TOOLNEED: SELECTIVE VERIFICATION WITH MATCHED NO-TOOL BEHAVIOR

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool augmentation changes language models from generators into systems capable of initiating external actions. Before tool selection or argument generation comes a more fundamental decision: is external action warranted at all? We distinguish tool validity from behavioral necessity: a proposed action may be relevant, valid, executable, and even lead to a correct outcome, yet remain unnecessary relative to what the same model can accomplish without tools. In targeted construct analyses, matched no-tool behavior remains correct for 96.3% and 99.4% of successful valid tool paths on Qwen3.5 and DeepSeek, respectively. We introduce ToolNeed, a training-free, black-box-compatible method that selectively verifies proposed tool actions by eliciting matched no-tool behavior from the same target model and using it for contrastive arbitration. Candidate-blind routing determines when this verification is activated without conditioning on the concrete proposal. On SABEval, aggregate final-action Tool Invocation Rate falls from 91.2% to 8.1%, 43.0% to 8.5%, and 62.1% to 18.0% across three backbones. On the shared BFCL held-out Live subset and Non-Live split, ToolNeed achieves the highest Irrelevance accuracy in all six backbone–regime settings. Tool-required behavior is near-exactly preserved on Non-Live, while Live settings show modest preservation losses.

1 INTRODUCTION

Tool augmentation turns language generation into external action selection. A conventional language-model output expresses language; a tool-enabled output can initiate an interaction with an external system and alter the subsequent trajectory. This creates a first-class decision before tool selection, argument generation, and execution: *Should external action be initiated at all?* The answer shapes action correctness, trajectory quality, and calibrated interaction with external systems. Knowing how to act is incomplete without knowing when action is warranted.

Function calling evaluations have made substantial progress in measuring tool selection and argument correctness (Patil et al., 2025; Chen et al., 2025). External action calibration adds the decision of whether to engage those capabilities for the current request. Using SABEval (Liu et al., 2026), we find that our unmodified target models exhibit a systematic structural bias in action initiation. From D0 to D4, semantic irrelevance is held fixed as structural compatibility increases. Action initiation rises from 22.0% to 58.6% for Qwen3.7 and from 53.6% to 72.0% for Qwen3.5; DeepSeek remains extremely high at 88.9%–93.4%. These results show that models use interface compatibility as evidence for acting even when task necessity has not increased.

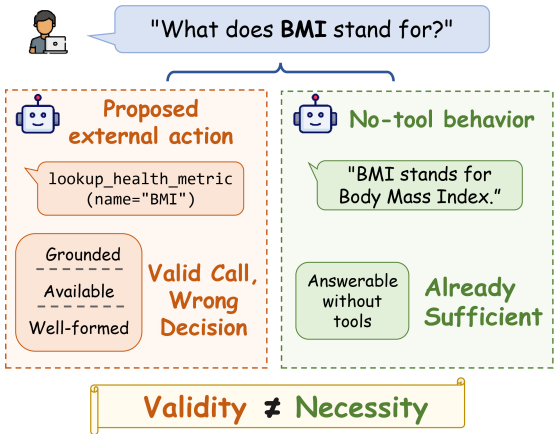


Figure 1: Motivation. A valid tool action can remain behaviorally unnecessary.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

This behavior motivates two distinct action-selection criteria. *Validity* asks whether a proposed action is well formed and compatible with the request and tool interface. *Necessity* asks whether external action is warranted relative to the capability available without it. A valid external action can remain unnecessary; Figure 1 illustrates this distinction with a minimal BMI example. In a targeted GSM8K analysis, matched no-tool behavior remains gold-correct for 96.3% of Qwen3.5 and 99.4% of DeepSeek naturally proposed tool actions that are relevant, valid, executable, and correct through the tool-assisted path. Validity is assessed from the proposal and task constraints; necessity emerges from a comparison with the behavior available without external action. Tool necessity is behavioral and comparative.

We introduce ToolNeed, a selective verification method that uses matched no-tool behavior to determine whether a proposed tool-enabled behavior should be retained. For the same conversation, target model, and decoding setting, ToolNeed removes tool availability to obtain this matched behavior. Candidate-blind routing activates verification conditionally, requirement-grounded verification evaluates candidate validity, and contrastive arbitration compares the Raw tool candidate with the matched no-tool behavior. ToolNeed is training-free and compatible with black-box model interfaces.

Our experiments follow the external action decision directly. Across all three backbones, SABEval shows large TIR reductions from Raw to ToolNeed as structural alignment increases. ToolNeed achieves the highest BFCL Irrelevance accuracy for every backbone on the shared held-out Live subset and Non-Live split. Non-Live tool-required behavior remains near-exactly preserved. Matched comparisons across evaluated regimes quantify the joint incremental contribution of matched no-tool behavior and contrastive arbitration.

Our contributions are threefold:

- We characterize the empirical distinction between tool validity and behavioral necessity: external action is warranted relative to what the same model can accomplish without it. SABEval exposes its behavioral consequence as structural alignment drives action initiation with semantic irrelevance held fixed.
- We introduce ToolNeed, a selective verification method that combines candidate-blind task grounding with matched no-tool behavior. Contrastive arbitration compares the Raw tool candidate and the matched behavior through training-free, black-box model access.
- Across three backbones, ToolNeed reduces SABEval final-action TIR by 34.4–83.1 points relative to Raw. On BFCL, it achieves the highest Irrelevance accuracy in all six backbone–regime settings; Non-Live tool-required behavior remains within one point of Raw.

2 Related Work

Tool-augmented language modeling developed from interleaved reasoning and environment interaction in ReAct to learned API-use decisions in Toolformer and large-scale instruction tuning over real-world APIs in ToolLLM (Yao et al., 2023; Schick et al., 2023; Qin et al., 2024). These systems established increasingly general mechanisms for selecting and executing external capabilities. We study the preceding decision boundary: whether external capability is warranted for the current request.

Adaptive tool use decisions. SMART studies tool overuse using rationales about when parametric reasoning or tool use is appropriate and the resulting data to train adaptive tool-using agents (Qian et al., 2025). MeCo is a fine-tuning-free trigger that requires access to model representations and quantifies metacognitive signals to decide when tools are needed (Li et al., 2025). When2Tool constructs controlled tool-necessity boundaries and shows that necessity is linearly decodable from pre-generation hidden states, motivating its Probe&Prefill intervention (Sun et al., 2026). Cheng et al. (2026) define tool necessity relative to each model's empirical capability and diagnose a knowing-doing gap using hidden-state probes. ToolNeed operationalizes model-relative necessity at inference time by comparing matched no-tool behavior through black-box interfaces. This lineage establishes adaptive tool engagement as a decision problem alongside tool selection and argument generation. ToolNeed provides training-free, black-box selective verification through matched no-tool behavior.

Runtime verification and behavioral intervention. ToolVerifier uses contrastive self-questions to distinguish close candidates during tool selection and parameter generation for unseen tools (Mekala et al., 2024). Reinforced Agent places a reviewer before tool execution and measures both helpful and harmful effects of inference-time feedback (Ta et al., 2026). COTA trains a small comparison-only advisor on same-prefix alternative branches and returns preferred alternatives as non-binding runtime advice (Jiang et al., 2026). ToolNeed combines candidate-blind task grounding, a controlled intervention on tool availability, and contrastive arbitration between realized tool-enabled and no-tool behaviors.

Evaluating tool use decisions. BFCL evaluates serial and parallel function calls, irrelevance/abstention, and broader agentic function calling behavior; ACEBench organizes tool use evaluation into Normal, Special, and Agent scenarios (Patil et al., 2025; Chen et al., 2025). When2Call evaluates decisions to invoke a tool, request clarification, or recognize that the available tools cannot answer a request (Ross et al., 2025). SABEval holds semantic irrelevance fixed as structural compatibility increases, exposing structural alignment bias in action initiation (Liu et al., 2026). Our evaluation connects these complementary views: SABEval evaluates unnecessary-action handling under increasing structural alignment, ACEBench supplies held-out legitimate-tool controls, and the shared BFCL held-out Live subset and Non-Live split test irrelevant-action handling and tool-required preservation.

3 METHOD

Let x be a conversation, $\mathcal{T}$ the available tools, and M the target model. Given $(x, \mathcal{T})$, the model produces a Raw tool candidate A_{tool}. We distinguish this proposal from A_{base}, the base behavior after selective requirement-grounded verification. ToolNeed selectively verifies proposed tool actions using two design principles: task interpretation precedes proposal inspection, and the retention decision is informed by matched no-tool behavior. Candidate-blind routing coordinates requirement-grounded verification and behavioral verification.

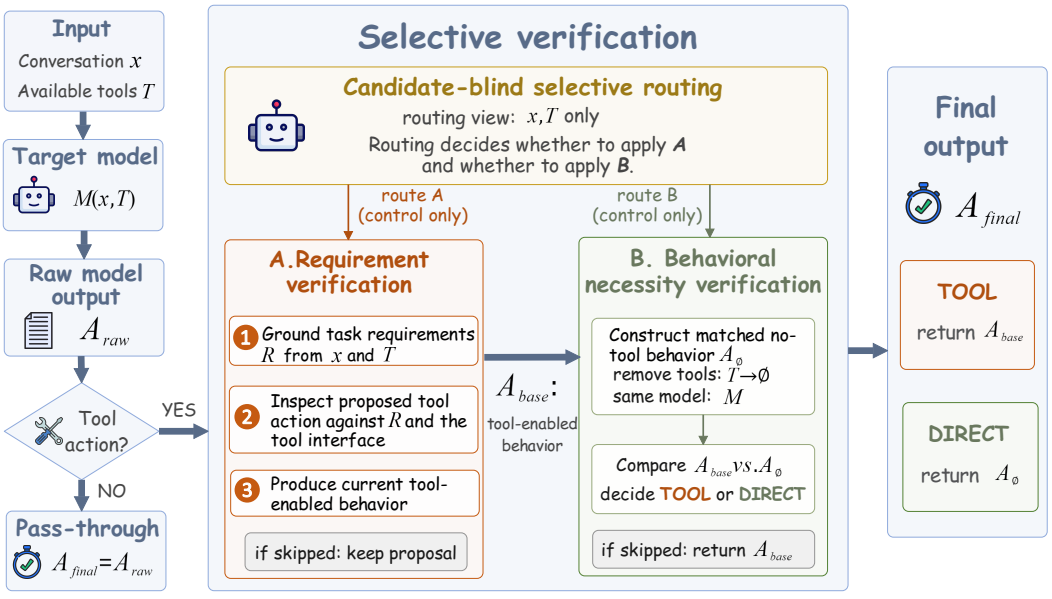


Figure 2: Overview of ToolNeed. Candidate-blind routing selectively activates requirement-grounded and behavioral verification for proposed tool actions. Contrastive arbitration compares the Raw tool candidate with matched no-tool behavior; TOOL returns the base behavior and DIRECT returns the matched no-tool behavior. Untriggered behavior passes through.

3.1 FROM TOOL VALIDITY TO TOOL NECESSITY

A candidate verifier can inspect whether A_{tool} names an available tool, supplies admissible arguments, and is grounded in the user's request. These properties establish validity. Necessity is relational: its value depends on the external action and the behavior available without it.

Let R denote requirements grounded in the conversation and tool interfaces. We make necessity observable through a controlled intervention on tool availability:

$$A_\emptyset = M(x; \mathcal{T} = \emptyset). \tag{1}$$

Here $A_\emptyset$ is the matched no-tool behavior produced by the same target model. It may be an answer, clarification, explanation, inability statement, refusal, or another ordinary language response. The implementation label DIRECT denotes selecting this behavior. Contrastive arbitration selects

$$y = V(x, \mathcal{T}, A_{\text{tool}}, A_\emptyset), \qquad y \in \{\text{TOOL}, \text{DIRECT}\}. \tag{2}$$

This construction evaluates necessity by contrasting the Raw tool candidate A_{tool} with $A_\emptyset$. Necessity belongs to their comparison; grounded requirements R provide the reference for validity. The arbitration decision governs whether the framework returns the base behavior A_{base} or the matched no-tool behavior.

3.2 REQUIREMENT-GROUNDED CANDIDATE VERIFICATION

Requirement-grounded verification establishes a reference for the task before candidate inspection. The routing view contains x and $\mathcal{T}$ and excludes the concrete function and arguments in A_{tool}. When verification is activated, requirements R are derived from the conversation and available tool interfaces, grounded against observable evidence, and then used to evaluate the proposed action. This ordering implements a simple principle: task interpretation precedes candidate evaluation. We denote the resulting base behavior by A_{base}; when the requirement path is skipped or preserves the proposal, $A_{\text{base}} = A_{\text{tool}}$. An INVALID Stage B decision replaces the candidate with the fixed inability response. A VALID decision retains the proposed tool behavior.

Separating task grounding from candidate inspection limits proposal-conditioned rationalization. Stage B retains candidate-aware checks of tool selection and argument consistency, and routing applies this validity criterion selectively.

3.3 BEHAVIORAL NECESSITY VERIFICATION

Behavioral necessity verification applies the intervention $\mathcal{T} \to \emptyset$ with the target model, conversation, and decoding regime fixed. Tool availability is the controlled change, and $A_\emptyset$ is the resulting matched no-tool behavior. Arbitration proceeds once $A_\emptyset$ has been generated successfully; reported evaluations contain completed $A_\emptyset$ generations and arbitration outcomes. The verifier receives x, the available tool schemas $\mathcal{T}$, A_{tool}, and $A_\emptyset$ and judges whether the external capability contributes enough to warrant action.

Behavioral verification can be activated through the helper Router when requirement-grounded verification is skipped for the same instance. Arbitration receives the Raw tool candidate A_{tool} and $A_\emptyset$. If $y =$ DIRECT, the system returns $A_\emptyset$; if $y =$ TOOL, it returns A_{base}, which can differ from the Raw tool candidate. Arbitration determines whether the proposed external action is warranted. Requirement verification determines the tool-enabled behavior returned when TOOL is selected.

3.4 SELECTIVE INFERENCE

Raw non-tool behavior passes through unchanged. For a Raw tool candidate, candidate-blind routing determines whether verification intervenes without conditioning that decision on the concrete proposal. A bypass preserves the current tool path. For Qwen targets, a Target Router `APPLY_HYBRID` decision activates the requirement and behavioral paths; a `PRESERVE_RAW` decision skips requirement verification and delegates the behavioral trigger to the helper Router. For the DeepSeek target, the Target Router itself supplies the behavioral trigger, avoiding a duplicate helper Router call for the same model. Behavioral comparison can run when requirement verification is skipped. Appendix A gives the operational procedure.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We evaluate external action calibration through unnecessary-action handling and legitimate-tool preservation. SABEval constructs semantically irrelevant tool contexts at increasing degrees of structural alignment, denoted D0–D4 (Liu et al., 2026). We report final-action Tool Invocation Rate (TIR, %, lower is better). Following SABEval, ACEBench-English Single and Parallel function calling categories provide legitimate-tool preservation controls (Chen et al., 2025). We pre-freeze 20 of the 100 examples in each category for development and evaluate all methods on the remaining 80 held-out examples, reporting category accuracy (%, higher is better). Table 1 reports these unnecessary-action and preservation evaluations together.

Following SABEval's seed-42 template-level split procedure, we sort the 101 template identifiers and deterministically instantiate a 40/20/41 train/validation/test split. We evaluate the frozen 41-template test subset: 41 templates $\times$ 10 sibling pairs $\times$ 5 queries yield 2,050 cases per D-level and 10,250 across D0–D4; development and test template overlap is zero. ACEBench controls are scored by its category checker. For BFCL v4 Live (Patil et al., 2025), we reserve a 200-example category-stratified confirmation set and evaluate all methods on its deterministic 2,051-example complement: 218 Simple, 1,009 Multiple, and 824 Irrelevance tasks. Non-Live contains 1,240 tasks: 400 Simple, 200 Multiple, 200 Parallel, 200 Parallel-Multiple, and 240 Irrelevance. Every backbone and method uses the same task IDs. All BFCL values are category accuracy; we report categories directly and construct no unified Overall score.

We compare Raw, Reason-then-Act (Sun et al., 2026), Reinforced Agent with Progressive Feedback (RA-PF), our benchmark-compatible adaptation of Progressive Feedback (Ta et al., 2026), and ToolNeed on DeepSeek-V4-Flash, Qwen3.7-Flash, and Qwen3.5-9B. RA-PF uses up to five reviewer decisions and four intervening target-model revisions, with early stopping upon acceptance. A DeepSeek-V4-Flash reviewer evaluates each provisional response; the evaluated target backbone generates revisions with the original tool interface. We preserve the progressive-feedback state machine and reviewer criteria, adapt serialization to each benchmark, and replace the reference GPT-4o configuration. For Qwen targets, the target model executes the Target Router, Stage A, Stage B, and matched no-tool generation. DeepSeek-V4-Flash executes the conditional helper Router and contrastive arbitration. For the DeepSeek target, the target model executes all verification components and skips the duplicate helper Router call. All three backbones use the same benchmark splits, comparison protocol, temperature-0 decoding convention, and evaluation procedures. Thinking is disabled throughout; Raw and matched no-tool generation use the same deterministic decoding regime. SABEval inference clusters by its 41 test templates, using 10,000 template-cluster bootstrap replicates for 95% confidence intervals and 100,000 paired sign-flip permutations for p values. BFCL uses exact McNemar tests and 10,000-sample task-level paired bootstrap confidence intervals. Holm correction uses six prespecified families: the six SABEval and BFCL Live Irrelevance ToolNeed–RA-PF headline tests; separate three-backbone SABEval families against Raw and Reason-then-Act; the six BFCL Live Simple/Multiple preservation tests against Raw; the three BFCL Non-Live Irrelevance tests against RA-PF; and the twelve BFCL Non-Live tool-required preservation tests against Raw. Mechanism analyses retain their reported inferential protocols outside these headline families. Model identifiers, evaluation procedures, and statistical protocols are summarized in Appendix B and Appendix C. Behavioral verification is selectively gated, with realized trigger frequency and inference overhead varying substantially across benchmarks and backbones (Appendix F). An anonymized implementation and reproduction instructions are provided in the supplementary material.

4.2 DOES STRUCTURAL ALIGNMENT BIAS ACTION INITIATION?

Raw TIR rises from D0 to D4 for all three backbones: 88.9% to 93.4% for DeepSeek, 22.0% to 58.6% for Qwen3.7, and 53.6% to 72.0% for Qwen3.5. User intent remains irrelevant to every offered tool as structural alignment increases, so task necessity is fixed across levels. The trend shows that structural alignment biases action initiation: models increasingly treat interface compatibility as evidence that external capability is warranted. Qwen exhibits the largest shifts, and DeepSeek sustains an extremely high action rate throughout.

Table 1: Unnecessary-action handling on SABEval with legitimate-tool preservation controls from ACEBench-English. SABEval reports final-action Tool Invocation Rate (TIR, %, ↓); ACEBench reports category accuracy (%, ↑). Single and Parallel each use 80 held-out examples after reserving 20 category examples for development. Best values are bold; second-best values are underlined within each model block.

Model	Method	SABEval					ACEBench	
		D0 TIR ↓	D1 TIR ↓	D2 TIR ↓	D3 TIR ↓	D4 TIR ↓	Single Acc. ↑	Parallel Acc. ↑
DeepSeek-V4-Flash	Raw	88.9	92.0	90.0	91.5	93.4	<u>56.3</u>	70.0
	Reason-then-Act	49.0	50.3	52.0	54.6	58.6	**60.0**	<u>71.3</u>
	RA-PF	<u>7.4</u>	<u>11.7</u>	<u>11.8</u>	<u>16.8</u>	<u>20.4</u>	<u>56.3</u>	**75.0**
	ToolNeed	**4.3**	**6.7**	**7.7**	**9.3**	**12.3**	<u>56.3</u>	68.8
Qwen3.7-Flash	Raw	22.0	37.0	45.6	51.6	58.6	<u>66.3</u>	<u>72.5</u>
	Reason-then-Act	8.6	9.6	10.6	<u>11.0</u>	<u>13.3</u>	61.3	48.8
	RA-PF	<u>5.3</u>	<u>8.0</u>	<u>9.5</u>	11.8	15.4	**67.5**	**77.5**
	ToolNeed	**4.4**	**7.2**	**8.9**	**9.5**	**12.6**	65.0	<u>72.5</u>
Qwen3.5-9B	Raw	53.6	56.1	62.3	66.3	72.0	<u>62.5</u>	<u>68.8</u>
	Reason-then-Act	29.1	31.5	32.7	34.1	36.5	60.0	56.3
	RA-PF	**10.5**	**13.2**	**13.0**	**16.2**	**20.8**	**63.8**	**73.8**
	ToolNeed	<u>11.1</u>	<u>16.2</u>	<u>18.8</u>	<u>20.1</u>	<u>23.8</u>	<u>62.5</u>	<u>68.8</u>

4.3 Necessity vs. Preservation

Relative to Raw, ToolNeed lowers aggregate SABEval TIR by 83.09, 34.44, and 44.09 points for DeepSeek, Qwen3.7, and Qwen3.5. Their 41-template cluster CIs are $[-88.30, -77.27]$, $[-42.14, -26.81]$, and $[-51.84, -36.09]$ (raw $p = 1.00 \times 10^{-5}$ each); all remain significant after Holm correction ($p_{\text{Holm}} = 3.00 \times 10^{-5}$ each). Against Reason-then-Act, adjusted p values are 3.00×10^{-5}, .18763, and 4.00×10^{-5} in the same order.

Relative rankings against RA-PF are backbone-dependent. DeepSeek favors ToolNeed ($\Delta = -5.54$, CI $[-8.55, -3.00]$, raw $p = 3.00 \times 10^{-5}$, Holm $p = .00012$). Qwen3.7 differs by -1.46 points ($[-3.52, +0.71]$, .18950, .2039), and Qwen3.5 favors RA-PF by $+3.25$ points ($[-0.46, +7.03]$, .10195, .2039); neither Qwen comparison is significant after correction.

ACEBench locates preservation independently of SABEval. ToolNeed exactly matches Raw on both Qwen3.5 controls and Qwen3.7 Parallel, differs by one item on Qwen3.7 Single, and matches DeepSeek Raw and RA-PF on Single (45/80). On DeepSeek Parallel, ToolNeed scores 55/80: one item below Raw (-1.25 points, C/R $= 0/1$, CI $[-3.75, 0]$, $p = 1$) and five below RA-PF (-6.25 points, $0/5$, $[-12.50, -1.25]$, $p = .0625$). These held-out category controls expose unnecessary-action handling and legitimate-tool capability as distinct empirical dimensions.

4.4 Generalization to BFCL

Table 2 evaluates all three backbones on the shared held-out Live subset and Non-Live split. ToolNeed achieves the highest Irrelevance accuracy in all six backbone–regime comparisons. Live gains over Raw are 32.40, 9.22, and 5.58 points for DeepSeek, Qwen3.7, and Qwen3.5. Gains over RA-PF are 14.08, 6.31, and 4.13 points (Holm $p = 5.904 \times 10^{-21}$, 6.42×10^{-10}, and .00012). Non-Live gains over RA-PF are 13.75, 3.75, and 4.58 points, all significant after correction (3.072×10^{-8}, .0078125, and .0192108). Appendix C reports paired outcomes and intervals.

Five of six Live Simple/Multiple comparisons show a significant decrease after Holm correction; Qwen3.5 Simple is the sole nonsignificant comparison. On Non-Live, 7 of 12 tool-required comparisons exactly match Raw, the remaining five differ by at most 1.0 point, and no significant degradation is detected after correction (Holm $p = 1$ throughout). RA-PF is strongest in several tool-required cells, and ToolNeed leads every Irrelevance comparison.

Table 2: BFCL category accuracy (%) on a shared held-out Live subset and the Non-Live split. Within each panel, all three backbones use the same task set. Tool-required categories measure preservation of legitimate function calling; Irrelevance evaluates behavior when the offered tools should not be invoked. All columns use BFCL category evaluators.

(a) BFCL v4 Live held-out subset

		Tool-required		**Irrelevance**
Model	Method	Simple ↑	Multiple ↑	Irrelevance ↑
DeepSeek-V4-Flash	Raw	61.9	67.1	49.2
	Reason-then-Act	61.5	67.7	61.8
	RA-PF	**62.8**	**68.7**	67.5
	ToolNeed	58.3	64.4	**81.6**
Qwen3.7-Flash	Raw	56.9	67.2	76.5
	Reason-then-Act	55.0	**67.6**	82.3
	RA-PF	**58.7**	67.4	79.4
	ToolNeed	53.2	66.3	**85.7**
Qwen3.5-9B	Raw	58.7	60.0	77.3
	Reason-then-Act	53.7	60.3	73.9
	RA-PF	**59.2**	**61.4**	78.8
	ToolNeed	57.8	59.0	**82.9**

(b) BFCL Non-Live

		Tool-required				**Irrelevance**
Model	Method	Simple ↑	Multiple ↑	Parallel ↑	Parallel-Multiple ↑	Irrelevance ↑
DeepSeek-V4-Flash	Raw	**96.0**	95.0	92.5	88.5	41.7
	Reason-then-Act	52.0	35.0	51.0	18.0	75.4
	RA-PF	95.8	**95.5**	**93.0**	**89.5**	76.3
	ToolNeed	95.8	95.0	92.5	87.5	**90.0**
Qwen3.7-Flash	Raw	90.0	84.5	87.0	87.5	91.3
	Reason-then-Act	77.3	83.0	70.5	77.0	90.8
	RA-PF	**90.5**	**85.5**	**91.5**	**89.5**	91.3
	ToolNeed	90.0	84.5	87.0	87.0	**95.0**
Qwen3.5-9B	Raw	92.3	**93.5**	87.0	**88.5**	83.8
	Reason-then-Act	69.8	70.0	69.5	65.0	84.6
	RA-PF	**92.5**	**93.5**	**91.0**	**88.5**	85.8
	ToolNeed	92.0	**93.5**	87.0	87.5	**90.4**

5 Analysis

5.1 Validity and Behavioral Necessity

We examine the relation between successful tool validity and behavioral necessity through a targeted GSM8K construct analysis. We condition on tool calls naturally proposed by each target that are relevant, valid, executable, and gold-correct after completing the tool-assisted path. We then remove tool availability with the conversation, target model, and decoding regime fixed, and evaluate the frozen matched no-tool behavior with the same gold checker.

Table 3: GSM8K construct analysis conditioned on naturally proposed, relevant, valid, executable, and gold-correct tool-assisted paths. The rate is the conditional prevalence of a gold-correct matched no-tool behavior (V1N0), with exact 95% confidence intervals.

Target	Successful valid tool paths	Matched no-tool correct	Conditional rate
Qwen3.5-9B	1,141	1,099	96.32% [95.06, 97.33]
DeepSeek-V4-Flash	526	523	99.43% [98.34, 99.88]

Among 1,141 qualifying Qwen3.5-9B tool paths, 1,099 matched no-tool behaviors are also gold-correct (96.32%, exact 95% CI $[95.06, 97.33]$). A parallel construct analysis with DeepSeek-V4-Flash finds 523 of 526 (99.43%, $[98.34, 99.88]$). These are successful tool actions: the proposed calls are relevant, valid, executable, and lead to correct task solutions. Their high conditional V1N0 prevalence shows that successful tool validity itself provides weak evidence that external action is behaviorally necessary. Necessity is relative to the target model and decoding setting because it is defined against the matched behavior available under that configuration.

GSM8K provides gold-checked construct-level evidence for the distinction between validity and behavioral necessity. The When2Tool Retriever diagnostic next evaluates whether the framework acts on that distinction under Valid=1 qualification.

5.2 Validity-Conditioned Necessity Decisions

We evaluate Qwen3.5-9B on the When2Tool Retriever medium+hard targeted diagnostic (Sun et al., 2026). All 59 evaluated cases satisfy Valid=1: the target naturally proposes a relevant, valid, executable Retriever action whose completed tool-assisted path is gold-correct. Removing tool availability under the same conversation, target, and decoding regime partitions these paths into 42 V1N0 cases with gold-correct matched no-tool behavior and 17 V1N1 cases whose matched no-tool behavior is gold-incorrect.

Table 4: When2Tool Retriever medium+hard targeted diagnostic. All 59 evaluated cases satisfy Valid=1; V1N0 requires DIRECT, V1N1 requires TOOL, and balanced accuracy macro-averages the two class-specific accuracies.

Method	V1N0 DIRECT ↑	V1N1 TOOL ↑	Bal. Acc. ↑	Final correct ↑
ToolNeed-Base	0/42	17/17	50.00%	59/59
ToolNeed-Helper	0/42	17/17	50.00%	59/59
RA-PF	0/42	17/17	50.00%	59/59
ToolNeed	**10/42**	**17/17**	**61.90%**	**59/59**

ToolNeed is a selective necessity-sensitive intervention framework. ToolNeed-Base, ToolNeed-Helper (Helper-only), and RA-PF retain every external action. Of the 42 V1N0 cases, behavioral verification is triggered for 10, and all 10 are converted to DIRECT; the remaining 32 follow routing bypasses. All 17 V1N1 cases also follow routing bypasses and are retained as TOOL before arbitration. All ten interventions are correct, yielding 100% intervention precision, zero regressions, and 59/59 final task correctness. ToolNeed improves over ToolNeed-Base, ToolNeed-Helper, and RA-PF with 10 corrections, zero regressions, and 49 ties in each paired comparison (exact McNemar $p = .001953$). The balanced-accuracy gain is 11.90 points, with paired bootstrap 95% intervals of $[+5.95, +17.86]$ against ToolNeed-Base and $[+5.95, +19.05]$ against RA-PF. These results show selective identification of valid-but-unnecessary actions and preservation of every valid-and-necessary action in the diagnostic.

GSM8K establishes the empirical validity–necessity gap at construct level. The When2Tool Retriever medium+hard targeted diagnostic tests alignment between the method and construct under Valid=1, and the matched ToolNeed-Base comparison quantifies ToolNeed's behavioral path. SABEval and BFCL provide large-scale action-level evidence.

5.3 Contribution of ToolNeed's Behavioral Path

We quantify the joint incremental effect of ToolNeed's behavioral path with a matched ToolNeed-Base (NoBehavior) ablation. ToolNeed-Base and ToolNeed share the same routing, grounding, Stage A, Stage B, and tool-enabled behavior before arbitration, A_{base}. ToolNeed-Base returns A_{base} directly; ToolNeed additionally generates the matched no-tool behavior $A_{\emptyset}$ and performs contrastive arbitration. This comparison isolates the joint increment from matched no-tool generation and arbitration.

Across the five matched regimes in Table 5, ToolNeed improves over ToolNeed-Base. On BFCL Live, the comparison yields 38 corrections versus 6 regressions for Qwen3.7 and 42 versus 8 for Qwen3.5, increasing pooled split-level accuracy by 1.56 and 1.66 points. Non-Live yields 5/1 and

Table 5: Matched ToolNeed-Base → ToolNeed behavioral path ablation. C/R denotes corrections/regressions. BFCL $\Delta > 0$ favors ToolNeed; SABEval $\Delta < 0$ favors ToolNeed because lower TIR is better. BFCL pooled task accuracy is a split-level diagnostic.

Target	Benchmark	C/R	Δ (pp)	95% CI	p
Qwen3.7	BFCL Live	38/6	$+1.56$ Acc.	$[+0.93, +2.19]$	9.43×10^{-7}
Qwen3.5	BFCL Live	42/8	$+1.66$ Acc.	$[+0.98, +2.34]$	1.16×10^{-6}
Qwen3.7	BFCL Non-Live	5/1	$+0.32$ Acc.	$[0, +0.73]$	.2188
Qwen3.5	BFCL Non-Live	14/2	$+0.97$ Acc.	$[+0.40, +1.61]$	.00418
Qwen3.5	SABEval	3,837/0	-37.43 TIR	$[-44.04, -30.79]$	9.9999×10^{-6}

14/2 corrections/regressions, respectively. On Qwen3.5 SABEval, ToolNeed reduces aggregate TIR from 55.42% to 17.99%, with 3,837 corrections and zero regressions; template-clustered inference gives a 95% CI of $[-44.04, -30.79]$ points and $p = 9.9999 \times 10^{-6}$.

The BFCL gains concentrate in Irrelevance. Live Irrelevance rises by 4.61 and 5.10 points for Qwen3.7 and Qwen3.5. Simple shifts by -2.29 and -2.75 points, and Multiple shifts by -0.10 and -0.20 points. Non-Live Irrelevance rises by 2.08 and 5.83 points. Simple, Multiple, and Parallel are unchanged for Qwen3.7 and shift by -0.25, 0, and 0 for Qwen3.5; Parallel-Multiple shifts by -0.50 points for both targets. Appendix D provides the full ablation definition and statistical details.

5.4 Information Value of Matched No-Tool Behavior

ToolNeed and Masked-NoTool share routing and triggers, Stage A and Stage B, A_{base}, the arbiter model and instruction, DIRECT tie-breaking, decoding, and return semantics. Masked-NoTool replaces the realized $A_\emptyset$ shown to the arbiter with "[WITHHELD]"; the paired comparison measures the incremental decision information carried by matched no-tool behavior. ToolNeed improves BFCL Irrelevance by 4.13 and 2.91 points for Qwen3.7 and Qwen3.5 Live, and by 2.08 points for each target on Non-Live. Realized no-tool content provides additional decision-relevant information for Irrelevance. The Live preservation cost is concentrated mainly in Simple, with shifts of -2.29 and -2.75 points; Multiple changes are -0.10 and -0.20 points. Non-Live tool-required aggregates remain near-exact at -0.10 and -0.20 points. Token budgets vary; Appendix E gives the matched details.

5.5 Semantic Adequacy of Direct Behavior

As supporting response-level semantic evidence, we audit a frozen stratified sample of 40 SABEval Raw-TOOL→ToolNeed-DIRECT conversions. A blinded DeepSeek-V4-Flash evaluator rates 34/40 sampled responses (85.0%; exact 95% CI $[70.2\%, 94.3\%]$) as adequately satisfying the user request without a material factual error. The sample rates are 80.0% for Qwen3.5 and 90.0% for Qwen3.7; the six non-passing cases consist of three factual errors and three task-critical omissions.

6 Conclusion

Tool augmentation introduces an external action selection problem: a valid tool action can remain unnecessary when the model's no-tool behavior already satisfies the request. ToolNeed selectively verifies proposed tool actions by eliciting matched no-tool behavior and applying contrastive arbitration under candidate-blind routing. Across SABEval and complementary BFCL regimes, this training-free, black-box-compatible method improves unnecessary-action handling. BFCL Irrelevance gains generalize across all three backbones, and Non-Live evaluations show near-exact preservation of legitimate tool-required behavior. Calibrated tool use begins with knowing when external capability is truly warranted.

Reproducibility Statement. An anonymized implementation is provided in the supplementary material, including the ToolNeed pipeline, exact ToolNeed prompts, method configurations, benchmark adapters, and scripts for reproducing the reported analyses and statistical tests. A maintained public repository will be released after the double-blind review period.

AI USE STATEMENT

Generative AI tools assisted with data organization and manuscript editing. The authors independently developed the scientific ideas, method design, experimental protocols, analysis, and conclusions, and verified all reported results. The authors take full responsibility for the work.

REFERENCES

Chen Chen, Xinlong Hao, Weiwen Liu, Xu Huang, Xingshan Zeng, Shuai Yu, Dexun Li, Yuefeng Huang, Xiangcheng Liu, Wang Xinzhi, and Wu Liu. ACEBench: A comprehensive evaluation of LLM tool usage. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 12970–12998. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.697. URL https://aclanthology.org/2025.findings-emnlp.697/.

Yize Cheng, Chenrui Fan, Mahdi JafariRaviz, Keivan Rezaei, and Soheil Feiz. Model-adaptive tool necessity reveals the knowing-doing gap in LLM tool use. *arXiv preprint arXiv:2605.14038*, 2026.

Yanze Jiang, Mingxuan Li, Yuhao Wang, Shengfang Zhai, and Jiaheng Zhang. Don't solve, just compare: Tiny advisors for runtime intervention in LLM agents. *arXiv preprint arXiv:2608.21027*, 2026. URL https://arxiv.org/abs/2608.21027.

Wenjun Li, Dexun Li, Kuicai Dong, Cong Zhang, Hao Zhang, Weiwen Liu, Yasheng Wang, Ruiming Tang, and Yong Liu. Adaptive tool use in large language models with meta-cognition trigger. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13346–13370. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.655. URL https://aclanthology.org/2025.acl-long.655/.

Yilong Liu, Xixun Lin, Pengfei Cao, Ge Zhang, Fang Fang, and Yanan Cao. Do LLMs know tool irrelevance? demystifying structural alignment bias in tool invocations. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 31934–31958. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1473. URL https://aclanthology.org/2026.acl-long.1473/.

Dheeraj Mekala, Jason E Weston, Jack Lanchantin, Roberta Raileanu, Maria Lomeli, Jingbo Shang, and Jane Dwivedi-Yu. TOOLVERIFIER: Generalization to new tools via self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 5026–5041. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-emnlp.289. URL https://aclanthology.org/2024.findings-emnlp.289/.

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48371–48392. PMLR, 2025. URL https://proceedings.mlr.press/v267/patil25a.html.

Cheng Qian, Emre Can Acikgoz, Hongru Wang, Xiusi Chen, Avirup Sil, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. SMART: Self-aware agent for tool overuse mitigation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 4604–4621. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.239. URL https://aclanthology.org/2025.findings-acl.239/.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein,

Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Hayley Ross, Ameya Sunil Mahabaleshwarkar, and Yoshi Suhara. When2Call: When (not) to call tools. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3391–3409. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.174. URL https://aclanthology.org/2025.naacl-long.174/.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-2997. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Chung-En Sun, Linbo Liu, Ge Yan, Zimo Wang, and Tsui-Wei Weng. LLM agents already know when to call tools—even without reasoning. *arXiv preprint arXiv:2605.09252*, 2026. URL https://arxiv.org/abs/2605.09252.

Anh Ta, Junjie Zhu, and Shahin Shayandeh. Reinforced agent: Inference-time feedback for tool-calling agents. In *Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM)*, pp. 136–147. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.gem-main.13. URL https://aclanthology.org/2026.gem-main.13/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

A Framework Procedure

ToolNeed applies selective verification to a provisional external action and returns either the tool-enabled behavior or its matched no-tool behavior.

1. The target model produces a provisional response from the conversation and available tools; proposed external actions enter selective verification.
2. Candidate-blind routing coordinates activation of requirement-grounded candidate verification and behavioral verification according to the target-specific routing rule.
3. When requirement verification is activated, the framework grounds task requirements before inspecting the proposal and produces the current tool-enabled behavior without tool reselection.
4. When behavioral verification is activated, the framework obtains the matched no-tool behavior $A_\emptyset$ by holding the conversation, target model, and decoding convention fixed as tool availability is removed, then contrastively arbitrates between the Raw tool candidate A_{tool} and $A_\emptyset$.
5. A TOOL decision returns A_{base}, and a DIRECT decision returns $A_\emptyset$; when behavioral verification is inactive, the framework preserves the current or base tool-enabled behavior.

A.1 Operational Details

Table 6 records the frozen module interfaces. Candidate blindness applies to the Target Router, helper Router, and Stage A: each observes the conversation and available tool schemas without the concrete candidate-selected function or arguments. Stage B first exposes the candidate, after deterministic grounding and filtering have fixed the admitted requirement set.

For Qwen targets, `APPLY_HYBRID` from the Target Router activates both the requirement path and the behavioral path. A Target Router decision of `PRESERVE_RAW` skips Stage A and Stage B,

Table 6: Operational interfaces in the frozen ToolNeed implementation.

Module	Executor	Trigger	Visible inputs	Hidden inputs	Output
Target Router	Target model	Provisional tool action	Conversation; tool schemas	Concrete function and arguments	`APPLY_HYBRID` or `PRESERVE_RAW`
Stage A requirement grounding	Target model	Target Router: `APPLY_HYBRID`	Conversation; tool schemas	Concrete function and arguments	Proposed request-grounded requirements
Deterministic grounding/filter	Deterministic rules	Stage A completion	Stage A requirements; conversation; schemas	Concrete function and arguments	Admitted requirement set
Stage B	Target model	Admitted requirement set	Requirements; conversation; schemas; Raw tool candidate	$A_\emptyset$	Candidate decision and A_{base}
Helper Router	DeepSeek for Qwen targets	Qwen Target Router: `PRESERVE_RAW`	Conversation; tool schemas	Concrete function and arguments	Behavioral trigger
Matched no-tool generation	Same target model	Behavioral trigger active	Same conversation and decoding; empty tool set	Raw tool candidate and A_{base}	$A_\emptyset$
Arbitration	DeepSeek helper for Qwen; DeepSeek target for DeepSeek	Completed $A_\emptyset$ generation	Conversation; schemas; Raw tool candidate; $A_\emptyset$	A_{base}	TOOL or DIRECT

after which the DeepSeek helper Router determines whether to activate the behavioral path. For the DeepSeek target, the Target Router itself supplies the behavioral trigger, avoiding a duplicate same-model call with the same Router prompt; `APPLY_HYBRID` runs Stage A and Stage B to obtain A_{base} and also activates generation of $A_\emptyset$ followed by arbitration.

Here A_{base} is the base behavior after requirement verification, and $A_\emptyset$ is generated by the same target from the same conversation and decoding configuration with tools removed. Arbitration receives the Raw tool candidate and $A_\emptyset$. A TOOL decision returns A_{base}; a DIRECT decision returns $A_\emptyset$. The procedure preserves the selected behavior without tool reselection.

Exact prompts. The placeholders below substitute the instance-specific conversation, schemas, requirements, and candidate without changing the instruction text. The helper Router reuses the Router prompt verbatim.

Router.

> Before seeing any proposed tool call, determine whether the currently available tool set contains a DIRECT, goal-grounded capability needed for the user's request.
>
> Return PRESERVE_RAW when at least one available tool directly:
>
> – performs an explicitly requested operation,
> – produces an explicitly requested output,
> – completes an explicitly requested suboperation,
> – resolves a concrete missing requirement already present in the interaction, or
> – establishes a clearly necessary operational prerequisite.
>
> Do NOT treat a tool as directly supportive merely because it:
>
> – is about the same topic,
> – provides related or correlated information,
> – could help infer the answer,
> – supplies optional context,
> – or might be useful in a hypothetical future step.
>
> If the tools are only topically or indirectly related to the request, return APPLY_HYBRID.
>
> For multi-tool tasks, a tool set is supported when its tools directly cover explicit requested suboperations; no single tool needs to solve the whole task.
>
> For multi-turn tasks, a tool that directly resolves a concrete current missing requirement or prerequisite counts as legitimate support.
>
> If you genuinely cannot determine whether support is direct or merely topical, return PRESERVE_RAW.
>
> Return JSON only: `{"route": "APPLY_HYBRID" or "PRESERVE_RAW"}`

Current conversation/history JSON: ⟨*conversation*⟩
Current available function schemas JSON: ⟨*schemas*⟩

Stage A.

Before seeing any proposed action, identify only the requirements that are already justified by the current user goal and interaction state.
Do not invent hypothetical future steps.
List:
1. direct goal requirements;
2. currently known missing requirements;
3. necessary prerequisites that must be satisfied to continue;
4. explicitly required suboperations.
A requirement should be included only if it is justified by the current request, conversation history, observed state, or available tool schemas.
Do not include merely useful context, optional information, or speculative future needs.
Return only valid JSON: {"direct_goal_requirements": ["..."], "known_missing_requirements": ["..."], "necessary_prerequisites": ["..."], "required_suboperations": ["..."]}
Current conversation/history JSON: ⟨*conversation*⟩
Current available function schemas JSON: ⟨*schemas*⟩

Stage B.

You are given a frozen requirement set created before the proposed candidate was shown.
Decide whether the candidate satisfies, resolves, establishes, or completes at least one requirement that was already present in the frozen requirement set.
The candidate is VALID only if its documented operation/output/state effect directly satisfies, resolves, establishes, or completes at least one requirement that was already present in the frozen requirement set.
Do not create a new requirement after seeing the candidate.
Do not approve the candidate because it:
– is in the same domain;
– has compatible parameters;
– might be useful later;
– provides generally related information;
– could hypothetically support a future step.
Allowed matched_requirement_type values:
– EXPLICIT_GOAL
– KNOWN_MISSING
– HARD_PRECONDITION
– REQUIRED_SUBOPERATION
– NONE
Return only valid JSON: {"decision": "VALID" or "INVALID", "matched_requirement_type": "EXPLICIT_GOAL" | "KNOWN_MISSING" | "HARD_PRECONDITION" | "REQUIRED_SUBOPERATION" | "NONE", "matched_requirement": "..."}
Frozen requirement set JSON: ⟨*requirements*⟩
Minimal current conversation/history JSON: ⟨*conversation*⟩
Current available function schemas JSON: ⟨*schemas*⟩
Candidate response/action JSON: ⟨*Raw candidate*⟩

Arbitration.

You are a contrastive action validator for tool-calling.
Evaluate whether each action actually completes the user's current requested operation or output:
– A_TOOL may be correct only if the tool operation is necessary and directly grounded in the current user goal.

– A_DIRECT may be correct if it answers, refuses, asks clarification, or otherwise handles the request sufficiently without the removed tool.

If both are plausible, choose TOOL only when the user goal clearly requires external/tool capability that A_DIRECT cannot provide. Otherwise choose DIRECT.
Return JSON only: {"choice":"TOOL"} or {"choice":"DIRECT"}

Current conversation/history JSON: ⟨*conversation*⟩

Current available function schemas JSON: ⟨*schemas*⟩

A_TOOL candidate JSON: ⟨*Raw candidate*⟩

A_DIRECT no-tool alternative text: $\langle A_\emptyset \rangle$

Table 7: Frozen decoding and generation configuration. All modules use temperature 0 with thinking disabled.

Generation	Maximum output tokens
Raw	512
Router	128
Stage A	512
Stage B	256
$A_\emptyset$	512
Arbitration	128

The target executors are Qwen3.7-Flash, Qwen3.5-9B, and DeepSeek-V4-Flash. Each target executes its Target Router, Stage A, Stage B, and matched no-tool generation. DeepSeek-V4-Flash executes the conditional helper Router and contrastive arbitration for Qwen targets. For the DeepSeek target, it also executes arbitration and bypasses the duplicate same-model helper Router. Qwen3.5-9B uses local deterministic generation with `do_sample=False`; all target and verification calls use temperature 0 with thinking disabled.

A malformed Router response maps to `PRESERVE_RAW`, a Stage B parse failure retains Raw, and malformed arbitration maps to TOOL, retaining A_{base}. Qwen materialization preserves A_{base} while direct generation is incomplete; DeepSeek propagates an unrecovered generation failure. Arbitration proceeds after successful $A_\emptyset$ generation, and completed reported runs contain completed $A_\emptyset$ generations and arbitration outcomes for every triggered case.

B Additional Experimental Details

The target models are DeepSeek-V4-Flash, Qwen3.7-Flash, and Qwen3.5-9B. All evaluations use deterministic temperature-0 decoding with thinking disabled. Following SABEval's seed-42 template-level procedure, we sort its 101 template identifiers and instantiate a 40/20/41 train/validation/test split. The frozen 41-template test subset contains 10,250 tasks (2,050 per D0–D4), and development and test are template-disjoint. For ACEBench-English Single and Parallel, 20 of 100 examples per category are frozen for development and the remaining 80 per category form the evaluation controls. BFCL uses the shared 2,051-example held-out Live subset and Non-Live split described in Section 4. ACEBench and BFCL are scored with their category evaluators. The same task IDs, comparison protocol, decoding convention, and evaluation procedures are used across the three target backbones.

RA-PF baseline. RA-PF is a faithful benchmark-compatible adaptation of Reinforced Agent's Progressive Feedback mechanism. The frozen Raw response initializes the current provisional response. At each pass, the reviewer observes the conversation, available tools, and current response. A review with `error=false` accepts the response; with `error=true`, the feedback is injected as a system message and the same target backbone generates a revision using the original tool interface. The state machine permits up to five reviewer decisions and four intervening revisions, with early stopping upon acceptance. If the fifth decision rejects, RA-PF returns the last provisional response.

Table 8: Roles in the benchmark-compatible RA-PF adaptation.

Role	Executor
Target	Evaluated target backbone
Reviewer	DeepSeek-V4-Flash
Reviser	Evaluated target backbone with the original tool interface

C Paired Statistical Analysis

SABEval inference uses the 41 held-out test templates as clusters. Confidence intervals come from 10,000 template-cluster bootstrap replicates, and paired p values come from 100,000 template-level sign-flip permutations. Negative SABEval Δ favors ToolNeed because lower TIR is better. BFCL paired binary outcomes use exact McNemar tests and 10,000-sample task-level paired bootstrap confidence intervals. Holm correction follows the six prespecified families in Section 4; mechanism analyses retain their reported inferential protocols outside these families.

Table 9: SABEval template-clustered comparisons of ToolNeed against each baseline. Confidence intervals use 10,000 template-cluster bootstrap replicates; cluster p values use 100,000 paired sign-flip permutations over the 41 test templates. Differences are TIR percentage points, with negative values favoring ToolNeed.

Backbone	Baseline	Δ TIR	Cluster 95% CI	Raw p	Holm p
DeepSeek	Raw	-83.09	$[-88.30, -77.27]$	1.00×10^{-5}	3.00×10^{-5}
Qwen3.7	Raw	-34.44	$[-42.14, -26.81]$	1.00×10^{-5}	3.00×10^{-5}
Qwen3.5	Raw	-44.09	$[-51.84, -36.09]$	1.00×10^{-5}	3.00×10^{-5}
DeepSeek	RTA	-44.83	$[-52.37, -37.35]$	1.00×10^{-5}	3.00×10^{-5}
Qwen3.7	RTA	-2.10	$[-5.14, +0.83]$	.18763	.18763
Qwen3.5	RTA	-14.82	$[-20.50, -9.21]$	2.00×10^{-5}	4.00×10^{-5}
DeepSeek	RA-PF	-5.54	$[-8.55, -3.00]$	3.00×10^{-5}	.00012
Qwen3.7	RA-PF	-1.46	$[-3.52, +0.71]$	.18950	.2039
Qwen3.5	RA-PF	$+3.25$	$[-0.46, +7.03]$	.10195	.2039

All nine rows use 41-template cluster inference. The RA-PF rows belong to the six-test headline family together with the three BFCL Live Irrelevance comparisons; Raw and RTA each form separate three-backbone families.

Table 10: BFCL paired ToolNeed–RA-PF Irrelevance comparisons. C/R denotes corrections/regressions, differences are percentage points, and exact McNemar p values are unadjusted.

Benchmark / regime	Backbone	Metric	Δ	C/R	95% CI	Raw p	Holm p
BFCL Live	DeepSeek	Irrelevance	$+14.08$	138/22	$[+11.17, +16.87]$	9.840×10^{-22}	5.904×10^{-21}
BFCL Live	Qwen3.7	Irrelevance	$+6.31$	61/9	$[+4.49, +8.25]$	1.284×10^{-10}	6.42×10^{-10}
BFCL Live	Qwen3.5	Irrelevance	$+4.13$	50/16	$[+2.18, +6.07]$	3.328×10^{-5}	.00012
BFCL Non-Live	DeepSeek	Irrelevance	$+13.75$	35/2	$[+9.17, +18.33]$	1.024×10^{-8}	3.072×10^{-8}
BFCL Non-Live	Qwen3.7	Irrelevance	$+3.75$	9/0	$[+1.67, +6.25]$	.00390625	.0078125
BFCL Non-Live	Qwen3.5	Irrelevance	$+4.58$	15/4	$[+1.25, +8.33]$	.0192108	.0192108

The Live rows share the six-test headline family with the three SABEval RA-PF comparisons. The Non-Live rows form a separate three-test family.

Five of the six adjusted Live comparisons are significant; Qwen3.5 Simple is the exception.

ToolNeed exactly matches Raw in seven of twelve Non-Live tool-required comparisons; the remaining five differ by at most 1.0 point. No significant degradation is detected after Holm correction.

Table 11: BFCL Live tool-required ToolNeed–Raw paired comparisons. The six Simple/Multiple tests form one Holm family. Differences are percentage points.

Backbone	Category	Δ	C/R	95% CI	Raw p	Holm p
DeepSeek	Simple	-3.67	0/8	$[-6.42, -1.38]$	.0078125	.03125
DeepSeek	Multiple	-2.68	0/27	$[-3.77, -1.78]$	1.490×10^{-8}	8.94×10^{-8}
Qwen3.7	Simple	-3.67	0/8	$[-6.42, -1.38]$	.0078125	.03125
Qwen3.7	Multiple	-0.89	0/9	$[-1.49, -0.40]$	.00390625	.01953125
Qwen3.5	Simple	-0.92	1/3	$[-2.75, +0.92]$	.625	.625
Qwen3.5	Multiple	-0.99	2/12	$[-1.78, -0.30]$	.0129395	.03125

Table 12: BFCL Non-Live tool-required ToolNeed–Raw preservation family. All twelve Holm-adjusted p values equal 1. Differences are percentage points.

Backbone	Category	Δ	Holm p
DeepSeek	Simple	-0.25	1
DeepSeek	Multiple	0	1
DeepSeek	Parallel	0	1
DeepSeek	Parallel-Multiple	-1.00	1
Qwen3.7	Simple	0	1
Qwen3.7	Multiple	0	1
Qwen3.7	Parallel	0	1
Qwen3.7	Parallel-Multiple	-0.50	1
Qwen3.5	Simple	-0.25	1
Qwen3.5	Multiple	0	1
Qwen3.5	Parallel	0	1
Qwen3.5	Parallel-Multiple	-1.00	1

D Matched Behavioral Path Ablation

ToolNeed-Base (NoBehavior) and ToolNeed share identical routing, grounding, Stage A, Stage B, and the same base behavior before arbitration, A_{base}. ToolNeed-Base returns A_{base} directly. ToolNeed additionally generates the matched no-tool behavior $A_{\emptyset}$ and performs contrastive arbitration. The ToolNeed-Base $\rightarrow$ ToolNeed comparison isolates the joint incremental contribution of matched no-tool generation and arbitration.

ToolNeed-Helper is a descriptive alternative variant with a trigger set and Stage B provider and path distinct from ToolNeed. It appears in the validity-conditioned When2Tool comparison; ToolNeed-Base (NoBehavior) supplies the matched behavioral path ablation.

Table 13: Matched ToolNeed-Base $\rightarrow$ ToolNeed transitions. C/R denotes corrections/regressions. BFCL differences are pooled split-level accuracy diagnostics; SABEval reports final-action TIR.

Target	Benchmark	C/R	Δ (pp)	p
Qwen3.7	BFCL Live	38/6	$+1.56$ Acc.	9.43×10^{-7}
Qwen3.5	BFCL Live	42/8	$+1.66$ Acc.	1.16×10^{-6}
Qwen3.7	BFCL Non-Live	5/1	$+0.32$ Acc.	.2188
Qwen3.5	BFCL Non-Live	14/2	$+0.97$ Acc.	.00418
Qwen3.5	SABEval	3,837/0	-37.43 TIR	9.9999×10^{-6}

The BFCL 95% confidence intervals are $[+0.93, +2.19]$ for Qwen3.7 Live, $[+0.98, +2.34]$ for Qwen3.5 Live, $[0, +0.73]$ for Qwen3.7 Non-Live, and $[+0.40, +1.61]$ for Qwen3.5 Non-Live. For Qwen3.5 SABEval, ToolNeed-Base records 5,681/10,250 tool actions and ToolNeed records 1,844/10,250. A 10,000-replicate bootstrap over the 41 test templates gives $[-44.04, -30.79]$ points, and a 100,000-sample template-level sign-flip test with plus-one correction gives $p = 9.9999 \times 10^{-6}$ (seed 20260903). The paired outcomes are 3,837 Tool-to-Direct transitions, zero Direct-to-Tool transitions, and 6,413 ties.

Table 14: BFCL category-wise ToolNeed-Base → ToolNeed differences in percentage points. Positive differences favor ToolNeed.

Target	Regime	Simple	Multiple	Irrelevance
Qwen3.7	Live	−2.29	−0.10	+4.61
Qwen3.5	Live	−2.75	−0.20	+5.10

Target	Regime	Simple	Multiple	Parallel	Parallel-Multiple	Irrelevance
Qwen3.7	Non-Live	0	0	0	−0.50	+2.08
Qwen3.5	Non-Live	−0.25	0	0	−0.50	+5.83

Table 15: Qwen3.5 SABEval final-action TIR (%) for the matched behavioral path ablation. Lower is better.

Variant	Total	D0	D1	D2	D3	D4
ToolNeed-Base	55.42	44.59	51.22	54.98	60.29	66.05
ToolNeed	17.99	11.12	16.20	18.78	20.05	23.80

E Masked-NoTool Information Ablation

ToolNeed and Masked-NoTool share the same Raw tool candidate, routing and trigger decisions, Stage A and Stage B computations, A_{base}, arbiter model, arbitration instruction, DIRECT tie-breaking rule, decoding, and final return semantics. ToolNeed presents the realized matched no-tool behavior $A_{\emptyset}$ in the A_{DIRECT} section. Masked-NoTool presents the fixed string "[WITHHELD]" in that section. A DIRECT decision returns the same frozen $A_{\emptyset}$ in both variants. The comparison measures the incremental decision information carried by observing the realized matched no-tool behavior. Token budgets differ between variants.

Table 16: Matched paired ToolNeed minus Masked-NoTool comparisons. C/R/T denotes corrections, regressions, and ties from the perspective of ToolNeed. Live tool-required categories are reported separately; Non-Live tool-required rows pool the applicable function calling categories.

Target	Regime	Scope	Δ Acc.	C/R/T	95% CI	McNemar p
Qwen3.7	Live	Simple	−2.29	0/5/213	$[-4.59, -0.46]$	.0625
Qwen3.7	Live	Multiple	−0.10	0/1/1008	$[-0.30, 0]$	1
Qwen3.7	Live	Irrelevance	+4.13	34/0/790	$[+2.79, +5.58]$	1.16×10^{-10}
Qwen3.5	Live	Simple	−2.75	0/6/212	$[-5.05, -0.92]$	.0313
Qwen3.5	Live	Multiple	−0.20	0/2/1007	$[-0.50, 0]$	.500
Qwen3.5	Live	Irrelevance	+2.91	26/2/796	$[+1.70, +4.13]$	3.03×10^{-6}
Qwen3.7	Non-Live	Irrelevance	+2.08	5/0/235	$[+0.42, +4.17]$	.0625
Qwen3.7	Non-Live	Tool-required	−0.10	0/1/999	$[-0.30, 0]$	1.0
Qwen3.5	Non-Live	Irrelevance	+2.08	6/1/233	$[0, +4.17]$	.125
Qwen3.5	Non-Live	Tool-required	−0.20	0/2/998	$[-0.50, 0]$	.5

The Live matched scores for Qwen3.7 are Base/Masked/ToolNeed $= 55.50/55.50/53.21$ on Simple, $66.40/66.40/66.30$ on Multiple, and $81.07/81.55/85.68$ on Irrelevance. For Qwen3.5 they are $58.72/58.72/55.96$, $59.96/59.96/59.76$, and $77.31/79.49/82.40$, respectively. Mechanism analyses use internally matched frozen prediction sets to estimate component effects within each controlled comparison. The Live preservation cost is concentrated mainly in Simple, Multiple changes are small, and Non-Live tool-required preservation remains near-exact.

F Inference Accounting

On-policy accounting includes successful model outputs that are executed by the frozen ToolNeed control flow and consumed by the final path. Historical eager or precomputed outputs, helper-only controls, untriggered cached outputs, and legacy duplicates are excluded. Provider failures without a returned model output are recorded separately from successful-output calls. Table 17 reports ToolNeed's absolute activity, its paired increment over RA-PF, and the realized behavioral-trigger rate.

Table 17: Frozen on-policy inference accounting. ToolNeed calls and total input-plus-output tokens are per task; deltas are ToolNeed minus RA-PF; trigger is the realized behavioral-verification rate. A dash marks unavailable RA-PF token metadata.

Target	Benchmark	TN calls	TN tokens	Δ calls	Δ tokens	Trigger
DeepSeek	SABEval	5.4211	3323.17	$+1.6464$	$+756.24$	87.74%
DeepSeek	BFCL Live	2.8113	2894.23	$+0.2526$	-28.33	20.28%
DeepSeek	BFCL Non-Live	2.4129	1410.72	$+0.0210$	–	10.32%
Qwen3.7	SABEval	3.0788	1897.05	$+0.1614$	$+102.43$	40.9%
Qwen3.7	BFCL Live	3.0385	3251.55	$+0.7860$	$+693.88$	14.3%
Qwen3.7	BFCL Non-Live	2.8355	1596.52	$+0.6532$	$+201.52$	1.9%
Qwen3.5	SABEval	3.5012	2028.31	-0.0475	-257.05	58.9%
Qwen3.5	BFCL Live	2.8805	2996.04	$+0.4846$	$+297.65$	10.1%
Qwen3.5	BFCL Non-Live	2.8879	1623.27	$+0.6395$	$+171.82$	2.7%

Inference overhead is backbone- and regime-dependent, with realized activity partly governed by behavioral-trigger frequency. Calls and tokens report model-inference activity; monetary cost is outside this accounting. DeepSeek BFCL Non-Live has complete ToolNeed accounting, and 74 of 1,240 RA-PF tasks lack token metadata. Its token delta is omitted without interpolation. Available latency records do not support an aggregate latency estimate.

G GSM8K Construct Analysis

The targeted construct analysis scans naturally proposed tool actions and applies a fixed qualification sequence. Table 18 reports the complete attrition. A qualifying tool path is relevant to the request, has valid arguments, executes successfully, and produces a gold-correct answer after tool use. For each qualifying instance, the frozen matched no-tool response is generated from the same conversation, target model, and decoding regime, with tool availability removed as the controlled intervention. The gold checker is used for qualification and outcome evaluation; its verdict and the reference answer are never exposed to a verification method.

Table 18: GSM8K targeted construct-analysis attrition. The final columns partition successful valid tool paths by matched no-tool correctness.

Target	Scanned	Raw Tool	Valid	Exec.	Relevant	Qualifying	V1N0	V1N1
Qwen3.5-9B	1,319	1,276	1,276	1,236	1,203	1,141	1,099	42
DeepSeek-V4-Flash	600	597	594	567	539	526	523	3

V1N0 denotes a relevant, valid, executable proposal whose completed tool-assisted path is gold-correct and whose frozen matched no-tool response is also gold-correct. V1N1 satisfies the same successful-tool-path criteria and has a gold-incorrect matched no-tool response. The conditional V1N0 prevalence is $1{,}099/1{,}141 = 96.32\%$ for Qwen3.5-9B and $523/526 = 99.43\%$ for DeepSeek-V4-Flash. Exact Clopper–Pearson 95% confidence intervals are $[95.06, 97.33]$ and $[98.34, 99.88]$, respectively. These rates quantify the conditional prevalence of V1N0 among successful provisional tool calls under the specified target, decoding configuration, and tool protocol. DeepSeek-V4-Flash supplies cross-backbone construct evidence under the same target-relative definition.

H When2Tool Retriever Targeted Diagnostic

The When2Tool Retriever medium+hard targeted diagnostic evaluates 100 official RetrieverEnv candidate tasks at medium and hard difficulty with Qwen3.5-9B, temperature 0, thinking disabled, and the official RetrieverEnv gold-answer evaluator.

Valid=1 requires that the raw target proposes the Retriever tool, the proposal is relevant, valid, and executable, and the completed tool-assisted path is gold-correct. For each qualifying path, the frozen matched no-tool behavior uses the same target, conversation, and decoding regime with tool availability removed. Gold-correct matched no-tool behavior defines V1N0; gold-incorrect

Table 19: Qualification attrition for the When2Tool Retriever medium+hard targeted diagnostic. Successful valid paths satisfy the Valid=1 qualification.

Difficulty	Candidate	Raw Tool	Valid/Exec.	Tool-path correct	V1N0	V1N1
Medium	50	48	48	47	42	5
Hard	50	50	50	12	0	12
Combined	100	98	98	59	42	17

matched no-tool behavior defines V1N1. The gold verdict determines diagnostic labels and final task correctness.

ToolNeed-Base, ToolNeed-Helper, and RA-PF select TOOL for all 59 qualifying paths, producing 0/42 V1N0 DIRECT decisions, 17/17 V1N1 TOOL decisions, 50.00% balanced accuracy, and 59/59 final correctness. ToolNeed produces 10/42, 17/17, 61.90%, and 59/59, respectively. Behavioral verification is triggered for ten V1N0 cases, all of which become DIRECT; the other 32 V1N0 cases and all 17 V1N1 cases follow routing bypasses. Each ToolNeed comparison against ToolNeed-Base, ToolNeed-Helper, and RA-PF contains 10 wins, zero losses, and 49 ties, with exact McNemar $p = .001953$. Paired bootstrap 95% confidence intervals for the balanced-accuracy difference are $[+5.95, +17.86]$ points against ToolNeed-Base and $[+5.95, +19.05]$ against RA-PF. The ten DIRECT interventions are gold-correct, giving 100% intervention precision.

I SABEVAL DIRECT-RESPONSE SEMANTIC AUDIT

We audit a frozen stratified sample of 40 non-empty SABEval responses converted from Raw TOOL to ToolNeed DIRECT. The sample contains 20 Qwen3.5 and 20 Qwen3.7 cases, with four cases from each D0–D4 level for each backbone. A blinded task-level semantic evaluator receives the user request and necessary conversation context together with the returned DIRECT response. The evaluation view excludes the Raw response, method identity, backbone, D-level, and selection condition. The evaluator uses DeepSeek-V4-Flash under a fixed task-level adequacy rubric: PASS denotes a response that adequately satisfies the user request in context without a material factual error. Concise answers and contextually appropriate clarification or inability responses qualify under the same criterion.

Table 20: SABEval direct-response semantic adequacy on the frozen 40-case stratified audit.

Group	Pass	Rate (%)
Overall	34/40	85.0
Qwen3.5	16/20	80.0
Qwen3.7	18/20	90.0
D0	6/8	75.0
D1	7/8	87.5
D2	6/8	75.0
D3	7/8	87.5
D4	8/8	100.0

Overall semantic adequacy is 34/40 (85.00%), with an exact Clopper–Pearson 95% confidence interval of $[70.16\%, 94.29\%]$. The six non-passing responses comprise three factual errors and three task-critical incomplete responses; the audit contains zero irrelevant responses, unjustified refusal or inability responses, and hallucinated responses.

J FAILURE ANALYSIS

The evidence bundle contains both a repaired and a retained case. In the repair case, the input "Whopper" is paired with a food-changing function whose enum contains `BURGER` but excludes "Whopper." The proposed call is structurally plausible, and comparison with matched no-tool behavior selects an ordinary explanation, illustrating a valid-looking TOOL-to-DIRECT intervention. The BFCL Irrelevance evaluator certifies the absence of an incorrect invocation of the provided tool;

full semantic assessment of the no-tool behavior lies outside this evaluator's scope. In the retained case, a request for music events in Fresno on a specified date produces an event-search proposal; behavioral comparison retains the tool action because the no-tool behavior is less useful.

These examples also characterize the current failure modes. Weak no-tool behavior can make an answerable request appear tool-dependent, and an aggressive DIRECT choice can damage tool-required tasks. On Qwen3.7, behavioral verification produces 38 paired gains and 6 losses relative to ToolNeed-Base; five losses appear across the reported Simple/Multiple category totals. Together, the two cases illustrate the behavioral comparison underlying both correction and retention decisions. Live BFCL identifies the main remaining errors in aggressive DIRECT choices, and Non-Live shows that the same mechanism can operate with near-exact tool-required preservation.