Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Where Do Language Models Hide the Right Answer?

Bhavishay Sheoran
Indian Institute of Technology Roorkee https://orcid.org/0009-0000-5298-5217
Mayank Goswami
mayankfph@iitr.ac.in
Indian Institute of Technology Roorkee https://orcid.org/0000-0001-8829-9980

Research Article

Keywords: large language models, answer position bias, multiple-choice question generation, clustered data, generalised estimating equations, prompt robustness, educational assessment

Posted Date: November 17th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11739168/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Where Do Language Models Hide the Right Answer?

Answer-Position Bias in LLM-Generated Multiple-Choice Questions Across Families, Versions, Decoding and Prompts

Bhavishay Department of Physics, IIT Roorkee
Email: bhavishay1@ph.iitr.ac.in
Mayank Goswami Department of Physics, IIT Roorkee
Email: mayank.goswami@ph.iitr.ac.in [1]

[1]Department of Physics, IIT Roorkee
[2]MF School of AI and DS, IIT Roorkee
mayank.goswami@ph.iitr.ac.in

***Abstract*—Students increasingly practise with multiple-choice questions (MCQs) written by large language models (LLMs), yet a quiz measures knowledge only if the correct answer is equally likely to appear in every position. A recent preprint showed that LLMs place answers non-uniformly under deterministic decoding. We ask a narrower and measurable question: *does answer-position bias carry over to a model family and to model versions outside that study, does it survive the sampling and prompting choices of ordinary use, and can it be removed by asking for a balanced key?* We generated 2,819 MCQs, 2,109 of them with a standard prompt, from `gemini-3-flash-preview`, `gpt-oss-120b` and `qwen3.8-27b`, varying temperature (0–1), questions per request (5–25), option count (4 or 5) and prompt wording, and analysed them with the 293 generation requests, not the questions, as the independent units. Every model under-uses the last position: the pooled empirical share is 6.1% (128/2,109) against a fair 20–25%. The favoured position is stable within a family across versions (Qwen: B; Gemini: B/C) and different in the new family (gpt-oss: A). Temperature, batch size, option count and wording do not remove the bias. A one-sentence balancing instruction removes it for two models but not for Qwen, and for gpt-oss it introduced mis-keyed items (7.2%). A guesser who learns the favoured position from other requests answers 38–57% of unknown items correctly, against 25% by chance. We state explicitly which findings are new, which reproduce prior work and which qualify it.**

***Index Terms*—large language models, answer-position bias, multiple-choice question generation, clustered data, generalised estimating equations, prompt robustness, educational assessment**

I. Introduction

AN LLM can write a 50-question practice quiz in seconds, and students use such quizzes to judge their own readiness. That judgement is sound only if the answer key carries no positional signal: with four options, each position should hold the correct answer with probability 1/4. This study began with a midterm-preparation bank in which 109 of 151 single-answer items (72%) were keyed B.

Prior work has established the basic phenomenon. LLMs that *answer* MCQs prefer some option letters regardless of content [3], [4], and a 2026 preprint [1] showed that LLMs that *generate* MCQs also place answers non-uniformly. It tested ten text and five vision-language models under deterministic decoding, found family-specific favourites with D under-represented, showed that answer positions can be decoded from hidden states, and reported that asking for a balanced key reduces the bias, especially in larger models.

Showing again that the bias exists would add little. The open question is whether it *transfers*. Practitioners use models and versions that no study has measured, sample at non-zero temperature, request questions in batches, and phrase prompts in their own words. A finding that holds only for the models and decoding settings of one study does not tell a teacher whether the next quiz they generate is safe to use.

What is new. This paper tests transfer along four axes and is explicit about which results extend prior work (Table I).

- *Across families and versions.* We add a family absent from [1] (gpt-oss) and newer versions of two families it studied (Gemini 3 Flash; Qwen 3.8). The direction of the favourite is preserved within each family across versions, while the new family has a different favourite, and every model avoids the last slot.
- *Across decoding and request settings.* We measure the bias at $T \in \{0, 0.7, 1\}$, with 5, 10 and 25 items per request and with 4 or 5 options, and test for serial dependence within a request.
- *Across prompts and under mitigation.* We test an independently worded prompt and the balancing instruction on all three models.
- *Consequence and inference.* We estimate the exploitable advantage out of sample. All inference treats the generation request as the sampling unit (request-level bootstrap, Rao–Scott correction, permutation and generalised estimating equations).

TABLE I
COMPARISON WITH THE CLOSEST PRIOR WORK AND WHAT THIS WORK ADDS

Aspect	Preprint [1]	This work	**What this work adds**
Model families	Llama, Mistral, Qwen3, DeepSeek, Gemini (2.5 Flash, 3 Pro); 10 text + 5 vision-language models	Gemini 3 Flash, Qwen 3.8 (27B), gpt-oss (120B); 3 text models	A family not in [1] (gpt-oss) and newer versions of two studied families: tests transfer across families and versions
Items	Three task sets (knowledge MCQs, reading comprehension, visual QA), 200–480 items per task	2,819 knowledge MCQs (2,109 standard), 5 subjects, 22 conditions, exact n and request counts	Request-level dataset with documented exclusions, released with one reproducible script
Decoding	Deterministic only	$T = 0$, 0.7, 1.0	First measurement under sampling; bias persists at all T; $T = 0$ sharpens the favourite (exploratory)
Request settings	Not varied	5, 10, 25 items per request; 4 vs. 5 options; within-request order	No batch-size effect; no detectable serial dependence; avoidance follows the last *position*, not the letter D
Prompt wording	One template	Standard and independently reworded instruction	Bias unchanged by rewording ($V = 0.07$): not a template artefact
Research questions	Is generation biased? Is position planned internally? Can steering or prompting change it?	Does the bias transfer across families, versions, decoding, request settings and wording? Does balancing remove it? How exploitable is it?	Shifts the question from *existence* to *transfer* and *practical consequence*
Statistical methods	Distributional comparison; hidden-state probing; activation steering	Request-clustered inference: cluster bootstrap, Rao–Scott χ^2, request-level permutation, GEE, Holm; effect sizes w, V, OR	Inference that treats requests, not items, as independent, with exact denominators, effect sizes and adjusted p-values
Mitigation	Balanced prompting; label removal; activation steering	Balanced prompting on all three models	Balancing *fails* for Qwen ($w = 0.49$): instruction-level fixes are model-dependent
Practical impact	Not measured	Out-of-sample exploit with request-level splits	A guesser scores 38–57% vs. 25%; up to +32 marks per 100 items
Key findings	Family-specific favourites; D rare; position decodable from hidden states; balancing helps larger models	Last slot rare in every model (6.1%); favourite stable within families across versions; robust to T, batch, k, wording	See Table III for the status of each finding (new / reproduces / qualifies)

II. Background

Human item writers also avoid the edge positions, and guessing test-takers favour the middle [2]. LLMs trained on human text may inherit this habit. At each step an LLM samples token i with probability $P(i) = e^{z_i/T} / \sum_j e^{z_j/T}$, a Boltzmann form with logits z_i and temperature T. As $T \to 0$ the sampler selects the argmax; as $T \to \infty$ it approaches uniform.

III. Methodology

A. Data Generation

Questions were generated through the official APIs on 10–15 October 2026: `gemini-3-flash-preview` (Google GenAI API), and `openai/gpt-oss-120b` and `qwen3.8-27b` (both served by Groq). Each *request* was an independent call with no history, asking for n university-level MCQs on one topic (physics, biology, world history, geography or mathematics, rotated) with exactly k labelled options and one correct answer, returned as JSON. Two prompt variants were used: an *independently worded* instruction, and the standard prompt plus "Distribute the correct answers randomly and evenly across A, B, C, D" (*balanced*). Requests that failed JSON validation were discarded. Table II gives the conditions.

TABLE II
EXPERIMENTAL CONDITIONS (ITEMS n; REQUESTS m)

Model	Prompt	T	Items/request, k	n	m
Gemini	standard	0, .7, 1	10, 4	570	57
gpt-oss	standard	0, .7, 1	10, 4	600	60
gpt-oss	standard	.7	5 / 25, 4	349	46
gpt-oss	standard	.7	10, 5	190	19
Qwen	standard	0, .7	10, 4	400	40
gpt-oss	reworded	.7	10, 4	201	20
all three	balanced	.7	10, 4	509	51
Total				2,819	293

No items were excluded for an invalid answer letter.

B. Statistical Analysis

Items generated in one request share topic, context and sampling state, so they need not be independent. We therefore treat the *request* as the sampling unit throughout. Let $O_{c,i}$ be the number of items in request c keyed to position i, with n_c items in the request, m requests and $N = \sum_c n_c$ items in total.

a) Shares and intervals: The pooled share $\hat{p}_i = \sum_c O_{c,i}/N$ is reported with exact denominators. Its 95% interval comes from a *request-level (cluster) bootstrap* with 2,000 resamples of whole requests, using the percentile method. For comparison, we also give the exact binomial (Clopper–Pearson) interval [5], which assumes independent items.

b) Uniformity: The Pearson statistic $X^2 = \sum_i (O_i - N/k)^2/(N/k)$ is divided by the mean generalised design effect $\hat{\delta} = \max\big(1, \frac{1}{k-1}\sum_i k\hat{p}_i(1-\hat{p}_i)d_i\big)$, where d_i is the ratio of the linearised cluster variance of $\hat{p}_i$ to its binomial variance (first-order Rao–Scott correction), and the result is referred to χ^2_{k-1}. The effect size is Cohen's $w = \sqrt{X^2/N}$.

c) Factor effects: Temperature, batch size and prompt are tested by χ^2 homogeneity with p-values from 2,000 permutations of *whole requests* between conditions. Effect sizes are given as Cramér's V. Serial dependence is tested by permuting item order within each request.

d) Regression: For $Y \in \{\text{last}, \text{first}\}$ we fit a logistic GEE with an exchangeable working correlation within requests and covariates model, T, batch size, k and topic. We report odds ratios (OR) with robust 95% CIs. The intercept is the log-odds for the *reference cell* (Gemini, $T = 0$, 5 items per request, $k = 4$, reference topic). Because Gemini was never run with five items per request, this cell lies outside the design, and the intercept is not interpreted as a population rate. Population rates are given by the empirical pooled shares.

e) Multiplicity: p-values are Holm-adjusted within each family of tests: uniformity, factor effects, and exploratory analyses.

f) Reproducibility: One documented script, with fixed seed 20261027, rebuilds every table from the raw request logs and records exclusions, software versions and input files.

C. Validity of Answer Keys

Thirty randomly sampled Gemini standard-prompt items were checked by hand, and all 30 keys were correct (Clopper–Pearson 95% CI $[0.884, 1]$). A second random sample of 30 items (15 Qwen standard, 15 gpt-oss balanced; seed 7) was checked by an LLM assistant with web verification of disputed facts. All 15 Qwen keys were correct; 13 of 15 gpt-oss balanced keys were correct (28/30 overall, 95% CI $[0.78, 0.99]$). Because both errors came from the balanced prompt, all 209 gpt-oss balanced items were then reviewed the same way (Section IV-G).

IV. Results

A. Status of Each Finding

Table III lists every finding and its relation to prior work. The remaining subsections give the evidence.

B. Every Standard-Prompt Condition Is Non-Uniform

Table IV reports position shares for each condition. The item-level χ^2 rejects uniformity everywhere ($p < 10^{-5}$), with large effects ($w = 0.43$–0.93). Accounting for clustering does not change this: the request-level Rao–Scott p-values remain below 10^{-4} in every condition, also after Holm adjustment across the 15 uniformity tests. Estimated design effects were small (0.4–1.8), consistent with the near-zero within-request correlation in Table V. The two motivating banks, (18, 109, 22, 2) and (12, 44, 40, 4), have no request structure and are reported only as motivation.

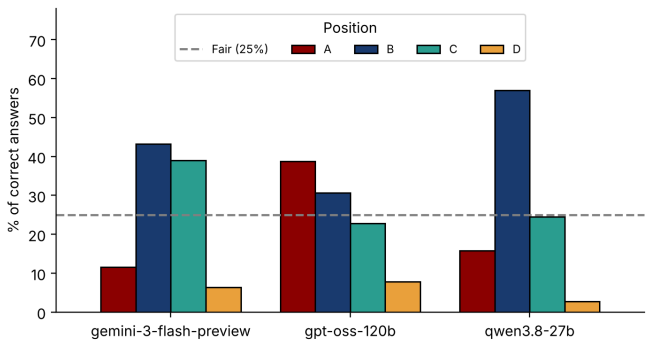


Fig. 1. Answer-position shares per model (standard prompt, four options, all conditions pooled). The dashed line marks the fair rate.

C. Transfer Across Families and Versions

Pooled over standard-prompt conditions (Fig. 1), the empirical last-position share is 6.3% for Gemini (36/570; request-bootstrap 95% CI 4.4–8.4), 7.8% for gpt-oss (74/949; 6.1–9.4) and 2.8% for Qwen (11/400; 1.2–4.2). Over all 2,109 standard items the share is 6.1% (128/2,109; binomial 95% CI 5.1–7.2), against a fair 20–25%.

The favourite's direction is preserved within each family across versions. Qwen3 favoured B in [1], and Qwen 3.8 places 57.0% of answers in B. Gemini 2.5 Flash and 3 Pro favoured B/C, and Gemini 3 Flash places 82.1% of answers in B or C. The new family behaves differently: gpt-oss places 38.8% of answers in A. Because the prompts of the two studies differ, this cross-study comparison is qualitative.

Table V reports the request-clustered GEE models. For the last position, no factor survives Holm adjustment: temperature, items per request and option count have no detectable effect, and Qwen's lower odds (OR 0.45, $p = 0.014$) and the lower odds with five options (OR 0.48, $p = 0.045$) are only nominally significant. For the first position, gpt-oss has 5.3 times Gemini's odds ($p < 10^{-4}$ after Holm). The estimated within-request correlations are close to zero ($\hat{\rho} = -0.004$ and 0.047), so clustering changes the inference only slightly. As explained in Section III-B, the intercepts (0.12 and 0.06) refer to a reference cell outside the design and are not population rates.

D. Decoding Temperature

Every distribution remains non-uniform at every temperature (Fig. 2). The pre-specified homogeneity tests across temperatures are small and not significant: Gemini $\chi^2 = 4.35$, df $= 6$, $p = 0.63$, $V = 0.06$; gpt-oss $\chi^2 = 12.49$, df $= 6$, $p = 0.052$, $V = 0.10$; Qwen $\chi^2 = 7.49$, df $= 3$, $p = 0.058$, $V = 0.14$ (request-permutation $p = 0.57$, 0.17 and 0.12). In an exploratory analysis, the share of each model's favourite is higher at $T = 0$ than at $T > 0$: Gemini B 46.5% vs. 41.4%, gpt-oss A 49.5% vs. 35.3%, Qwen B 63.5% vs. 50.5%. To avoid selecting the favourite on the same data, it is defined from the $T > 0$ data only. The differences are +5.1 (95% request-bootstrap CI −3.0 to

TABLE III
FINDINGS AND THEIR RELATION TO PRIOR WORK [1]

Finding	Relation to [1]
Generated answer keys are non-uniform	Reproduces
Last position under-used (6.1%)	Reproduces; extended to a new family and to $k = 5$
Qwen favours B; Gemini favours B/C	Reproduces, on newer versions
gpt-oss favours A	New family
Bias survives $T = 0.7$ and 1.0	New
$T = 0$ sharpens each favourite	New; exploratory, not significant after Holm
Batch size has no effect; no serial dependence	New
Avoidance follows position, not letter	New; suggestive (D increase $p = 0.035$, not significant after Holm)
Bias unchanged by independent wording	New
Balancing removes bias for Gemini, gpt-oss	Reproduces
Balancing fails for Qwen	Qualifies ($p = 0.072$ for any change; still $w = 0.49$)
Balancing introduces mis-keyed items (gpt-oss 7.2%)	New
Balanced keys follow a predictable A→B→C→D rotation (gpt-oss 77.7% of pairs)	New
Out-of-sample exploit accuracy 38–57%	New

TABLE IV
CORRECT-ANSWER POSITION (%) UNDER THE STANDARD PROMPT. FAIR RATE 25% (20% FOR $k = 5$)

Model	T	Items/req.	k	n	A	B	C	D	E	w	p (item)	p (request, RS)	p (Holm)
Gemini	0.0	10	4	200	11.5	**46.5**	38.0	4.0	–	0.71	1.3×10^{-21}	1.6×10^{-21}	2.2×10^{-20}
Gemini	0.7	10	4	170	12.9	**40.0**	38.8	8.2	–	0.58	2.2×10^{-12}	2.2×10^{-12}	2.4×10^{-11}
Gemini	1.0	10	4	200	10.5	**42.5**	40.0	7.0	–	0.65	2.3×10^{-18}	2.3×10^{-18}	3.0×10^{-17}
gpt-oss	0.0	10	4	200	**49.5**	25.0	18.0	7.5	–	0.62	1.8×10^{-16}	2.0×10^{-9}	2.0×10^{-8}
gpt-oss	0.7	10	4	200	**35.0**	32.0	26.0	7.0	–	0.44	2.9×10^{-8}	7.2×10^{-6}	3.6×10^{-5}
gpt-oss	1.0	10	4	200	**35.5**	32.0	25.5	7.0	–	0.44	2.0×10^{-8}	2.0×10^{-8}	1.4×10^{-7}
gpt-oss	0.7	5	4	200	**38.5**	30.0	23.0	8.5	–	0.44	2.0×10^{-8}	4.0×10^{-8}	2.4×10^{-7}
gpt-oss	0.7	25	4	149	34.2	**35.6**	20.8	9.4	–	0.43	5.1×10^{-6}	1.8×10^{-5}	7.4×10^{-5}
gpt-oss	0.7	10	5	190	**38.9**	27.4	16.3	13.7	3.7	0.61	2.7×10^{-14}	4.8×10^{-9}	3.8×10^{-8}
Qwen	0.0	10	4	200	12.5	**63.5**	22.0	2.0	–	0.93	1.6×10^{-37}	1.6×10^{-37}	2.5×10^{-36}
Qwen	0.7	10	4	200	19.0	**50.5**	27.0	3.5	–	0.68	7.4×10^{-20}	4.1×10^{-15}	4.9×10^{-14}

TABLE V
REQUEST-CLUSTERED LOGISTIC GEE (EXCHANGEABLE, TOPIC-ADJUSTED): ODDS RATIOS WITH ROBUST 95% CIS ($N = 2{,}109$ ITEMS, 222 REQUESTS)

	Y: last option			Y: first option		
Factor	OR [95% CI]	p	p_{Holm}	OR [95% CI]	p	p_{Holm}
gpt-oss vs. Gemini	1.16 [0.72, 1.84]	0.54	1	**5.28** [3.78, 7.38]	$< 10^{-4}$	$< 10^{-4}$
Qwen vs. Gemini	0.45 [0.24, 0.85]	0.014	0.15	1.34 [0.86, 2.08]	0.19	1
Temperature (per unit)	1.33 [0.74, 2.39]	0.34	1	0.75 [0.51, 1.12]	0.16	1
10 vs. 5 items/req.	0.86 [0.53, 1.38]	0.53	1	1.03 [0.69, 1.52]	0.90	1
25 vs. 5 items/req.	1.13 [0.57, 2.23]	0.73	1	0.79 [0.47, 1.32]	0.37	1
$k = 5$ vs. $k = 4$	0.48 [0.23, 0.98]	0.045	0.45	0.99 [0.61, 1.63]	0.98	1
Within-request $\hat{\rho}$	−0.004			0.047		

13.5), +14.3 (1.2 to 27.3) and +13.0 (1.5 to 24.0) percentage points, with request-permutation $p = 0.18$, 0.034 and 0.050. The direction is the same in all three models, but none survives Holm adjustment ($p_{\text{Holm}} \geq 0.10$), so this result is suggestive only. This sharpening is what a softmax predicts when each question has its own scores (Section V-A).

E. Request Size and Serial Dependence

Requesting 5, 10 or 25 items does not change the distribution ($\chi^2 = 3.01$, df = 6, request-permutation $p = 0.85$, $V = 0.05$). Consecutive items in a request repeat a letter no more often than when the order within each request is shuffled (observed 0.275, 0.328, 0.252; permutation null 0.313, 0.333, 0.295; $p = 0.14$, 0.92, 0.25). This shows no detectable serial dependence, but it does not establish independence, which is why all inference above is request-level.

F. Position, Not Letter

With five options the share falls steadily from first to last (Fig. 3). The first-position share is unchanged (35.0% vs. 38.9%; difference +3.9 points, 95% CI −11.6 to 20.0, $p = 0.65$). D roughly doubles once it is no longer last (7.0%→13.7%; +6.7 points, CI 1.0 to 12.4, $p = 0.035$), and the new last option E is at least as rare as D was (3.7%

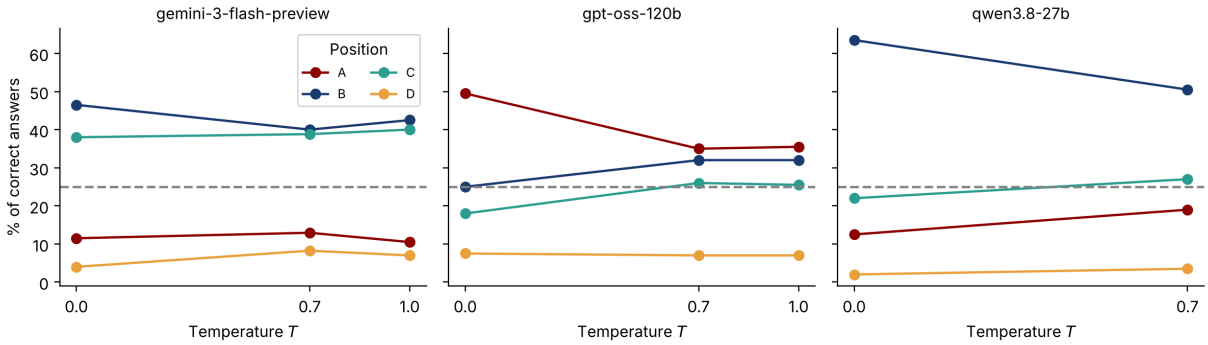


Fig. 2. Position shares versus temperature (four options, 10 items per request). The dashed line marks the fair rate.

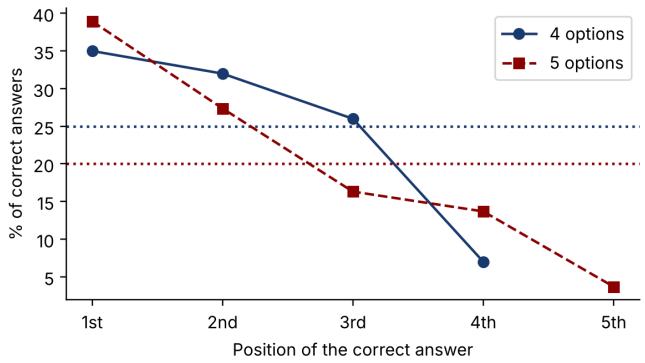


Fig. 3. gpt-oss at $T = 0.7$ with four and five options. Dotted lines mark the fair rates.

vs. 7.0%; −3.3 points, CI −7.4 to 0.8, $p = 0.19$). The D increase does not survive Holm adjustment ($p_{\text{Holm}} = 0.42$), so the position-not-letter interpretation is supported by the direction of all three contrasts but needs a larger sample to confirm.

G. Wording and Balancing

With the independently worded prompt, gpt-oss's distribution is unchanged ($\chi^2 = 1.70$, df $= 3$, request-permutation $p = 0.75$, $V = 0.07$; Table VI, Fig. 4). The balancing instruction changes Gemini's and gpt-oss's distributions ($V = 0.26$ and 0.21; request-permutation $p = 0.0005$, Holm $p = 0.007$ for both), and both become consistent with uniform ($w = 0.10$ and 0.14; $p = 0.79$ and 0.25). For Qwen the change is small and not significant at the request level ($V = 0.14$, $p = 0.072$), and the distribution remains strongly biased ($w = 0.49$, $p = 2.4 \times 10^{-9}$), still favouring B at 41.5%. The first result reproduces [1]; the Qwen result qualifies it, because instruction-level balancing is not a reliable fix across models.

Balancing can cost key correctness. A full review of the 209 gpt-oss balanced items found 15 (7.2%, 95% CI 4.1–11.6%) whose keyed letter pointed to a wrong option, for example "Treaty of Westphalia ended the Napoleonic Wars" and "the final defeat of Napoleon in 1815 was the Battle of Leipzig". The errors cluster in late requests,

TABLE VI
PROMPT VARIANTS ($T = 0.7$, 10 ITEMS PER REQUEST, $k = 4$)

Model	Prompt	n	A	B	C	D	w	p_{unif}	V_{std}	p_{std}
gpt-oss	standard	200	35.0	32.0	26.0	7.0	0.44	10^{-5}	–	–
gpt-oss	reworded	201	32.3	32.8	24.4	10.4	0.36	10^{-4}	0.07	0.75
Gemini	balanced	100	25.0	28.0	26.0	21.0	0.10	0.79	0.26	0.0005
gpt-oss	balanced	209	27.8	29.2	21.5	21.5	0.14	0.25	0.21	0.0005
Qwen	balanced	200	17.5	41.5	31.5	9.5	0.49	10^{-9}	0.14	0.072

p_{unif}: Rao–Scott vs. uniform (order of magnitude); p_{std}: request-permutation vs. standard.

where the model appears to cycle the keyed letter through A–D regardless of where it placed the correct text. Against 0/30 errors in the Gemini standard-prompt check, this suggests that a balancing instruction can be satisfied by mis-keying rather than by moving the correct answer. Re-scoring the 15 items by the position of the truly correct option leaves the distribution consistent with uniform (29.2/28.7/22.0/20.1%, χ^2 $p = 0.15$), so the debiasing itself is real, but a balanced key must be checked for correctness before use. The Gemini and Qwen balanced items were not reviewed in full.

Balancing replaces a positional pattern with a sequential one. The gpt-oss balanced keys are not a random arrangement. In 77.7% of consecutive pairs within a request, the next key is the next letter in the cycle A→B→C→D→A, against about 25% expected if keys were independent with the observed shares ($\sum_i p_i\, p_{i+1} = 0.25$). Repeated letters almost disappear (5.3% of consecutive pairs, against $\sum_i p_i^2 = 0.25$ expected), and 12 of the 21 requests carry exactly the key `ABCDABCDAB`. The marginal counts therefore look uniform while the sequence is highly predictable: a student who notices the rotation can predict the next key from the previous one about three times in four. This also explains the mis-keyed items above: the model appears to satisfy the instruction by rotating the letter rather than by moving the correct answer.

H. Exploitability

A guesser learns the most frequent position from a random half of the *requests* and always picks it on the other half. Table VII reports the mean accuracy over

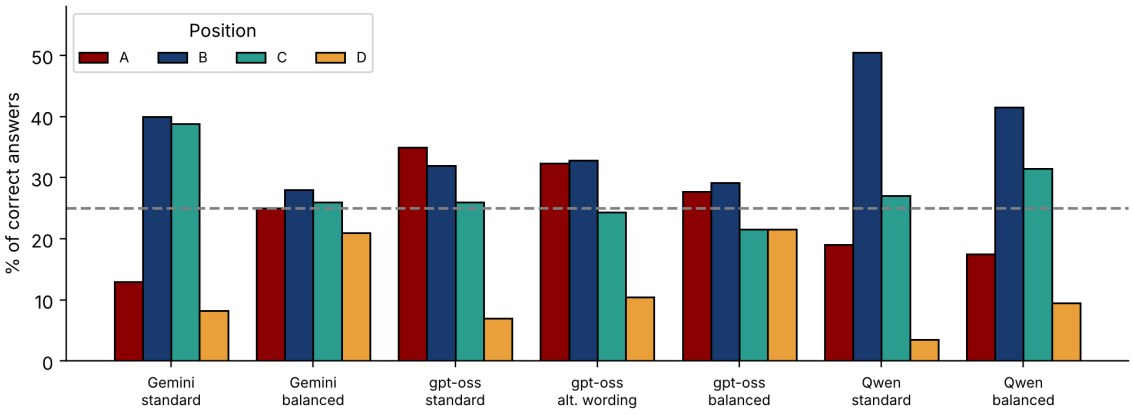


Fig. 4. Standard prompt compared with the reworded and balanced prompts ($T = 0.7$, 10 items per request, four options).

TABLE VII
Out-of-Sample Guessing Accuracy (Standard Prompt)

Model	k	Chance	Mean	Split range	Gain
Gemini	4	25.0	**42.5**	[36.2, 45.9]	1.70×
gpt-oss	4	25.0	**38.7**	[33.9, 43.0]	1.55×
gpt-oss	5	20.0	36.6	[22.0, 46.0]	1.83×
Qwen	4	25.0	**57.0**	[51.0, 63.0]	2.28×

Values in %. Split range: 2.5–97.5 percentiles over 2,000 request-level splits.

2,000 such splits. The bracketed values are the 2.5th and 97.5th percentiles of the split-to-split distribution. They describe how much the score varies between splits; they are *not* a binomial or bootstrap confidence interval for the population accuracy. The range is asymmetric when two positions have similar shares. For Gemini, B (43.2%) and C (38.9%) are close, so some training halves select C, which pulls the distribution's lower tail down, while the upper tail is capped near the share of B.

A student who knows a fraction q of the answers and guesses the rest scores $S = q + (1 - q)\, p_{\text{guess}}$. On a 100-item Qwen quiz, the gain over random guessing is 32.0, 24.0, 16.0 and 8.0 marks for $q = 0$, 0.25, 0.5 and 0.75: the weaker the student, the larger the inflation.

V. Discussion

A. Why a Single Softmax Fails at $T = 0$

If every question shared the same position scores, fitting the softmax to gpt-oss at $T = 1$ would give $p_i(T) = \hat{p}_i^{1/T} / \sum_j \hat{p}_j^{1/T}$, which predicts that A approaches 100% as $T \to 0$. The observed value is 49.5% (Fig. 5). Scores therefore vary across questions, and at $T = 0$ the observed share is the fraction of questions whose favourite position scores highest. Greedy decoding sharpens a favourite but does not create one.

B. Possible Causes

The study establishes *what* transfers, not *why*. Plausible mechanisms include inherited item-writing habits [2]; a writing order in which the known answer is placed early and the later slots are filled with distractors; planning of the answer position before the options are written [1]; and family-specific training, which is consistent with family being the strongest factor and the favourite persisting across versions.

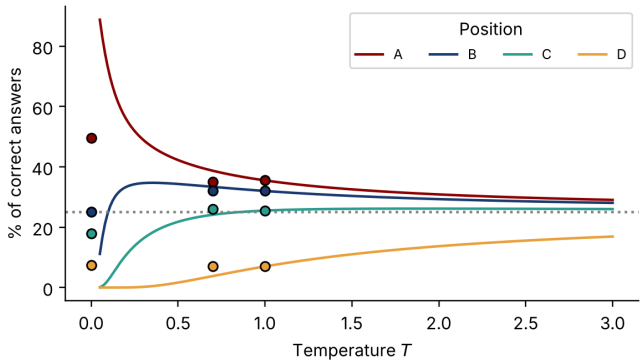


Fig. 5. Single-softmax prediction (lines, fitted at $T = 1$) compared with observed gpt-oss shares (points).

C. Implications

Neither decoding settings, request size nor rewording makes a generated key safe, and a balancing instruction works only for some models. The reliable fix is mechanical: shuffle the options, or assign each item a uniformly random target position, before use.

D. Limitations

Three model families; 100–600 items per condition; key validity checked on samples (and on all gpt-oss balanced items), partly with LLM assistance; cross-study comparisons that are qualitative because the prompts differ; English JSON prompts only; and exploratory $T = 0$ and option-count contrasts that do not survive Holm adjustment.

VI. Conclusion

Answer-position bias in LLM-generated MCQs transfers. It appears in a model family absent from prior work, it keeps its direction across model versions within a family, and it survives changes to temperature, request size, option count and wording. Every model avoids the last slot. Instruction-level balancing removes the bias for two of three models but not the third, and can trade positional bias for mis-keyed answers and a predictable A–D rotation, and a guesser who knows the bias can answer 38–57% of unknown items correctly. Future work will test automated position assignment as a verified mitigation, and transfer to further model versions.

Data and Code Availability

The raw data (one row per generated question), the data-collection notebook and the analysis notebook that reproduces every table are publicly available at https://github.com/Etliou/MCQ_BIAS.

References

[1] X. Tang, X. Duan, and Z. G. Cai, "Do large language models plan answer positions? Position bias in multiple-choice question generation," arXiv:2605.01846, May 2026.

[2] Y. Attali and M. Bar-Hillel, "Guess where: The position of correct answers in multiple-choice test items as a psychometric variable," *J. Educ. Meas.*, vol. 40, no. 2, pp. 109–128, 2003.

[3] C. Zheng, H. Zhou, F. Meng, J. Zhou, and M. Huang, "Large language models are not robust multiple choice selectors," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024.

[4] P. Pezeshkpour and E. Hruschka, "Large language models sensitivity to the order of options in multiple-choice questions," in *Findings Assoc. Comput. Linguist.: NAACL*, 2024.

[5] C. J. Clopper and E. S. Pearson, "The use of confidence or fiducial limits illustrated in the case of the binomial," *Biometrika*, vol. 26, no. 4, pp. 404–413, 1934.