-absolute error (MAE) for a simple and intuitive performance evaluation, negative log-likelihood (NLL) for how well the LLM models the ground truth answer, and expected calibration error (ECE) to measure model calibration.

Baselines. We use the base LLM (Llama3, Qwen2), Verbalized LLM (Tian et al., 2023), temperature scaling (TS; Guo et al., 2017), and chain-of-thought with self-consistency (CoT + SC; Wang et al., 2023). Verbalized LLM is the base LLM that explicitly outputs confidence level on multiple choices in its response. This baseline tests whether the base LLM already exhibits reliable and extractable confidence levels beyond token-level probabilities. TS is a post-hoc calibration method that tunes the temperature parameter for each dataset. CoT + SC enables reasoning and multiple samples, designed to serve as a stronger baseline.

Our approach. PPT (Distribution), where fine-tuning was performed on full posterior probabilistic labels, is our primary method. We also consider a variant, called PPT (Mean), that fine-tunes on the same dataset but targets only the mean of the distribution. This ablates the distributional target and tests the utility of such a target. In addition, we implement *forward sampling* to generate distributional

Table 2: Transfer to external benchmarks on LLAMA3-8B and QWEN2-7B. NLL/ECE include a temperature-scaling (TS) block. Best per column in **bold** (within each block).

(a) NLL across benchmarks (lower is better).

Method: LLAMA-3	OE	PDB	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	7.09	4.91	2.91	1.69	3.49	1.26	0.98	1.15
PPT (Dist.)	**4.53** ±0.02	**4.54** ±0.02	**1.25** ±0.01	**1.01** ±0.01	**1.78** ±0.02	**0.88** ±0.02	**0.61** ±0.01	**0.80** ±0.01
PPT (Mean)	5.08 ±0.02	**4.55** ±0.00	1.86 ±0.00	1.27 ±0.01	2.36 ±0.02	0.98 ±0.01	0.74 ±0.00	0.91 ±0.01
Base + CoT + SC	12.37	5.94	2.63	1.52	2.46	1.33	0.58	1.39
PPT (Dist.) + CoT + SC	**4.86** ±0.01	**4.50** ±0.01	**1.33** ±0.05	**0.98** ±0.01	**1.50** ±0.05	**0.99** ±0.01	**0.47** ±0.01	**0.86** ±0.02
Base + TS	—	—	—	1.01	1.76	0.95	0.62	0.84
PPT (Dist.) + TS	—	—	**1.08** ±0.00	**0.96** ±0.01	**1.62** ±0.03	**0.87** ±0.02	**0.60** ±0.01	**0.80** ±0.00
Method: QWEN-2	OE	PDB	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	6.91	5.67	2.56	1.96	4.81	0.84	0.90	2.18
PPT (Dist.)	**3.86** ±0.00	**4.88** ±0.03	**1.40** ±0.01	**1.12** ±0.01	**3.04** ±0.03	**0.60** ±0.00	**0.51** ±0.01	**1.23** ±0.01
PPT (Mean)	4.59 ±0.08	**4.82** ±0.04	2.06 ±0.02	1.67 ±0.01	4.48 ±0.06	0.75 ±0.01	0.76 ±0.01	1.88 ±0.02
Base + CoT + SC	10.65	6.40	2.80	1.61	2.90	1.27	0.56	2.04
PPT (Dist.) + CoT + SC	**5.02** ±0.03	**5.12** ±0.01	**1.65** ±0.04	**1.13** ±0.01	**1.99** ±0.01	**0.82** ±0.01	**0.44** ±0.01	**1.70** ±0.01
Base + TS	—	—	—	1.01	4.81	**0.58**	0.49	1.03
PPT (Dist.) + TS	—	—	**1.09** ±0.00	**0.94** ±0.01	**3.16** ±0.04	0.60 ±0.01	**0.47** ±0.01	**0.96** ±0.01

(b) Expected Calibration Error (15-bin) (lower is better).

Method: LLAMA-3	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	0.467	0.281	0.462	0.196	0.158	0.289
Verbalized	0.255	**0.084**	0.470	0.141	0.211	0.284
PPT (Dist.)	**0.218** ±0.002	0.112 ±0.006	**0.275** ±0.009	**0.064** ±0.010	**0.058** ±0.010	**0.094** ±0.009
PPT (Mean)	0.380 ±0.000	0.216 ±0.002	0.377 ±0.002	**0.060** ±0.003	0.118 ±0.005	0.221 ±0.007
Base + CoT + SC	0.301	0.175	0.270	0.136	**0.062**	0.236
PPT (Dist.) + CoT + SC	**0.227** ±0.015	**0.112** ±0.005	**0.209** ±0.009	**0.126** ±0.005	**0.066** ±0.005	**0.160** ±0.006
Base + TS	—	0.100	0.245	**0.023**	0.062	0.126
PPT (Dist.) + TS	**0.025** ±0.002	**0.069** ±0.004	**0.223** ±0.014	0.032 ±0.003	**0.051** ±0.010	**0.080** ±0.002
Method: QWEN-2	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	0.464	0.273	0.475	0.134	0.135	0.337
PPT (Dist.)	**0.278** ±0.005	**0.180** ±0.001	**0.381** ±0.002	**0.060** ±0.001	**0.092** ±0.001	**0.246** ±0.003
PPT (Mean)	0.415 ±0.002	0.254 ±0.001	0.475 ±0.005	0.119 ±0.002	0.128 ±0.004	0.324 ±0.001
Base + CoT + SC	0.319	0.185	0.265	0.158	**0.076**	**0.272**
PPT (Dist.) + CoT + SC	**0.271** ±0.007	**0.158** ±0.002	**0.238** ±0.005	**0.085** ±0.001	**0.071** ±0.006	**0.272** ±0.002
Base + TS	—	0.133	0.475	0.035	0.062	0.128
PPT (Dist.) + TS	**0.056** ±0.004	**0.095** ±0.000	**0.383** ±0.003	**0.015** ±0.002	**0.050** ±0.006	**0.110** ±0.001

(c) Accuracy across benchmarks; OpenEstimate (OE) and PosteriorDB (PDB) reported as MAE (lower is better).

Method: LLAMA-3	OE (MAE) ↓	PDB (MAE) ↓	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	29.0	26.4	37.8%	60.0%	42.2%	63.6%	78.6%	54.5%
Verbalized	—	—	35.3%	42.8%	16.0%	43.1%	54.8%	50.7%
PPT (Dist.)	**23.3** ±0.3	**20.6** ±0.2	**39.3%** ±0.0%	**61.2%** ±0.3%	**43.2%** ±0.7%	**66.8%** ±1.2%	**79.6%** ±0.5%	**58.1%** ±0.4%
PPT (Mean)	23.8 ±0.1	21.9 ±0.1	36.8% ±0.0%	59.5% ±0.2%	41.7% ±0.1%	61.6% ±0.4%	77.4% ±0.2%	54.2% ±0.2%
Base + CoT + SC	34.2	25.6	**46.8%**	65.2%	56.0%	**72.0%**	**85.6%**	59.2%
PPT (Dist.) + CoT + SC	**24.6** ±0.4	**20.6** ±0.5	**46.6%** ±0.6%	**67.5%** ±0.4%	**57.6%** ±1.1%	70.2% ±0.3%	84.3% ±0.2%	**66.8%** ±0.4%
Method: QWEN-2	OE (MAE) ↓	PDB (MAE) ↓	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	14.2	25.4	**39.6%**	65.1%	**41.1%**	**80.1%**	83.6%	61.3%
PPT (Dist.)	**13.4** ±0.1	**22.4** ±0.1	38.9% ±0.3%	**65.5%** ±0.1%	40.4% ±0.1%	79.4% ±0.1%	**84.1%** ±0.1%	**63.8%** ±0.4%
PPT (Mean)	14.7 ±0.2	23.6 ±0.3	**39.4%** ±0.2%	64.9% ±0.2%	38.8% ±0.5%	79.3% ±0.2%	83.7% ±0.2%	61.1% ±0.1%
Base + CoT + SC	**24.8**	**21.0**	**42.4%**	**68.0%**	**51.8%**	**79.8%**	**85.3%**	**64.6%**
PPT (Dist.) + CoT + SC	25.2 ±0.1	**21.1** ±0.3	41.1% ±0.5%	67.3% ±0.2%	**51.6%** ±0.7%	78.9% ±0.5%	84.7% ±0.3%	63.5% ±0.1%

targets, instead of full posterior inference, as it allows creation of orders-of-magnitude, bigger datasets due to its inherent scalability.

PPT transfers to external benchmarks. *Calibration and NLL.* Table 2 shows that PPT (Distribution) substantially reduces NLL and ECE relative to the base model across all reported benchmarks for both LLAMA and QWEN. This suggests that probabilistic-program-mediated fine-tuning improves the model's raw uncertainty estimates.

Temperature scaling (TS) remains a strong post-hoc calibration baseline. In several cases, Base+TS closes much of the NLL and ECE gap between the base and fine-tuned models. However, applying temperature scaling on top of PPT (Distribution) consistently yields further improvements, with lower NLL and ECE on most benchmarks. This indicates that PPT and post-hoc calibration are complementary: temperature scaling adjusts confidence on a target distribution, while PPT changes the underlying model to produce better raw uncertainty estimates.

Accuracy and MAE. PPT also improves accuracy on these benchmarks, especially on PDB and OpenEstimate. This is consistent with the goal of improving quantitative inductive estimation: relatively simple inference structures and prior elicitation. Although the gains are modest for other benchmarks, our results suggest that improved prior and calibration can contribute to not only improving answer distribution, but also correctly flipping the argmax answer.

CoT + SC. PPT's stronger calibration and comparable accuracy extend toward CoT + SC. Because including chain-of-thought gives the model a decisive advantage, we test both base model and PPT model with CoT + SC. This result shows that PPT is amenable to CoT, with CoT offering a similar boost to accuracy in base and PPT model.

Scaling across model sizes. Finally, Table 3 in the Appendix shows that results that hold for smaller models scale across model-sizes. It compares QWEN2 models of 7B, 14B, and 72B fine-tuned with PPT with their base-instruct versions.

Fine-tuning without distributional target reduces improvement. For both LLAMA and QWEN, PPT (Distribution) and PPT (Mean) use identical training data except for the target, but PPT (Distribution) outperforms on every metric. PPT (Distribution) also attains higher accuracy and stronger post-TS calibration than models fine-tuned with forward sampling data (Table 4 in the Appendix). This suggests that distributional information plays an important role in the improvements observed in external benchmark transfers.

5 DISCUSSION

Our results suggest a different way to think about reasoning post-training. Standard reasoning supervision is often framed around problems with single verifiable answers, but inductive reasoning requires models to maintain uncertainty over latent explanations and outcomes. PPT uses probabilistic programs to construct such a supervision at scale. Our results demonstrate that PPT fine-tuning allows models to internalize uncertainty, generalizing to entirely unseen motifs and domains, while remaining complementary to traditional temperature scaling techniques.

More specifically, on held-out synthetic scenarios, PPT substantially reduces MAE relative to base LLMs and strong closed-source baselines, including settings with unseen motifs and a held-out healthcare domain. With human-labeled sports scenarios as external verification, PPT improves alignment with human judgments, reaching a correlation comparable to the original MSA programs. On external benchmarks, PPT improves OpenEstimate prior-elicitation and PosteriorDB posterior alignment, and consistently reduces NLL and ECE across multiple-choice tasks, with gains that remain complementary to temperature scaling and chain-of-thought. Together, these results suggest that probabilistic programs can serve not only as test-time reasoning tools, but also as scalable supervision sources for training LLMs to form calibrated beliefs.

Limitations and future directions. Our experiments focus on numeric and multiple-choice queries, which allow precise measurement of posterior distributions, MAE, NLL, and ECE. This does not fully cover open-ended inductive reasoning. Extending PPT to more naturalistic settings with intermediate reasoning steps and/or interactions is an important direction.

Conclusion. Our results show that using LLMs to generate probabilistic programs that are in turn used to train LLMs on natural-language inductive inference problems is an effective way of improving the capabilities of these models. Fine-tuning on data generated from probabilistic programs improves held-out probabilistic estimation, alignment with human judgments, transfer to external estimation and multiple-choice benchmarks, and raw calibration. These results suggest that probabilistic programs can serve not only as test-time reasoning tools, but also as scalable supervision sources for training LLMs to form calibrated beliefs.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic data sets, implement methods, clean and reformat dataset, support qualitative and thematic data analysis (the last just through converting data into figures using AI coding tool). We also note that the problem of how to generate synthetic datasets with LLM is itself a topic of this paper.

We have not used generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, interpret results.

We have reviewed all AI-assisted work. For example, LLM-generated code was verified and tested for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We identify no specific ethical concerns arising from this work. Our comparisons with human judgments use results from previously published studies; we conducted no new human-subject experiments.

REPRODUCIBILITY STATEMENT

We describe the data-generation pipeline, training procedures, and experimental settings in the paper and appendix. The accompanying code repository will be open-sourced after the review process. A complete pipeline is also provided in the supplementary materials.

REFERENCES

Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep Universal Probabilistic Programming. *Journal of Machine Learning Research*, 20:1–6, 2019.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006. ISBN 0387310738.

David M. Blei. Build, compute, critique, repeat: Data analysis with latent variable models. In *Annual Review of Statistics and Its Application*, 2014.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Prateek Chhikara. Mind the confidence gap: Overconfidence, calibration, and distractor effects in large language models. *Transactions on Machine Learning Research (TMLR)*, 2025. URL https://openreview.net/forum?id=lyaHnHDdZl.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. 2024.

Noah D. Goodman and Andreas Stuhlmüller. DIPPL: The design and implementation of probabilistic programming languages. 2014. URL http://dippl.org. Accessed: September 2026.

Noah D. Goodman, Joshua B. Tenenbaum, and Tobias Gerstenberg. Concepts in a probabilistic language of thought. 2014.

Thomas L Griffiths, Charles Kemp, and Joshua B Tenenbaum. Bayesian models of cognition. In *Cambridge Handbook of Computational Cognitive Sciences*. Cambridge University Press, 2008.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1321–1330, 2017.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Edward J Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. Amortizing intractable inference in large language models. In *International Conference on Learning Representations*, 2024.

Akshay K Jagadish, Julian Coda-Forno, Mirko Thalmann, Eric Schulz, and Marcel Binz. Human-like category learning by injecting ecological priors from large language models into neural networks. In *Forty-first International Conference on Machine Learning*, 2024.

Akshay K Jagadish, Mirko Thalmann, Julian Coda-Forno, Marcel Binz, and Eric Schulz. Meta-learning ecological priors from large language models explains human learning and decision making. *arXiv preprint arXiv:2509.00116*, 2025.

Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Måns Magnusson, Jakob Torgander, Paul-Christian Bürkner, Lu Zhang, Bob Carpenter, and Aki Vehtari. posteriordb: Testing, benchmarking and developing bayesian inference algorithms. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 1198–1206. PMLR, 2025.

R Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L Griffiths. Embers of autoregression: Understanding large language models through the problem they are trained to solve. *arXiv preprint arXiv:2309.13638*, 2023.

Linlu Qiu, Fei Sha, Kelsey Allen, Yoon Kim, Tal Linzen, and Sjoerd van Steenkiste. Bayesian teaching enables probabilistic reasoning in large language models. *Nature Communications*, 17(1): 1238, 2026.

Alana Renda, Jillian Ross, and Jacob Andreas. Openestimate: Evaluating llms on reasoning under uncertainty with real-world data. In *International Conference on Learning Representations*, volume 2026, pp. 2350–2368, 2026.

Milena Rmus, Akshay Kumar Jagadish, Marvin Mathony, Tobias Ludwig, and Eric Schulz. Generating computational cognitive models using large language models. *Advances in Neural Information Processing Systems*, 38:87796–87833, 2025.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: an adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, August 2021.

Maohao Shen, Subhro Das, Kristjan Greenewald, Prasanna Sattigeri, Gregory W. Wornell, and Soumya Ghosh. Thermometer: Towards universal calibration for large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 44687–44711, 2024.

Joshua B Tenenbaum, Charles Kemp, Thomas L Griffiths, and Noah D Goodman. How to grow a mind: statistics, structure, and abstraction. *Science*, 331(6022):1279–1285, 2011.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Lukas Twist, Jie M Zhang, Mark Harman, Don Syme, Joost Noppen, and Detlef Nauck. LLMs love Python: A study of LLMs' bias for programming languages and libraries. *arXiv preprint arXiv:2503.17181*, 2025.

Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah Goodman. Hypothesis search: Inductive reasoning with language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, Red Hook, NY, USA, 2022. ISBN 9781713871088.

Lionel Wong, Gabriel Grand, Alexander K. Lew, Noah D. Goodman, Vikash K. Mansinghka, Jacob Andreas, and Joshua B. Tenenbaum. From word models to world models: Translating from natural language to the probabilistic language of thought, 2023.

Lionel Wong, Katherine M Collins, Lance Ying, Cedegao E Zhang, Adrian Weller, Tobias Gerstenberg, Timothy O'Donnell, Alexander K Lew, Jacob D Andreas, Joshua B Tenenbaum, and Tyler Brooke-Wilson. Modeling open-world cognition as on-demand synthesis of probabilistic models. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 47, 2025.

Mike Wu and Noah Goodman. Foundation posteriors for approximate probabilistic inference. In *Advances in Neural Information Processing Systems*, 2022.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit Bayesian inference. In *International Conference on Learning Representations*, 2022.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024a.

Zonglin Yang, Li Dong, Xinya Du, Hao Cheng, Erik Cambria, Xiaodong Liu, Jianfeng Gao, and Furu Wei. Language models as inductive reasoners. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 209–225. Association for Computational Linguistics, 2024b.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

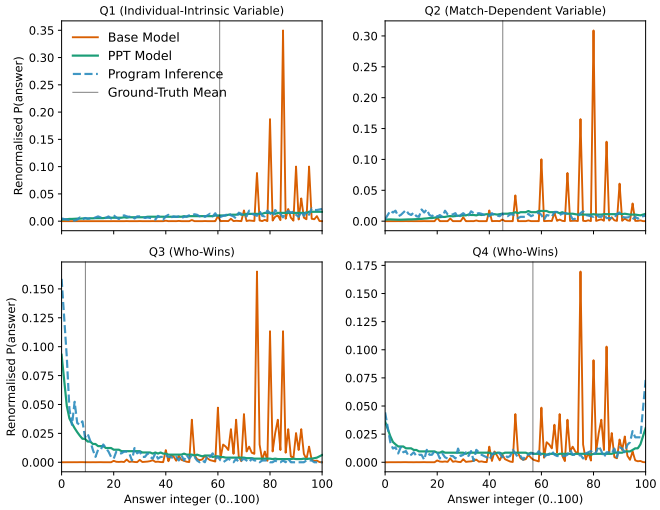
Liyi Zhang, Michael Y. Li, R. Thomas McCoy, Theodore Sumers, Jian-Qiao Zhu, and Thomas L. Griffiths. What should embeddings embed? Autoregressive models represent latent generating distributions. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. Featured Certification.

BACKGROUND
In this event, lifters are competing in a powerlifting championship. The winner of each flight is the one with the highest combined total. Performance is driven by an athlete's intrinsic strength and bracing ability, though it is also subject to change based on their focus stability and confidence on the day.
Athletes compete either individually or in teams of four.
All flights take place on the same day.

CONDITIONS
In the first flight, Jamal beat Tariq.
In the second flight, Jamal, Malik, Omar, and Zayd beat Tariq, Amir, Bilal, and Kareem.
In the third flight, Jamal, Amir, Bilal, and Kareem lost to Tariq, Malik, Omar, and Zayd.
In the fourth flight, Jamal, Malik, Bilal, and Kareem beat Tariq, Amir, Omar, and Zayd.

QUERIES
Query 1: Out of 100 random athletes, where do you think Jamal ranks in terms of intrinsic strength?
Query 2: On a percentage scale from 0 to 100%, how high do you think Jamal's focus stability was during the third flight?
Query 3: In a new flight later this same day between Jamal, Malik, Omar, and Zayd (Team 1) and Tariq, Amir, Bilal, and Kareem (Team 2), who would win and by how much?
Query 4: In a new flight later this same day between Jamal, Amir, Omar, and Kareem (Team 1) and Tariq, Malik, Bilal, and Zayd (Team 2), who would win and by how much?

(a) Scenario and query text.



(b) LLM posteriors and program-inferred posteriors (ground truth).

Figure 4: Effects of Program-based Posterior Training. (a) Example held-out scenario and queries. (b) Posteriors over answer-token probabilities from the base LLM and LLM fine-tuned on data from probabilistic programs (LLAMA3-8B-INSTRUCT). Blue dashed line is programmatic inference result, treated as ground truth. A text appended before the scenario (see Appendix Section A.2) instructs the LLM that Q3 and Q4 are estimated on 0-100 scale, where lower number means Team 1 more likely wins. LLM posteriors are more similar to those of the program.

A APPENDIX

A.1 ADDITIONAL EXPERIMENTS

A.1.1 PIPELINE VALIDATION

We take reference probabilistic programs from ForestDB (Goodman & Stuhlmüller, 2014) and re-derive their quantities through our pipeline. There are six tasks from the dataset: three Lee & Wagenmakers binomial problems, a Gaussian mean/SD problem, a signal-detection model, and Monty Hall. Runs from our pipeline generated programs deviate from the reference program runs by $1.6\%0.5\%$ on posterior means and $3.0\%1.4\%$ on posterior standard deviations (SE across quantities).

There are a few interesting deviations: In half the tasks the generated program chose a different prior (such as Normal$(0, 31.6)$ vs. Normal$(0, 100)$ or a different representation (Monty Hall: discrete enumeration vs. a continuous relaxation discretized at query time), but our programs' inference draws still landed close to the reference posteriors.

A.1.2 FINE-TUNING

Progressively changing scenarios We manually edit the scenario by using one action at a time (flip a win or loss or to swap two players), creating 8 scenarios S1 to S8 where player Jamal progressively goes from weaker to stronger. All methods in general correctly show a progressive increase in its mean estimate on Jamal's strength. However, base LLM plateaus more than the PPT-LLM, which still increases by 1-2 % each step for the S7 and S8, as well as maintaining closer estimates with the program (Figure 5).

Scaling across model sizes Table 3 shows that PPT results that hold for smaller models extend across model sizes.

Posterior sampling and forward sampling We compare two approaches we used to generate data for LLM fine-tuning: *posterior sampling* and *forward sampling*. Posterior sampling fine-tunes on a distributional target, whereas forward sampling, by construction, generates massive amounts of data with only a point estimate available for each query.

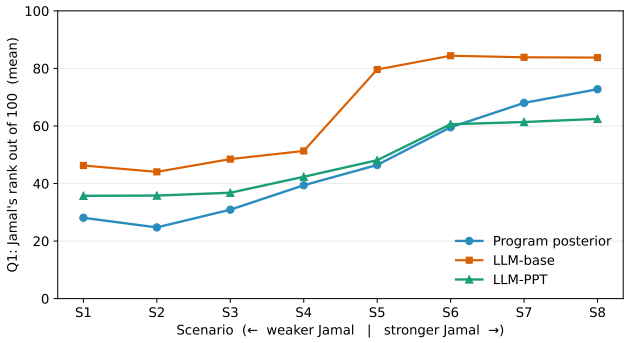


Figure 5: Mean estimates on Jamal's strength on eight scenarios that progressively goes from weaker to stronger Jamal. *All methods in general correctly show a progressive increase in its mean estimate on Jamal's strength. However, base LLM plateaus more than the PPT-LLM, which still increases by 1-2 % each step for the S7 and S8, as well as maintaining closer estimates with the program.*

Table 4 shows that *posterior sampling* attains better accuracy while *forward sampling* attains comparable-to-better raw calibration and negative log-likelihood. Meanwhile, when temperature scaling (TS) is applied, *posterior sampling* gains an edge on these two metrics.

A.2 Implementational details

Example scenario and data Figure 4 shows an example of a scenario, probabilitic program label to its queries, and LLM answers.

Hyperparameters Here we detail the experimental setup used in our experiments. We use PyTorch with Torchtune to fine-tune Llama-3-8B-Instruct and Qwen-2-7B-Instruct. Each experiment uses a single A100 GPU with 40GB memory. All methods use the AdamW optimizer (Loshchilov & Hutter, 2019), batch-size of 2, inner-loops with 5 gradient steps, LoRA adapters with rank $= 8$ following the standard practice, and learning rate is tuned in $[10^{-6}, 10^{-4}]$.

An 'epoch' here is defined as training on 200 batches from the train set, and all models train up to 500 epochs, using an early-stopping of 50 epochs.

Scenario examples Here we show a scenario and query example from each domain: sports, healthcare, general. While most scenarios follow a certain writing format, we also show one free-form scenario.

Here is an instruction block that the LLM sees when responding to each scenario.

LLM Instruction

Answer the query in the scenario and return only an integer wrapped in < and >. For example, <x>. Use 0-100 scale. For a query on individual rank, a higher number means a higher ranking (e.g. 100 means the individual ranks highest in that criterion; 1 is lowest). For a query on which of the two teams wins, a smaller number means the first team more likely wins.

A sports scenario example is given in the main text in Figure 4.

Here are the other scenarios. We also add the program-inferred posterior mean as answers for illustration.

Healthcare Scenario

BACKGROUND
In this endocrine day clinic, patients are monitored during a series of metabolic stress tests. In each episode, the group that responds better depends on the average insulin sensitivity / glucose regulation efficiency and autonomic stability (orthostatic tolerance) of the patients, modulated by missed meals / fasting that episode, acute stress level that episode, and

Table 3: Scaling the backbone: base vs. PPT (Distribution) on QWEN2-7B, QWEN2.5-14B and QWEN2.5-72B (all Instruct), same recipe and evaluation suite as Table 2. PPT rows are mean ± SE over seeds where more than one seed is available.

(a) NLL across benchmarks (lower is better).

Method: QWEN2-7B	OE	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	6.91	2.56	1.96	4.81	0.84	0.90	2.18
PPT (Distribution)	**3.86** ±0.00	**1.40** ±0.01	**1.12** ±0.01	**3.04** ±0.03	**0.60** ±0.00	**0.51** ±0.01	**1.23** ±0.01
Method: QWEN2.5-14B	OE	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	11.08	4.59	2.44	3.27	1.20	0.97	3.49
PPT (Distribution)	**3.86** ±0.06	**1.52** ±0.07	**1.01** ±0.03	**1.20** ±0.09	**0.56** ±0.02	**0.42** ±0.02	**1.34** ±0.11
Method: QWEN2.5-72B	OE	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	7.83	2.46	1.29	1.91	0.57	0.25	1.74
PPT (Distribution)	**3.58**	**1.07**	**0.61**	**0.81**	**0.30**	**0.13**	**0.70**

(b) Expected Calibration Error (15-bin) (lower is better).

Method: QWEN2-7B	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	0.464	0.273	0.475	0.134	0.135	0.337
PPT (Distribution)	**0.278** ±0.005	**0.180** ±0.001	**0.381** ±0.002	**0.060** ±0.001	**0.092** ±0.001	**0.246** ±0.003
Method: QWEN2.5-14B	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	0.423	0.198	0.218	0.100	0.076	0.250
PPT (Distribution)	**0.292** ±0.011	**0.147** ±0.004	**0.156** ±0.007	**0.082** ±0.002	**0.060** ±0.001	**0.198** ±0.005
Method: QWEN2.5-72B	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	0.364	0.142	0.161	0.076	0.033	0.188
PPT (Distribution)	**0.199**	**0.087**	**0.099**	**0.046**	**0.023**	**0.149**

(c) Accuracy across benchmarks; OpenEstimate (OE) reported as MAE (lower is better).

Method: QWEN2-7B	OE (MAE) ↓	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	14.2	**39.6%**	65.1%	**41.1%**	**80.1%**	83.6%	61.3%
PPT (Distribution)	**13.4** ±0.1	38.9% ±0.3%	**65.5%** ±0.1%	40.4% ±0.1%	79.4% ±0.1%	**84.1%** ±0.1%	**63.8%** ±0.4%
Method: QWEN2.5-14B	OE (MAE) ↓	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	18.2	52.3%	**77.3%**	75.9%	**88.9%**	92.0%	73.2%
PPT (Distribution)	**13.2** ±0.5	**52.6%** ±0.1%	**77.4%** ±0.2%	**78.3%** ±0.8%	88.1% ±0.3%	**92.9%** ±0.1%	**74.9%** ±0.5%
Method: QWEN2.5-72B	OE (MAE) ↓	BT	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Base	**9.0**	55.8%	81.9%	79.6%	**90.3%**	96.3%	79.1%
PPT (Distribution)	10.8	**57.6%**	**82.6%**	**80.7%**	90.2%	**96.7%**	**79.7%**

caffeine/alcohol intake that episode. Patients are under the care of either Dr. Smith or Dr. Jones, and all clinical events take place on the same day.

CONDITIONS
Dr. Smith cares for Taylor, Ava, Beth, Cara, Dana, Elsa, Faye, Gina, Hope, Iris, Judy, Kara, Lana, and Mary, whereas Dr. Jones cares for Nina, Opal, Page, Quin, Rosa, Sara, Tara, Uma, Vera, Willa, Xena, Yara, and Zara.
In the first episode, Taylor, Ava, Beth, Cara, Dana, Elsa, and Faye responded better than Gina, Hope, Iris, Judy, Kara, Lana, and Mary.
In the second episode, Taylor, Nina, Opal, Page, Quin, Rosa, and Sara responded better than Tara, Uma, Vera, Willa, Xena, Yara, and Zara.
In the third episode, Taylor, Gina, Hope, Iris, Judy, Kara, and Lana responded better than Nina, Opal, Page, Quin, Rosa, Sara, and Tara.

QUERIES
Query 1: On a percentage scale from 0 to 100%, how strong was missed meals / fasting that episode in the third clinical episode for Gina?

Table 4: Transfer to external benchmarks on LLAMA-3-8B. NLL/ECE include a temperature-scaling (TS) block. Best per column in **bold** (within each block).

(a) NLL across benchmarks (lower is better).

Method	OE	BT	BT-guided	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Posterior (Dist)	**4.53** ±0.02	1.25 ±0.01	1.13 ±0.00	1.01 ±0.01	1.78 ±0.02	**0.88** ±0.02	**0.61** ±0.01	**0.80** ±0.01
Forward (Mean)	4.79 ±0.01	**1.11** ±0.01	**1.08** ±0.00	**0.95** ±0.00	**1.52** ±0.03	1.05 ±0.01	0.67 ±0.01	0.89 ±0.02
Posterior (Dist) + TS	—	**1.08** ±0.00	**1.09** ±0.00	**0.96** ±0.01	**1.62** ±0.03	**0.87** ±0.02	**0.60** ±0.01	**0.80** ±0.00
Forward (Mean) + TS	—	1.09 ±0.00	**1.09** ±0.00	**0.97** ±0.00	1.73 ±0.05	0.93 ±0.01	0.63 ±0.01	**0.80** ±0.00

(b) Expected Calibration Error (15-bin) (lower is better).

Method	BT	BT-guided	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Posterior (Dist)	0.218 ±0.002	0.134 ±0.007	0.112 ±0.006	0.275 ±0.009	**0.064** ±0.010	**0.058** ±0.010	**0.094** ±0.009
Forward (Mean)	**0.091** ±0.015	**0.040** ±0.009	**0.043** ±0.001	**0.105** ±0.003	0.222 ±0.012	0.126 ±0.006	**0.072** ±0.014
Posterior (Dist) + TS	**0.025** ±0.002	**0.020** ±0.003	0.069 ±0.004	**0.223** ±0.014	**0.032** ±0.003	**0.051** ±0.010	**0.080** ±0.002
Forward (Mean) + TS	**0.027** ±0.007	0.038 ±0.007	**0.045** ±0.004	**0.206** ±0.008	0.069 ±0.004	0.066 ±0.003	**0.082** ±0.005

(c) Accuracy across benchmarks; OpenEstimate (OE) reported as MAE (lower is better).

Method	OE (MAE) ↓	BT	BT-guided	MMLU	TruthfulQA	HellaSwag	ARC-C	Winogrande
Posterior (Dist)	**23.3** ±0.3	**39.3%** ±0.0%	37.3% ±0.2%	**61.2%** ±0.3%	**43.2%** ±0.7%	**66.8%** ±1.2%	**79.6%** ±0.5%	**58.1%** ±0.4%
Forward (Mean)	27.7 ±0.0	38.0% ±0.5%	**38.3%** ±0.6%	60.2% ±0.2%	**42.5%** ±0.7%	64.4% ±0.6%	77.3% ±0.4%	55.4% ±0.3%

Answer 1: 73
Query 2: In a new clinical episode later this same day involving Gina, Hope, Iris, Judy, Kara, Lana, and Mary versus Nina, Opal, Page, Quin, Rosa, Sara, and Tara, who would have the better outcome and by how much?
Answer 2: 6
Query 3: In a new clinical episode later this same day involving Taylor, Ava, Beth, Cara, Dana, Elsa, and Faye versus Tara, Uma, Vera, Willa, Xena, Yara, and Zara, who would have the better outcome and by how much?
Answer 3: 5
Query 4: Out of 100 random patients, where do you think Taylor ranks in terms of intrinsic insulin sensitivity / glucose regulation efficiency?
Answer 4: 68

General Domain Scenario

BACKGROUND
In this event, watchmakers are competing in a series of timed challenges to assemble intricate mechanical watches. In each round, the team that achieves the better outcome depends on the average assembly precision of the watchmakers, based on their intrinsic mechanical dexterity modulated by four other factors: lighting quality, tool sharpness, eye fatigue, and ambient humidity. Watchmakers compete in teams of five, and all challenges take place on the same day.

CONDITIONS
In the first challenge, Julian, Clara, Thomas, Elise, and Victor completed the watch assembly more successfully than Marcus, Sophie, Daniel, Chloe, and Felix. In the second challenge, Liam, Nora, Oliver, Maya, and Lucas completed the watch assembly more successfully than Henry, Emma, Wyatt, Grace, and Felix. In the third challenge, Julian, Clara, Thomas, Elise, and Victor completed the watch assembly more successfully than Liam, Nora, Oliver, Maya, and Felix.

QUERIES
Query 1: On a percentage scale from 0 to 100%, how strong was eye fatigue in the third event for Felix?

Answer 1: 66
Query 2: In a new situation later this same day involving Liam, Nora, Oliver, Maya, and Lucas (Team 1) versus Marcus, Sophie, Daniel, Chloe, and Felix (Team 2), who would have the better outcome and by how much?
Answer 2: 30
Query 3: In a new situation later this same day involving Julian, Clara, Thomas, Elise, and Victor (Team 1) versus Henry, Emma, Wyatt, Grace, and Lucas (Team 2), who would have the better outcome and by how much?
Answer 3: 22
Query 4: Out of 100 random watchmakers, where do you think Felix ranks in terms of intrinsic mechanical dexterity?
Answer 4: 34

Free-Form Scenario

Seaside Pickleball Invitational: Coaching Staff Chat Log
Head Coach Martinez: Let's review today's tournament results. How did our fixed doubles pairs look out there?
Asst Coach Davis: It was an interesting day, definitely heavily influenced by the coastal weather. The morning started dead calm and cool. Tariq and Leo played Jin and Carlos first. Tariq and Leo won comfortably. Tariq's baseline drives were incredibly sharp right out of the gate.
Head Coach Martinez: Good start. How about the mid-day matches?
Asst Coach Davis: The sun got absolutely brutal around noon. Tariq and Leo played Emma and Noah. Tariq and Leo won narrowly. To be honest, Leo looked like he was really dragging his feet in the heat and missing easy kitchen volleys that he normally puts away.
Head Coach Martinez: I actually caught the singles exhibition right after lunch. Tariq played Jin 1-on-1. Tariq won by a lot. He was covering the whole court and barely looked like he was breaking a sweat.
Asst Coach Davis: Yeah, individually, Tariq is a machine. But the late afternoon doubles match was a different story. Tariq and Leo went up against Sarah and Maya. By then, the coastal wind was howling, which was completely messing with the lightweight ball on lobs and drops. Plus, it was Tariq and Leo's third doubles match of the day in the sun. Sarah and Maya ended up winning narrowly.
Head Coach Martinez: Tough break, but it makes sense. Sarah and Maya play a very grounded, tactical game that probably holds up well when the wind picks up. Did any other matches happen before sunset?
Asst Coach Davis: Just one evening match, right after the wind completely died down and the temperature dropped back to normal. Jin and Carlos played Emma and Noah. Jin and Carlos won comfortably, looking very coordinated.

***
Post-Tournament Analytical Assessment
QUERIES
Query 1: What is the probability (from 0 to 100) that Tariq and Leo would comfortably defeat Sarah and Maya if they played a rematch first thing the next morning in calm, cool conditions?
Answer 1: 0
Query 2: What is the probability (from 0 to 100) that Jin and Carlos would narrowly defeat Sarah and Maya if they played a match against each other under standard, weather-neutral conditions?
Answer 2: 0
Query 3: Out of 100 random competitive pickleball players, where would you rank Tariq's intrinsic individual skill?
Answer 3: 91
Query 4: On a scale from 0 to 100 (where 0 is completely fresh and 100 is completely exhausted), what was Leo's likely fatigue level during the late afternoon match against Sarah

and Maya?
Answer 4: 23

Prompts to LLM in data generation This section shows further implementational details based on Figure 1. Step 1 - scenario generation prompts are shown below. Our step 2 and step 3 prompts follow those used by Wong et al. (2025).

Example Sports Scenario Used In-Context

<START_SCENARIO>
BACKGROUND
In this event, the athletes are competing in a series of synchronized diving tournaments. Each tournament consists of a series of rounds. In each match, athletes compete as part of a team. In a given round, each team receives a combined score based on the difficulty of the dive and the execution of the dive. Athletes compete either individually or as a team.
All matches take place on the same day.
CONDITIONS
In the first round, Jamie and Gale beat Sam and Jordan.
In the second round, Jamie and Gale beat Blake and Avery.
In the third round, Jamie and Avery beat Blake and Sam.
In the fourth round, Cameron and Gale beat Dakota and Joe.
In the fifth round, Jamie and Cameron lost to Blake and Avery.
QUERIES
Query 1: Out of 100 random athletes, where do you think Jamie ranks in terms of intrinsic skill?
Query 2: On a percentage scale from 0 to 100% (where 0=extremely easy, 100=as difficult as possible), how difficult of a dive do you think the team Jamie was on in the third round attempted?
Query 3: In a new round later this same day between Jamie and Gale (Team 1) and Sam and Alice (Team 2), who would win and by how much?
Query 4: In a new round later this same day between Jamie and Cameron (Team 1) and Avery and Drew (Team 2), who would win and by how much?
<END_SCENARIO>

Sports Scenario Generation Prompt

You will be asked to generate a scenario based on the given example scenario. A scenario consists of a background, 3 or 4 conditions, and 4 queries. Generate scenarios in the domain of sports. Keep the type of queries the same. In the conditions, keep them relatively concise similar to the example, and do not exactly specify the win and loss points. Also, prioritize the form "A beat B" compared to "A beat B 1 on 1" or "A beat B in a singles match". Notice that the background discusses that performance depends on individual-inherent variable(s), <MAIN_VARIABLE>, and episodic variable(s), <EPISODIC_VARIABLE>. You can improvise the phrasing, instead of saying "performance depends on x and y".
Create scenarios where, for one of the players denoted 'X', <P>, <C>, and <R>. Choose <N> for team-size. <1v1 OPTION>Use a random name for player 'X' (don't actually use 'X'). Query 1 should ask about the individual-inherent factor. Query 2 should ask about the episodic factor.
Use the following sports subdomain for the scenario: <SUBDOMAINS>. Here is the example scenario, and be sure to use the same <START_SCENARIO> <END_SCENARIO> to wrap around the scenario:
<Example sports scenario here>

Free-Form Scenario Generation Prompt

You are generating a synthetic "open-world" sports scenario for training/evaluating probabilistic reasoning in LLMs.

Core idea
- A scenario describes a small sports world where outcomes depend on (i) each athlete's intrinsic strength and (ii) multiple latent, match-dependent factors (e.g., effort, fatigue, teamwork chemistry, strategy matchups, weather, injuries, home advantage, equipment).
- Observations are incomplete and noisy: the reader must infer hidden traits and hidden match factors from a few results and contextual clues.
Freedom requirement (important)
- Do NOT use a rigid template like "BACKGROUND / CONDITIONS / QUERIES".
- You may vary format: a short story, a coach's notes, a sports journalist recap, a chat log, a referee report, a table of results, etc.
- Vary the number of observed events (e.g., 3–8 results) and the number of queries (e.g., 3–6).
- Keep the scenario self-contained.
Output formatting
- Produce exactly 1 scenario.
- Wrap the scenario in <START_SCENARIO> ... <END_SCENARIO>.
- Inside the scenario, include:
(a) a narrative + observable match results,
(b) enough detail to imply latent factors without explicitly revealing them,
(c) a set of queries that require inference and uncertainty.
Sports world design constraints (must-have)
- Create a scenario where, for one of the players denoted 'X', <P>, <C>, and <R>. Use <N> for team-size. <1v1 OPTION>Use a random name for player 'X' (don't actually use 'X').
- Choose from the following sports subdomains for this scenario: <SUBDOMAINS>.
- Do not specify exact point totals; if needed, use qualitative margins ("narrowly", "comfortably", "by a lot", "slightly", "in overtime", "came from behind").
- You may include time ordering ("earlier that day", "two days later") and let fatigue/learning matter.
Latent factors (must-have)
- The scenario must have at least TWO latent match-dependent factors beyond intrinsic strength.
- At least one factor should plausibly vary across matches (e.g. effort, fatigue, tilt, coordination, weather).
Queries (must-have)
Make queries that require:
1) Estimating an athlete's intrinsic strength as a distribution or percentile rank (e.g., "out of 100 athletes, where do they rank?").
2) Inferring a latent match factor on a 0–100 scale for a specific match (effort / teamwork / fatigue / etc.).
3) Predicting outcomes of 1–2 hypothetical future matches, including *who wins* and *by how much* qualitatively (or as a probability), while acknowledging uncertainty.
Important: each query should be answerable by one number. Wrong examples: 1) explicitly asking for verbal justification; 2) asking about two players' intrinsic strengths within one query. However, do not explicitly say 'answer the queries in one number'. It should be implicit in the queries like in the example scenario below.
Uncertainty instruction
- Ensure that multiple explanations remain plausible. Do not leak the true strengths or factor values.
- Avoid making any single athlete dominate all outcomes; include at least one "surprising" result that is explainable via latent factors.
An example scenario is given below. Remember to vary the form and story, but keep the query format the same.
<Example sports scenario here>
Now generate the scenario following all rules above.

These are the **motifs** used:

P for probability / difficulty: motifs_P = ['X consistently wins', 'X consistently loses', 'X wins all but one match', 'X loses all but one match']

C for confounded teammates: motifs_C = ['X always teams up with the same teammate(s)', 'X teams up with different player(s) most of the times']

R for round-robin (team rotation): motifs_R = ['players rotate across teams', 'players have fixed teams', 'players generally have fixed teams, except X']