

Request routing services undermine the reproducibility of open weight LLMs

Joe Higton, Joshua A. Tucker, Christopher Barrie*

NYU Center for Social Media, AI, and Politics

August 1, 2026

Abstract

Large language models (LLMs) are now a standard tool for classifying text in social science. To make these classifications reproducible, researchers are often advised to use open weight models and to set the sampling temperature to zero. We show that this advice is insufficient when models are accessed through a request-routing service, in which queries are sent to different hosting providers through a single API. Using a popular request-routing service, OpenRouter, we classify British manifesto sentences for ideology 100 times per sentence across eight models and 33 hosting providers, alongside five model configurations served on our own hardware, generating approximately 1m classification draws. When using the request-routing service, open weight models consistently fail to return deterministic outputs. Depending on the model, only 26% to 92% of sentences receive the same label in all repeated classifications. This variation is not solely due to different providers fielding queries: for five of the eight models, the majority of classification variance arises within providers. On our own hardware, the same task is perfectly reproducible at temperature=0. We conclude that reproducibility depends on the entire model hosting environment, and recommend that researchers report the fullest available metadata and raw responses when using LLMs.

1 Introduction

Large language models (LLMs) are rapidly displacing trained human coders and supervised classifiers as the default method for classifying texts in social science (Gilardi, Alizadeh, and

*JH conducted the analyses, produced the tables and figures, and wrote the first draft of the paper. All of the authors contributed to the design of the project and revising the text. We thank Sol Messing for helpful feedback at an earlier stage of this project.

Kubli 2023; Le Mens and Gallego 2025; Törnberg 2025; Ziems et al. 2024). One of the main methodological concerns with LLMs is reproducibility: LLM output is stochastic and proprietary models are updated or retired without notice (Barrie, Palmer, and Spirling 2024; Ollion et al. 2024; Palmer, Smith, and Spirling 2024; Reiss 2023). The standard remedy is: (a) to use an open weight model, so the exact artifact behind a measurement can be named, archived, and rerun (Spirling 2023); and (b) to set the sampling temperature to zero, so the model returns its single most likely answer (e.g., Törnberg 2024). Together these theoretically make possible the deterministic code and data replication pipeline commonly expected within the discipline (King 1995). However, prior work has shown that hosted LLM APIs can return different outputs and task accuracies across repeated runs even under nominally deterministic settings (Atil et al. 2025).

In this paper we show that this advice falls short of guaranteeing reproducibility, and that implementing it through a request-routing service can increase the fragility of LLM-derived classifications. We examine a popular service for accessing open weight models, OpenRouter (Aubakirova et al. 2026). OpenRouter allows researchers to access open weight models with minimal technical effort by providing a single API for a wide range of open weight (as well as other closed commercial) models. Their service routes LLM queries to third-party “providers” who host the models and return responses to queries. We test whether researchers can achieve reproducible results using OpenRouter, by examining whether identical queries return identical answers, and compare it to running models locally on a personal computer.

We contribute four main findings. First, exact reproducibility fails for every open weight model accessed through OpenRouter: at temperature zero, the share of the 100 sentences that receive a single identical answer across all 100 draws ranges from 26% to 92% across the eight models. Second, different providers serve substantially different versions of the same models in a way that may not be apparent to users. Third, limiting queries to a single provider does not restore reproducibility: for five of the eight open models, most classification variance sits within providers, not between them. Fourth, when using our own hardware the task is perfectly reproducible: all five local configurations return one identical answer on every sentence across all 100 draws at temperature zero. We conclude that using an open weight model at temperature zero is not, by itself, sufficient to make an LLM-based measurement reproducible. Researchers should therefore report how the model was accessed, archive the fullest available serving metadata and raw responses, and use repeated draws to measure any remaining instability (Feuerriegel et al. 2026).

2 Data and measures

We classify sentences from British party manifestos as used in Benoit et al. (2016), labelled for their left-right position on a five-point scale. We draw a random sample of 100 sentences from 1,508 sentences that expert labellers in Benoit et al. unanimously assign to a policy domain. Our prompt elicits a single generic left-right placement on a five-point scale (see Appendix A for the full prompt). Each model classifies each sentence 100 times at each temperature in $\{0, 0.5, 1\}$.

We structure our experiments around three arms. The first is designed to mirror the typical way a researcher might use OpenRouter. We query their API and only specify the temperature and model. OpenRouter then routes the requests to a third-party hosting provider. Here, “routing” means selecting which cloud inference provider will process a request for the specified model. Unless the user manually chooses a provider OpenRouter will select one dynamically, based on factors such as price, performance, and availability, meaning repeated requests may be served by different providers.¹ The second arm is designed to test provider-level variation more directly, so we manually route requests across all providers that OpenRouter serves for each model. The third arm is a local run of five smaller models on consumer computers.

Our evaluation strategy focuses on whether models return identical responses to identical queries. The logic behind this strategy is that if it is not possible to reproduce a classification immediately then it is unlikely the classification will be reproducible in the longer run. Concretely, we employ two metrics. The first metric is stability. For a given model and temperature, let $y_{id} \in \{1, \dots, 5\}$ be the label returned by the d -th draw on sentence i , for $d = 1, \dots, N_i$. Sentence i is *perfectly stable* if all of its draws return one label, $y_{i1} = \dots = y_{iN_i}$, and the *stability rate* of a model at a given temperature is the share of the 100 sentences that are perfectly stable, $\frac{1}{100} \sum_{i=1}^{100} \mathbf{1}\{y_{i1} = \dots = y_{iN_i}\}$. Our second measure is the variance of each sentence’s draws, which we decompose into a between-provider and a within-provider component. For a given model and temperature, let n_{ik} be the number of draws for sentence i from provider k , $\bar{y}_{ik}$ the mean label, $\tilde{s}_{ik}^2$ the population variance, and $\bar{y}_i$ the mean over all of the sentence’s draws across providers. The variance across all N_i draws for sentence i then decomposes as:

$$\text{Var}_d(y_{id}) = \underbrace{\sum_k \frac{n_{ik}}{N_i} \tilde{s}_{ik}^2}_{\text{within providers}} + \underbrace{\sum_k \frac{n_{ik}}{N_i} (\bar{y}_{ik} - \bar{y}_i)^2}_{\text{between providers}}. \quad (1)$$

The between-provider term is variation associated with differences across providers; the within-provider term is variation across draws served by the same provider.²

1. See: <https://openrouter.ai/docs/guides/routing/provider-selection>. Last accessed 07/23/26.

2. Code for this analysis was produced working collaboratively with Claude Code.

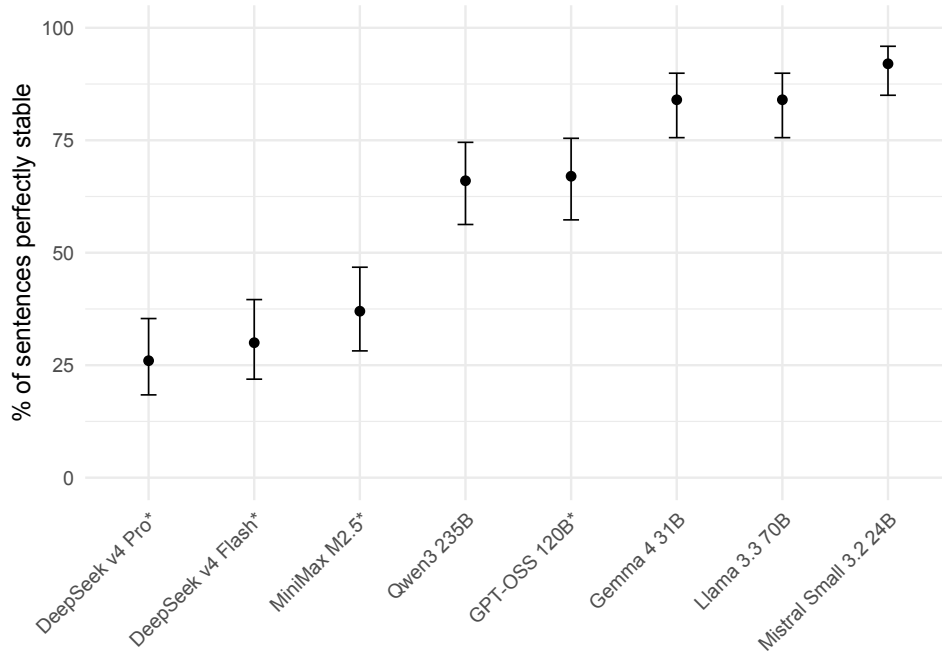


Figure 1: Share of the 100 sentences that are perfectly stable at temperature=0, with Wilson 95% intervals. Asterisks mark models that emitted chain-of-thought reasoning tokens.

3 Results

3.1 No open weight model is fully stable through OpenRouter

We find that none of the open weight models are fully stable, as shown in Figure 1. Researchers using OpenRouter with temperature=0 will find that their classifications are unlikely to reproduce exactly. The most stable model, Mistral Small, is stable on 92 of 100 sentences (95% CI 85.0–95.9), and second is Llama 3.3 70B on 84 (75.6–89.9). The DeepSeek v4 flagships are the least stable, at 30% (Flash) and 26% (Pro).

In Appendix B we supplement these results by calculating the modal label share, i.e. the percentage of draws for each text that return the most common label. In all cases, the median text has a modal label share of over 94%, suggesting that even unstable models return a single classification most of the time, though there is a significant minority of texts for which the modal label share is closer to 80%.

3.2 Instability occurs both between and within providers

OpenRouter sends queries to many providers, though it does not do it evenly. In our first run the average model-sentence cell was served by 11.1 distinct providers, with the largest provider for each model serving an average of 34.6% of the model’s queries.

There is substantial variation in responses between providers, as shown in Figure 2. No-

tably, providers are not scattered randomly around zero: within a model, several providers sit persistently to one side of the mean. In Appendices H and C we show downstream consequences of this disagreement: providers sometimes disagree on the classification of close to 1/3 of texts, which could move the mean left-right party placement by up to 0.35 points on the 1-5 scale, which is, for example, more than half of DeepSeek v4 Pro’s estimated 0.59-point Labour–Liberal Democrat gap (Appendix H).

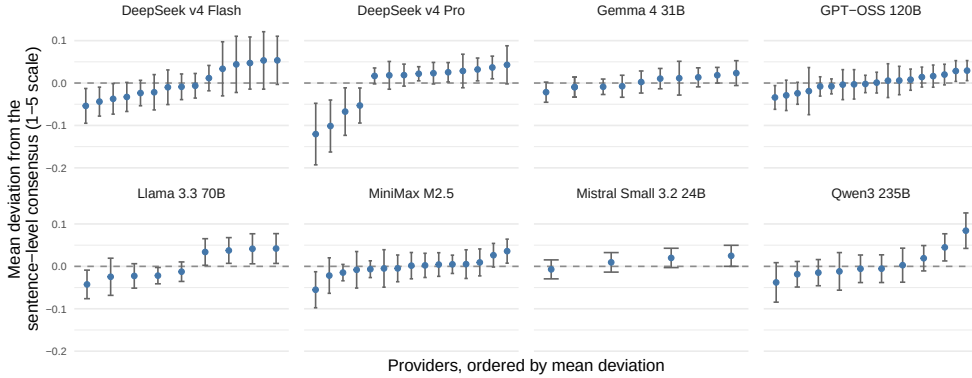


Figure 2: Provider-level disagreement at temperature zero (provider-balanced design; only provider–sentence cells with ≥ 10 usable draws). For each model and sentence, the consensus is the mean label across all draws from all providers. Each point is then a provider’s mean per-sentence deviation from that consensus, with 95% intervals clustered by sentence.

Providers vary significantly in how they configure the same model under the hood, as shown in Table 1. Different providers use different model quantization, i.e. the numerical precision used to represent model weights. Quantization approximates full-precision weights using fewer bits and can thereby alter the token probabilities produced when the model runs (Dettmers et al. 2022; Frantar et al. 2023).³

Providers also differ in whether they have chain-of-thought reasoning enabled by default (Wei et al. 2022), and we found four providers (GMICloud, DekaLLM, SiliconFlow, and Together) that mixed reasoning-on and reasoning-off within the same model for identical API requests (see table A4). Finally, providers are also inconsistent and sometimes contradictory in how they report whether models used reasoning or not. For example, with GPT-OSS 120B, BaseTen’s response object reports 0 in the `reasoning_tokens` field, but returns the full chain-of-thought text in `message.reasoning`, indicating reasoning was enabled, while Amazon Bedrock reports a reasoning-token count in `reasoning_tokens` but returns no chain-of-thought text at all.

However, not all of the instability we observe can be explained by OpenRouter sending requests to different providers. In our second run, we augment our initial data so that every

3. In Section 3.3 we isolate this effect directly, running the same model locally at three quantizations: each is perfectly stable on its own, but they disagree with each other on up to 19 of 100 sentences.

Table 1: OpenRouter providers for each model at the time of writing.

Model	No. of providers	Advertised precision (endpoints)
DeepSeek v4 Flash	16	fp8 (9), unlisted (7), fp4 (1)
DeepSeek v4 Pro	14	fp8 (8), unlisted (6), fp4 (1)
GPT-OSS 120B	17	unlisted (9), fp4 (6), bf16 (3), fp16 (1), fp8 (1)
Gemma 4 31B	11	unlisted (4), bf16 (3), fp4 (3), fp8 (3)
Llama 3.3 70B	10	fp8 (6), bf16 (3), unlisted (3), fp16 (1)
MiniMax M2.5	16	fp8 (11), unlisted (6)
Mistral Small 3.2 24B	4	fp8 (2), bf16 (1), unlisted (1)
Qwen3 235B	10	unlisted (6), fp8 (5)

model-sentence-provider combination has at least ten draws.⁴ When we split the draw variance of each sentence into a within-provider and a between-provider component, we see that for five of the eight models the within-provider variance component is the majority (58% for DeepSeek v4 Flash, 70% for Pro). Between-provider variance dominates for Llama (89%), Mistral (78%), and Qwen (64%).

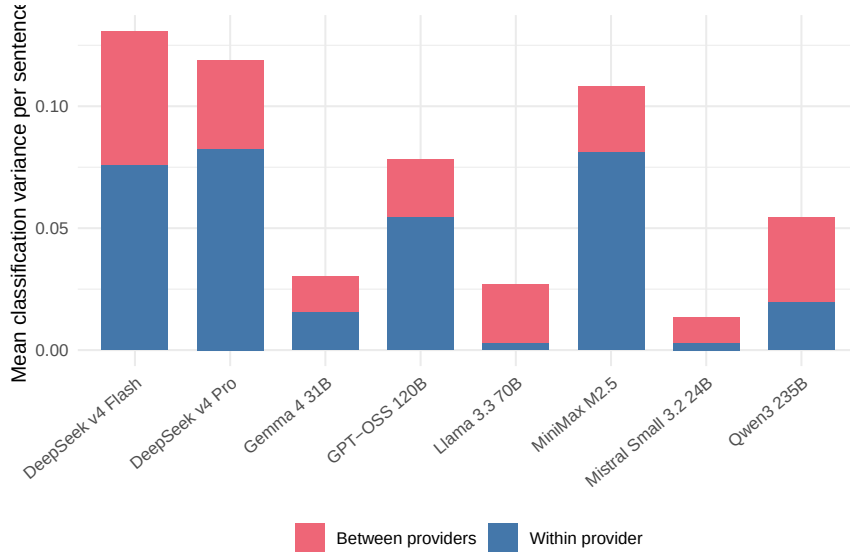


Figure 3: Within- vs. between-provider components of classification variance at temperature=0. For variance at all three temperatures see Figure A4.

⁴ Appendix J details this process, including which provider-model combinations failed to return any valid responses.

3.3 Local models are perfectly stable

To demonstrate the stability of local models we test five smaller language models on Apple hardware using Ollama software (see Appendix I for hardware and software details). We use Mistral Small 3.2 24B, Gemma 4 12B, and Qwen3.5 9B, running the latter at differing quantization levels (bf16, q8_0, and q4_K_M) to isolate the effect of quantization on identical weights and hardware.⁵ The experimental design is otherwise the same as above.

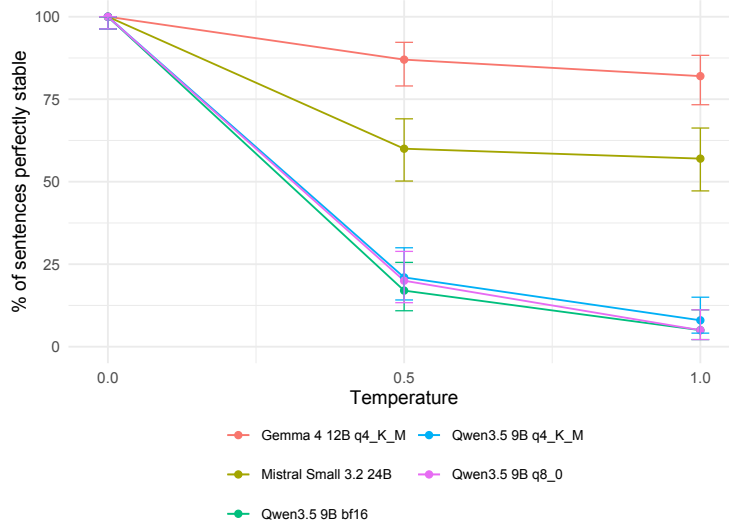


Figure 4: Local arm: share of the 100 sentences perfectly stable, by configuration and temperature, with Wilson 95% intervals. At temperature zero every configuration is perfectly stable (100/100).

Figure 4 shows the stability measure for the local arm: at temperature zero all five configurations are exactly deterministic across all draws, while at temperatures 0.5 and 1 the same configurations are unstable (Table A5). We also find that different quantizations of the same model vary: the q4_K_M quantization of Qwen3.5 9B disagrees with the bf16 model on 19 of 100 sentences, even though each configuration internally is perfectly deterministic (Table A6). In Appendix H we show that this is enough to reverse a point estimate of the ideological ordering of Labour and the Liberal Democrats in our sample: the estimated Labour–Liberal Democrats gap is 0.12 at bf16 (Labour more right-wing), but -0.14 at q4_K_M (Labour more left-wing).

4 Discussion

Our experiments test the promise that using open weight models with temperature=0 can guarantee reproducible results. We find that this is not the case when using a request routing

⁵ bf16 stores weights in 16-bit bfloat format, while q8_0 and q4_K_M use 8-bit and mixed 4-bit quantization, respectively, as per the GGUF/llama.cpp notation used by Ollama.

service, even within a single hosting provider. Beyond instability in the classifications themselves, routed access is also operationally unreliable: providers listed as available sometimes never accept a request, and which providers are available for a given model can change on the timescale of a single study (see Appendix J). These findings are consistent with recent research showing that the behaviour of hosted LLMs depends on the provider and serving infrastructure, not only on the named model itself (Atıl et al. 2025; Li et al. 2026; Lin et al. 2026; Pape, Evertz, and Schönherr 2026).

The instability between providers makes sense due to the variation in model configuration. Variation within providers, which accounts for the majority of variance in most of the models we tested, is harder to explain. We see two primary explanations for why this is, each with support in our data. The first is providers sometimes serve different model configurations behind a single endpoint without the user changing the request parameters (Table A4). These types of changes may be common behind the scenes as providers tweak models to improve latency and uptime, and may not always be made visible to the user.

The second is that other queries processed at the same time as a given request can affect token probabilities and therefore model response, leading to variation even with all configuration parameters held constant. This can happen because providers batch requests together, changing the order in which finite-precision operations are evaluated and thereby slightly perturbing token probabilities even at temperature=0 (He 2025; Yuan et al. 2025). Appendix E explains the mechanism in more detail. Because this mechanism relies on very small differences between numbers, it is most likely to cause a change in the model’s output when the two highest-probability classifications are nearly tied. Consistent with the mechanism, we show in Appendix F that sentences whose classification varies more at higher temperatures (suggesting they are closer to being tied between two classifications) are also less stable at temperature=0 (pooled $r = 0.81$).⁶ Because LLM text generation is sequential, reasoning models may amplify this pattern, as a single flipped token early in the chain can cascade into a different final label (Yuan et al. 2025). Consistent with this prediction, 3/4 of the least stable models in Figure 1 are reasoning models.

4.1 Implications and recommendations

Our results show that researchers turning to open weight models for reproducibility reasons ought not to assume that using temperature=0 will automatically generate reproducible results.

6. We also find evidence for this mechanism in local runs. In a pilot run, we found minor variation in responses depending on how we ordered requests to Ollama (though holding the request order fixed, outputs were reproducible). See Appendix I for more details.

Just as proprietary models can be modified behind the scenes by OpenAI or Anthropic (Chen, Zaharia, and Zou 2024), hosting providers of open models can do the same. The gold standard for reproducibility is still best met by using a local model run on physical hardware the researcher controls, but this is not always possible—researchers may require frontier models that cannot be run on consumer hardware, or the technical implementation costs may be too large. In this case, researchers should make sure they record and publish the hosting provider they used for their analysis, the full JSON response object for every classification, and all of the available model configuration details. Figure 5 summarizes the resulting hierarchy of access arrangements and the documentation required under each.

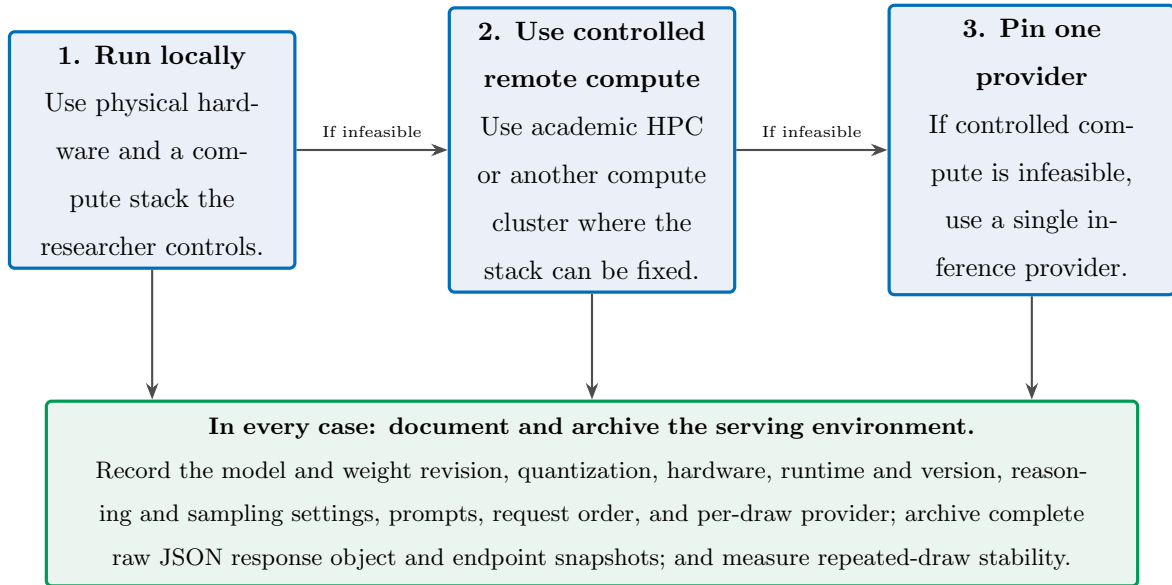


Figure 5: Preferred access hierarchy for reproducible LLM classification.

The larger challenge for researchers is that in such a fast-moving field, using LLMs reproducibly requires researchers to keep track of every feature of an LLM that could plausibly impact output. Here we have highlighted quantization and reasoning as two variables of which researchers need to be aware, but as the technology develops there will be new parameters that are consequential for reproducibility. In the shorter term, it will be important for researchers to take into account, and try to quantify, potential sources of error arising from machine annotations. These include not only routing service routing variation but also an enormous array of other parameter choices (Messing 2026).

In the longer term researchers would also benefit from a research-grade public compute optimized for social scientific needs. Deterministic, reproducible inference is an active area of research in LLM systems (He 2025; SGLang Team 2025; Yuan et al. 2025), with implementations available via LLM serving frameworks vLLM and SGLang.⁷ However, as far as we can

7. See <https://docs.vllm.ai/en/latest/usage/reproducibility/> and https://docs.sglang.io/docs/advanced_

tell, no inference provider makes a batch-invariant deterministic serving mode available via an API, perhaps because it can slow down inference by 25–45% (SGLang Team 2025). As such, deterministic inference is largely restricted to researchers who control their own serving infrastructure. In social scientific applications, reproducibility and stable model configurations matter more than the latency and throughput prioritized by commercial routing services. A research-grade public service should therefore combine versioned models and deterministic serving with the accessibility and economies of scale that make commercial routing services attractive.

[features/deterministic_inference/](#). Accessed July 27, 2026.

References

- Atıl, Berk, Sarp Aykent, Alexa Chittams, Lisheng Fu, Rebecca J. Passonneau, Evan Radcliffe, Guru Rajan Rajagopal, et al. 2025. “Non-Determinism of “Deterministic” LLM System Settings in Hosted Environments.” In *Proceedings of the 5th Workshop on Evaluation and Comparison of NLP Systems*, 135–148. Mumbai, India: Association for Computational Linguistics.
- Aubakirova, Malika, Alex Atallah, Chris Clark, Justin Summerville, and Anjney Midha. 2026. *State of AI: An Empirical 100 Trillion Token Study with OpenRouter*. arXiv: 2601.10088 [cs.AI].
- Barrie, Christopher, Alexis Palmer, and Arthur Spirling. 2024. *Replication for language models problems, principles, and best practice for political science*.
- Benoit, Kenneth, Drew Conway, Benjamin E. Lauderdale, Michael Laver, and Slava Mikhaylov. 2016. “Crowd-sourced Text Analysis: Reproducible and Agile Production of Political Data.” *American Political Science Review* 110 (2): 278–295. ISSN: 0003-0554, 1537-5943.
- Chen, Lingjiao, Matei Zaharia, and James Zou. 2024. “How Is ChatGPT’s Behavior Changing over Time?” *Harvard Data Science Review* 6 (2).
- Dettmers, Tim, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. “LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale.” In *Advances in Neural Information Processing Systems*.
- Feuerriegel, Stefan, Christopher Barrie, M. J. Crockett, Laura K. Globig, Killian L. McLoughlin, Dan-Mircea Mirea, Arthur Spirling, et al. 2026. “A reporting checklist for large language models in behavioural science.” *Nature Human Behaviour* 10, no. 7 (2026): 1182–1186. ISSN: 2397-3374.
- Frantar, Elias, Saleh Ashkboos, Torsten Hoefer, and Dan Alistarh. 2023. *GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers*, arXiv:2210.17323, 2023. arXiv: 2210.17323[cs.LG].
- Gilardi, Fabrizio, Meysam Alizadeh, and Maël Kubli. 2023. “ChatGPT outperforms crowd workers for text-annotation tasks.” *Proceedings of the National Academy of Sciences* 120 (30): e2305016120.
- He, Horace. 2025. *Defeating Nondeterminism in LLM Inference*. Thinking Machines Lab: Connectionism, September 10, 2025.

- King, Gary. 1995. “Replication, Replication.” *PS: Political Science & Politics* 28 (3): 444–452.
- Le Mens, Gaël, and Aina Gallego. 2025. “Positioning Political Texts with Large Language Models by Asking and Averaging” [in en]. *Political Analysis* 33 (3): 274–282. ISSN: 1047-1987, 1476-4989.
- Li, Haorui, Zhenghui He, Xuanzi Liu, Yang Xu, Dongsheng Liu, Jiakang Ma, Lupan Wu, Yangjie Wu, Xiongchao Tang, and Tianhui Shi. 2026. *When Is the Same Model Not the Same Service? A Measurement Study of Hosted Open-Weight LLM APIs*. ArXiv:2605.02821 [cs.PF].
- Lin, Guanjie, Yinxin Wan, Shichao Pei, Ting Xu, Kuai Xu, and Guoliang Xue. 2026. *Behavioral Consistency and Transparency Analysis on Large Language Model API Gateways*. ArXiv:2604.21083 [cs.CR].
- Messing, Solomon. 2026. *Hidden Measurement Error in LLM Pipelines Distorts Annotation, Evaluation, and Benchmarking*. arXiv: 2604.11581 [cs.CL].
- Ollion, Étienne, Rubing Shen, Ana Macanovic, and Arnault Chatelain. 2024. “The dangers of using proprietary LLMs for research.” *Nature Machine Intelligence* 6:4–5.
- Palmer, Alexis, Noah A. Smith, and Arthur Spirling. 2024. “Using proprietary language models in academic research requires explicit justification.” *Nature Computational Science* 4:2–3.
- Pape, David, Jonathan Evertz, and Lea Schönherr. 2026. *The Silent Hyperparameter: Quantifying the Impact of Inference Backends on LLM Reproducibility*. ArXiv:2605.19537 [cs.LG].
- Reiss, Michael V. 2023. *Testing the reliability of ChatGPT for text annotation and classification: A cautionary remark*. ArXiv:2304.11085.
- SGLang Team. 2025. *Towards Deterministic Inference in SGLang and Reproducible RL Training*. LMSYS Org blog, September 22, 2025.
- Spirling, Arthur. 2023. “Why open-source generative AI models are an ethical way forward for science.” *Nature* 616:413.
- Törnberg, Petter. 2024. “Best Practices for Text Annotation with Large Language Models.” *Sociologica* 18 (2): 67–85.
- Törnberg, Petter. 2025. “Large Language Models Outperform Expert Coders and Supervised Classifiers at Annotating Political Social Media Messages.” *Social Science Computer Review* 43 (6): 1181–1195. ISSN: 0894-4393, 1552-8286.

- Wei, Jason, 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 35*, 24824–24837. New Orleans, Louisiana, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS). ISBN: 978-1-7138-7108-8.
- Yuan, Jiayi, Hao Li, Xinheng Ding, Wenya Xie, Yu-Jhe Li, Wentian Zhao, Kun Wan, Jing Shi, Xia Hu, and Zirui Liu. 2025. *Understanding and Mitigating Numerical Sources of Nondeterminism in LLM Inference*. arXiv: 2506.09501 [cs.CL].
- Ziems, Caleb, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. “Can Large Language Models Transform Computational Social Science?” *Computational Linguistics* 50 (1): 237–291.

Supplementary Appendix

A Task, prompt, and response handling

The system prompt reads:

You are a political text classifier. You will be given a sentence from a British political manifesto. Classify it on a 1-5 scale: 1 = very left-wing, 2 = left-wing, 3 = neither, 4 = right-wing, 5 = very right-wing. Respond with ONLY the number.

And the user message:

Classify this sentence from a British political manifesto on a 1-5 scale: 1 = very left-wing, 2 = left-wing, 3 = neither, 4 = right-wing, 5 = very right-wing.
{sentence}

We keep numeric responses and 1–5 classifications in JSON or tool-call fields, and record free-text, empty, and off-scale responses as missing.

B Modal label frequency

The stability rate we use in the main text records a sentence as unstable if there is more than one unique label across all the usable draws we get from a model (usable meaning, we exclude missing or invalid responses). This makes it a binary measure. To show the magnitude of disagreement, we here calculate the modal-label share. For each (model, sentence, temperature) cell, this is the percentage of usable draws that return the cell’s most common label. A value of 100% is a perfectly stable cell, while a value near 50% indicates that the two most common labels are close to a tie.

At temperature zero, the lowest per-model median modal-label share is 96%. For five of the eight OpenRouter models the median is 100%. This means that even for models that are perfectly stable on only a minority of sentences, most cells still have one answer that dominates. However, the minimum cell-level shares at temperature zero are close to 50% for several models. Increasing temperature broadens these distributions, although the median remains at least 94% for every hosted model at temperature one.

We repeat this for the local arm in Figure A2. At temperature zero, every sentence–model cell has a modal label share of 100%.

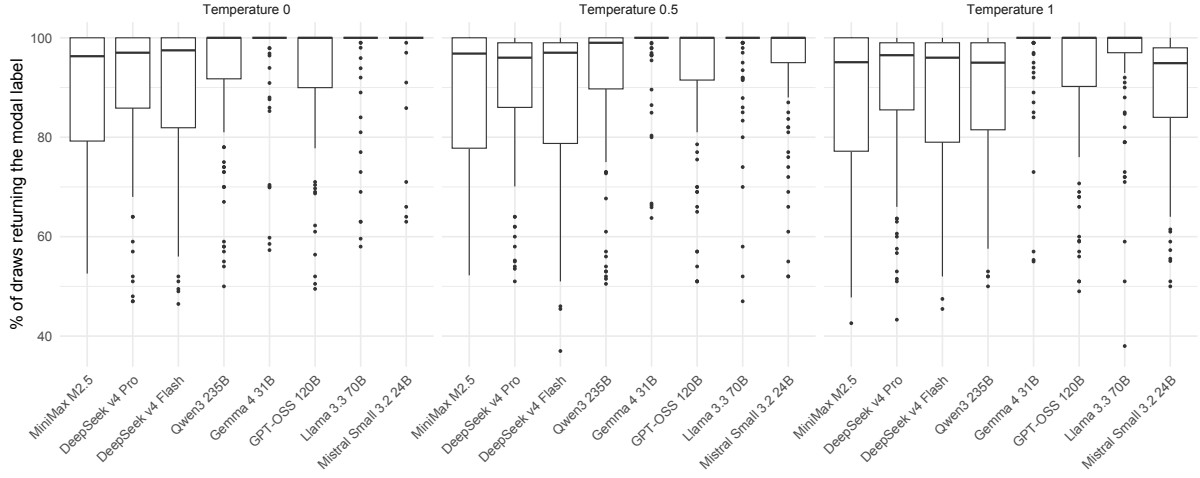


Figure A1: Distribution of the modal-label share across the 100 sentences in the hosted arm, by model and temperature. The modal-label share is the percentage of a cell’s usable draws that return its most common label. Boxes show the interquartile range and median; whiskers and points follow the standard boxplot convention. Models are ordered by their median share across the displayed temperatures.

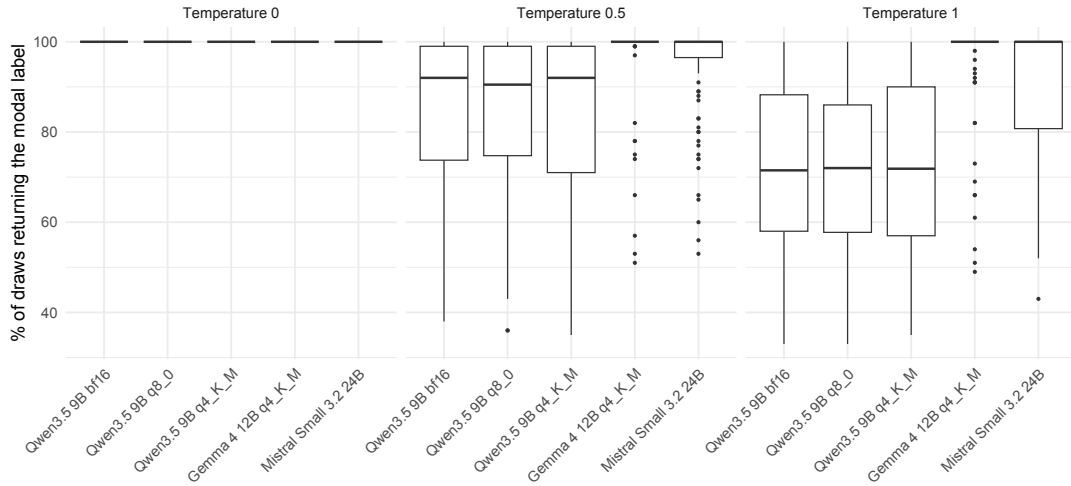


Figure A2: Distribution of the modal-label share across the 100 sentences in the local arm, by model configuration and temperature. At temperature zero all five distributions collapse to 100%.

C Provider effects on average labels

In Table A1 we count the sentences where provider variation could affect the classification of a sentence for a researcher taking the average of ten draws per sentence. To do this we take a simple mean per sentence from a given provider-model (at temperature 0) and round it to the nearest whole number. The “ ≥ 1 pt” column counts the sentences (of 100) on which this process could lead to different labels by 1 point on the 1-5 ideology scale from at least two providers; the “ ≥ 2 pts” column does the same but for 2 points difference. For DeepSeek v4 Flash, providers disagree on 29 of 100 sentences, and across the eight models there are 24 two-point disagreements.

Because we are averaging over only ten draws, sentences which are close to a tie between two classifications may round up or down by chance even if instability is not structured by provider. To guard against this inflating the amount of disagreement between providers, we calculate a permutation null. We do this by re-dealing the observed draws to providers at random. We find that every model’s observed count exceeds its null (all $p \leq 0.001$). In practical terms, this means for DeepSeek v4 Flash, the label a researcher publishes could differ depending on which provider served their requests on 29 of 100 sentences. Rounding luck accounts for about 19 of those, while the remainder is provider-level variation.

Table A1: Provider effects at temperature zero (provider-balanced design, dead pairs excluded): observed label flips across providers vs. a permutation null, and internally stable providers per model.

Model	Providers	≥ 1 pt	null	≥ 2 pts	null	Stable providers
DeepSeek v4 Flash	15	29	18.6	9	0.6	0 / 15
DeepSeek v4 Pro	14	29	17.1	6	0.2	0 / 14
GPT-OSS 120B	18	22	13.5	4	0.5	1 / 18
Gemma 4 31B	11	13	5.5	1	0.0	3 / 11
Llama 3.3 70B	9	17	5.0	1	0.0	3 / 9
MiniMax M2.5	16	38	22.5	3	0.2	0 / 16
Mistral Small 3.2 24B	4	6	1.3	0	0.0	1 / 4
Qwen3 235B	10	28	12.6	0	0.0	1 / 10

Note: “ ≥ 1 pt”/“ ≥ 2 pts”: sentences (of 100) whose rounded per-provider mean label differs across providers by at least that much; each “null” column gives the count expected under a permutation null in which all providers are identical measurement tools (1,000 permutations; all observed ≥ 1 -point counts exceed the null’s 95th percentile, $p \leq 0.001$; Table A2). “Stable providers”: providers for which every adequately sampled sentence (≥ 10 usable draws) is perfectly stable; 2.1% of provider–sentence cells fall below this sampling threshold.

C.1 Permutation null for the provider label flips

Within each (model, sentence) cell at temperature zero we shuffle the usable draws across providers, preserving each provider’s draw count, and recompute the across-provider spread of rounded per-provider mean labels (1,000 permutations, fixed seed). Table A2 reports observed counts against the null distribution. For every model the observed ≥ 1 -point count exceeds the null 95th percentile.

Table A2: Observed label-flip counts vs. the permutation null, by model and flip size. p is the share of permutations with a count at least as large as the observed one (smallest attainable value $1/(1,000 + 1)$). Rows with observed 0 have $p = 1$ by construction.

Model	Flip size	Observed	Null mean	Null 95th pct.	p
DeepSeek v4 Flash	≥ 1 point	29	18.6	21.05	0.001
DeepSeek v4 Flash	≥ 2 points	9	0.6	2.00	0.001
DeepSeek v4 Pro	≥ 1 point	29	17.1	19.00	0.001
DeepSeek v4 Pro	≥ 2 points	6	0.2	1.00	0.001
GPT-OSS 120B	≥ 1 point	22	13.5	15.00	0.001
GPT-OSS 120B	≥ 2 points	4	0.5	1.00	0.001
Gemma 4 31B	≥ 1 point	13	5.5	7.00	0.001
Gemma 4 31B	≥ 2 points	1	0.0	0.00	0.005
Llama 3.3 70B	≥ 1 point	17	5.0	7.00	0.001
Llama 3.3 70B	≥ 2 points	1	0.0	0.00	0.002
MiniMax M2.5	≥ 1 point	38	22.5	26.00	0.001
MiniMax M2.5	≥ 2 points	3	0.2	1.00	0.002
Mistral Small 3.2 24B	≥ 1 point	6	1.3	3.00	0.001
Mistral Small 3.2 24B	≥ 2 points	0	0.0	0.00	1.000
Qwen3 235B	≥ 1 point	28	12.6	14.00	0.001
Qwen3 235B	≥ 2 points	0	0.0	0.00	1.000

D Completion length and reasoning defaults

Table A3: Completion tokens per draw and the share of draws that use reasoning, per model (free-routing run, all temperatures). No request asked for or disabled reasoning; hosted models ran on provider defaults. The local arm disables chain-of-thought explicitly.

Model	Access	Median	p90	Max	Reasoning draws (%)
DeepSeek v4 Flash	OpenRouter	142	268	1,421	74.9
DeepSeek v4 Pro	OpenRouter	195	355	2,520	85.3
GPT-OSS 120B	OpenRouter	114	175	505	96.7
Gemma 4 31B	OpenRouter	2	2	2	0.0
Llama 3.3 70B	OpenRouter	2	2	331	0.0
MiniMax M2.5	OpenRouter	221	334	1,388	100.0
Mistral Small 3.2 24B	OpenRouter	2	2	328	0.0
Qwen3 235B	OpenRouter	1	2	3	0.0

NB that the maximum completion lengths in Llama and Mistral are 331 and 328 tokens with no reasoning because in these cases they generated long, non-reasoning responses as their main answer to the prompt.

We use a conservative 5% and 95% cutoff for classifying a provider as mixing reasoning modes.

Table A4: Share of draws with reasoning tokens (%), by provider, for the five models that emitted reasoning in either hosted run (both hosted runs pooled; “–” = provider not on that model’s menu). We use two rules to check for reasoning: `reasoning_tokens > 0`, or any text present in `message.reasoning`.

Provider	DeepSeek v4 Flash	DeepSeek v4 Pro	GPT-OSS 120B	Gemma 4 31B	MiniMax M2.5
AkashML	0.0	–	–	–	100.0
Alibaba	99.8	100.0	–	–	–
Amazon Bedrock	–	–	100.0	–	–
AtlasCloud	100.0	100.0	–	–	100.0
Baidu	100.0	100.0	–	–	–
BaseTen	–	–	100.0	–	–
Cerebras	–	–	100.0	–	–
Chutes	–	–	–	0.0	100.0
DeepInfra	0.0	0.0	100.0	0.0	100.0
DekaLLM	–	–	66.7	–	–
DigitalOcean	0.0	0.0	100.0	–	100.0
Fireworks	100.0	100.0	–	–	–
Friendli	–	–	–	–	100.0
GMICloud	100.0	58.4	–	–	–
Google	–	–	99.9	–	–
Groq	–	–	100.0	–	–
Inceptron	–	–	–	–	100.0
Mancer 2	–	–	100.0	–	–
Mara	–	–	100.0	–	100.0
Minimax	–	–	–	–	100.0
ModelRun	–	–	–	0.0	–
Morph	0.0	–	–	–	–
Nebius	–	–	100.0	–	–
Novita	100.0	100.0	100.0	0.0	100.0
Parasail	100.0	100.0	100.0	0.0	100.0
Phala	–	–	100.0	0.0	100.0
SambaNova	–	–	100.0	0.0	–
SiliconFlow	91.6	100.0	100.0	0.0	100.0
StreamLake	100.0	100.0	–	–	100.0
Together	–	100.0	100.0	9.4	–
Venice	100.0	100.0	–	0.0	99.5
WandB	0.0	0.0	100.0	0.0	100.0

E Finite-precision arithmetic and batched inference

The below is a summary of the explanations of how finite-precision arithmetic and request batching can produce divergent outputs found in Yuan et al. (2025) and He (2025).

To generate the next token in a sequence, an LLM computes a “score” (a logit) for every candidate token. Each logit is the result of a large number of multiplications and additions performed using finite-precision numbers, which can store only a limited number of significant digits. This means that intermediate results must be rounded to the nearest representable number, so changing the order or grouping of the same operations means different rounding errors which produces slightly different final logits.⁸ Practically, this happens because in high-throughput situations changing batch sizes cause inference libraries to use different kernel configurations or reduction strategies, which changes the grouping of floating-point operations for a request. This means server load determines which other requests are processed alongside a given request, so identical user requests can produce slightly different logits even at temperature zero.

F Correlation of instability across temperatures

The finite-precision variation described above suggests variation at temperature-zero should concentrate on sentences whose top two answers nearly tie, as this is when such small differences can affect final output. We can measure this with repeated temperature-1 draws: at higher temperatures, repeated draws approximate underlying token probabilities, so a sentence whose repeated temperature-1 draws are spread across several labels is one the model itself finds close, assigning similar probabilities to two or more labels. We therefore take the standard deviation of each sentence’s temperature-1 draws as a sentence-level measure of closeness and ask whether it predicts instability at temperature zero. We find across (model, sentence) pairs, the temperature-1 standard deviation correlates with the temperature-zero standard deviation at a pooled $r = 0.81$.

8. A simple illustration that He (2025) uses is that in a Python interpreter, $(0.1 + 1e20) - 1e20$ returns 0, whereas $0.1 + (1e20 - 1e20)$ returns 0.1: in the first ordering, 0.1 is rounded away when it is added to the much larger number.

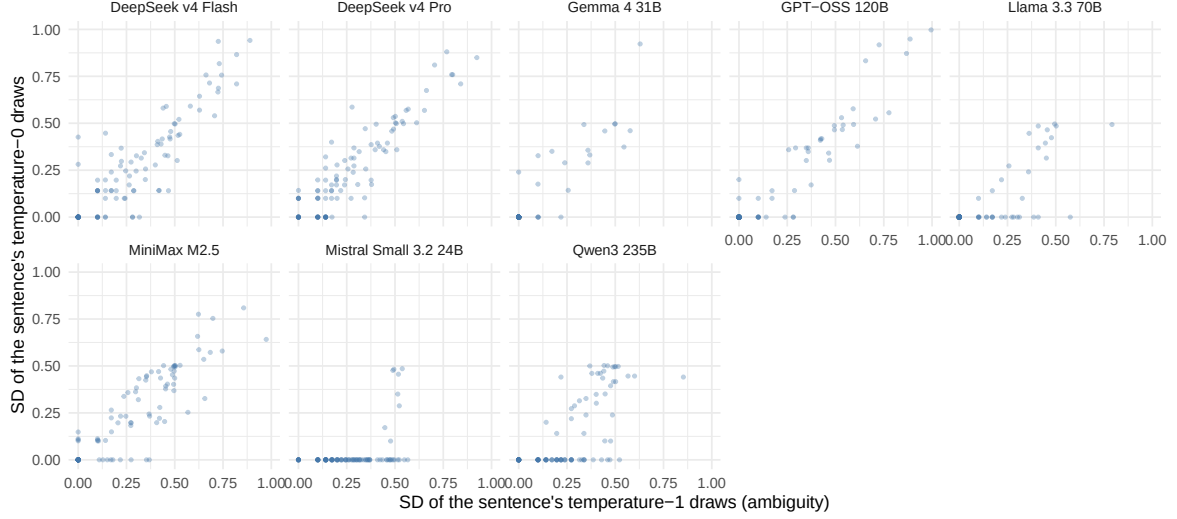


Figure A3: Sentence-level dispersion of each model’s temperature-1 draws (x-axis) against the dispersion of its temperature-0 draws (y-axis), one point per (model, sentence). Pooled $r = 0.81$; per-model correlations with the binary temperature-0 instability indicator range 0.45–0.85.

G Variance decomposition

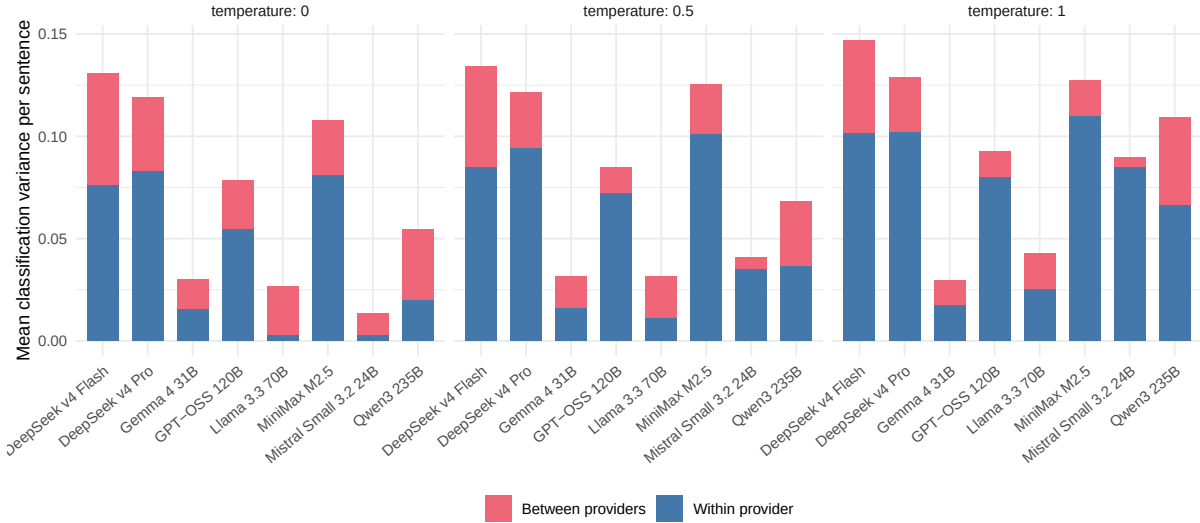


Figure A4: The decomposition of equation 1 at all three temperatures. The within-provider share at temperature zero is 58% (DeepSeek v4 Flash), 70% (Pro), 52% (Gemma), 75% (MiniMax), 70% (GPT-OSS), 36% (Qwen3), 11% (Llama), and 22% (Mistral).

H Downstream effect on party positions

Figure A5 plots each party’s mean placement, by provider and, in the local run, by quantization. Provider choice moves Labour’s placement under DeepSeek v4 Pro across a 0.35-point range (2.85–3.20), more than half that model’s Labour–LibDem gap of 0.59. The ordering itself is

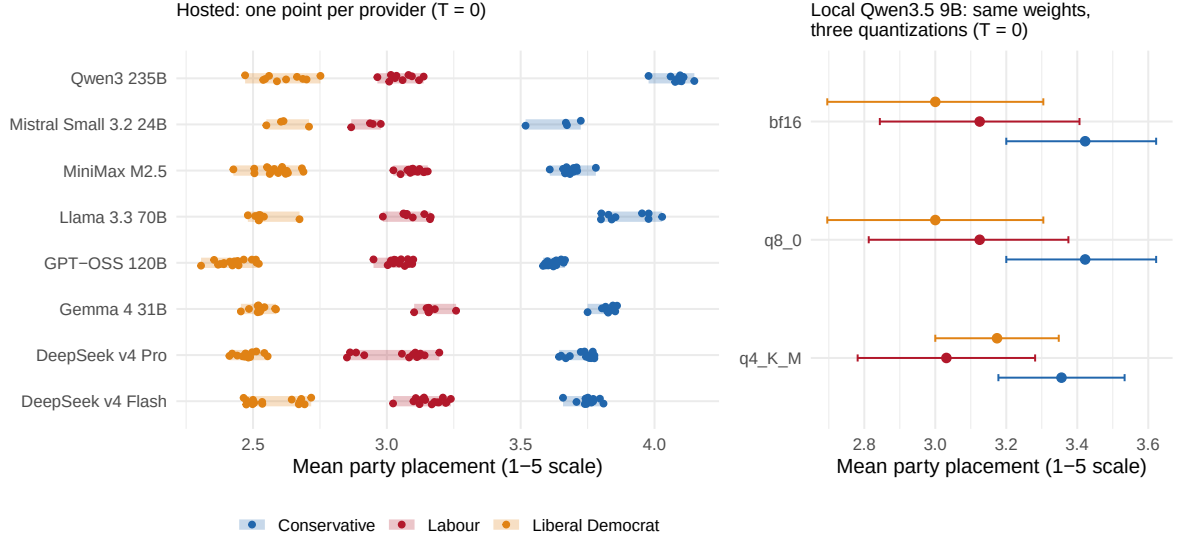


Figure A5: The downstream quantity: each party’s mean placement on the 1–5 scale (Conservative 45 sentences, Labour 32, Liberal Democrat 23). Left: hosted open models at temperature zero, one point per provider (segments span the provider range). Every one of the 97 provider–model pairs preserves the ordering $\text{Con} > \text{Lab} > \text{LD}$, but placements move by up to 0.35 scale points with provider choice. Right: the local Qwen3.5 9B quantization ladder with sentence-bootstrap 95% intervals (2,000 resamples): at bf16 and q8_0 Labour sits right of the Liberal Democrats; at q4_K_M the estimated ordering reverses.

provider-robust: all 97 pairs recover $\text{Con} > \text{Lab} > \text{LD}$.

With the local models, the estimated Labour–LibDem gap is 0.12 at bf16 and q8_0 but -0.14 at q4_K_M. With 100 sentences this sits within sampling uncertainty (change in gap 0.27, bootstrap 95% interval -0.05 to 0.59), so we do not claim a true swap.

I Local runs

Hardware and software. The local arm was split across two Apple computers, with each model configuration run entirely on one machine. The Qwen3.5 9B q4_K_M and q8_0 configurations and Gemma 4 12B ran on a Mac mini with an Apple M2 chip and 16 GB of memory, using Ollama v0.30.5 with embedded `llama.cpp` revision `6f3a9f3de`. Mistral Small 3.2 24B and the Qwen3.5 9B bf16 configuration ran on a MacBook Pro with an Apple M4 Pro chip and 24 GB of memory, using Ollama v0.31.1 with embedded `llama.cpp` build b9840 (revision `8c146a8`). Ollama source and releases are available at <https://github.com/ollama/ollama>.

Request ordering, prompt caching, and stability. In a pilot of the local arm, we ran each sentence’s 100 draws consecutively. At temperature zero, draws 2–100 were always identical, but in three of the 100 Mistral cells draw 1 alone differed. We suspect this pattern arose

Table A5: Local arm: share of the 100 sentences perfectly stable across 100 draws, by temperature. Serial requests to Ollama on a single machine per model; draws interleaved across prompts so that no request follows an identical one.

Model (local)	$T = 0$	$T = 0.5$	$T = 1$
Mistral Small 3.2 24B	100	60	57
Gemma 4 12B q4_K_M	100	87	82
Qwen3.5 9B bf16	100	17	5
Qwen3.5 9B q8_0	100	20	5
Qwen3.5 9B q4_K_M	100	21	8

Table A6: The same weights at three precisions: Qwen3.5 9B served locally at bf16, q8_0, and q4_K_M, temperature zero. Each precision is perfectly deterministic, so each has exactly one answer per sentence; the table compares those answers across precisions.

Precisions compared	Texts agreeing	Largest difference
bf16 vs q8_0	100 / 100	0
bf16 vs q4_K_M	81 / 100	2
q8_0 vs q4_K_M	81 / 100	2

from the local server’s prompt cache, which stores intermediate representations of a prompt so that an immediately repeated prompt does not need to be processed in full again. With the consecutive ordering the first draw followed a different sentence and therefore required a fresh prompt computation, whereas draws 2–100 reused the cached computation. This difference can introduce small numerical differences, although each path is itself deterministic. To test this we reran the local arm with draws interleaved across sentences and temperatures, ensuring that consecutive requests did not share a prompt and so minimizing cache reuse. The rerun was completely stable at temperature zero. This suggests that request order and cache state form part of the local serving environment, even though a fixed local setup and request order remain reproducible.

J Response failures

Free routing samples providers unevenly, so the second run pinned requests to each provider on each open model’s menu, aiming for ten draws per (provider, sentence, temperature) cell.

The pooled hosted results file contains 787,410 saved response records, of which 391,086 contain usable classifications, across 33 providers. Calls that failed before a response could be saved were recorded separately in error logs and are not included in those totals. The balanced run repeatedly retried failed cells in an effort to reach the target number of draws. Across these retry rounds, the run generated 1,466,994 failure records, of which 1,333,020 came from five provider-model pairs that never, or almost never, accepted a request. The final invocation spent 42 hours retrying these pairs, and a re-test six days later found all five still listed and still unavailable (Table A7). Llama 3.3 70B additionally produced 387,614 saved responses without a usable classification in the balanced run, compared with 13,184 usable classifications. The provider pools also drifted on the timescale of a single study: between our two runs, one day apart, a provider vanished from MiniMax M2.5’s menu, according to our endpoint snapshots.

In the end we did not quite achieve full coverage: 2.1% of provider-sentence cells at temperature zero have fewer than ten usable draws, despite our retry rounds.

Table A7: The five dead provider-model pairs. “Failed attempts”: error records accumulated against the pair during the provider-balanced run (pinned requests); “usable draws”: classifications the pair ever returned (all from the earlier free-routing run). All five appeared as available on OpenRouter’s public endpoint listings throughout, and again at a re-test six days after the run (recorded in the project’s decision log).

Model	Provider	Failure mode	Failed attempts	Usable draws
Llama 3.3 70B	SambaNova	http_404	324,822	482
Llama 3.3 70B	Cloudflare	timeout	19	0
DeepSeek v4 Flash	Cloudflare	http_404	252,630	996
DeepSeek v4 Flash	DeepSeek	http_404	377,549	0
DeepSeek v4 Pro	DeepSeek	http_404	378,000	0