LLM ROUTING AS REASONING: A MAXSAT VIEW

Son Nguyen, Xinyuan Liu, Ransalu Senanayake
Arizona State University
{snguye88,xinyua11,ransalu}@asu.edu

ABSTRACT

Routing a query through an appropriate LLM is challenging, particularly when user preferences are expressed in natural language and model attributes are only partially observable. We propose a constraint-based interpretation of language-conditioned LLM routing, formulating it as a weighted MaxSAT/MaxSMT problem in which natural language feedback induces hard and soft constraints over model attributes. Under this view, routing corresponds to selecting models that approximately maximize satisfaction of feedback-conditioned clauses. Empirical analysis on a 25-model benchmark shows that language feedback produces near-feasible recommendation sets, while no-feedback scenarios reveal systematic priors. Our results suggest that LLM routing can be understood as structured constraint optimization under language-conditioned preferences.

1 INTRODUCTION

Routing as reasoning under partial observability. LLM routing is the process of dynamically selecting the most appropriate language model for a given query. Most prior work treats routing as learning a policy that trades accuracy for cost. FrugalGPT (Chen et al., 2024) learns a cascading strategy for a two-model scenario, routing strong models only when needed. Hybrid LLM (Ding et al., 2024) utilizes the predicted difficulty of a query. Other approaches are trained from preference data (Ong et al., 2025). While effective, these methods predominantly focus on accuracy, and the router typically outputs a single endpoint. With the proliferation of LLMs into real-world industrial use cases (Bommasani et al., 2022), LLM routing has now become an operational problem because a production system must choose among many candidate *LLM endpoints* with different cost–quality trade-offs, and the routing policy is rarely stationary (Gama et al., 2014; Lu et al., 2018). In addition to non-stationarity, a router faces *partial observability*: endpoints have latent properties (e.g., robustness to long-context errors, instruction adherence, reasoning reliability) that are not fully specified *a priori*. Thus, routing needs to be considered as a *reasoning* problem in which the controller must interpret user intent and act under uncertainty.

Language feedback as constraints. A method to route is to provide a short natural-language direction key d (e.g., "select a cheaper model") from the current model. The router must map d into a set of constraints over endpoints. Taking a symbolic representation, language feedback (LF) can be injected as logical constraints (hard or soft), and routing corresponds to selecting an assignment that satisfies them.

A MaxSAT view of LLM routing. Currently, most prior work provides limited support for (i) compositional *natural-language*, and (ii) multi-model scenarios. In contrast, we formulate routing as a MaxSAT problem (i.e., satisfy as many clauses/constraints as possible) (AnsóTegui et al., 2013; Hansen & Jaumard, 1990; Biere et al., 2009). We propose a unifying abstraction: the router approximately solves a weighted satisfiability instance where (i) *hard clauses* encode safety/feasibility and specification-level requirements, and (ii) *soft clauses* encode preferences whose weights are conditioned on d. Importantly, our hard/soft distinction is not the common "hard/soft resource bounds" pattern (Chen et al., 2026); rather, *hardness is induced by language semantics* (certainty/negotiability), and weights are *injected per query* by the direction key. Under this view, absence of feedback corresponds to a vacuous constraint set, which should yield a neutral solution unless the router carries implicit priors (i.e., default clause weights).

Contributions.

1. We formalize language-conditioned LLM routing as constrained selection under partial observability, and represent the router's behaviors as logical postconditions.
2. We provide a reasoning-centric MaxSAT/MaxSMT interpretation in which direction keys inject hard/soft constraints and clause weights, yielding a unified abstraction for both language feedback and default (no-feedback) routing as weighted satisfiability.
3. We empirically characterize a black-box LLM router on a 25-endpoint benchmark with 34 objective attributes, showing that (i) language feedback produces high-precision, near-feasible shortlists, and (ii) no-feedback routing frequently yields a consistent "robust core" explainable by sparse implicit priors $w(\bot)$ rather than unstructured randomness.

2 METHODOLOGY

2.1 PROBLEM SETUP

Endpoints and partial observability. We are given a finite set of M endpoints, $\mathcal{A} = \{a^{(m)}\}_{m=1}^{M}$. Each endpoint $a^{(m)}$ has (i) *observable* attributes $\mathbf{o}^{(m)} \in \mathbb{R}^K$ (e.g., output price, cached-input price, context window, max output tokens), and (ii) *latent* attributes $\mathbf{z}^{(m)}$ (e.g., instruction-following reliability, reasoning stability, failure modes), which are not directly available to the router at decision time. A routing interaction provides:

$$(p, \mathcal{A}, a_0, d),$$

where p is the user prompt, $a_0 \in \mathcal{A}$ is the current endpoint and $d \in \Sigma^\star \cup \{\bot\}$ is a direction key, with $\Sigma^\star$ denoting the set of all finite strings over an alphabet Σ. We define $d = \bot$ as either not having a language feedback or a direction key that shows no additional preferences (e.g. NONE, I don't have any feedback, etc.)

2.2 CLAUSE INJECTION

We consider the router as a single LLM agent, details in Appendix B.1. The agent receives $(p, \mathcal{A}, a_0, d)$ and outputs a binary output mask C. We model the semantics of a direction key d as a set of constraints over endpoint attributes. Let $\Phi(d)$ denote a multiset of formulas (constraints) derived from d. In the simplest instantiation, each formula is a predicate over observable attributes,

$$\varphi(\mathbf{o}^{(m)}, \mathbf{o}^{(0)}) \in \{0, 1\},$$

comparing candidate endpoint m to the current endpoint 0 (e.g., "cheaper output price").

Since natural language is ambiguous, we introduce:

1. **Hard constraints** $\Phi_H(d)$: must-hold requirements (precise keys).
2. **Soft constraints** $\Phi_S(d)$: preferences with uncertain scope (vague keys).

We now compile routing into a weighted satisfiability instance. Introduce decision variables $\{x_m\}_{m=1}^{M}$, where $x_m = 1$ indicates recommending endpoint $a^{(m)}$.

For each constraint $\varphi_j \in \Phi(d)$ and each endpoint m, define a satisfaction literal:

$$\ell_{m,j} := \varphi_j(\mathbf{o}^{(m)}, \mathbf{o}^{(0)}) \in \{0, 1\}.$$

For each soft constraint $\varphi_j \in \Phi_S(d)$ we associate a nonnegative weight $w_j(d) \in \mathbb{R}_{\geq 0}$, which captures how strongly the direction key d expresses preference j. To model routing as selecting a non-trivial shortlist, we impose a recommendation-budget constraint $L(d) \leq \sum_{m=1}^{M} x_m \leq U(d)$ (e.g., $L(d) = 1$ under LF). Moreover, soft constraints should contribute *only when an endpoint is selected*. We introduce auxiliary variables $y_{m,j} \equiv (x_m \wedge \ell_{m,j})$ and maximize,

$$\max_{\mathbf{x},\mathbf{y}} \sum_{m=1}^{M} \sum_{j \in \Phi_S(d)} w_j(d)\, y_{m,j} - \lambda(d) \sum_{m=1}^{M} x_m,$$

subject to (i) hard constraints $x_m \to \ell_{m,j}$ for all $j \in \Phi_H(d)$ and (ii) definitional constraints $y_{m,j} \leftrightarrow (x_m \wedge \ell_{m,j})$. This yields a weighted MaxSMT instance in which language feedback injects weighted clauses, while $\lambda(d)$ captures a preference for smaller (more robust) recommendation sets. In our experiments, we do not execute an external solver; instead, we treat the LLM router as a black-box policy and analyze its outputs through this MaxSMT lens by testing whether observed selections are consistent with clause satisfaction and implicit weights.

Regime-specific constraints. We distinguish two evaluation modes. (*i*) *User-facing recommendation* (LF): we impose a shortlist budget $L(d) \leq \sum_m x_m \leq U(d)$ and use $\lambda(d) > 0$ to encourage concise sets. (*ii*) *Completeness evaluation* (LF-PD): to measure whether the router captures a precise constraint, we disable shortlist regularization by setting $\lambda(d) = 0$ and drop the budget constraint, requiring $x_m \leftrightarrow \bigwedge_{j \in \Phi_H(d)} \ell_{m,j}$ so that all satisfying endpoints are included.

2.3 Postconditions

From the binary output mask C, we extract output set $\mathcal{C} \subseteq \mathcal{A}$ with $\mathcal{C} = \{a \in \mathcal{A} \mid C(a) = 1\}$. We consider six possible postconditions from the output set:

1. Zero(C) := ($|\mathcal{C}| = 0$)
2. One(C) := ($|\mathcal{C}| = 1$)
3. Some(C) := ($1 < |\mathcal{C}| < M$, and $\mathcal{C}$ exhibits a *structured* preference pattern)
4. All(C) := ($\mathcal{C} = \mathcal{A}$)
5. Random(C) := ($1 < |\mathcal{C}| < M$, and $\mathcal{C}$ is *unstructured*)
6. Fail(C) := C violates the required mask format prior to parsing/normalization

Neutrality as a hard clause. When $d = \perp$, the intended postconditions are Zero(C) or All(C). Under no-feedback, neutrality means the distribution over outputs should not systematically privilege particular endpoints beyond an explicitly defined tie-break rule. Any non-trivial subset choice must be explained by *implicit priors*, i.e., default non-uniform weights $w_j(\perp)$ that the LLM router carries even without feedback, forming the postcondition Some(C). This yields a testable prediction: no-feedback routing will preferentially select endpoints that satisfy a stable conjunction of "robustness" predicates (e.g., high capability, low price).

Case S. An output set cannot be determined as Some(C) or Random(C) when ($1 \leq |\mathcal{C}| \leq M$) until we analyze the results. We define this as **Case S** and our desired result as Case S* $\subseteq$ Case S.

$$\text{Case S} = \text{Some}(C) \vee \text{Random}(C)$$

Failure handling. To ensure robustness to variations in the LLM agent's behavior, we require that the output set remain well-defined regardless of the agent's execution status. In particular, we treat Fail(C) as a special case of other cases ($|\mathcal{C}| = 0 \to$ Fail(C) = Zero(C)) $\wedge$ ($|\mathcal{C}| = 1 \to$ Fail(C) = One(C)) $\wedge$ ($1 < |\mathcal{C}| < M \to$ Fail(C) = Case S). The processing rules for C are detailed in Appendix B.2.

3 Experiments

Setup. We evaluate an LLM router over OpenAI models with $M = 25$ endpoints and $K = 34$ objective attributes (Appendix B.3). We remove the model names when feeding the model zoo to the agent to prevent possible external knowledge leakage (Dodge et al., 2021; Lee et al., 2022; Carlini et al., 2023).

Direction keys. We categorize direction keys into: (i) **LF** (language feedback) and (ii) **NF** (no feedback; $d \in \{\texttt{NONE}, \texttt{NONE.}\}$). Within LF, we further distinguish:

- **LF-PD** (precise description): e.g., "I want a model with cheaper cached input"; "I want a model with cheaper output prices."

- **LF-GD** (general direction): e.g., "I want a cheaper model."

We use output price as the baseline attribute for comparison.

Metrics. Let $T(d)$ indicate the *ground-truth target set* induced by an interpreted constraint (e.g., models with strictly cheaper output price than a_0). We evaluate $\delta(\mathcal{C};T) = \frac{|\mathcal{C}\cap T|}{|T|}$ and $\mathcal{P}(\mathcal{C};T) = \frac{|\mathcal{C}\cap T|}{|\mathcal{C}|}$, with $\delta(\mathcal{C};T)$ denoting the coverage percentage and $\mathcal{P}(\mathcal{C};T)$ denoting the precision of the output set.

3.1 LF ANALYSIS

In Figure 1, the agent achieves $\mathcal{P} = 100\%$ for both scenarios, therefore forming Case S* = Some(C). However, even when asked to follow a specific direction, the agent still ignores some models ($\delta = 93.1\%$ for LF-PD and 80.0% for LF-GD). This suggests that under uncertainty, the agent tries to offer a smaller set for robustness.

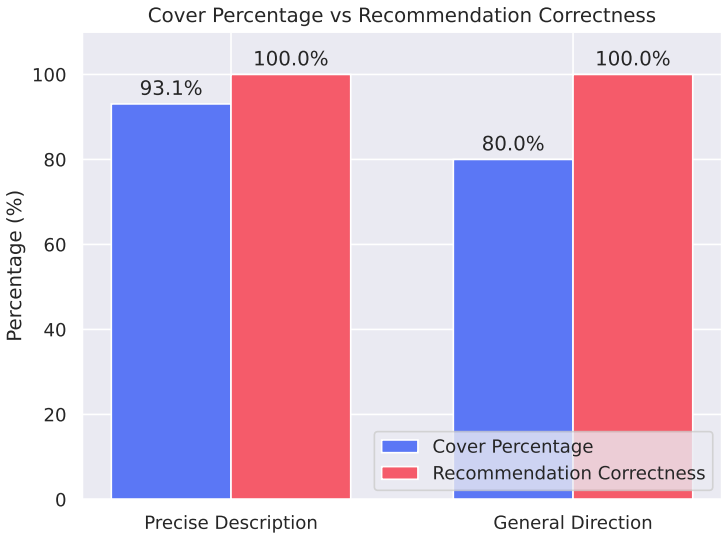

Figure 1: Coverage Percentage and Recommendation Precision of both LF groups

3.2 NF ANALYSIS

When $d = \perp$ (NF), the intended semantics is a vacuous constraint set $\Phi(d) = \emptyset$; thus the neutral postconditions are Zero(C) or All(C). However, in around 50% of cases (Table 1), its postcondition presents Case S.

NF behavior is best explained not as arbitrary randomness, but as implicit weighted satisfiability: the router preferentially selects endpoints satisfying a stable conjunction of high-weight "robustness" predicates even when the user provides no constraints. We further make the implicit-prior hypothesis falsifiable in Appendix B.5. This also suggests Case S = Some(C).

Algorithmic interpretation. The fitted $\hat{w}(\perp)$ induces a per-endpoint prior score $s_m(\perp) = \sum_j \hat{w}_j(\perp)\psi_j(a^{(m)})$. Selecting endpoints with large $s_m(\perp)$ yields an explicit baseline predictor for Some(C) and makes the MaxSMT abstraction operational: even with $\Phi(\perp) = \emptyset$ at the interface level, the router behaves as if it injects a small set of high-weight robustness clauses by default.

4 FUTURE DIRECTIONS.

In this paper, we showed that language-conditioned LLM routing can be usefully interpreted as structured constraint optimization. While our LF experiments demonstrate high-precision, near-feasible recommendation sets, our analysis primarily focuses on the *no-feedback* (NF) regime, where the intended semantics is a vacuous constraint set. The router frequently violates neutrality by returning a consistent, structured subset rather than All(C) or Zero(C). We view this reproducible structure as evidence of implicit clause weights $w(\perp)$ – a form of stable, language-free *default reasoning* that can be probed through logical postconditions and explained via a small predicate library, directly connecting routing behavior to LLM logical reasoning and consistency.

The next steps would be to generalize and strengthen our finding by further experimenting on a larger range of agents and prompts. Moreover, we look forward to developing mitigation methods.

REFERENCES

Carlos AnsóTegui, Maria Luisa Bonet, and Jordi Levy. Sat-based maxsat algorithms. *Artif. Intell.*, 196:77–105, March 2013. ISSN 0004-3702. doi: 10.1016/j.artint.2013.01.002. URL https://doi.org/10.1016/j.artint.2013.01.002.

Artificial Analysis. Ifbench benchmark leaderboard. URL https://artificialanalysis.ai/evaluations/ifbench. Accessed: 2026-02-16.

A. Biere, A. Biere, M. Heule, H. van Maaren, and T. Walsh. *Handbook of Satisfiability: Volume 185 Frontiers in Artificial Intelligence and Applications*. IOS Press, NLD, 2009. ISBN 1586039296.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2022. URL https://arxiv.org/abs/2108.07258.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models, 2023. URL https://arxiv.org/abs/2202.07646.

Edward Chen, Natalie Dullerud, Thomas Niedermayr, Elizabeth Kidd, Ransalu Senanayake, Pang Wei Koh, Sanmi Koyejo, and Carlos Guestrin. Modeling multi-objective tradeoffs with monotonic utility functions. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026. URL https://openreview.net/forum?id=Cu3coRhlM8.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=cSimKw5p6R. Featured Certification.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. Hybrid llm: Cost-efficient and quality-aware query routing. In *Proceedings of the 12th International Conference on Learning Representations*, 2024.

Jesse Dodge, Ana Marasovic, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, Matt Gardner, and William Agnew. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

João Gama, Indrundefined Žliobaitundefined, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4), March 2014. ISSN 0360-0300. doi: 10.1145/2523813. URL https://doi.org/10.1145/2523813.

Pierre Hansen and Brigitte Jaumard. Algorithms for the maximum satisfiability problem. *Computing*, 44(4):279–303, 1990.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8424–8445, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.577. URL https://aclanthology.org/2022.acl-long.577/.

Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2018. ISSN 2326-3865. doi: 10.1109/tkde.2018.2876857. URL http://dx.doi.org/10.1109/TKDE.2018.2876857.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025.

OpenAI. Compare models. https://platform.openai.com/docs/models/compare. Accessed: 07 and 22 December 2025.

xAI. Grok 4.1 model card, 2025. URL https://data.x.ai/2025-11-17-grok-4-1-model-card.pdf. Accessed on 10 December 2025.

APPENDIX

A EXPERIMENTAL RESULTS

A.1 DATA AND FIGURES

Table 1: Count and Percentage of each outcome for different keys. Without specifying *how many models*, the Agent will always recommend some models ($|\mathcal{C}| > 1$) to the user.

Direction Key	Zero (%)	One (%)	Case S (%)	All (%)	Count
I want a model with cheaper cached input.	0	0	12 (100%)	0	12
I want a cheaper model.	0	0	5 (100%)	0	5
I want a model with cheaper output prices.	0	0	1 (100%)	0	1
NONE	0	0	101 (51.01%)	97 (48.99%)	198
NONE.	0	0	10 (50%)	10 (50%)	20

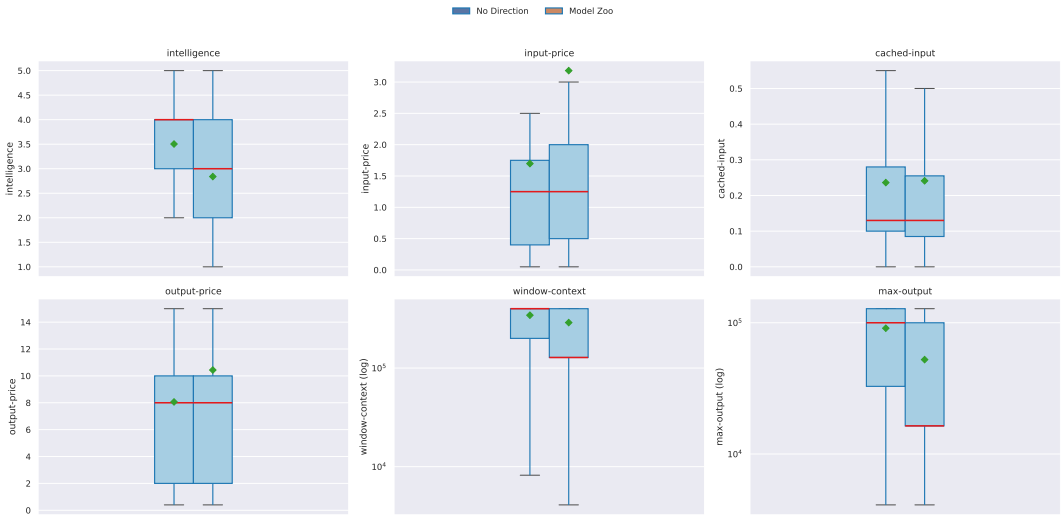

Figure 2: Percentile and Mean of NF set and model zoo on key objectives.

A.2 NF RESULTS

As seen in Figure 3, there are eight endpoints with clearly superior recommendation rates: o4-mini, GPT 5, GPT 5.1, o3, GPT 5 mini, and GPT 5.2, o3-mini, and GPT 5 nano. This set of choice has:

- The only model with the maximum intelligence score: o3.
- A set of intelligence = 4 with lowest price, and the only model with intelligence = 4 and speed > 3.
- One endpoint with intelligence = 3 and one with intelligence = 2, both of which have the lowest cost in their respective intelligence groups.
- All of them support *cached input*, meaning the potential lower cost at deployment.

Case S. The set of recommended endpoints provides a group of robust choices for the user to choose from. The router never recommends endpoints with intelligence $= 1$, and the price means are lower than that of the model zoo. Relative to the full model zoo, the sets also exhibit lower mean prices and substantially higher medians of context window and max output. Furthermore, the majority of recommended endpoints have reasoning enabled and cached input available.

This suggests that the router behaves as if it optimizes a non-uniform default preference structure, i.e., an implicit clause-weight vector $w(\perp)$.

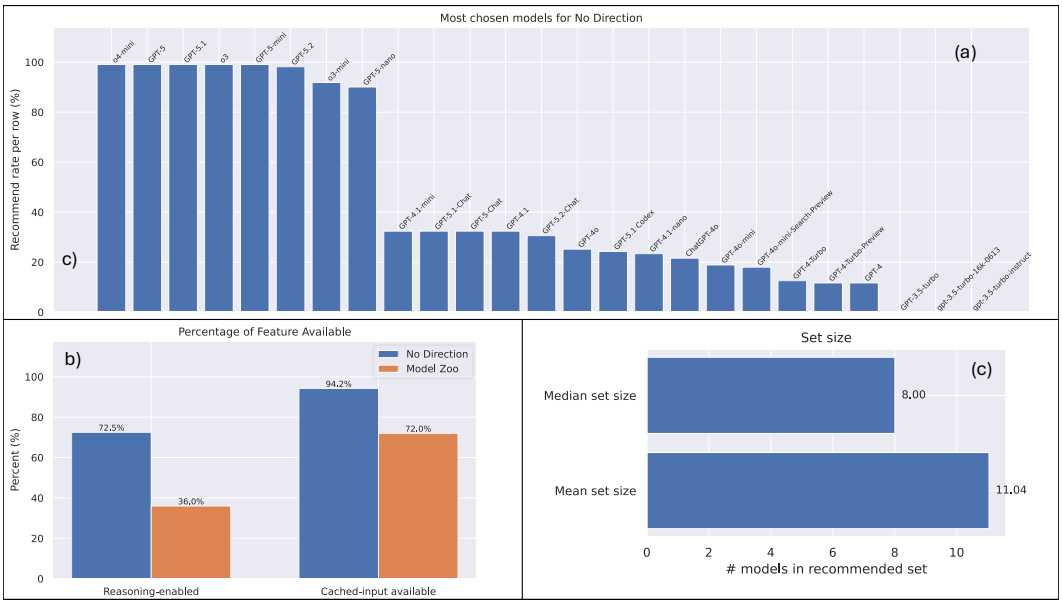

Figure 3: Direction key without language feedback set results for Case S. (a) per-endpoint selection frequency, b) reasoning/cached-input prevalence, (c) set-size statistics.

B Experiment Setup

B.1 LLM agent choice

We use Grok 4.1 Fast (xAI, 2025), a model with moderate capabilities in instruction following but fair long context reasoning, achieving 52.7% and 68.0%, respectively, from Artificial Analysis benchmarks (Artificial Analysis).

B.2 Output set extract rules

- Extract all standalone binary tokens 0, 1
- If $|C| < M$, append zeros until $|C| = M$
- If $|C| > M$, truncate $|C|$ to its first M elements

B.3 The "Model Zoo" Dataset

The 25 OpenAI (OpenAI) LLM and all the objective-level metadata are detailed in Table 2. When evaluating, we drop `model` and `model-id`.

Table 2: OpenAI Model Pool

id	model-id	model	Intelligence	Speed	Text In	Image In	Voice In	Video In	Text Out	Image Out	Audio Out	Video Out	Reasoning	Input Price	Cached Price	Output Price	Context Window	Max Output	Know. Cutoff	Comp. Endpt	Resp. Endpt	Assist. Endpt	Batch Endpt	Fine-Tune Endpt	Streaming	Func. Calling	Struct. Output	Fine-Tuning	Distillation	Pred. Outputs	Rate Lim (Free)	Rate Lim (T1)	Rate Lim (T2)	Rate Lim (T3)	Rate Lim (T4)	Rate Lim (T5)
1	openai/chatgpt-4o-latest	ChatGPT-4o	3	3	1	1	0	0	1	0	0	0	0	5	-	15	128k	16k	23-Oct	1	1	0	0	0	1	0	0	0	0	1	-	30k	450k	800k	2M	30M
2	openai/gpt-4.1	GPT-4.1	4	3	1	1	0	0	1	0	0	0	0	2	0.5	8	1M	32k	24-Jun	1	1	1	1	1	1	1	1	1	1	1	-	30k	450k	800k	2M	30M
3	openai/o4-mini	o4-mini	4	3	1	1	0	0	1	0	0	0	1	1.1	0.28	4.4	200k	100k	24-Jun	1	1	0	1	1	1	1	1	1	0	0	-	100k	2M	4M	10M	150M
4	openai/gpt-3.5-turbo-16k	GPT-3.5-16k	1	2	1	0	0	0	1	0	0	0	0	3	-	4	16k	4k	21-Sep	1	1	0	1	0	0	0	0	1	0	0	-	200k	2M	800k	10M	50M
5	gpt-3.5-turbo-instruct	GPT-3.5-Inst	1	2	1	0	0	0	1	0	0	0	0	1.5	-	2	4k	4k	21-Sep	1	1	0	0	0	0	0	0	1	0	0	-	200k	2M	800k	10M	50M
6	openai/gpt-3.5-turbo	GPT-3.5-Trb	1	2	1	0	0	0	1	0	0	0	0	0.5	-	1.5	16k	4k	21-Sep	1	1	0	1	1	0	0	0	1	0	0	-	200k	2M	4M	10M	50M
7	openai/gpt-4-turbo-prev	GPT-4-Prev	2	3	1	0	0	0	1	0	0	0	0	10	-	30	128k	4k	23-Dec	1	1	1	0	0	0	0	0	1	0	0	-	30k	450k	600k	800k	2M
8	openai/gpt-4-turbo	GPT-4-Turbo	2	3	1	1	0	0	1	0	0	0	0	10	-	30	128k	4k	23-Dec	1	1	1	1	0	1	1	0	0	0	0	-	30k	450k	600k	800k	2M
9	openai/gpt-4.1-mini	GPT-4.1-Mini	3	4	1	1	0	0	1	0	0	0	0	0.4	0.1	1.6	1M	32k	24-Jun	1	1	1	1	1	1	1	1	1	0	1	40k	200k	2M	4M	10M	150M
10	openai/gpt-4.1-nano	GPT-4.1-Nano	2	5	1	1	0	0	1	0	0	0	0	0.1	0.03	0.4	1M	32k	24-Jun	1	1	1	1	1	1	1	1	1	0	1	40k	200k	2M	4M	10M	150M
11	openai/gpt-4	GPT-4	2	3	1	0	0	0	1	0	0	0	0	30	0	60	8k	8k	23-Dec	1	1	1	1	1	1	0	0	1	0	0	-	10k	40k	80k	300k	1M
12	openai/gpt-5.1-codex	GPT-5.1-Code	4	3	1	1	0	0	1	0	0	0	1	1.25	0.13	10	400k	128k	24-Sep	0	1	0	0	0	1	1	1	0	0	0	-	500k	1M	2M	4M	40M
13	openai/gpt-4o-mini-search	GPT-4o-Search	2	4	1	0	0	0	1	0	0	0	0	0.15	-	0.6	128k	16k	23-Oct	1	0	0	0	0	1	0	1	0	0	0	40k	200k	2M	4M	10M	150M
14	openai/gpt-4o-mini	GPT-4o-Mini	2	4	1	1	0	0	1	0	0	0	0	0.15	0.08	0.6	128k	16k	23-Oct	1	1	1	1	1	1	1	1	1	0	1	40k	200k	2M	4M	10M	150M
15	openai/gpt-4o	GPT-4o	3	3	1	1	0	0	1	0	0	0	0	2.5	1.25	10	128k	16k	23-Oct	1	1	1	1	1	1	1	1	1	1	1	-	30k	450k	800k	2M	30M
16	openai/gpt-5-chat	GPT-5-Chat	3	3	1	1	0	0	1	0	0	0	0	1.25	0.13	10	128k	16k	24-Sep	1	1	0	0	0	1	1	1	0	0	0	-	30k	450k	800k	2M	30M
17	openai/gpt-5-mini	GPT-5-Mini	3	4	1	1	0	0	1	0	0	0	1	0.25	0.03	2	400k	128k	24-May	1	1	0	1	0	1	1	1	0	0	0	-	500k	2M	4M	10M	180M
18	openai/gpt-5-nano	GPT-5-Nano	2	5	1	1	0	0	1	0	0	0	1	0.05	0.01	0.4	400k	128k	24-May	1	1	0	1	0	1	1	1	0	0	0	-	200k	2M	4M	10M	180M
19	openai/gpt-5.1-chat	GPT-5.1-Chat	3	3	1	1	0	0	1	0	0	0	0	1.25	0.13	10	128k	16k	24-Sep	1	1	0	0	0	1	1	1	0	0	0	-	30k	450k	800k	2M	30M
20	openai/gpt-5.1	GPT-5.1	4	4	1	1	0	0	1	0	0	0	1	1.25	0.13	10	400k	128k	24-Sep	1	1	0	0	0	1	1	1	0	1	0	-	500k	1M	2M	4M	40M
21	openai/gpt-5	GPT-5	4	3	1	1	0	0	1	0	0	0	1	1.25	0.13	10	400k	128k	24-Sep	1	1	0	1	0	1	1	1	0	1	0	-	500k	1M	2M	4M	40M
22	openai/o3-mini	o3-mini	4	3	1	0	0	0	1	0	0	0	1	1.1	0.55	4.4	200k	100k	23-Oct	1	1	1	1	0	1	1	1	0	0	0	-	100k	200k	4M	10M	150M
23	openai/o3	o3	5	1	1	1	0	0	1	0	0	0	1	2	0.5	8	200k	100k	24-Jun	1	1	0	1	0	1	1	1	0	0	0	-	30k	450k	800k	2M	30M
24	openai/gpt-5.2	GPT-5.2	4	4	1	1	0	0	1	0	0	0	1	1.75	0.18	14	400k	128k	25-Aug	1	1	0	0	0	1	1	1	0	1	0	-	500k	1M	2M	4M	40M
25	openai/gpt-5.2-chat	GPT-5.2-Chat	3	3	1	1	0	0	1	0	0	0	0	1.75	0.18	14	128k	16k	25-Aug	1	1	0	0	0	1	1	1	0	0	0	-	30k	450k	800k	2M	40M

B.4 PROMPTS

IMPORTANT: Your role is to choose which language models align with the user's preferences and DIRECTION.

You will receive:
- A free-form prompt that may contain a line 'DIRECTION: ...' describing the user's current preference direction (e.g. wanting a cheaper model).
- Metadata for the last chosen model, so you can compare (e.g. find models that are cheaper, faster, etc.).
- A CSV table of all available models.

There are [N] models (rows) in the CSV, in fixed order.

Your only task:
1) For each model (each row), decide if it matches the user's preferences and DIRECTION, possibly relative to the last chosen model.
2) Output exactly [N] integers (space-separated), one per row in the CSV, in the same order. Use '1' if the model is acceptable, '0' if it is not.
3) Output no other words.

Last chosen model (CSV; may be '(none yet'): [Last Model Metadata]
CSV of ALL models: [Full Model Zoo CSV]

B.5 NF IMPLICIT-PRIOR BASELINE

Data. We analyze NF runs ($d = \perp$) with $M = 25$ endpoints. We condition the explanatory model on Case S runs and exclude All(C) runs, since All(C) contains no information about relative endpoint preferences. In our data, out of 218 NF runs we observe 111 Case S runs and 107 All(C) runs.

Clause library. We construct an interpretable predicate library $\{\psi_j\}_{j=1}^{J}$ with $J = 19$ boolean predicates over endpoint metadata. The full list is: `cached_input_available`, `reasoning_enabled`, `cached_and_reasoning`, `int_ge_4`, `int_is_max`, `speed_gt_3`, `speed_ge_4`, `cheap_output_global_q0.25`, `cheap_input_global_q0.25`, `cheap_output_in_int_tier_q0.25`, `ctx_ge_median`, `ctx_ge_p75`, `maxout_ge_median`, `maxout_ge_p75`, `cheap_cached_price_global_q0.25`, `int_ge_4_and_cached`, `cached_and_cheap_output_in_tier`, `int_ge_4_and_cheap_output_in_tier`, `ctx_p75_and_cached`. Quantiles are computed over the full model zoo.

Model and hyperparameter selection. We fit ℓ_1-regularized logistic regression over run–endpoint inclusion pairs. To avoid leakage across endpoints within the same run, we select the regularization strength R via group cross-validation where groups are runs. We scan $R \in \{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10\}$ and compute CV-AUC. We choose the sparsest model whose CV-AUC is within 0.02 of the best CV-AUC, yielding $R = 0.02$.

Permutation test. To test whether the fitted model exceeds chance while preserving set-size effects, we permute endpoint identities independently within each run (thus preserving $|\mathcal{C}|$ per run), refit the model, and recompute metrics. p-values are computed as the fraction of permuted trials achieving metrics at least as large as the observed metrics.

Explanatory baseline. We instantiate this hypothesis by fitting a sparse explanatory model for NF endpoint inclusion conditioned on Case S. Let $x_m^{(r)} \in \{0, 1\}$ denote whether endpoint m is selected in NF run r. We construct a library of $J = 19$ interpretable boolean predicates $\{\psi_j\}_{j=1}^{J}$

over endpoint metadata and fit an ℓ_1-regularized logistic model:

$$\Pr(x_m^{(r)} = 1 \mid d = \perp,\ \text{Case S}) = \sigma\left(b + \sum_{j=1}^{J} w_j(\perp)\, \psi_j(a^{(m)})\right),$$

where $\sigma(\cdot)$ is the logistic sigmoid, b is an intercept, and $w(\perp) \in \mathbb{R}^J$ are clause weights. We condition on Case S runs because All(C) provides no information about relative endpoint preference. Hyperparameters are selected by group cross-validation over runs; we choose the sparsest model within 0.02 AUC of the best cross-validated AUC.

Results. Training on the 111 Case S runs, the selected sparse model achieves strong predictive power: pair-level AUC $= 0.8839$ (chance ≈ 0.5), Spearman correlation between predicted probabilities and empirical per-endpoint selection rates $= 0.8810$, and top-8 recovery $= 8/8$. A permutation test that preserves set sizes by permuting endpoint identities within each run confirms statistical significance: AUC under the null has mean 0.5031 while the observed AUC is 0.8839 ($p = 0.005$); top-8 overlap under the null has mean 2.81 while the observed overlap is 8 ($p = 0.005$); Spearman correlation is also significant ($p = 0.020$). Therefore, Case S exhibits Some(C) behavior, highly structured and can be captured by a small number of metadata predicates.

Recovered implicit clauses. At the selected regularization level, the learned $w(\perp)$ is sparse:

$$\hat{b} = -1.8043, \qquad \hat{w}(\perp) \neq 0 \text{ on 6/19 clauses.}$$

Table 3 lists the nonzero weights. The largest positive weights correspond to reasoning enabled, large max output (maxout $\geq$ 75th percentile), and long context (context $\geq$ median), as well as conjunctions involving cached input and reasoning. This provides an explicit explanation for the recurring NF robust core: when neutrality is violated (Case S), the router behaves as if it injects a small set of high-weight robustness clauses by default rather than selecting endpoints uniformly at random.

Clause ψ_j	Weight $\hat{w}_j(\perp)$
`maxout_ge_p75`	0.9317
`reasoning_enabled`	0.9272
`ctx_ge_median`	0.8011
`cached_and_reasoning`	0.7762
`ctx_p75_and_cached`	0.2243
`cached_and_cheap_output_in_tier`	0.0306

Table 3: Sparse estimate of implicit NF clause weights $\hat{w}(\perp)$ fitted on Case S runs (111 runs, $M = 25$ endpoints, $J = 19$ candidate predicates).