Instruction-Tuned Language Models Cannot Sample from Distributions They Can Describe

Anonymous ACL submission

Abstract

Silicon sampling uses language models as proxies for human survey respondents, treating each model call as an independent draw from the persona's response distribution. We show this draw does not exist: instruction-tuned models do not sample from distributions, they collapse to a single output. The same persona on the same question returns the same answer on more than half of items in a public-opinion benchmark. The collapse is sharp: the model's internal probabilities concentrate on a single option, and the failure is substantially amplified by instruction tuning: across three model families with materially different post-training pipelines, every instruction-tuned model fails on every task we test, while base models fail far less often. Strikingly, the same model that cannot sample from a distribution can describe it accurately in a single call. We call this gap the KNOWS/DOES split, and trace it to a degenerate sampling primitive visible in the logits and induced by alignment training. Exploiting this split, asking the model to describe the response distribution in one call more than halves the error against human survey data compared to persona aggregation. For applications that require per-persona outputs, we propose Prompt-Perturbed Argyle (PPA), which reduces the same error by 21% at no added cost.

1 Introduction

Silicon sampling uses language models as proxies for human survey respondents (Argyle et al., 2023; Park et al., 2023; Santurkar et al., 2023). The standard pipeline prompts the model with a set of personas, treats each call as one draw from the corresponding persona-conditional response distribution, and aggregates the responses to produce a population estimate. The method has been adopted across agent-based simulation (Park et al., 2023, 2024) and survey methodology (Horton et al., 2023; Aher et al., 2023). A growing literature reports that silicon sampling pipelines systematically miscalibrate the human distributions they target (Bisbee et al., 2024; Boelaert et al., 2025; Tjuatja et al., 2024; Hu and Collier, 2024; Gligorić et al., 2024), with proposed causes including representational gaps in training data and biases acquired through alignment. We argue that the miscalibration follows from a more basic failure: the per-call output of an instruction-tuned language model is not a sample from the conditional distribution the pipeline assumes.

We characterize this failure on a controlled set of synthetic categorical targets. Asked for a uniform random integer in 1–100, an instruction-tuned model returns 42 in 78% of calls (Table 2). Across seven categorical targets covering skewed, bimodal, and multi-way distributions, the model produces a single answer in over 94% of calls. The failure is visible at the representational level: the top-two logit gap at temperature zero exceeds 14 nats on some targets, beyond the recoverable range of any decoding temperature available through standard APIs. Algorithmic chain-of-thought scaffolds do not address the failure either, because the random number the model draws inside the scaffold is itself non-uniform.

The KNOWS/DOES split (Figure 1) has direct consequences for silicon sampling. On 100 OpinionQA items (Santurkar et al., 2023) drawn from Pew American Trends Panel surveys (Pew Research Center, 2024), standard persona aggregation fails to produce variation: across 50 repeated calls on the same (persona, item) pair, 57% of pairs return the same answer every time, and 80% return the same answer in over 90% of calls. The describe pathway recovers what aggregation cannot. A single call asking the model for the response distribution directly cuts the gap to the human results by more than half (TV distance 0.22 vs. 0.46), and the advantage holds even when personas are drawn

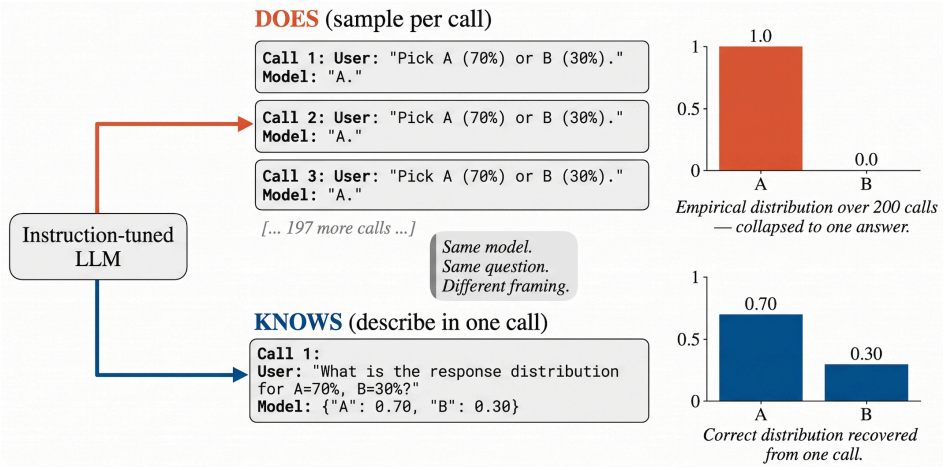

Figure 1: The KNOWS/DOES split on the skewed binary target. When asked to sample one outcome per call, the instruction-tuned model collapses to a single answer across 200 repeated calls (empirical: A=1.00, B=0.00). When asked to describe the distribution in one call, the same model recovers the correct shape (A=0.70, B=0.30). The pattern holds across all five tasks tested (§3.5; Table 4). Failure persists across temperature, top-p, and prompt-level corrections.

to match the actual Pew respondents on demographics. Some applications still need a separate output per simulated respondent. For those, we propose Prompt-Perturbed Argyle (PPA), which randomizes option ordering and question phrasing across calls. PPA reduces the same gap by 21% at no added cost, driven by the per-call answer variation it induces.

Contributions. Prior work has documented mode collapse under RLHF (Padmakumar and He, 2024), silicon sampling miscalibration (Bisbee et al., 2024; Boelaert et al., 2025; Santurkar et al., 2023), and the asymmetry that describing a distribution outperforms aggregating per-call samples (Meister et al., 2025; Xiao et al., 2025; Zhang et al., 2025). We make four contributions.

1. A controlled characterization of the per-call sampling failure, including a chain-of-thought analysis showing why algorithmic scaffolding cannot fix it (§3.2–§3.3).

2. Direct empirical evidence that the failure is induced by alignment training, across three model families with materially different post-training pipelines (§3.4).

3. A demographically-matched Argyle baseline that closes the persona-mismatch confound in prior describe-vs-sample evaluations (§4).

4. Prompt-Perturbed Argyle (PPA): a new method for per-respondent applications that reduces TV by 21% at no added cost, with a controlled mechanism analysis (§5).

2 Related Work

Silicon sampling and its miscalibration. LLMs-as-respondents originated with Argyle et al. (2023) and the paradigm has been extended to opinion dynamics (Park et al., 2023, 2024), social experiments (Aher et al., 2023), and economic decisions (Horton et al., 2023). A parallel literature documents systematic miscalibration of these pipelines: LLM-generated populations show smaller effective sample sizes than the human populations they approximate (Bisbee et al., 2024), between-group variance collapses (Boelaert et al., 2025), the resulting estimates skew toward specific demographic viewpoints (Santurkar et al., 2023), response biases diverge from human survey patterns (Tjuatja et al., 2024), and prompt-framing sensitivity destabilizes the estimates further (Hu and Collier, 2024; Gligorić et al., 2024). These failures are typi-

cally attributed to representational gaps in training data or to social biases acquired through alignment. We identify a more proximate cause: the per-call sampling primitive is structurally degenerate in instruction-tuned models, and this failure propagates into every aggregate the pipeline produces.

Verbalized sampling and describer–executor asymmetry. Meister et al. (2025) and Xiao et al. (2025) document that asking the model to describe a distribution outperforms aggregating per-call samples on opinion benchmarks, and Zhang et al. (2025) propose Verbalized Sampling (VS), which prompts the model to verbalize the distribution as JSON probabilities. The describe pathway we evaluate is structurally identical to VS, and on our 100-item OpinionQA benchmark the two are statistically indistinguishable (Wilcoxon $p = 0.14$). We extend this line of work mechanistically, tracing the asymmetry to a degenerate sampling primitive visible in the logits and linking it to alignment training, and methodologically, with PPA for pipelines where single-call description cannot substitute.

Mode collapse, RLHF, and decoding. Instruction-tuned and RLHF-aligned models exhibit reduced output diversity relative to base models (Padmakumar and He, 2024), with proposed mechanisms including reward-model overfitting (Ouyang et al., 2022; Stiennon et al., 2020) and KL regularization toward a peaked reference policy (Bai et al., 2022; Glaese et al., 2022). Standard decoding controls (temperature, top-k, top-p) operate on a next-token distribution that is fixed at inference time (Holtzman et al., 2020; Fan et al., 2018). We sharpen this picture in two ways. First, the collapse on uniform-target prompts is categorical rather than statistical: more than 94% of calls produce the same option, and the logit gap at $T = 0$ exceeds 14 nats on some targets, far beyond any temperature the API permits. Second, our base-vs-instruct comparison establishes the alignment-induced direction empirically.

Formal versus functional competence. The KNOWS/DOES split sits within the broader dissociation Mahowald et al. (2024) call formal vs. functional competence: models can introspect on knowledge better than they can produce it (Kadavath et al., 2022), can be elicited to express calibrated uncertainty over outputs they cannot generate correctly (Lin et al., 2022), and exhibit verbalize–execute splits on procedural and probabilistic tasks (Xiao et al., 2025). We extend the frame to distributional generation, providing a specific mechanism in the form of a degenerate sampling primitive visible at the logit level and identifying alignment training as the cause.

LLMs as data sources and in-context learning. The use of LLMs for synthetic training data (Wang et al., 2023; Honovich et al., 2023), evaluation benchmarks (Liu et al., 2022), and information retrieval queries (Bonifacio et al., 2022) shares silicon sampling's assumption that per-call outputs are draws from a target. Our findings imply this assumption requires verification for any aggregate-dependent pipeline. The KNOWS pathway's success connects to explanations of in-context learning as structured pattern-matching (Brown et al., 2020; Wei et al., 2022; Kojima et al., 2022; Xie et al., 2022; Min et al., 2022). PPA exploits persona conditioning (Park et al., 2022; Aher et al., 2023) not as a sampling mechanism but as a way to land each call on a different deterministic answer.

3 Characterizing the Per-Call Sampling Failure

This section characterizes the per-call sampling failure on a controlled set of synthetic categorical targets. We document the failure at three levels: how the model samples from a stated target distribution, what its logits look like at temperature zero, and what it does when asked for a random number with no target specified. The third level matters because the most natural way to fix the first — walking the model through a sampling algorithm via chain-of-thought — depends on the model being able to produce a random number internally, and we show this primitive is broken.

3.1 Setup

Unless noted, we use `gpt-4o` at temperature 1.0 with 200 calls per condition for behavioral runs, and temperature 0 with top-20 logprobs for logit reads. The base-vs-instruct comparison of §3.4 additionally uses `Llama-3.1-8B`, `Mistral-7B-v0.3`, and `Qwen2.5-7B` with their instruction-tuned counterparts. Full setup, prompts, and parsing rules are in Appendix A; cross-family replication on five additional instruction-tuned families is in Appendix Q. Throughout, we report total-variation (TV) distance between the empirical distribution

of 200 calls and the target distribution, which is bounded in $[0, 1]$ with 0 indicating perfect match.

3.2 The failure is categorical

The model handles the two boundary cases. A point target ("always output 5") and an unspecified uniform ("output any digit 1–5") both produce usable distributions. Every shape between fails (Table 1; full per-target results in Appendix E): across the five non-trivial targets, the model produces a single answer in over 94% of calls.

Target	Target distribution	Empirical distribution
Skewed binary	$[0.70, 0.30]$	$[1.00, 0.00]$
Mixture	$[0.70, 0.30]$	$[0.98, 0.02]$
Three-way	$[0.50, 0.30, 0.20]$	$[0.96, 0.03, 0.01]$
Bimodal	$[0.5, 0, 0, 0, 0.5]$	$[1.00, 0, 0, 0, 0]$
Skewed 5-way	$[0.40, 0.30, 0.15, 0.10, 0.05]$	$[0.94, 0.04, 0.01, 0.01, 0.00]$

Table 1: Per-target results across the five non-trivial targets, 200 calls each. The empirical distribution concentrates on a single answer in over 94% of calls on every target, regardless of the shape the target requests.

Two features of Table 1 go beyond simple miscalibration. First, the bimodal row shows the model honoring the zero-probability constraint perfectly (mass 0 on options 2, 3, 4) while completely ignoring the 50/50 part of the same instruction. The same prompt simultaneously satisfies a point-shaped constraint and violates a distribution-shaped one. Second, swap tests confirm that the model is reading the instruction rather than producing a fixed default: swapping a target from $[A{=}0.7, B{=}0.3]$ to $[A{=}0.3, B{=}0.7]$ relocates the model's single output from A to B but never produces the intended distribution.

The failure is not sampling noise either. At temperature zero, the top-two logit gap ranges from 3.75 to over 16 nats across our targets (Appendix F), and recovering a 70/30 target from a 14-nat gap would require a sampling temperature near 17, far above the API maximum of 2. Decoding adjustments cannot fix the failure because it occurs before sampling.

3.3 The failure originates in a degenerate sampling primitive

If the failure lives before sampling, where in the model does it live? We trace it to the model's behavior on direct random-number prompts that contain no persona, no distribution, and no task framing.

Asked to "pick a uniform random X" across five output formats, the model concentrates on a format-specific favored value in every case (Table 2). All five reject uniformity at $p < 0.001$.

Format	Favored value	Share
Integer 1–100	42	78%
Integer 1–10	7	97%
Alphabet A–Z	G	47%
Coin flip	heads	85%
Float $[0, 1]$	0.55 (std 0.11)	—

Table 2: The model concentrates on a format-specific favored value when asked for a uniform random sample (200 calls per format). All five formats reject uniformity at $p < 0.001$. Full results in Appendix G.

The concentration is not a lexical effect. Excluding the modal value by name relocates the mass to a different favored value rather than spreading it: "Do not pick 42" yields 37 in 52% of calls. Cross-family replication shows the same pattern on five other instruction-tuned families (Appendix Q), each with its own favored value (47, 53, 73, 1 on the 1–100 task). This degenerate primitive turns out to be what algorithmic chain-of-thought scaffolds also rely on, which we return to in §3.5.

3.4 The failure is induced by alignment training

The failure is not inherent to language modeling. We compare each instruction-tuned model against its pretrained base counterpart on the same five non-trivial targets, across three families with materially different post-training pipelines: Llama-3.1-8B (RLHF-based), Mistral-7B-v0.3 (publicly described as supervised fine-tuning without RLHF), and Qwen2.5-7B (a multi-stage pipeline that includes DPO). Full protocol is in Appendix B.

	Llama-3.1-8B		Mistral-7B-v0.3		Qwen2.5-7B	
Task	Base	Inst	Base	Inst	Base	Inst
uniform 1–5	0.10	**0.29**	0.15	**0.26**	0.22	**0.80**
fair coin	0.15	**0.21**	0.11	**0.36**	0.29	**0.50**
skewed binary	0.06	**0.30**	0.17	**0.30**	0.24	**0.30**
bimodal 5-way	0.20	**0.44**	0.34	**0.50**	0.33	**0.50**
skewed 5-way	0.13	**0.37**	0.14	**0.59**	0.20	**0.59**
mean	0.13	0.32	0.18	0.40	0.26	0.54
# fail (>0.20)	1/5	5/5	1/5	5/5	5/5	5/5

Table 3: V0 (plain sampling) TV-to-target across three families, base vs. instruction-tuned counterpart, 200 calls per cell. Bold cells indicate failure at the TV > 0.20 threshold.

Across all three families, the instruction-tuned model has higher TV than its pretrained base counterpart on every task (Table 3). The mean gap is

0.19 on Llama, 0.22 on Mistral, and 0.28 on Qwen. The Mistral-v0.3 result is particularly informative because v0.3 is publicly described as SFT-centered without an RLHF stage, yet the gap appears, suggesting that the failure is induced by instruction tuning broadly rather than by RLHF specifically.

3.5 The model can describe what it cannot sample

Given that the failure is alignment-induced and lives in the sampling primitive, a natural question is whether the model still has the distributional knowledge in another form. We test this through a controlled ten-intervention study on the five non-trivial categorical tasks (Table 4). Eight interventions (V0–V7) instruct the model to produce one sample per call, including six prompt-level corrections and algorithmic chain-of-thought. Two interventions (V8, V9) instruct the model to describe the distribution in a single call, either as a list of N exemplars or as JSON probabilities. Full prompts are in Appendix H.

ID	Intervention	Type	TV
V0	Plain sampling	DOES	0.52
V1	Verbatim feedback	DOES	0.47
V2	Apologetic framing	DOES	0.48
V3	Third-person framing	DOES	0.50
V4	Explicit bias-naming	DOES	0.52
V5	In-context demos	DOES	0.50
V6	Step-by-step reasoning	DOES	0.43
V7	Algorithmic CoT	DOES	0.48
V8	List of N exemplars	KNOWS	**0.03**
V9	JSON probabilities	KNOWS	**0.03**

Table 4: Ten prompt-level interventions on five categorical tasks, mean TV across tasks (200 calls per task). Every DOES intervention fails on every task (TV > 0.20); both KNOWS interventions succeed on all five. Per-task TV and full intervention text in Appendix I.

The pattern in Table 4 is sharp: every DOES intervention fails on every task, and both KNOWS interventions succeed within 0.05 TV. V7, the algorithmic chain-of-thought intervention, fails for a specific reason: the random u the model writes inside the inverse-CDF scaffold is itself non-uniform on every task ($p < 10^{-13}$; Appendix J), inheriting the degenerate primitive of §3.3. V8's success does not reflect internal sampling. Across 200 calls per task, the cross-call sequence-rank correlation in V8's output lists exceeds 0.97, meaning the model produces near-identical orderings each call. The fair coin alternates H-T-H-T like a person trying to look random; the skewed five-way produces runs of repeated values that match the target proportions, more like a tally than a sample. V8 is effectively a parametric description rendered as a list: the model lays out approximately correct counts rather than drawing independent samples (Appendix P). The pattern is stable across list lengths $N \in \{2, 5, 10, 20\}$ (Appendix K). V8, V9, and the describe pipeline used in §4 all succeed by converting per-call sampling into a single-call counting or specification operation.

We call this gap between describing and sampling the *KNOWS/DOES split*: the model knows what the target distribution looks like but cannot enact that knowledge through per-call output.

4 Application to Silicon Sampling on OpinionQA

This section applies the KNOWS/DOES framework to silicon sampling on OpinionQA (Santurkar et al., 2023), which maps Pew American Trends Panel survey items (Pew Research Center, 2024) to 5-point Likert responses, using the Pew distribution as ground truth.

4.1 Setup

We use 100 OpinionQA items selected by a pre-registered rule from a fixed wave list, restricted to 5-point Likert items with usable ground truth, with each item run under 100 personas sampled from ANES demographic strata following the Argyle protocol (Argyle et al., 2023). The same persona set is used for all standard-Argyle, PPA, and decoding-sweep experiments, with `gpt-4o` at temperature 1.0. Full setup is in Appendix A. We evaluate three pipelines:

- **Standard Argyle** (baseline): for each (persona, item) pair, we make one call with the persona description, question, and response options, and aggregate the 100 responses per item.
- **Demographically-matched Argyle**: for each item, we sample 100 personas weighted to match the actual demographic distribution of Pew respondents on that item, using Pew's per-wave survey weights and the 12 demographic variables exposed in OpinionQA. Otherwise identical to standard Argyle. Full protocol in Appendix C.
- **Describe pathway**: one call per item, no personas. The prompt asks the model to return

the population response distribution directly as JSON probabilities. Full protocol in Appendix C.

4.2 Results

We evaluate the three pipelines on 100 OpinionQA items. Persona-aggregation pipelines use 100 personas per item with one call per cell. The describe pipeline uses one call per item with no personas. For the persona-aggregation pipelines we additionally run 50 repeated calls on 100 cells (20 items $\times$ 5 personas) to measure per-(persona, item) Shannon entropy (Appendix D, Table 10).

Pipeline	Per-call entropy	Deterministic cells	TV-to-Pew
Standard Argyle	9%	57%	0.46
Describe pathway	—	—	**0.22**

Table 5: Standard Argyle vs. the describe pathway on 100 OpinionQA items. Per-call entropy is as a fraction of the 5-Likert maximum. Matched Argyle is reported separately (Appendix C); its TV (0.484) is statistically indistinguishable from standard Argyle's, ruling out persona-mismatch as the explanation.

The categorical collapse documented in §3 on synthetic targets replicates on real opinion items: most (persona, item) cells return the same answer every call. Demographic matching reduces TV by only 0.025 (matched Argyle TV $= 0.484$ vs. standard $= 0.459$; paired Wilcoxon $p = 0.086$, $n = 100$; Appendix C), ruling out persona-mismatch as the explanation for persona aggregation's poor performance. The gap is not about which personas the pipeline samples; it is about whether each call is asked to sample one response (DOES) or describe the full population distribution (KNOWS). Because the synthetic targets are arbitrary distributions the model cannot have memorized, the KNOWS pathway's success on them rules out recall of training-set statistics as the explanation; OpinionQA tests the same mechanism on a real-data application. Describe matches Verbalized Sampling (Zhang et al., 2025) here ($p = 0.14$; Appendix C).

5 Prompt-Perturbed Argyle (PPA)

The describe pathway of §4 is the right choice when the application needs only a population estimate, but some applications require one simulated response per persona. Generative-agent simulations (Park et al., 2023, 2024) condition downstream reasoning on each agent's individual response, so a single-call describe pipeline cannot substitute. This section presents Prompt-Perturbed Argyle (PPA), a same-cost modification of standard Argyle for these applications.

5.1 Method

Under standard Argyle, the same persona on the same item produces the same single answer in every call (§4.2), so aggregating across calls cannot recover the within-stratum distribution. PPA addresses this by perturbing surface features of the prompt across the 100 calls per item, holding the persona's semantic content fixed.

PPA randomizes three surface features per call:

- **Option ordering** (4 choices): ascending, descending, or one of two shuffled orderings of the Likert options.
- **Question phrasing** (3 choices): direct, scale-anchored ("on a scale of 1 to 5…"), or third-person.
- **Prompt position** (3 choices): persona in system message, persona in user message, or persona in user message with brief recap.

Each call independently draws one combination uniformly at random from the $4 \times 3 \times 3 = 36$ possibilities. Standard Argyle uses a fixed canonical setting (ascending order, direct phrasing, persona in system message). Both pipelines make exactly 100 calls per item, one per persona, so the API cost is identical. Full protocol and surface-feature templates are in Appendix L.

The intuition is that these three features are the most plausible per-call anchors for an instruction-tuned model — the model's default response depends on the order it sees options, the surface form of the question, and where the persona is placed in the context. Perturbing all three across calls is intended to dislodge the per-call anchor without changing what the question is asking. We test in §5.3 whether the perturbations actually increase per-call entropy, and whether the entropy increase predicts the TV improvement.

5.2 Results

PPA reduces mean TV by 0.10 (a 21% relative reduction) at no added API cost (Wilcoxon paired $p = 8.2 \times 10^{-10}$). The effect is driven by the semantic perturbations rather than generic surface variation. On a pre-registered 13-item development subset of the 100 items, a control intervention that randomizes only whitespace, punctuation,

Pipeline	Mean TV	Δ vs. baseline
Standard Argyle (baseline)	0.46	—
PPA	**0.36**	**−0.10**

Table 6: PPA vs. standard Argyle on 100 OpinionQA items, same persona set (100 personas per item). PPA reduces mean TV by 0.10 (21% relative) at no added API cost (Wilcoxon paired test across 100 items: $p = 8.2 \times 10^{-10}$). Decoding-parameter sweeps (temperature, top-p) capture less than 20% of PPA's effect; see Appendix N.

Predictor	Coefficient	95% CI
ΔH (PPA entropy increase)	0.27^{***}	$[0.16, 0.38]$
H_{Pew} (item ambiguity)	0.04 (n.s.)	$[-0.07, 0.15]$
R^2	0.55	
N items	20	

Table 7: OLS regression of per-item TV improvement on per-item per-call entropy increase, controlling for Pew per-item entropy. Each 1-nat increase in per-call entropy predicts a 0.27 reduction in TV ($p < 0.001$). Item difficulty is non-significant.

and prompt prefix captures 13% of PPA's TV reduction, while option ordering and question phrasing together account for the remaining 87% (Appendix L).

5.3 Analysis: PPA's mechanism is per-call entropy

§4.2 showed that standard Argyle returns the same single answer on nearly every call. PPA varies the prompt across calls, so the same persona can produce different answers, raising per-call entropy. We test whether PPA's TV improvement comes specifically from this entropy increase.

For each of 20 OpinionQA items (5 from each pre-registered cell), we measure per-(persona, item) entropy across 50 calls under standard Argyle (H_{std}) and under PPA (H_{PPA}). This denser protocol — 5 personas per item, 50 calls per cell — is required because per-cell Shannon entropy needs many repeated calls on the same (persona, item) pair.

For each item i, we compute:

$$\Delta H_i = H_{\text{PPA},i} - H_{\text{std},i} \quad (1)$$

$$\Delta\text{TV}_i = \text{TV}_{\text{std},i} - \text{TV}_{\text{PPA},i} \quad (2)$$

Positive ΔTV_i means PPA improves accuracy. To separate the entropy mechanism from item difficulty, we control for the per-item Pew entropy $H_{\text{Pew},i}$:

$$\Delta\text{TV}_i = \alpha + \beta_1 \Delta H_i + \beta_2 H_{\text{Pew},i} + \varepsilon_i \quad (3)$$

Fitting Equation 3 across the 20 items yields a significant positive coefficient on ΔH (Table 7): PPA's effect on TV tracks how much it raises per-call entropy, not how ambiguous the item is. Items where PPA increases per-call entropy by more than 0.7 nats show TV improvements of 0.15–0.27; items where PPA barely changes per-call entropy show no improvement. The raw correlation between ΔH and ΔTV is $r = 0.73$ ($p = 2 \times 10^{-4}$); bootstrap resampling confirms the ΔH coefficient is positive in 100% of 1000 resamples (95% CI $[0.18, 0.37]$).

5.4 Ruling out alternative explanations

Two follow-up tests address potential confounds (Appendix O; Table 8). First, non-semantic perturbations of the prompt — whitespace, punctuation, prefix variation — raise per-call entropy by essentially zero, two orders of magnitude smaller than PPA, and yield correspondingly negligible TV improvements. What matters is the entropy increase, not prompt variation in general. Second, raising decoding temperature to $T = 1.5$ produces a smaller entropy increase and a smaller TV improvement than PPA, but the TV improvement per nat of entropy added is essentially the same: 0.239 for temperature, 0.255 for PPA. The same return per nat emerges from two unrelated interventions, consistent with an entropy-generic mechanism.

Intervention	ΔH (nats)	ΔTV	$\Delta\text{TV}/\Delta H$
PPA	0.417	−0.106	0.255
Non-semantic controls			
Whitespace + punct. + prefix	0.003	−0.014	—
Decoding-level analogue			
Temperature $T = 1.5$	0.069	−0.016	0.239

Table 8: Mechanism controls. Non-semantic perturbations raise per-call entropy by essentially zero and yield negligible TV improvement. Temperature raises entropy by less than PPA but produces the same TV improvement per nat ($\Delta\text{TV}/\Delta H \approx 0.25$ for both).

Two follow-up tests rule out alternative explanations (Appendix O). First, non-semantic perturbations of the prompt — whitespace, punctuation, prefix variation — raise per-call entropy by only $\Delta H = 0.003$ nats on average, two orders of magnitude smaller than PPA's $\Delta H = 0.417$, and yield correspondingly small TV improvements ($\Delta\text{TV} = 0.014$ vs. 0.106 for PPA). What matters

is the entropy increase, not prompt variation in general. Second, raising decoding temperature to $T = 1.5$ produces a smaller entropy increase ($\Delta H = 0.069$) and a smaller TV improvement ($\Delta\text{TV} = 0.016$), but the ratio $\Delta\text{TV}/\Delta H$ is essentially the same as PPA's: 0.239 for temperature, 0.255 for PPA. The same TV improvement per nat of entropy increase emerges from two unrelated interventions, consistent with an entropy-generic mechanism. Bootstrap resampling of the regression confirms the ΔH coefficient is positive in 100% of 1000 resamples (95% CI $[0.18, 0.37]$).

6 Discussion

We have documented a per-call sampling failure in instruction-tuned language models, traced it to alignment training through a base-vs-instruct comparison, and shown that the model's distributional knowledge remains accessible through a describe-style interface. Applying the framework to silicon sampling, we have provided two mechanism-grounded mitigations evaluated on 100 OpinionQA items.

Relation to mode-collapse literature. Mode-collapse research has documented reduced output diversity in RLHF-aligned models (Padmakumar and He, 2024), with proposed causes framed at training time (Ouyang et al., 2022; Bai et al., 2022). Our findings refine this picture in two ways. First, the collapse on uniform-target prompts is categorical rather than statistical, with the logit gap at temperature zero exceeding 14 nats on some targets, beyond the recoverable range of any temperature setting available through standard APIs. Second, the base-vs-instruct comparison in §3.4 provides direct evidence that the failure is alignment-induced, consistent with the broader formal-vs-functional dissociation discussed by Mahowald et al. (2024).

Mechanism evidence for PPA. The entropy-correlation regression in §5.3 shows that PPA's TV improvement is predicted by its per-call entropy increase, with item ambiguity controlled, and two follow-up tests rule out alternative explanations: non-semantic perturbations that do not raise per-call entropy do not improve TV, and an independent intervention that does raise entropy — decoding temperature — produces the same TV improvement per nat of entropy added. The describe pathway operates by a different mechanism, bypassing the per-call pipeline rather than increasing per-call entropy, and its advantage over Argyle survives the demographic-matching control of §4.2.

Implications for other LLM applications. The per-call assumption is inherited by several other LLM applications. Generative-agent simulations (Park et al., 2023, 2024) treat each agent's per-turn response as a draw from a distribution, synthetic data generation pipelines (Wang et al., 2023; Honovich et al., 2023) and automated dataset construction (Liu et al., 2022) treat per-call outputs as samples from a target, and LLM-as-judge protocols assume that ratings across calls are independent draws. For instruction-tuned models, distributional fidelity should be verified rather than assumed.

Practical guidance. When the application needs only a population estimate, a describe pipeline is preferable: a single call returning a JSON distribution is roughly $2.1\times$ more accurate than persona aggregation on OpinionQA, and the advantage holds against demographically-matched baselines. When per-respondent outputs are required and a describe pipeline cannot substitute, PPA reduces TV by 21% at no added API cost through randomization of option ordering and question phrasing. When sampling from a fixed target distribution is the goal, having the model describe the target as JSON and sampling externally with a standard library converts the DOES task into a KNOWS one.

7 Conclusion

Instruction-tuned language models do not sample from distributions; they collapse to a single output that survives temperature, top-p, and prompt-level corrections. Yet the same models can describe those distributions accurately in a single call, and our base-vs-instruct comparison shows that the per-call sampling failure is induced by alignment training rather than inherent to the architecture. For applications that need population estimates, asking the model to describe the response distribution directly more than halves the error against human survey data; for applications that need per-respondent outputs, Prompt-Perturbed Argyle reduces the same error by 21% at no added cost. The broader implication is that any LLM pipeline that treats per-call outputs as samples from a target distribution should verify that assumption rather than inherit it.

Limitations

Sampling-as-output tasks only. Our findings concern whether the model's output distribution across many calls matches a target. Per-token correctness tasks (translation, summarization, factual recall) are out of scope.

Describe pathway accuracy as upper bound. The describe pathway's TV-to-Pew is the model's same-session ceiling for these items, not an absolute ceiling for agent simulation. On populations outside the model's training distribution, the describer's accuracy may degrade.

PPA's structural ceiling. PPA redistributes mass within the model's existing prior across the available options. It cannot make the model assign probability to options the prior excludes; only the describe pathway can. When population estimation is the goal and the items may diverge from the model's prior, prefer the describe pathway.

Entropy-correlation scope. The entropy-correlation regression establishes the per-call mechanism for PPA on 20 OpinionQA items (5 from each pre-registered cell). The same mechanism is plausibly responsible for the describe-pathway advantage. But describe is a single-call method for which per-call entropy is undefined. We do not claim the regression directly demonstrates describe's mechanism.

Within-family scaling. The cross-family panel uses different-generation models (e.g. `gpt-5.4` vs. `gpt-4o`), conflating size and generation. We do not systematically test within-family scaling at fixed generation. Whether the failure intensifies, attenuates, or persists with scale within a single family is left to follow-up work.

Base-vs-instruct scope. The base-vs-instruct comparison uses 8B-parameter models because comparable open base-instruct pairs are not as cleanly available at larger scales. The cross-family panel (Appendix Q) extends to frontier-scale instruct models but cannot supply the matched base counterpart. The qualitative direction (alignment training induces the failure) is established at 8B and replicates qualitatively at frontier scale on the instruct side; an end-to-end base-vs-instruct comparison at 70B+ scale would strengthen the causal claim.

Ethical Considerations

Data. We use publicly available Pew ATP data (Pew Research Center, 2024) via OpinionQA (Santurkar et al., 2023); no human-subjects research was conducted.

Artifacts. All models and datasets are used under their respective licenses; our released prompts, code, and benchmark results are for research use.

Risks. LLM-simulated populations should not substitute for real human survey data in policy-relevant settings; the describe pathway and PPA mitigate but do not eliminate the failure, and application-specific validation against human ground truth remains necessary.

AI assistants. We used AI coding assistants and writing-editing tools during preparation; all experimental design, analysis, and conclusions are the authors' own.

References

Gati V. Aher, Rosa I. Arriaga, and Adam Tauman Kalai. 2023. Using large language models to simulate multiple humans and replicate human subject studies. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*.

Lisa P. Argyle, Ethan C. Busby, Nancy Fulda, Joshua R. Gubler, Christopher Rytting, and David Wingate. 2023. Out of one, many: Using language models to simulate human samples. *Political Analysis*, 31(3):337–351.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*.

James Bisbee, Joshua D. Clinton, Cassy Dorff, Brenton Kenkel, and Jennifer M. Larson. 2024. Synthetic replacements for human survey data? The perils of large language models. *Political Analysis*, 32(4):401–416.

Julien Boelaert, Samuel Coavoux, Étienne Ollion, Ivaylo Petev, and Patrick Präg. 2025. Machine bias: How do generative language models answer opinion polls? *Sociological Methods and Research*, 54(3):1156–1196.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. InPars: Unsupervised

dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2387–2392.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, Lucy Campbell-Gillingham, Jonathan Uesato, Po-Sen Huang, Ramona Comanescu, Fan Yang, Abigail See, Sumanth Dathathri, Rory Greig, Charlie Chen, and 15 others. 2022. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*.

Kristina Gligorić, Myra Cheng, Lucia Zheng, Esin Durmus, and Dan Jurafsky. 2024. NLP systems that can't tell use from mention censor counterspeech, but teaching the distinction helps. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428.

John J. Horton, Apostolos Filippas, and Benjamin S. Manning. 2023. Large language models as simulated economic agents: What can we learn from *homo silicus*? Technical Report w31122, National Bureau of Economic Research.

Tiancheng Hu and Nigel Collier. 2024. Quantifying the persona effect in LLM simulations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10289–10307.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, and 17 others. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 22199–22213.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. WANLI: Worker and AI collaboration for natural language inference dataset creation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6826–6847.

Kyle Mahowald, Anna A. Ivanova, Idan A. Blank, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko. 2024. Dissociating language and thought in large language models. *Trends in Cognitive Sciences*, 28(6):517–540.

Nicole Meister, Carlos Guestrin, and Tatsunori 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 (NAACL)*, pages 24–49.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 11048–11064.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Vishakh Padmakumar and He He. 2024. Does writing with language models reduce content diversity? In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*.

Joon Sung Park, Lindsay Popowski, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2022. Social simulacra: Creating populated prototypes for social computing systems. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST)*.

Joon Sung Park, Carolyn Q. Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S. Bernstein. 2024. Generative agent simulations of 1,000 people. *arXiv preprint arXiv:2411.10109*.

Pew Research Center. 2024. American Trends Panel. https://www.pewresearch.org/the-american-trends-panel/. Accessed 2026.

Shibani Santurkar, Esin Durmus, Faisal Ladhak, Cinoo Lee, Percy Liang, and Tatsunori Hashimoto. 2023. Whose opinions do language models reflect? In *Proceedings of the 40th International Conference on Machine Learning (ICML)*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Lindia Tjuatja, Valerie Chen, Tongshuang Wu, Ameet Talwalkar, and Graham Neubig. 2024. Do LLMs exhibit human-like response biases? A case study in survey design. *Transactions of the Association for Computational Linguistics*, 12:1011–1026.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-Instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 24824–24837.

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*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit Bayesian inference. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*.

Jiayi Zhang, Simon Yu, Derek Chong, Anthony Sicilia, Michael R. Tomz, Christopher D. Manning, and Weiyan Shi. 2025. Verbalized sampling: How to mitigate mode collapse and unlock LLM diversity. *arXiv preprint arXiv:2510.01171*.

A Experimental setup

Main model. `gpt-4o` accessed via the OpenAI Chat Completions API. The cross-family replication (Appendix Q) uses additional API endpoints (`gpt-5.4`, `claude-sonnet-4.5`, `gemini-3-flash`, `deepseek-chat`) and the open-weight `llama-3.3-70b-instruct` accessed via a hosted inference API.

Base-vs-instruct models. `meta-llama/Llama-3.1-8B` and `meta-llama/Llama-3.1-8B-Instruct`, accessed via HuggingFace, loaded in bfloat16 on a single NVIDIA A100 (80GB) GPU. See Appendix B for full protocol.

Sampling. Behavioral sampling uses temperature 1.0, top-p 1.0, max-tokens 50, with $N = 200$ calls per condition. Logit reads use temperature 0 and the API's `top_logprobs` parameter set to 20.

Personas. We use 100 personas constructed from ANES 2020 demographic strata following the protocol of Argyle et al. (2023), with demographic-attribute templates listed in the released code. The same persona set is used for all standard-Argyle OpinionQA experiments (baseline, PPA, decoding-parameter sweeps, ablations). Matched-Argyle uses per-item demographic-matched personas as described in Appendix C.

Items. The 100 OpinionQA items are drawn deterministically by item-key sort, restricted to 5-point Likert with usable ground truth. The first 50 items come from the primary wave list (Pew ATP Waves 26, 27, 34, 42, 43, 50, 54). The second 50 items extend the wave list (in pre-committed order: W92, W36, W41, W49, W82, W45, W32). The full item list, the FACTS/VALUES annotation, and the entropy-bin labels are committed to the repository.

Parsing. For Likert tasks, we accept any prefix-numeric response in {1, 2, 3, 4, 5} and a small set of synonym strings ("strongly agree"→5, etc.); parse failures (less than 2% of calls across all conditions) are excluded from analysis. For RNG tasks, we accept only the bare integer/letter/float and reject any chain-of-thought, free text, or out-of-range value.

Statistics. For a discrete sample space $\mathcal{X}$ and two distributions P, Q over $\mathcal{X}$, the total-variation (TV) distance is

$$\mathrm{TV}(P, Q) = \frac{1}{2} \sum_{x \in \mathcal{X}} |P(x) - Q(x)|.$$

TV is bounded in $[0, 1]$, with 0 indicating perfect match and 1 indicating disjoint support. We use TV throughout the paper for two reasons: it is symmetric (unlike KL divergence, which is undefined when the target assigns zero probability to an option the empirical distribution covers), and it has a direct probabilistic interpretation as the maximum difference in event probabilities between P and Q.

In all experiments, we report TV between the empirical distribution over N model calls and the target distribution specified by the prompt (for the synthetic targets of §3) or by the Pew ground truth (for OpinionQA in §4).

Prompt templates. Verbatim prompt text for every condition is in the released code repository under `prompts/`; see Appendix H for the seven DOES corrections and two KNOWS interventions used in §3.5.

B Base-vs-instruct protocol

Models. `meta-llama/Llama-3.1-8B` (base, no instruction tuning) and `meta-llama/Llama-3.1-8B-Instruct` (the standard instruction-tuned counterpart). Both loaded in bfloat16 on a single A100 GPU.

Prompting. Base models do not reliably follow zero-shot instructions, so we use a 3-shot in-context format for V0 on the base model: three (Q, A) example pairs followed by the test question, with one canonical Q/A example for V8. The instruct model uses the standard chat template (zero-shot). This is the standard approach for eliciting task-following from a base model without conflating "cannot follow instruction" with "cannot sample."

Sampling. Temperature 1.0, top-p 1.0, 200 V0 calls per task, 10 V8 calls per task (each returning a list of 20). Total: 2 models $\times$ 2 conditions $\times$ 5 tasks $\times$ 200 calls $=$ 4,000 generations.

Parsing. The same parsers used elsewhere in the paper. Parse-failure rates on the instruct V0 condition are elevated on two cells (skewed binary: 156/200 failures, bimodal 5-way: 118/200) due to the instruct model wrapping its answer in hedge text (e.g., "As an AI I cannot truly sample, but here is A"). The valid-parse subset on these cells shows $P(\text{mode}) = 1.0$ on skewed binary (44/44 the same option), confirming the failure pattern rather than indicating noise. Base-model parse failures are more uniform (2–33 per cell, mostly 10–20%) and do not change the qualitative result.

V8 results. Table 9 reports V8 (list-of-$N = 20$) TV-to-target for the base and instruct models on the same five tasks. Both succeed on three of five tasks (TV < 0.10), and the base model's V8 performance is no worse than the instruct model's — on three tasks the base model is closer to the target. This rules out a "base model cannot follow the V8 instruction" confound and shows that the KNOWS pathway is already present pre-alignment.

Task	Base V8 TV	Instruct V8 TV
uniform 1–5	0.00	0.04
fair coin	0.09	0.06
skewed binary	0.03	0.13
bimodal 5-way	0.47	–*
skewed 5-way	0.24	0.09

Table 9: V8 (list-of-$N = 20$) TV-to-target on `Llama-3.1-8B` base vs. instruct. * Instruct V8 on bimodal 5-way produced 10/10 parse failures (the model refused or could not produce a JSON list honoring zero-probability constraints), so TV is undefined. Both models succeed at V8 on 3/5 tasks (TV < 0.10). The KNOWS pathway is preserved on the base model.

Why 8B vs. a larger model. The 8B model fits comfortably in bfloat16 on a single A100. Comparable base-vs-instruct pairs are not as cleanly available at 70B (different fine-tuning lineages). The 8B result establishes the qualitative direction; scaling within Llama-3 (e.g., 70B base vs. instruct, if released) would refine it.

C Demographically-matched Argyle protocol

Describe vs. Verbalized Sampling. Verbalized Sampling (Zhang et al., 2025) prompts the model to verbalize the response distribution as JSON probabilities in a single call. Our describe pathway uses the same query structure. On the 100 OpinionQA items, the two pipelines produce statistically indistinguishable TV-to-Pew (Describe: 0.22; VS: 0.21; Wilcoxon paired $p = 0.14$, $n = 100$). We use the describe pathway in the main results because it

integrates with the broader KNOWS/DOES framework, but the underlying mechanism — single-call distributional query — is the same.

Goal. Control for persona-prior mismatch in the describe-vs-Argyle comparison. The standard Argyle baseline uses 100 generic ANES-derived personas reused across all items. A skeptic could argue describe wins because these personas don't match the actual Pew respondent demographic distribution for each item. Matched Argyle eliminates this confound by drawing personas from each item's actual respondent demographic distribution.

Per-item respondent distributions. For each of the 100 OpinionQA items, we identify the Pew ATP wave it comes from, load the wave's respondent-level data from the OpinionQA release (Santurkar et al., 2023), filter to respondents who answered the item (non-null response), and compute the weighted demographic distribution across the 12 demographic variables exposed in OpinionQA (`CREGION`, `AGE`, `SEX`, `EDUCATION`, `CITIZEN`, `MARITAL`, `RELIG`, `RELIGATTEND`, `POLPARTY`, `INCOME`, `POLIDEOLOGY`, `RACE`) using the wave's survey weights (`WEIGHT_W{wave}`).

Persona sampling. For each item, we sample 100 personas from this weighted respondent distribution (sampling with replacement). Each sampled persona is formatted as a natural-language description using the existing project's persona template; we use the canonical persona-formatting code so the only difference from standard Argyle is which personas are used.

API calls and aggregation. For each (matched persona, item) pair: a single Argyle call at temperature 1.0 using `gpt-4o`. Total: 100 items $\times$ 100 matched personas $=$ 10,000 calls. Aggregate to per-item LLM response distribution; compute TV against the Pew per-item distribution.

Result. Matched Argyle mean TV $= 0.484$, vs. standard Argyle TV $= 0.459$. The two are not significantly different (paired Wilcoxon $p = 0.086$, $n = 100$ items); if anything matched Argyle is slightly worse, suggesting that matching personas to demographics slightly amplifies the model's tendency to stereotype the demographic group. Describe still beats matched Argyle by ΔTV $= 0.269$ ($p = 4.5 \times 10^{-16}$).

D Per-cell entropy on OpinionQA

For 20 OpinionQA items (5 from each pre-registered cell) and 5 personas per item, we run standard Argyle 50 times per (persona, item) cell. Per-cell entropy statistics across the 100 cells:

Metric	Value
Cells with entropy $= 0$ (perfectly deterministic)	57/100
Cells with top response in $> 90\%$ of calls	80/100
Cells with top response in $> 95\%$ of calls	75/100
Mean per-cell entropy	0.145 nats
Mean per-cell entropy as % of max	9.0%
Median per-cell entropy	0.000

Table 10: Per-cell entropy across 100 (persona, item) cells under standard Argyle, 50 calls per cell. The §3–§4 categorical collapse replicates on real OpinionQA items.

The §3–§4 categorical collapse is not a synthetic-task artifact. On real opinion items under standard persona-conditioned Argyle, more than half of the (persona, item) cells produce a perfectly deterministic response across 50 repeated calls.

E Seven-target panel

Per-target results at $N = 200$ calls each, $T = 1.0$.

Target	target q^*	P(mode)	TV	N
point (always 5)	$[0, 0, 0, 0, 1]$	1.000	0.000	200
mixture 5:70%/4:30%	$[0, 0, 0, 0.3, 0.7]$	1.000	0.300	200
skewed binary 0.7/0.3	$[0.7, 0.3]$	1.000	0.300	200
bimodal 1 & 5, never 2/3/4	$[0.5, 0, 0, 0, 0.5]$	1.000	0.500	200
three-way 0.5/0.3/0.2	$[0.5, 0.3, 0.2]$	0.980	0.480	200
skewed 5-way	$[0.40, 0.30, 0.15, 0.10, 0.05]$	1.000	0.600	200
uniform 1–5	$[0.2, 0.2, 0.2, 0.2, 0.2]$	0.945	0.743	200

Table 11: Seven-target panel results. The model produces a single answer in over 94% of calls on every non-point target.

Mean $P(\text{mode})$ across the six non-point targets is 0.987.

Swap test. Repeating the skewed binary target with options A and B swapped:

Condition	A	B	mode
$[A = 0.7, B = 0.3]$	200/200	0/200	A
$[A = 0.3, B = 0.7]$ (swap)	0/200	200/200	B
no-instruction baseline	200/200	0/200	A

Table 12: Swap test on skewed binary. The model relocates its single output but never produces a distribution.

The swap relocates the modal output (A $\rightarrow$ B) but never produces the $[0.3, 0.7]$ shape; the model treats the instruction as identifying a single answer.

F Logit-level results

At temperature 0 with `top_logprobs=20`, restricted to relevant outputs and renormalized. Top-2 logit gap (in nats) on the four non-point targets, with and without the distribution instruction.

Target	no instruction	with instruction	widening
skewed binary $[0.7, 0.3]$	6.50	14.75	+8.25
three-way $[0.5, 0.3, 0.2]$	—	3.75	—
bimodal $[0.5, 0, 0, 0, 0.5]$	—	9.21	—
skewed 5-way $[0.40, 0.30, 0.15, 0.10, 0.05]$	0.00	≥ 16	$\geq +16$

Table 13: Top-2 logit gap (nats) on non-point targets at $T = 0$. Adding the distribution instruction widens the gap rather than narrowing it.

Within-support $P(\text{mode})$ at $T = 0$ across all four with-instruction conditions: ≥ 0.976. The skewed 5-way case requires sampling temperature $\geq 16/\log(0.40/0.30) \approx 56$ to recover the target, far above the API maximum of 2.

G RNG primitive across formats

Five conditions probing the model's ability to produce randomness directly, no persona or task framing. $N = 200$ calls per condition at $T = 1.0$.

Condition	Support	Modal value	P(mode)	Coverage	p vs. uniform
R1 int 1–100	$\{1, \ldots, 100\}$	42	0.775	15/100	$\chi^2 = 11{,}971$, $< 10^{-300}$
R2 int 1–10	$\{1, \ldots, 10\}$	7	0.970	4/10	$\chi^2 = 1{,}683$, $< 10^{-300}$
R3 float [0,1]	$[0, 1]$	mean 0.552, std 0.108, range [0.27, 0.76]	n/a		KS $D = 0.370$, 5×10^{-25}
R4 coin H/T	$\{H, T\}$	H	0.854	2/2	binomial, 2×10^{-25}
R5 letter A–Z	$\{A, \ldots, Z\}$	G	0.465	16/26	$\chi^2 = 1{,}183$, 7×10^{-234}

Table 14: RNG primitive across five output formats. Every format rejects uniformity at $p < 0.001$; the model concentrates on a format-specific favorite.

Exclusion conditions. For each of the three discrete cases, we re-ran the probe with the modal value excluded by name. The model honored the exclusion in every case (no violations) but relocated mass onto a different favored value.

Condition	Violations	New mode	P(new mode)	Coverage (remaining)	TV vs. uniform
C1 exclude 42	0/200	37	0.520	20/99	0.854
C2 exclude 7	0/200	3	0.710	7/9	0.663
C3 exclude G	0/200	K	0.155	22/25	0.485

Table 15: Exclusion conditions: the model honors the exclusion but relocates mass onto a different favorite. The collapse is structural, not lexical.

H Correction-strategy prompts (V0–V9)

DOES interventions (V0–V7).

- **V0 plain sampling.** The bare task prompt at $T = 1.0$, no correction.
- **V1 verbatim feedback.** "You have been outputting only one answer; please vary your outputs to reflect the target distribution."
- **V2 apologetic framing.** Phrased as an acknowledgment of past failure ("I have noticed I keep giving the same answer. Let me try to do better.").
- **V3 third-person framing.** The task is framed as describing what a hypothetical person would do.
- **V4 explicit bias-naming.** The prompt names mode collapse directly and instructs the model to counteract it.
- **V5 in-context demonstration.** 20 shuffled exemplars matching the target distribution are provided before the task prompt.
- **V6 step-by-step reasoning.** Standard chain-of-thought scaffolding before the answer.
- **V7 algorithmic chain-of-thought.** Three visible steps: state the cumulative distribution, draw a uniform random $u \in [0, 1]$, output the option whose cumulative range contains u.

KNOWS interventions (V8–V9).

- **V8 list-of-N.** A single call returns N exemplars whose empirical distribution should match the target.
- **V9 parametric specification.** A single call returns the target distribution as JSON probabilities; an external library performs the sampling.

I KNOWS/DOES per-task results

TV-to-target for each intervention $\times$ task. $N = 200$ calls per cell at $T = 1.0$ on `gpt-4o`. Full intervention text is in Appendix H.

Variant	uniform 1–5	fair coin	skewed binary	bimodal 5-way	skewed 5-way
V0 plain sampling	0.743	0.475	0.300	0.500	0.600
V1 verbatim	0.655	0.305	0.300	0.600	0.500
V2 apologetic	0.685	0.295	0.300	0.600	0.500
V3 third-person	0.590	0.500	0.300	0.600	0.500
V4 explicit-bias	0.580	0.500	**0.125**	0.780	0.600
V5 20-shot ICL	0.650	0.485	0.300	0.600	0.460
V6 reasoning	0.700	0.110	0.300	0.520	0.500
V7 CoT-explicit-prob	0.700	0.435	0.300	0.460	0.490
V8 list-of-N=20	**0.000**	**0.045**	**0.050**	**0.050**	**0.000**
V9 sampler-spec	**0.045**	**0.020**	**0.010**	**0.025**	**0.035**

Table 16: TV-to-target for every intervention $\times$ task. Bold cells indicate TV ≤ 0.05 (the KNOWS threshold). Every KNOWS cell is below the threshold; no DOES cell is.

The separation is clean at the per-task level, not just on the across-task mean reported in §3.5. Across $8 \times 5 = 40$ DOES cells, the minimum TV is 0.110 (V6 on the fair coin) and only 2/40 fall below 0.20, both on the skewed binary where the baseline is already close to target. Across the $2 \times 5 = 10$ KNOWS cells, the maximum TV is 0.050, and 5/10 are below 0.025. The gap is not a property of any single task: every task contributes the same qualitative ordering, with both KNOWS variants strictly dominating all eight DOES variants on that task.

J CoT-RNG bridge analysis

We test whether V7's failure inherits from the degenerate primitive of §3.3 by parsing the model's stated u values from the V7 traces. 1000 responses across 5 tasks (200 each), 0 parse failures.

Task	u mean	u std	u range	KS vs $U[0,1]$
uniform digit	0.465	0.166	[0.12, 0.87]	$p = 2.3 \times 10^{-13}$
fair coin	0.451	0.185	[0.12, 0.74]	$p = 1.4 \times 10^{-17}$
skewed binary	0.460	0.059	[0.23, 0.65]	$p = 6.4 \times 10^{-31}$
bimodal 5-way	0.349	0.144	[0.12, 0.82]	$p = 9.3 \times 10^{-47}$
skewed 5-way	0.370	0.107	[0.12, 0.65]	$p = 2.7 \times 10^{-33}$

Table 17: V7 CoT-internal u distribution per task. u is non-uniform on every task.

Bridge $u \to$ answer agreement. For each task we apply the inverse-CDF map to the model's stated u and ask whether the resulting answer matches the answer the model actually produced. Table 18 reports the per-task agreement rate.

Task	bridge agreement
uniform digit	37.5%
fair coin	99.5%
skewed binary	100.0%
bimodal 5-way	97.5%
skewed 5-way	20.0%

Table 18: Bridge $u \to$ answer agreement per task. Three tasks execute the inverse-CDF mapping on a restricted u; two write u and ignore it.

K V8 N-sweep

V8 wraps the same query in a single call that requests a list of N samples. We sweep $N \in \{1, 2, 5, 10, 20\}$ on four categorical tasks (40 calls per cell at $T = 1.0$, `gpt-4o`) to test whether the list-construction effect depends on list length.

At $N = 1$, V8 collapses to ordinary per-call sampling and the failure returns in full: every task shows TV ≥ 0.30, matching the V0–V7 range. Going from $N = 1$ to $N = 2$ already cuts TV by roughly half on three of four tasks. By $N = 5$ every task is below 0.20, and by $N = 20$ four of four are at or near zero. The transition is sharp rather than gradual, which is consistent with the interpretation in §3.5: V8 is not benefiting from more independent draws (which would predict $1/\sqrt{N}$ shrinkage in sampling noise but no change in mean), it is switching modes from per-call sampling to single-call counting as soon as the model has enough slots to lay out a recognizable distributional shape.

Task	N=1	N=2	N=5	N=10	N=20
uniform digit	0.60	0.24	0.00	0.01	0.01
fair coin	0.45	0.00	0.06	0.00	0.00
skewed binary	0.30	0.06	0.09	0.05	0.08
skewed 5-way	0.50	0.30	0.20	0.01	0.01

Table 19: V8 TV-to-target across list lengths. V8 reverts to single-answer failure at $N = 1$; by $N = 5$ every task is below 0.20.

L PPA ablation

The PPA ablation uses a 13-item pre-registered development subset of the 100 OpinionQA items. The subset was selected before any PPA results were computed (item keys committed in the released code). We use this smaller set for ablations because each variant requires a separate run with 100 personas per item; the development subset keeps the API budget tractable while preserving the same persona protocol as the main experiment.

Condition	Mean TV	Δ	% of PPA
(a) Single-knob ablation			
A0 baseline	0.481	—	—
A1 order-only	0.416	−0.064	60%
A2 phrasing-only	0.422	−0.059	56%
(b) Cumulative ablation			
A0 baseline	0.481	—	—
+ order	0.416	−0.064	60%
+ order + phrasing	0.375	−0.105	99%
+ order + phrasing + position	0.374	−0.106	100%
(c) Non-semantic perturbation control			
N0 baseline	0.489	—	—
N1 whitespace-only	0.485	−0.004	4%
N2 punctuation-only	0.481	−0.008	8%
N3 prefix-only	0.476	−0.013	13%
N4 all three combined	0.475	−0.014	13%
Full PPA (reference)	0.374	−0.106	100%

Table 20: PPA ablation on the 13-item development subset. (a) Each knob alone gives comparable effect size. (b) Position adds negligible value beyond order and phrasing. (c) Non-semantic perturbations capture only 13% of PPA's TV reduction. Numerator and denominator of the "% of PPA" column are both computed on this subset.

M PPA per-item results

Per-cell breakdown (2×2)**.** Table 21 reports baseline TV, PPA TV, and the paired Wilcoxon test within each of the four pre-registered (domain × entropy) cells.

Cell	n	Baseline TV	PPA TV	Δ	Wilcoxon p
FACTS × HI_ENT	15	0.421	0.316	−0.104	0.002
FACTS × LO_ENT	21	0.480	0.325	−0.155	3.6×10^{-4}
VALUES × HI_ENT	27	0.504	0.448	−0.056	0.007
VALUES × LO_ENT	37	0.430	0.337	−0.093	1.3×10^{-4}

Table 21: Pre-registered 2×2 cell breakdown. Every cell improves; both factual cells confirm at $p < 0.01$.

95% bootstrap CIs on Δ (1000 resamples within each cell): FACTS-HI [−0.16, −0.04]; FACTS-LO [−0.22, −0.09]; VALUES-HI [−0.09, −0.02]; VALUES-LO [−0.14, −0.05].

Representative items. Table 22 lists up to five items from each of the four cells, ranked by the size of PPA's improvement, to illustrate where the gains concentrate.

Item	Wave	Domain	Entropy	Baseline TV	PPA TV	Δ
NEIGHINTERA	W32	F	HI	0.674	0.384	−0.290
WORRY2a	W54	F	HI	0.493	0.212	−0.281
WORRY2d	W54	F	HI	0.401	0.196	−0.206
TRACKCO1b	W49	F	HI	0.564	0.414	−0.150
PP1	W49	F	LO	0.682	0.194	−0.488
MADEUPSMCLICK	W45	F	LO	0.609	0.243	−0.366
FAMNEAR	W32	F	LO	0.637	0.323	−0.314
NEIGHSAMEB	W32	F	LO	0.466	0.166	−0.300
ECON5_i	W54	V	HI	0.654	0.381	−0.273
RACESURV5b	W43	V	HI	0.678	0.448	−0.230
RACESURV47d	W43	V	HI	0.568	0.416	−0.152
LEGALIMMIGAMT	W92	V	HI	0.464	0.314	−0.150
RESTRICTWHO	W45	V	LO	0.874	0.255	−0.619
GAP21Q15_f	W82	V	LO	0.699	0.369	−0.330
RACESURV41	W43	V	LO	0.563	0.253	−0.310
RACESURV5a	W43	V	LO	0.357	0.074	−0.283

Table 22: Representative items, ranked by largest PPA improvement. F = factual; V = values; HI/LO = entropy bin.

N Baseline sweeps

Decoding-parameter sweeps on the same 13-item pre-registered development subset used in Appendix L, 100 personas per item.

T	Mean TV ($n = 13$)	Δ vs. $T = 1.0$
0.7	0.490	+0.011
1.0	0.479	0 (baseline)
1.3	0.471	−0.008
1.5	0.463	−0.016

Table 23: Temperature sweep. Best $T = 1.5$ captures $\approx 17\%$ of PPA's gain.

p	Mean TV ($n = 13$)	Δ vs. $p = 1.0$
0.5	0.490	+0.001
0.7	0.493	+0.004
0.9	0.482	−0.007
1.0	0.489	0 (baseline)

Table 24: Top-p sweep. Best $p = 0.9$ captures $\approx 7\%$ of PPA's gain.

Neither sweep approaches PPA's effect size; the dominant factor in PPA's improvement is the semantic perturbation, not the decoding distribution.

O PPA mechanism: non-semantic controls and temperature analogue

This appendix details the two follow-up tests reported in §5.3. The aim is to ask whether the entropy-correlation result reflects the mechanism we claim — that PPA helps by raising per-call entropy — or some alternative such as "any prompt variation helps" or "the gain is specific to semantic perturbation."

Non-semantic controls. On the same 20-item subset, we compare PPA's per-call entropy increase

against four non-semantic perturbation conditions matched to PPA in structure but not in content: whitespace variation (N1), punctuation variation (N2), prefix variation (N3), and all three combined (N4). Each non-semantic condition perturbs surface form across 50 calls per (persona, item) cell without changing option ordering or question phrasing.

Condition	Mean H (nats)	ΔH vs. std	ΔTV
Standard Argyle	0.146	—	—
PPA	0.562	$+0.417$	-0.106
N1 whitespace	0.146	$+0.000$	-0.004
N2 punctuation	0.138	-0.007	-0.008
N3 prefix	0.157	$+0.011$	-0.013
N4 all three	0.149	$+0.003$	-0.014

Table 25: Per-cell entropy and TV-to-Pew on the 20-item subset under PPA and four non-semantic perturbation conditions. PPA raises per-call entropy by 0.417 nats; the four non-semantic conditions average $\Delta H = 0.002$.

The four non-semantic conditions raise per-call entropy by essentially zero on average ($\Delta H = 0.002$), two orders of magnitude smaller than PPA's $\Delta H = 0.417$, and yield TV improvements of -0.004 to -0.014 — about 13% of PPA's effect size. The TV improvement tracks the entropy increase, not the presence of prompt variation.

Temperature analogue. We compare PPA's $\Delta\text{TV}/\Delta H$ slope against an independent intervention that also raises per-call entropy: decoding temperature. Using the temperature sweep of Appendix N, $T = 1.5$ raises per-call entropy by $\Delta H = 0.069$ and reduces TV by $\Delta\text{TV} = 0.016$. The slope $\Delta\text{TV}/\Delta H$ is 0.239 for temperature and 0.255 for PPA — within 7% of each other.

Intervention	ΔH (nats)	ΔTV	$\Delta\text{TV}/\Delta H$
PPA	0.417	-0.106	0.255
$T = 1.5$	0.069	-0.016	0.239

Table 26: $\Delta\text{TV}/\Delta H$ slope is nearly identical for PPA and temperature increase. Per nat of per-call entropy added, both interventions remove roughly the same amount of TV.

Two interventions with very different mechanics — semantic prompt perturbation and decoding-temperature increase — produce essentially the same TV improvement per nat of entropy added. Combined with the non-semantic null result, this is consistent with an entropy-generic mechanism: any intervention that raises per-call entropy by the same amount produces proportional aggregate improvement, regardless of whether the entropy comes from prompt-level or decoding-level variation.

Bootstrap stability. The entropy-correlation regression of §5.3 uses $N = 20$ items. We assess the robustness of the $\beta_{\Delta H}$ coefficient by resampling with replacement 1000 times. The coefficient is positive in 1000/1000 resamples (mean 0.270, 95% bootstrap CI $[0.176, 0.373]$). Hold-out R^2 from 1000 random 50/50 splits has mean 0.095 (median 0.305), with 81.4% of splits showing $R^2 > 0$. The point estimate of the coefficient is robust; out-of-sample R^2 is unstable, which is expected at this sample size and does not affect the directional claim.

P V8 internal structure

To check whether V8 is producing genuinely independent draws or constructing a parametric description, we examine three properties of its $N = 20$ outputs across 200 calls per task (§3.5). Under independent sampling (Hypothesis A), within-call lag-1 autocorrelation should be near zero, cross-call sequence-rank correlation should be at chance, and run-length statistics should match a binomial null. Under construction (Hypothesis B), at least one of these signatures should depart sharply from those predictions.

Task	Within-call lag-1 autocorr	Cross-call seq-rank ρ	Run-length ratio	Hypothesis
uniform digit	-0.221	1.000	0.820	B (constructs)
fair coin	-0.494	1.000	0.665	B (constructs)
skewed binary	-0.039	1.000	1.213	B (constructs)
bimodal 5-way	$+0.282$	0.995	2.596	B (constructs)
skewed 5-way	$+0.550$	0.966	2.436	B (constructs)

Table 27: V8 internal structure. Every task assigns to construction (Hypothesis B) by cross-call sequence-rank correlation ≥ 0.97.

The cross-call sequence-rank correlation is the cleanest diagnostic: across 200 independently sampled calls, the ordering of the 20 items inside each call agrees with the ordering in every other call at $\rho \geq 0.97$. Independent draws cannot produce that. Within-call autocorrelation also departs from the independence null in opposite directions on different tasks (strongly negative on the fair coin, where the model alternates H-T-H-T; strongly positive on the skewed 5-way, where the model groups runs of repeated values that match the target proportions). Both signatures point to V8 constructing a single fixed sequence each call rather than drawing 20 independent samples.

Q Cross-family replication

To address the single-model scope of the main paper, we replicate the structural claims on five additional instruction-tuned model families; $N = 200$ calls per cell at $T = 1.0$.

Models. `openai/gpt-5.4`, `anthropic/claude-sonnet-4.5`, `meta-llama/llama-3.3-70b-instruct`, `google/gemini-3-flash`, `deepseek/deepseek-chat` (V3.2).

RNG probe. Every instruction-tuned model has a degenerate RNG primitive; the specific favored value is family-dependent.

Model	int 1–100 fav.	P(mode)	coverage	KS p on float $[0, 1]$
`gpt-4o` (main paper)	42	0.775	15/100	5×10^{-25}
`gpt-5.4`	73	0.110	59/100	$< 10^{-30}$
`claude-sonnet-4.5`	47	1.000	1/100	$< 10^{-30}$
`llama-3.3-70b`	53	0.890	4/100	$< 10^{-30}$
`gemini-3-flash`	42	0.910	8/100	$< 10^{-30}$
`deepseek-chat`	1	0.330	18/100	$< 10^{-30}$

Table 28: Cross-family RNG probe. Every instruction-tuned model concentrates on a family-specific favorite.

Seven-target panel. Table 29 reports P(mode) on each of the seven targets from the main paper across the five additional model families. Every model concentrates at $P(\text{mode}) \geq 0.46$ on at least one non-point target; Claude-Sonnet-4.5 hits $P(\text{mode}) = 1.0$ on all seven.

Target	gpt-5.4	claude-4.5	llama-70b	gemini-3	deepseek
point 5	1.000	1.000	1.000	1.000	1.000
mixture 5/4	0.860	1.000	1.000	1.000	0.887
skewed binary	0.770	1.000	0.830	0.990	0.990
bimodal 1/5	0.650	0.950	0.945	0.775	0.792
three-way	0.945	1.000	0.985	0.663	0.720
skewed 5-way	0.500	1.000	0.985	0.610	0.870
uniform 1–5	0.460	1.000	0.940	0.610	0.860
≥ 0.90 count	2/7	7/7	6/7	3/7	2/7

Table 29: Cross-family seven-target panel: P(mode) on each target.

KNOWS/DOES dichotomy. Tables 30 and 31 report V8 (KNOWS pathway) and V0 (plain DOES) TV-to-target across the five families. V8 succeeds across all 25 (model, task) cells (max TV = 0.050), while V0 fails on 21 of 25; the split mirrors the gpt-4o pattern of §3.5.

Summary. The structural claims generalize qualitatively across five instruction-tuned families: every model has a degenerate RNG primitive (distinct favored values per family), every model exhibits categorical or strongly attenuated collapse on most non-uniform targets, and V8 succeeds across all 25 (model, task) cells. Combined with the base-vs-instruct comparison of §3.4, this is consistent with alignment training as the proximate cause of the per-call sampling collapse in deployment-relevant models.

	V8 TV-to-target				
Task	gpt-5.4	claude-4.5	llama-70b	gemini-3	deepseek
uniform digit	0.000	0.000	0.008	0.000	0.000
fair coin	0.000	0.050	0.000	0.000	0.000
skewed binary	0.000	0.000	0.000	0.000	0.000
bimodal 5-way	0.000	0.000	0.015	0.000	0.000
skewed 5-way	0.000	0.000	0.000	0.000	0.000

Table 30: V8 KNOWS TV-to-target across families. V8 succeeds universally: max TV = 0.050.

Task	gpt-5.4	claude-4.5	llama-70b	gemini-3	deepseek
uniform digit	0.295	0.800	0.735	0.600	0.630
fair coin	0.195	0.500	0.269	0.055	0.455
skewed binary	0.130	0.300	0.140	0.295	0.300
bimodal 5-way	0.500	0.240	0.600	0.320	0.220
skewed 5-way	0.350	0.500	0.630	0.490	0.365

Table 31: V0 (plain DOES) baseline TV per (model, task). V0 fails on 21 of 25 cells (TV > 0.20).