

The Impact of Language Mixing on Bilingual LLM Reasoning

Yihao Li¹, Jiayi Xin¹, Miranda Muqing Miao¹, Qi Long¹, Lyle Ungar¹

¹University of Pennsylvania

{liyihao, jiayixin, miaom}@seas.upenn.edu
qlong@upenn.edu, ungar@cis.upenn.edu

Abstract

Proficient multilingual speakers often intentionally switch languages in the middle of a conversation. Similarly, recent reasoning-focused bilingual large language models (LLMs) with strong capabilities in both languages exhibit **language mixing**—alternating languages within their chain of thought. Discouraging this behavior in DeepSeek-R1 was found to degrade accuracy, suggesting that language mixing may benefit reasoning. In this work, we study language switching in Chinese-English bilingual reasoning models. We identify reinforcement learning with verifiable rewards (RLVR) as the critical training stage that leads to language mixing. We show that language mixing can enhance reasoning: enforcing monolingual decoding reduces accuracy by 5.6 percentage points on MATH500. Additionally, a lightweight probe can be trained to predict whether a potential language switch would benefit or harm reasoning, and when used to guide decoding, increases accuracy by 2.92 percentage points. Our findings suggest that language mixing is not merely a byproduct of multilingual training, but is a *strategic reasoning behavior*.¹

1 Introduction

Multilingual speakers sometimes mix languages during reasoning, which is a phenomenon known in linguistics as *code-switching* (Appel and Muysken, 2005; Özkar et al., 2025). Though switching languages seems to disrupt coherence, multilingual speakers persist in this behavior for practical reasons. Languages vary in how they organize thoughts and some express certain concepts (e.g., numbers) more efficiently than others (Boroditsky, 2001; Haun et al., 2011; Pica et al., 2004; Miura et al., 1988). Language mixing helps them express ideas more precisely, fill lexical gaps when one language falls short (Kuzyk et al., 2020), and reduce

cognitive load by directing more mental effort toward the reasoning task itself (Lehti-Eklund, 2013).

Large Language Models (LLMs) have evolved from English-centric models to those with strong multilingual abilities, with some achieving true bilingualism through balanced English-Chinese training (Liu et al., 2024; Qwen et al., 2025). How these bilingual models differ from primarily monolingual LLMs raises intriguing questions for computational linguists. One striking phenomenon in this space is language mixing, with recent state-of-the-art RL-trained English-Chinese bilingual LLMs such as DeepSeek-R1 (Guo et al., 2025) and QwQ-32B (Team, 2024) displaying human-like language mixing behavior in their chain-of-thought: they respond in languages different from the prompt and switch languages (sometimes repeatedly) during their reasoning process.

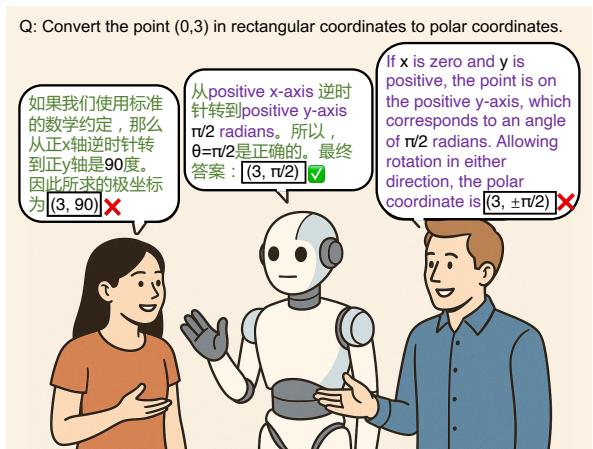


Figure 1: An illustration of bilingual code-switching improving reasoning performance. Two monolingual speakers, one in Chinese and the other in English, fail to solve a math problem, while an LLM robot that code-switches between both succeeds. Black text denotes language-agnostic content.

Proficient multilingual speakers of both languages can benefit from reasoning with code-switching. Can LLMs similarly benefit? (Fig-

¹Code available at <https://github.com/liyihao0302/language-mixing-in-llm-reasoning>

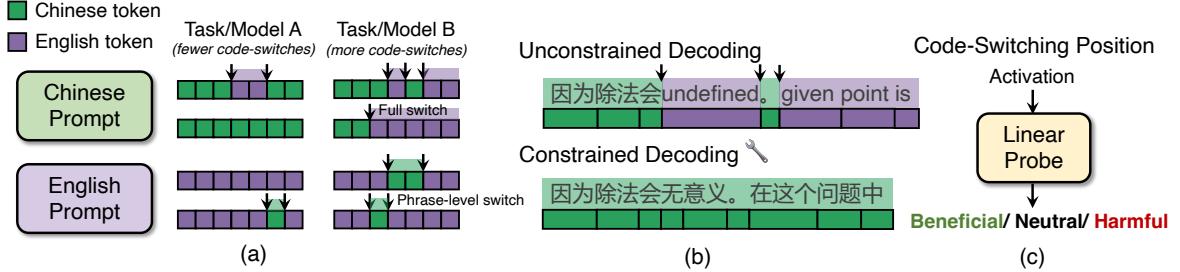


Figure 2: Overview of our analysis of language mixing in LLM reasoning. (a) We identify common language mixing patterns and triggers that lead to increased language mixing (Section 2). (b) We compare unconstrained bilingual outputs with constrained monolingual outputs to evaluate the impact of language mixing on reasoning performance (Section 3). (c) We train a probe to classify code-switches as {Beneficial, Neutral, or Harmful}, and use it to guide decoding (Section 4).

ure 1). The parallel seems plausible: both humans and LLMs potentially share needs for expressivity, precision, reduced cognitive load, and efficiency (which for LLMs translates to using fewer tokens and shorter context windows). Supporting this, DeepSeek-R1 demonstrates a performance degradation when a language consistency reward is introduced during training (Guo et al., 2025). These findings motivate our study into how language mixing affects LLM reasoning, centered on a key question: *Do LLMs reason better or worse with English-Chinese language mixing?*

To study whether language mixing **causally** improves reasoning performance, we first observe that stronger reasoning **correlates** with increased mixing, and identify reinforcement learning with verifiable rewards (RLVR) (Lambert et al., 2024) as the critical training stage that induces it (Section 2). We then test **causation** through interventions in both directions: a) *decreasing* language mixing through constrained monolingual decoding degrades performance (Section 3); b) *strategically enhancing* language mixing through probe-guided decoding improves performance (Section 4). Together, these findings suggest language mixing is not a random artifact but a potentially deliberate, *useful strategy* for enhancing LLM reasoning.

Our contributions are summarized as follows:

- ★ We demonstrate that bilingual chain-of-thought reasoning with language mixing **causally** enhances performance: unconstrained bilingual outputs significantly outperforming monolingually constrained ones ($p < 0.05$), and probe-guided yields further gains.
- ★ We identify RLVR as the critical training stage that triggers language mixing, suggesting this

behavior may emerge from natural optimization.

- ★ We introduce probe-guided decoding, which integrates a lightweight real-time probe in the generation loop to trigger beneficial bilingual switches, improving reasoning with minimal overhead.

2 Where does Language Mixing Occur?

2.1 Detecting Code-Switches

Code-switching, by definition, means switching between languages in a single conversation. As illustrated in Fig. 2(a), segments of Chinese (in green) and segments of English (in purple) alternate, and these transitions represent code-switching occurrences. In written text, elements such as mathematical expressions or code (typically composed of English tokens) are language-agnostic and universally used across speakers of different languages. Thus, a paragraph written in Chinese that includes mathematical expressions using English tokens should not be considered language mixing. We define a *code-switching position* as the first **text** token (in either English or Chinese) where the language switches from one to another, excluding any language-agnostic content such as math expressions. These positions correspond to the arrow markers shown in Fig. 2(a).

Based on this definition, we implement a rule-based procedure to detect Chinese-English code-switching. We first filter out language-agnostic tokens (e.g., LaTeX math) and then distinguish English from Chinese tokens using ASCII encoding (see Appendix A.4 for details). We evaluate code-switching behavior with three key statistics on bilingual datasets that contain parallel English-Chinese versions of each problem (by translating from the original language):

Table 1: Language-mixing statistics across QwQ and DeepSeek-R1 series for Chinese (ZH) and English (EN) prompts. **% Prob.**: percentage of problems with code-switch; **Switch**: average number of switches per problem; **Tokens/Switch**: mean tokens between consecutive switches; **Non-prompt (%)**: fraction of tokens in a language different from the prompt; **White**: pretraining only; **Grey**: SFT + RLHF; **Pink**: RLVR. DeepSeek-V3-Base is excluded from interpretation due to frequent generation of irrelevant content.

Model	ZH				EN			
	% Prob.	Switch	Tokens/Switch	Non-prompt (%)	% Prob.	Switch	Tokens/Switch	Non-prompt (%)
Qwen2.5-32B	14.8%	1.98	667.96	1.42%	0.0%	0.00	0.00	0.00%
Qwen2.5-32B-Instruct	8.8%	0.36	1986.71	0.23%	0.0%	0.00	0.00	0.00%
QwQ32B-Preview	77.4%	7.22	217.03	4.28%	0.6%	0.02	1.50×10^5	0.00%
QwQ32B	29.2%	6.20	585.85	0.48%	0.5%	0.01	2.85×10^5	0.00%
DeepSeek-V3-Base	32.2%	9.95	190.78	2.53%	4.2%	1.51	980.76	1.18%
DeepSeek-V3	8.4%	0.39	3574.98	0.08%	0.4%	0.01	1.50×10^5	0.02%
DeepSeek-R1-Zero	10.9%	0.21	7048.94	0.82%	0.0%	0.00	0.00	0.00%
DeepSeek-R1	27.1%	4.39	688.31	0.38%	0.0%	0.00	0.00	0.00%
DeepSeek-R1-Distill-Llama-8B	23.6%	2.46	1128.53	0.31%	0.0%	0.00	0.00	0.00%
DeepSeek-R1-Distill-Qwen-32B	21.2%	1.94	1292.15	0.24%	0.0%	0.00	0.00	0.00%

- **Switch count:** The total number of switches (back and forth) between languages when processing problems under English and Chinese prompts.
- **Tokens between switches:** The average number of tokens generated between consecutive language switches, quantifying how frequently the model alternates between languages measured in tokens.
- **Non-prompt language fraction:** The fraction of tokens generated in a language different from the prompt language (shown as the shaded area between arrows in Fig. 2(a)), measuring how long the model stays in the non-prompt language. This evaluates the extent and persistence of language mixing.

2.2 Tracing the Evolution of Language Mixing in LLMs

The development of multilingual reasoning LLMs has followed a temporal progression of introducing training strategies, with the introduction of generative pre-training, post-training methods such as supervised fine-tuning (SFT), reinforcement learning with human feedback (RLHF) (Ouyang et al., 2022), and most recently, reinforcement learning with verifiable rewards (RLVR). RLVR replaces learned reward models with simple "pass-fail" checks that assign fixed rewards for correct outcome (Lambert et al., 2024), achieving remarkable gains in reasoning (Chen et al., 2025a; Xie et al., 2025). Frequent English–Chinese mixing between the two highest-resource languages in open-weight models like QwQ32B (Team, 2024)

and DeepSeek-R1 (Guo et al., 2025) is a recent observation, likely triggered by the newly popularized RLVR training strategy. Here, we aim to identify exactly which training stage(s) trigger such pronounced English–Chinese mixing.

To do so, we trace the evolution of language-mixing behavior across iterations of QwQ32B and DeepSeek-R1 models. For the QwQ series, we examine Qwen2.5-32B (base model with pre-training only), Qwen2.5-32B-instruct (post-trained with SFT and RLHF) (Qwen et al., 2025), and two generations trained with RLVR: QwQ32B-preview (Team, 2024) and QwQ32B (Team, 2025). For the DeepSeek-R1 series, we analyze DeepSeek-V3-base (the foundation model with only pre-training), DeepSeek-V3 (with SFT and RLHF applied) (Liu et al., 2024), DeepSeek-R1-zero (a version without language consistency reward, where language mixing was documented), DeepSeek-R1 (with language consistency reward implemented), and various DeepSeek-R1 distilled variants (Guo et al., 2025). We evaluate language mixing occurrences across these models using MATH500 (in both English and Chinese versions). To ensure comparable analysis, we prompt base and instruct models for lengthy chain-of-thought reasoning to match output lengths (see Appendix A.3).

Comparing the three model groups in Table 1—base (white), SFT/RLHF (grey), and RLVR (pink)—we observe that RLVR models exhibit the most language mixing, followed by pre-trained base models, while SFT/RLHF models demonstrate the least mixing behavior.

Pretraining. During large-scale pretraining, LLMs are exposed to web-scale multilingual

<p>Phrase-level Switching for Precision or Efficiency</p> <p>→ 比如当 $x = 0$ 时, 因为除法会 undefined. <i>For example, when $x=0$, the division becomes undefined.</i></p> <p>→ Now, since AB is parallel to DE, the height from DE to C is the same as from AB to C, scaled by the ratio of 相似 triangles. <i>similar triangles</i></p>	<p>Using English for Technical Terminology</p> <p>→ 这看起来有点像 Cauchy 方程的变种。 <i>This looks somewhat like a variant of the Cauchy equation.</i></p> <p>→ 我记得有一个 Pythagorean identity: $1 - \sin^2 x = \cos^2 x$. <i>I remember there's a Pythagorean identity.</i></p> <p>→ 我记得有 polynomial long division 这种方法。 <i>I remember there's a method called polynomial long division.</i></p>
<p>Switching to Match Reasoning/Answer Formats</p> <p>→ wait a minute, 我需要确认一下这个结论。 <i>Wait a minute, I need to double-check this conclusion.</i></p> <p>→ maybe 我需要找到一个多项式 $r(x)$, <i>Maybe I need to find a polynomial $r(x)$,</i></p> <p>→ 所以, 最终答案是 13535. **Final Answer** <i>So, the final answer is 13535. **Final Answer**</i></p>	<p>Full Switching when Facing Cognitive Challenges</p> <p>→ 可能题目有误, 或者我哪里理解错了。或者, perhaps the problem is to show that $CD = 4$, ...[EN] <i>Maybe there's an error in the question, or I misunderstood something. Or, perhaps ...</i></p> <p>→ 也许我还需要进一步化简。或者, maybe I made a mistake in calculating the cross product. ...[EN] <i>Maybe I still need to simplify further. Or, maybe ...</i></p>

Figure 3: Four patterns of code-switching observed in LLM outputs. **Top left:** Phrase-level switching, often short and used for precision or efficiency. **Top Right:** Switching to English for technical terms. **Bottom left:** Switching to match reasoning or answer formats. **Bottom right:** Full switch to another language when the model is unable to find a solution.

corpora, yet they rarely encounter natural code-switched input, which is far more common in speech than in text. But training data contains natural code-switching (Wu et al., 2025), so LLMs may learn to code-switch to some extent. As Table 1 shows (white rows), Qwen2.5-32B exhibits minimal but non-zero code-switching, while DeepSeek-V3-Base displays frequent switches. However, we discount DeepSeek-V3-Base’s behavior as it tends to ramble with irrelevant content without proper ability to terminate generation with [EOS].

SFT and RLHF. Supervised fine-tuning (SFT) trains on human curated, high-quality responses that are predominantly monolingual for human readability, and reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) reinforces these monolingual outputs by assigning them higher rewards. As a result, language mixing is strongly suppressed: Qwen2.5-32B-Instruct and DeepSeek-V3-Instruct exhibit the fewest code-switching instances among the model families.

RLVR. RLVR does not constrain the reasoning chain—what matters most is reaching the correct outcome. By optimizing directly for outcome-based rewards, it explores a much larger search space without relying on human priors (similar to the shift from AlphaGo to AlphaZero (Silver et al.,

2017)). We observe that RLVR models produce frequent code-switching in both families, with QwQ-32B-preview exhibiting the most mixing at 77.4% for Chinese problems and 0.6% for English problems. As we later show, trajectories that mix languages tend to achieve higher accuracy, suggesting that RLVR’s outcome-focused optimization naturally encourages language mixing when it leads to better performance.

To understand why RLVR triggers language mixing, we analyze checkpoints from multiple training steps using Tina-Open-RS1 (Wang et al., 2025a). Our findings reveal that language mixing increases progressively with RL training steps, with code-switching responses consistently outperforming monolingual responses within trajectory groups. GRPO consequently upweights these higher-advantage code-switching responses, reinforcing the mixing behavior (detailed experimental settings and results in Appendix A.3).

Finally, we note findings within model families. Contrary to claims in their paper (Guo et al., 2025), DeepSeek-R1-Zero displays fewer code-switching instances than DeepSeek-R1 in our testing. This discrepancy may result from our use of greedy decoding. QwQ32B-preview exhibits substantially more code-switching than the newer QwQ32B release, potentially hinting at added language consistency constraints in the updated model. We select QwQ32B-preview for our subsequent analyses.

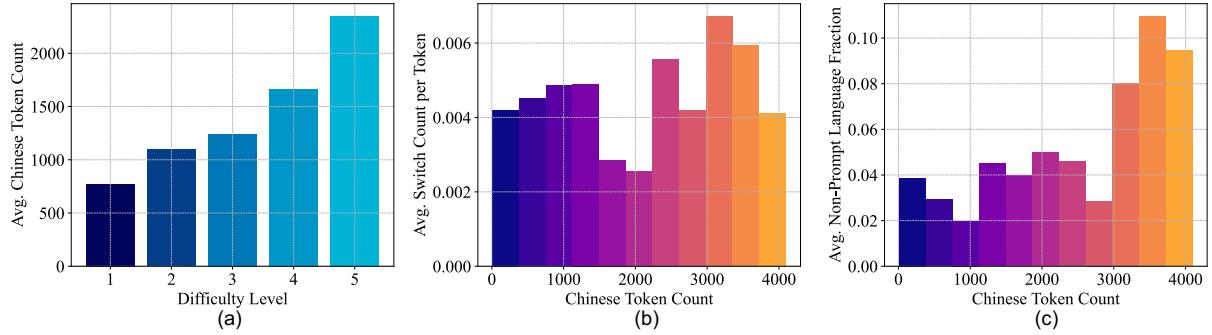


Figure 4: Quantitative analysis of language-mixing behavior in Math500 responses. (a) Correlation between problem difficulty level and response token count for Chinese prompts. (b) Normalized switch count and non-prompt language fraction as functions of token count, showing both code-switching frequency and non-prompt language use increase as chain-of-thought reasoning lengthens.

2.3 Characterizing Code-Switching Behavior

Code-switching patterns. Based on analysis of QwQ32B-Preview outputs, we identify four main patterns of switches as shown in Figure 3. The most common pattern is phrase-level switching in the top-left examples of Figure 3, driven primarily by a need for precision or efficiency. Certain concepts may be more clearly expressed in one language, with less ambiguity and often using fewer tokens. For instance, the use of *undefined* in the first example is more precise and less ambiguous than its Chinese counterparts: 无意义 (which can mean “meaningless,” as in “He felt his effort was meaningless”) or 未定义 (which may imply something is not yet defined but could be). It also requires fewer tokens—*undefined* is a single token, while both Chinese alternatives require two.

The second pattern (top right) involves switching to English for domain-specific terminology, likely because the model has limited capacity to store specialized translations across multiple languages. The third pattern (bottom left) shows language switching to conform to specific reasoning or answer formats, such as interjecting “wait, let me double check this” or concluding with “Final answer: ...” in English within otherwise Chinese responses. These formats may originate from supervised fine-tuning on data containing such patterns or reflect the model’s emergent self-reflective cues that aren’t well-aligned across languages. The fourth pattern (bottom right) involves switching entirely to another language when the model encounters difficulties or recognizes errors in its reasoning. This behavior may suggest a strategy to “clear its mind” or to seek cues in another language. However, this pattern typically appears in more

challenging problems, and even after switching languages, the model often fails to reach the correct solution.

These patterns, particularly the first two, mirror common human multilingual behavior and reveal the specific mechanisms through which language mixing may enhance reasoning.

Quantifying language mixing behavior. In QwQ-32B-preview responses to the MATH500 dataset, 77.4% of answers to Chinese prompts exhibit language mixing, with an average of 7.22 code-switches per problem, compared to just 0.6% for English prompts. It is already notable that English prompts (with math expressions fully in English tokens) occasionally trigger Chinese token generation. However, Chinese-to-English switching occurs far more frequently, indicating that English remains the model’s dominant or preferred language for reasoning.

We analyze how language mixing behavior relates to problem complexity and response length. Figure 4(a) demonstrates the correlation between token count in responses to Chinese prompts and MATH500 problem difficulty levels (5 discrete levels). Figures 4(b) and (c) quantify switch frequency and non-prompt language fraction as functions of token count. Since these statistics are normalized by token count, we can conclude that longer chain-of-thought reasoning exhibits slightly increased code-switching frequency and a growing fraction of non-prompt language use. This indicates that when tackling more difficult problems, the model adopts as a strategy to use greater language mixing—both switching between languages more frequently and shifting more toward the non-prompt language.

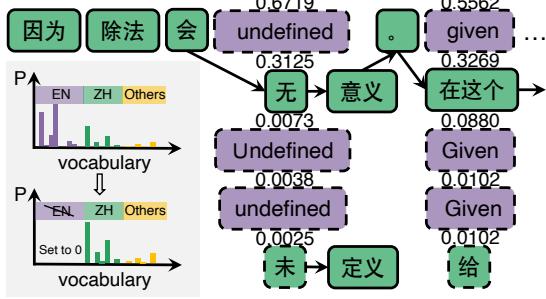


Figure 5: Token-level constrained decoding: We mask out tokens from the undesired language, forcing generation in the target language.

3 Do LLMs reason better or worse with language mixing?

3.1 Constrained Decoding

We expect that reasoning trajectories differ between monolingual and bilingual thinking, as languages have different structural focuses and are tied to distinct contexts (Keysar et al., 2012). Our goal is to determine whether these trajectories actually differ in practice and if one is superior to others in reasoning outcomes. Similar to how bilingual humans can be instructed to respond in a single language, we can constrain LLMs to generate outputs exclusively in one language. By applying this constraint, we ablate code-switching capabilities from the model, enabling direct comparison between unconstrained bilingual outputs and constrained monolingual outputs in terms of reasoning performance.

Specifically, during decoding we enforce token-level language constraints by allowing only tokens from the designated language (Fig. 5). We apply two types of constraints. In the **no-switch mode**, we prohibit the model from generating tokens in a particular language by masking those tokens in the vocabulary, which enforces strictly monolingual output. In the **forced-switch mode**, the model is required to switch languages at a specified token position, at which point only tokens from the target switch-to language are allowed.

3.2 Constrained vs. Unconstrained Decoding

Language mixing can enhance reasoning. Under the default unconstrained decoding, overall accuracy for MATH500 in English and Chinese is balanced. We then compared unconstrained bilingual outputs with constrained monolingual outputs under Chinese prompts (Figure 6(a)) and English prompts (Figure 6(b)). Forcing responses to be monolingual Chinese under Chinese prompts re-

duces accuracy by 5.6 pp. A paired t-test confirms this drop is statistically significant ($p = 0.0017$). These results provide strong evidence that LLMs can reason more effectively with language mixing than when restricted to monolingual outputs on certain reasoning tasks, potentially by leveraging the strengths of both languages.

Language mixing may also hurt reasoning. We then analyzed responses on Gaokao Cloze problems (Figure 7). Unconstrained responses to Chinese prompts outperform those to English prompts. This is as expected, since Gaokao-like problems (from the Chinese college entrance examination) would predominantly appear in Chinese within the pretraining data.

But contrary to our observations in MATH500, constrained monolingual Chinese decoding outperforms unconstrained bilingual decoding (Figure 7(a)). We attribute this to an imbalance in monolingual reasoning capabilities where Chinese performance exceeds English for these problems. However, the model still defaults to switching into English, a strategy that is generally helpful but detrimental in this setting.

These negative results for Gaokao Cloze do not imply that language mixing is inherently harmful. Rather, they suggest that QwQ32B-preview’s built-in mixing strategy is suboptimal. In the next section, we show that *strategic* language mixing, guided by a probe to decide when to switch, consistently improves LLM performance across datasets.

4 Can we steer the model toward strategic language mixing?

4.1 Probe-Guided Decoding

As we’ve shown in the previous section, language mixing is not always beneficial for reasoning. Code-switching can help, harm, or have no impact on the reasoning trajectories, which consequently impacts the overall reasoning outcome. Harmful code-switches may disrupt coherent reasoning chains, while helpful ones can reduce cognitive demands, address lexical gaps, or beneficially reset problematic reasoning directions. Here, we hypothesize that the beneficial, harmful, or neutral impact of each code-switch follows predictable patterns that could be decoded from model activations during generation.

To quantify the impact of code-switching, we compare full generations with and without a switch at each token position and label switches as

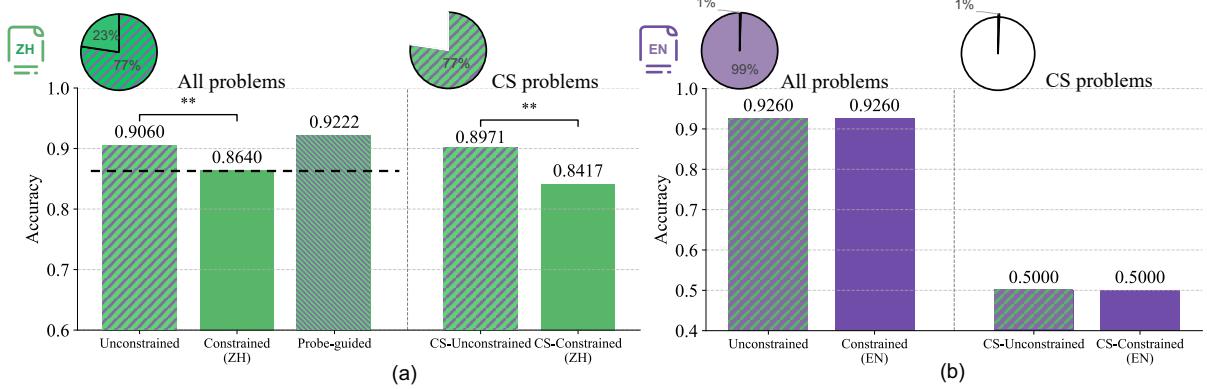


Figure 6: Comparison of accuracies for unconstrained, constrained, and probe-guided decoding on MATH500. (a) Chinese prompts (green); (b) English prompts (purple). Bar charts show accuracy for Chinese-only (solid green), English-only (solid purple), and language-mixing (hatched) outputs. Pie charts indicate the proportion of code-switching (CS) problems. Under Chinese prompts, unconstrained decoding significantly outperforms constrained monolingual decoding.

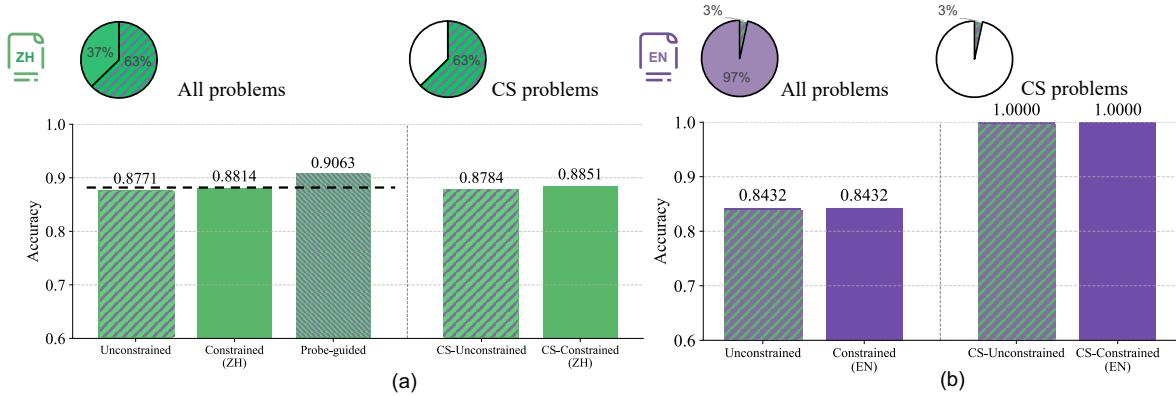


Figure 7: Comparison of accuracies for unconstrained, constrained, and probe-guided decoding on Gaokao Cloze. (a) Chinese prompts (green); (b) English prompts (purple).

{Beneficial, Neutral, or Harmful}. We apply constraints at a single token position, either by preventing a natural switch (no-switch mode) or forcing a switch where one would not naturally occur (forced-switch mode). A switch is labeled **Beneficial** if it leads to a correct answer that the monolingual version does not; **Harmful** if it causes an otherwise correct answer to become incorrect; and **Neutral** if it has no effect on the final output.

In practice, we collect all natural switching positions and synthesize additional switches at high language entropy positions. We then train a lightweight three-layer MLP probe (Fig.8) on hidden activations extracted from the LLM. We augment activations with three *meta features*: ① *is_natural* (natural or synthetic switch), ② *switch_direction* (Chinese to English or vice versa), and ③ *language_entropy* (entropy of the model’s predicted language distribution).

With the trained probe, we can control the decoding process by predicting online whether a code-switch is beneficial or harmful, and applying token-level constraints accordingly. If a natural switch is classified as **Harmful**, we suppress it using constrained decoding in no-switch mode. If a high language entropy position is classified as **Beneficial**, we trigger a forced switch at that step using forced-switch mode. This allows us to steer the model toward strategic language mixing with minimal computational overhead, using only a lightweight and easily deployable MLP probe during decoding.

4.2 Performance of Probe-Guided Decoding

Probe achieves positive utility score. Since the probe is ultimately used to guide decoding decisions rather than to precisely classify code-switching impacts, its effectiveness should be assessed in terms of its practical impact on multilin-

gual reasoning, i.e., the overall performance gain compared to unconstrained decoding, rather than classification metrics like F1 score. Here, we use decision utility as a measure to approximate the effectiveness of the probe given the helpful/ neutral/ harmful ground truth labels we collected from a large number of reasoning chains with code-switch interventions. (see Appendix A.7).

Specifically, we assign a utility score of $+1$ when the probe correctly classifies a **Harmful** or **Beneficial** switch, and a score of -1 when it suppresses a **Beneficial** switch or allows a **Harmful** one. We achieve positive utility score for all trained datasets, with the highest utility score being $s = 0.0107 \approx 1/93$. While this score may appear small, each Chinese prompt naturally results in about 8 potential code-switches on average. This means that 1 in 12 questions is expected to benefit directly from a correctly identified helpful switch, corresponding to a potential 8.3 pp gain in accuracy.

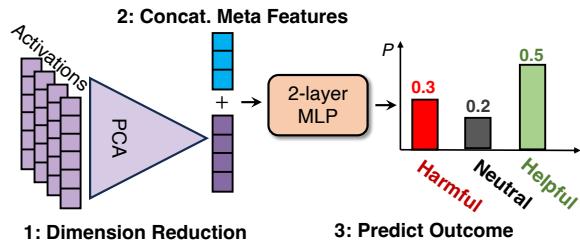


Figure 8: Architecture of the probe. The model classifies each code-switch as {**Beneficial**, **Neutral**, **Harmful**} using hidden activations at the switching step along with meta features.

Probe-Guided decoding further improves reasoning. We integrated the trained probe with optimal thresholds into our end-to-end decoding pipeline to assess its practical impact on LLM reasoning accuracy. We evaluate this intervention on the MATH500 and Gaokao Cloze, where it yields accuracy improvements of 1.62 pp and 2.92 pp, respectively.

We evaluate the probe’s effectiveness both within and across datasets. Given the relatively small problem sets, we conducted five random train/test splits to ensure robust evaluation. As shown in Table 2, our results demonstrate consistent improvements in both MATH500 and Gaokao Cloze. We further applied probes trained on one dataset to different test sets. Results show consistent accuracy improvements across most scenarios

(Table 3), demonstrating that our probe learns generalizable patterns that enhance reasoning through strategic language mixing.

Without improving the LLM’s inherent reasoning ability in either language, we achieve noticeable performance improvements solely by training a language decision module (the probe) to guide strategic language mixing. This is analogous to teaching a bilingual speaker to mix languages wisely without teaching them more math—the underlying knowledge remains unchanged, but the strategic mix of languages enhances problem-solving effectiveness. Importantly, these gains are not due to simple language dominance effects. For Gaokao Cloze, where Chinese dominates, our probe-guided strategy does not simply constrain all outputs to Chinese but instead introduces strategic English mixing, with 66.8% of interventions promoting English usage.

Examining the probe’s switching strategy reveals interesting patterns. Examples of helpful switches include converting "柯西-施瓦茨不等式" to "Cauchy-Schwarz inequality" to use more grounded English terminology, and in response to a Chinese prompt, switching from "表达式" to "notation" to create stronger referential coherence with the previously introduced notation (see full context in Appendix A.10).

Overall, as shown in the left panels of Figures 6 and 7, probe-guided *strategic* language mixing consistently outperforms the monolingual baseline (dashed line) by selecting near-optimal mixing strategies that boost reasoning performance.

5 Related Work

Multilingual Reasoning in LLMs. As LLMs have evolved from primarily English-centric systems to incorporate more balanced multilingual corpora, they have developed substantial multilingual capabilities (Cui et al., 2023; Faysse et al., 2025; Yang et al., 2024; Liu et al., 2024). However, these models still underperform when reasoning in non-English languages, particularly low-resource ones. This is evidenced by their superior performance on English-translated questions (Shi et al., 2022) and their tendency to switch to English against instructions (Marchisio et al., 2024; Hinck et al., 2024; Guo et al., 2025), a limitation long attributed to training data imbalance (Kew et al., 2024; Papadimitriou et al., 2023). Mechanistic interpretability studies have investigated

Table 2: Within-dataset evaluation of probe-guided decoding on MATH500 and Gaokao Cloze. Accuracy gain is reported in percentage points (pp).

Dataset	Test Utility (mean \pm std)	Accuracy Gain (pp; mean \pm std)
MATH500	0.0056 ± 0.0029	$+1.62 \pm 1.01$
Gaokao_Cloze	0.00014 ± 0.0040	$+2.92 \pm 2.12$

Table 3: Cross-dataset generalization of probe-guided decoding, indicating some transferability across datasets.

Train \rightarrow Test	Test Utility	Accuracy Gain (pp)
MATH500 \rightarrow Gaokao_Cloze	-0.0004	$+2.12$
Gaokao_Cloze \rightarrow MATH500	$+0.0024$	$+3.00$
<i>Evaluation on AIME2024</i>		
MATH500 \rightarrow AIME2024	$+0.0036$	$+0.00$
Gaokao_Cloze \rightarrow AIME2024	$+0.0054$	$+3.33$
MATH500 + Gaokao_Cloze \rightarrow AIME2024	$+0.0033$	$+3.00$

whether multilingual LLMs truly reason in non-English languages, revealing that some models can "think" in latent non-English languages for specific tasks (Wendler et al., 2024; Zhong et al., 2024a) and that distinct language-specific neural circuits exist within these systems (Zhao et al., 2024; Tang et al., 2024; Zhang et al., 2024). With the same aim of understanding multilingual reasoning in LLMs, we focus on bilingual code-switching *within* responses to examine how polyglot models reason differently from proficient monolingual systems.

Code-Switching in LLMs. Code-switching can emerge in LLMs from exposure to human-generated mixed-language text in training corpora (Wang et al., 2025b). The unintended mixing of languages in LLM outputs has been negatively characterized as language confusion, primarily observed when models processing low-resource languages shifted toward English during generation (Marchisio et al., 2024; Chen et al., 2025b; Wang et al., 2025b).

Only recently have models begun to more frequently mix English and Chinese—two high-resource and structurally distinct languages—within their reasoning chains. This behavior has emerged in models trained with reinforcement learning (Guo et al., 2025; Team, 2024; Xie et al., 2025), where optimizing for outcome-based rewards appears to override the preference for monolingual output. Notably, enforcing language consistency in DeepSeek-R1 resulted in a measurable drop in performance, suggesting a trade-off between language consistency and reasoning ability (Guo et al., 2025). Though a follow-up study using a smaller

model claimed language mixing harms reasoning, this conclusion was based on a single logic puzzle dataset and lacks generalizable evidence (Xie et al., 2025). Given these conflicting findings, our work aims to systematically evaluate the impact of code-switching on reasoning performance.

6 Conclusion

We investigate the impact of English-Chinese language mixing on LLM reasoning. First, we find that language mixing correlates with question difficulty, and by tracing model development, we show that RLVR training is the key trigger for this behavior. To establish causality between increased mixing and reasoning gains, we compare unconstrained bilingual decoding to constrained monolingual decoding on MATH500, finding a statistically significant accuracy boost for the bilingual outputs. Next, we train a lightweight probe to predict the utility of each potential switch and incorporate it into the decoding process across all datasets, resulting in consistent performance improvements over monolingual responses. Altogether, these results suggest that language mixing is not a random artifact of multilingual training but a deliberate strategy that LLMs adopt to improve complex reasoning.

For multilingual LLM reasoning, our findings suggest that language mixing can be beneficial, calling for caution before applying rigid language constraints and motivating future causal analyses. More broadly, we propose that language mixing can extend beyond spoken languages, occurring across modalities (Jiang et al., 2022; Chern et al., 2025). We encourage future research to explore these broader forms of language mixing in LLMs.

7 Limitations

Our study is limited in scope. The benchmarks we tested are limited to math tasks, and evaluating other domains such as science or logic puzzles is needed to assess the generality of our conclusions on LLM reasoning. We only focus on English-Chinese mixing, and it remains an open question whether similar patterns extend to other language pairs. Intervention analyses focus on QwQ32B-Preview, as broader evaluation across models is limited by the lack of access to RL-trained models that exhibit language mixing (public models such as DeepSeek-R1 and its distilled variants are constrained by enforced language consistency).

We also lack strictly controlled comparisons across training stages. RLVR models are typically trained with additional stages such as SFT, preventing us from isolating the specific contribution of RLVR. Our claim that RLVR promotes language mixing is therefore preliminary, and future work should test this more systematically on controlled model variants.

Finally, our use of hard constrained decoding may inherently reduce performance by imposing an extra language constraint. Future work could explore finer or continuous control over switching frequency and provide stronger empirical comparisons between unconstrained and constrained decoding.

References

René Appel and Pieter Muysken. 2005. *Language contact and bilingualism*. Amsterdam University Press.

Lera Boroditsky. 2001. Does language shape thought?: Mandarin and english speakers’ conceptions of time. *Cognitive psychology*, 43(1):1–22.

Hardy Chen, Haoqin Tu, Fali Wang, Hui Liu, Xianfeng Tang, Xinya Du, Yuyin Zhou, and Cihang Xie. 2025a. Sft or rl? an early investigation into training rl-like reasoning large vision-language models. *arXiv preprint arXiv:2504.11468*.

Yiyi Chen, Qiongxiu Li, Russa Biswas, and Johannes Bjerva. 2025b. Large language models are easily confused: A quantitative metric, security implications and typological analysis. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3810–3827.

Ethan Chern, Zhulin Hu, Steffi Chern, Siqi Kou, Jiadi Su, Yan Ma, Zhijie Deng, and Pengfei Liu. 2025. Thinking with generated images. *Preprint, arXiv:2505.22525*.

Yiming Cui, Ziqing Yang, and Xin Yao. 2023. Efficient and effective text encoding for chinese llama and alpaca. *arXiv preprint arXiv:2304.08177*.

Manuel Faysse, Patrick Fernandes, Nuno M Guerreiro, António Loison, Duarte Miguel Alves, Caio Corro, Nicolas Boizard, João Alves, Ricardo Rei, Pedro Henrique Martins, Antoni Bigata Casademunt, François Yvon, Andre Martins, Gautier Viaud, CÉLINE HUDELOT, and Pierre Colombo. 2025. CroissantLLM: A truly bilingual french-english language model. *Transactions on Machine Learning Research*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Daniel BM Haun, Christian J Rapold, Gabriele Janzen, and Stephen C Levinson. 2011. Plasticity of human spatial cognition: Spatial language and cognition covary across cultures. *Cognition*, 119(1):70–80.

Musashi Hinck, Carolin Holtermann, Matthew Lyle Olson, Florian Schneider, Sungduk Yu, Anahita Bhawandiwala, Anne Lauscher, Shao-Yen Tseng, and Vasudev Lal. 2024. Why do LLaVA vision-language models reply to images in English? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13402–13421, Miami, Florida, USA. Association for Computational Linguistics.

Albert Qiaochu Jiang, Sean Welleck, Jin Peng Zhou, Timothee Lacroix, Jiacheng Liu, Wenda Li, Mateja Jamnik, Guillaume Lample, and Yuhuai Wu. 2022. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *The Eleventh International Conference on Learning Representations*.

Tannon Kew, Florian Schottmann, and Rico Sennrich. 2024. Turning English-centric LLMs into polyglots: How much multilinguality is needed? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13097–13124, Miami, Florida, USA. Association for Computational Linguistics.

Boaz Keysar, Sayuri Lynn Hayakawa, and Sun Gyu An. 2012. The foreign-language effect: Thinking in a foreign tongue reduces decision biases. *Psychological Science*, 23(6):661–668.

Olivia Kuzyk, Margaret Friend, Vivianne Severdija, Pascal Zesiger, and Diane Poulin-Dubois. 2020. Are there cognitive benefits of code-switching in bilingual children? a longitudinal study. *Bilingualism: Language and Cognition*, 23(3):542–553.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Hanna Lehti-Eklund. 2013. [Code-switching to first language in repair—a resource for students’ problem solving in a foreign language classroom](#). *International Journal of Bilingualism*, 17(2):132–152.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. [Let’s verify step by step](#). In *The Twelfth International Conference on Learning Representations*.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, and 1 others. 2024. [Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model](#). *arXiv preprint arXiv:2405.04434*.

Kelly Marchisio, Wei-Yin Ko, Alexandre Bérard, Théo Dehaze, and Sebastian Ruder. 2024. [Understanding and mitigating language confusion in llms](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6653–6677.

Mathematical Association of America. 2024. American invitational mathematics examination (aime). <https://www.maa.org/math-competitions/aime>. Accessed: 2025-05-01.

Irene T Miura, Chungsoon C Kim, Chih-Mei Chang, and Yukari Okamoto. 1988. [Effects of language characteristics on children’s cognitive representation of number: Cross-national comparisons](#). *Child development*, pages 1445–1450.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. [Training language models to follow instructions with human feedback](#). *Advances in Neural Information Processing Systems*, 35:27730–27744.

Basak Özkara, Gülay Cedden, Christiane Von Stutterheim, and Patric Meyer. 2025. [Code-switching and cognitive control: a review of current trends and future directions](#). *Frontiers in Language Sciences*, 4:1515283.

Isabel Papadimitriou, Kezia Lopez, and Dan Jurafsky. 2023. [Multilingual bert has an accent: Evaluating english influences on fluency in multilingual models](#). In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1194–1200.

Pierre Pica, Cathy Lemer, Véronique Izard, and Stanislas Dehaene. 2004. [Exact and approximate arithmetic in an amazonian indigene group](#). *Science*, 306(5695):499–503.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, and 1 others. 2022. [Language models are multilingual chain-of-thought reasoners](#). In *The Eleventh International Conference on Learning Representations*.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. 2017. [Mastering chess and shogi by self-play with a general reinforcement learning algorithm](#). *Preprint*, arXiv:1712.01815.

Tianyi Tang, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Wayne Xin Zhao, Furu Wei, and Ji-Rong Wen. 2024. [Language-specific neurons: The key to multilingual capabilities in large language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5701–5715.

Qwen Team. 2024. [QwQ: Reflect deeply on the boundaries of the unknown](https://qwenlm.github.io/blog/qwq-32b-preview/). <https://qwenlm.github.io/blog/qwq-32b-preview/>. Accessed: 2025-05-01.

Qwen Team. 2025. [Qwq-32b: Embracing the power of reinforcement learning](https://qwenlm.github.io/blog/qwq-32b/). <https://qwenlm.github.io/blog/qwq-32b/>. Accessed: 2025-05-01.

Shangshang Wang, Julian Asilis, Ömer Faruk Akgül, Enes Burak Bilgin, Ollie Liu, and Willie Neiswanger. 2025a. [Tina: Tiny reasoning models via lora](#). *Preprint*, arXiv:2504.15777.

Zhijun Wang, Jiahuan Li, Hao Zhou, Rongxiang Weng, Jingang Wang, Xin Huang, Xue Han, Junlan Feng, Chao Deng, and Shujian Huang. 2025b. [Investigating and scaling up code-switching for multilingual language model pre-training](#). *arXiv preprint arXiv:2504.01801*.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. [Do llamas work in english? on the latent language of multilingual transformers](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394.

Linjuan Wu, Haoran Wei, Huan Lin, Tianhao Li, Baosong Yang, and Weiming Lu. 2025. [Enhancing llm language adaption through cross-lingual in-context pre-training](#). *arXiv preprint arXiv:2504.20484*.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. 2025. [Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning](#). *arXiv preprint arXiv:2502.14768*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan

Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. *Qwen2 technical report. Preprint*, arXiv:2407.10671.

Ruochen Zhang, Qinan Yu, Matianyu Zang, Carsten Eickhoff, and Ellie Pavlick. 2024. *The same but different: Structural similarities and differences in multilingual language modeling.* *arXiv preprint arXiv:2410.09223*.

Yiran Zhao, Wenxuan Zhang, Guizhen Chen, Kenji Kawaguchi, and Lidong Bing. 2024. *How do large language models handle multilingualism?* In *Advances in Neural Information Processing Systems*, volume 37, pages 15296–15319. Curran Associates, Inc.

Chengzhi Zhong, Fei Cheng, Qianying Liu, Junfeng Jiang, Zhen Wan, Chenhui Chu, Yugo Murawaki, and Sadao Kurohashi. 2024a. *Beyond english-centric llms: What language do multilingual language models think in?* *arXiv preprint arXiv:2408.10811*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2024b. *AGIEval: A human-centric benchmark for evaluating foundation models.* In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2299–2314, Mexico City, Mexico. Association for Computational Linguistics.

A Appendix

A.1 Overall Setup

All model inference is conducted using half-precision (float16) quantization. For the Qwen model series, we run inference on either three NVIDIA V100 GPUs (32 GB each) or a single NVIDIA A100 GPU (80 GB). Tasks involving probing and probe-guided decoding, which require extracting model activations are executed exclusively on the A100 GPUs. The DeepSeek family of models (including V3 and R1 variants) are queries with OpenRouter API.

Decoding is performed using greedy decoding with a temperature of 0.0, ensuring deterministic outputs across runs. We use flexible token limits, allowing each response to continue until the model emits an end-of-sequence token. In practice, we iteratively doubled the token budget until EOS or until a looping pattern was detected.

A.2 Evaluation datasets

We evaluate model behavior across four math reasoning benchmarks:

- Math500 ([Lightman et al., 2023](#)): A curated dataset of 500 high school and early undergraduate-level math word problems, designed to test symbolic reasoning and arithmetic across diverse topics.
- AIME2024 ([Mathematical Association of America, 2024](#)) Invitational Mathematics Examination): A benchmark of 30 challenging math problems targeted at advanced high school students.
- Gaokao Cloze: A set of standardized math questions from the Chinese college entrance examination. These problems are extracted from the AGIEval benchmark ([Zhong et al., 2024b](#)). Gaokao Cloze contains 118 fill-in-the-blank problems.

All problems are translated into both English and Chinese to support code-switching and bilingual evaluation. Translations are first generated using GPT-4o, followed by manual review and correction.

We evaluate model performance using Pass@1 accuracy, defined as the percentage of problems correctly solved on the first attempt. Specifically, We extract the final answer from the model’s output

Table 4: Pass@1 and token usage statistics by dataset and prompt language.

Dataset	Prompt Language	Pass@1	Token Count (mean \pm std, max)
MATH500	Chinese	0.906	2093.32 \pm 3492.81 (32768)
	English	0.926	2224.41 \pm 2609.91 (32768)
AIME2024	Chinese	0.367	11094.57 \pm 10110.03 (32768)
	English	0.533	6741.53 \pm 3738.44 (16384)
GAOKAO_CLOZE	Chinese	0.877	1516.70 \pm 1050.03 (5972)
	English	0.843	2473.74 \pm 1811.84 (8192)

using a pattern-based parser (*boxed*) and compare it against the ground truth. For outputs that do not exactly match the reference, we perform manual evaluation to account for correct answers expressed in alternative formats. When compared to the QwQ32B-preview (Team, 2024) report (MATH500 90.6%, AIME 50.0%), we observe no discrepancies in reasoning outcome (Table 4).

A.3 RLVR Triggers Language Mixing

Identifying training stages that trigger language mixing. To ensure a fair comparison, we explicitly prompt both base and SFT/RLHF models to produce extended chain-of-thought reasoning. On MATH500, their average token counts closely match those of the RLVR models but remain slightly lower (Table 5).

How does RLVR trigger language mixing? We examine RLVR checkpoints at successive training steps, where outcome rewards rise monotonically. Using Tina-Open-RS1 (LoRA+GRPO on DeepSeek-R1-Distill-Qwen-1.5B) (Wang et al., 2025a), we generate responses to a held-out set of English MATH500 problems and compare in Table 6(a) degree of language mixing and (b) average reward relative to group average for code-switching (CS) vs monolingual (Mono) responses. We find that:

- Language mixing increases progressively with RL training steps (25.0% \rightarrow 100%).
- Code-switching responses consistently outperform monolingual responses within trajectory groups (Avg Score CS $>$ Avg Score Mono). GRPO thus upweight these higher-advantage code-switching responses, reinforcing the mixing behavior.

A.4 Rule-Based Code-Switch Detection

First, we remove domain-specific mathematical content, which is language-agnostic by nature. We strip LaTeX-style math expressions using regular expressions that match content enclosed in dollar signs, () delimiters, and [] environments. In addition, we filter digits, parentheses, brackets, mathematical operators, and Greek letters, as these symbols are typically language-agnostic in reasoning contexts.

Second, we detect language boundaries by identifying continuous runs of characters belonging to either the Chinese Unicode range (U+4E00 to U+9FFF) or ASCII alphabetic characters. To improve precision, we implement the following filtering rules for English token candidates: ① We exclude domain-specific terms common in mathematical discourse, including mathematical functions (e.g., sin, cos, ln), standard variable names (e.g., ab, bc), and geometric designations (e.g., ABCD). ② Single-letter English tokens are discarded to prevent false positives from isolated variable names. ③ All-capitalized sequences of 2-3 characters are filtered, as these typically represent geometric entities rather than English words.

Finally, we identify language transitions by tracking adjacent language segments. A code-switch is recorded when the language classification of adjacent valid segments changes (e.g., from Chinese to English or vice versa). For each switch, we capture the switch direction, the text content at the boundary, and the position within the full response. Additionally, we track the starting language of each response to establish the baseline language context.

A.5 Detailed Implementation of Constrained Decoding

We apply constrained decoding using the same detection rule as in Appendix A.4, but perform code-switch detection online. Our state-transition model

Table 5: Mean token counts for Chinese (CH) and English (EN) prompts on MATH500 (float16, V100 \times 3, truncated at 4096).

Model	CH Token Count	EN Token Count
Qwen2.5-32B (Length-matched)	1041.37	970.75
Qwen2.5-32B-Instruct (Length-matched)	671.66	1008.99
QwQ32B-Preview	1580.44	1869.58
QwQ32B	3419.92	3546.39
DeepSeek-V3-Base (Length-matched)	1556.07	1792.66
DeepSeek-V3 (Length-matched)	1441.09	1604.09
DeepSeek-R1-Zero	1863.33	2145.74
DeepSeek-R1	2413.32	2457.40
DeepSeek-R1-Distill-Llama-8B	2467.37	2600.42
DeepSeek-R1-Distill-Qwen-32B	2185.93	2473.86

Table 6: Evolution of Code-Switching (CS) and Monolingual (Mono) performance across RL steps. Mean scores relative to the group average are shown in parentheses.

RL Step	Avg. Score (CS)	Avg. Score (Mono)	% CS
00	0.438 (+0.118)	0.281 (-0.039)	25.00%
400	0.524 (+0.047)	0.386 (-0.091)	65.63%
600	0.662 (+0.107)	0.407 (-0.148)	57.81%
800	0.681 (+0.064)	0.441 (-0.176)	73.44%
1200	0.555 (-0.007)	0.611 (+0.049)	85.94%
1800	0.688 (+0.000)	N/A	100.00%
2400	0.688 (+0.000)	N/A	100.00%

is defined as follows:

- **Math mode:** Enter math mode when a left bracket—{, [, (—is detected. Exit math mode when the matching right bracket—},],)—appears.
- **Chinese/English mode:** Switch to Chinese mode if a token’s Unicode code point falls within the CJK range. Otherwise, remain or switch to English mode.

To handle composite Chinese tokens—pairs of tokens that only form Chinese characters when combined—we scan for these specific sequences. Upon detection of any such composite token, we transition into Chinese mode.

A.6 Probing for Beneficial Code-Switches

To identify code-switching positions that are beneficial to reasoning accuracy, we train a lightweight probe (Figure 8) on hidden representations extracted from QwQ-32B-Preview. Specifically, we concatenate activations from a selected set of

transformer layers and project it into a lower-dimensional space using a PCA transformation fitted on the training set. We also add three related *Meta Features*: ❶ is natural (whether a switch is natural or synthetic), ❷ switch direction (whether the switch direction is from Chinese to English or not), ❸ language entropy (the entropy calculated from the model probability of output Chinese or English token). These feature are appended to the hidden embedding after PCA.

The probe comprises two separate three-layer MLP heads—one for “no-switch” and one for “forced-switch” decoding. We train it by jointly minimizing (1) the negative decision utility and (2) a weighted cross-entropy loss against an approximated three-class distribution (beneficial/neutral/harmful). Rather than using one-hot labels, we infer this distribution from the change in token count: if decoding without a switch produces substantially more tokens, we treat code-switching as more likely to be beneficial.

To address class imbalance, we use a class weight of {1.0, 0.1, 1.0} for {Beneficial,

Neutral, **Harmful**} to downweight the majority class (Neutral) and upweight the minority class (**Beneficial** and **Harmful**). During inference, the probe outputs predicted probabilities for each class after softmax. To maximize decision utility, we apply a thresholding strategy: if the predicted probability of a harmful switch exceeds a threshold τ_{harm} , we suppress the switch; if the predicted probability of a beneficial switch exceeds a threshold τ_{help} , we enforce the switch. Thresholds are selected via a grid search on a held-out validation set to maximize a custom utility metric that penalizes missed beneficial switches and incorrectly allowed harmful switches.

A.7 Probe Performance and Utility

Training Data Collection and Statistics. We collect training data for the probe using token-level constrained decoding focused on positions of natural switches or with high language entropy. We begin by identifying natural code-switching positions and apply the **no switch mode** to collect examples where switching is suppressed. If the number of natural switches falls short of a threshold, we supplement the dataset by introducing synthetic switches using the **forced switch mode**. We provide detailed statistics of the activation data we collected for Math500 and Gaokao Cloze in Table 7 and Table 8, respectively. The statistics reveal a strong class imbalance: the majority of code-switching instances fall into the Neutral category.

Table 7: Class distribution of the MATH 500 dataset across train, validation, and test splits.

Class	Train	Validation	Test
Harmful	773	127	204
Neutral	7,120	803	1,699
Helpful	894	125	241
Total	8,787	1,055	2,144

Table 8: Class distribution of the Gaokao Cloze dataset across train, validation, and test splits.

Class	Train	Validation	Test
Harmful	172	23	73
Neutral	1,427	199	394
Helpful	260	36	86
Total	1,859	258	553

The language entropy at each token position is defined as, where p'_{en} and p'_{ch} is reweighted to sum to 1.

$$H = -p'_{\text{en}} \log_2(p'_{\text{en}}) - p'_{\text{ch}} \log_2(p'_{\text{ch}}),$$

where p'_{en} and p'_{ch} are renormalized to satisfy $p'_{\text{en}} + p'_{\text{ch}} = 1$.

$$H_{\text{final}} = (p_{\text{en}} + p_{\text{ch}}) H.$$

This ensures that we only consider positions where both CH and EN are probable and uncertainty is high.

Hyperparameters and Experimental Setup.

We use a stratified train/validation/test split of 70%/10%/20% by problem ID, ensuring that code-switch examples from the same problem do not appear in multiple splits. All experiments are conducted on a single NVIDIA A100 GPU. The probe uses intermediate layer activation derived from five transformer layers (layers 63, 47, 31, 15, and 0), with additional metadata features. We reduce the input dimensionality using PCA, followed by a projection layer of dimension 512 and a hidden layer of size 512. Training is run for 30 epochs with a batch size of 256 and a learning rate of 1×10^{-4} .

Table 9: Hyperparameters for the probe model used across all datasets.

Hyperparameter	Value
Selected Layers	[63, 47, 31, 15, 0]
Use Metadata	True
PCA Dimension	512
Projection Dimension	512
Hidden Dimension	512
Coefficient for CE loss	0.5
Class Weights	[1.0, 0.1, 1.0]
Number of Epochs	30
Batch Size	256
Learning Rate	1×10^{-4}

Decision Utility. We use decision utility as an approximate measure of the probe’s effectiveness, based on our ground-truth labels of helpful, neutral, and harmful interventions collected from extensive reasoning chains with code-switch interventions.

These utility matrices (Table 10 and 11) reveal a strong asymmetry in decision costs—for example, in Case 1 any non-harmful prediction (N or B) yields zero utility, even though a balanced F1

Table 10: The utility matrix for Case 1: natural switch (we block when pred = Harmful)

GT ↓ Pred →	Harmful (H)	Neutral (N)	Beneficial (B)
Harmful (H)	+1	0	0
Neutral (N)	0	0	0
Beneficial (B)	-1	0	0

Table 11: The utility matrix for Case 2: no natural switch (we inject when pred = Beneficial)

GT ↓ Pred →	Harmful (H)	Neutral (N)	Beneficial (B)
Harmful (H)	0	0	-1
Neutral (N)	0	0	0
Beneficial (B)	0	0	+1

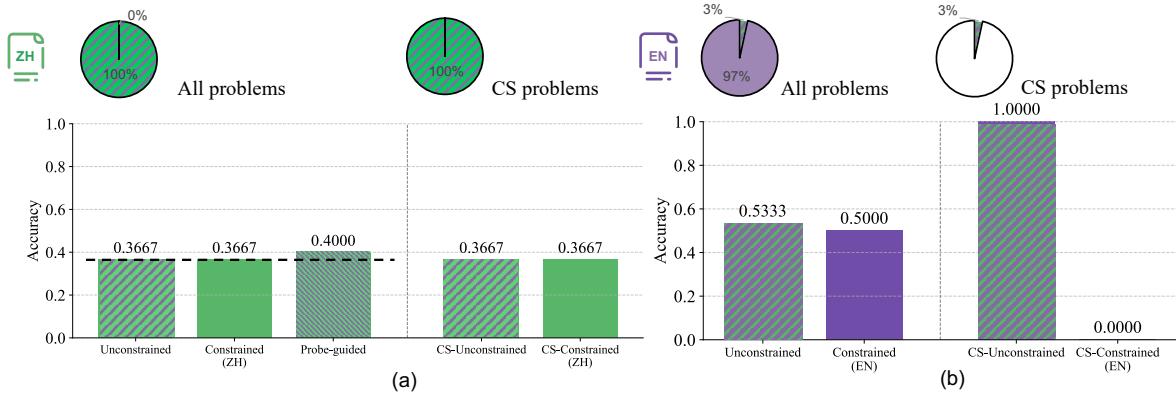


Figure 9: Comparison of accuracies for unconstrained, constrained, and probe-guided decoding on AIME2024. **(a)** Chinese prompts (green); **(b)** English prompts (purple).

would penalize N vs B errors equally. By focusing on *decision utility*, we target only those errors that actually affect decoding outcomes, making it a more appropriate measure than balanced classification metrics.

Evaluating Performance Gain. The lightweight probe is integrated into the decoding loop at each step. At every token position, we first generate the model’s natural output without intervention, then classify the position into Case 1—where a natural code-switch occurred—or Case 2—where no switch was generated but the language entropy exceeds a predefined threshold. In Case 1, the probe issues a decision to suppress the detected switch; in Case 2, it decides whether to inject a switch into the decoding stream.

With a strict train/validation/test partition, we measure net gains by contrasting probe-guided decoding with unconstrained decoding on the test set across five random splits. In Figures 6 and 7, the “probe-guided” bars represent the uncon-

strained baseline plus the average gain over those splits—not the raw performance, which would be inflated by training-set leakage.

A.8 Performance Comparison on AIME2024

We compare reasoning performance across unconstrained decoding, constrained monolingual decoding, and probe-guided decoding in Figure 9. For this experiment, the probe is trained solely on Gaokao Cloze and applied to AIME2024 with no AIME2024 training data. Unlike Figures 6 and 7, the probe-guided bar in Figure 9 shows raw performance, revealing a 3.33 pp net gain. This demonstrates both the probe’s cross-dataset generalization and the consistent benefit of strategic language mixing.

A.9 Examples of Language Mixing in QwQ-32B-Preview

Question: 将点(0, 3)从直角坐标系转换为极坐标系，需要找到对应的极径 r 和极角 θ 。极坐标表示为 (r, θ) ，其中 $r > 0$ 且 $0 \leq \theta < 2\pi$ 。

Answer: 首先，回忆一下直角坐标和极坐标的转换公式：

$$1. r = \sqrt{x^2 + y^2}$$

$$2. \theta = \tan^{-1}\left(\frac{y}{x}\right)$$

但是，这些公式在某些情况下需要特别处理，比如当 $x = 0$ 时，因为除法会`undefined`。

given point is (0,3), so $x = 0$ and $y = 3$ 。

首先计算 r ：

$$r = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$$

所以， $r = 3$

接下来是 θ ：

通常， $\theta = \tan^{-1}\left(\frac{y}{x}\right)$ ，但在这里 $x = 0$ ，所以这个表达式`undefined`。

这意味着点在 y 轴上。具体来说，since $x = 0$ and $y = 3$, which is positive y -axis。

在极坐标系中， θ 是从正 x 轴顺时针或逆时针测量到径向线的角度。

对于positive y -axis, θ is $\frac{\pi}{2}$ radians (or 90 degrees)。

所以， $\theta = \frac{\pi}{2}$

因此，极坐标是 $(3, \frac{\pi}{2})$

但为了确保没有错误，让我再检查一下。

点(0,3)位于直角坐标系的positive y -axis上。

在极坐标中， r 是点到原点的距离，which is 3，正确。

θ 是从positive x -axis到连接原点和点的线段的角度，顺时针或逆时针，取决于约定。在这里，我们使用标准的数学约定，即逆时针方向。

从positive x -axis逆时针转到positive y -axis是 $\frac{\pi}{2}$ radians。

所以， $\theta = \frac{\pi}{2}$ 是正确的。

另外，需要确认 θ 的范围是 $0 \leq \theta < 2\pi$, which $\frac{\pi}{2}$ satisfies。

所以，最终答案是 $(3, \frac{\pi}{2})$

Final Answer

$$\boxed{\left(3, \frac{\pi}{2}\right)}$$

Question: In the land of Ink, the money system is unique. One Trinket is equal to 4 Blinkets, and 3 Blinkets are equal to 7 Drinkets. In Trinkets, what is the value of 56 Drinkets?

Answer: Let's tackle this problem step by step. So, in the land of Ink, they have these different units of money: Trinkets, Blinkets, and Drinkets. The relationships between them are given, and I need to find out what 56 Drinkets are worth in Trinkets.

First, I need to understand the relationships between these units. It says that one Trinket is equal to 4 Blinkets. So, if I have one Trinket, it's the same as having 4 Blinkets. Got it.

Next, it says that 3 Blinkets are equal to 7 Drinkets. So, 3 Blinkets = 7 Drinkets. That means that one Blinket must be worth $7/3$ Drinkets, but maybe I don't need to go that far yet.

I need to find out what 56 Drinkets are worth in Trinkets. To do that, I probably need to convert Drinkets to Blinkets first, and then convert Blinkets to Trinkets.

So, let's start by converting Drinkets to Blinkets. If 3 Blinkets = 7 Drinkets, then how many Blinkets are there in 56 Drinkets?

I can set up a proportion for this. If 3 Blinkets are equal to 7 Drinkets, then x Blinkets are equal to 56 Drinkets. So, $3/7 = x/56$.

Wait, actually, that might not be the right way to set it up. If 3 Blinkets = 7 Drinkets, then the ratio of Blinkets to Drinkets is 3:7. So, to find out how many Blinkets correspond to 56 Drinkets, I can set up the proportion like this:

...

Then, to find out how many Trinkets are in 56 Drinkets, I divide 56 by the number of Drinkets per Trinket, which is $28/3$.

So, Trinkets = $56 / (28/3) = 56 * 3/28 = 6$ Trinkets.

一致的结果是6 Trinkets.

I think it's safe to conclude that 56 Drinkets are worth 6 Trinkets.

Final Answer

A.10 Examples of Probe-Guided Decoding

In the example below, the model initially switches to English for “Wait, no, in standard notation...,” then immediately switches back to Chinese with “所以，根据标准notation.”(So, according to the standard notation.) The probe intervenes at “表达式,” forcing it into English so that “notation” (marked in red) clearly refers back to the earlier “notation” mentioned in the preceding sentence.

Question: 在 $\triangle ABC$ 中, $B = 60^\circ$, $AC = \sqrt{3}$, 则 $AB + 2BC$ 的最大值为().

Answer: 在三角形ABC中, 已知角B=60度, 边AC= $\sqrt{3}$, 要求AB + 2BC的最大值。

首先, 这是个三角形的问题, 已知一个角和对边, 可以考虑用正弦定理或者余弦定理来处理。

已知角B和边AC (即边a, 因为对角B), 根据正弦定理:

$$a / \sin B = b / \sin A = c / \sin C = 2R$$

其中, $a = \sqrt{3}$, $B = 60^\circ$, R是外接圆半径。

$$\text{所以, } 2R = a / \sin B = \sqrt{3} / \sin 60^\circ = \sqrt{3} / (\sqrt{3}/2) = 2$$

$$\text{所以, } b = 2R \sin A = 2 \sin A$$

$$c = 2R \sin C = 2 \sin C$$

$$\text{所以, } AB = c = 2 \sin C$$

$$BC = a = \sqrt{3}$$

Wait, no, in standard notation:

$$a = BC, b = AC, c = AB$$

所以, 根据标准notation:

$$b = AC = \sqrt{3}$$

$$a = BC$$

$$c = AB$$

$$\text{所以, } a / \sin A = b / \sin B = c / \sin C = 2R$$

$$\text{所以, } 2R = b / \sin B = \sqrt{3} / \sin 60^\circ = \sqrt{3} / (\sqrt{3}/2) = 2$$

...

最终答案

$$2\sqrt{7}$$

In the following example, the probe switches “优雅” to “elegant” in the sentence “I prefer to find a more elegant approach.” In Chinese, calling a solution “elegant” can sound immodest, but English freely uses “elegant” in technical contexts. By switching languages, the probe avoids cultural awkwardness and uses a term that feels natural for describing a mathematical solution.

Question: 对于某些 x 值, $0 < x < 180$,

$$\begin{aligned} & \tan 53^\circ \tan 81^\circ \tan x^\circ \\ &= \tan 53^\circ + \tan 81^\circ + \tan x^\circ. \end{aligned}$$

求 x 。

Answer:

我遇到了一个三角函数的问题, 题目是:

对于某些 x 值, $0 < x < 180$, 有

$$\tan 53^\circ * \tan 81^\circ * \tan x^\circ = \tan 53^\circ + \tan 81^\circ + \tan x^\circ.$$

求 x 的值。

首先, 我需要理解这个等式的意思。看起来是一个关于正切函数的等式, 涉及到三个角度: 53度、81度和 x 度。这些角度的正切值之间有某种乘积等于它们的和。

我回忆了一下三角函数的identities, 看看有没有什么公式可以直接应用在这里。我记得有一个公式是关于三个角的正切加法公式, 特别是在当三个角加起来等于180度的时候。... 但是, 我更希望找到一个更elegant的方法, 而不必计算具体的tangent values。

Final Answer**

$$46^\circ$$

”